Automated Evolutionary Design of Self-Assembly and Self-Organising Systems

Germán Terrazas Angulo

Thesis submitted to The University of Nottingham

for the degree of Doctor of Philosophy

October 2008

Abstract

Self-assembly and self-organisation are natural construction processes where the spontaneous formation of aggregates emerges throughout the progressive interplay of local interactions among its constituents. Made upon cooperative self-reliant components, self-assembly and self-organising systems are seen as distributed, not necessarily synchronous, autopoietic mechanisms for the bottom-up fabrication of supra-structures. The systematic understanding of how nature endows these autonomous components with sufficient "intelligence" to combine themselves to form useful aggregates brings challenging questions to science, answers to which have many potential applications in matters of life and technological advances. It is for this reason that the investigation to be presented along this thesis focuses on the automated design of self-assembly and self-organizing systems by means of artificial evolution. Towards this goal, this dissertation embodies research on evolutionary algorithms applied to the parameters design of a computational model of self-organisation and the components design of a computational model of self-assembly. In addition, an analytical assessment combining correlation metrics and clustering, as well as the exploration of emergent patterns of cooperativity and the measurement of activity across evolution, is made. The results support the research hypothesis that an adaptive process such as artificial evolution is indeed a suitable strategy for the automated design of self-assembly and self-organising systems where local interactions, homogeneity and both stochastic and discrete models of execution play a crucial role in emergent complex structures.

Acknowledgements

I would like to give special thanks to my parents for their unconditional support and strength and for fostering my interest in science since the early years of my education. These special thanks are also extended to Natalio Krasnogor who has outstandingly shaped, motivated and fed my interest in scientific research since the early stages of my undergraduate studies.

I would also like to acknowledge everyone from the ASAP Group, in particular Jason Atkin, Cyril Schoreels, Lin Li, Camille Beyrouthy, Jaume Bacardit, Geert De Maere and Steven Gustafson, who have given me a warm welcome not only to the group but also to the social life in the City of Nottingham. Many thanks to Amalia Cifor for having that patience and courage to show me an alternative way to see life, and for her support and understanding throughout the last months of my thesis writing.

Thanks Mom for taking care of my beloved Lili during her last days of life.

- To my parents, Lourdes and Germán.

Contents

List of Figures

List of Tables

Glossary of Acronyms

CHAPTER 1

Introduction

This first chapter aims to present this thesis in four introductory sections covering the goals, methodological aspects, structure and publications. In particular, the first section summarises the purpose of the dissertation and the research area from which the topics of study originate. The second section gives a concrete description of the problems to be investigated together with the proposed computational strategies and assessment methodologies. The third section follows, setting out an organisational scheme for understanding how the chapters interrelate to one another, and finally, the fourth section closes this introduction with a list of publications produced by the research findings.

1.1 Dissertation Goals

The purpose of this thesis is to embark upon the automated design of self-assembly and self-organised systems by means of artificial evolution. Towards this goal, the research is centred on the application of genetic algorithms for the design optimisation of cellular automata parameters and self-assembly Wang tiles. This investigation is also focused on a protocol

of assessment comprising correlation metrics, clustering, generalised secondary structures and evolutionary activity. To begin with, this dissertation starts with the definitions of self-assembly and self-organization where common features and differences along with examples found in nature are enumerated. This is followed by the characterization of cellular automata and Wang tiles as computational models of self-organised and self-assembly systems. After that, the study, classification and exemplification of evolutionary algorithms in design and optimization of self-assembly and self-organised systems are given. The thesis follows with the presentation of an evolutionary approach for the design optimisation of cellular automata parameters and for the design optimisation of self-assembly Wang tiles. The results of the methodologies introduce a set of protocols for the analyses and characterization of the fitness functions and the evolutionary activity. Finally, the thesis concludes with an overview of the achieved results, contributions and future directions.

1.2   Methodological Aspects

The methodology employed in this research involves the use of a simple genetic algorithm to be applied to the automated design optimisation of cellular automata parameters and self-assembly Wang tiles which work as models of self-organised and self-assembly systems respectively. These two archetypal domains represent a continuous design optimisation problem with fixed length encoding (cellular automata) and a discrete design optimisation problem with variable length encoding (self-assembly Wang tiles). The genetic algorithms are equipped with three independent evaluation procedures: lattice scanner, universal similarity metric and morphological image analysis, all of them fully explained in Chapter 5,

Chapter 6 and Chapter 7. With this evolutionary methodology at hand, the research is centred on addressing the following questions:

Is it possible to make an evolutionary-driven specification of the laws (rules, parameter values) governing an observed cellular automata dynamics ?

Is it possible to make an evolutionary design of the set of tiles needed to obtain a particular supra-structure by means of self-assembly ?

In order to determine whether or not the proposed methods are effective, four protocols of analyses are employed: fitness distance correlation, clustering, the emergence of generalised secondary structures and evolutionary activity waves, all of which are discussed in Chapter 8 and Chapter 9. On the one hand, both fitness distance correlation and clustering are applied in order to assess the genotype-phenotype-fitness mapping. In particular, fitness distance correlation appraises the genotype-fitness relationship whereas the clustering studies the phenotype-fitness affinity. Hence, at this stage of the research, the interest is put on answering the following:

Is the genotype-fitness of an individual well correlated ? I.e. are the distances among genotypes and the distances among fitness values equivalent ?

Are the fitness functions properly distinguishing the phenotypes ?

On the other hand, the emergence of generalised secondary structures and evolutionary activity waves embody analyses to be done at the phenotypic level of the self-assembly Wang tiles design optimisation problem. Thus, the study of generalised secondary structures enumerates and quantifies which type of self-assembly patterns evolve, whereas evolutionary activity waves envisage an adaptive phenomena behind evolution. Both analyses shed light on how the underlying self-assembly mechanism develops throughout the evolutionary process, i.e. they address the questions:

Given an evolved self-assembly Wang tile system, how would a target shape emerge out of the interaction of tiles ?

Given a set of generalised secondary structures, is there any way to identify which are the most likely to participate in the self-assembly process ?

Given a genetic algorithm for the design optimisation of self-assembly Wang tiles, is there any way to measure how the glue strengths participate along the evolutionary process ?

1.3   Structure of the Dissertation

The structure of this thesis is organised into background, proposed approach, experiments and results, analyses and conclusions within ten chapters. The map depicted in Figure 1.1 shows an schematic relation of the chapters contained in this dissertation.

> Introduction
> CHAPTER 1
> PAGE 1

> Self-Organisation
> CHAPTER 2
> PAGE 10

> Self-Assembly
> CHAPTER 2
> PAGE 14

> Models
> CHAPTER 2
> PAGE 20

> Evolutionary Design
> CHAPTER 3
> PAGE 38

> Proposed Methodology
> CHAPTER 4
> PAGE 55

> An Evolutionary Approach to Cellular Automata Parameter Design (Self-Organisation)
> CHAPTER 5
> PAGE 72

> An Evolutionary Approach to Cellular Automata Rules Design (Self-Organisation)
> CHAPTER 6
> PAGE 88

> An Evolutionary Approach to Self-Assembly Wang Tiles Design (Self-Assembly)
> CHAPTER 7
> PAGE 107

> Genotype-Phenotype-Fitness Analysis
> CHAPTER 8
> PAGE 156

> Self-Assembly Dynamics
> CHAPTER 9
> PAGE 207

> Conclusions
> CHAPTER 10
> PAGE 231

FIGURE 1.1: *Schematic structure of the dissertation starting with the notions of self-organization, self-assembly and proposed models. After that, evolutionary algorithms are presented as an approach for the automated design of self-assembly and self-organised systems. Then follows the experiments and results for the automated tailoring of cellular automata and self-assembly Wang tiles by means of artificial evolution. Finally, a set of protocols for the analysis and assessment of the applied methodology is presented.*

In what follows, Chapter 2 deploys the background needed to understand the research area and proposes cellular automata and Wang tiles as computational models of self-organization and self-assembly systems. Chapter 3 focuses on evolutionary algorithms in design and optimisation as a methodology that could address the automated tailoring of the presented models. Chapter 4 characterises the problems to be addressed and overviews the operational aspects of the proposed methodology. Later on, Chapter 5, Chapter 6 and

Chapter 7 detail the experiments performed and the achieved results by the proposed approach. After that, Chapter 8 and Chapter 9 present the protocols of analyses employed to assess the proposed methodology. Finally, Chapter 10 concludes the dissertation, summarizing the contributions and establishing future research directions.

1.4 Publications

The research to be presented in this thesis resulted in the following publications which have been submitted and accepted in international conferences and journals:

[1] L. Li, P. Siepmann, J. Smaldon, **G. Terrazas.** and N. Krasnogor. Automated Self-Assembling Programming. In N. Krasnogor, S. Gustafson, D. Pelta and J. L., editors, *Systems Self-Assembly: Multidisciplinary Snapshots*, pages 281–307. Elsevier, 2008. The contribution of this book chapter is shown in Chapter 9.

[2] **G. Terrazas.**, M. Gheorghe, G. Kendall and N. Krasnogor. Evolving Tiles for Automated Self-Assembly Design. In *IEEE Congress on Evolutionary Computation*, pages 2001–2008. IEEE Press, 2007. (**Best Paper award and Best Student Paper award**). The contribution of this paper is shown in Chapter 7.

[3] **G. Terrazas.**, P. Siepmann, G. Kendall and N. Krasnogor. An Evolutionary Methodology for the Automated Design of Cellular Automaton-based Complex Systems. *Journal of Cellular Automata*, 2(1):77–102, 2007. This journal paper contributes to Chapter 6.

[4] P. Siepmann, **G. Terrazas.** and N. Krasnogor. Evolutionary Design for the Behaviour of Cellular Automaton-Based Complex Systems. In *7th International Conference of Adaptive Computing in Design and Manufacture*, pages 199–208. IP-CC, 2006. The contribution of this paper is shown in Chapter 5.

[5] N. Krasnogor, **G. Terrazas.**, D. Pelta and G. Ochoa. A Critical View of the Evolutionary Design of Self-Assembly System. In *7th International Conference on Artificial Evolution*, volume 3871, pages 179–188. Lecture Notes in Computer Science, Springer, 2005. The contribution of this paper is shown in Chapter 7.

[6] **G. Terrazas.**, N. Krasnogor, G. Kendall and M. Gheorghe. Automated Tile Design for Self-Assembly Conformations. In *IEEE Congress on Evolutionary Computation*, volume 2, pages 181–222. IEEE Press, 2005. The contribution of this paper is shown in Chapter 7.

[7] **G. Terrazas.**, N. Krasnogor, M. Georghe, F. Bernardini, S. Diggle and M. Camara. An Environment Aware P-Systems model of Quorum Sensing. In *First Conference on Computability in Europe*, volume 3526, pages 479–485. Lecture Notes in Computer Science, Springer-Verlag, 2005. The contribution of this paper is shown in Chapter 10.

[8] F. Bernardini, M. Gheorghe, N. Krasnogor and **G. Terrazas.**. Membrane Computing - current results and future problems. In *First Conference on Computability in Europe*, volume 3526, pages 49–53. Lecture Notes in Computer Science, Springer-Verlag, 2005. The

contribution of this paper is shown in Chapter 10.

[9] N. Krasnogor, M. Gheorghe, **G. Terrazas.**, S. Diggle, P. Williams and M. Camara. An Appealing Computational Mechanism Drawn from Bacterial Quorum Sensing. *Bulletin of the European Association for Theoretical Computer Science*, (85):135–148, 2005. The contribution of this paper is shown in Chapter 10.

1.5 Icons Used in this Dissertation

Here is a list of icons and meanings to be employed along this dissertation with the purpose to assist and facilitate the reading.

DEFINITION – introduces a description, explanation or clarification of a new concept.

CONCEPTUAL QUESTION – establishes a conjecture that originates an important query in the line of the research.

IMPORTANT FACT – highlights an important analysis, a remarkable conclusion or an essential statement of the research.

TECHNICAL DETAIL – gives key description of an algorithm, data set or parameters to be employed in an experiment.

REMINDER – alerts the reader with a concept, hypothesis, analysis or conclusion that should be kept in mind for further reading.

CHAPTER 2

Self-Organisation and Self-Assembly

The aim of this chapter is to establish the foundations needed to understand the research area of this dissertation. The scope is mainly divided in two sections: introduction and models. The former introduces self-organisation and self-assembly systems enumerating examples observed in nature, highlighting their features and characteristics as well as noting the differences between both concepts. After that, the models section describes some of the existing approaches for modelling self-organisation and self-assembly proposing cellular automata and self-assembly Wang tiles as computational models for the two.

2.1 Introduction

Self-organisation and self-assembly are closely related concepts. Although these terms share many similarities and are sometimes confused, there are, indeed, important differences. For this reason, it is the purpose of this section to introduce both concepts and to contrast those points where they differ.

2.1.1   Self-Organisation

In physical and biological systems, *self-organisation* refers to pattern-formation processes via interactions among similar components or events internal to a given system without external influences. In this way, organisation emerges at the global level as a result of local information exchanged among the constituents of the system. In other words, the achieved patterns or self-organised structures are a collection of entities arranged due to cascades of iterative interactions across space and time where complexity unfolds progressively.

The main differences among physical and biological self-organising systems lie in the *complexity* of their components and their *governing laws*. For instance, entities of biological systems like cells, fishes or ants are far more complex than physical elements such as crystals. At the physical or chemical level, only physical laws govern the system whilst in biological systems genetically controlled components (i.e. information processing) are also influencing the behavioural interactions among the living entities.

The way in which components interact is crucial to understand how self-organisation works in biological systems. Two important mechanisms of interaction are *positive feedback* and *negative feedback*. The former takes an initial change in a system and reinforces it in the same direction as the initial deviation. In contrast, negative feedback reinforces changes in the opposite direction thus breaking and shaping the process, and hence inhibiting any positive feedback process that may be at work. Self-organisation in biological systems also arises from multiple interactions among individuals transferring information via *signals* or *cues*. In some self-organising groups of living entities, the individuals usually contribute with simple behavioural rules, known as *rules of thumb* [20], based upon the information

collected from its neighbours. For example, a fish belonging to a fish school neither has feedback regarding the direction of the school nor has all the trajectories of its neighbours. However, it has neighbouring information that keeps it moving in the same direction trying to stay close to the group and to avoid collisions with its neighbours (see Figure 2.1 (a)). This information gathered from nearby members of the group is not the only source of knowledge used by an organism belonging to a self-organising system. In those systems where individuals contribute to a collective effort, the common (shared) medium could also play an important role defining a kind of stimulus-response interaction called *stigmergy*. For instance, the stimuli provided by the emerging structure when termites build a nest can induce further activity such as performing additional labour or stopping building.

Self-organisation is an exciting research arena spanning physics, chemistry, biology and computer science. Not only is the organised behaviour in flocks, fish schools, fireflies synchronization and ants foraging (depicted in Figure 2.1 (a-d)) an example of naturally occurring self-organising systems, but also the patterns observed in the visual cortex of the macaque monkey, skin pigmentation of fish, pigmentation on shells of molluscs (Figure 2.2 (a-b)) and the zebra coat patterns qualify as examples of self-organising systems. On the other hand, technology and computer science have experienced an explosion of self-organising artefacts and technologies, e.g. the peer-to-peer systems [21], data bases [22] [23], automated self-assembly programming [24] and patterns obtained from the application of cellular automata rules [25][26][27].

(a)

(b)

(c)

(d)

FIGURE 2.1: *One of the most awesome instances of self-organisation in nature is the sight of thousands of living entities orchestrating together as a coordinate unit. Some eye-catching examples are schools of fish (a), fireflies (b), flocks (c) and ant trails (d). In schools of fish and flocks, each individual of the group bases its behaviour on monitoring the velocity and position of the nearest neighbours. The self-organisation among fireflies is observed by the synchronized flashing of their abdominal lantern. The interaction among ants is signalled via the chemical trails known as pheromones which act as positive (negative) feedback as they are enriched (evaporated). Extracted from [1] [2].*
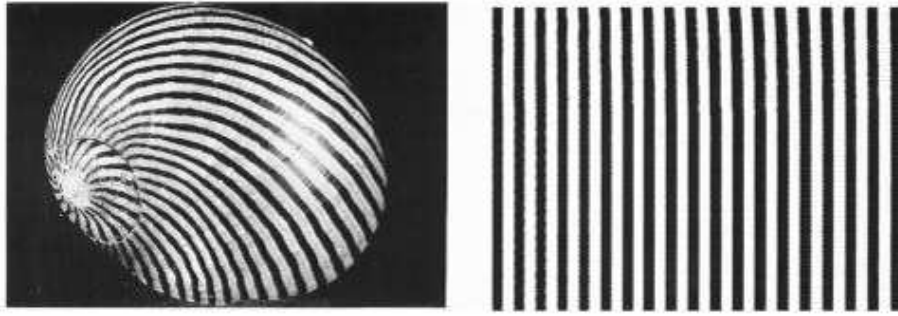
(a)
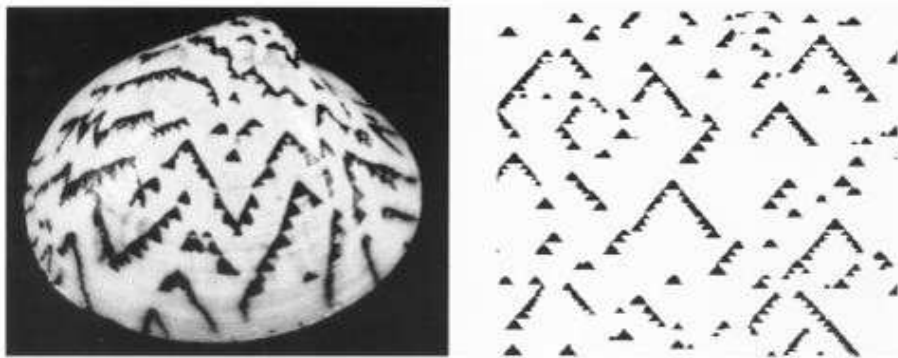
(b)

FIGURE 2.2: *"Pattern" refers to a particular self-organised collection of objects across space and time. The pigmentation of the shells of molluscs observed in Neritina ziczac (a) and Lioconcha castrensis (b) is a beautiful natural example. Early works suggest that these patterns develop from simple rules continually iterating among the components of the system as the cellular automaton snapshots show at the right side of the figure. Extracted from [3].*

2.1.2 Self-Assembly

Self-assembly is a distributed process that creates structures from scratch through the statistical exploration of components' aggregates without external intervention. In general, self-assembly components are autonomous, have no pre-programmed master assembly plan, and interact with their local environment and other components. Self-assembly is a mechanism whose power, as a reusable engineering concept, lies in the fact that it is a distributed, not necessarily synchronous, control mechanism for the bottom-up manufacture of complex structures. This control mechanism is distributed 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. Instead, each component has a very limited behavioural repertoire which tells it what to do under a reduced set of well defined conditions.

From nano-particles to planets, self-assembly is a phenomenon studied in terms of physics, chemistry, computer science and biology. The observed prominent features of self-assembly systems in nature at different scales are *robustness*, *versatility*, *reversibility* and *mass transportation*. Robustness comes from the fact that self-assembly systems comprise a large number of parts that can replace one another in case of failure. A robust system can withstand a variety of errors, perturbations, or even partial destructions on its functionality getting back to its initial state by means of adaptation to the environmental changes and generating order out of chaos [28]. Versatility, on the other hand, is given by the possibilities of re-configuring the way in which components relate to each other subject by, for example, attraction and repulsion of inherent forces. In some self-assembled structures like the poly-

mer brushers in [29] or the metastable crystals in [30], the association between components

is expected to be reversible in the sense that the strength of the attracting forces must be

comparable to those of repelling order. For example, the collision between molecules must

be reversible for crystallization to occur, otherwise their resulting arrangement will end up

in glass. Notice that there are also other system-dependent features that can be described

at this stage but it is not the intend to consider these within the scope of this dissertation.

For instance, in chemical self-assembly systems where the components are embedded in

a solution, assuring mobility of the constituents becomes a crucial feature as gravity and

friction with the environment have a strong influence on their Brownian motion.
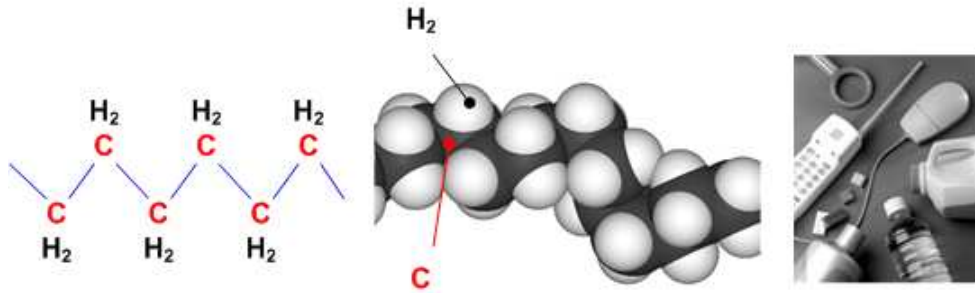
FIGURE 2.3: *Polymerization is an example of a self-assembly process taking place at microscopic level from where materials like polyethylene is manufactured. Polyethylene comprises a cross linked chain of ethylene monomers (H_2C) double bounded at the carbon (C) molecule. This resultant polymer is a thermoplastic commodity heavily used in consumer products such as telephones, plastic containers and toys. Extracted from [4].*
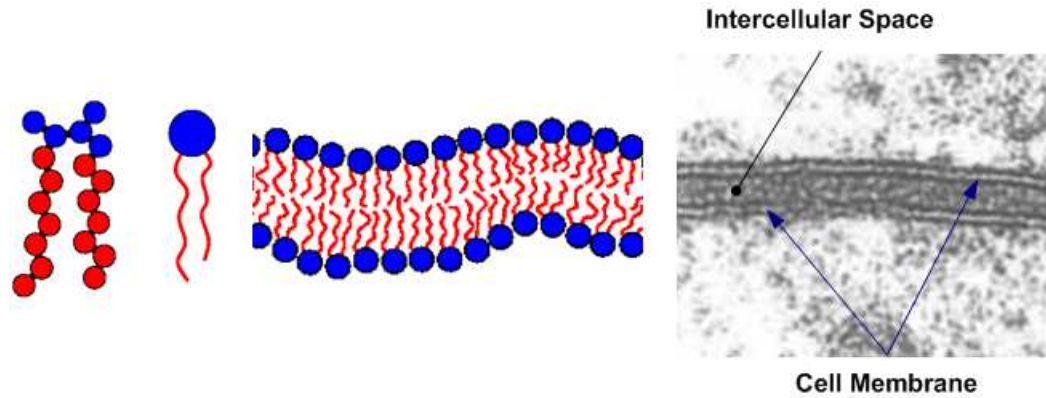
FIGURE 2.4: *Lipids are a special kind of surfactant which consist of a hydrophilic head and a hydrophobic double tail. In an aqueous system, the heads orientate towards the environment while the tails minimise their contact with water making lipids self-assemble in a stable two-dimensional sheet. These lipid bilayers have many functions in living organisms including structural components of cell membranes which separate compartments inside the cell to protect the important processes and events vital in life. Extracted from [5] [6].*

FIGURE 2.5: *Snow forms as water condenses into a tiny droplet which grows as more and more water vapour condenses onto its surface. After that, the cold air freezes this droplet into ice crystals of unique shape which eventually fall from the sky. Whilst falling, they come into contact with warmer air that makes them melt as they descend. This melting acts like a glue causing crystals to self-assemble into larger structures called snowflakes. Extracted from [7].*

Self-assembly systems are ubiquitous in nature and can be found at all scales. At the microscopic level, lipid membranes (Figure 2.4), nucleic acid structures and other complex cell entities are well-known biological examples of self-assembly. Molecular crystals such as the crystalline form of table sugar and rock candy or colloids such as milk, butter, asphalt or snow formation (Figure 2.5) are everyday cases of self-assembly where molecules are kept together by interactive forces. Other striking examples of self-assembly are the soap-like products making shampoos and other cosmetics. These materials are composed by a class of molecules called amphiphiles which is a combination of hydrophobic and hydrophilic molecules that, when mixed with water, take up complex and ordered structures.

2.1.3 Differences between Self-Organisation and Self-Assembly

Many attempts have been made to try to classify and separate self-organisation and self-assembly. For example, a study of animal societies [31], defines self-organisation as a phenomenon taking place at the organismal level that is characterized by the combination of *positive feedback*, *negative feedback*, *stochasticity and randomness*, and *multiple local interactions*. At this "organismal" level, self-assembly is outlined as a mechanism for obtaining individual-based structures, e.g. army ant bridges (Figure 2.6 (a)) or bee curtains (Figure 2.6 (b)). Under this definition, self-assembly and stigmergy (defined in Section 2.1.1) seem to have many features in common such as the movement of individuals over a structure. However, one of the remarkable differences among them is that in self-assembly only animate active entities take part in the pattern-formation process whilst in stigmergy inanimate entities are also involved as part of the stimulus-response interaction. Moreover, it is argued that self-organisation and self-assembly also differ on whether positive feedback is present

or not. In studies related to assemblages in living systems, this interaction mechanism is considered as part of self-assembly [31] [28]. For instance, in Eciton ants societies, the formation of chains of ants (Figure 2.6 (a)) attracts others to continue the chain. In other words, positive feedback operates through the already built structure, attracting more ants to get located at the end of the aggregation, i.e. at the lowest part of the structure.

(a)   (b)
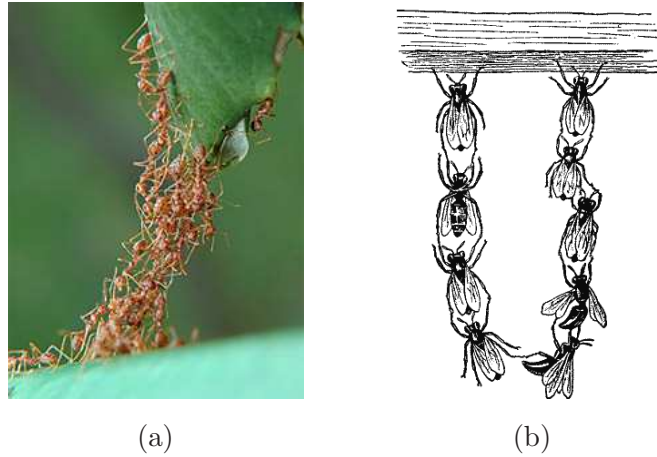
FIGURE 2.6: *Self-assembly at organismal level as a mechanism for obtaining individual-based structures like army ant bridges (a) or bee curtains (b). Extracted from [8] [9].*

Following the biological line in [32], the difference between self-organisation and self-assembly is stated in terms of the *information* stored in the cooperative entities. For instance, in self-assembly each component carries an explicit encoding describing the type of component with which to interact. On the other hand, in self-organisation, these components follow more general rules without any type of encoding to get the final pattern.

A physical approach differentiates self-assembly from self-organisation in terms of energy [33]. In the first case, the formation of aggregates is related with the minimisation of energy in a closed system. Self-assembly involves processes of structure formation which occur via a relaxation into a thermodynamic equilibrium state. In contrast, self-organisation

is said to be far from any thermodynamical/chemical equilibrium nor restricted to enclosed environments and it is generally thought to be a kinetically governed process. This is also called conservative self-organisation or arrested self-assembly [34]. The following table summarises the main features and differences between self-assembly and self-organisation previously discussed.

	Self-Organisation	**Self-Assembly**
Type of Interaction	With the local environment throughout positive/negative feedback, and with other components.	With the local environment throughout positive feedback and with other components.
Type of Systems	Operative in open systems, i.e. inputs/outputs of energy and matter.	Restricted to closed systems, i.e. absence of inputs/outputs of energy or matter.
Type of Components	Autonomous components with general rules of interaction.	Autonomous components with specific rules of interaction.
Type of Structures	Complex organised structures and coordinated behaviour.	Complex aggregates.
Thermodynamic State	The organisation among entities emerges far-from-equilibrium.	The construction of assemblies emerge by means of close-to-equilibrium dissipative processes which minimise the energy of the system.

TABLE 2.1: *Features and differences between self-organisation and self-assembly according to the type of interaction, systems , components, formed structures and thermodynamics of the system.*

2.2   Models

As seen in the previous section, self-organisation and self-assembly are complex systems in nature mainly characterized by organic or living entities belonging to a particular environment and achieving global order as the result of local interactions. For instance, as a concrete example of modelling natural systems, five different approaches for studying the icosahedral virus capsid assembly are reviewed in [35]. In an effort to understand the self-assembling of proteins which are to form the shell that encase the viral genome (Figure 2.7), five models have been proposed: kinetic, molecular dynamics, reaction landscape, differential equations and a Monte Carlo simulation.

(a)   (b)
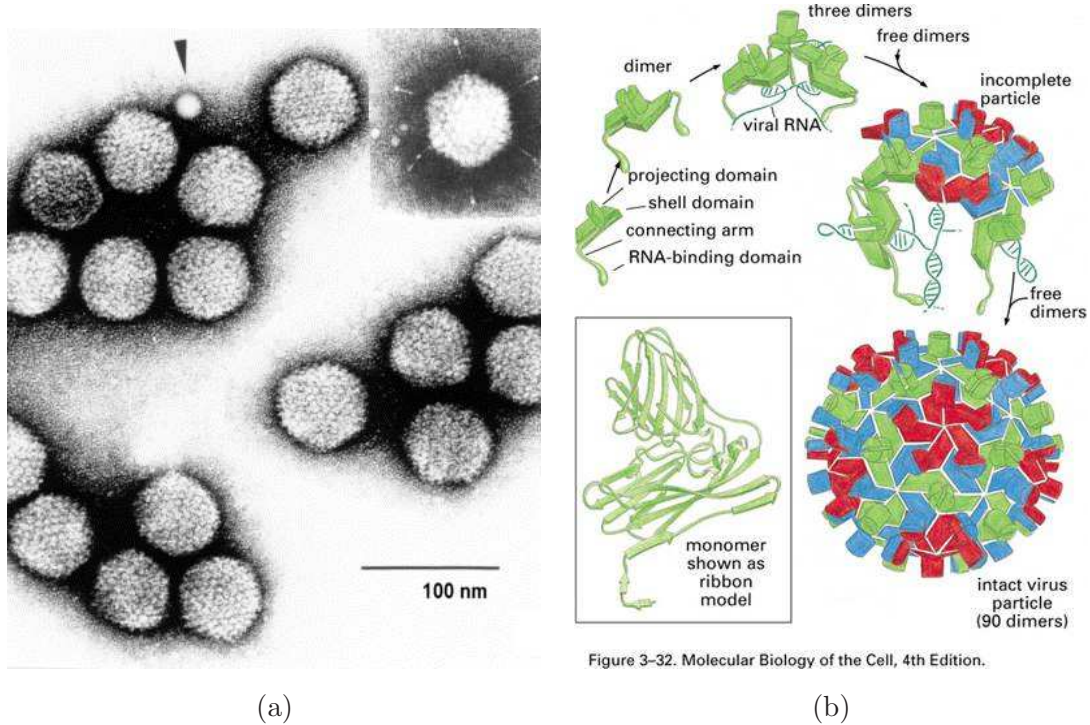
FIGURE 2.7: *Electron micrograph of Human adenovirus, a vertebrate with icosahedral capsid symmetry with a diameter of 80-110nm (a). Pathway of self-assembling dimers forming the virus capsid to be deposited inside the host cell (b). Extracted from [10][11].*

The first idea attempts to model nucleation events, contact free energy and elongation rate, although it is argued that some molecular details cannot be captured. Inspired by local rules theory, the molecular dynamics model has been used in order to study the oligomer-oligomer bindings during the assembly process [36]. This approach results in the creation of a set of local rules capable of explaining the capsid formation, although they are not universal and they introduce more restrictions than is necessary for a correct assemblage. The generation of a reaction landscape in order to understand the assembly pathway has been done in terms of the enumeration of the possible intermediate structures and its corresponding reaction paths [37]. According to the overall reported results, this model has been found useful only to analyse the beginning of the assembly process. Since the three reported models fail to explore the role of malformations in the self-assembly process, a model based on differential equations capable of tracking all possible single monomer addition reactions is proposed in terms of concentrations of the population of various oligomers. Although this approach yields reasonably good results, it was observed that the number of differential equations grows rapidly for large structures generating a difficult and slow task to perform. Even though this algebraic representation could work well when the population is large, differential equations are difficult to formulate as they are meant for continuous processes and they are hard to understand due to the abstraction level. In contrast to differential equations, simulations like Monte Carlo use a different approach to modelling the behaviour of each individual and incorporating certain features like randomness observed in biological systems. In this case, a Monte Carlo simulator has been set up with discrete protein subunits in order to monitor the addition reactions of every single particle. The

Monte Carlo simulation starts with a given number of monomer unbound in 2D, a run time, volume and forward/reverse rate. The simulation starts with choosing a random molecule (a monomer at the first step) and a monomer. After that, an acceptance rule is evaluated, deciding whether a forward reaction between the chosen objects is attempted or not. In case of success, a random number is generated in order to establish the type of angle within which these monomers have been bonded. After that, a reverse reaction will be attempted, choosing a new random molecule and a reverse reaction acceptance rule capable of generating an open structure if the closing bond resulting from the acceptance rule breaks. Finally, the simulation advances one step in time and repeats the process.

As the example shows, different perspectives of the same problem highlighting some key features and ignoring others are achieved by alternative modelling approaches although none of them can completely capture the complete system being modelled. Since the purpose of this dissertation is to embark into the automated design of self-organising and self-assembly systems by means of artificial evolution, a model capable of representing the elements of interest of these systems, states and interactions is needed. Given that the primary goal of any representation is to capture the essence of the underlying phenomena without loss of generalization, this research considers that local interactions, statistical exploration of configurations among entities, and emergent global order are some of the important features that a model for self-organisation and self-assembly should be able to capture. For this reason, an executable deterministic model like cellular automata and a multi-agent system like Wang tiles are chosen as experimental target models of complex phenomena for self-organisation and self-assembly systems respectively. One of the advan-

tages of agent-based models is that they are applicable in cases where a small number of entities govern the structure formation, an aspect which differential equations are not sufficient to describe [38]. Multi-agent models also provide a flexible versatile tool to describe self-assembly in complex systems reflecting the origin of new qualities and the unfolding of complexity [39]. In the following two sections, each of these models is introduced.

2.2.1 Cellular Automata

A Cellular Automaton (CA) is a massively parallel, homogeneous computational device that operates based on local interactions. By choosing the proper local rules, lattice and neighbourhood, CAs have been employed as models in many applications such as addressing pattern formation of multicellular spheroids [40], screening of cultures of fibroblasts [41] [42], protein folding prediction problem [43], emergent pattern interpretation in forest ecosystem dynamics [44], the study of pattern formation in reaction-diffusion systems [45], behaviour analysis in hydrodynamical systems in the field of discrete fluids [46], the quantification of spatial and temporal correlations occurring in biological processes of plants [47], land use change modelling [48] or traffic phenomena such as jams in [49], to name but a few. Nevertheless, self-organisation has also been studied by means of CAs [50] [51] from where the emergent structures are widely applied to different branches of science such as the simulation of processes taking place in physics, biology, social sciences or chemistry.

Conversely, many CA patterns can be observed in natural systems but the underlying CA rule is unknown. The understanding of how self-organisation works is the key problem the observer faces which, translated in CA terms, stands for identifying the sys-

tem and its rules capable of predicting the observed output. Essentially, this is an inverse problem where the observer knows the effects but ignores the causes. The purpose of this research is to study CAs as models of self-organisation in order to extract the parameter values and rules from captured spatio-temporal behaviour. This retrospective search is very difficult to deal with since the solving methodologies tend to be introspective [52], they are meant to divide the problem [53] [54] or they are automatic only for particular cases [55] [56].

A CA is defined as an infinite, regular grid of cells, each of which can be in one of a finite number of states. At a given time step t_i, the state of a cell is updated in agreement with a transition function that applies a CA rule to the cell in question. In general, a CA rule is defined in terms of the states of a cell's neighbourhood at t_{i-1}. According to the type of neighbourhood, CAs are classified into one-dimensional CA, two-dimensional CA or three-dimensional CA. In the first case, the neighbourhood comprises the adjacent cells, i.e. the ones located at the left and right hand side of the cell of interest. The simplest CA is a binary, nearest-neighbour, one-dimensional automaton. Such automata are called "elementary cellular automata" by S. Wolfram, who has extensively studied their various properties [57]. There exists 256 such automata, each indexed by a unique binary number whose decimal representation is known as the "elementary rule" for the particular automaton. Each rule is encoded with a binary array of length 8 where each part is associated with a possible configuration given by the state of the cell containing the rule and the state of its two immediate surrounding cells. During runtime, the state of the first row of cells is randomly initialized with either 0 or 1. After that, at every time step t_i, a cell

c_i executes its rule considering its internal state plus the state of its two adjacent cells c_{i-1} and c_{i+1}. This determines the state of c_i at time step t_{i+1} according to the related value to that configuration. An illustrative example considering a one-dimensional automaton associated to the elementary rule 121 is shown in the following figure where the rule is defined in Figure 2.8 (a). Its application to a single line of cells at time step t_i is depicted in Figure 2.8 (b) that gives as a result the emergent spatio-temporal pattern like the one captured in Figure 2.8 (c) after the the iterative applications of the rule across the time.
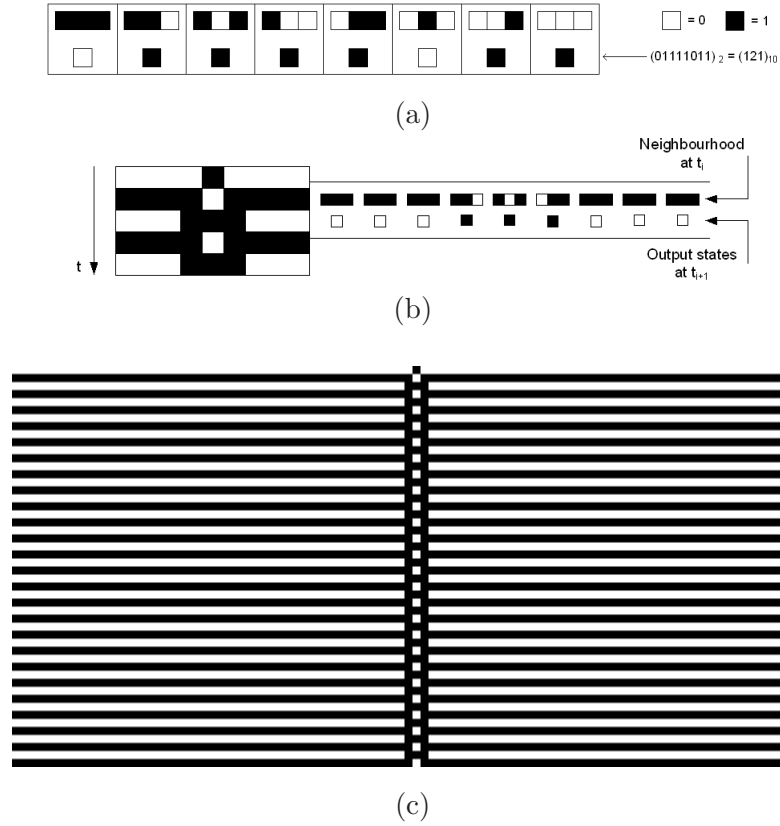
FIGURE 2.8: *Encoding of the elementary rule 121 (a). A CA executing the rule 121 where each possible neighbourhood configuration is associated with an output state used as the new state for the next time step (b). An emergent spatio-temporal pattern (c).*

N1	N1	N2	N2	N3
N1	N1	N2	N2	N3
N4	N4	N5	N5	N6
N4	N4	N5	N5	N6
N7	N7	N8	N8	N9

(a)   (b)   (c)
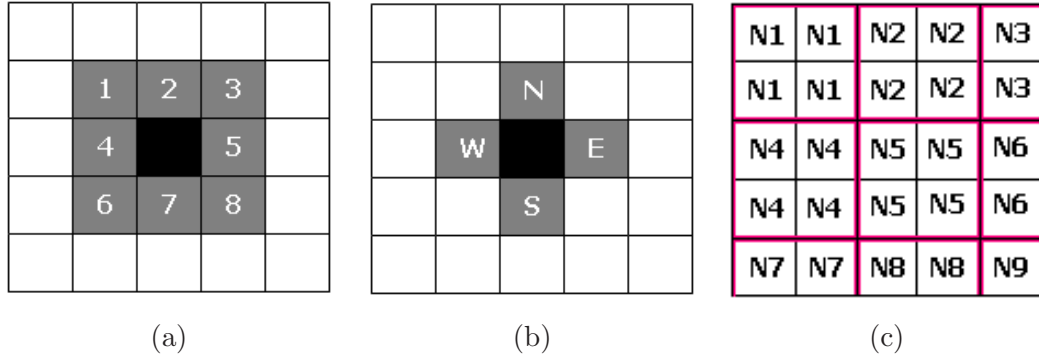
FIGURE 2.9: *Illustrations of Moore vicinity defining its eight neighbouring cells (a), von Newman neighbourhood with the corresponding north, south, east and west cells (b), and Margolus neighbourhood dividing the grid in neighbourhoods (Ni) of four cells (c).*

In case of two-dimensional CA, the neighbourhood of a cell is defined in terms of the surrounding cells. For instance, the Moore neighbourhood depicted in Figure 2.9 (a) uses the eight surrounding cells of the cell in question for the update process, i.e. using these eight states as input to the update function. Similarly, the von Neumann neighbourhood shown in Figure 2.9 (b) only uses the four cells – defined as north, south, east and west – that are strictly adjacent to the central cell. In contrast, the Margolus neighbourhood from Figure 2.9 (c) divides the grid into groups of four cells, to which the update function is locally applied (i.e. using only the information in this group of four cells). To allow propagation through the grid, the grouping of the cells changes on each update. There are also some extended models such as the Extended Moore where the distance of the neighbourhood is bigger than a radius of one.

From the self-organisation point of view, pattern formation is observed in the widely known Sierpinski Triangle. This fractal has been named after the mathematician

Waclaw Sierpinski who described it in [58]. Based on a recursive process of replication, shrinking and relocation of an equilateral triangle, a close approximation of this gasket can be implemented by a one-dimensional CA using the elementary rule 90 defined in [57] and shown in Figure 2.10 (a). After several iterations, the emergent pattern resembles a collection of recurrent triangles organised one on the top of the other as depicted in Figure 2.10 (c). The Sierpinski Triangle emerges in nature as in the pigmentation of the Lioconcha castrensis previously shown in Figure 2.2 (b). Multiple engineering applications using this gasket in order to achieve multi-band frequency operation are found in [59], [60] and [61].
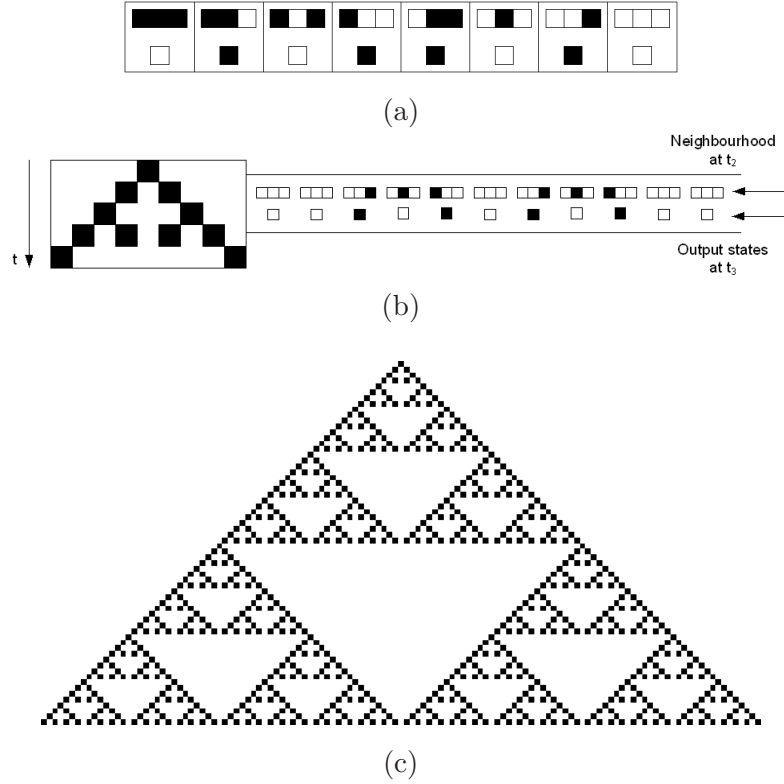
FIGURE 2.10: *Encoding of the elementary rule 90 (a). A CA executing the rule 90 where each possible configuration of three cells is associated with an output state used as the new state for the next time step (b). The Sierpinski Triangle in a lattice of* 300×300 *cells (c).*

2.2.2   Wang Tiles

Considered as a set of equal-sized components, square in shape and coloured at their edges, Hao Wang defined in [62] a formal system called *Wang tiles*. Having an infinite amount of tile copies arranged one by one from left to right and from top to bottom with matching colours at their touching edges, it has been shown in [63][64] that there is no algorithm for determining whether a set of tiles can tile a plane periodically or aperiodically, fact that corresponds to the halting problem and establishes Wang tiles as Turing Universal [65] [66]. The idea of matching colours between tile edges was captured as *binding rules* [67], extending Wang's system with an effective mechanism to embody self-assembly components. As a result, the building blocks of universal computation in self-assembly were set up and an enriched model of Wang tiles with striking theoretical properties and applications emerged.

One of the most prominent examples underlying the idea of self-assembly Wang tiles is the *DNA tiles* model [67]. Based on the physiochemical properties that allow DNA strands to assemble, tiles are manufactured using four strings of DNA with "sticky ends" at both the right and left edges that associate according to Watson-Crick complementarity. A bulk of tiles is left in solution undergoing self-assembly subject to the control of binding forces and thermal energy. This bottom-up wet lab experiment ends up in the formation of two-dimensional crystalline self-assembled nano structures proving that the engineering of local interactions are particularly amenable to the design and synthesis of complex objects. This abstract DNA assembly framework made of structured particles, environment, cohesive forces and driving forces sets up the basis for DNA cubes [68], DNA bar codes [69], DNA-based nanoarrays [70], and DNA origami [71].

Inspired by the physical process of crystallization, a self-assembly model using Wang tiles is defined in [12] [72]. This model is called *Tile Assembly Model*. It comprises a set of square tiles with labelled edges associated to strength values. Formally speaking, a tile system is defined as a quadruple $(\mathcal{T}, t_s, g, \tau)$ where $\mathcal{T}$ is a finite set of non-empty tile types, t_s is a seed tile belonging to $\mathcal{T}$, g is a strength function and τ is a threshold parameter. Elements of $\mathcal{T}$ are tile types t defined as a 4-tuple $(\sigma_N, \sigma_E, \sigma_S, \sigma_W)$ indicating the associated labels at the north, east, south and west edges of the tile, which are assumed to be square. This definition works together with a lattice of unit square locations, similar to CAs, where the functions N, E, W, S are directions used to indicate relative positions in the grid. Thus, tiles are defined as (t, pos) where t is a tile type and pos is its position (i, j) in the lattice. Considering an unlimited supply of tiles, aggregates are formed by placing one tile next to another whenever they have matching labels at their touching edges and the summed strength given by g is greater than τ.
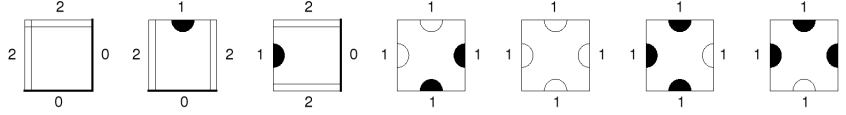
FIGURE 2.11: *An example set of the Tile Assembly Model where the bold line, the half full circle, the half empty circle and the double line define four types of edge labels associated to strengths 0, 1, 1, and 2 respectively. Extracted from [12].*

In order to illustrate how this model works, consider the seven tiles depicted in Figure 2.11. In this picture, the bold line, the half full circle, the half empty circle and the double line define four types of edge labels whereas the strength associated to pairs of matching labels is given by 0, 1, 1, and 2. Considering t_s as the tile with strength-2 edges and the threshold parameter τ set to 2, there is a unique possibility of assembly that

satisfies both matching labels at their touching edges and summed strengths greater than τ. A step by step example process of six tiles self-assembling is shown in Figure 2.12 (a-e). The continuation of this self-assembly strategy, reveals the Sierpinski Triangle pattern embedded in the aggregate after several steps (see Figure 2.13).

(a)   (b)   (c)   (d)   (e)
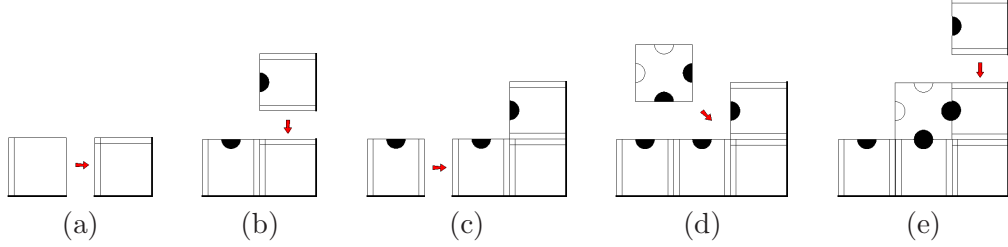
FIGURE 2.12: *Step-by-step self-assembly process of six tiles. Two tiles self-assemble at the left and right hand sides of the seed tile (a - b). In a similar way, another tile self-assembles to the left side of the aggregation (c). By matching dark semi-circles, a strength-1 edges tile self-assembles to the middle of the aggregate (d). Following this process, another tile self-assembles at the right hand side of the aggregate (e)*
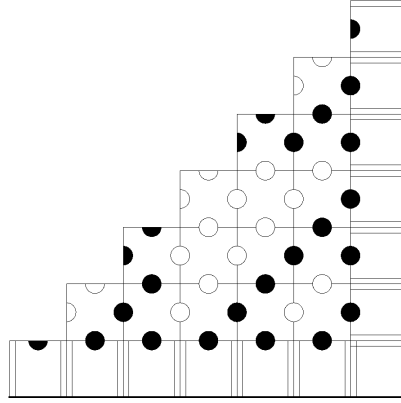
FIGURE 2.13: *The tiles self-assembling strategy reveals the Sierpinski Triangle pattern embedded in the aggregate as several computational steps take place under $\tau = 2$. Extracted from [12].*

In contrast to the dynamics defined above, the defined self-assembly Wang tiles model allows tiles to perform a random walk across the lattice. In this case, if two tiles

collide, they either self-assemble, i.e. they link, or they remain separated. In addition, the decision of whether they assemble or not depends only on whether the glue types at their colliding edges are compatible or not. This compatibility is computed by an interaction function which evaluates the strength between the two labels or glue types of the colliding edges against the specific kinetic energy associated to the tile set. A formal definition of a tile system T_{sys} is given in Equation 2.1 where $\mathcal{T}$ is a finite set of non-empty Wang tiles, Σ is a set of symbols, g is called the *glue function*, $\mathcal{L}$ is a lattice and τ is a threshold.

$$T_{sys} = (\mathcal{T}, \Sigma, g, \mathcal{L}, \tau)$$

$$\mathcal{T} = \{t | t = (c_0, c_1, c_2, c_3)\} \text{ where } c_0, c_1, c_2, c_3 \in \Sigma$$

$$g :: \Sigma^2 \rightarrow \mathcal{Z}^+ \cup \{0\}$$

$$\tau \in \mathbb{Z}^+ \cup \{0\} \tag{2.1}$$

The elements defined by Σ are glue types labelling the edges associated to a tile. This set also includes the special symbol λ representing an edge with no glue type. The glue function g computes the strength associated to a pair of glue types at the colliding edges of two adjacent tiles t_i and t_j located in $\mathcal{L}$. Note that g is a symmetric and surjective function, that is $\forall\ (c_i, c_j) \in \Sigma^2$, $g(c_i, c_j) = g(c_j, c_i)$ and $\forall\ (c_i, c_j) \in \Sigma^2$ there is at least one $z \in \mathcal{Z}^+ \cup \{0\}$ such that $g(c_i, c_j) = z$ respectively. In the particular case where $c_i = \lambda$ or $c_j = \lambda$ then $g(c_i, c_j) = 0$. The lattice $\mathcal{L}$ is a two-dimensional surface with size $W \times H$ composed by a finite set of interconnected unit squared cells where tiles belonging to $\mathcal{T}$ are located. Each cell can be occupied by one tile at any time. Each cell defines a position into

the lattice with an associated coordinate value $(x, y) \in \mathbb{N} \times \mathbb{N}$ where $x \leq W$ and $y \leq H$. The threshold τ models the temperature associated to the tile system and it takes positive values. The role of this parameter is to define the kinetic energy of the system. That is, the higher τ is, the more likely a tile will bounce off other tiles due to the energy associated with its motion. In order to show how this model operates, consider the set of three tiles depicted in Figure 2.14 (a) where the colours at their edges represent different glue types and the strength among them is encoded by the matrix of Figure 2.14 (b). Thus, the interaction function is computed in terms of the cell value associated to a given pair of colours and the specific kinetic energy τ.

	C_1	C_2	C_3	C_4
C_1	1	4	3	7
C_2	4	8	4	2
C_3	3	4	2	5
C_4	7	2	5	0

(a) (b)
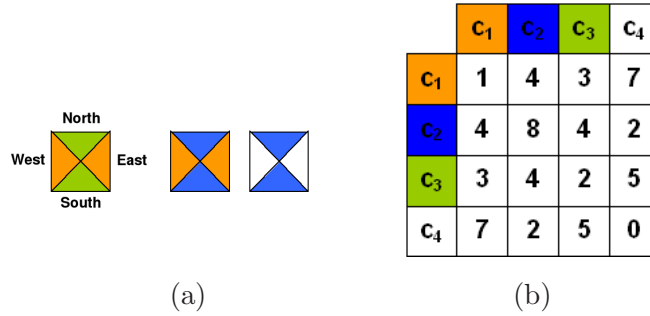
FIGURE 2.14: *A set of three self-assembly Wang tiles where the colours at their edges represent different glue types (a). An arbitrary matrix encoding the glue interactions (b).*

Assuming the existence of an infinite amount of tiles randomly distributed across the lattice and $\tau = 3$, Figure 2.15 shows a section of the lattice where a step-by-step tiles self-assembling takes place. At the beginning, two tiles self-assemble into a two-tiles aggregate since the glue strength between white and orange is greater than τ (Figure 2.15 (a)). After that, another tile moves downwards whilst approaching the aggregation – in this case no collision takes place (Figure 2.15 (b)). In the following time step, a third tile moves upwards self-assembling to the existing aggregate since the glue strength between

blue labelled edges is greater than τ (Figure 2.15 (c)). During the next two time steps, another random walk without collision and an extra tile self-assembly take place (Figures 2.15 (d)-(e)). Finally, the last tile performs a random movement locating itself adjacent to the aggregate (Figures 2.15 (f)). However, since the compatibility between orange and green is smaller than τ, the tile randomly moves to the left where it self-assembles to the rest of the aggregation (Figures 2.15 (g)).

(a) (b) (c) (d)

(d) (e) (f) (g)
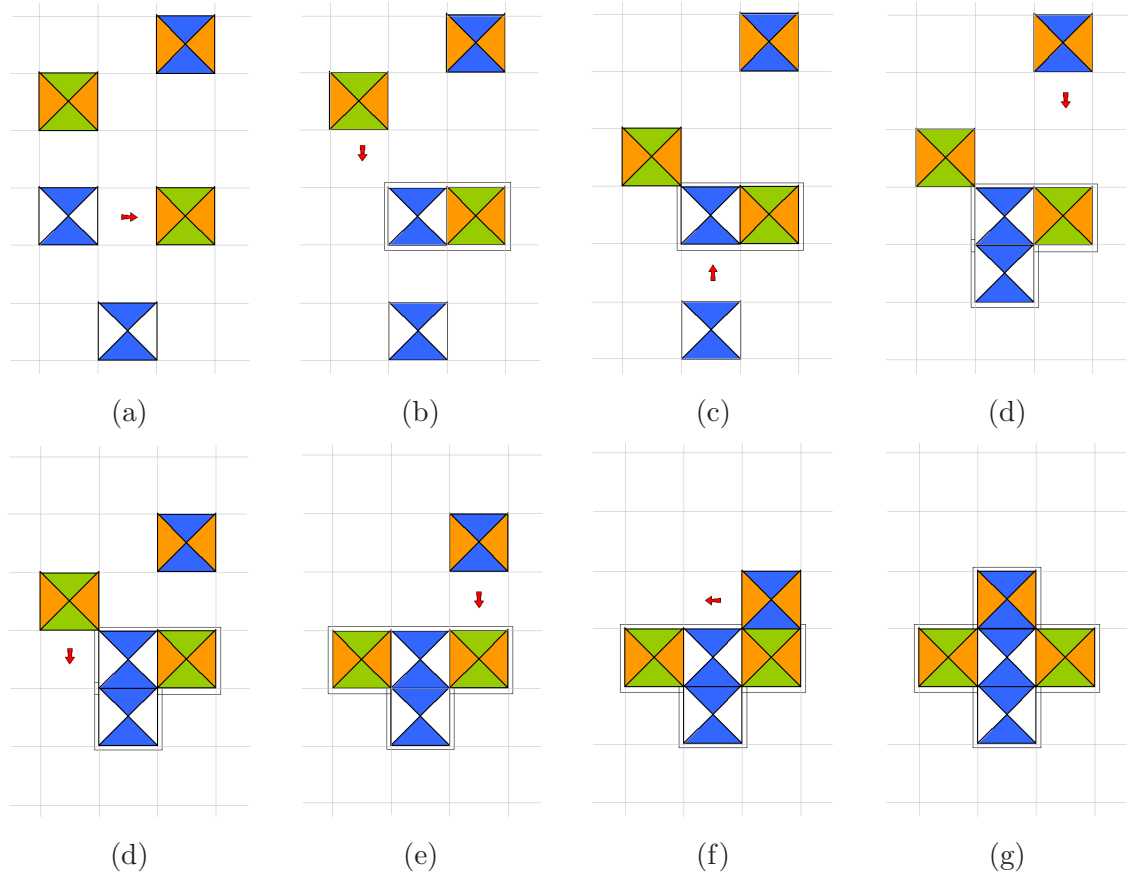
FIGURE 2.15: *A step-by-step self-assembly of five tiles randomly distributed across the lattice (a to g).*

2.2.3 Wang Tiles Complexity

The study of the complexity of solving a problem provides the means for quantifying the upper and lower bounds required to find the solution of a given problem. Standard complexity measures in computer science are calculated in terms of space and time. For example, one may seek to determine the complexity for a certain word (sequence of symbols). Thus, the *space complexity* is given as the length in part of the minimal instruction which generates the wanted sequence, whereas the *time complexity* is defined as the processing time required to generate the word out of the minimal instruction.

There are different ways of studying the efficiency in which a given collection of tiles self-assemble in a particular geometric shape. The computational complexity of such system has been analysed in terms of both program size and time with the aim of answering the following two questions:

Which is the minimum amount of tile types needed to create an arbitrary shape by means of self-assembly ?

Given the minimum amount of tiles, what is the minimum expected time required to create that given shape ?

The most representative cases study the computational complexity of self-assembly focus in the construction of rectangular shapes. In particular, the minimisation of the program size complexity for assembling a solid square of $N \times N$ tiles is given in terms of a

Tile Assembly Model with infinite supply of a finite number of tile types labelled with 0 or 1 [73]. The authors propose to construct a square employing binary counters which are basically rows of binary numbers (rows of tiles) "written" from bottom to top and from right to left. The approach claims an upper bound of $O(log\ N)$ tile types.

Another mechanism for self-assembling a square of $N \times N$ tiles is given in [74]. In this case, the proposed method exploits the inherent parallelism of assembly achieving time $\Theta(N)$. The work also contributes a set of tiles using optimal program size $\Theta(\frac{log\ N}{log\ log\ N})$ to self-assemble the square. This result is achieved due to the combination of a base conversion phase with the binary counters idea presented in [73]. That is, in the first step numbers are represented in base $\Theta(\frac{log\ N}{log\ log\ N})$ employing only $\Theta(\frac{log\ N}{log\ log\ N})$ digits. After that, a parallel self-assembly process that simulates base conversion to binary is employed, hence still achieving optimum time complexity.

Adleman et al. [75] define "Minimum Tile Set Problem (MTSP)" and "Tile Concentrations Problem (TCP)" as the optimum program size and optimum running time for the self-assembly of generalized shapes. In particular, the authors focus their research in the self-assembly of trees and squares of $N \times N$ tiles. For the first definition, it is demonstrated that the problem is both NP and NP-hard by giving a polynomial time algorithm of verification and reducing 3-CNF SAT to MTSP respectively. For the TCP, a $O(Log\ N)$-approximation deterministic algorithm that works for a large class of tile systems is presented.

In [76], a connection between self-assembly and computation is given by considering a shape as the output of the "program" executed by a set of tiles which self-assemble in the shape. Thus, the minimal number of tile types needed to self-assemble an arbitrary shape is bounded both above and below by the Kolmogorov complexity of the shape. In this case, the Kolmogorov complexity of a shape is defined as the smallest self-assembly program outputting a shape in terms of a list of locations, i.e. $K(S) = min\{ |s| \ s.t. \ U(s) =< S >\}$ where $K(S)$ is the Kolmogorov complexity of the shape S, $U(s)$ is a universal Turing machine executing a string s and $< S >$ is a binary encoding of the shape S. Although the analysis concerns a scaled up version of the shape, it is shown that the scale must not be a factor but that smaller versions of the same shape might require larger number of tile types.

Four generalized models of the original Tile Assembly Model, namely the flexible glues model, the multiple-temperature model, the multiple-tile model and the unique shape model are introduced and examined in [77]. For all them, an analysis of the program size complexity for the self-assembly of thin and thick rectangular shapes of $k \times N$ with $k \leq N$ is performed. A rectangular shape $k \times N$ is considered thin when $k < \frac{logN}{log \ log \ N \ - \ log \ log \ log \ N}$ and thick otherwise. The contributions regarding upper and lower bounds are summarised in Table 2.2.

An extra variety of the Tile Assembly Model is given in [78]. The purpose of this extended model is to simulate catalysis and self-replication in terms of a time-dependent glue strength approach. In other words, the glue strength between two juxtaposed tiles is

considered as dependent on the time for which they have been in neighbouring positions. Under these circumstances, the authors claim that the program size complexity of self-assembling $k \times N$ rectangular shape with $K \geq 2$ is $O(\frac{log\ N}{log\ log\ N})$.

	Thin Rectangles		Thick Rectangles
Original	$\Omega(\frac{N^{\frac{1}{k}}}{k})$	$O(N^{\frac{1}{k}})$	$O(\frac{log\ N}{log\ log\ N})$
Flexible Glue	$\Omega(\frac{N^{\frac{1}{k}}}{k})$	$O(N^{\frac{1}{k}})$	$O(\sqrt{log\ N})$
Multi-Temperature	$O(\frac{log\ N}{log\ log\ N})$		$O(\frac{log\ N}{log\ log\ N})$
Multi-Tile	$\Omega(\frac{log\ N}{log\ log\ N})$	$O(N^{\frac{1}{k}})$	$O(\frac{log\ N}{log\ log\ N})$
Unique Shape	$\Omega(\frac{N^{\frac{1}{k}}}{k})$	$O(N^{\frac{1}{k}})$	$O(\frac{log\ N}{log\ log\ N})$

TABLE 2.2: *Results regarding the lower bounds (Ω) and upper (O) bounds on program size complexity under five models for the self-assembly of thin and thick rectangular shapes.*

As well as the results on computational complexity presented above, researchers have found it feasible to extend these results in order to solve NP-complete problems in terms of tiles self-assembly for which there are unknown polynomial-time algorithms. A procedure for solving SubsetSum [79] [80] and a system that solves k-SAT non-deterministically in [81] are among the most familiar examples found in literature. Since this topic lies outside the scope of this dissertation, any further description about the details of implementation is omitted.

CHAPTER 3

Evolutionary Design

The purpose of this dissertation is to investigate automated design of self-organised and self-assembly systems by means of artificial evolution. For this reason, this chapter aims to survey the application of evolutionary algorithms as strategies for evolutionary design. The beginning intends to introduce a general view of evolutionary algorithms as methodologies applied to solve design problems. In particular, a variety of evolutionary design approaches among illustrative examples found in the literature are mentioned outlining the representation of the individuals, the initialization stage, the evaluation mechanisms and the expected goals. The second part of the chapter continues by focusing on how evolutionary algorithms have been successfully employed in evolutionary design optimisation of self-organising and self-assembly systems. In this part, a collection of works related to the research topic addressed in this dissertation highlighting motivation, subjacent models, main technical features and findings are showcased.

3.1 Aspects of Evolutionary Design

Evolutionary Algorithms (EAs) [82][83][84] are powerful stochastic search methods [85] [86] loosely inspired by Darwinian evolutionary processes [87]. An EA maintains a population of *genotypes*, each of which represents the solution to an optimisation (e.g. [88]), planning (e.g. [89, 90]) or control problem (e.g. [91]). This population of solutions goes through a series of *evaluation*, *recombination* and *mutation* phases defining what is called a *generation*. In the evaluation, the assessment of each genotype takes place, this being a fitness value that must be optimised. After that, the best solutions reproduce giving birth to similar offspring. The schema depicted in Figure 3.1 represents the flowchart for a generic EA.
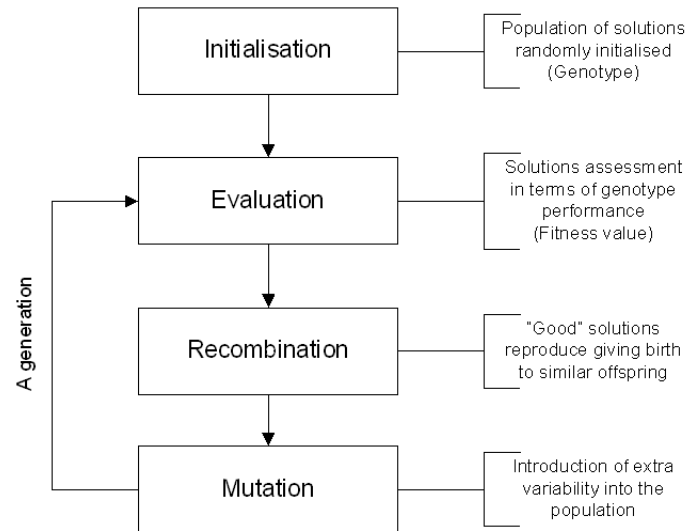
FIGURE 3.1: *An EA flowchart comprising the minimal set of features to be an evolutionary system which performs very well on problems where non-linear, stochastic or chaotic components are present.*

The idea to use this biologically-inspired method in order to solve different types of problems has been studied long before computers were used extensively as recorded in the collection

of "fossilized" readings available in [92]. Based on natural selection, reproduction, mutation and the survival of the fittest, evolutionary algorithms such as Genetic Algorithms (GAs) [93] [94] [95] [96], Genetic Programming (GP) [97], Evolutionary Programming (EP) [98], Evolution Strategy (ES) [99] [100] and Classifier System (CS) [101] [102] [103] have been broadly developed across both theory and applications during the last decades. Parameter optimisation [104] [105], heuristics design [106] [107] as well as many real-world problems such as DNA sequence alignment [108], cancer chemotherapy scheduling [109], stock market [110] [111], image processing [112] [113] and many multi-objective optimization problems [114] [115], to name but a few, form the most important application areas of EAs. In addition, constraint handling, machine learning, evolutionary art and design form an emerging field of applications of Darwinian ideas. Also, EAs have been applied to many design problems [116] giving birth to a branch of evolutionary computation called evolutionary design. A wide range of examples are showcased in [117] where four major types of evolutionary design have been identified according to the applicability and complexity of the algorithms and the type of the genotype representations: *evolutionary design optimisation*, *creative evolutionary design*, *evolutionary art* and *evolutionary artificial life forms*.

In *evolutionary design optimisation* the features of an established design are improved by means of artificial evolution. Usually, genotypes are specific to the type of problem and the relation genotype-phenotype is approaching one-to-one as there is almost no need for a mechanism to reconstruct the encoded representation. For instance, the design optimization of crushers in comminution circuits has been treated with an ES in [13].
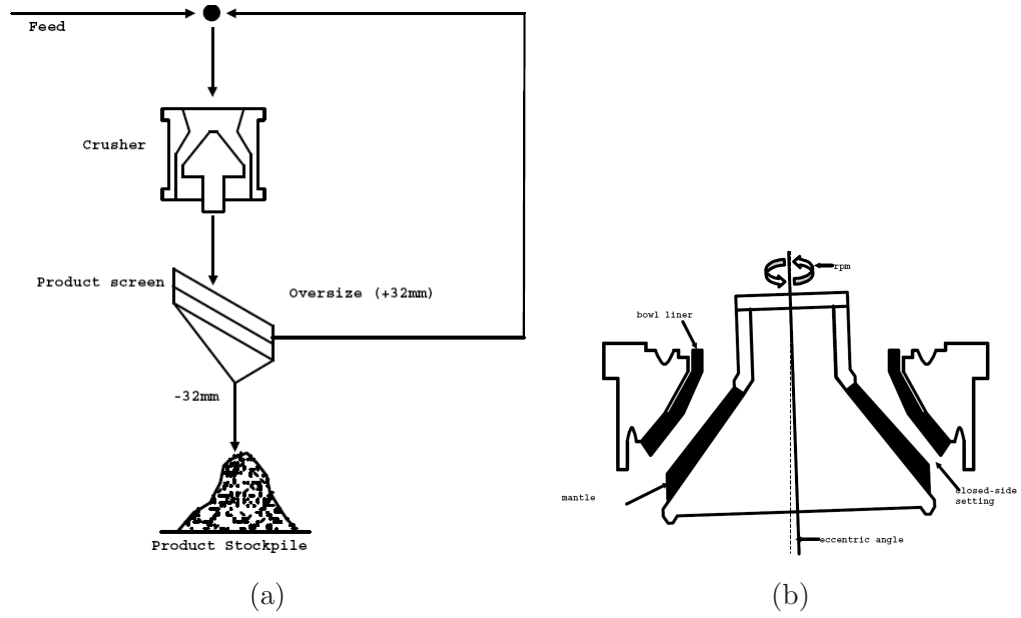
FIGURE 3.2: *A comminution circuit: particles of material enter the crusher from the top, the material is broken into small pieces by physical pressure at the closed-side of the conical head from where those pieces smaller than a certain size take part of the final product whilst the bigger ones recirculate (a). A detailed diagram of a cone crusher spinning at a certain rmp with an eccentric angle where the distance between the mantle and the the bowl liner determines the closed-side setting (b). Extracted from [13].*

The term "comminution" refers to the collection of physical processes like crushing, gridding and screening applied to a stream of material in order to change the size of its particles in the stream. For instance, a mill performs a comminution process over corn in order to obtain flour. One of the crucial components of this working circuit is the crusher (Figure 3.2 (b)) which has a conical shape and is composed of an inner crushing surface called a mantle, a conical crushing head that spins around the axis of the machine and an outer part called a bowl. Into this device, located at the closest point, there is a gap between the bowl and the crushing head defined as closed-side. During the comminution process (Figure 3.2 (a)), the particles of material enter the crusher from the top and whilst

the conical head spins, the material is broken into small pieces by the physical pressure at the closed-side. Thus, those pieces smaller than a certain size constitute part of the final product whilst the bigger ones recirculate. In this work, the capacity of the circuit, i.e. the capacity and power requirements of the crusher together with the size of the product are optimized. For this purpose, real numbers represent the closed-side setting, the rotational speed and the eccentric angle of the crushing head, whilst an array of coordinates map the geometrical representation of the shapes of the two liners. The initial population is set up with the real values of these pieces and a multi-objective optimisation takes place with a mutation operator. Although the design of these features are subject to physical and modelling constraints, the implemented strategy gives acceptable results, removing the need of arbitrary weightings made by engineers.

Creative evolutionary design, on the other hand, aims at generating novel and original forms from scratch. This class of evolutionary design is further divided in two: conceptual evolutionary design and generative evolutionary design. In the former, the embryologies mapping genotype to phenotype are rudimentary due to the simple encoding of the individuals. Conversely, complex and more general genotypes, advanced embryologies and more sophisticated GAs capable of evolving variable length representations are often part of the generative evolutionary design. Many approaches tackling designability problems of objects are available in the literature. For instance, a basic canonical genetic algorithm for the creation of three dimensional tables is employed in [14] [118]. A decomposition of the solid into a collection of adjoining not intersecting cuboids with variable width, height,

depth and position in space has been implemented in this approach. Thus, the genotype encodes primitives like the size, location, mass, stability, flat surface and supportiveness of a table. Starting with size and location, different experiments incorporating these primitives were gradually performed. As evolution progresses, individuals are evaluated by a fitness function mainly implemented with a table builder that reconstructs the phenotype from the genotype in order to assess its functionality. The GA employed in this case has produced some remarkably fit designs with the right size, nearly perfect flat surfaces and almost right mass like the tables shown in Figure 3.3.
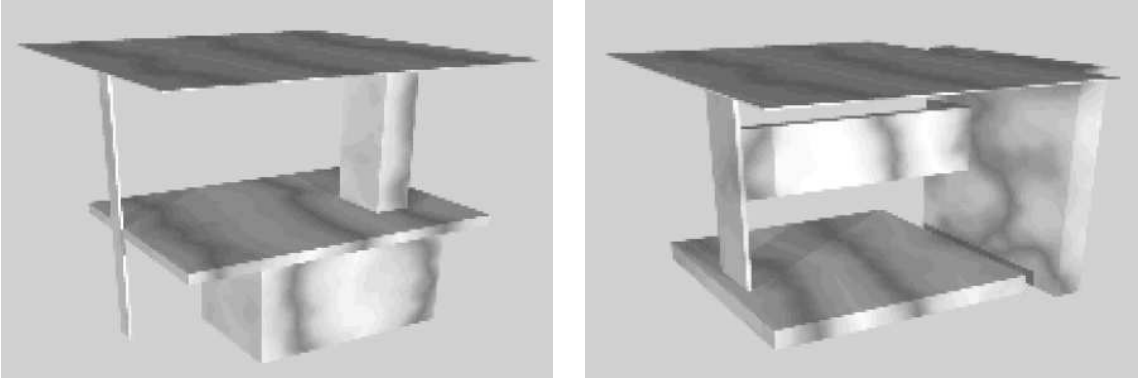
FIGURE 3.3: *Evolved table designs with nearly perfect flat top surfaces, great stability and almost right mass. These tables are composed by adjoining not intersecting cuboids with variable width, height, depth and position in space. Extracted from [14].*

In *evolutionary art*, shapes and images are evolved from scratch. In contrast to creative evolutionary design, the evaluation of individuals is subject to a human evaluator normally based on aesthetic appeal. In most of the cases, population sizes are small since a quick judgement is needed in every generation. Moreover, representations range from equations to grammar rules and unsophisticated GAs are employed in most of the cases where

there is no crossover operator. For instance, the evolution of prescriptive representations for automated design and assembly of a physical object is presented in [119]. Descriptive representations differ from prescriptive representations in the sense that the former is based on what to build whilst the latter is focused on how to build. The approach proposes the evolution of assembly sequences to build a goal structure by means of *situated development* which is a kind of artificial ontogeny where a developmental process to obtain the phenotype from its indirect representation encoded in the genotype is subject to physics. In this case, each genotype encodes an assembly plan represented by a sequential set of parametrized instructions such as move, rotate, put brick and take brick in order to construct the target object. Two types of goals have been used as a target structure: an explicit goal and an implicit goal. For the former, the target structure is represented with a bitmap and the genotypes are evaluated in terms of length, mass, number of missing and number of wrong located bricks. In the case of an implicit goal, the obtained structures encoded in the genotypes are evaluated in terms of length, mass and a shaded area [120] that measures the total amount of open volume beneath a structure.

A more general approach to evolutionary art is *interactive evolutionary computation* [121]. In this method, the optimisation is based on human subjective evaluation. According to the application domain, a human not only plays the role of the fitness function of a conventional EA, but also reduces the search space by intervening in the selection of the elite or modifying individuals of the population [122]. Contrary to the traditional EAs, the problems tackled with interactive evolutionary computation are modeled using small population size and few generations in order to avoid the human fatigue problem. Some of

these evolutionary design approaches are based on GAs like in CA parameters design [123], signal processing [124], ergonomic chair design [125] and biomorphs design [126].

The main goals in *evolutionary artificial life forms* are to have a deeper understanding of those mechanisms employed across natural evolution in order to find explanations related to forms and facts observed in nature. In this type of evolutionary design, populations are initialized mostly with random individuals although some approaches also consider the fittest individuals coming as outputs from previous experiments. In general, many representations are inspired in the genome structure of natural organisms requiring the implementation of complex genetic operators. These representations are occasionally flexible, variable in length and complex embryologies mapping genotypes to phenotypes are needed. For example, virtual creatures composed of hundreds of parts are designed by evolution in [15]. The main process for producing these living animates consists of a GA, an encoding method based in L-systems and a fabrication method for constructing creatures from the encoding. The population of the GA consists of a set of L-systems production rules randomly initialised, each of which encodes a creature. Thus, in each generation, an assembly method takes each individual, constructs the corresponding creature and returns its fitness value calculated according to the distance covered by the creature's centre of mass. As a result, the most common evolved forms were rolling creatures, serpents like the one in Figure 3.4 (a), inch-worms like in Figure 3.4 (b) and four legged walking creatures, among others.

(a)   (b)
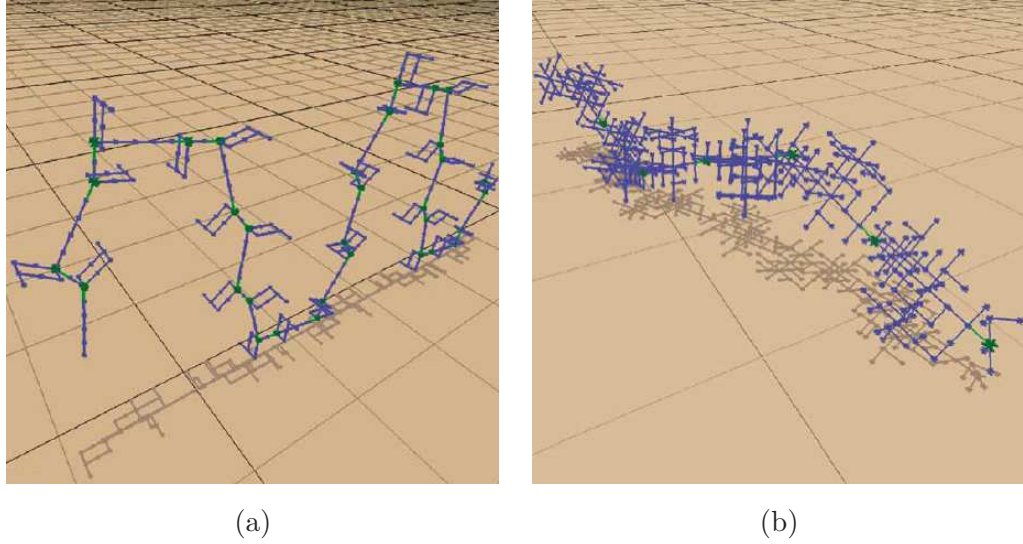
FIGURE 3.4: *A serpent (a) and an inch-worm (b) are some of the most common creatures resulting from evolving L-systems production rules with a GA where the fitness value is calculated according to the distance covered by the creature's centre of mass. Extracted from [15].*

3.2 EAs in Design Optimisation of Self-Organisation and Self-Assembly

Studies concerning nano scale systems like the sub-cellular world as well as those exploring macro interactive objects like galaxies, have featured the concepts of self-organisation and self-assembly as crucial mechanisms for understanding emergent systems' behaviour. As a result of these observations, many self-assembly and self-organisation computational models have been developed [127] [65] [128] opening the path to an extended variety of problems where EAs could be used since evolutionary techniques promise acceptable solutions when applied to problems that must accommodate and adapt to changes in the environment. In what follows, a collection of works on self-organisation and self-assembly design optimisation are surveyed showcasing their purpose, subjacent models and technical details of the employed evolutionary mechanisms.

The automated design of self-assembling *smart blocks* which are capable of building target structures is tackled with a GA-based approach in [17]. Each smart block is defined as a two-dimensional object comprised of four edges and an internal state machine. Each edge is associated with either a positive, negative or neutral polarity whereas the state machine contains a set of rules which are capable of changing the polarity of the edges whenever a sticking or unsticking event is captured. The edge state of a smart block is specified in terms of a vector $Q = (a_1, a_2, a_3, a_4)$ where $a_i \in \{-1, +1, 0\}$, e.g. the edge state of the smart block in Figure 3.5 is $Q = (+1, 0, 0, -1)$. The smart blocks live in a 2-dimensional multi-agent environment which attempts to simulate a simplified physical reality. When two blocks become adjacent and their colliding faces turn out to have opposite polarities, they attach to one another triggering a sticking event to be captured by their internal state machines.
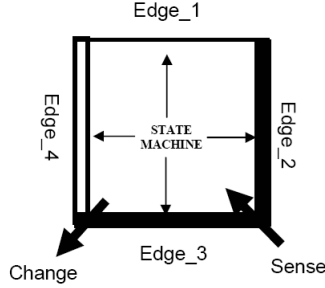
FIGURE 3.5: *Structure of a smart block where a single thin line stands for positive (+1), double lines for negative (-1) and a single thick line for neutral (0). The internal state machine can change the polarity of each edge after sensing sticking or unsticking events. Extracted from [16].*

If the colliding faces have equal polarity, the smart blocks repel. Otherwise, if a neutral polarized face becomes adjacent to any other nothing takes place, i.e. the smart blocks neither stick nor repel. After the event in the state machine is triggered, a change in the edge

polarities of the smart block takes place, i.e. a rule is executed. Rules are encoded as Table 3.1 shows where each line represents a sticking edge (S_i) initial edge state $(a_{1i}, a_{2i}, a_{3i}, a_{4i})$ and final edge state $(b_{1i}, b_{2i}, b_{3i}, b_{4i})$.

Sticking Edge	Initial Edge States	Final Edge States
S_1	$a_{11}\ a_{21}\ a_{31}\ a_{41}$	$b_{11}\ b_{21}\ b_{31}\ b_{41}$
S_2	$a_{12}\ a_{22}\ a_{32}\ a_{42}$	$b_{12}\ b_{22}\ b_{32}\ b_{42}$
$\vdots$	$\vdots$	$\vdots$
S_n	$a_{1n}\ a_{2n}\ a_{3n}\ a_{4n}$	$b_{1n}\ b_{2n}\ b_{3n}\ b_{4n}$

TABLE 3.1: *Results regarding the upper and lower bounds on program size complexity under five models for the self-assembly of thin and thick rectangular shapes.*

Based on these interaction mechanisms, the self-assembly process begins placing a smart block in the environment. All smart blocks are initialised with the same edge state, i.e. the same polarity. For each of the subsequent blocks, all the sticking possibilities are checked with the current built structure and a location is finally chosen at random. The whole process stops once the structure either can no longer grow or contains 100 blocks. Thus, in order to proceed with the automatic design of smart blocks, a GA evolving a colony of rule sets encoded in a string is employed. After the simulation, parent rules are chosen using a roulette wheel selection and according to the values returned by the fitness function. In particular, two different types of evaluation methods were used: shape matching and stable structure. The first fitness function measures how close the final aggregate matches

the desired one by counting the number of smart blocks located in and out of the target structure. On the other hand, in the second evaluation method the goal is to generate stable structures, i.e. aggregates that stop growing after reaching a certain polarization at its borders for which further sticking events cannot occur. Three classes of experiments were performed addressing how the initial edge states affect the final structure, what kind of internal states help get a stable structure and what is the effect of imposing symmetry on the agents. Some of the achieved structures are depicted in Figure 3.6.
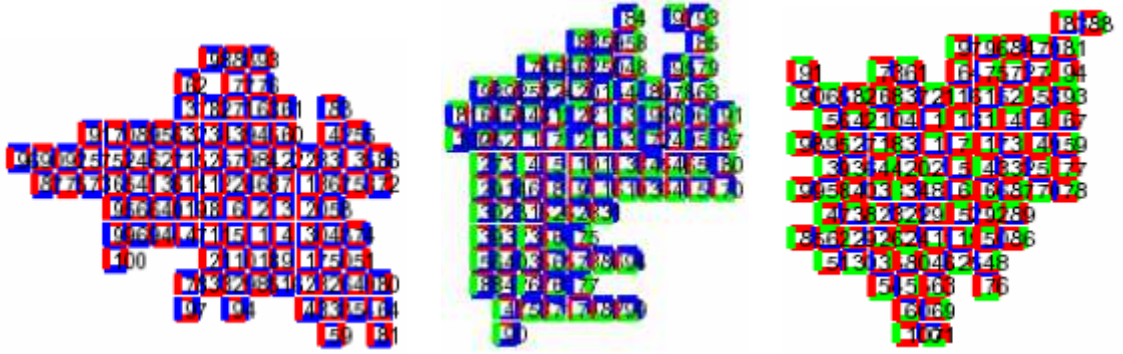
FIGURE 3.6: *Three different assembled structures generated from varied initial block states. Extracted from [17].*

Extending the previous approach, a model for the automated design of *artificial enzymes* was introduced in [16]. Enzymes are biopolymers that mediate a variety of bioprocesses such as fermentation, animal's digestion, sugar conversion in plants, etc.. On this occasion, the agents are classified in *sea blocks* and *enzyme blocks* both sharing a two-dimensional multi-agent environment and performing the same type of block-to-block interactions as described above. An enzyme is defined as a stable assembled structure composed of enzyme blocks each of which has an internal state machine initially set up with the

same set of rules. Enzymes are assumed static, i.e. with fixed number of enzyme blocks, shape and surrounding polarities, and capable of producing sea blocks assemblages. A sea block moves with Brownian-like motion and when it becomes adjacent to an enzyme block, they stick to one another triggering a simulated catalyst reaction where a transition in their internal states and change in their edge polarities take place. As the simulation runs, a finite sequence of reactions induce the creation of a structure by means of self-assembly. Once a certain number of sea blocks is assembled into the structure, another sequence of reactions separates the formed aggregation from the enzyme, i.e. they split. Hence, in order to perform the automated design of artificial enzymes, a GA with individuals encoding the enzyme size, its shape and its edge states together with the internal rule set of sea blocks is defined. In each generation, an individual is evaluated according to whether the number of blocks, the shape and the polarities of the self-assembled conformations are the same as the original enzyme ranking high those individuals capable of generating enzymes. Three experiments were carried out: one-block enzyme design, three-block enzyme design and four-block enzyme design. For each generation, every member of the population defines a potential enzyme which is allowed to grow in the environment until it splits into two or more objects: the enzyme itself and aggregations. Thus, in the first experiment, one-block enzymes produced aggregations with a maximum size of six blocks from where some of them were self-replications, i.e. sea blocks with the same polarities of an enzyme. On the other hand, using four-block enzymes has speeded up the process achieving larger and more regular aggregations than with one-block enzymes like the one shown in Figure 3.7. All in all, the experimental results show that the cooperation taking place among enzymes and

agents can make the generation of self-assembled structures more flexible and general than the approach followed in [17]. Following the idea of this model, an approach for designing self-assembly DNA-based structures is presented in [129].
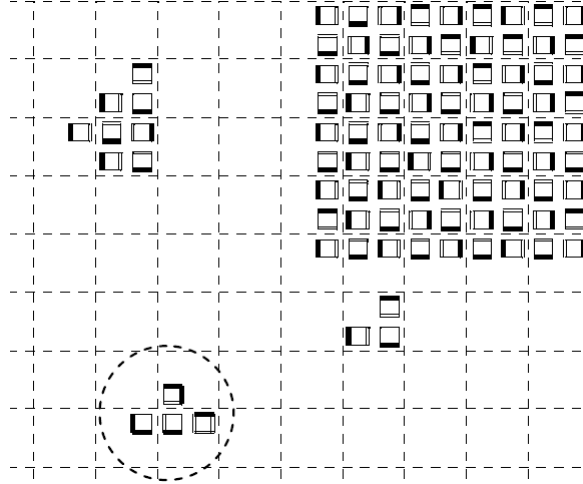
FIGURE 3.7: *A four-block enzyme (circled) and other self-assembled independent structures in the same figure where some of them are larger than the enzyme itself. Extracted from [16].*

Another relevant example of an EA applied to the design of self-assembly experimental settings is given in [130]. In this approach, a population of amphiphiles living in a real chemical system exhibiting complex varieties of self-assembled molecular aggregations like micelles, oil droplets and vesicles is optimised according to its *turbidity level.* This functionality is a straightforward optical property of a system where high-throughput assays can be easily achieved with a spectrophotometer. In this study, the genotype represents a receipt of 16 amphiphiles $g = (g_1, g_2, ..., g_{16})$ where g_i represents the number of volume units per amphiphile type. At the initialization stage, the GA was set up with 30 randomly initialized recipes each of which is produced and tested in the laboratory using multiwell plates hosting a single recipe at a time and which is capable of containing up to

three replicates of each recipe. The turbidity readings at each generation are employed to generate the recipes for the subsequent. Thus, a recipe's fitness is calculated as the mean turbidity minus its standard deviation and the next generations of recipes are computed according to a specific procedure based on the probability value assigned to each individual. The findings demonstrate that the turbidity of vesicle populations mainly composed by unilamellar, multilamellar and oil droplets was optimised as their sizes correlated with the turbidity level. It is thought that variances of this approach can be applied to optimize an array of other kinds of chemical properties even though the bottleneck in the methodology is caused by the time required for laboratory procedures.

In relation to the previous work, a different evolutionary algorithm for designing micelles formations is presented in [18] where a GA is linked to a model of chemical reaction systems based on a dynamic-bounding dissipative particle dynamics (dbDPD) framework. In this case, the GA has been employed in order to optimize the parameters of a dbDPD model using a set of vectors of chemical system variables $g = (g_1, ..., g_N)$ as individuals of the population. In contrast to the standard GA, parents are selected from the entire population of genomes tested in previous generations, i.e. not just those tested in the immediately preceding generation. Therefore, every parent produces one child genome g_c by mutation with the constraint that g_c does not duplicate a previous individual tested by the EA. With this scheme, one of the experiments is focused on amphiphile aggregations in water where the EA tunes dbDPD parameters for 500 particles of water and 500 particles of tail-head dimers. As micelles spontaneously self-assemble and their size is maximised, the fitness function is based in a micelles-detecting algorithm that locates all the amphiphiles

structures made up of a core of tails surrounded by heads. For this experiment, the results show that the EA had no difficulty in creating large micelles although good solutions were found at early generations and sometimes in a more gradual way. In the same work, this evolutionary approach is also applied for the design of ligation of uniform oligomers and non-uniform trimmers. Two comparative states of the evolutionary design of micelles is shown in Figure 3.8.

(a)   (b)
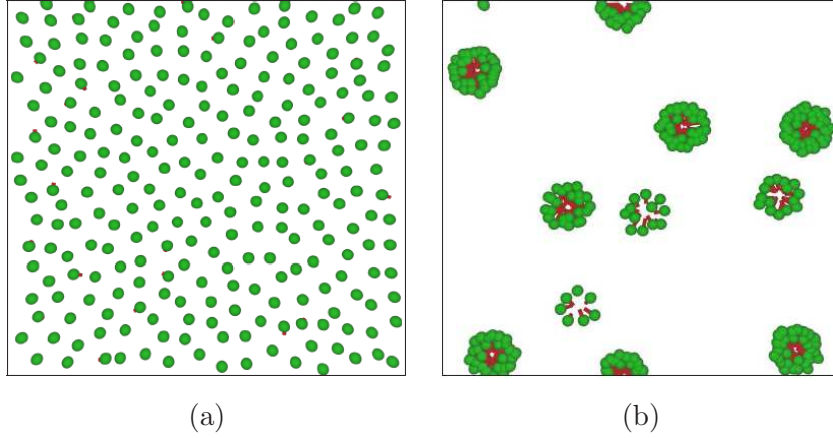
FIGURE 3.8: *Before (a) and after (b) the evolutionary design of dbDPD parameters for the micelles design. Extracted from [18].*

An approach for evolving swarms capable of building 3D structures is presented in [19]. In this case, a set of similar rule-based swarms living in a continuous world seeded with cubic construction objects is designed by artificial evolution in order to produce interesting 3D target structures. A swarm consists of a number of equally acting agents each of which decides on a specific action whenever a collision takes place with a cubic construction element. Thus, the available set of actions allows a swarm agent to create or destroy a local construction element, to destroy a remote collision object, to declare itself or another agent as the swarm centre or to recant the swarm centre. As an agent moves during the simulation

period, its velocity is updated with an acceleration vector calculated according to the value

of six weighted steering urges defined as centre, separation, alignment, cohesion, ground and

random. In the GA, an agent is represented by a set of five consecutive vectors representing

each of the six weighted steering urges, a maximum value for acceleration, a maximum

value for velocity and a set of rules which determine its constructional abilities. Each rule

is defined as a set of preconditions and an consequential action to take place whenever an

construction element is found. The recombination among individuals takes place using a

two point crossover assigning the smallest set of rules to the resultant offspring. The fitness

function is defined as the constructional difference between the reached structure and the

desired 3D one measuring the covering obtained volume, non covering obtained volume and

the desired volume. The findings of this approach result are very promising as interesting

structures have been found for seven different tested targets. Figure 3.9 depicts one of the

targets and the achieved results.
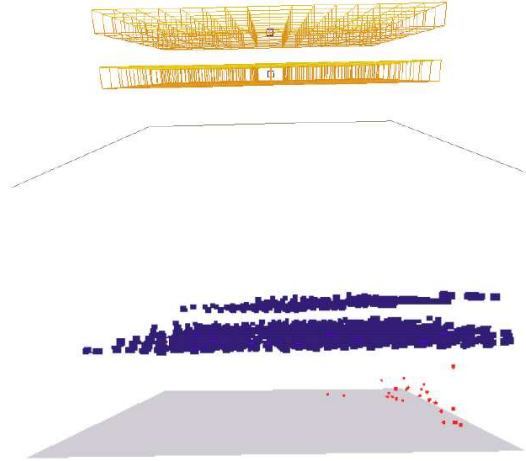
FIGURE 3.9: *Two parallel planes (in yellow) given as target structures and evolved swarms generating two-level flats (in blue). Extracted from [19].*

CHAPTER 4

Proposed Methodology

As it was seen in Chapter 3, employing evolutionary algorithms for design optimization of self-organised and self-assembly systems has been taking place at all scales in many fields of science. Following this line and in connection to the computational models of self-organisation and self-assembly proposed in Chapter 2, this chapter enlarges on a GA-based approach for the design optimisation of CAs parameter values and self-assembly Wang tiles. In the following sections, an introduction and characterization of the problems to be addressed and the reason for which they are the topic of investigation in this thesis. The characterisation is done in terms of the relationship genotype-phenotype-fitness as well as the domain of the problems to be treated. After that, overall descriptions of both the architecture and the operational aspects of the EA to be employed are given. Finally, an information-based metric and a morphological image analysis method are presented as the two fitness functions that embody the evaluation stage of the proposed strategies. Notice that neither of these two are part of development, although they are key elements contributing to this investigation.

4.1   Characterization of the Problems

This dissertation focuses on evolutionary design optimisation of self-organised and self-assembly systems. In particular, the interest lies in self-organised and self-assembly problems where the mapping from genotype to phenotype and then from phenotype to fitness is a highly complex, non-linear and in some cases stochastic relationship. It is non-linear because different genotypes could lead to the same phenotype and, on the other hand, different phenotypes might encode the same genotype due to stochasticity. This intricate relationship makes the assessment of the genotype very difficult since the same (different) fitness value could be assigned to different (the same) genotypes. The intricate mapping is sketched in Figure 4.1.
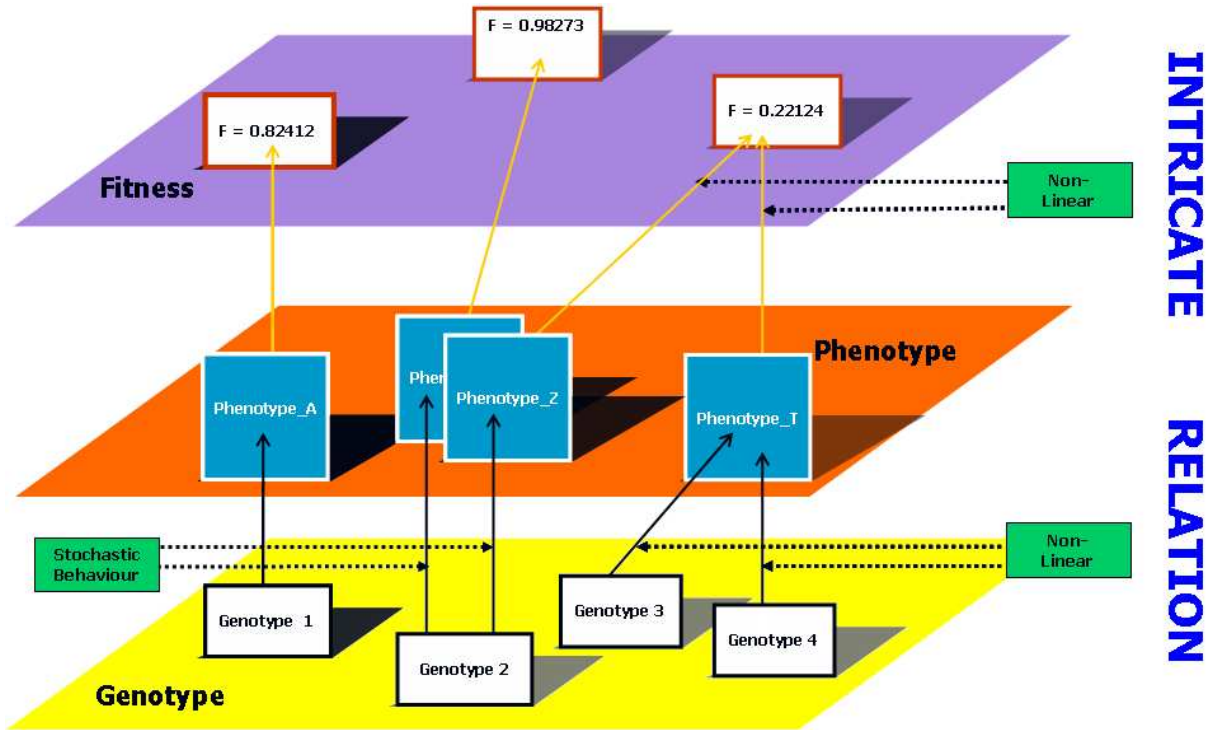
FIGURE 4.1: *The highly complex, non-linear and stochastic relationship taking place across the mapping from genotype to phenotype and then from phenotype to fitness.*

On the one hand, the interest in combining CAs with EAs lies in the use of a method for the automated manufacture of spatio-temporal behaviour resulting from the correct combination of input parameter values and rules. This goal is related to a more general question:

Is it possible to make an evolutionary-driven specification of the laws governing an observed CAs dynamics?

Two types of CAs evolutionary design optimisation problems are presented: continuous optimisation with fixed length individuals and discrete optimisation with fixed length individuals. In the first case, it is a continuous optimisation problem because the EA will evolve CA input parameter values of real numbers domain. In the second case it is discrete because the EA will evolve the CA rules where their encoding corresponds to integer domain. In both cases, it is fixed length because the structure of the individuals is defined in terms of either the amount of input parameters to tune or the number of rules to design.

On the other hand, the interest in combining self-assembly Wang tiles with EAs has its grounds on the use of a method for the automated construction of supra-structures that emerge as a result of tiles interaction. In this case, the research goal is to answer the question:

Is it possible to make an evolutionary design of the glue types associated to the tile edges in order to obtain a particular supra-structure by means of self-assembly?

In particular, the employed agent-based system represents a discrete optimisation problem with variable length individuals. It is discrete because the domain of the parameters to evolve is given by a finite number of enumerable glue types and variable length size because the supra-structure to construct is independent of the diversity of tiles. Thus, both discrete/continuous and fixed/variable problems are covered.

4.2   Cellular Automata Parameters Design

In order to undertake the evolutionary design optimisation of CA parameters, a GA will be employed with the aim to fine tune the parametric quantities – in what follows the term parameters will be used in a general way when referring to either numerical values or CA rules unless it is specified. When a collection of parameters (GA chromosomes) is set up to the CA model, the iterative application of CA rules gives results in a spatio-temporal behaviour to be captured in a two-dimensional pattern which is compared in terms of similarity against a user defined (target) pattern. This process, together with replication and mutation operators, is repeated for a certain number of generations.

The proposed GA is shown in Figure 4.2. Each P_i represents a fixed length individual mapping CA parameters. After random initialisation, in the evaluation stage, each P_i is set up in the CA generating a candidate pattern (phenotype) which is compared against a target pattern for similarity using an information distance-based metric described in more details in the following section. This metric returns a numerical representation of similarity that is considered as the fitness of each individual. Henceforth, those individuals generating similar patterns to the target are better ranked and become most likely to survive.
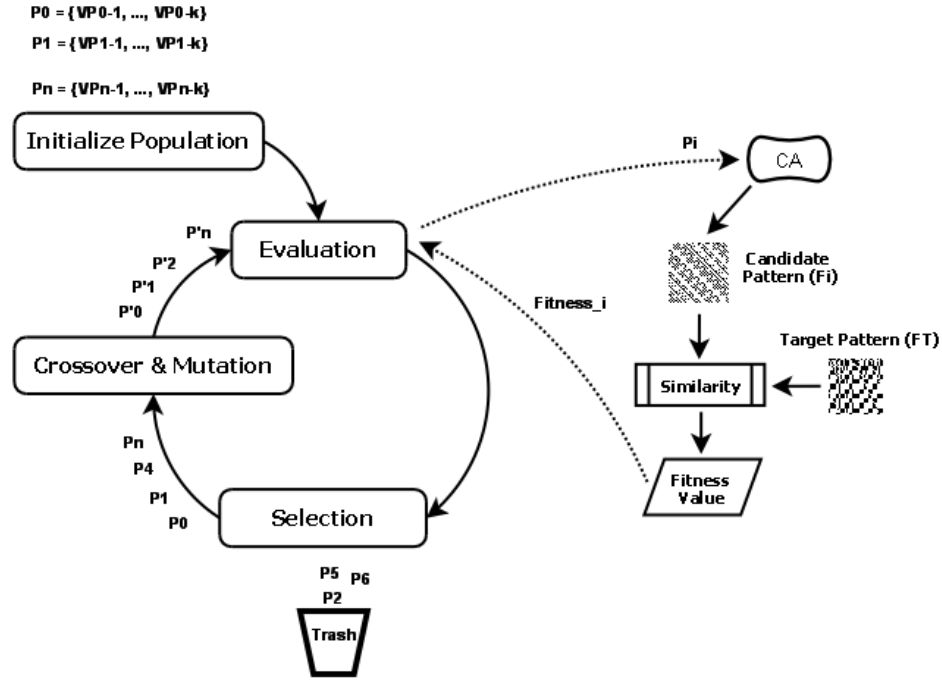
FIGURE 4.2: *Evolutionary approach for the evolutionary design optimisation of CA parameters. A population of parameter sets (genotype) is randomly initialised. After that, each individual is set up as input of the CA generating a candidate pattern (phenotype) to be compared to a target for similarity. This returns the fitness value of a genotype. Later on, the application of genetic operators follows where the best ranked individuals are likely to pass throughout selection, crossover and mutation stages.*

4.3 Self-assembly Wang Tiles design

The proposed approach for the automatic design of self-assembly Wang tiles follows a similar methodology to the one previously described. In this case, the main goal of the GA is to evolve a population of self-assembly Wang tile families. Broadly speaking, a self-assembly Wang tile family is a template representing a class of tile and with the capability to be instantiated with several identical copies. This descriptor comprises a set of glue types each of which associated to the four sides of a self-assembly Wang tile as defined in Chapter 2.
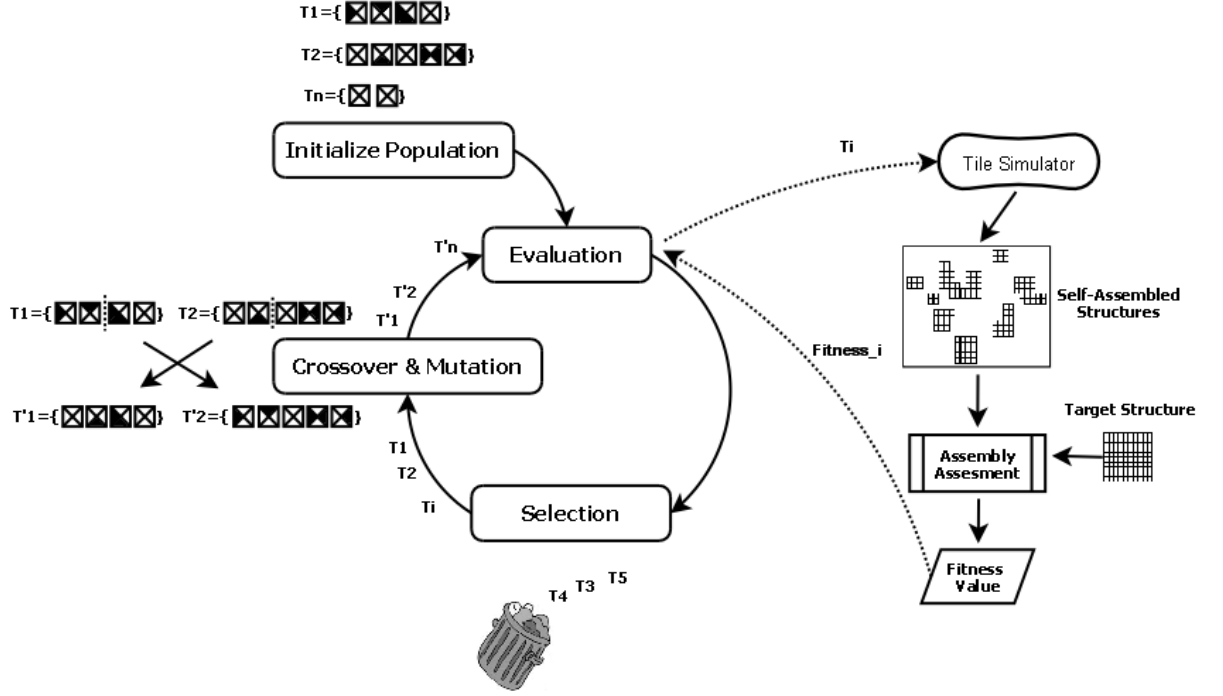
FIGURE 4.3: *Evolutionary approach for the evolutionary design optimisation of self-assembly Wang tiles. A population of self-assembling Wang tiles family (genotype) is randomly initialised. After that, each individual is set up into a tiles simulator from where the emerging self-assembled aggregations (phenotypes) are compared against a target structure for similarity. This comparison returns in the fitness of the individual. Later on, the application of genetic operators follows where the best ranked individuals are likely to pass throughout selection, crossover and mutation stages.*

Thus, each tile family provides a number of tiles which are randomly located into a simulation environment where they perform a random walk and interact with others for a fixed amount of time self-assembling in aggregates. Once the simulation finishes, aggregates are compared for similarity to a user defined (target) structure. This process, together with replication and mutation operators, is applied to the population and repeated for a certain number of generations. The proposed GA is depicted in Figure 4.3. In the beginning, variable length individuals of the population $(T_1, T_2, \ldots, T_n)$ are created as sets of self-assembly Wang tile families each of which are generated with four random glue types. At

evaluation stage, a number of instances from each of the families of an individual (T_i) is located into a tile simulator comprised of a lattice surface, an interaction function and a temperature threshold. The lattice surface is a finite two dimensional structure made of square sites where tiles are located and can move under Brownian motion. Whenever two tiles become adjacent they either self-assemble into supra structures or bounce off according to whether the glue types at their colliding edges are compatible or not. This compatibility is computed by the interaction function which evaluates the strength between the two glue types against the temperature of the system. After a finite number of simulation steps the assembly assessment takes place comparing the emergent aggregates to the user defined structure for shape similarity. This metric returns a numerical representation that is considered as the fitness $(Fitness_i)$ of each individual. Thus, individuals capable of creating aggregates similar to the specified target are better ranked and become the most likely to survive across generations.

4.4 Fitness Functions

The following sections present a domain independent metric to compute similarity and a method to characterise morphology of structures. These two are key external contributions to this research and embody the fitness functions employed in the two evolutionary approaches described in Section 4.3 and Section 4.2.

4.4.1 Universal Similarity Metric

Mathematically speaking, the term *similarity* refers to the degree of symmetry in both analogy and resemblance between two or more objects in respect to one or multiple common

features. The assessment of *similarity* can be described as a feature-matching process that contrasts common and distinctive characteristics of entities under comparison [131]. This concept is employed in a large number of research fields summarized in [132].

There are many approaches to measure the similarity [133]. One of them is called Representational Distortion (RD) where the similarity between two entities is given in terms of the "complexity" required to "distort" or "transform" the representation of one object into the representation of another object [132]. In other words, the simpler the transformation distorting one representation to the other, the more similar the analysed objects are assumed to be. This underlying idea led to a new class of appropriate metrics for measuring effective similarities between objects represented as finite binary strings. This class is built upon the Kolgomorov complexity [134] [135] which measures the amount of information needed to specify an observed object. In this way, the *Normalised Information Distance (NID)* [136] is defined as an information metric, proved to be universal among a wide class of computable normalized information measures. Thus, given the observed data x and y, their similarity in terms of information distance is established as:

$$NID(x, y) = \frac{Max\Big\{K(x|y^*), K(y|x^*)\Big\}}{Max\Big\{K(x), K(y)\Big\}}$$

where

$K(x)$ is the Kolmogorov complexity of x

$K(x|y^*)$ is the Kolmogorov complexity of x given y^* $\qquad$ (4.1)

In the formula shown above, x^* is the shortest Turing machine program capable of computing x and $K(x)$ is the length of x^*. The subjacent idea is that those representations generated by a short program are considered simpler or less complex than those generated by a long program. For example, a program that writes a string of $1's$ is simpler than a program for the creation of a random string. Likewise, $K(x|y^*)$ is the length of a shortest program to produce x from the input y^*. For example, given the sequences $x_1 = 12345678$ and $y_2 = 23456789$ the values of $K(y_2|x_1^*)$ and $K(x_1|y_2^*)$ are considered small because the simple instructions *add 1 to each digit* as x_1 and *subtract 1 from each digit* as y_2 suffice to transform one into the other. In the same way, the sequences $x_3 = 12345678$ and $y_4 = 246810121416$ related by the instruction *multiply each digit by 2* and *divide each digit by 2* are presumed to have similar information distance. On the other hand, considering the sequences $x_5 = 12345678$ and $y_6 = 357911131517$ they are viewed as less similar in terms of information distance than the previous examples because two operations *multiply each digit by 2 and add 1* are required to transform one into the other.

Since the Kolmogorov complexity is not computable, an approximation of NID can be developed in terms of a standard compressor C. The result of this approach is called *Normalised Compressed Distance (NCD)* which after applied to the observed data results in a non-negative number r such that $0 < r < 1 + \epsilon$. Thus, small (large) values of r denote a big (small) similarity and ϵ represents an upper bond constant due to imperfections found in the implementation of C. The definition of NCD is given in Formula 4.2 and in [137] its (quasi-) universality as a metric is proven.

$$NCD(x,y) = \frac{C(xy) - Min\Big\{C(x), C(y)\Big\}}{Max\Big\{C(x), C(y)\Big\}}$$

where

$C(xy)$ is the compressed size of x and y concatenated

$$C(x) \text{ is compressed size of } x \tag{4.2}$$

Broadly speaking, this compression-based similarity distance, commonly known as the *Universal Similarity Metric (USM)*, determines the similarity in terms of information distance between pairs of objects according to the most dominant common feature. USM concerns to the classification of genomes [138], classification of music pieces [139], plagiarism of computer programs [140], image registration, letters phylogeny [141], protein structure comparison [142] [143], genotyping [144] [145], tumor subclassification [146] and many others.

In order to choose the most accurate compressor C, a set of experiments with compressors Jzip, JBzip2, Jgzip, Zlib, JazzGzip, Jbar and PPMPZ were conducted. For this purpose, six images (Figure 4.4) were created. Among them, img_0 depicts an arbitrary square whilst among the other five, img_i $1 \leq i \leq 5$, one replicates the square four times and the others show randomly introduced modifications. Thus, img_0 with itself and with each img_i taken in turns were set as parameters to USM values of which are shown in Table 4.1.
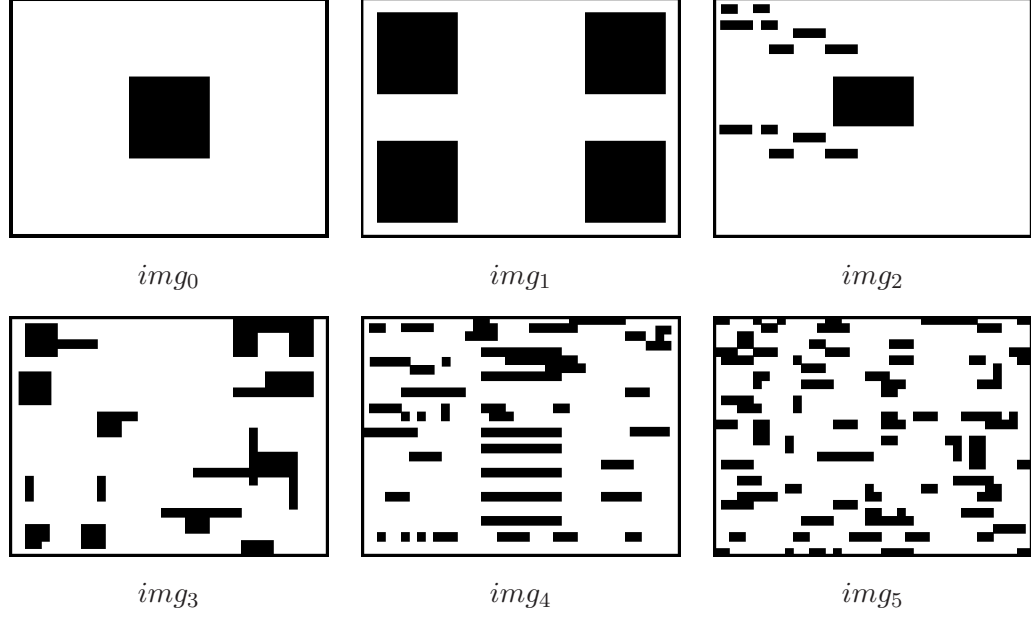
FIGURE 4.4: *An arbitrary square captured in a black and white image (img$_0$) and four randomly introduced modifications (img$_1$, img$_2$, img$_3$, img$_4$ and img$_5$).*

	Jzip	JBzip2	Jgzip	Zlib	JazzGzip	Jbar	PPMPZ
USM(img$_0$, img$_0$)	0.897	0.503	0.105	0.963	0.684	0.997	0.858
USM(img$_0$, img$_1$)	0.996	0.771	1.358	1.047	1.011	0.998	0.984
USM(img$_0$, img$_2$)	0.990	0.801	1.254	1.056	1.046	0.998	0.932
USM(img$_0$, img$_3$)	1.128	0.868	1.233	1.185	1.019	0.997	0.992
USM(img$_0$, img$_4$)	1.111	0.873	0.931	1.168	1.040	0.995	0.993
USM(img$_0$, img$_5$)	1.216	0.955	1.390	1.266	1.076	0.996	1.019

TABLE 4.1: *Results for $USM(img_0, img_0)$ and $USM(img_0, img_i)$ implemented with Jzip, JBzip2, Jgzip, Zlib, JazzGzip, Jbar and PPMPZ. Blue colour indicates the best compression performance.*

Since in this case the purpose of the USM is to assess how similar two given images are – approaching to 0 when identical and to 1 when there is a small amount of information in common – and since compression technologies tend to look for regularities in the data they are compressing – they reduce storage space by saying "just repeat that block of common information with this symbol" instead of detailing the whole thing – the idea of the experiments lies in that the compression ratio of two images with information in common should be smaller than the compression ratio of two images with dissimilar content. In other words, $USM(img_0, img_0)$ is expected to achieve the smallest value since the black square is repeated twice and the compressor would use one symbol for encoding the replicated information. With a similar argument, $USM(img_0, img_1)$ is expected to rank in second place since the black square is repeated four times and the compressor would use four symbols for encoding the replicated information. However, when $USM(img_0, img_2)$ is evaluated, the common information could be seen limited to a section of the black square. This reduced similarity plus the fact that img_2 contains short, scattered, black stripes, which have nothing in common with img_0, should make the resultant USM value to rank in third place. A similar reasoning applied to the rest of the parameters tells that $USM(img_0, img_3)$ is expected to rank fourth, $USM(img_0, img_4)$ to rank fifth and $USM(img_0, img_5)$ to be the closest to 1 since the similarity tends to be null as the randomly introduced modifications generate a bigger number of scattered tiny squares.

From the conducted experiments, it turns out that JBzip2 is the most appropriate compression algorithm since its associated trend is monotonic increasing (see Figure 4.5), i.e. matching with the performed analysis.
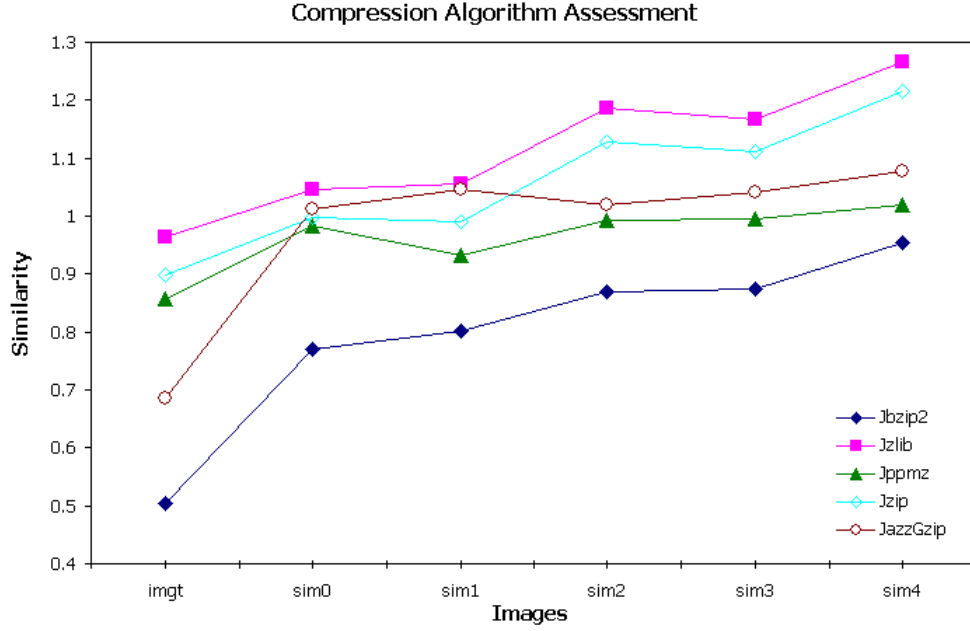
FIGURE 4.5: *Assessment trends of five compression algorithms implementing the similarity metric for comparing img_0 towards itself and img_0 towards each of the img_i $1 \leq i \leq 5$.*

The plots support a visual comparison given that the similarity in terms of information between img_0 and img_1 ranks first, between img_0 and img_2 ranks second, between img_0 and img_3 ranks third, between img_0 and img_4 ranks fourth and between img_0 and img_5 ranks last. The trends for compressors Jzip and PPMPZ show that these two algorithms might perform well as approximations. However, Jzip assigns to $USM(img_0, img_4)$ an inaccurate value since img_3 is more similar to img_0 than img_4 is, and similarly PPMPZ assigns to $USM(img_0, img_2)$ an unexpected value since img_1 is visually more similar to img_0 than img_2 is. It is also interesting to note that, with the exception of JBzip2, the minimum value for similarity is always bigger than 1. One of the reasons for this happening might be related to extra information needed for storing the encoded compression which translated to JBzip2 means the best storing performance.

4.4.2   Additive Shape Descriptors – Minkowski Functionals

Different approaches to describe and characterize complex structures are, for instance, Fourier transformations [147] [148] [149] [150] and wavelet transforms [151] [152] [153] [154]. Although for regular patterns Fourier transformations are able to provide both length scales and orientation order, they fail when trying to distinguish irregular structures of different topology [155] in terms of content and shape. Similarly, wavelet analysis is based on the extraction of local amplitudes and phases but it becomes a non-trivial, numerically inefficient method in most of the cases [156].

In contrast, the *Morphological Image Analysis method (MIA)* characterises the morphology of a structure in terms of geometrical (i.e. configuration) and topological (i.e. connectivity) descriptors [157] [158] that uniquely identify any object. Broadly speaking, *morphology* refers to the study of the internal composition of a structure, that is the arrangement of composites and how they aggregate to form a whole. With origins in the calculation of the so-called *Minkowski functionals* [159], MIA involves the computation of the area (A), the boundary length or perimeter (U), and the connectivity or Euler-Poincaré characteristic (χ) that describes the connectivity of a given structure.

As an example, if one considers the 2-dimensional black pixelated drawings of Figure 4.6, then the area (A) is the number of black pixels that make up the pattern, the perimeter (U) is the number of lines making up both internal and external borders, and the Euler (χ) is given by the connectivity of the pattern, i.e. the number of connected black pixels regions minus the number of completely enclosed white pixelated regions in the image.
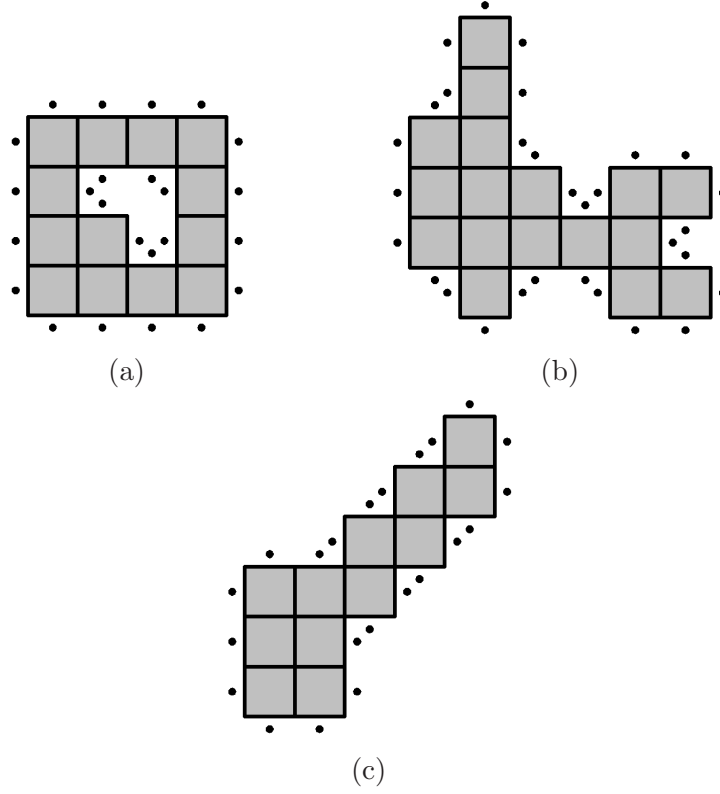
FIGURE 4.6: *Minkowski functionals for three pixelated patterns where in (a) $A = 13$, $U = 24$ and $\chi = 0$, (b) $A = 17$, $U = 30$ and $\chi = 1$ and (c) $A = 12$, $U = 22$ and $\chi = 1$.*

Since rotated occurrences of a given structure have the same morphological values, the radius of gyration (Rg) of the area about the x-axis is additionally calculated. The radius of gyration describes the way in which the mass of a structure is distributed around its centroidal axis in terms of its moment of inertia and its area.

Research works belonging to the field of statistical physics [160], cosmology [161] [162], biology [163], seismology [164] and polymer sciences [165] [166] have demonstrated that MIA is a robust approach to typify both 2-dimensional and 3-dimensional structures. For instance, the geometrical and topological characterization of irregular spatial-temporal

pattern structures emerging in a chemical reaction-diffusion system is performed in [156]. The structures to be characterized are digitized images of 512×480 square lattice of pixels with 256 grey levels depicting hexagonal, lamellar and turbulence structures resulting from gradual variation of temperature and reagent concentrations in the system. Since these variations unfold in different structural patterns, the goal is to capture their morphology using Minkowski functionals in order to develop an analytical transition model. Thus, a detailed characterization of the spatial structure is performed quantifying the area, boundary line and connectivity considering different level contours. These give, as a result, the input to a set of formulae embodied in functional equations to be approximated with polynomials of very low degree. This resulting polynomial model makes up a mathematical approach which is capable of capturing the transitions from hexagonal point and stripe patterns to turbulent structures. Comparing experimental data, the authors conclude that this analytical model quantitatively describes irregular spatial patterns and their transition similar to thermodynamic phase transitions.

4.5 Conclusions

This chapter has defined the type of problems to be addressed in the rest of this dissertation. In particular, the interest lies in the evolutionary design optimisation of self-organised and self-assembly systems, both problems characterised by presenting a highly complex, non-linear genotype-phenotype-fitness relationship. In addition, the problems have been featured as continuous optimisation with fixed length individuals, discrete optimisation with fixed length individuals and discrete optimisation with variable length individuals.

The foundations of two evolutionary design optimisation approaches have also been presented. The first one is intended for the automated design optimisation of CA parameters. Its interest lies not only in producing spatio-temporal patterns that emerge from the application of CA rules, but also in answering whether it is possible to make an evolutionary-driven specification of the laws governing a given system's dynamics. In a similar way, the second approach is intended for the automated design optimisation of self-assembly Wang tiles. In this case, the proposed methodology is intended to contribute to an evolutionary mechanism for the computer-assisted specification of tiles capable of self-assembling into a user-defined structure.

This chapter has also presented a general overview of the GA topology to be employed in the rest of the dissertation. In addition, an information-based metric (USM) and a Morphological Image Analysis method (Minkowski functionals) were introduced as the mechanisms in charge to assess the performance of the individuals of the population. On the one hand, the USM is a domain independent metric that determines the similarity among two given objects in terms to the most dominant common features. In addition to this, experiments and analyses to determine the best data compressors for the USM implementation were also given. On the other hand, the Minkowski funtionals were presented as a mechanism to characterize complex structures in terms of geometrical and topological descriptors such as area, perimeter and Euler characteristic.

In what follows, an experimental implementation for each of the systems to design will be fully described. Details about the population, genetic representation, initialisation phase, evaluation process, selection scheme and genetic operators will be provided.

CHAPTER 5

An Evolutionary Approach to Cellular Automata Parameter Design

The purpose of this chapter is to report on an EA based approach for the continuous design optimisation of CAs input parameter values by means of artificial evolution. This methodology contributes a strategy for the identification of CAs for which spatio-temporal snapshots are available. The idea is to consider the CA as a black box device where the performance of its input parameter values is measured in terms of information distance between the associated emergent behaviour and a sample snapshot. Essentially, the interest lies in answering the following: having a CA together with a captured spatio-temporal behaviour, what are the causes underlying the given effect ? To begin with, the architectural characteristics of the methodology are introduced. Next, the conducted experiments along with the technical details of the utilised models and the experimental data sets are presented. Finally, the analysed results and conclusions complete this experimental part of the dissertation. The research to be reported in this chapter has been published in "7th International Conference of Adaptive Computing in Design and Manufacture" [167].

5.1   Architecture

The following approach employs a GA whose goal is to tune up a collection of CA input parameters in the following way. When a CA receives a specific set of input parameter values, the iterative application of rules across time results in an associated spatio-temporal behaviour which is captured in a two-dimensional pattern. This pattern, interpreted as a visual image, is then compared in terms of similarity to a user defined target. Similarity is then given by a numerical score or value that specifies how (dis)similar the evolved pattern is with the target one. Repeating this two-step process, i.e. generation and comparison, over different input parameter values results in a collection of captured patterns where those better resembling the target are considered more fit. Hence, the better ranked a pattern is, the closer the collection of parameter values are to those that gave rise to the original (target) pattern.
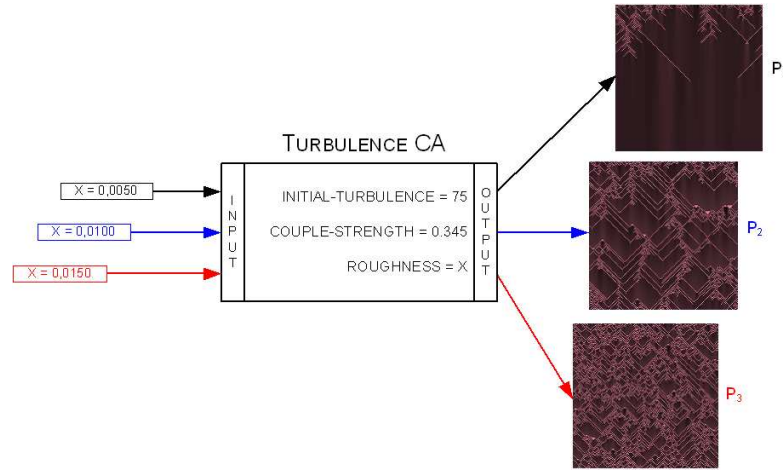
FIGURE 5.1:   *Turbulence CA with input parameters INITIAL-TURBULENCE, COUPLE-STRENGTH and ROUGHNESS. By fixing INITIAL-TURBULENCE = 75 and COUPLE-STRENGTH = 0.345, the snapshots P_1, P_2 and P_3 are obtained when ROUGHNESS is set up to 0.0050, 0.0100 and 0.0150 respectively.*

In order to illustrate this approach with an example, consider the Turbulence CA (to be defined in Subsection 5.1.1) with input parameters INITIAL-TURBULENCE, COUPLE-STRENGTH and ROUGHNESS where the first two are fixed to 75 and 0.345 respectively and the last one is left variable. So, if ROUGHNESS takes values 0.0050, 0.0100 and 0.0150 in turns, then the associated patterns P_1, P_2 and P_3 will be created (see Figure 5.1).

(a)   (b)
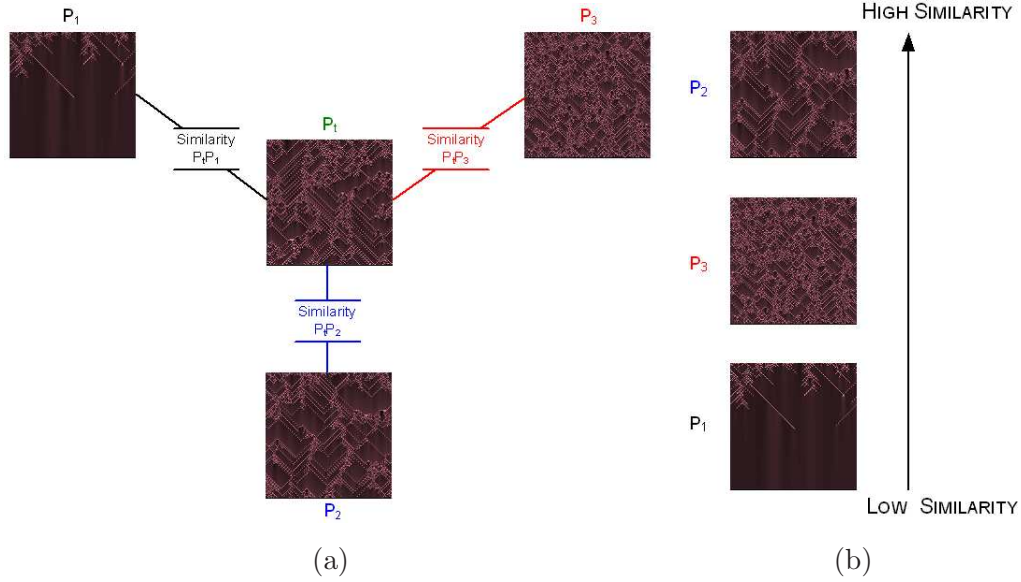
FIGURE 5.2: *Process of comparison by similarity. A target snapshot P_t is compared to each of the three generated patterns P_1, P_2 and P_3 (a). The comparison by similarity produces a ranking where those patterns better resembling the target snapshot are better positioned in similarity scale (b).*

Following the example, consider the pattern P_t, depicted in Figure 5.2 (a), as the target pattern. Then, the proposed approach is that P_1, P_2 and P_3 will be compared in terms of similarity to P_t where those better resembling the target will be highly ranked. In this case, since P_2 is more similar to P_t than P_3 is to P_t, plus the fact that P_1 is even less similar, it emerges that the obtained ranking of similarity results in P_2, P_3 and P_1, as Figure 5.2 (b) shows. Thus, pattern creation and pattern comparison together with recombination and

mutation operators will be repeated for a certain number of generations over a population of individuals which represent sets of CA input parameter values. Figure 5.3 depicts a chartflow of the extended GA where initialisation corresponds to the individuals' generation; evaluation is the stage involving pattern creation and pattern comparison; and crossover and mutation are the moment when recombination and mutation operators are applied upon the entire population.
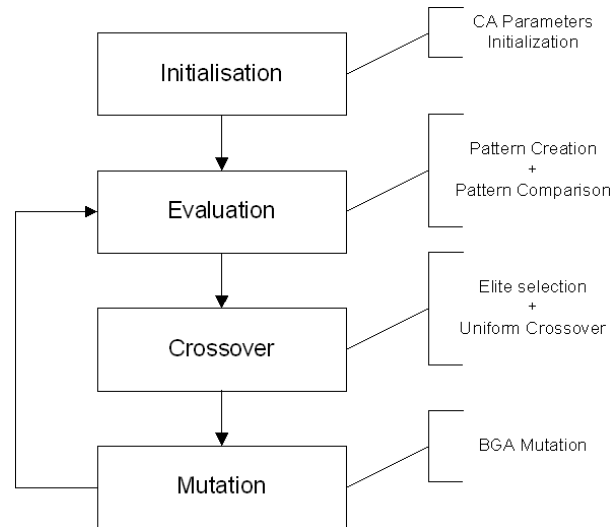
FIGURE 5.3: *Flowchart of the extended GA for evolving cellular automata.*

5.1.1 Problem Description

Informally introduced before, the **Turbulence CA**, a NetLogo [168] implementation, is the chosen model for this evolutionary design optimisation. Taking roots in StarLisp and Logo, NetLogo is a multi-agent programming language, modelling tool and authoring environment for simulating complex phenomena. The chosen CA is deterministic and it has been selected as particularly well-suited to evolutionary design since the combination of different values assigned to its input parameters are very well reflected in the resultant snapshots.

Turbulence CA is based on a couple map lattice (CML) which is an abstract representation introduced in 1983 by Kunuhuko Kaneko as a model to characterize spatio-temporal chaos and pattern formation. Viewed as a dynamical system, CML has been applied for modelling biological problems such as heart rhythm, electrical activities in neural tissues, fluid dynamics, population dynamics and neural dynamics to name but a few [169]. In particular, Turbulence CA helps in understanding the relationship between the turbulence, laminarity and viscosity of a fluid flowing through a pipe showing how the roughness of its surface can affect the fluid's behaviour. Thus, a cell in the lattice represents either a laminar or a turbulent behaviour subject to the value of its surrounding cells. This model consists of three continuous variables: INITIAL-TURBULENCE, COUPLING-STRENGTH (determining the extent to which cells influence their neighbours) and ROUGHNESS (controlling the friction upon the modelled fluid). Four snapshots are supplied in Figure 5.4.

FIGURE 5.4: *Sample snapshots of spatio-temporal patterns from Turbulence CA library.*

The following table summarises the key concepts underlying the problem to be addressed. **Problem name** gives a general self-explanatory outline of the problem; **Instance** characterises the particular problem to address; i.e. the CA description emphasising the input parameters to design; **Solution** details the expected type of answers for solving the problem; and **Measure** indicates the mechanism by which the solution's performance is scored.

Problem name	Turbulence CA parameter values design
Instance	a CA simulating a fluid flowing through a pipe for understanding the relationship between turbulence, viscosity and roughness
Solution	three real numbers associated to turbulence, viscosity and roughness working as input of a CA which outputs a spatio-temporal pattern captured in a bitmap image
Measure	comparison of the captured CA output against a user defined (target) pattern in terms of similarity by means of USM

TABLE 5.1: *Problem name, instance, solution and measure for the automated design optimisation of Turbulence CA input parameter values.*

5.1.2 Population, Genetic Representation and Initialisation

The representation of the individuals is given by a collection of real-coded genotypes. That is, each individual's chromosome represents a set of input parameter values in terms of real numbers. This is not only a more intuitive representation in this context, but research [170] suggests that real-coded GAs may be more efficient than their binary counterparts. Hence, an individual of the population is defined as a collection of three real numbers i, c and r representing INITIAL-TURBULENCE, COUPLING-STRENGTH and ROUGH-NESS respectively. Formally speaking, the population and its individuals are defined as:

$$Pop = \{Ind_1, Ind_2, \ldots, Ind_n\}$$

$$Ind_j = \{i_j, c_j, r_j\} \tag{5.1}$$

The three genes of each individual are randomly initialised with real values belonging to the ranges shown in Inequality 5.2 and to an accuracy of five decimals precision.

$$0.0 \leq i_j \leq 100.0$$

$$0.0 \leq c_j \leq 1.0$$

$$0.0000 \leq r_j \leq 0.0250 \tag{5.2}$$

To sum up, evaluation, recombination and mutation operators are applied and repeated for a certain number of generations over this population in which those individuals capable of generating patterns similar to the user defined pattern will survive and is likely to guide the evolutionary process across generations.

5.1.3   Selection Scheme and Genetic Operators

During the evolutionary process, offspring are obtained using uniform crossover [171] where, for each gene, the algorithm determines randomly from which parent to draw an allele as exemplified in Table 5.2. Mutation is implemented using the Breeder Genetic Algorithm operator [172] which selects a value from a constant-size distribution either side of the initial value. Individuals are selected to be parents using roulette wheel selection [173] which essentially assigns to each individual a "slice" of the roulette wheel whose size is proportional to the individual's fitness. Thus, fitter parents have a greater probability of being selected when the virtual wheel is spun, but all individuals in the population have some chance of selection. The $(\mu + \lambda)$ replacement strategy is employed, where the children and parents are considered together and the best (fittest) μ individuals are chosen to form

the next generation's population. The GA is run over 100 generations with a population size of 20, i.e. $\mu = 20$. At each generation, 10 offspring are created, i.e. $\lambda = 10$. Crossover between parents occurs with 0.7 probability ($XProb$) and mutation with 0.3 probability ($MProb$) per individual.

Individuals	$XProb = 0.7$	Offspring
$Ind_i = (a_1^i, a_2^i, a_3^i, a_4^i, a_5^i, a_6^i, a_7^i)$		$Ind_i^o = (a_1^i, a_2^j, a_3^i, a_4^i, a_5^i, a_6^j, a_7^i)$
$Ind_j = (a_1^j, a_2^j, a_3^j, a_4^j, a_5^j, a_6^j, a_7^j)$		$Ind_j^o = (a_1^j, a_2^i, a_3^j, a_4^j, a_5^j, a_6^i, a_7^j)$

TABLE 5.2: *Uniform crossover with probability* 0.7 *applied to individuals Ind_i and Ind_j giving birth to offspring Ind_i^o and Ind_j^o with the second, third, and sixth alleles interchanged.*

5.1.4 Evaluation Procedure

The evaluation phase consists of two stages: pattern creation and pattern comparison.

Pattern Creation

As previously stated, an individual Ind_j is initialised with three real numbers representing the input parameters: INITIAL-TURBULENCE, COUPLING-STRENGTH and ROUGH-NESS. At this stage, three parameter values are set up as input to the CA where iterative applications of the CA rules give rise to an associated spatio-temporal behaviour pattern (P_j) to be captured in an image. To illustrate, instances of images capturing the spatio-temporal behavioural patterns resulting from three individuals that encode the minimum, median and maximum values for each of the input parameters are shown in Figure 5.5.

(a)   (b)   (c)
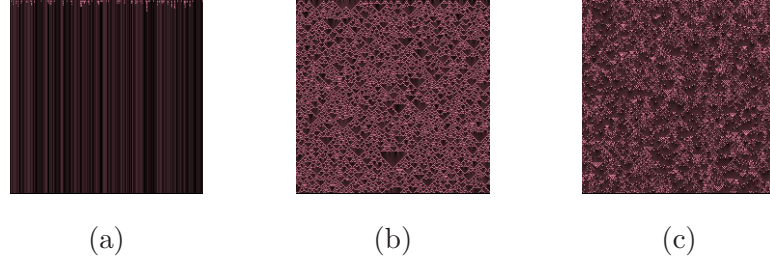
FIGURE 5.5: *Turbulence CA spatio-temporal behaviour patterns generated after applying the rules capped at* 200 *iterations with random initialisation upon individuals: (a)* $\{i = 50.5, c = 0.0, r = 0.0000\}$; *(b)* $\{i = 50.5, c = 0.50, r = 0.0125\}$; *(c)* $\{i = 100.0, c = 1.0, r = 0.0250\}$.

Pattern Comparison

Once the image capturing the spatio-temporal behaviour pattern is generated, its comparison against a target pattern (T) takes place. In order to perform this task, two arbitrary groups of target patterns comprising five instances each were specified. For their creation, the initial state of the cells were randomly initialised and the CA rules were executed capped at 200 iterations. For the first group, the parameters were set up as INITIAL-TURBULENCE = 50.5, COUPLING = 0.5 and ROUGHNESS = 0.0010 whilst for the second group INITIAL-TURBULENCE = 50.5, COUPLING = 0.6 and ROUGHNESS = 0.0010 (all shown in Figure 5.6). Each of these snapshots is used a as target pattern in separate GA experiments. Thus, during the evaluation stage of an experiment, the fitness function compares each P_j to the current T for similarity using the USM introduced in Chapter 4. This metric returns a numerical representation that is considered as the fitness of each individual, i.e. the value to minimise by the GA. Hence, those individuals generating patterns that are similar to the target pattern are better ranked and become more likely to survive throughout generations.
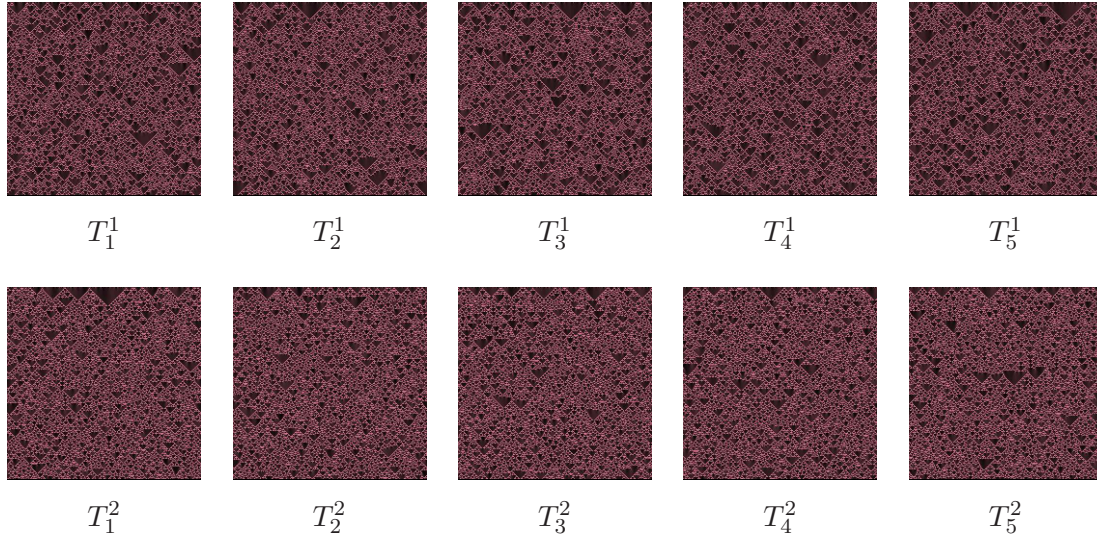
FIGURE 5.6: *Two arbitrary groups of target snapshots for Turbulence CA created with random initialisation, rules execution capped at* 200 *iterations with INITIAL-TURBULENCE* = 50.5, *COU-PLING* = 0.5, *ROUGHNESS* = 0.0010 *(T_1^1 - T_5^1) and INITIAL-TURBULENCE* = 50.5, *COU-PLING* = 0.6, *ROUGHNESS* = 0.0010 *(T_1^2 - T_5^2).*

Given that for all the generated spatio-temporal behaviour the first row of cells is randomly initialised, the captured patterns might differ for each initialisation. This influences the comparison for similarity which would certainly give different results for the same individual. For this reason, a reliable estimation of the true fitness is needed and individuals must be evaluated several times. Consequently, each individual is evaluated five times and its fitness is calculated as the average.

5.2   Results

This section aims to present the results of the GA experiments using the parameters summarised in Table 5.3 for each run. For all the experiments the population size, the amount of generations, the number of times an individual is evaluated, the crossover probability and the mutation probability were fixed. The length of the individuals is also fixed and is

given by the amount of CA input parameters, i.e. length 3 in case of Turbulence CA.

Population Size	Generations	Evaluations	XProb	MProb
20	100	5	0.7	0.3

TABLE 5.3: *GA parameters for Turbulence CA evolutionary design optimisation.*

A visual inspection of a representative result exposes visual features that can be picked out for both groups of target patterns. For instance, an inspection of Figure 5.7 reveals that the evolved pattern (P_5^1) and the target snapshot (T_5^1) share the density of light pixels, with a number of dark triangles dispersed throughout the image. Moreover, the larger triangles at the top of T_5^1 have also been successfully represented in P_5^1.

$$T_5^1 \qquad\qquad P_5^1 \qquad\qquad T_3^2 \qquad\qquad P_3^2$$
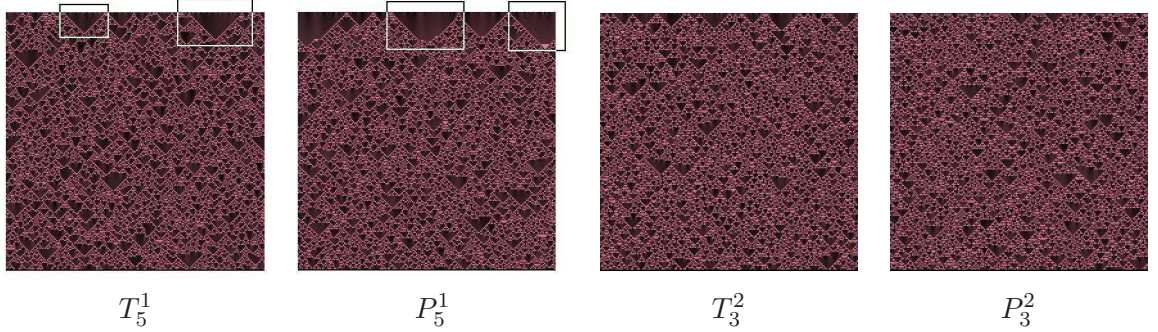
FIGURE 5.7: *Representative target patterns (T_5^1 and T_3^2) and designoids (P_5^1 and P_3^2) resulting from the first and second data set of Turbulence CA. The annotations show successfully achieved and well-produced features at the top of P_5^1.*

Table 5.4 list the best individuals found in the experiments using the ten target patterns presented in Section 5.1.4. The evolved patterns, called designoids [174], are labelled under **P** followed by three columns i_p, c_p and r_p containing the genes of the fittest individuals. The target patterns and their creational values appear in column **T** and

i_T, c_T, r_T respectively. In the last column, **USM(P, T)** lists the fitness values of the evolved individuals resulting from the similarity measure against the target.

P	i_P	c_P	r_P	T	i_T	c_T	r_T	USM(P, T)
P_1^1	62.90586	0.5235	0.00074	T_1^1	50.5	0.5	0.001	0.95657
P_2^1	69.47874	0.48819	0.00094	T_2^1	50.5	0.5	0.001	0.95738
P_3^1	51.97374	0.51587	0.0113	T_3^1	50.5	0.5	0.001	0.95725
P_4^1	71.83373	0.46997	0.0008	T_4^1	50.5	0.5	0.001	0.95817
P_5^1	43.03929	0.51941	0.00838	T_5^1	50.5	0.5	0.001	0.95653
P_1^2	55.92625	0.56941	0.00791	T_1^2	50.5	0.6	0.001	0.95785
P_2^2	65.98295	0.58869	0.01377	T_2^2	50.5	0.6	0.001	0.95657
P_3^2	68.37877	0.60256	0.0114	T_3^2	50.5	0.6	0.001	0.95608
P_4^2	53.86632	0.60256	0.00673	T_4^2	50.5	0.6	0.001	0.95915
P_5^2	48.21686	0.60334	0.00549	T_5^2	50.5	0.6	0.001	0.95781

TABLE 5.4: *Results for the first and second data set of Turbulence CA snapshots. Best designoids (P) with evolved individuals (i_P, c_P, r_P), targets (T) with their corresponding creational values (i_T, c_T, r_T), and fitness (USM(P, T)).*

Certainly, the quality of the results is difficult to discern when inspecting the snapshots in Figure 5.7. However, if a more heterogeneous target pattern is employed, the distinction could be much clearer. For this reason, a new target, T^3, is defined with a low value for i and giving more influence to r (see Figure 5.8). The results for this experiment are listed in Table 5.5 and visually available in P^3 of Figure 5.8. These findings are noticeably better since it has a USM value of 0.91980 which is substantially lower than the USM values obtained in the previous experiments. Moreover, the evolved pattern is very similar to the

target since it achieves half image with zero turbulence area followed by the inverted, pink triangular structures and the turbulent region at the bottom.

$$T^3 \qquad\qquad P^3$$
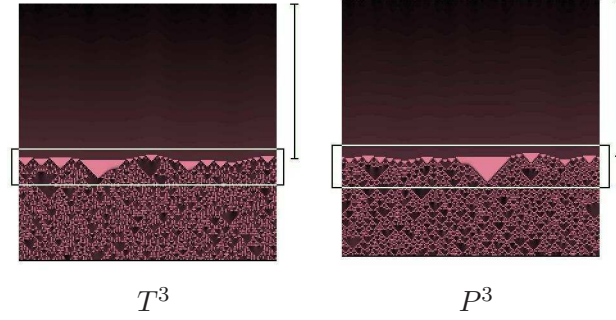
FIGURE 5.8: *Target pattern (T^3) and designoid (P^3) from the extra experiment of Turbulence CA parameter values design problem. The annotations show particularly well-produced features since the pink triangular structures, the upper plain area and the lower rough one have been successfully achieved.*

P	i_P	c_P	r_P	T	i_T	c_T	r_T	USM(P, T)
P^3	8.85724	0.54241	0.00356	T^3	6.98285	0.83854	0.00377	0.9198

TABLE 5.5: *Result for the extra experiment. Best designoid (P) with evolved individual (i_P, c_P, r_P), target (T) with its creational values (i_T, c_T, r_T) and fitness (USM(P, T)).*

In order to do an analytical evaluation of the results shown in Table 5.4 and Table 5.5, consider the error of the genes $e(g)$ and the average error $E(P)$ of an individual defined as follows. Let $Ind_P = \{i_P, c_P, r_P\}$ be one of the fittest individuals of the Turbulence CA parameter values design problem associated with a spatio-temporal behaviour pattern P and the target snapshot T obtained from the target values i_T, c_T and r_T. Then, $e(g)$ is

defined as in Equation 5.3 and $E(P)$ as in Equation 5.4.

$$e(g) = \frac{\mid g_T - g_P \mid}{g_T} \tag{5.3}$$

$$E(P) = \frac{e(i_P) + e(c_P) + e(r_P)}{3} \tag{5.4}$$

Thus, computing $e(g)$ and $E(P)$ for the individuals of Table 5.4 and Table 5.5 an

errors list depicted in Table 5.6 is obtained. Looking at this last table, it is evident that

the GA has mixed success in approximating the actual target parameters.

P	**$e(i)$**	**$e(c)$**	**$e(r)$**	**$E(P)$**
P_1^1	0.24566	0.04699	0.25948	0.18405
P_2^1	0.37582	0.02361	0.06288	0.1541
P_3^1	0.02918	0.03173	10.29811	3.45301
P_4^1	0.42245	0.06005	0.19738	0.22663
P_5^1	0.14774	0.03883	7.3825	2.52302
P_1^2	0.10745	0.05098	6.91322	2.35722
P_2^2	0.30659	0.01885	12.7747	4.36672
P_3^2	0.35404	0.00426	10.40107	3.58646
P_4^2	0.06666	0.00426	5.73318	1.9347
P_5^2	0.04521	0.00556	4.48623	1.51234
P^3	0.26843	0.35314	0.05552	0.22569

TABLE 5.6: *Designoids (P), errors of the genes (e(g)) and average errors (E(P)) calculated for each of the best individuals obtained in the three experiments.*

On the one hand, for the first target snapshots, the error margins range from a most satisfactory 4.3% ($e(c)$ on P_3^2) to a widely inaccurate 1277% ($e(r)$ on P_2^2). It is interesting to note that the worst errors are all for parameter r (ROUGHNESS). This suggests that it may be the case that r is the least influential in the generation of the images, and indeed a brief experimentation with the Turbulence program reveals that this is the case, at least when combined with these values of $i = 50.5$ and $c = 0.50$. It appears that when i, the initial turbulence, is high, as in this case, a change in r makes little difference, but when i is low, r is far more influential. This agrees with the physical dynamics of fluid flow which the system is modelling – if the fluid is initially perturbed, one can intuitively surmise that the roughness of the pipe will have a lesser effect than when the fluid is initially undisturbed. These observations show that, although interesting, exact approximation of the parameters is not necessarily a good indication as to the similarity (or lack thereof) of two spatio-temporal behaviours. On the other hand, when analysing P^3 case, the parameters have all been approximated to within about 35% of the target with a combined error value $E = 0.22569$, and interestingly, r is now the most accurate approximation ($e(r) = 0.05552$).

Overall, these visual inspections together with the analytical analyses are just a further confirmation of the highly complex, non-linear and stochastic nature of the genotype-phenotype mapping for this inverse problem and why it is difficult to solve.

5.3   Conclusions

Finding the appropriate input values for a CA system which is capable of reconstructing the operational mode of the cells in order to achieve a specific snapshot of automaton

behaviour is a challenging open problem. This chapter contributed a GA based approach for the continuous design optimisation of CAs input parameter values. The strategy of the methodology is the search for a solution in a particular parametric space such that when assigned as input of a CA, the captured spatio-temporal behaviour maximizes the measure of similarity when compared with a specific target.

One of the most remarkable features of the presented approach is its abstract operation, since it is clearly independent of the type of neighbourhood, dimension of the subjacent lattice and the way interactions take place across the CA. In contrast to other approaches, this proposal does not consider any type of introspective analysis since the CA is taken as a black box device and emphasis for solving this inverse problem is given to the quality of the output – which is analysed in terms of information distance (USM) – rather than on its modus operandi. From the algorithmic point of view, this design optimisation of input parameter values by means of artificial evolution contributes to a high abstraction level strategy since it addresses the problem as a whole instead of tackling it with a sequence of strategies corresponding to problem subdivisions.

The experiments have been conducted for Turbulence CA where the search has been performed over a parametric space comprising three different input variables, each of which takes values over a continuous range. The achieved results reveal that a significantly different genotype can, in fact, result in a similar phenotype – yet another illustration of the complex, non-linear nature of the genotype-phenotype-fitness mapping in these systems. All this correlates in fact with what is established as ill possessed characterisation: different inputs could be the causes of a given effect.

CHAPTER 6

An Evolutionary Approach to Cellular Automata Rule Design

Having presented an evolutionary approach to CA input parameter design, this chapter aims to extend the methodology for the automated design optimisation of CAs input rules. In order to perform this task, an invention called Meta-automata CA is introduced. With this cellular machine at hand, the idea behind is to consider the CA as a black box device where the performance of its rules is assessed in terms of information distance between the associated emergent behaviour and a given sample snapshot. In this way, the main concern lays in answering the following: having a CA together with captured spatio-temporal behaviour, what are the set of rules underlying the given effect? The architectural features of the methodology are introduced in the first place. After that, the technical details of the innovative model are presented followed by the experimental data sets and the conducted experiments. The chapter finishes with analyses of the experimental results and conclusions of the applied methodology. The research to be reported in this chapter has resulted in a journal paper published in "Journal of Cellular Automata" [175].

6.1 Architecture

The methodology proposed here employs a GA whose goal is to design the input rules of the Meta-automaton CA. This cellular machine is an innovation that allows partitions over the system's spatial and temporal dynamics. The approach for the automated design of the rules takes place in the following way. The CA is associated with specific input rules and the values of the cells are randomly initialised. The rules are iteratively applied across time resulting in an associated spatio-temporal behaviour to be captured in a visual pattern. After that, the pattern is compared in terms of similarity with a user defined target, process that returns a numerical value that specifies how (dis)similar the evolved pattern is with the target one. Repeating both generation and comparison steps over different set of input rules results in a collection of captured patterns where those better resembling the target are considered more fit. Hence, the better ranked a pattern is, the closer the collection of rules are to those that gave rise to the original (target) pattern.
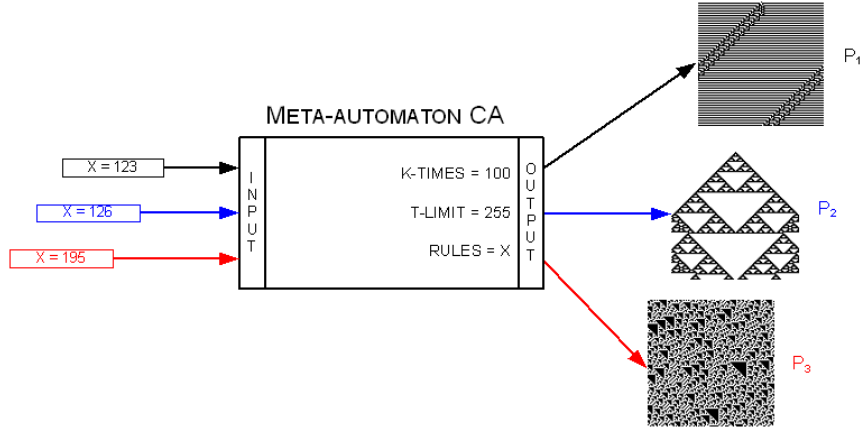
FIGURE 6.1: *Meta-automaton with input parameters K-TIMES, T-LIMIT and RULES. By fixing K-TIMES $= 50$ and T-LIMIT $= 255$, the snapshots P_1, P_2 and P_3 are obtained when RULES is set up to* 123, 126 *and* 195 *respectively.*

For a better understanding of the methodology, consider the Meta-automaton CA, with input parameters K-TIMES and T-LIMIT fixed to 50 and 255 respectively, and the parameter RULES left variable. So, if RULES takes values 123, 126 and 195 in turns, then the associated patterns P_1, P_2 and P_3 will be generated as depicted in Figure 6.1.
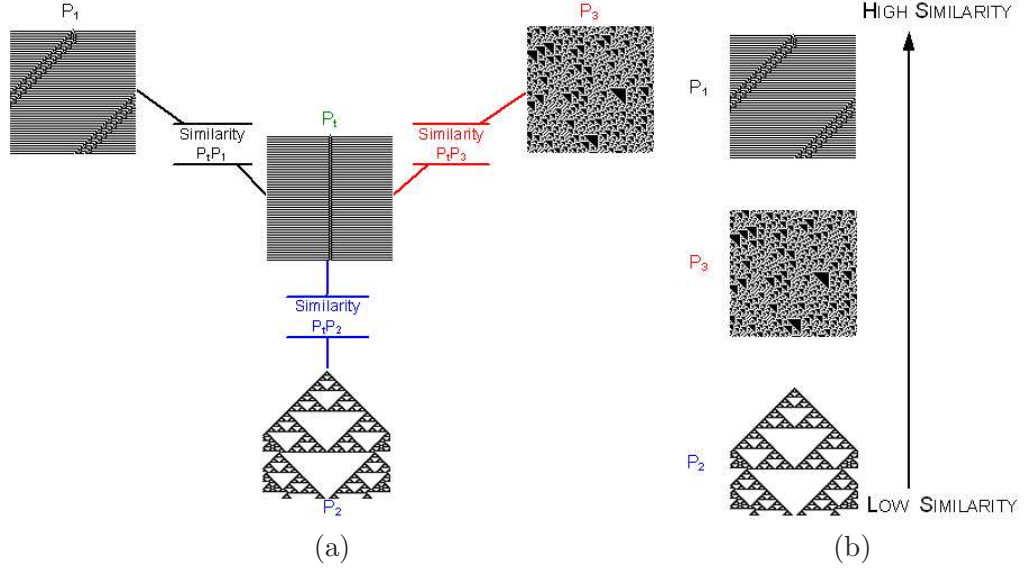
FIGURE 6.2: *Process of comparison in terms of similarity. A target snapshot P_t is compared to each of the three generated patterns P_1, P_2 and P_3 (a). The comparison by similarity produces a ranking where those patterns better resembling the target snapshot are better positioned in alikeness scale (b).*

Consider the pattern P_t depicted in Figure 6.2 (a) as the user defined target. Then, patterns P_1, P_2 and P_3 will be compared in terms of similarity to P_t from where those better resembling the target will be highly ranked. Since P_1 is more similar to P_t than P_3 is to P_t, plus the fact that P_2 is even less similar, the obtained ranking of similarity results in P_1, P_3 and P_2, as Figure 6.2 (b) shows. Generation and comparison together with recombination and mutation operators are repeated for a certain number of generations over a population of individuals which represent sets of input CA rules (see Figure 5.3).

6.1.1 Problem Description

The **Meta-automaton CA** has been implemented with NetLogo. This system is based on a one-dimensional binary CA of radius 1 where the cells are associated with one of the so called "256 elementary rules" [57]. The purpose of this automaton is to show how the change of dynamics along space and time affects the information flow; to understand how rules behave in a given configuration; and how different combinations of rules could affect the complexity of the system. Example patterns of the CA are shown in Figure 6.3.
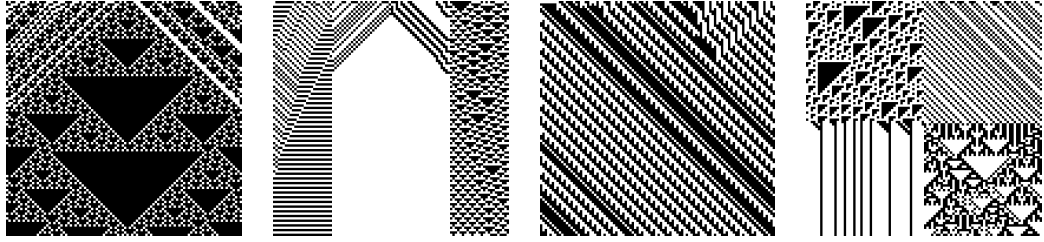
FIGURE 6.3: *Sample snapshots of spatio-temporal patterns from Meta-automaton model.*

The Meta-automaton CA can be seen as an instance of the so called non-uniform CA [176]. In particular, this model allows partitions over the system's spatial and temporal dynamics using two discrete variables: K-TIMES and T-LIMIT – the former indicates that groups of k-consecutive cells are associated to the same rule whilst the latter allows a re-assignation of rules to take place every t time steps.

Table 6.1 summarises the key concepts underlying the problem to be tackled. In this table, **Problem name** is a self-explanatory label describing the problem in general terms; **Instance** describes the particular problem, i.e. the CA description emphasising on the parameters to be designed; **Solution** stands for the type of answers that solve the problem; and **Measure** specifies how the solution performance is scored.

Problem name	Meta-automaton CA rules design
Instance	a one-dimensional CA of radius 1 allowing combinations of rules distributed across space and time to affect the complexity of the system
Solution	a collection of n rules linked to n-groups of k-consecutive cells of the CA which outputs a spatio-temporal behaviour pattern captured in a bitmap image. k is integer taking values 100, 50 and 25
Measure	comparison of the captured CA output against a user defined (target) pattern in terms of similarity by means of USM

TABLE 6.1: *Problem name, instance, solution and measure for the evolutionary design optimisation of Meta-automaton CA rules.*

6.1.2 Population, Genetic Representation and Initialisation

Given that the cells of the CA are associated with elementary rules, a collection of integer numbers for encoding the Meta-automaton CA rules is employed. Hence, an individual of the population is defined as a sequence of n natural numbers, each of them representing an elementary rule which is initially generated with a random value belonging to $[0, 255]$. Formally speaking, the population and its individuals are defined as:

$$Pop = \{Ind_1, Ind_2, \ldots, Ind_m\}$$

$$Ind_j = \{e_1^j, e_2^j, \ldots, e_n^j\}$$

$$\text{where } 0 \le e_i^j \le 255 \tag{6.1}$$

For each of these individuals, evaluation, recombination and mutation operators are applied and repeated for a certain number of generations where those capable of creating patterns similar to the target pattern will survive and is likely to guide the evolutionary process across generations.

6.1.3 Selection Scheme and Genetic Operators

During the evolutionary process, individuals are selected to be parents using roulette wheel selection, offspring are obtained using uniform crossover and mutation is implemented using the Breeder Genetic Algorithm operator. For technical characteristics and features refer to Section 5.1.3 in Chapter 5.

6.1.4 Evaluation Procedure

The evaluation phase consists of two stages: pattern creation and pattern comparison.

Pattern Creation

As previously stated, each individual Ind_j is initialized with a sequene of n integers (rules) ranging from 0 to 255. At this stage, these rules are linked and applied to groups of k-consecutive CA cells giving rise to an associated spatio-temporal behaviour pattern (P_j) to be captured in an image. Three sample individuals initialised with rules 0, 133, 150 and 254 together with its associated spatio-temporal patterns are shown in Figure 6.4.

(a)　　　　　　　(b)　　　　　　　(c)
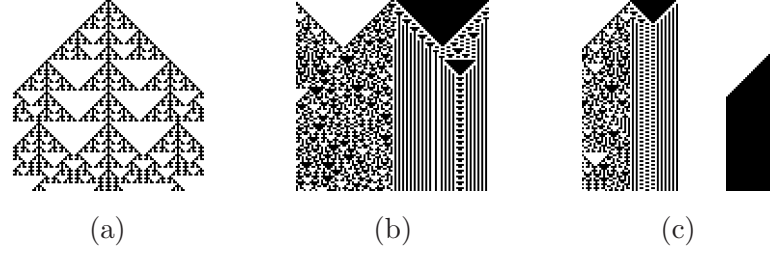
FIGURE 6.4: *Meta-automaton CA spatio-temporal behaviour patterns generated after applying the rules upon individuals: (a) {150}; (b) {150, 133}; (c) {150, 133, 0, 254}.*

Pattern Comparison

After the image capturing the spatio-temporal behaviour pattern is generated, its comparison against a target pattern (T) takes place. In order to perform this task, three arbitrary data sets of target snapshots are defined. The cells of the first data set were associated with the same rule without reassignment across time, i.e. K-TIMES = 100 and T-LIMIT at infinitum. The target patterns and their correspondent rules are depicted in Figure 6.5.

T_1^1　　　　　T_2^1　　　　　T_3^1　　　　　T_4^1　　　　　T_5^1

T_6^1　　　　　T_7^1　　　　　T_8^1　　　　　T_9^1　　　　　T_{10}^1
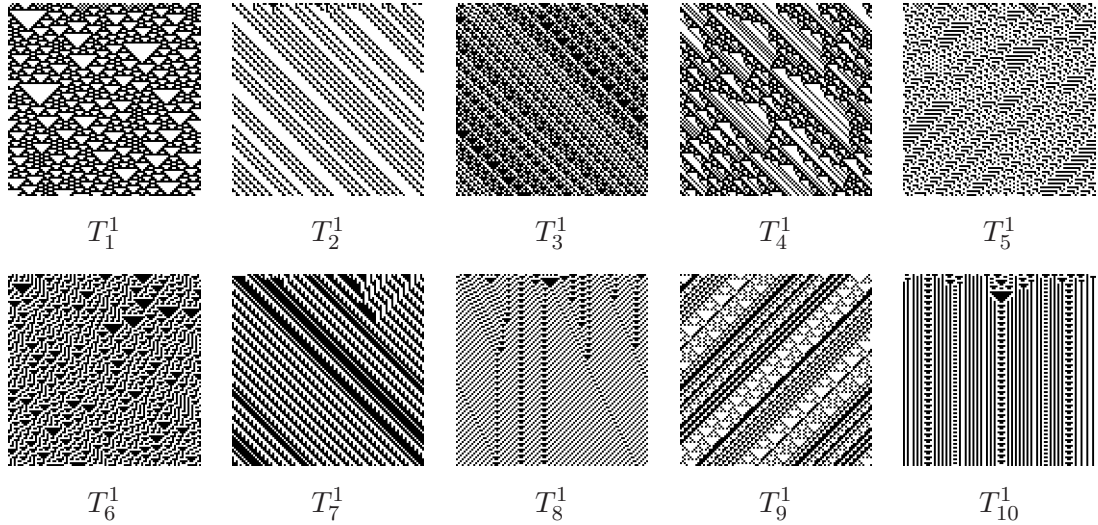
FIGURE 6.5: *Target patterns produced by the Meta-automaton CA using rule 122 for T_1^1, rule 148 for T_2^1, rule 181 for T_3^1, rule 120 for T_4^1, rule 97 for T_5^1, rule 135 for T_6^1, rule 229 for T_7^1, rule 131 for T_8^1, rule 154 for T_9^1 and rule 133 for T_{10}^1.*

For the second data set of target snapshots (Figure 6.6), the spacial dynamics were divided in two, i.e K-TIMES = 50 and $n = 2$. Thus, given two random rules chosen from the pool of 256 rules, the first consecutive 50 cells were associated with one rule and the remaining 50 with another rule. As in the previous data set, the states of the first row were randomly initialised with no reassignment of rules during runtime.

$$T_1^2 \qquad T_2^2 \qquad T_3^2 \qquad T_4^2 \qquad T_5^2$$

$$T_6^2 \qquad T_7^2 \qquad T_8^2 \qquad T_9^2$$
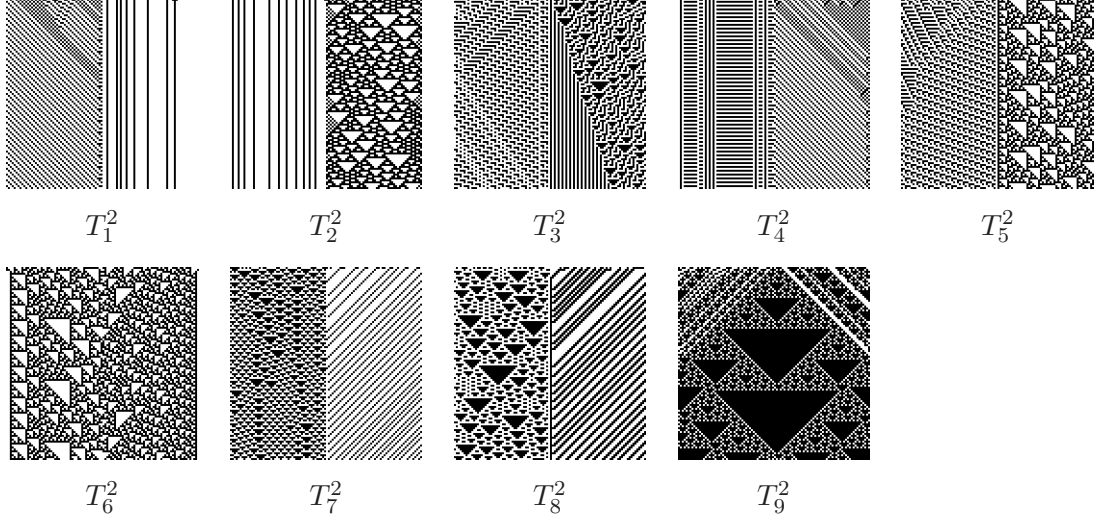
FIGURE 6.6: *Target patterns produced by the Meta-automaton CA changing dynamics over space with two different using rules 177-132 for T_1^2, rules 68-122 for T_2^2, rules 65-135 for T_3^2, rules 5-57 for T_4^2, rules 25-60 for T_5^2, rules 60-102 for T_6^2, rules 147-2 for T_7^2, rules 129-46 for T_8^2 and rules 167-180 for T_9^2.*

In order to provide an even more challenging data set, the spacial dynamics were divided into four in the last set of target patterns, i.e. K-TIMES = 25 and $n = 4$. Thus, cells were divided in groups of 25 consecutive cells to which a randomly selected rule from the pool of 256 was applied. As before, there was no reassignment of rules across time. The patterns for the third data set are shown in Figure 6.7.
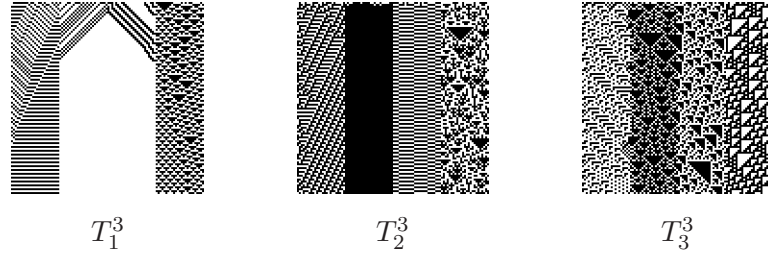
FIGURE 6.7: *Target patterns produced by the Meta-automaton CA changing dynamics over space with rules 49-34-84-147 for T_1^3, rules 61-251-23-165 for T_2^3 and rules 41-183-195-110 for T_3^3.*

Each of the snapshots listed previously is set as a target pattern in separate GA experiments. Thus, during the evaluation stage of an experiment, the fitness function compares each P_j to the current T for similarity using the USM introduced in Chapter 4. This metric returns a numerical representation that is considered as the fitness of each individual, i.e. the value to minimise by the GA. All in all, those individuals generating patterns that are similar to the target are better ranked and become more likely to survive.

Given that for all the generated complex behaviour the first row of cells is randomly initialized, the captured patterns might differ for each initialisation. This influences the comparison for similarity which would certainly give different results for the same individual. For this reason, a reliable estimation of the true fitness is needed and individuals must be evaluated several times. Consequently, each individual is evaluated five times and its fitness is calculated as the average.

6.2 Results

This section aims to present the results of the GA experiments using the parameters summarised in Table 6.2 for each run. For all the experiments the population size, the amount

of generations, the number of times an individual is evaluated, the crossover probability and the mutation probability were fixed. The length of the individuals is given by the amount of input rules, i.e. length n in case of the Meta-automaton CA.

Population Size	Generations	Evaluations	XProb	MProb
20	100	5	0.7	0.3

TABLE 6.2: *GA parameters for Meta-automaton CA evolutionary design optimisation.*

P	e_1^P	T	e_1^T	USM(P, T)	Similarity
P_1^1	122	T_1^1	122	0.993958124	Correct
P_2^1	6	T_2^1	148	1.042744644	Mirror
P_3^1	181	T_3^1	181	0.984504855	Correct
P_4^1	106	T_4^1	120	0.983119009	Mirror
P_5^1	97	T_5^1	97	0.985429776	Correct
P_6^1	195	T_6^1	135	0.976343879	None
P_7^1	195	T_7^1	229	1.048922986	None
P_8^1	131	T_8^1	131	1.00218998	Correct
P_9^1	169	T_9^1	154	0.987069886	Low
P_{10}^1	133	T_{10}^1	133	0.950053315	Correct

TABLE 6.3: *Results for the first group of Meta-automaton CA snapshots. Best designoids (P), evolved rules (e_1^P), targets (T) with their creational rules (e_1^T), fitness (USM(P,T)) and similarity levels (Similarity).*

Table 6.3 lists the results of the experiments conducted over the first data set shown in Section 6.1.4. In this table, the labels in column **P** refers to the obtained designoids whilst e_1^P itemizes the genes of the fittest individuals. In the following two columns, **T** refers to the target snapshots followed by the creation rules listed under e_1^T. In the last pair of columns, the fitness values appear below **USM** and **Similarity** classifies the degree of visual likeness between an obtained designoid and a target snapshot.

As the last column of Table 6.3 shows, the GA has evolved the same rules that produce the target snapshots in five out of ten experiments. This is the case of T_1^1, T_3^1, T_5^1, T_8^1 and T_{10}^1 in which the evolved rules achieve designoids with high level of similarity. Further analyses reveal that two extra experiments have evolved rules which are capable of producing designoids mirroring the targets. This is the case when experimenting with T_2^1 and T_4^1, as depicted by the plots in Figure 6.8, where the diagonal streams have been successfully reproduced although in opposite directions.

T_2^1 P_2^1 T_4^1 P_4^1
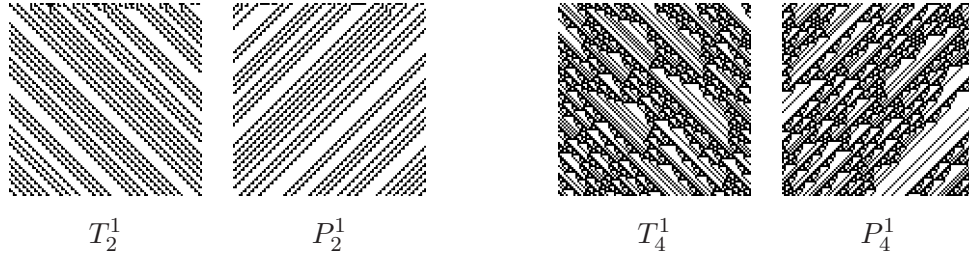
FIGURE 6.8: *Target patterns (T_2^1 and T_4^1) and designoids (P_2^1 and P_4^1) resulting from the first data set of Meta-automaton CA rules design problem. The evolved rules generate spatio-temporal patterns which mirror the target snapshots.*

Nevertheless, experiments over T_9^1 resulted in a significant rule capable of generating a low level resemblant designoid, i.e. an evolved pattern with few features in common

to the target snapshot. In this particular case, from a visual comparison between the target snapshot and the evolved pattern (Figure 6.9) we can argue that the diagonal black strips together with their orientation appearing in P_9^1 were very well captured by the USM.

$$T_9^1 \qquad\qquad P_9^1$$
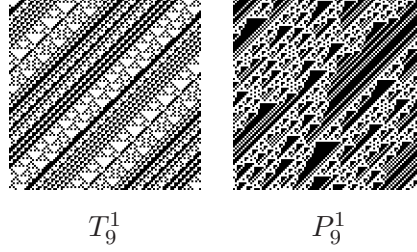
FIGURE 6.9: *Target pattern (T_9^1) and its designoid (P_9^1) resulting from the first data set of Meta-automaton CA rules design problem. The evolved rule generates a spatio-temporal pattern with low level degree of similarity although it is capable of capturing some underlying black diagonal structures appearing in the target.*

Unfortunately, none of the previous analyses apply to the findings when T_6^1 and T_7^1 were set up as target snapshots. In these experiments, neither mirrored patterns nor similar structures with different orientations were achieved by the evolved individuals as depicted in the designoids of Figure 6.10.

$$T_6^1 \qquad\qquad T_7^1 \qquad\qquad P_6^1/P_7^1$$
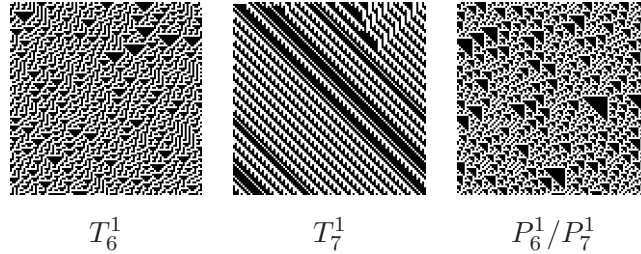
FIGURE 6.10: *Target patterns (T_6^1 and T_7^1) and designoids (P_6^1/P_7^1) from the first data set of Meta-automaton CA rules design problem. The evolved rules are not able to generate spatio-temporal patterns with any particular feature appearing in the targets.*

The findings for the second data set indicate that none of the evolved individuals have achieved the target rules (see Table 6.4). However, in some cases, the GA evolved rules capable of generating spatio-temporal patterns which mirror the target (see Figure 6.11).

P	e_1^P	e_2^P	T	e_1^T	e_2^T	USM(P, T)	Similarity
P_1^2	164	177	T_1^2	177	132	0.818578016	Mirror
P_2^2	122	100	T_2^2	68	122	0.885830497	Mirror
P_3^2	215	146	T_3^2	65	135	0.948304844	None
P_4^2	115	192	T_4^2	5	57	0.870995252	None
P_5^2	26	125	T_5^2	25	60	0.96081944	None
P_6^2	183	20	T_6^2	60	102	0.964207074	None
P_7^2	130	147	T_7^2	147	2	0.905361748	Mirror
P_8^2	126	16	T_8^2	129	46	0.958283213	Captured
P_9^2	91	167	T_9^2	167	180	0.993560531	Captured

TABLE 6.4: *Results for the second group of Meta-automaton CA snapshots. Best designoids (P), evolved rules (e_i^P), targets (T) with their creational rules (e_i^T), fitness (USM(P,T)) and similarity levels (Similarity).*

Analysing the composition of these three particular individuals (P_1^2, P_2^2 and P_7^2 in Table 6.4) it is interesting to observe that in all the cases only one of the rules has been successfully evolved, i.e. the same rule as that in the target individual. It is interesting to note that this successful rule appears in the opposite position to where it is originally located in the target individual. For instance, rule 177 is associated with the first group of CA cells when generating the target snapshot (T_1^2) and with the second group of CA cells when generating the designoid (P_1^2).
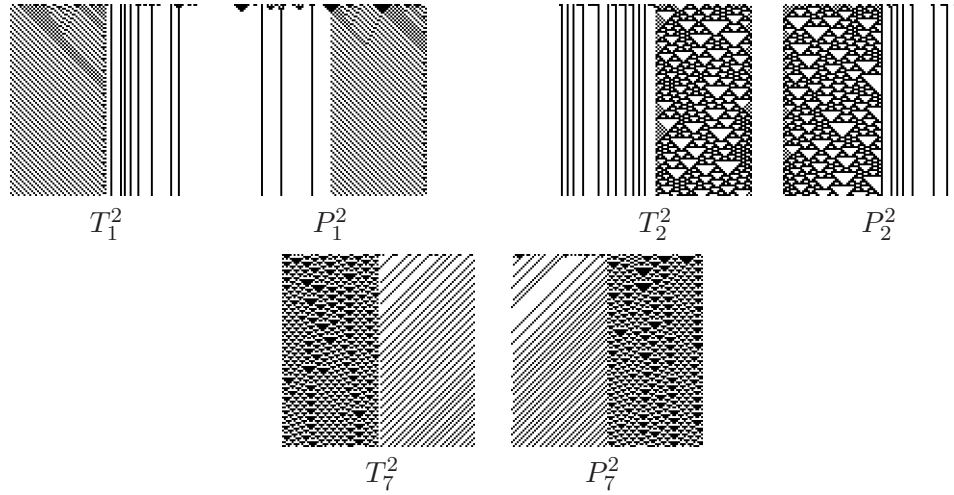
FIGURE 6.11: *Target patterns and designoids using two rules: T_1^2 and its mirror P_1^2, T_2^2 and its mirror P_2^2, and T_7^2 and its mirror P_7^2.*

Even more interesting is what occurs in the second case since the unsuccessful rule emerges as near equivalent to the one in the target. In this context, a rule e_i is considered "equivalent" to a rule e_j if the spatio-temporal behaviour obtained by the application of e_i generates a similar spatio-temporal behaviour to the one obtained with e_j. For instance, the rule 132 associated with the second group of cells when generating the target snapshot T_1^2 and the rule 164 associated with the first group when generating the designoid P_1^2 achieve a near-identical pattern. In order to perform a further analysis, the Hamming distance among the binary representation of these rules is computed. All the calculations indicate a difference in one position, hence supporting the observed similarity. Note that the encoding of the individuals is not binary, and that given two binaries with Hamming distance of 1 this does not always map to similar spatio-temporal behaviours in the Meta-automaton CA. This last analysis is summarised in Table 6.5.

P	e_i^P	$(e_i^P)_2$	**T**	e_i^T	$(e_i^T)_2$	$D_H((e_i^P)_2, (e_i^T)_2)$
P_1^2	164	10100100	T_1^2	132	10000100	1
P_2^2	100	11000100	T_2^2	68	01000100	1
P_7^2	130	10000010	T_7^2	2	00000010	1

TABLE 6.5: *Analysis of the evolved rules. Designoids (P), evolved rules (e_i^P) and binary representations (($e_i^P)_2$), targets (T) with their creational rules (e_i^T) and binary representations (($e_i^T)_2$), and Hamming distances ($D_H((e_i^P)_2, (e_i^T)_2)$).*

Still interesting are the findings when experimenting with T_8^2. As shown in Figure 6.12, the achieved designoid resembles the target in the following way. Structurally speaking, the patterns appearing at the left side have been perfectly achieved although they complement one another. That is, the light triangles in the target snapshot become dark in the designoid and vice versa.

$$T_8^2 \qquad P_8^2$$
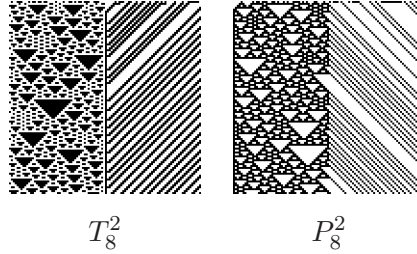
FIGURE 6.12: *Captured similarities for a Meta-automata pattern using two rules: target pattern T_8^2 and its designoid P_8^2.*

Moreover , the diagonal wavy lines at the right side of the target snapshot have been mirrored in the designoid although much thinner. Performing a further analysis (see Table 6.6), the Hamming distance between the binary representation of the first rules of

T_8^2 and P_8^2 indicates a difference in 8 positions, hence supporting the established comple-
ment as seen in previous analysis. Note that complementary rules do not always generate
complementary patterns.

P	e_1^P	$(e_1^P)_2$	T	e_1^T	$(e_1^T)_2$	$D_H((e_1^P)_2, (e_1^T)_2)$
P_8^2	126	01111110	T_8^2	129	10000001	8

TABLE 6.6: *Analysis of the evolved rules. Designoid (P), evolved rule (e_1^P) and binary representation ($(e_1^P)_2$), target (T) with its creational rule (e_1^T) and binary representation ($(e_1^T)_2$), and Hamming distance ($D_H((e_i^P)_2, (e_i^T)_2)$).*

Table 6.7 summarises the results of the third data set. As the evolved rules reveals,
none of them matches the rules of the targets.

P	e_1^P	e_2^P	e_3^P	e_4^P	T	e_1^T	e_2^T	e_3^T	e_4^T	USM(P, T)	Similarity
P_1^3	73	141	188	230	T_1^3	49	34	84	147	0.907788419	Captured
P_2^3	38	140	105	234	T_2^3	61	251	23	165	0.917228868	Captured
P_3^3	61	120	146	196	T_3^3	41	183	195	110	0.940763235	Captured

TABLE 6.7: *Results for the third group of Meta-automaton CA snapshots. Best designoids (P), evolved rules (e_i^P), targets (T) with their creational rules (e_i^T), fitness (USM(P,T)) and similarity levels (Similarity).*

However, a visual inspection of the obtained designoid shown in Figure 6.13 sup-
ports the idea that some relevant features from the target snapshots were captured. For
instance, it is interesting to note that two rules capable of producing the inverted "V-
shape", appearing in the middle of the target snapshot, were discovered although with

inverted colouring and in a different position.

$$T_3^3 \qquad\qquad P_3^3$$
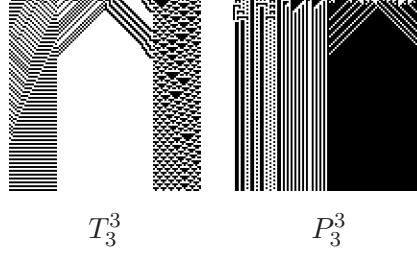
FIGURE 6.13: *Target snapshot T_3^3 and its resulting designoid P_3^3 from the third data set of Meta-automaton CA rules design problem. Two achieved rules generate an inverted "V-shape" with opposite colouring and located at a different position.*

In addition, the last two designoids also display some captured features generated by the evolved individuals. For instance, a rule capable of generating the second strip of T_2^3 (Figure 6.14) was evolved by the GA as shown in the fourth position of the associated designoid (P_2^3). Nevertheless, some of the structures emerging in the chaotic pattern of the last strip of the same target have been simulated, although not captured, in the spatio-temporal behaviour generated by the third rule of the evolved individual.

$$T_2^3 \qquad\qquad P_2^3 \qquad\qquad\qquad T_3^3 \qquad\qquad P_3^3$$

FIGURE 6.14: *Target snapshots together with their resulting designoids from the third data set of Meta-automaton CA rules design problem. Rules capable of generating the black strip and the last line of emergent structures in T_2^3 are captured at the fourth and third positions in P_2^3. On the other hand, the white triangular entities appearing in the third strip of P_3^3 resemble the black structures in the second frame of T_3^3.*

Finally, consider the visual similarities found among the target snapshot T_3^3 and its associated designoid P_3^3 exposed in Figure 6.14. These two captured patterns reveal that the triangular entities appearing in the third strip of the designoid accurately resemble the ones in the second frame of the target although the colour of the dark triangles become lighter in the designoid and vice versa.

6.3 Conclusions

Finding the appropriate input for a CA system capable of reconstructing the operational mode of the cells in order to achieve a specific snapshot of automaton behaviour is a challenging open problem. This chapter contributed an innovative CA model and a GA based approach for the evolutionary design optimisation of CA rules. The Meta-automaton CA – an invention of the author – is based on a one-dimensional binary CA of radius 1 that allows partitions over the system's spatial and temporal dynamics. The strategy of the proposed methodology is the searching for the appropriate combination of rules so that when assigned onto a lattice partition, the captured emergent behaviour maximizes the measure of similarity when compared with a specific snapshot.

Analyses of the results of the experiments revealed that a number of different rules can achieve very similar spatio-temporal behaviour. Hence, as also seen in the results for the Turbulence CA snapshots (Chapter 5), a significantly different genotype can, in fact, result in a similar phenotype – yet another illustration of the complex, non-linear nature of the genotype-phenotype-fitness mapping in these systems. All this correlates in fact with

what is established as ill-possessed characterisation: different inputs could be the causes of a given effect.

In addition, it is evident from these experiments that, although the USM's information distance-based metric works well in many cases, it has a number of shortcomings. As illustrated by the last set of experiments, one of the most obvious drawbacks is its blindness to complementary snapshots. Similarly, the USM does not differentiate between mirror patterns – the actual information content of an image is identical to its mirror image. Therefore, it is a logical progression to extend the fitness function using an additional measurement such as Hamming distance or an entropy analysis. In the first case, using a measure of Hamming distance involves calculating the colour difference between target and designoid on a pixel-by-pixel basis. Alternatively, an image's entropy is an estimate of the distance between two images based on calculating the frequency of the appearance of different sub-blocks or fragments of the images. In either case, we consider that the fitness function would need to be extended to either a multi-objective setting or a single weighted function.

CHAPTER 7

An Evolutionary Approach to Self-Assembly Wang Tiles Design

Finding the appropriate combination of autonomous entities capable of arranging themselves together in order to perform a common task is a challenging open problem for the design and development of distributed cooperative systems. This chapter reports on an evolutionary algorithm approach for the design optimisation of self-assembly Wang tiles. The following section summarises Wang tiles as a model of physical and biological systems. After that, the GA is explained in detail, in particular its population, genetic representation, initialisation phase, evaluation process, selection scheme and genetic operators. The chapter also includes the conducted experiments where the technical details of the utilised models and the experimental data sets will be presented. Finally, the analysed results and conclusions complete this experimental part of the dissertation. The research to be reported in this chapter has been published in "Proceedings of the 7th International Conference on Artificial Evolution" [177], "2005 IEEE Congress on Evolutionary Computation" [178] and "2007 IEEE Congress on Evolutionary Computation" [179].

7.1 Architecture

The approach to be proposed employs a GA whose goal is to design a collection of Wang tiles that will self-assemble into a specific target shape in the following way. A set of tiles is dropped into a simulation environment comprising a two-dimensional lattice, a strength matrix and a glue function. In this environment, tiles are randomly located on the lattice where they perform a random walk for a certain amount of time. Tiles eventually collide and, according to the strength of the colliding sides, they either stick to one another or bounce off. Once the simulation finishes, the scattered tiles, together with the aggregates found in the lattice, define a kind of layout called *configuration*. It is in this configuration where a user defined (target) shape is surveyed, after which a value is returned indicating how successful the tiles self-assembled in that shape. The repetition of this two-step process, i.e. tiles simulation and search for the target shape over different sets of tiles, gives a collection of configurations in which the ones receiving large values of success attribute the associated tile sets a high rank. Hence, the more successful a configuration, the better designed is the collection of associated tiles.

As an example, consider the three sets of self-assembly Wang tiles S_1, S_2 and S_3 shown in Figure 7.1. Each of the sets is dropped in turn into the simulator where tiles perform a random walk and interact with one another during a certain period of time. Thus, each simulation generates an associated configuration $Conf_1$, $Conf_2$ and $Conf_3$ as Figure 7.1 shows. Following the example, if a square of 5×5 tiles is considered as the target shape, then S_1 ranks first, S_2 ranks second and S_3 follows since the aggregates achieved by S_1 are the most similar in shape to a square of 5×5 tiles.
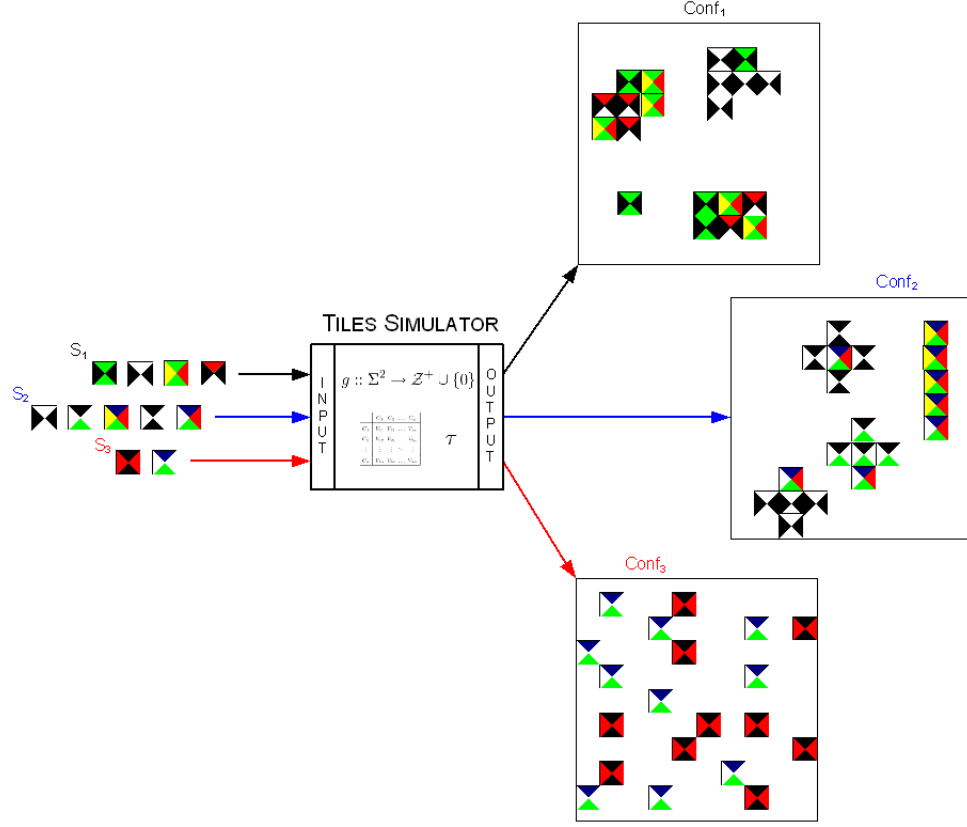
FIGURE 7.1: *Three different self-assembly Wang tiles sets (S_1, S_2 and S_3) are dropped in turn into a simulator comprising a lattice, glue function and strength matrix. After performing random walks and interacting with one another, tiles embodying self-assembled aggregates define a layout called configuration ($Conf_1$, $Conf_2$ and $Conf_3$).*

To sum up, the described process together with recombination and mutation operators is iteratively applied, for a certain number of generations, to a population of tile sets where those capable of self-assembling in aggregates similar in shape to the user defined target will survive and guide the evolutionary process across generations. Unlike many problems for which the fitness function is a simple mapping from the genotype encoding a solution to its fitness value, in this case there is a complex *genotype-phenotype-fitness* mapping that makes the evolutionary process a complex system.

Figure 7.2 depicts a chart flow of the extended GA where initialisation corresponds to the process when the individuals comprising the set of tiles are created; evaluation is the stage involving simulation and self-assembly assessment; and crossover and mutation shows the process when one-point recombination and gene-wise mutation operators are mapped across the entire population.
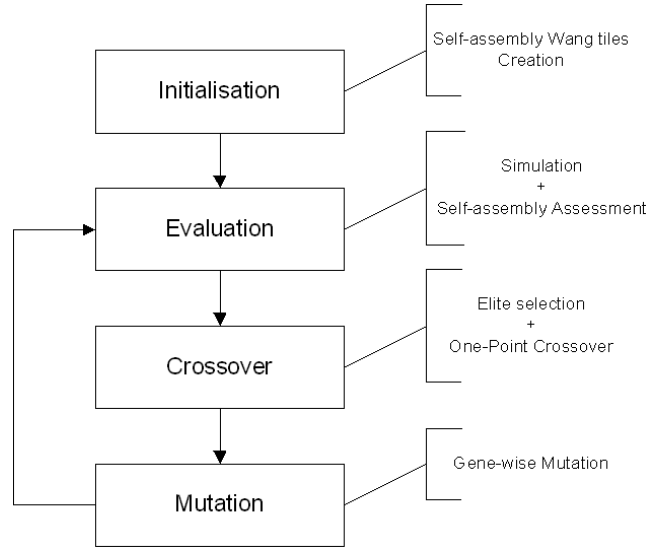
FIGURE 7.2: *Flowchart of the extended GA for evolving self-assembly Wang Tiles.*

7.1.1 Problem Description

The schematic problem description for the design optimisation of self-assembly Wang tiles is summarised in Table 7.1. In this table, **Problem name** is a self-explanatory label shortly describing what the problem is about; **Instance** describes the particular problem to be addressed; **Solution** stands for the type of answers that solve the problem and **Measure** indicates how the solution performance is scored.

Problem Name	Self-assembly Wang tiles design optimisation
Instance	a Wang tiles system comprising: - a two-dimensional square lattice where tiles perform random walks, rotate and interact with one another with either a deterministic or a probabilistic stickiness criteria - a symmetric matrix encoding the strength among glue types - an interaction function that evaluates a glue strength against the kinetic energy of the system
Solution	a collection of self-assembly Wang tiles
Measure	comparison between a user defined shape and the aggregations produced by the solution's self-assembly process

TABLE 7.1: *Problem name, instance, solution and measure for the automated design optimisation of self-assembly Wang tiles.*

7.1.2 Population, Genetic Representation and Initialisation

An individual of the population is defined as a collection of self-assembly Wang tile families. That is, each individual's chromosome is a class of tile that can be instantiated with several identical copies. Since tiles are square in shape, each tile family is arbitrarily initialized with four randomly chosen glue types to be assigned to each of its edges. A formal definition for population (Pop) and its variable length individuals (Ind_i) to be evolved in the evolutionary design optimisation of self-assembly Wang tiles is given as:

$$Pop = \{Ind_1, Ind_2, \ldots, Ind_n\}$$

$$Ind_i = \{T_1^i, T_2^i, \ldots, T_{k_i}^i\}$$

$$T_j^i = \{t | t = (C_0, C_1, C_2, C_3)\}$$

$$\text{where } 1 \leq k_i \leq 10$$

$$C_0, C_1, C_2, C_3 \in \Sigma \tag{7.1}$$

The elements defined by Σ are glue types labelling the edges associated with a tile. The GA keeps a population of n individuals each of them encoding k_i tile families (T_j^i) up to a maximum length of 10. As it is not known a priori how many tile families are needed to encode a target shape, individuals are varying in length; $|Ind_i|$ could be different from $|Ind_j|$ for $1 \leq i, j \leq n$.

7.1.3   Evaluation Procedure

The evaluation phase consists of two stages: tiles simulation and self-assembly assessment.

Simulation

During the first stage, tiles are placed into a simulator composed of a two-dimensional square lattice and a *glue function*. In particular, each lattice site is capable of holding only one tile at a time. Essentially, for each tile family T_j^i encoded by an individual Ind_i, an equal number of tile instances drawn from the family is placed into an empty position in the lattice. After that, tiles drift, sticking or bouncing until the simulation runs its course.

In order to study how different dynamics affect the designability problem, four increasingly richer simulation environments classified according to the way tiles interact with each other are defined:

Model 1. When two tiles reach adjacent locations, the glue function evaluates the interaction between the glue types at touching edges using a symmetric matrix M of $\alpha \times \alpha$ glue strengths as shown in Table 7.2. If the resultant value is greater than the temperature τ (a *strong* interaction) both tiles self-assemble and remain in their locations; otherwise (a *weak* interaction occurs) they do not assemble and can eventually move apart. For example, Figure 7.3 shows an example with an interaction matrix of $\alpha = 2$ and $\tau = 4$. As it is depicted, the only way two tiles self-assemble is when black edges become adjacent as any other type of interaction is smaller than τ.

	C_1	C_2	$\ldots$	C_α
C_1	v_{11}	v_{12}	$\ldots$	$v_{1\alpha}$
C_2	v_{12}	v_{22}	$\ldots$	$v_{2\alpha}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
C_α	$v_{1\alpha}$	$v_{2\alpha}$	$\ldots$	$v_{\alpha\alpha}$

TABLE 7.2: *A symmetric matrix M of $\alpha \times \alpha$ encoding strengths v_{ij} between two glue types C_i and C_j for $1 \leq i, j \leq \alpha$.*

Model 2. In this case, if two tiles become adjacent, they are evaluated as in the previous model. However, the difference with **Model 1** is the introduction of a probabilistic rather than deterministic stickiness criteria: ρ per tile. For example, in Figure 7.4, tiles self-assemble as the sum of interactions is 6. In this case, the corner tile contributes with a strong interaction and remains in its position. In contrast, side tiles might still move as their
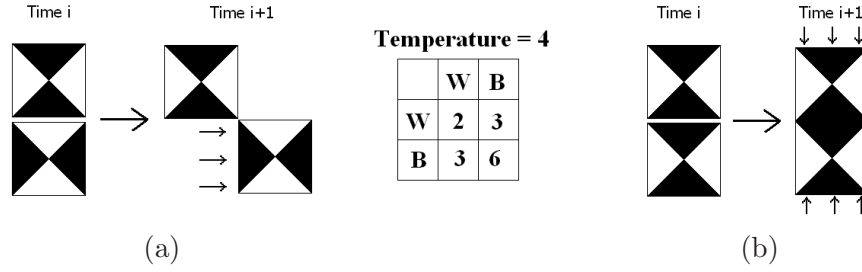
FIGURE 7.3: *Interaction among two tiles with a symmetric matrix of $\alpha = 2$ and $\tau = 4$. (a) The glue function evaluates the interaction between glue types at colliding edges at $Time\ i$ resulting in a weak interaction that keeps tiles on the move afterwards. (b) The glue function evaluates the interaction between glue types when collision takes place at $Time\ i$ resulting in a strong interaction that makes tiles self-assemble and stand still henceforth.*

interactions are weak. The concept of weak and strong interactions has been recognised as weak and strong bonds in [73]. Equation 7.2 defines how the probabilistic stickiness criteria ρ is updated for a given tile t. At the beginning, the probability of being moved is set to 1 and the tile is free to move on the lattice. As a tile interacts with others, ρ either decreases when t self-assembles to another tile or increases if there is a tile detaching from t. In any case, the calculation is performed in terms of the interaction value given by each of the tile edges, its neighbours and τ.
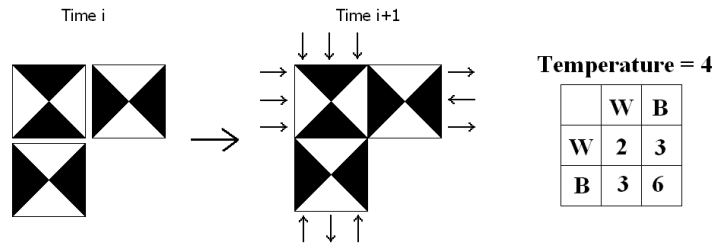
FIGURE 7.4: *Interaction among three tiles with a symmetric matrix of $\alpha = 2$ and $\tau = 4$. At $Time\ i$, the sum of interactions is 6 and the corner tile keeps the side tiles assembled although side tiles might still move at $Time\ i + 1$ since their interactions are weaker.*

$$\rho = Max\left(0, 1 - \sum_{j=1}^{4} contact(t, t_j).\frac{strength(t, t_j)}{\tau}\right)$$

$$t_j = \begin{cases} t_1 & \text{is the northern neighbour of } t \\ t_2 & \text{is the eastern neighbour of } t \\ t_3 & \text{is the southern neighbour of } t \\ t_4 & \text{is the western neighbour of } t \end{cases}$$

$$contact(t, t_j) = \begin{cases} 1 & \text{if } t \text{ is adjacent to } t_j \\ 0 & \text{otherwise} \end{cases}$$

$$strength(t, t_j) = \begin{cases} M[t^n, t_j^s] & \text{if } j = 1 \\ M[t^e, t_j^w] & \text{if } j = 2 \\ M[t^s, t_j^n] & \text{if } j = 3 \\ M[t^w, t_j^e] & \text{if } j = 4 \end{cases}$$

t^i is a tile edge, e.g. the north edge is t^n (7.2)

When a tile contributing with a weak interaction moves, a different arrangement of tiles appears and ρ is re-computed as interactions change. Figure 7.5 shows four alternatives in which the previous self-assembled structure could be modified. In the case of (a) and (d) the side tiles have moved away leaving a weak interaction among the remaining tiles and consequently disassembles the whole structure afterwards. Alternatively, in (b) and (c), the sum of interactions is either maintained or greater than the original arrangement. Nevertheless, both (b) and (c) show that with this model it is also possible to spin the whole structure.
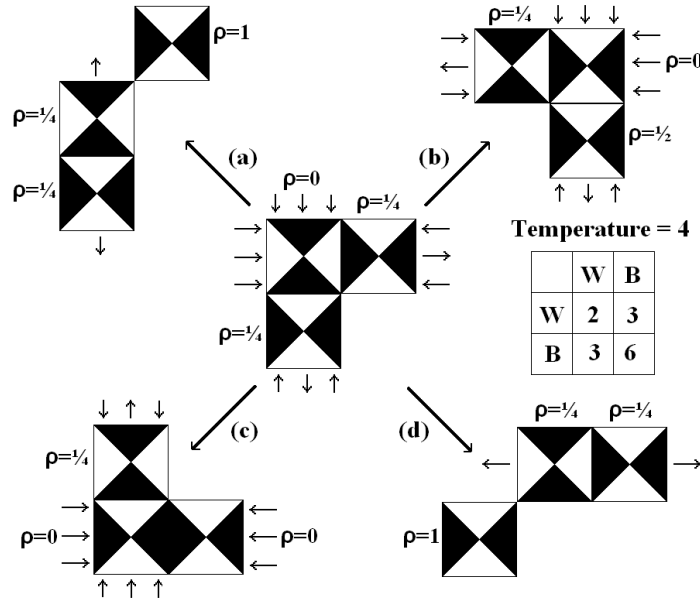
FIGURE 7.5: *Four alternatives in which a self-assembled structure composed by three tiles could be modified. Each tile re-calculates its ρ as interactions with other tiles change.*

In order to make the model richer, tiles are also considered as entities capable of rotating. That is, once a tile moves to a new position in the lattice, a random number is obtained. If it is less than zero then the tile rotates to the left, or else it rotates to the right. So, considering the orientation of the grey edge depicted in Figure 7.6, a tile is able to rotate 90 degrees clockwise or counter clockwise. Thereafter, the notation $C_1C_2C_3C_2$ encodes a tile by its glue types starting from the top edge going clockwise.
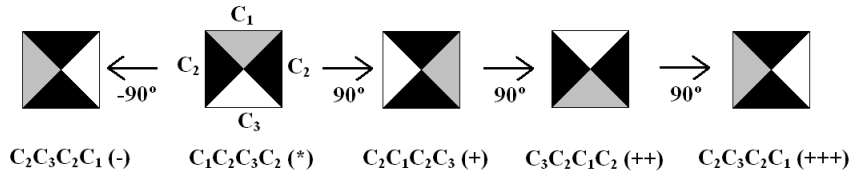
FIGURE 7.6: *Tile rotating clockwise or counter clockwise: + (-) indicates the amount of clockwise (counter clockwise) rotations of a tile marked with *. Notation $C_1C_2C_3C_2$ encodes a tile by its glue types starting from the top going clockwise.*

Hence, **Model 3** and **Model 4** are defined as the combination of the two previous models with rotation capability, i.e. tiles with deterministic criteria plus rotation and tiles with probabilistic criteria plus rotation. Figure 7.7 shows the hierarchical organisation of the models. **Model 2** results from extending **Model 1** with probabilistic criteria (ρ) whilst **Model 3** from extending **Model 1** with rotation ($\Diamond$). Finally, **Model 4** integrates both probabilistic stickiness criteria and rotation ($\rho \Diamond$).
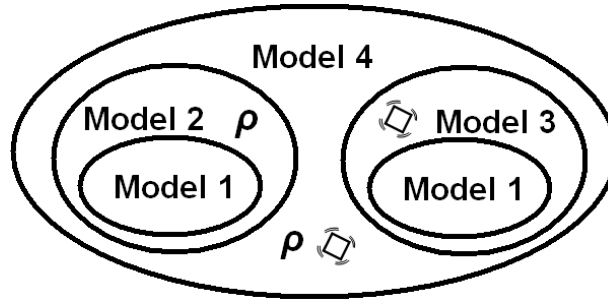
FIGURE 7.7: *Hierarchical organisation of the models.*

Self-Assembly Assessment

Having chosen a model, tiles are placed into the simulation environment and the simulation runs for a fixed number of time steps. Once the simulation stops, the final configuration is evaluated in terms of the similarity between the target shape and each of the self-assembled aggregations contained within the lattice. This assessment at phenotype level is assigned to the genotype as its associated fitness value.

A further complication that must be taken into consideration when measuring the fitness of an individual is that the self-assembly Wang tiles simulator is a stochastic process and for that reason individuals must be evaluated several times as to be able to obtain a

reliable estimation of its true fitness. Consequently, each individual is evaluated several times and its fitness is finally calculated as the average.

Three different approaches have been employed in turn in order to compute the individual fitness. The first one is a lattice scanning algorithm that searches within the individual's configuration for the region that best matches the user defined target. The second approach uses the USM as has been done for CAs. Finally, the third methodology utilises a morphological image analysis method based on the so called Minkowski functionals [157] introduced in Chapter 4.

7.1.4   Selection Scheme and Genetic Operators

In this approach, single individual elitism was employed. In addition, the traditional genetic operators one-point crossover and gene-wise mutation were chosen as part of the selection-recombination scheme. As the employed elitist strategy forces the population to preserve the best chromosome for the next generation, the highest ranked set of Wang tile families is chosen as elite according to its computed fitness value and passes to the following generation. The rest remains as a mating pool from where the new generation will be drawn.

The one-point crossover swaps the genetic material of paired individuals selected without replacement from the mating pool at a given crossing site. For instance, consider $Ind_e = \{T_1^e, T_2^e, \ldots, T_m^e\}$ and $Ind_f = \{T_1^f, T_2^f, \ldots, T_l^f\}$ as the two individuals chosen by roulette-wheel selection with length m and l respectively. Then the recombination operator picks the shortest individual followed by a random cutting point *between* its tiles. After that, the same crossing point is set in the longest individual and the chromosomic recombination takes place. Without loss of generality, if $m < l$ and the cutting point is j, such

that $1 \leq j < m$, then the following two offspring are created:

$$Ind_u = \{T_1^e, T_2^e, T_3^e, \ldots, T_j^e, T_{j+1}^f, T_{j+2}^f, \ldots, T_l^f\}$$

$$Ind_v = \{T_1^f, T_2^f, T_3^f, \ldots, T_j^f, T_{j+1}^e, T_{j+2}^e, \ldots, T_m^e\}$$

Once the set of offspring is completed, the gene-wise mutation operator is applied across the whole collection. In this process, the operator takes an individual and for each of its tile families one or more glue types is likely to be changed due to a *mutation probability* value. For instance, let $Ind_h = \{T_1^h, T_2^h, T_3^h, \ldots, T_p^h\}$ be an offspring and $T_j^h = (C_1, C_2, C_3, C_4)$ a tile family of Ind_h. If C_2 is likely to be mutated and C_x a randomly chosen glue type to replace C_2, then $T_j^h = (C_1, C_x, C_3, C_4)$. The GA is a generational GA, hence the set of offspring plus the elite individual becomes the current population after performing the mutation phase; the artificial evolution process continues afterwards.

7.2 Evaluation Method 1 – Lattice Scanning Approach

This section aims to report on the first methodology employed to evaluate the individuals of the GA used for the design optimisation of self-assembly Wang tiles. In this strategy, the performance of an individual is given in terms of the region of the lattice that best matches with the user defined shape which is a solid square of tiles.

7.2.1 Architecture

Based on our previous findings [177], the individuals of the initial population are created with three types of genes: random genes, genes that promote the self-assembly of columns

and genes that promote self-assembly of rows. The first archetype comprises tile families randomly generated whilst the remaining are hand-crafted ones such that their instances self-assemble in either columns or rows as depicted in Figure 7.8 and Figure 7.9 respectively. In particular, 10% of the individuals were column builders, another 10% of the individuals were row builders whilst the rest were left as randomly generated.
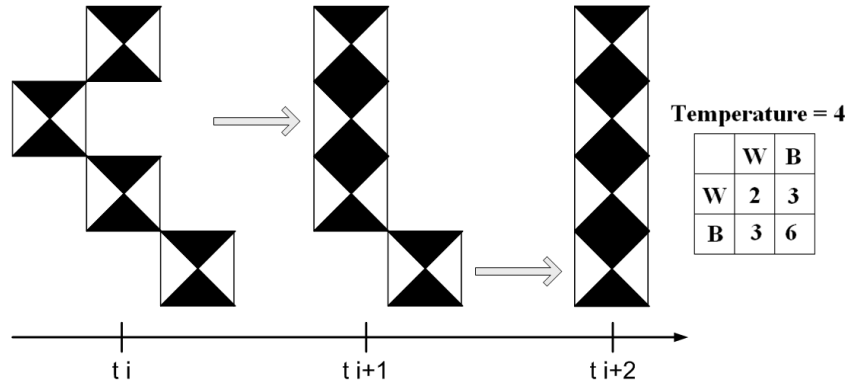
FIGURE 7.8: *Three snapshots across time of a simulation running under* **Model 1** *with* $\tau = 4$ *and a symmetric matrix of* $\alpha = 2$. *The tiles self-assemble promoting column aggregates.*
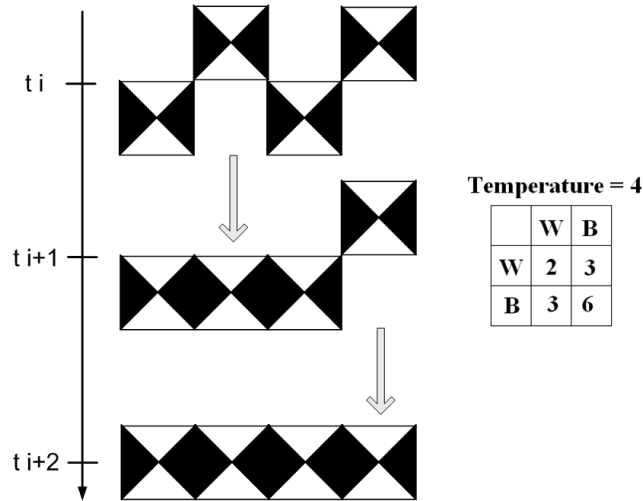
FIGURE 7.9: *Three snapshots across time of a simulation running under* **Model 1** *with* $\tau = 4$ *and a symmetric matrix of* $\alpha = 2$. *The tiles self-assemble promoting row aggregates.*

After initialisation, the population is subject to a series of generations comprising evaluation, crossover and mutation. It is then in the evaluation phase where instances of the Wang tile families are dropped into a simulator executing **Model 1** for 300 time steps after which it is necessary to assess the performance of the self-assembly process. For this purpose, the target shape is exhaustively sought across the configuration. That is, the procedure quantifies the number of tiles filling areas with the dimension of the target shape and the greatest quantity is returned as the fitness of the individual, value to maximise by the GA. Figure 7.10 shows a scanning example going from top left to bottom right on a final lattice configuration. As the self-assembly simulator executes a stochastic process, each individual is repeatedly evaluated and its fitness is the average of the maximum Hamming distances thus obtained.

7.2.2 Experiments

A number of eight experiments were run. For all of them the population size, the maximum length of the individuals, crossover and mutation probabilities, the amount of generations and the quantity of times each individual is evaluated were fixed. All these parameters related to the GA are sown in Table 7.3.

Population Size	k	Generations	Evaluations	XProb	MProb
100	10	100	20	0.3	0.01

TABLE 7.3: *Experiment parameters: population size, maximum length of the individuals (k), amount of generations, number of evaluations per individual, crossover probability value (XProb) and mutation probability value (MProb).*
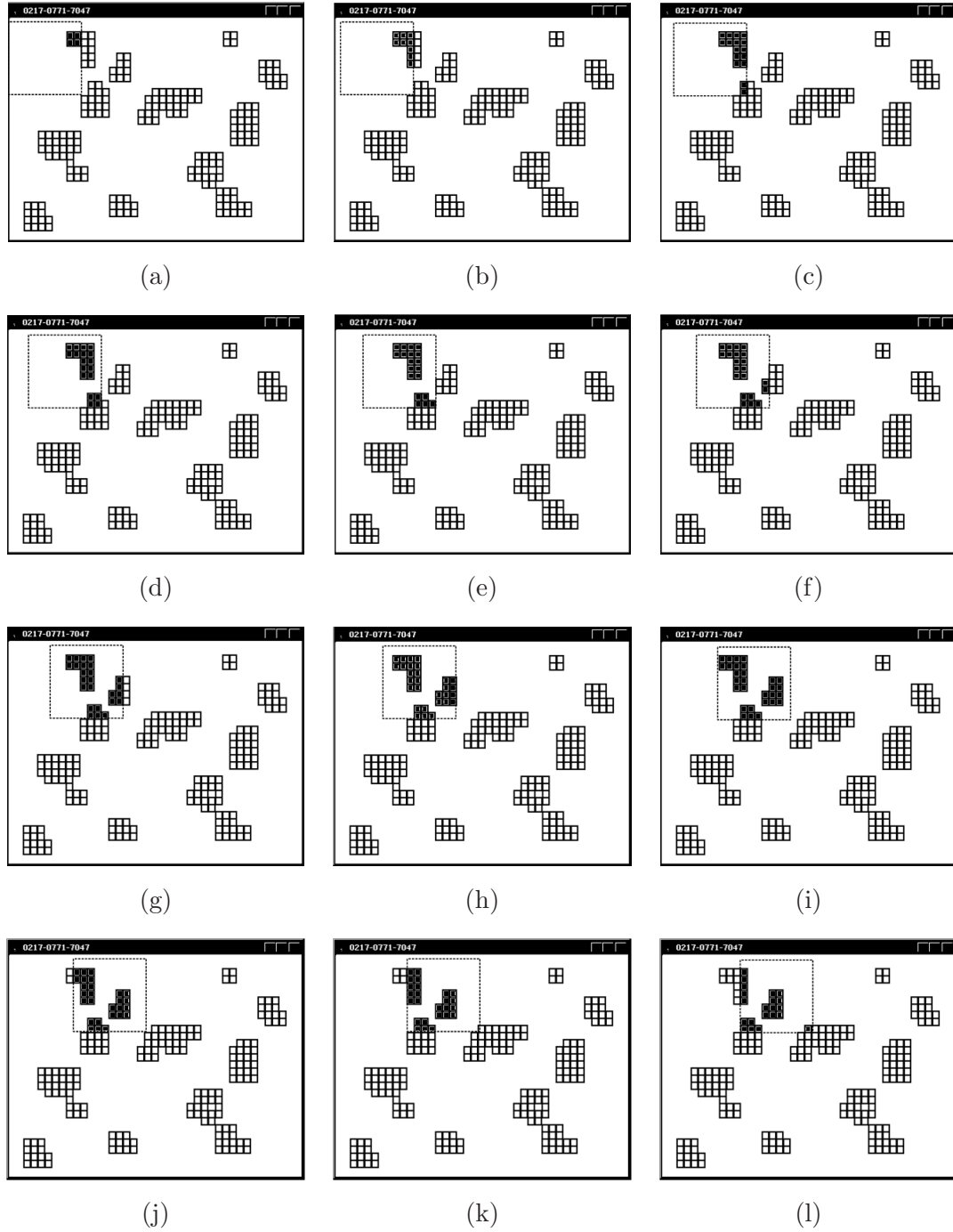
FIGURE 7.10: *Shape scanning example over a final two-dimensional lattice configuration. The target shape, denoted by a broken line area, is exhaustively sought from top left to bottom right over the two-dimensional lattice. The process (a - l) counts how many tiles are present within each possible region of 10×10 tiles and returns the biggest quantity found as the fitness value of the individual's simulation.*

In addition, the simulator parameters such as the length of the simulation, the temperature τ for the glue function, the size of the interaction matrix encoding the glue types strength, the range of the strength values filling the matrix, the dimension of the target shape and the lattice size are shown in Table 7.4.

Sim Length	τ	α	Strengths	Target Shape	Lattice
300	4	10	$[0, 9]$	10×10 tiles	40×30 cells

TABLE 7.4: *Simulator parameters: simulation length, temperature of the system (τ), number of glue types (α), range of discrete strength values, target shape and lattice dimensions.*

Table 7.5 shows the symmetric matrix of glue types used for encoding the interaction strength between α glue types. This structure is randomly initialised with discrete values belonging to the range $[0, 9]$ and remains fixed throughout all the experiments.

	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 7.5: *The symmetric matrix M of $\alpha \times \alpha$ glue types, randomly initialised with discrete strength values belonging to $[0, 9]$. The strength between two glue types C_i and C_j is computed indexing $M[C_i, C_j]$.*

7.2.3 Results

Figure 7.11 plots the evolution of the fitness versus generations for various GA experiments. As it is shown, the best initial fitness values in each of the experiments are always between 30.00 and 33.00. As the evolution progresses, the fitness rises up to 40.00 or more. It is interesting to note that for all the experiments, the fitness does not improve once it reaches values between 40.00 and 42.00. This plateau is sometimes reached at early generations as shown by the trends of experiments 3, 5, 6 and 8. The occurrence of the same phenomena usually takes slightly longer for experiments 1, 2, 4 and 7. In particular, the slope in experiment 1 suggests that there was enough population diversity throughout the evolutionary process to improve steadily. On the other hand, experiments 3 and 5 would suggest that considerable modifications were performed in the genomes present in the population in very short periods of time pointing to a "punctuated-equilibrium" type of dynamics [180]. That is, the individuals are generally stable and changing little along generations. In average, each experiment has run for 72 hours in a Pentium 4-3GHz computer.

Figure 7.12 shows 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 design of the tile families allow almost any pair of instances to self-assemble. Easy self-assembly produces conformations very dissimilar from the target shape. Due to the introduction of tiles that can build vertical or horizontal strips the resulting conformations converge to shapes that are closer to the target one.
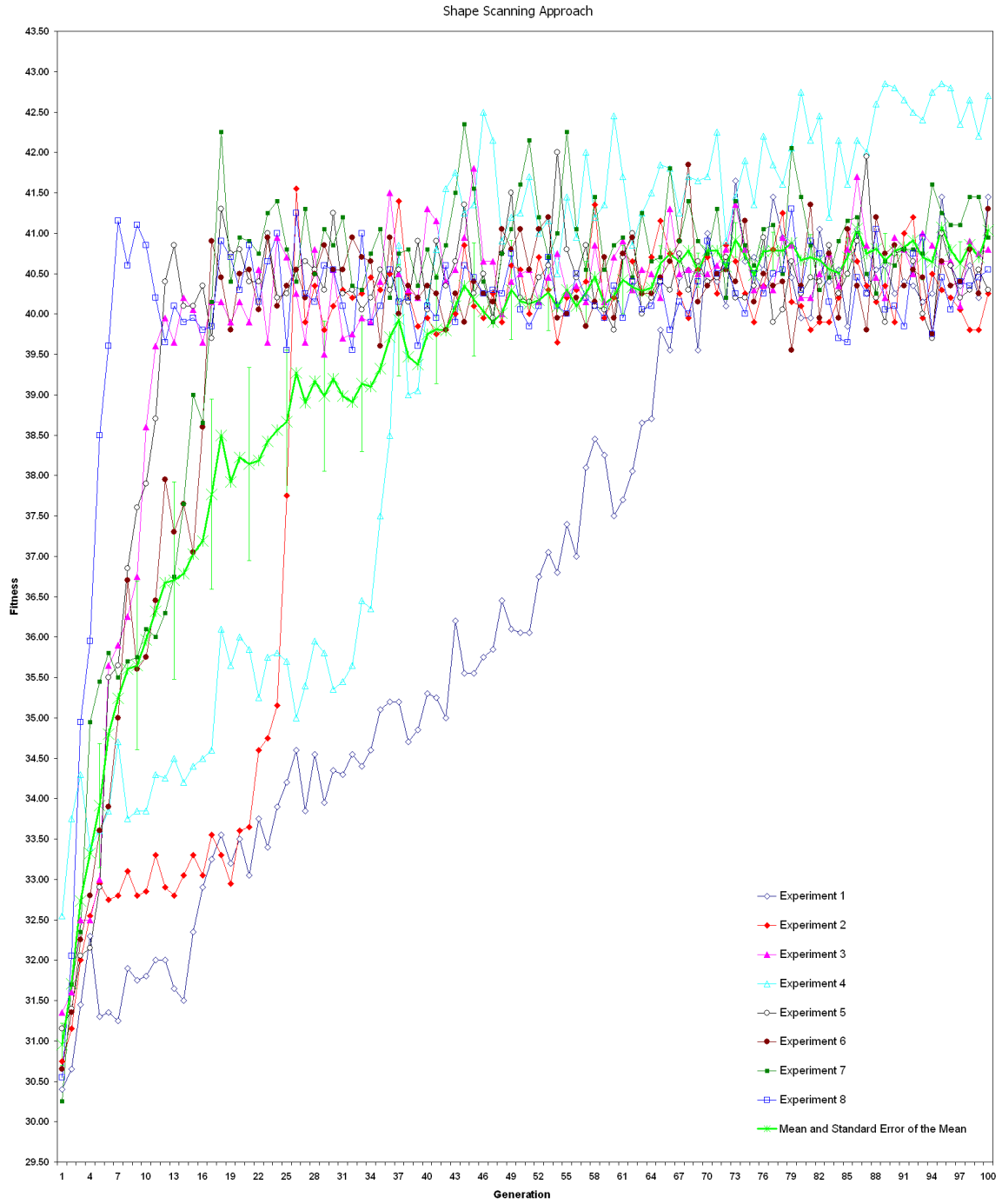
FIGURE 7.11: *Fitness evolution versus generations for the eight experiments. At early generations, the best individuals fill the target shape with around 30 and 33 tiles rising this number up to 40 or 42 towards the end. In some experiments, a fitness plateau is reached within the first 20 generations (experiments 3, 5, 6 and 8) although this phenomena could take slightly longer (experiments 1, 2, 4 and 7).*

Generation 0   *Generation* 15

Generation 30   *Generation* 45

Generation 60   *Generation* 75

Generation 90   *Generation* 99
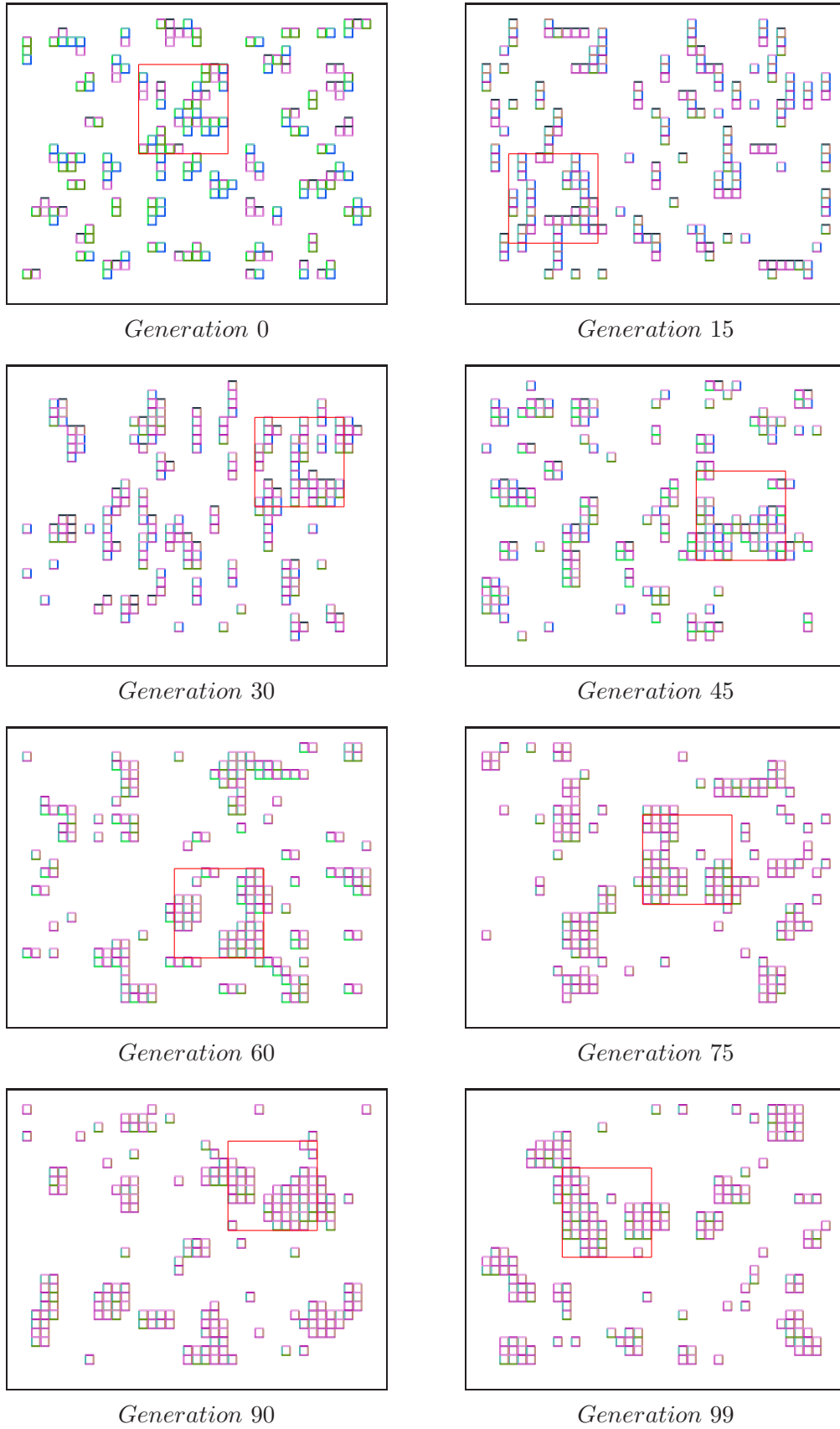
FIGURE 7.12: *Progress of GA experiment M1C when using the lattice scanning approach along generations 0, 15, 30, 45, 60, 75, 90 and 99.*

7.2.4 Summary

This section has presented a GA approach, the goal of which is to design a set of Wang tile families where instances are required to self-assemble into a 10×10 square. During this evolutionary process, the instances of an individual are executed under **Model 1** of the simulator. Once the simulation finishes, the target shape is exhaustively sought across the lattice computing Hamming distances between the content of every possible region of 10×10 tiles and the target shape from where the greatest quantity is returned as the fitness of the individual. Clearly, computing the fitness value is very expensive in terms of time since the sought area had to be moved across the lattice exhaustively.

In addition, the way the scanning takes place does not guarantee that the tiles filling an area of 10×10 are actually self-assembled. For example, the scanning site at Figure 7.10 (i) has a fitness value higher than the one in Figure 7.10 (a). However, the tiles captured at Figure 7.10 (i) belong to three disconnected aggregations, hence misleading the calculation of the fitness value. In order to solve this inconvenience, an additional mechanism for checking the connectivity between the tiles in the scanned area is required, hence increasing the cost per evaluation.

Nevertheless, it is evident that this methodology is strongly bounded to the morphology of the underlying lattice and the geometry of the tiles. More important, this lack of flexibility would be clearly manifested when the user decides to change the shape sought, e.g. circular, triangular or other geometrical figures (which might include holes). It is, therefore, the purpose of the next section to improve the current fitness function while preserving a black-box methodology independent from any specific morphology.

7.3 Evaluation Method 2 – Universal Similarity Metric Approach

This section aims to report on an information distance-based methodology employed to evaluate the individuals of the GA used for the design optimisation of self-assembly Wang tiles. This strategy operates in a black-box way since the fitness of an individual is given in terms of the information encoded in a configuration comprising the achieved self-assembled aggregates.

7.3.1 Architecture

In this approach, 100% of the individuals were initialised with random genes and the performance of the self-assembly process has been assessed using the Universal Similarity Metric (USM). Given that the USM is an information distance metric based in Kolmogorov complexity, both the configuration and the target shape should be converted so that information could be measured. Thus, the same way as the CA approach, once the simulation runs under **Model i** ($i \in [1, 4]$), both the target shape and the configuration are to be captured by binary images as shown in Figure 7.13. For the configuration, every position of the lattice will be black if it is occupied by a tile or white otherwise. In the case of the target shape, a black square representing the target shape is captured into a white image size of which match with the one capturing the configuration.

Considering img_t and img_i two images with same dimensions capturing the target shape and the self-assembly configuration evolved by the GA respectively, $USM(img_t, img_i)$ calculates the fitness of the individual. Given that the closer to 0, the more similar are the USM arguments, the objective here is to minimise the fitness function.
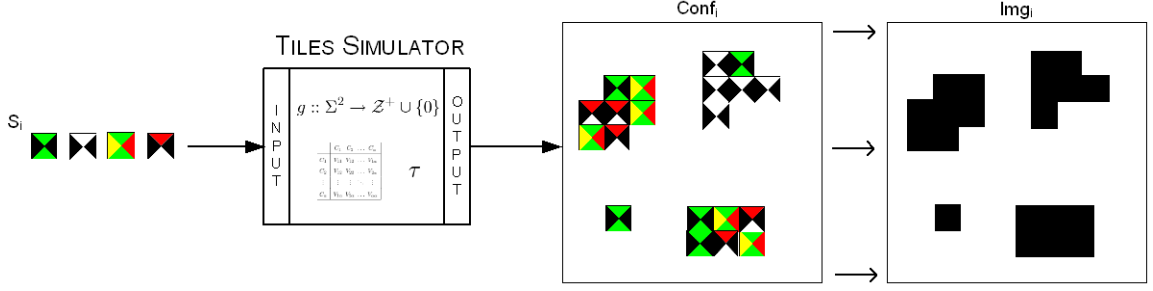
FIGURE 7.13: *A set of tiles S_i is dropped into the simulator where after a number of steps it achieves the configuration $Conf_i$. This output is later captured by a binary image Img_i where black squares map tiles located on the lattice.*

7.3.2   Experiments

For each **Model i** ($i \in [1, 4]$), a set of five GA experiments initialised with three different mutation probability values were run. For all of them, the population size, the maximum length of the individuals, the crossover probability and the amount of generations remained fixed. These parameters together with their values are shown in Table 7.6.

Population Size	k	Generations	Evaluations	XProb	MProb
100	10	300	10	0.7	0.10/0.05/0.01

TABLE 7.6: *GA parameters: population size, maximum length of the individuals (k), amount of generations, number of evaluations per individual, crossover probability value (XProb) and mutation probability value (MProb).*

In addition, the simulator parameters such as the length of the simulation, the temperature of the system τ, the number of glue strengths encoded in the interaction matrix α, the range of strength values filling the matrix, the dimension of the user defined shape, the lattice dimension and the compression algorithm implementing the information

distance metric are shown in Table 7.7. As in the previous approach, the same symmetric matrix shown in Table 7.5 has been employed.

Sim Length	τ	α	Strengths	Target Shape	Lattice	Compressor
9000	4	10	$[0, 9]$	10×10 tiles	800×600 cells	JBzip2

TABLE 7.7: *Simulator parameters: simulation length, temperature of the system (τ), number of glue types (α), range of discrete strength values, target shape dimension, lattice dimension and compression algorithm.*

7.3.3   Results

Table 7.8 summarises the GA findings for a simulator using JBzip2 where **MProb** holds the mutation probability value, **Best Individual** encodes the fittest individual with its fitness in column **Fitness** and **Id** labels the experiment as MiX where Mi stands for **Model i** ($i \in [1, 4]$) and X takes values A, B, C linked to the probability values $0.01, 0.10, 0.05$ respectively, e.g. M1B is the experiment using **Model 1** with probability value 0.10.

From the numerical results summarised under **Fitness** column in Table 7.8, it is interesting to note that when using rotation, i.e. **Model 3** and **Model 4**, the performance of the individuals is quite similar since their fitness values alternate between 0.81 and 0.83 regardless of whether the tiles are provided with probabilistic criteria or not. Conversely, if the execution model lacks rotation, i.e. **Model 1** and **Model 2**, there is a remarkable difference subject to whether probabilistic stickiness criteria is set or not. For example, the GA found more suitable numerical solutions when experimenting with **Model 1** than when performing the search with **Model 2**.

| **Deterministic Criteria & No Rotation** | | | |
MProb	Best Individual	Fitness	Id
0.01	{6152, 6602, 6600}	0.657738	M1A
0.10	{4770, 4172}	0.664593	M1B
0.05	{0261, 0067}	0.647844	M1C

| **Probabilistic Criteria & No Rotation** | | | |
MProb	Best Individual	Fitness	Id
0.01	{1071, 5809, 7773}	0.769912	M2A
0.10	{7175, 5175, 4473, 5100}	0.777932	M2B
0.05	{7477, 7376, 1100, 4707}	0.774544	M2C

| **Deterministic Criteria & Rotation** | | | |
MProb	Best Individual	Fitness	Id
0.01	{9898, 9898}	0.819800	M3A
0.10	{1052, 7554, 1302, 8217, 6023, 2780}	0.823540	M3B
0.05	{9898}	0.821831	M3C

| **Probabilistic Criteria & Rotation** | | | |
MProb	Best Individual	Fitness	Id
0.01	{6265}	0.819730	M4A
0.10	{7271}	0.817750	M4B
0.05	{6065}	0.816220	M4C

TABLE 7.8: *Results summary for the best GA experiments using a simulator set up with JBzip2. MProb holds the mutation probability value, Best Individual encodes the fittest individual with score under Fitness and Id labels the experiment.*

Since a design optimisation problem is addressed, it is important to discern whether the evolutionary process has managed to perform any structural optimisation when tailoring the tile families and, moreover, which are the characteristics of the achieved aggregates. Thus, the structures of the individuals under **Best Individuals** in Table 7.8 reveal that when rotation is set, i.e. **Model 3** and **Model 4**, most of the fittest individuals are short in length and their instances self-assemble in scattered small structures. These characteristics are summarised in Figure 7.16 and Figure 7.17. On the other hand, important differences in length and on the morphology of the aggregates appear when there is no rotation. For instance, it turns out that individuals of **Model 1** are usually of length 2 or 3 and that their instances self-assemble in long vertical strips as shown in Figure 7.14. Furthermore, individuals of **Model 2** are usually of length 3 or 4 and their instances self-assemble in small rectangular structures made of short vertical strips as Figure 7.15 shows.

An analysis of the plots obtained from the evolution of the fitness reveals that the fittest individuals were not found at early generations as with the lattice scanning approach. For example, the trends in Figure 7.18 show that in most of the cases the best individuals were found after half of the evolutionary process: generation 182 for **Model 1**, 152 for **Model 2**, 222 for **Model 3** and 174 for **Model 4**.

The curves in Figure 7.18 also depict that the more complex the model, the fewer modifications over the chromosomes are needed in order to achieve a relevant individual across generations. For instance, the plot for the best experiment with **Model 1** suggest 12 shifts whilst the plot for the best experiment with **Model 4** suggests 9 shifts.

In average, each experiment has run for 7 days in a Sun V20z dual Opteron 248 (2.2GHz).
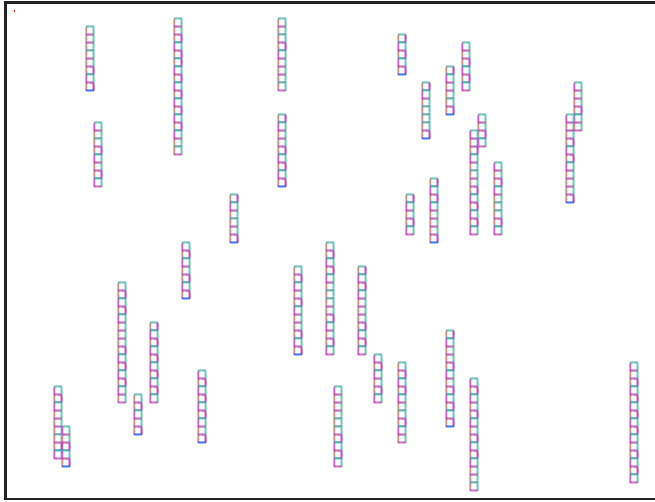
Experiment M1A	
MProb	0.01
Individual	{6152, 6602, 6600}
Fitness	0.657738
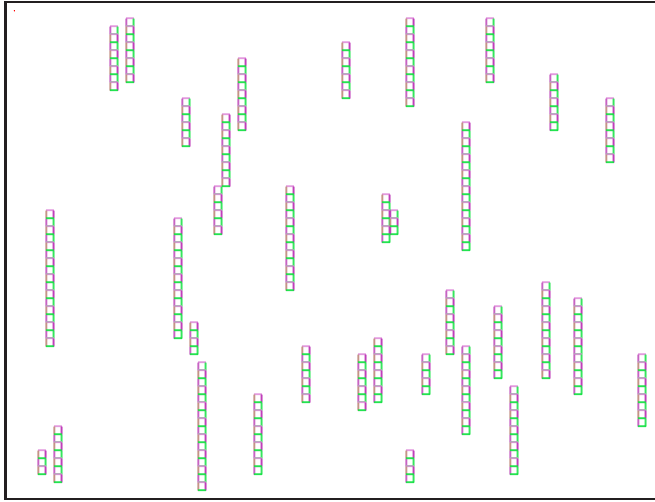
Experiment M1B	
MProb	0.10
Individual	{4770, 4172}
Fitness	0.664593
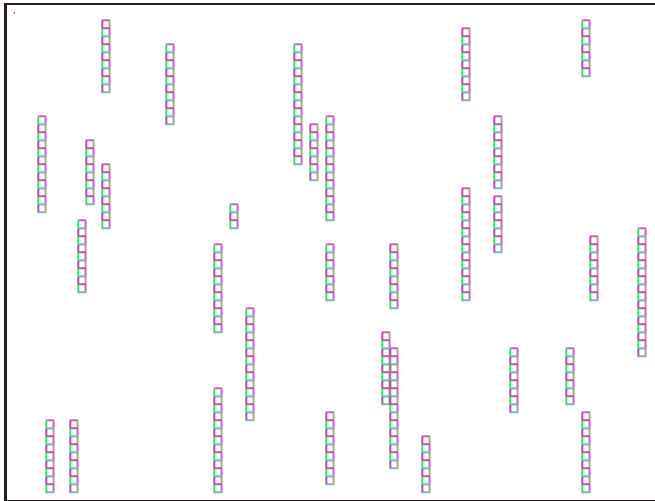
Experiment M1C	
MProb	0.05
Individual	{0261, 0067}
Fitness	0.647844

FIGURE 7.14: *Aggregations, encoding and fitness values of the best individuals achieved by the GA under $MProb = 0.01, 0.10, 0.05$. Tiles interact with deterministic stickiness criteria and no rotation (***Model 1***).*
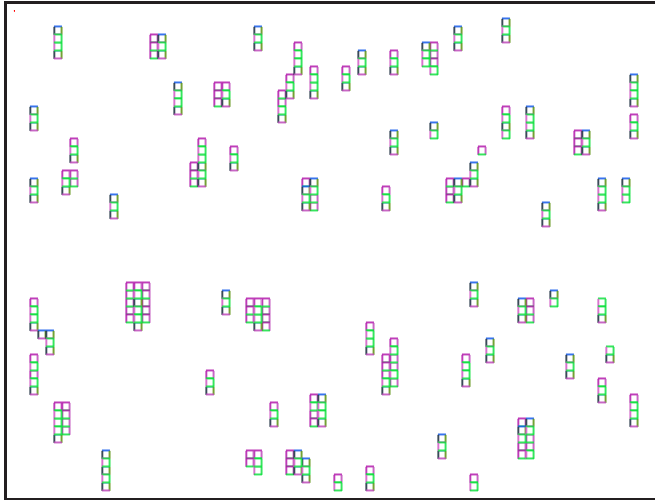
	Experiment M2A	
MProb	0.01	
Individual	{1071, 5809, 7773}	
Fitness	0.769912	
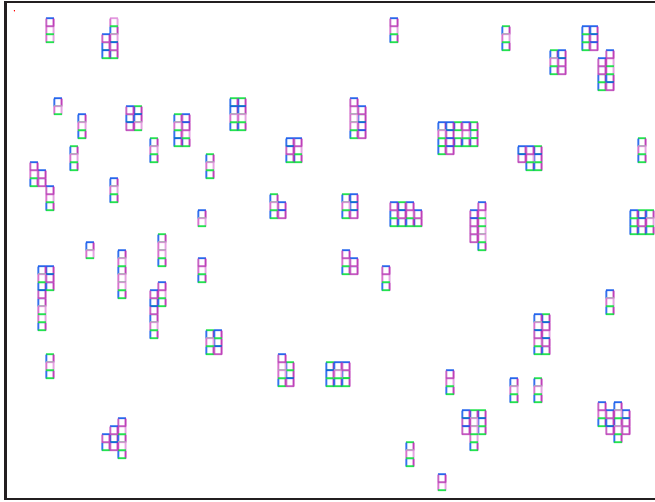
	Experiment M2B	
MProb	0.10	
Individual	{7175, 5175, 4473, 5100}	
Fitness	0.777932	
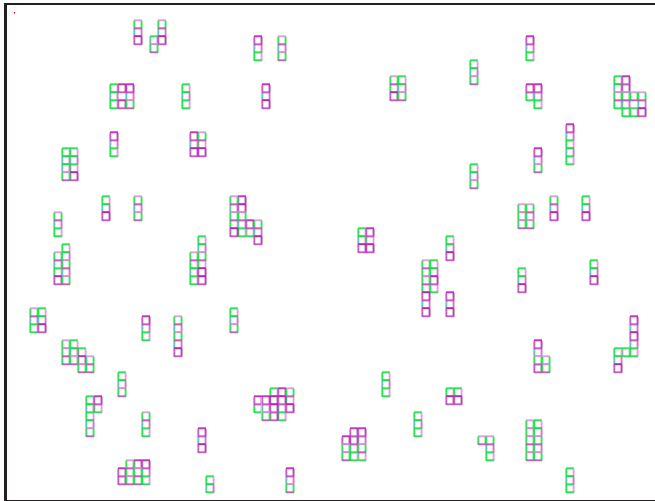
	Experiment M2C	
MProb	0.05	
Individual	{7477, 7376, 1100, 4707}	
Fitness	0.774544	

FIGURE 7.15: *Aggregations, encoding and fitness values of the best individuals achieved by the GA under $MProb = 0.01, 0.10, 0.05$. Tiles interact with probabilistic stickiness criteria and no rotation (*Model 2*).*
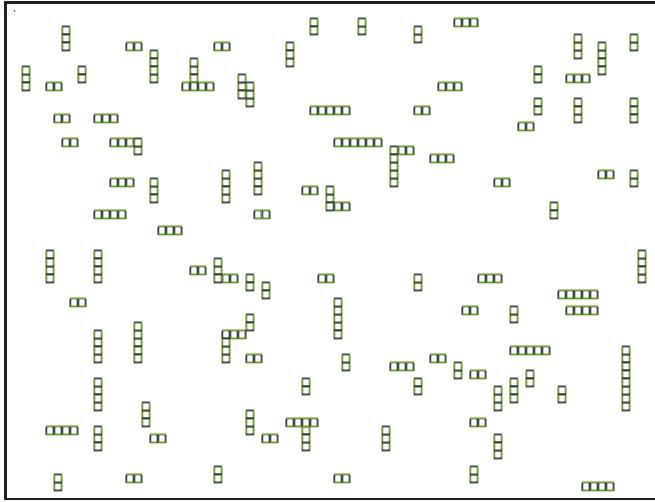
Experiment M3A	
MProb	0.01
Individual	{9898, 9898}
Fitness	0.819800
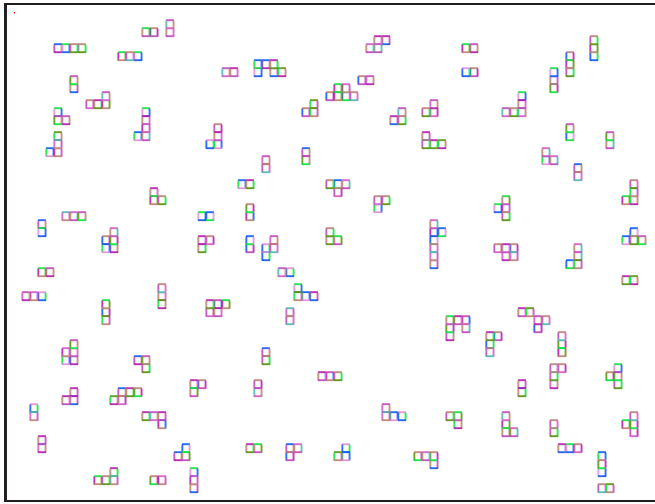
Experiment M3B	
MProb	0.10
Individual	{1052, 7554, 1302, 8217, 6023, 2780}
Fitness	0.823540
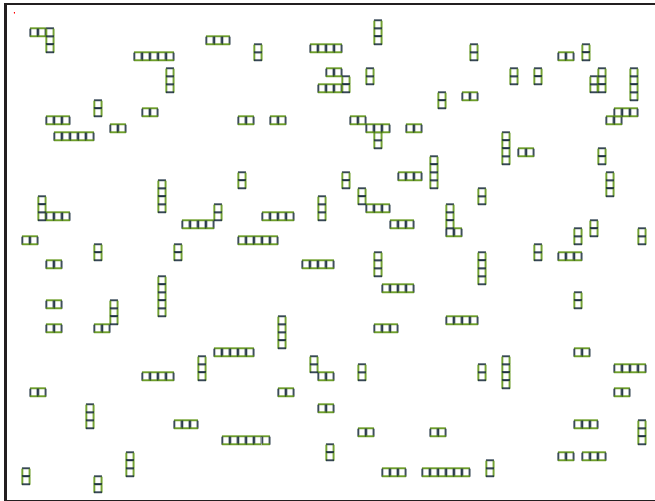
Experiment M3C	
MProb	0.05
Individual	{9898}
Fitness	0.821831

FIGURE 7.16: *Aggregations, encoding and fitness values of the best individuals achieved by the GA under $MProb = 0.01, 0.10, 0.05$. Tiles interact with deterministic stickiness criteria and rotation (***Model 3***).*
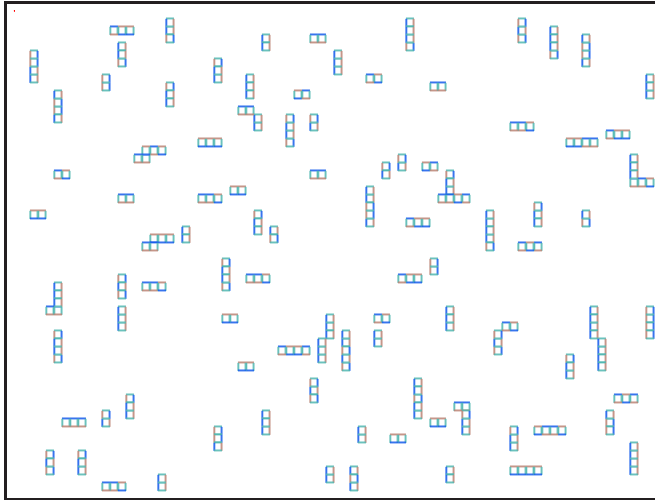
Experiment M4A	
MProb	0.01
Individual	{6265}
Fitness	0.819730
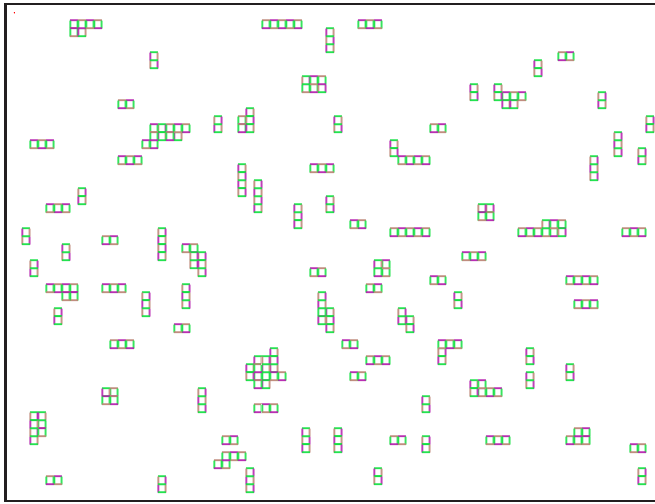
Experiment M4B	
MProb	0.10
Individual	{7271}
Fitness	0.817750
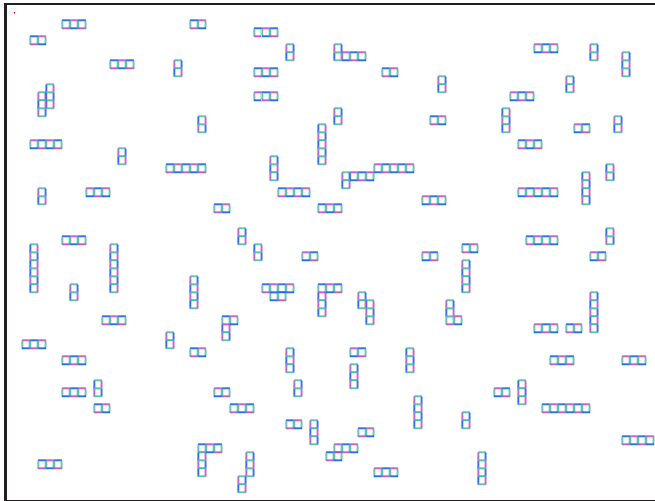
Experiment M4C	
MProb	0.05
Individual	{6065}
Fitness	0.816220

FIGURE 7.17: *Aggregations, encoding and fitness values of the best individuals achieved by the GA under $MProb = 0.01, 0.10, 0.05$. Tiles interact with both probabilistic stickiness criteria and rotation (*Model 4*).*

Comparing the number of changes made on each of the four models, it is certainly clear that modifications on those including rotation have a significant impact in terms of fitness value. For example, the numeric difference between the best individual of the first generation and the best individual of the last generation in **Model 1** (the simplest model) is 0.1846 whilst the same calculi in **Model 4** (the most complex model) is 0.023.

7.3.4 Summary

This section demonstrates a GA approach the goal of which is to design a set of Wang tile families where instances are expected to self-assemble in a shape of 10×10 tiles. During this evolutionary process, the instances of an individual are run with each of the four hierarchical models presented in Subsection 7.1.3. Once the simulation finishes, the output is assessed in terms of the USM introduced in Chapter 4.

After several GA experiments, the collected data has suggested that the simpler the model, the better the performance of the individuals. This observation is supported by the configurations captured in Figure 7.14, Figure 7.16, Figure 7.15 and Figure 7.17. According to them, long vertical strips were found when tiles interact under **Model 1** and **Model 3** whilst small scattered structures were achieved for **Model 2** and **Model 4**.

In comparison to the aggregates obtained by the lattice scanning method presented in Section 7.2, this new approach has shown that employing the USM does not address the self-assembly Wang tiles design optimisation problem as efficiently as when applied to CA parameters design optimisation. It is, therefore, the purpose of the next section to refine further the fitness function through the morphological characterisation of each aggregate.
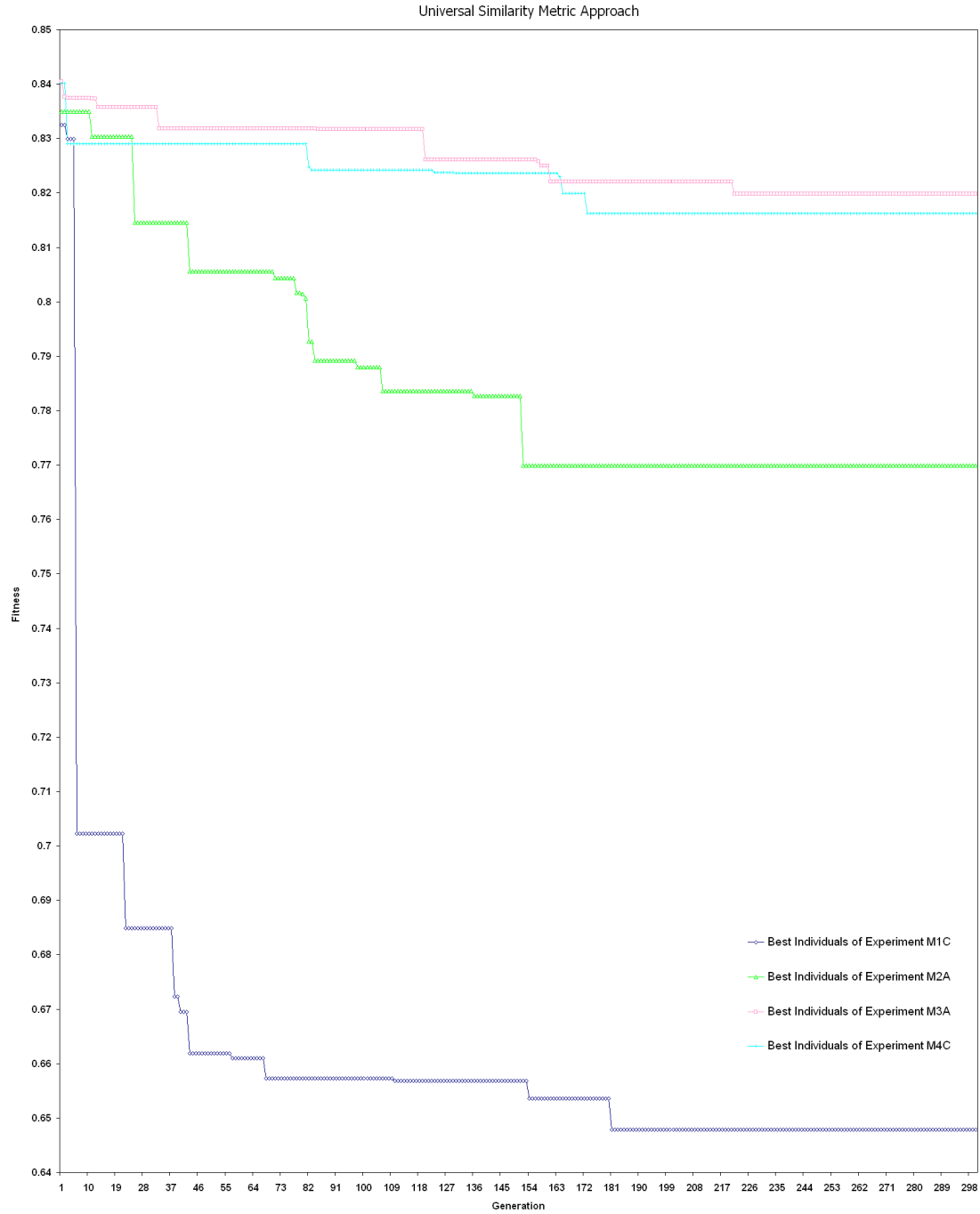
FIGURE 7.18: *Fitness evolution versus generations of the best experiments of each model against the same target. According to the fitness evolution, the more complex the model is, the worse the fitness values are. Experiments evolving self-assembly Wang tiles that interact with deterministic stickiness criteria not only achieved the best performance but also produced large diversity across generations.*

7.4   Evaluation Method 3 – Minkowski Functionals Approach

This section aims to report on the last methodology employed to evaluate the individuals of the GA used for the design optimisation of self-assembly Wang tiles. In contrast to the information distance metric, this new strategy operates in an introspective way since the performance of an individual is given in terms of the structural features of the achieved self-assembled aggregates.

7.4.1   Architecture

In common to the previous approach, the individuals of the initial population are set up with randomly created Wang tile families. After that, the population is subject to a series of generations comprising evaluation, crossover and mutation. It is then in the evaluation phase where instances of the Wang tile families comprising an individual are dropped into a simulator executing **Model i** ($i \in [1, 4]$). Thus, once the simulation finishes, the fitness is computed in terms of the Morphological Image Analysis method (MIA) considering the whole collection of self-assembled aggregates. For example, if an individual Ind_i produces a configuration $Conf_i$ as depicted in Figure 7.19, then its evaluation takes place in the following way. For each of the n aggregates the geometrical descriptors area (A_j), perimeter(U_j), the connectivity characteristic Euler (χ_j) and the radius of gyration (Rg_j) about the x-axis are calculated. These numerical features together with the correspondent A_T, U_T, χ_T and Rg_T of the target shape are evaluated by Equation 7.3 indicating how close, on average, the achieved aggregates to the target shape are. Thus, the higher this numerical value, the better an individual ranks.
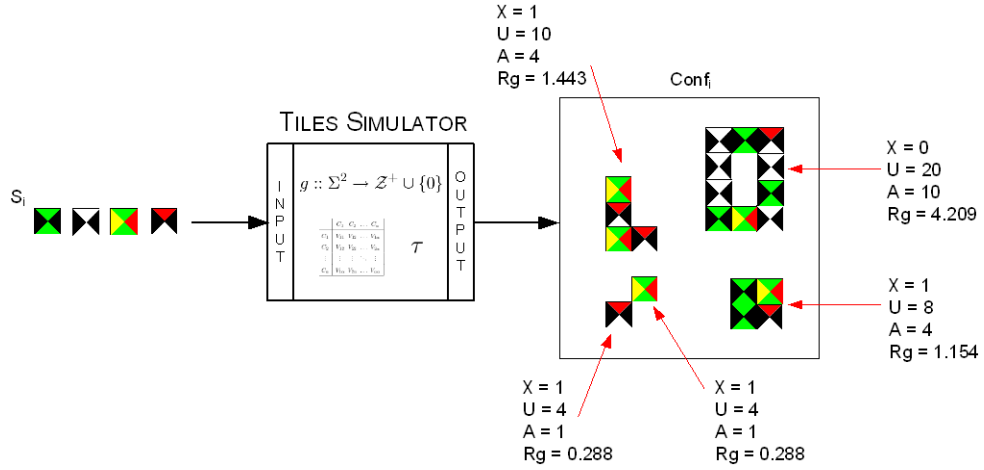
FIGURE 7.19: *A set of tiles S_i is placed into the simulator where after a number of steps achieves the configuration $Conf_i$. For each of the aggregates, the area (A_j), perimeter(U_j), radius of gyration (Rg_j) and Euler (χ_j) are calculated.*

$$Fitness(Ind_i) = \sqrt{(\Delta A)^2 + (\Delta U)^2 + (\Delta \chi)^2 + (\Delta Rg)^2}$$

$$\text{where } \Delta A = A_T - \frac{\sum_{j=1}^{n} \mid A_T - A_j \mid}{n}$$

$$\Delta U = U_T - \frac{\sum_{j=1}^{n} \mid U_T - U_j \mid}{n}$$

$$\Delta \chi = \chi_T - \frac{\sum_{j=1}^{n} \mid \chi_T - \chi_j \mid}{n}$$

$$\Delta Rg = Rg_T - \frac{\sum_{j=1}^{n} \mid Rg_T - Rg_j \mid}{n} \tag{7.3}$$

Equation 7.3 calculates the mean distance between the area of the target shape and each of the areas of the aggregates (ΔA), the mean distance between the perimeter of the target shape and each of the perimeters of the aggregates (ΔU), the mean distance between the Euler of the target shape and each of ones of the aggregates $(\Delta \chi)$ and the mean distances between the radius of gyration of the target shape and the ones of the aggregates (ΔRg).

7.4.2 Experiments

For each **Model i** ($i \in [1, 4]$), a set of five GA experiments initialised with three different mutation probability values were run. For all of them, the population size, the maximum length of the individuals, the crossover probability and the amount of generations remained fixed. These parameters together with their values are shown in Table 7.9.

Population Size	k	Generations	Evaluations	XProb	MProb
100	10	300	10	0.7	0.10/0.05/0.01

TABLE 7.9: *GA parameters: population size, maximum length of the individuals (k), amount of generations, number of evaluations per individual, crossover probability value (XProb) and mutation probability value (MProb).*

In addition, the simulator parameters such as the length of the simulation, the temperature of the system τ, the number of glue strengths encoded in the interaction matrix α, the range of strength values filling the matrix, the dimension of the user defined shape and the lattice dimension are shown in Table 7.10. As in the two previous approaches, the same symmetric matrix shown in Table 7.5 has been employed.

Sim Length	τ	α	Strengths	Target Shape	Lattice
9000	4	10	$[0, 9]$	10×10 tiles	800×600 cells

TABLE 7.10: *Simulator parameters: simulation length, temperature of the system (τ), number of glue types (α), range of discrete strength values, target shape and lattice dimensions.*

7.4.3 Results

The results for each of the GA experiments are summarised in Table 7.11 where **MProb** holds the mutation probability value, **Best Individual** lists the fittest individual with score **Fitness** and **Id** is the experiment in MiX notation where Mi stands for **Model i** ($i \in [1, 4]$) and X takes values A, B, C linked to the probability values $0.01, 0.10, 0.05$ respectively, e.g. M3C is the experiment where tiles undergo **Model 3** and $MProb = 0.05$.

Observing the values under **Fitness**, it is interesting to note that the results of those experiments where tiles are executed under **Model 1** and **Model 3** are quite similar since the fitness of the best individuals vary between 12.00 and 16.00. Conversely, when the execution model incorporates the probabilistic stickiness criteria, i.e. **Model 2** and **Model 4**, there is a remarkable difference subject to the rotation feature. For example, the GA has found more suitable solutions when experimenting with **Model 3** than with **Model 4**.

In order to discern whether the GA has performed any kind of optimisation, an additional examination over the data under **Best Individuals** reveals that the composition of the individuals tends to be homogeneous when rotation is adopted. For instance, the fittest individuals of **Model 3** and **Model 4** have been encoded with 1 or 2 families whilst the ones of **Model 1** and **Model 2** have been encoded with up to 4. On the other hand, from the geometrical characteristics of the achieved aggregates across Figure 7.20, Figure 7.21, Figure 7.22 and Figure 7.23, it is clear that their morphology is more similar to the one of the target shapes when tiles interact under **Model 2** or **Model 4** .

Deterministic Criteria & No Rotation			
MProb	**Best Individual**	**Fitness**	**Id**
0.01	{4483, 0859, 0869, 0999, 0969}	15.319270	M1A
0.10	{5909, 2449, 5809}	12.912152	M1B
0.05	{6622, 9900}	15.583095	M1C

Probabilistic Criteria & No Rotation			
MProb	**Best Individual**	**Fitness**	**Id**
0.01	{0217, 0771, 7047}	35.57111	M2A
0.10	{7517, 7410, 7557}	31.07997	M2B
0.05	{3550, 6467}	37.52435	M2C

Deterministic Criteria & Rotation			
MProb	**Best Individual**	**Fitness**	**Id**
0.01	{9999, 9088}	13.719727	M3A
0.10	{8890, 9999}	12.990824	M3B
0.05	{9999, 6576, 9999, 9999}	15.721738	M3C

Probabilistic Criteria & Rotation			
MProb	**Best Individual**	**Fitness**	**Id**
0.01	{7777, 7777, 7777, 7777, 7777, 7777}	27.830549	M4A
0.10	{7777, 7777}	27.004934	M4B
0.05	{7777}	28.063046	M4C

TABLE 7.11: *Results summary for the best GA experiments. MProb holds the mutation probability value, Best Individual encodes the fittest individual with score under Fitness and Id labels the experiment.*
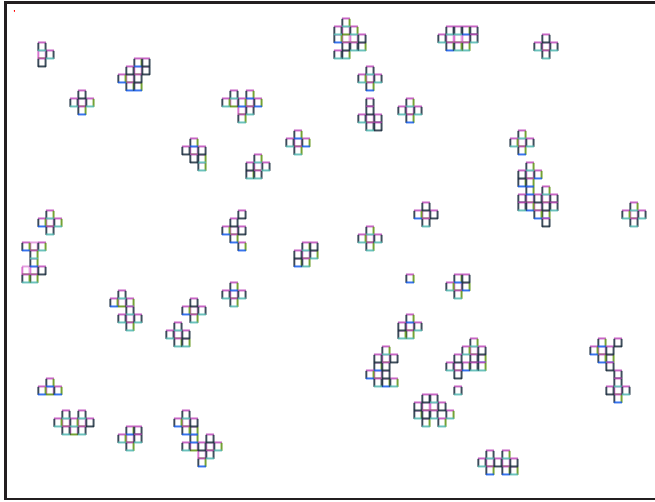
Experiment M1A	
MProb	0.01
Individual	{4483, 0859, 0869, 0999,0969}
Fitness	15.319270
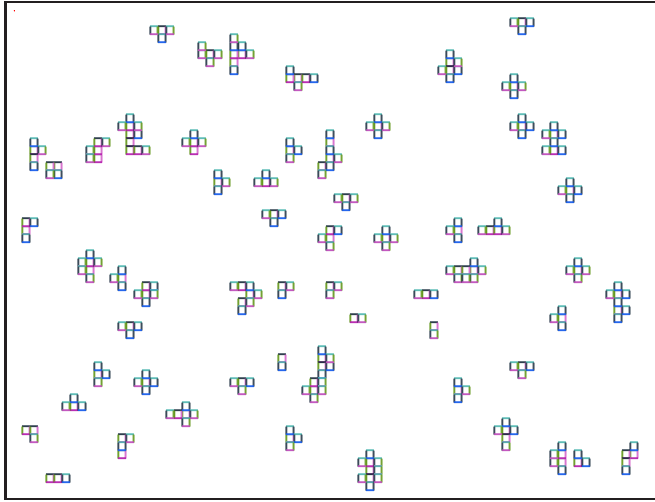
Experiment M1B	
MProb	0.10
Individual	{5909, 2449, 5809}
Fitness	12.912152
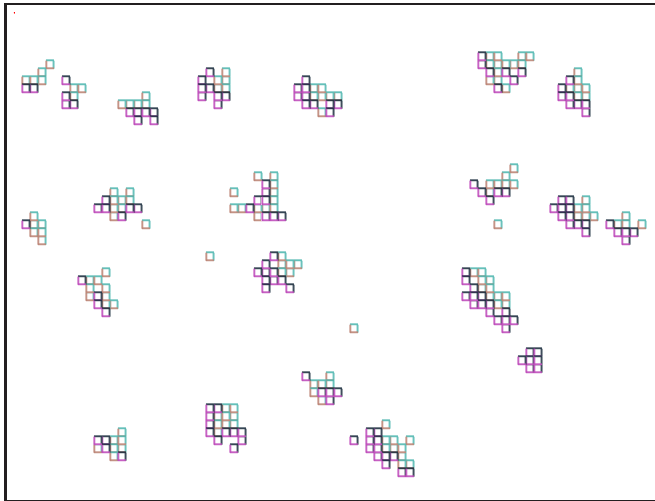
Experiment M1C	
MProb	0.05
Individual	{6622, 9900}
Fitness	15.583095

FIGURE 7.20: *Aggregations, encoding and fitness values of the best individuals achieved by the GA under $MProb = 0.01, 0.10, 0.05$. Tiles interact with deterministic stickiness criteria and no rotation (Model 1).*
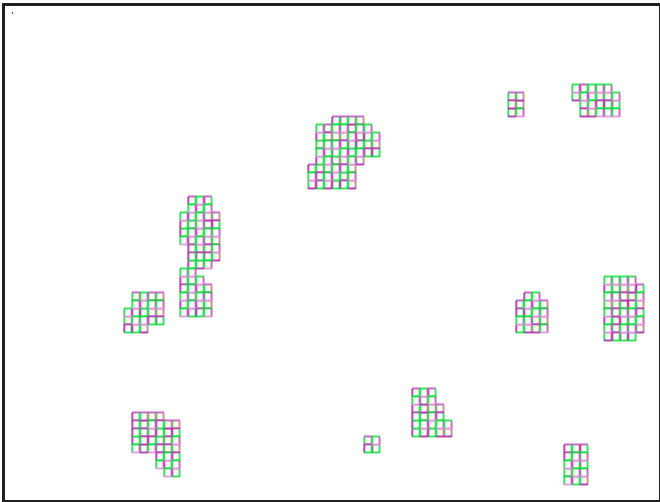
Experiment M2A	
MProb	0.01
Individual	{0217, 0771, 7047}
Fitness	35.57111
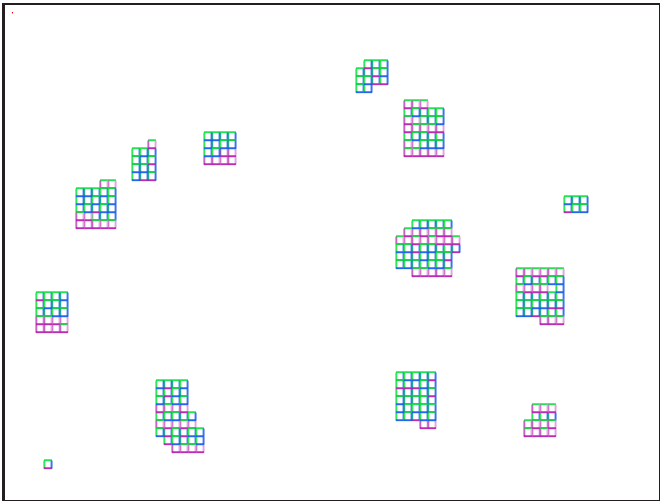
Experiment M2B	
MProb	0.10
Individual	{7517, 7410, 7557}
Fitness	31.07997
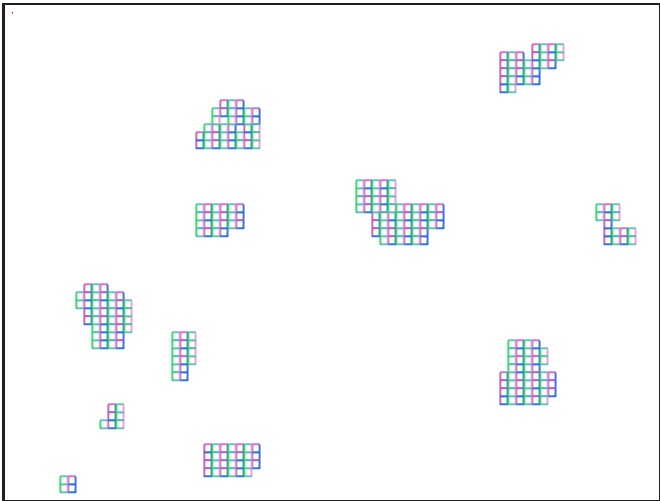
Experiment M2C	
MProb	0.05
Individual	{3550, 6467}
Fitness	37.52435

FIGURE 7.21: *Aggregations, encoding and fitness values of the best individuals achieved by the GA under $MProb = 0.01, 0.10, 0.05$. Tiles interact with probabilistic stickiness criteria and no rotation (***Model 2***).*
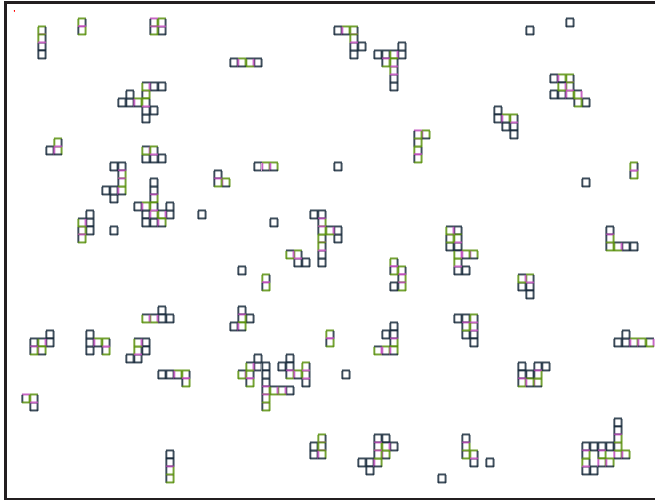
Experiment M3A	
MProb	0.01
Individual	{9999, 9088}
Fitness	13.719727
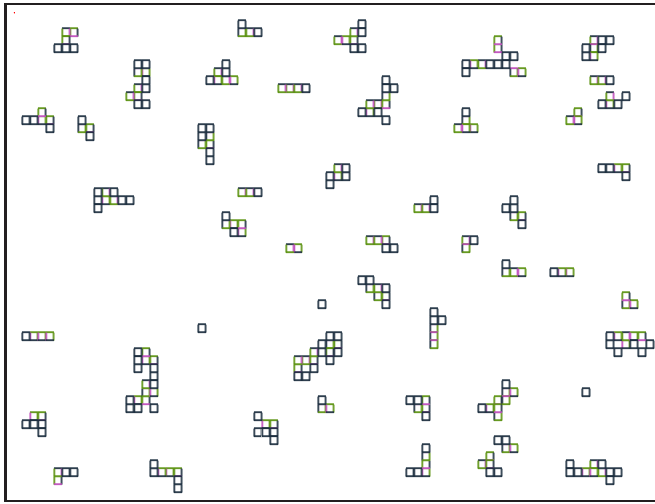
Experiment M3B	
MProb	0.10
Individual	{8890, 9999}
Fitness	12.990824
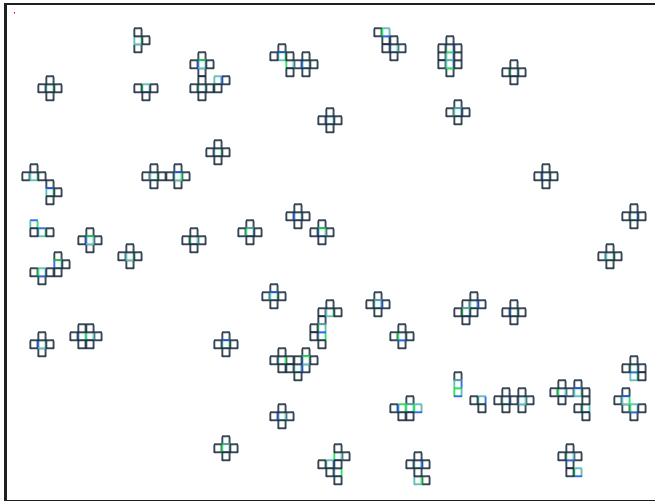
Experiment M3C	
MProb	0.05
Individual	{9999, 6576, 9999, 9999}
Fitness	15.721738

FIGURE 7.22: *Aggregations, encoding and fitness values of the best individuals achieved by the GA under $MProb = 0.01, 0.10, 0.05$. Tiles interact with deterministic stickiness criteria and rotation (***Model 3***).*

Experiment M4A	
MProb	0.01
Individual	{7777, 7777, 7777, 7777, 7777, 7777}
Fitness	27.830549

Experiment M4B	
MProb	0.10
Individual	{7777, 7777}
Fitness	27.004934

Experiment M4B	
MProb	0.05
Individual	{7777}
Fitness	28.063046
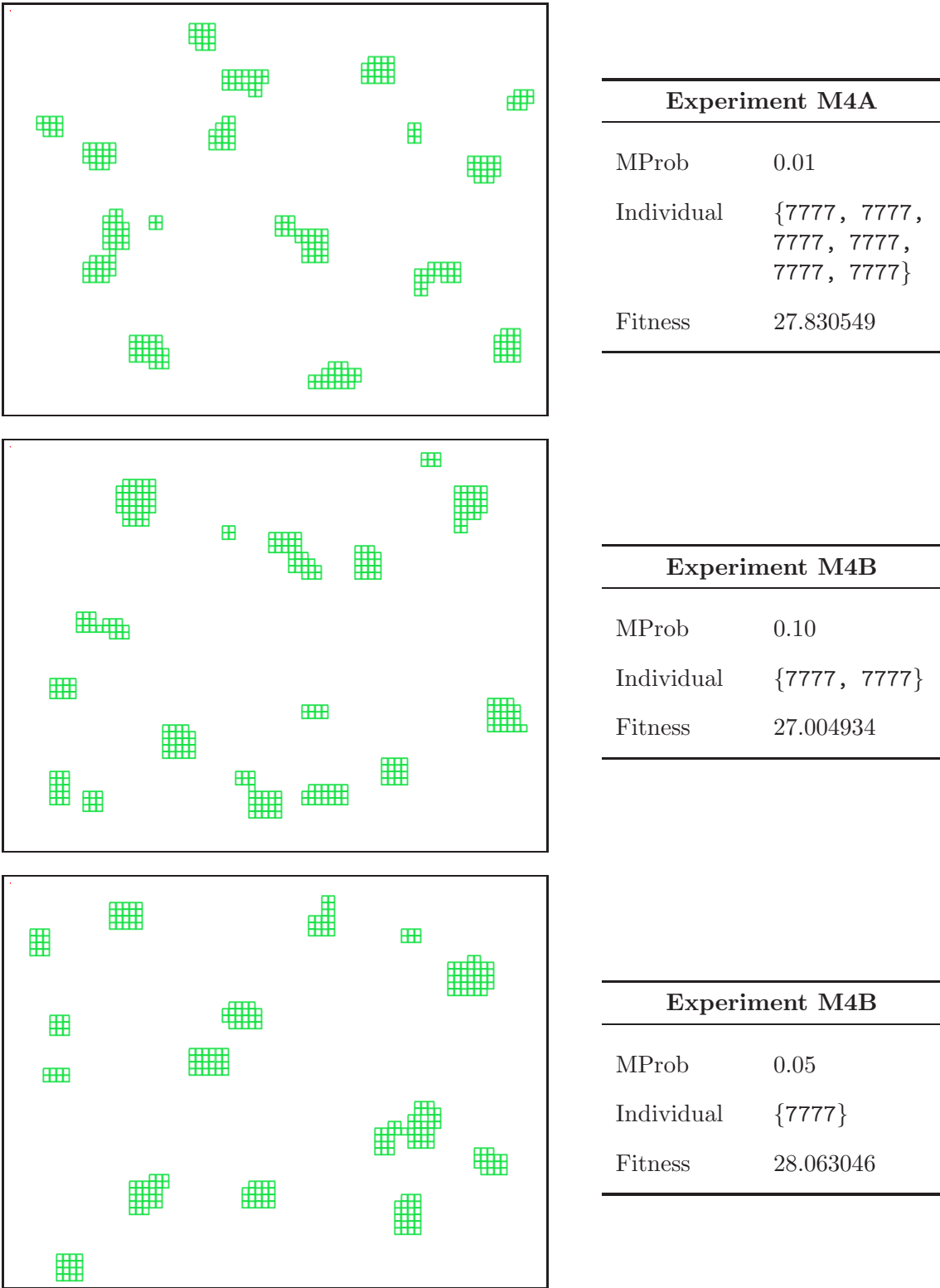
FIGURE 7.23: *Aggregations, encoding and fitness values of the best individuals achieved by the GA under $MProb = 0.01, 0.10, 0.05$. Tiles interact with probabilistic stickiness criteria and rotation (*Model 4*).*

Figure 7.24 depicts the fitness evolution versus generations of the best experiment for each of the models, i.e. $M1C$, $M2C$, $M3C$ and $M4C$. Each point in these curves represents the fitness of the best individual in a generation. The figure is reduced to half of the generations as from generation 150 to 300 the fitness oscillate between the same values. The analysis of these plots clearly reveal that the worse results were obtained when tiles undergo models with deterministic stickiness criteria, e.g. **Model 1** and **Model 3**. This observation across generations comes to support the partial analysis done over Table 7.11.

Moreover, the figure also shows that there is an increasing variance among the fitnesses of the best individuals of each generation as the model becomes more complex. This result suggests that the morphological difference on the achieved configurations across generations is more steady in **Model 1** and **Model 3** than in **Model 2** and **Model 4**. This indicates how important is the influence of the probabilistic stickiness criteria. In average, each experiment has run for 5 days in a Sun V20z dual Opteron 248 (2.2GHz).

7.4.4 Summary

This section has presented a GA approach whose goal is to design Wang tile families, instances of which are capable of crafting a square shape of 10×10 tiles by means of self-assembly. During this evolutionary process, instances of an individual are left to interact into a simulator executing one of the four hierarchical models presented in Subsection 7.1.3. Once the simulation finishes, the output is assessed with an introspective methodology that analyses the morphology of the achieved aggregates and returns a numerical value that acts as the fitness of the individual.
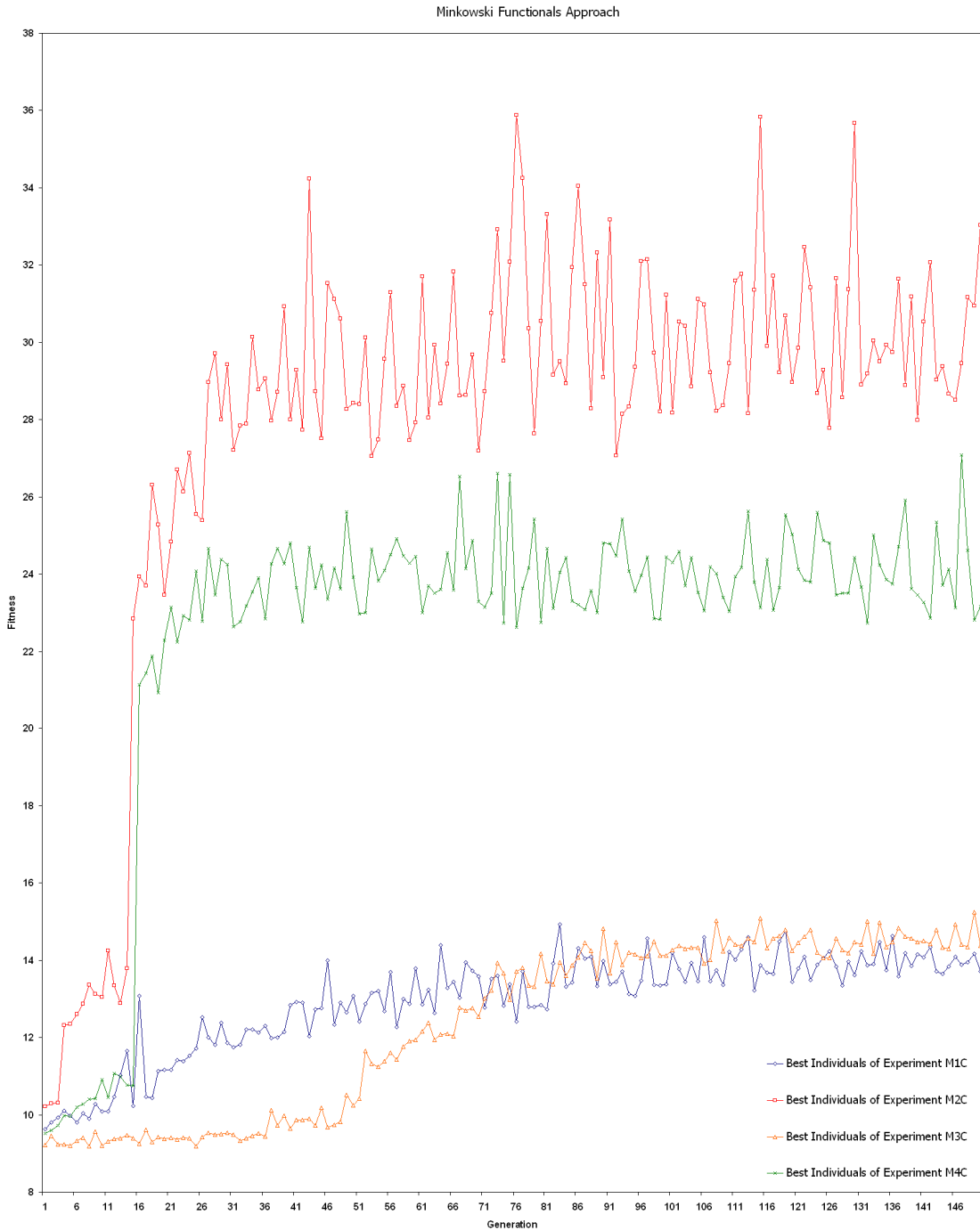
FIGURE 7.24: *Fitness evolution versus generations of the best experiments of each model against the same target. According to the fitness evolution, the more simple the model is, the worse the fitness values are. Experiments evolving self-assembly Wang tiles that interact with deterministic stickiness criteria had not only achieved the best performance but also produce large diversity across generations.*

The contribution given by this approach evaluates the genotype in terms of Euler connectivity and geometrical descriptors such as area, perimeter and radius of gyration. This implementation results in an evaluation mechanism which is expensive in terms of time given that, for each simulation, the aggregates of a configuration have to be detected and evaluated separately. However, the way this evaluation takes place produces a more accurate fitness calculation. From a more general point of view, this computed information gives not only the quality but also the amount of achieved aggregations an individual contributes.

After several GA experiments, the numerical findings of the GA experiments have suggested that the richer the model, the better the performance of the evolved individuals. This observation is supported by both the configurations captured in Figure 7.20, Figure 7.21, Figure 7.22, Figure 7.23, and the trends of the fitness evolution versus generation of Figure 7.24. According to these, large aggregates were found when tiles underwent **Model 2** or **Model 4** whilst these were small and scattered when subject to **Model 1** and **Model 3**, indicating that the probabilistic stickiness criteria makes an important contribution in the evolutionary design optimisation of self-assembly Wang tiles.

7.5 Conclusions

Engineering complex global structures or patterns from local interactions of a myriad of components is challenging. From a computational point of view, the automated design of local behaviour capable of achieving a particular global goal can be compared to the search for specific programs or rules that allow the components to decide whether to bind others under certain environmental properties. In this chapter, a GA approach to the automated

design of self-assembly Wang tiles using lattice scanning, USM and morphological image analysis as fitness function for individuals evaluation, has been presented.

In the lattice scanning method, the population has been mainly initialised with Wang tile families whose instances are capable of self-assembling in complex aggregates as rows and columns. These intermediate composites helped evolution to explore a more reduced search space since the GA is provided with some solutions which are morphologically close to the specified target shape. Although the evaluation mechanism is built in terms of an area that is scanned across the lattice counting the amount of embedded tiles, the final results do not actually reflect the quality of the self-assembled aggregations since the inspected sites do not consider whether the enclosed tiles are connected or not. Moreover, the underlying idea of scanning was inflexible since a small geometric change in the look up area would represent a considerable modification of the algorithm. Similarly, changes over the topological features of the lattice or the tiles would lead to a complete reimplementation and would eventually increase the evaluation cost.

In the information metric based approach, the population of the GA has been randomly initialized and the simulator has been provided with four increasing models of interaction in turn. In this case, once the simulation finishes, the obtained configuration is captured in a binary image that is compared in terms of information distance with another image that captures the target shape. Since this technique is based on Kolmogorov complexity, an approximation in terms of data compression algorithm (JBzip2) has been used.

The main features observed in this methodology were generality and independence at both a geometrical and a topological level. These characteristics provide the maximum flexibility and adaptation for both the evaluation of the self-assembled aggregates and the use of any kind of tiles lattice or target shape. Although it presents a more dynamic mechanism of evaluation, the quality of the results are yet not as favourable as the ones issued by the lattice scanning approach.

In the Minkowski functionals approach, the initial population of the GA has been randomly created and the simulator has been provided with the four models of interaction. In this procedure, after performing the simulation of an individual, each of the self-assembled aggregates found in the configuration is compared to the target shape in terms of Euler characteristic, area, perimeter and radius of gyration. The main feature observed is its ability for characterizing intricate forms that fuels the evolutionary process with accurate information about the self-assembly process. Although this approach might be hard to extend on different lattice topologies and dimensions, its implementation is not as rigid as the lattice scanning. Overall, given the impressive similarity among the user defined shape and the achieved aggregates, this fitness function emerges as the best among all the proposals for evaluating the individual's phenotype. Moreover, it does not require the implementation of a pre-processing stage to seed the population with intermediate structures nor any pre-assessment of external technologies such as compression algorithms.

In order to show how the evolution has progressed, the best GA experiment when employing the USM and Minkowski functionals is shown in a series of successive snapshots by the following figures. For instance, Figure 7.25 presents the progress of the GA experiment $M1C$ when using the information distance metric approach across generations 0, 5, 21, 67, 153 and 180. This group of snapshots shows that the configuration achieved by the best individual of the initial generation comprises several small structures scattered across the two-dimensional lattice. A few generations later, at the sixth generation, the GA has been able to evolve an individual that embodies vertical long strips uniformly distributed in the configuration. From this generation in advance, there is not a significant contrast in terms of aggregates' morphology although the associated genotypes have increased in fitness value. On the other hand, Figure 7.25 shows the progress of GA experiment $M2C$ when using the Minkowski functionals approach across generations 1, 8, 14, 24, 47 and 227. As in the previous experiment, the best individual of the first generation embodies small aggregates scattered across the lattice. After a few generations, the evolution has discovered individuals whose instances are capable of self-assembling in larger aggregates; keeping a stepwise improvement reflected in generations 14 and 24. Whilst the evolution steps up towards the last generations, the achieved aggregates have gradually grown in dimension nearly close to the target shape.

Generation 0

Generation 5

Generation 21

Generation 67

Generation 153
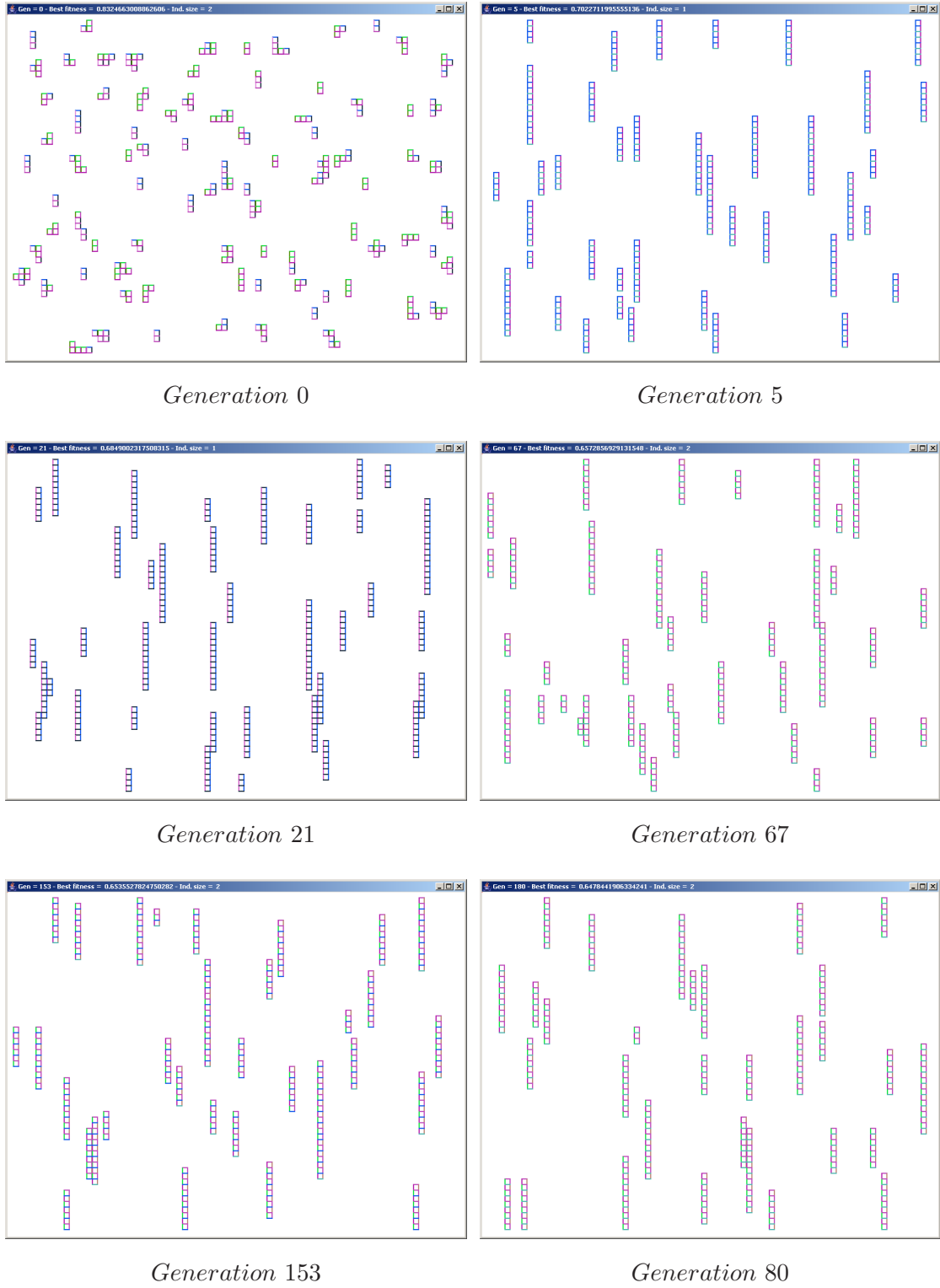
Generation 80

FIGURE 7.25: *Progress of GA experiment M1C when using the information distance metric approach along generations 0, 5, 21, 67, 153 and 180.*

Generation 1   *Generation* 8

Generation 14   *Generation* 24

Generation 47   *Generation* 227
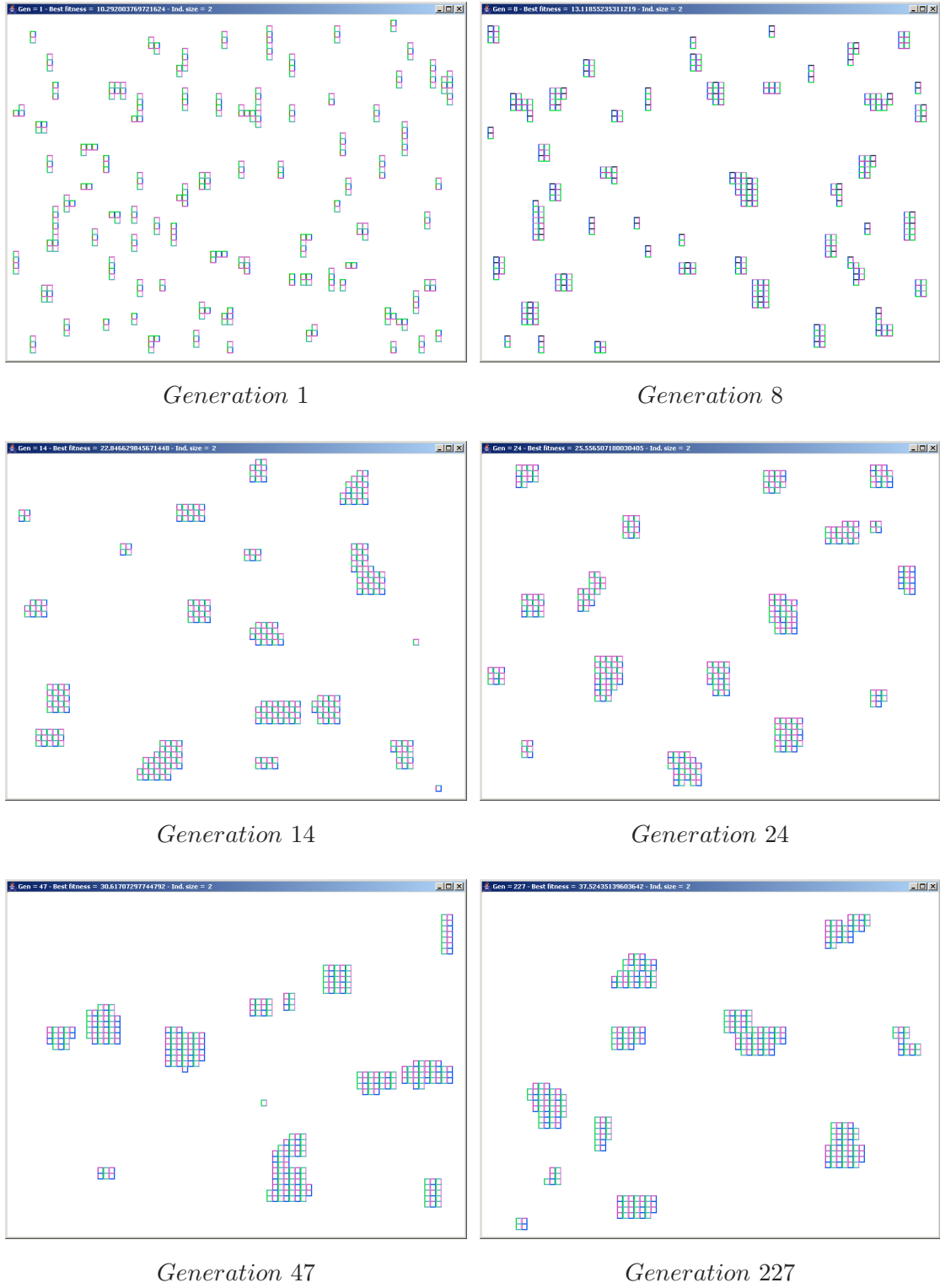
FIGURE 7.26: *Progress of GA experiment M2C when using Minkowski functionals approach along generations 1, 8, 14, 24, 47 and 227.*

CHAPTER 8

Genotype-Phenotype-Fitness Mapping Analysis

The previous two chapters reported on GAs for the design optimisation of CA parameters and self-assembling Wang tiles. Apart from the significant findings, nothing has been yet said about the efficiency by which individuals were evolved. In particular, given the fact that the mapping from genotype to phenotype and from this to fitness is clearly a complex, stochastic and non-linear relationship. One of the most common procedures would suggest running many GA experiments for different configurations followed by a fitness comparison, which is not only time-consuming but also inaccurate for such intricate mappings. Thus, this chapter aims to report on a complementary dual assessment to analyse whether the employed GAs are effective design optimisation methods for the two problems. The following sections present a summary statistic to measure how effectively the fitness of an individual correlates to its genotypic distance to a known optimum, and introduce clustering as a mechanism to verify how the objective function can effectively differentiate between dissimilar phenotypes and classify similar ones for the purpose of selection. Both proposals are applied to instances of problems seen in Chapter 5, Chapter 6 and Chapter 7.

8.1 Fitness Distance Correlation

Fitness Distance Correlation (FDC) [181] [182] is a summary statistic that performs a correlation analysis in terms of a known optimum and samples taken from the search space, predicting whether a GA will be effective for solving an optimisation problem. Thus, when facing a minimisation (maximisation) problem, a large positive (negative) correlation indicates that a GA may successfully treat the problem or that the problem is straightforward, whereas a large negative (positive) value suggests that employing a GA may not be effective or that the problem is misleading. However, a correlation around zero, i.e. $-0.15 \leq$ FDC ≤ 0.15, would suggest that more nuisances, perhaps including scatter plots analyses, of the fitness versus distance to the optimum should be done and, in general the problem is categorized as difficult. The formula for computing the FDC is shown in Equation 8.1, where n is the number of samples, f_i is the fitness of sample i with distance to the known optimum d_i, $\overline{f}$ and S_F are the mean and standard deviation of the fitness values, and $\overline{d}$ and S_D are the mean and standard deviation of the distances.

$$FDC = \frac{(1/n)\sum_{i=1}^{n}(f_i - \overline{f})(d_i - \overline{d})}{S_F S_D}$$

$$\overline{f} = \frac{1}{n}\sum_{i=1}^{n} f_i$$

$$\overline{d} = \frac{1}{n}\sum_{i=1}^{n} d_i$$

$$S_F = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(f_i - \overline{f})^2}$$

$$S_D = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(d_i - \overline{d})^2} \tag{8.1}$$

A study focused on whether FDC predicts the GA behaviour, and whether it detects differences in encoding and representation for a number of well-studied minimisation problems has been given in [181]. When predicting the GA behaviour, the FDC confirmed that Deb & Goldberg's 6-bit fully deceptive function and Whitley's 4-bit (F2 and F3) fully deceptive functions are indeed misleading since the correlation values were 0.30, 0.51 and 0.36 respectively, and the fitnesses tended to increase with the distance from the global optimum. In addition, FDC also confirmed that problems like Ackley's One Max, Two Max, Porcupine and NK landscape problems for $K \leq 3$ are easy since the correlation values resulted in -0.83, -0.55, -0.88 and -0.35 respectively. Nevertheless, the FDC indicated that NK(12,11) landscape, Walsh polynomials on 32 bits with 32 terms each of order 8, Royal Road functions R1 and R2, as well as some of the De Jong's functions like F2(12) are difficult since the resulting correlation values were close to 0.0. When the differences in encoding and representation were considered, experiments using Gray and binary coding led to the conclusions that the superiority depends on the number of bytes used to encode the numeric values. For instance, De Jong's F2 with binary coding is likely to make the search easier than with Gray coding when using 8 bits. In contrast, the correlation value of F12 indicated that Gray coding works better than binary when using 12 or 24 bits.

An attempt to transfer FDC to Genetic Programming in order to study the equivalent dynamics has been done in [183]. In this case, two different GP models based on trees built upon function and terminal sets have been studied. In the first model, individuals are forbidden to have as a child a function with an arity greater or equal than the arity of the father, whereas in the second model such restriction is removed. Thus, individuals

are evaluated with a fitness function defined in terms of their structural distance to the optimum and an unimodal trap. In order to measure the success of a trap function, the performance (p) was defined as the number of executions for which the global optimum has been found in less than 200 generations over a total of 100 experiments. Thus, p and the FDC have been calculated for various trap functions. The results have indicated that for the first model, the performance went from 0 to 1 as the correlation varied from 1 to -1, indicating that the FDC is an accurate predictor. In addition, the findings were similar for the second model although the number of trap functions with FDC $= -1$ was larger than the number found with the restricted model. In addition to these two experiments, similar results were achieved with another trial towards a different optimum. Additional satisfactory measures of problem difficulty analysing multimodal trap functions, royal trees and the max problem were shown in [184].

Despite its successful application on a wide benchmarking set of problems, FDC is still not considered to be a very accurate predictor in some other problems. A case where FDC failed as a difficulty predictor has been presented when studying a GA maximising *Ridge functions* [185]. Let S be a set of binary strings of length n, $\mid S \mid = 2^n$, a string $s \in S$ is evaluated with a ridge function f_n defined in Equation 8.2, where q is the number of contiguous 1 starting from the left in s, m is the total number 1 in s and p is the total number 0 in s. Considering an optimum string $s_{opt} = 111\ldots111 \in S$, a specific path P of length m is formed from the complement of the optimal towards the optimal as $P = (000\ldots000, 100\ldots000, 110\ldots000, \ldots, 111\ldots110, 111\ldots111)$ where all s_i on-path P are given suc-

cessively higher evaluations from the complement to the optimal as $f_n(s_i) < f_n(s_{i+1})$ for $1 \leq i \leq m$. For instance, for $n = 4$, $f_4(1100) = 6$ and $f_4(1010) = 2$.

$$f(s) = \begin{cases} n + q & \text{if } q > 0 \text{ and } m = q \\ p & \text{otherwise} \end{cases} \tag{8.2}$$

On the one hand, initial observations employing Hamming distance between an individual and s_{opt}, and Ridge functions starting with $n = 20$ have shown FDC values of 0.998906 tending to 1 as n tends to ∞ suggesting that Ridge functions would be difficult to solve. On the other hand, a GA equipped with Roulette selection, a population size of 100 individuals, best fit merge and a mutation rate of $1/n$ has proven that the problem is yet easy to treat since the average amount of evaluations needed to reach the optimum represented a small fraction of the search space when experimenting with $n = 20, 25, 30$. For instance, when $n = 30$ the average number of evaluations for achieving the optimum in 20 runs was 6921 which represents 0.0006% of the search space. In addition, experiments using a GA without mutation found similar results since the average amount of simulations has represented 0.0014% of the search space.

In summary, although FDC cannot be expected to be a perfect predictor of performance, previous work reported in [186] [187] [188] [189] [190] suggests that it is indeed a good *indicator* of performance. Therefore, the goal of the following sections is to assess how effectively the fitness of an individual correlates to its genotype when using USM and Minkowski functionals as fitness functions for the GA presented in Chapter 5, Chapter 6 and Chapter 7.

8.2   Correlating the Cellular Automata Design

In a complex system such as the two CA-based investigated in Chapter 5 and Chapter 6, the mapping from genotype to phenotype and then from phenotype to fitness is a highly complex and computationally expensive non-linear and in some cases stochastic relationship. Figure 8.1 shows the three stage mapping process from genotype (the real numbered parameters) onto a phenotype through the execution of the complex system (the CA model) itself and then from this phenotype (a spatio-temporal pattern) onto a numerical fitness value via the objective function (the USM).
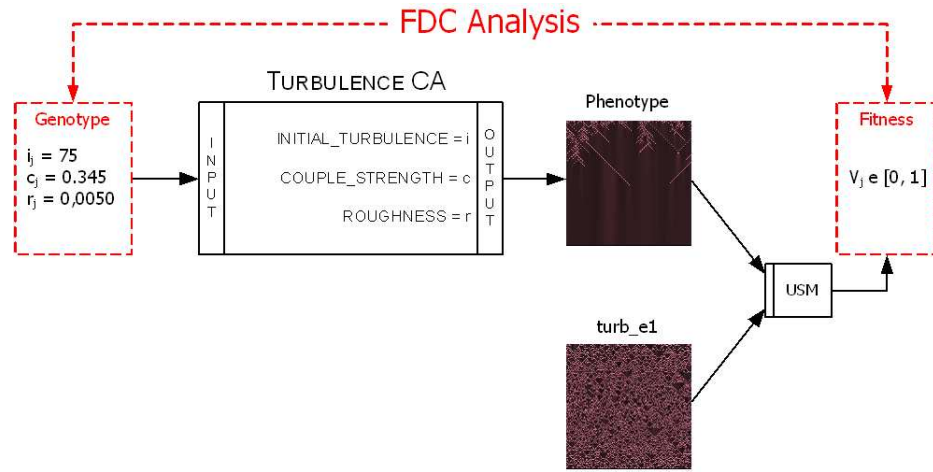
FIGURE 8.1: *Diagram of mappings from genotype onto phenotype and from phenotype onto numerical fitness value, and relationship to the Fitness Distance Correlation.*

As has been described, the FDC is a measure of how effectively the fitness of an individual correlates to its genotypic distance to a known optimum. That is, given two different genotypes, FDC measures the correlation of the (numerical) Euclidean distance between them against their fitness values assigned by the objective function. If there is only a small relationship between these two values, a parameter optimisation GA, or for that

matter any metaheuristic based on the same representation, will not be too successful. For this reason, since FDC performs an analysis of genotype-fitness relationship, the following section pursues a USM assessment with FDC in order to study its effectiveness as fitness function for the evaluation of captured spatio-temporal behaviours.

8.2.1 Experiments and Results

A data set comprising a number of spatio-temporal images was compiled for Turbulence CA and Meta-automaton CA introduced in Chapter 5 and Chapter 6. In order to extend the scope of the study, two CA are also included: CA Continuous and Gas Lattice CA which are one-dimensional and two-dimensional CAs respectively, both implemented in Netlogo.

CA Continuous consists of a discrete time-space representation where the states are encoded as continuous rather than discrete values allowing the model to create a rich variety of complex behaviour. For instance, one of the applications of this system is modelling the behaviour of a boiling liquid where at each time step, the value of a cell is a function of its neighbours, in essence representing the process of heat diffusion. Some examples generated with this implementation are captured in Figure 8.2.
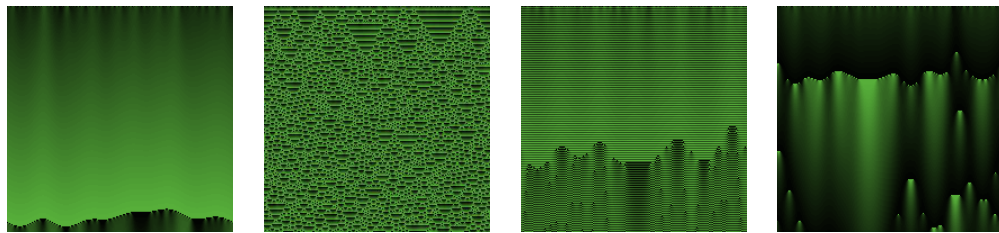
FIGURE 8.2: *Sample snapshots of spatio-temporal patterns from CA Continuous.*

Gas Lattice CA is a system inspired in the Navier-Stokes equation derived from the laws of conservation of mass, momentum and energy. Its dynamics are based on a set

of particles associated with a certain velocity, living in a lattice and moving in a given direction at every time step. Under these conditions, if two particles happen to share the same lattice site, they collide and change their velocity. Four snapshots of the Gas Lattice model are shown in Figure 8.3.
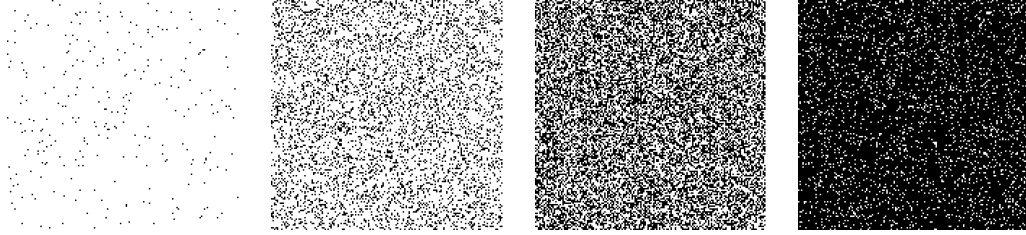
FIGURE 8.3: *Sample snapshots of spatio-temporal patterns from Gas Lattice CA.*

For each of the CAs, all the parameters, except one, were fixed and a number of groups were generated as the variable parameter is altered along a certain range. For each group, the CA was randomly initialised five times, hence resulting in 5 images per group. Each image was labelled with the form xyz_pQ where xyz refers to the CA model, p to the group and Q to the pattern occurrence within that group. For example, $turb\_e5$ refers to the fifth image in group e (i.e. the group generated with the variable parameter at value e) of Turbulence CA. Table 8.1 lists for all the models: the number of groups for each model, the number of images within the group, the NetLogo library and a list of parameters with their names, values and domain. These images are available in Appendix A, Appendix B, Appendix C and Appendix D. Hence, for each of the CAs, an image in turn was considered as a target designoid (T) against which the remaining snapshots of all the groups (T_i) were evaluated on fitness using the USM and on distance using Euclidean difference among the parameters of their associated genotypes (Ind_T and Ind_i) (see Equation 8.3).

	CA Continuous	Gas Lattice CA
#Groups	11	12
#Images	5 per group	5 per group
Library	CA Continuous	Gas Lattice Automaton
Parameter	ADD-CONSTANT	RADIUS
Values	0.004, 0.1, 0.201, 0.301, 0.402, 0.502, 0.603, 0.703, 0.803, 0.9, 0.996	1, 10, 20, 30, 40, 50, 60, 70, 70, 80, 90, 100
Type	Continuous	Discrete
Parameter	PRECISION-LEVEL	DENSITY
Values	16	0
Type	Discrete	Discrete

	Turbulence CA	Meta-automaton CA
#Groups	10	10
#Images	5 per group	5 per group
Library	Turbulence	Own model
Parameter	COUPLING-STRENGTH	K-TIMES
Values	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0	100
Type	Discrete	Discrete
Parameter	ROUGHNESS	RULES
Values	0.001	122, 148, 181, 120, 97, 135, 229, 131, 154, 133
Type	Continuous	Discrete

TABLE 8.1: *Data set names, number of obtained groups per data set, number of images produced per group, name of the NetLogo model, and name of the parameters used for the generation of the spatio-temporal patterns.*

$$f_i = f(T_i) = USM(T_i, T)$$

$$d_i = d(Ind_i, Ind_T) = (\sum_{k=1}^{n}(par_k^i - par_k^T)^2)^{1/2}$$

$$T \text{ is a target snapshot}$$

$$par_k^i \text{ is the } k\text{-parameter encoded in } Ind_i \qquad (8.3)$$

The ranges listed in Table 8.2 show that the computed FDC values go from 0.214095 to 0.534271 in three out of four systems, indicating that the parameter design optimisation is relatively easy for these CA models since the USM values have a relatively high correlation with the associated genotypes of the snapshots. In particular, this information is very well supported by the structures appearing in the scatter plots of Gas Lattice (see group e depicted in Figure 8.4), although other models indicate that the USM may only be effective in certain regions of the search space. For instance, plots of Turbulence may suggest that the USM will be effective as fitness function of a GA only for target designoids with COUPLING-STRENGTH values between 0.400 and 0.800 (see group f in Figure 8.5). Further plots in Appendix E, Appendix F and Appendix G.

CA Model	FDC range	CA Model	FDC range
Continuous	[0.231396, 0.483353]	Turbulence	[0.214095, 0.330793]
Gas Lattice	[0.351766, 0.534271]	Meta-automata	[-0.199424, 0.612101]

TABLE 8.2: *Resulting maximum and minimum fitness distance correlation values for CA Continuous, Turbulence, Gas Lattice and Meta-automata CA models.*
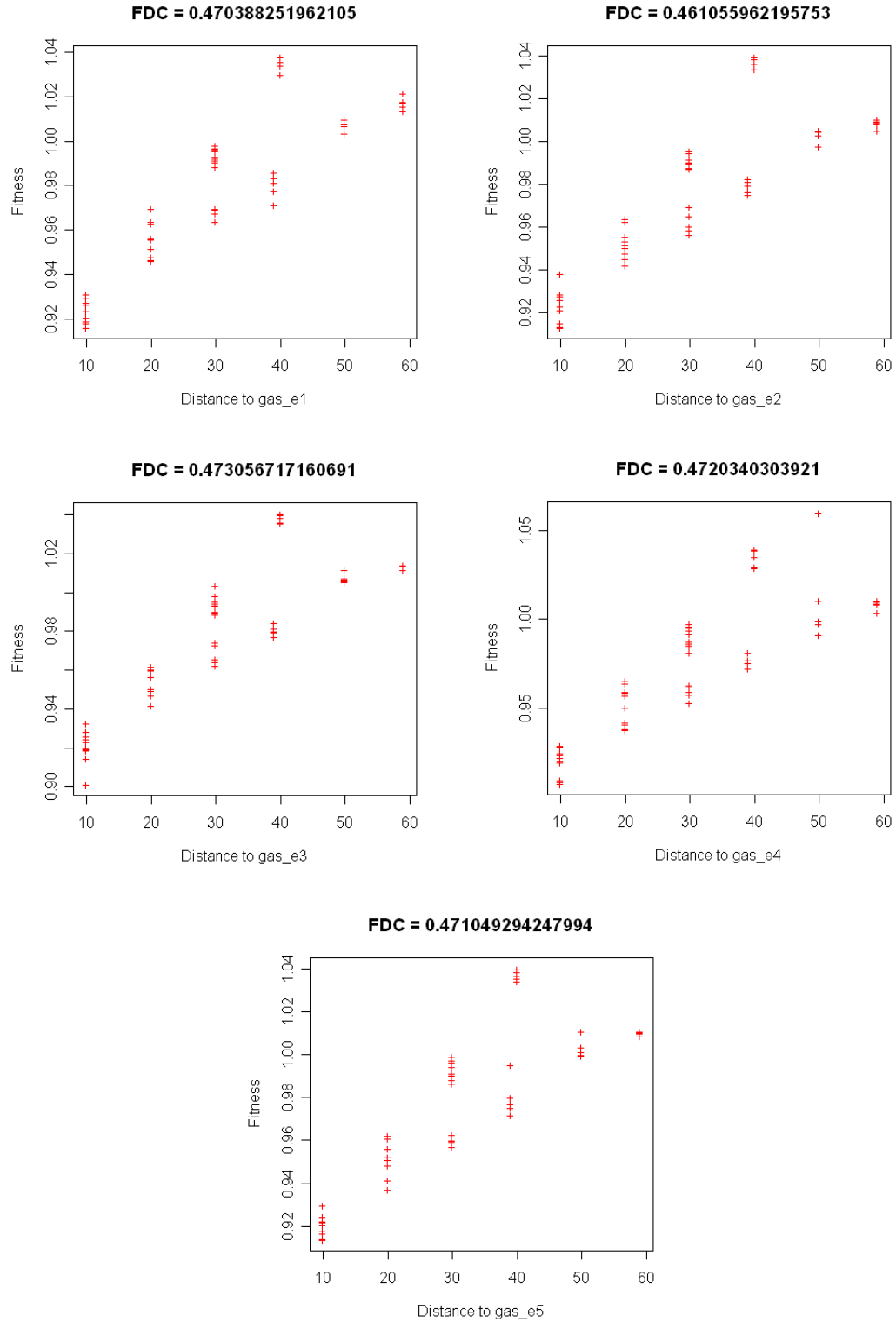
FIGURE 8.4: *Graphics of the resultant scatter plots and correlation coefficients for the group **e** of Gas Lattice model showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*
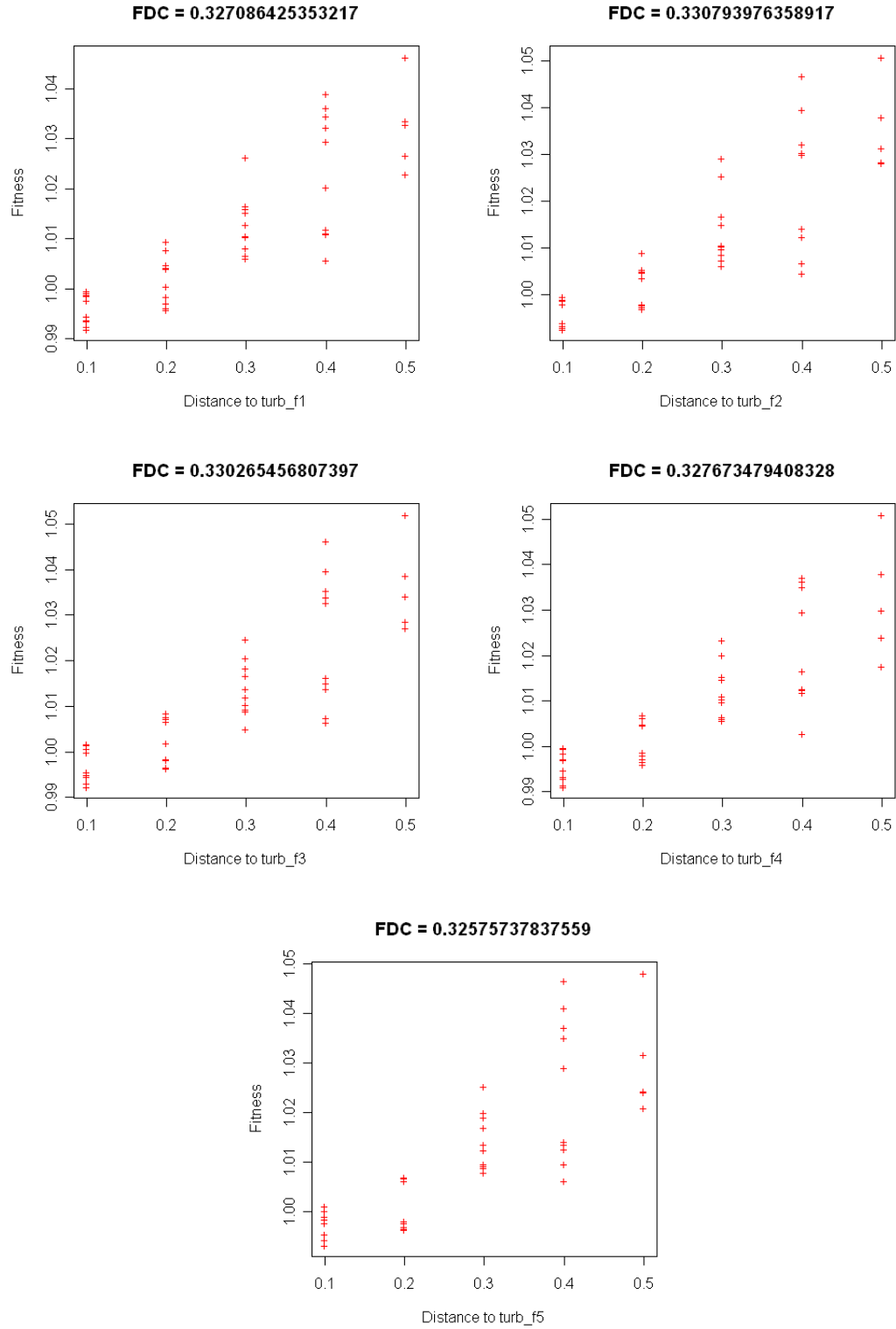
FIGURE 8.5: *Graphics of the resultant scatter plots and correlation coefficients for the group f of Turbulence model showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*

On the other hand, the findings obtained after applying the FDC to the Meta-automaton is more diverse. In this case, the figures range in total from -0.199424 to 0.612101, from where only 20% of the experiments reveal that USM values have a high correlation with the associated genotypes of the snapshots. In particular, the FDC values lay in $[-0.199424, 0.159087]$ for groups $metaK1\_bQ$, $metaK1\_cQ$, $metaK1\_gQ$ and $metaK1\_iQ$, for $1 \leq Q \leq 5$, indicating that it is difficult for the GA to achieve the optimum rule for any given snapshot of these four. However, the level of difficulty decreases when analysing groups $metaK1\_aQ$, $metaK1\_dQ$, $metaK1\_fQ$ and $metaK1\_hQ$, for $1 \leq Q \leq 5$, as their correlation values range in $[0.252358, 0.591984]$. In addition, any snapshot belonging to either group $metaK1\_eQ$ or group $metaK1\_jQ$, for $1 \leq Q \leq 5$, would be easier to evolve since their FDC values range in $[0.446261, 0.612101]$ (see group j depicted in Figure 8.6). Most of these findings match with the results obtained by the GA in Chapter 7 although some counter examples were found. For instance, the GA has found the correct rule when given $metaK1\_c1$ (T_3^1) as the target snapshot whereas the FDC analyses reported correlation coefficients which suggest that the problem is difficult to solve over five instances of this group. Conversely, the GA was not able to achieve a designoid when $metaK1\_f1$ (T_6^1) was set as the target snapshot although the FDC analyses concluded that the GA may successfully treat the problem. A special point to highlight is that the evolved mirror snapshot achieved for $metaK1\_d1$ (T_4^1) is indeed supported by the FDC analyses, although the same line of reasoning indicates the opposite in the case of $metaK1\_b1$ (T_2^1). Further scatter plots are available in Appendix H.
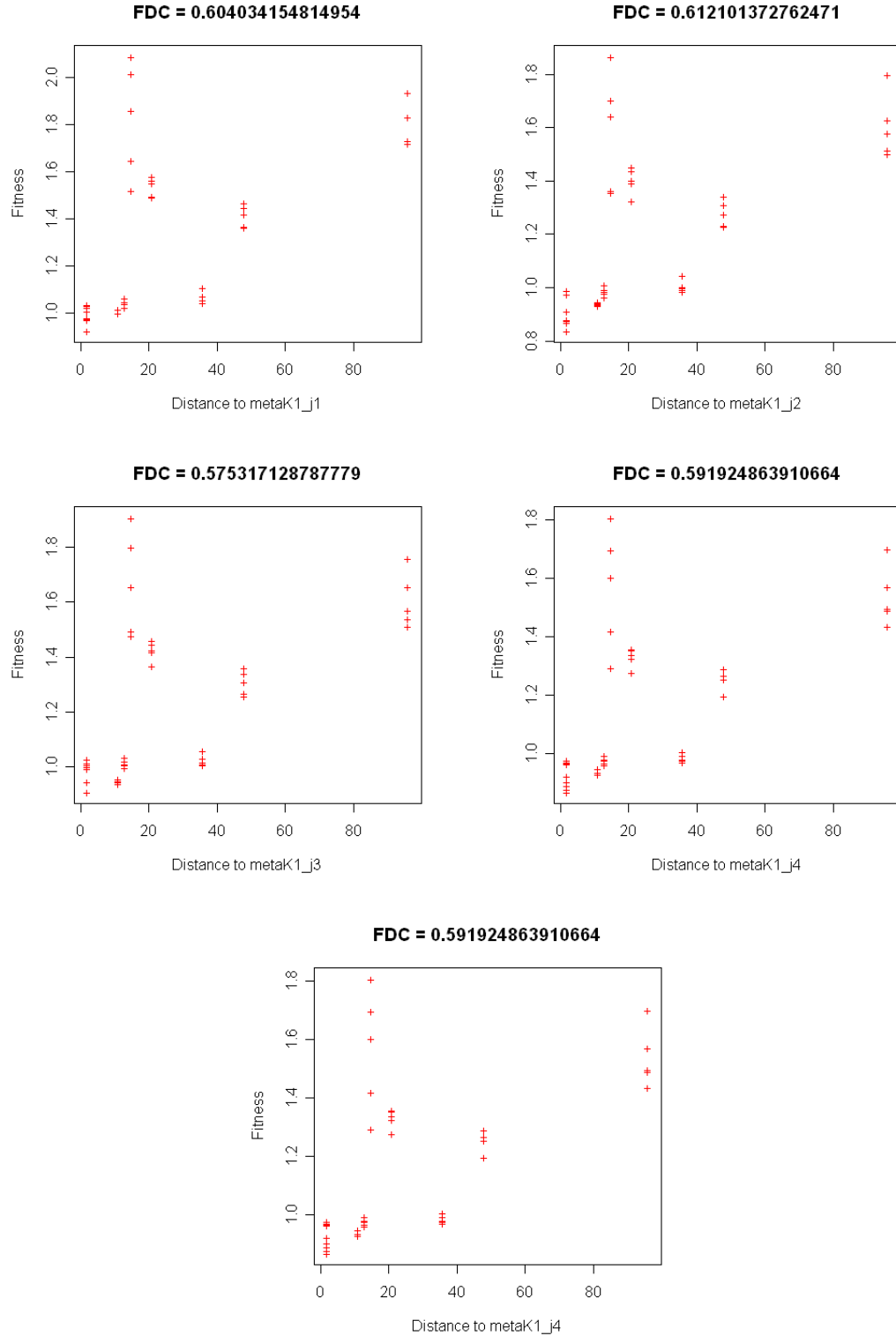
FIGURE 8.6: *Graphics of the resultant scatter plots and correlation coefficients for the group* **j** *of Meta-automaton model showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*

8.3 Correlating the Self-Assembly Wang Tiles Design

Another highly complex, non linear and stochastic relationship among mappings from genotype to phenotype and then from phenotype to fitness is observed when approaching the self-assembly Wang tiles design optimisation problem seen in Chapter 7. The three-stage mapping process from genotype (the set of tiles) onto phenotype through the execution of the complex system (simulator) and then from the assembled aggregates onto a numerical value via the objective function (the assembly assessment) is shown in Figure 8.7.
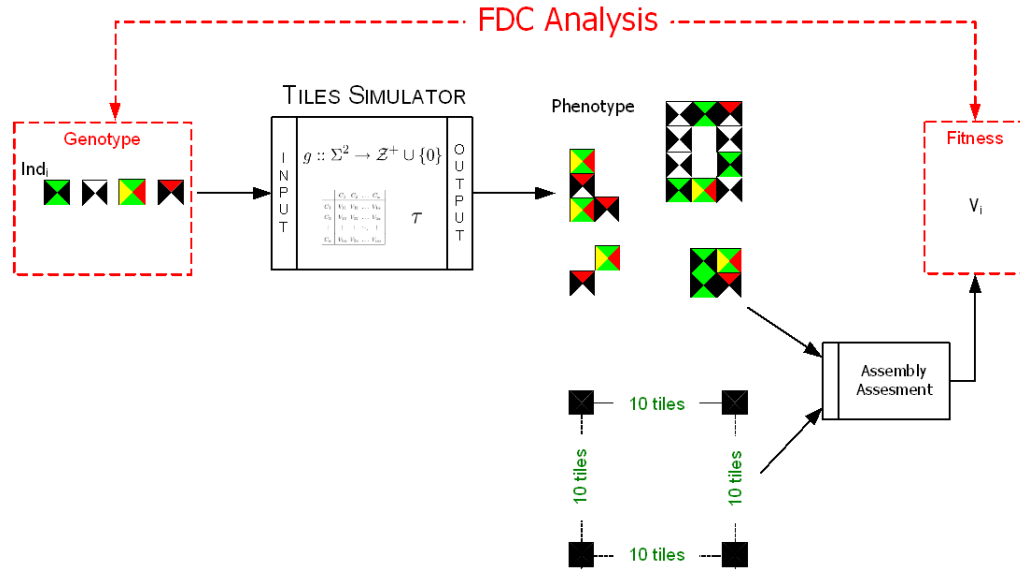
FIGURE 8.7: *Diagram of mappings from genotype onto phenotype and from phenotype onto numerical fitness value, and relationship to the Fitness Distance Correlation.*

Since different set of tiles may self-assemble in aggregates similar in shape to the target structure, it is again of interest to study how effectively the fitness of an individual correlates to its genotypic distance to a known optimum. Given that Minkowski funtionals represented the best approach, the following section pursues its FDC analysis in order to study its effectiveness as fitness function for the evaluation of the achieved aggregates.

8.3.1   Experiments and Results

In order to perform a FDC analysis, the best individual found by the Minkowski functionals approach when using probabilistic criteria and no rotation (Section 7.1.3 of Chapter 7) has been chosen. A data set comprising 500 individuals at different Hamming distances from the best individual was created. In particular, given two individuals Ind_i and Ind_j of same length, their Hamming distance H is defined as in Equation 8.4.

$$H(Ind_i, Ind_j) = \sum_{k=1}^{n} diff(T_k^i, T_k^j)$$

$$diff(T_i, T_j) = \sum_{l=0}^{3} c_l^i \ominus c_l^j$$

$$c_a \ominus c_b = \begin{cases} 1 & \text{if } c_a \neq c_b \\ \\ 0 & \text{otherwise} \end{cases}$$

$$c_a, c_b \in \Sigma \tag{8.4}$$

Thus, this 500 individuals data set comprises all the possible chromosomes at Hamming distance of 1 plus some other randomly generated individuals at greater distance, all of these systematically generated following the pseudocode described in Algorithm 1 where $DuplicateReplacing(T_k, c_l, c_{new})$ duplicates tile T_k replacing colour c_l with c_{new}, $DuplicateReplacing(Ind_i, T_k, T_{new})$ duplicates individual Ind_i replacing tile T_k with T_{new}, $TileAt(Ind_i, k)$ returns the tile at position k of an individual, and $Replace(T_k, c_l, c_{new})$ replaces colour c_l in tile T_k with colour c_{new}.

Algorithm 1 GenerateIndividuals

Input: Ind an individual

Output: S a set of individuals

1. **for all** tiles T_k in Ind_i **do**

2.   **for all** colours c_l in T_k **do**

3.   **for all** colour $c_{new} \in \Sigma$ **do**

4. $T_{new} \leftarrow DuplicateReplacing(T_k, c_l, c_{new})$

5. $Ind_{new} \leftarrow DuplicateReplacing(Ind_i, T_k, T_{new})$

6. $Insert(S, Ind_{new})$

7.   **end for**

8.   **end for**

9. **end for**

10. **while** $\mid S \mid < 500$ **do**

11. $Ind_{new} \leftarrow Duplicate(\text{randomly chosen } Ind_i \in S)$

12. $n \leftarrow Random(0, \mid Ind_{new} \mid)$

13.   **for all** k to n **do**

14. $T_k \leftarrow TileAt(Ind_{new}, k)$

15. $m \leftarrow Random(0, 3)$

16.   **for all** l to m **do**

17. $c_{new} \leftarrow Random(\Sigma \setminus c_l)$

18. $Replace(T_k, c_l, c_{new})$

19. $Insert(S, Ind_{new})$

20.   **end for**

21.   **end for**

22. **end while**

In total, each of the generated individuals was simulated 5 times giving as a result a group with equal number of final configurations. Thus, a configuration in turn was considered as a target $(Conf_T)$ against which the remaining configurations of all the groups $(Conf_i)$ were evaluated on fitness (f_i) and on distance (d_i) among their associate genotypes (see Equation 8.5).

$$d_i = H(ind_i, ind_T)$$

$$f_i = f(Conf_i) = Eval(Conf_i, Conf_T) = \sqrt{(\Delta\mathcal{A})^2 + (\Delta\mathcal{P})^2 + (\Delta\mathcal{N})^2}$$

$$\Delta\mathcal{A} = max\{A_1^T, \ldots, A_m^T\} - max\{A_1^i, \ldots, A_n^i\}$$

$$\Delta\mathcal{P} = \sum_{k=1}^{m} P_k^T - \sum_{k=1}^{n} P_k^i$$

$$\Delta\mathcal{N} = m - n \qquad (8.5)$$

Since a configuration comprises a collection of aggregates, a way is needed to perform an evaluation involving all its aggregations collectively. For this reason, considering a target configuration $Conf_T$ and an arbitrary one $Conf_i$ with aggregates $\{A_1^T, \ldots, A_m^T\}$ and $\{A_1^i, \ldots, A_n^i\}$ respectively, $Conf_i$ will be evaluated upon $Conf_T$ in terms of the difference in areas, perimeters and number of achieved aggregations as shown in Equation 8.5.

After performing the calculations, the findings show that the FDC values range from -0.331444 to 0.281457. Since Equation 8.5 defines a minimisation, 50.60% of the FDC values indicate that using a GA may not be effective, 44.72% that the problem is difficult to solve and a 4.68% that the GA may successfully treat the problem (see Figure 8.8). In
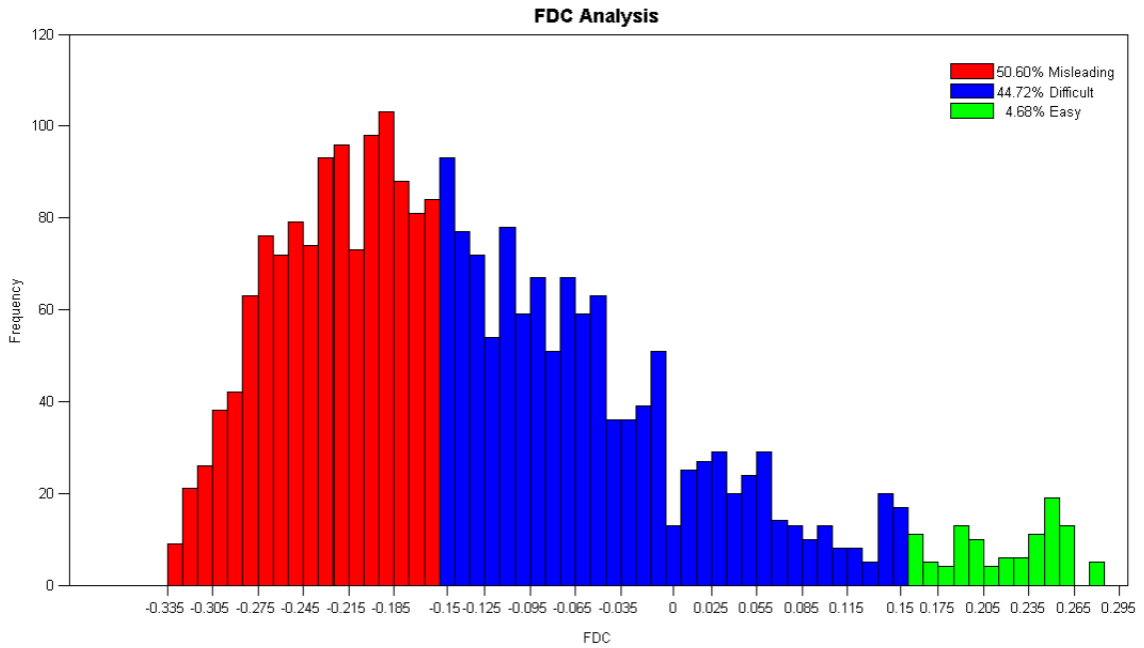
FIGURE 8.8: *Proportion of FDC values falling into difficult, misleading and easy to solve categories. From the 2500 analyses performed over 500 individuals, only a 4.68% reveals that a GA may successfully treat the problem.*

particular, visual inspections over scatter plots obtained from the values captured into the smallest percentage depict good correlation on 3 individuals (see Figure 8.9). Hence, from the sampling of 500 individuals and 2500 simulations subject to FDC analyses, it emerges that employing Minkowski functionals as fitness function does not offer a proper correlation upon the relationship genotype-fitness for half of the putative samples.

Contrary to the interpretations given by some of the FDC figures seen in this section, the results reported in Chapter 7 reveal that using Minkowsi functionals as evaluation method of a GA has positively addressed the self-assembly Wang tiles design problem. Henceforth, analyses on the phenotype-fitness relationship are expected to shed light on the reasons for which such evolutionary approach has been effective.
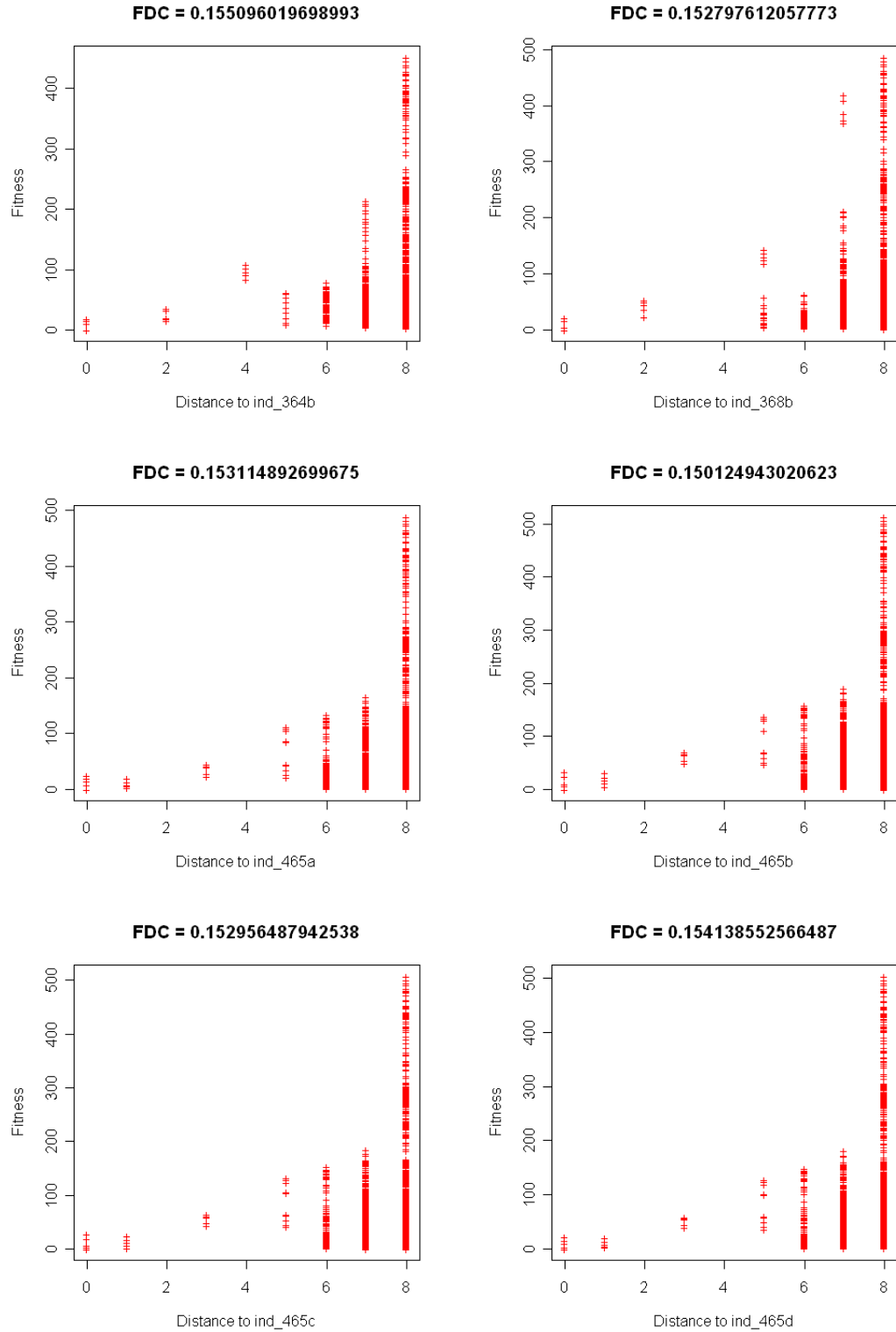
FIGURE 8.9: *Graphics of the resultant scatter plots and correlation coefficients for the self-assembly Wang tiles model showing that the Minkowski functionals has a relatively satisfactory correlation with the genotype for some of the self-assembly Wang tile families.*

8.4   Cluster Analysis

In all the problems investigated in this thesis, several different genotypes can encode the same phenotype and hence introduce severe noise in the FDC analysis. As, ultimately, selection is based on fitness which in turn depends on the phenotype, studying the phenotype-fitness mapping could shed light on why the GAs worked quite well, even though in some cases FDC says it should not. For this reason, this section introduces clustering as a method for analysing the phenotype-fitness relationship in both CAs design optimisation and self-assembly Wang tiles design optimisation.

Cluster analysis or clustering is a technique for grouping objects according to their similarities [191] [192] [193] [194]. In contrast to classification, clustering is an unsupervised task in which a set of objects is partitioned in groups, called clusters, according to their proximities such that those belonging to a cluster are more similar to each other than objects in a different cluster. A clustering procedure comprises four basic stages: feature selection or extraction, clustering algorithm selection, cluster validation and results interpretation. In the first stage, the features by which the objects will be distinguished are chosen. This pairwise affinity is then considered to compute a proximity matrix to which a cluster strategy is applied. The resulting partition of the data is subject to a subsequent testing criteria in order to validate the clustering process. Finally, a visualization and interpretation over the achieved clusters closes the procedure with the hope of providing meaningful insights coming from the original data. The whole inter-relation among these tasks can be seen as a sequential procedure as detailed and commented in Figure 8.10.

Which are the
attributes of
interest ?

Proximity matrix
of pairwise
alikeness

Feature
Selection

Interobject
Comparison

Data

Application of
clustering
algorithm

Clustering

Results
Interpretation

Validation

Meaningful
insight of data

Is the data
partition ok ?
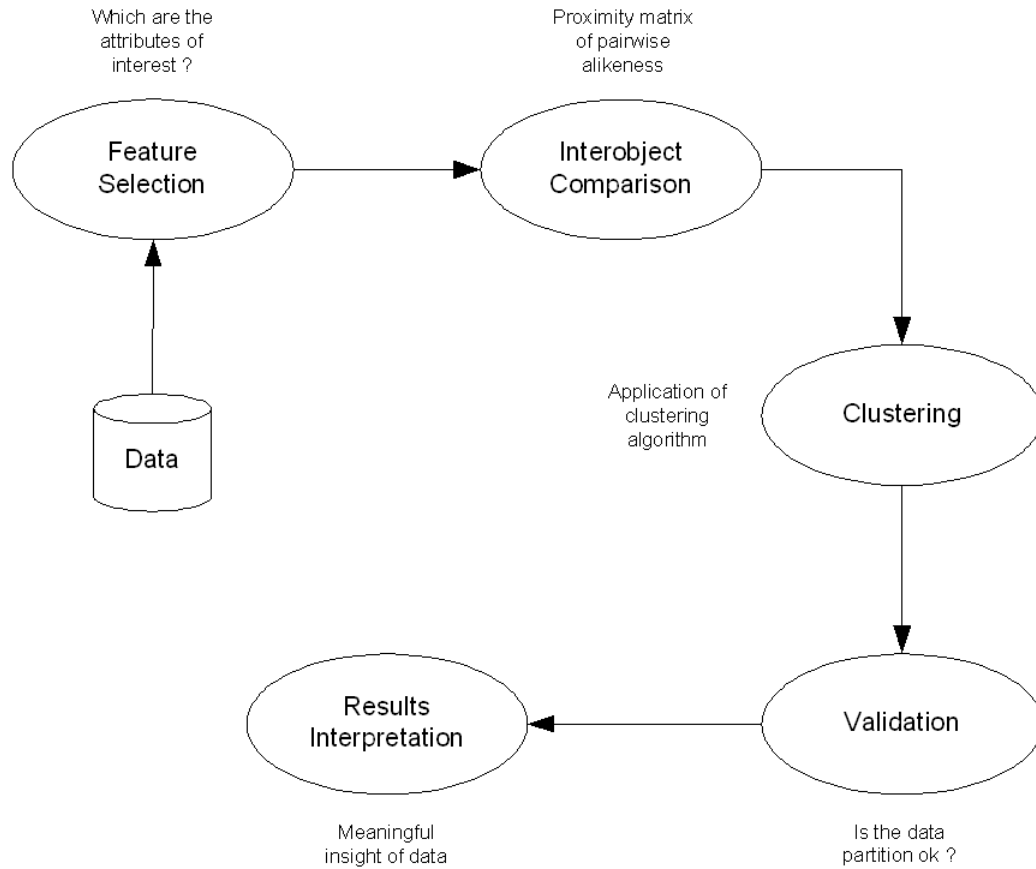
FIGURE 8.10: *Clustering procedure comprising feature selection, inter-object comparison, clustering, validation of data partition and results interpretation.*

Clustering algorithms are classified as hierarchical, partitioning, density-based partitioning, grid-based, evolutionary methods and so forth in [194] [195] [196]. Although there are slight differences among the proposed taxonomies, many are the common features associated with them [192]. For instance, according to their structure and operation, a clustering algorithm is agglomerative if clusters arise from singletons (bottom-up) or divisive if one super cluster is split in several ones (top-down). The sequential or simultaneous use of object features in the clustering process also plays an important role as it defines whether the

algorithm is monothetic or polythetic. Additionally, it is also reported that some methodologies allow objects to belong to a single cluster (hard classification) or to multiple clusters (fuzzy classification). Besides these three characterizations, a clustering algorithm can also be deterministic, stochastic, incremental or non-incremental if there are constraints on execution time or memory affecting the architecture of the algorithm.

The clustering methods appearing in the literature are, mainly, variants of the hierarchical agglomerative clustering. Among them, the single-link (SLINK), complete-link (CLINK) or minimum-variance [197] are the best-known where their differences lay on the way they characterise the similarity between pairs of clusters. In addition, the Unweighted Pair Group Method using arithmetic Average (UPGMA), Weighted Pair Group Method using arithmetic Average (WPGMA), the Unweighted Pair Group Method using Centroids (UPGMC) and the Weighted Pair Group Method using Centroids (WPGMC) are also broadly employed in many applications [198] [199] [200].

8.5 Clustering the Cellular Automata Snapshots

As pointed out in Section 8.2, the mapping genotype-phenotype and phenotype-fitness is a highly complex, non-linear relationship. In the CA-based systems previously investigated, there is a mapping process that goes from genotype (the real numbered parameters) onto phenotype through the execution of the CA model itself, and from this (the spatio-temporal snapshot) onto a numerical fitness value computed by the USM (see Figure 8.11).
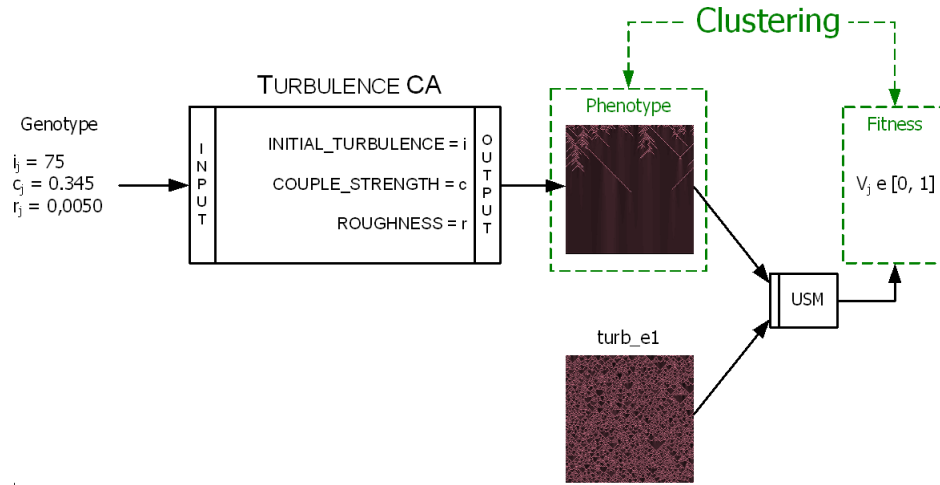
FIGURE 8.11: *Diagram of mappings from genotype onto phenotype and from phenotype onto numerical fitness value, and relationship to clustering analysis.*

Although FDC has been used to assess the quality of the representation vis-à-vis the fitness function, e.g. [201], it has never been combined with a clustering process to obtain better insight of the complex mapping described above. For this reason, to verify the phenotype-fitness relationship, this part of the research proposes the use of clustering and for this to be effective, the feature selection and clustering algorithm to be employed are presented in the next subsection. It is important to remark that the aim here is not to uncover any structure among the data since this is known *a priori* by their construction.

8.5.1 Experiments and Results

In order to proceed with the clustering experiments, the CA Continuous, Turbulence, Gas Lattice and Meta-automata snapshots generated for the FDC analysis in Section 8.2.1 are employed. As these CA spatio-temporal snapshots comprise the clustering input data, the feature selection for which objects are expected to be distinguished is their information

content. Thereby, the measure of affinity between every pair of snapshots (T_i, T_j) is computed in terms of USM, values of which are stored in a symmetric matrix M_s of $n \times n$ (n is the number of snapshots) as defined in Equation 8.6. Although a number of different clustering methods and representations are available, the unweighted pair-group method using arithmetic average (UPGMA) [202] has been chosen along with a logarithm dendogram representation to visualize and interpret the data partition.

$$M_s[i,j] = \begin{cases} USM(T_i, T_j) & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases}$$

M_s is a proximity matrix s.t. $M_s[i,j] = M_s[j,i]$

T_i, T_j are snapshots

$$1 \leq i, j \leq n \tag{8.6}$$

The pseudocode for the clustering algorithm is UPGMA outlined in Algorithm 2 where $MergeRows$ joins the content of row i and row j, $MergeColumns$ joins the content of column i and column j and $MakeNode$ associates i and j in a node that $InsertNode$ will add to the hierarchical structure T.

Algorithm 2 MakeCluster

Input: M_s a proximity symmetric matrix

Output: T a hierarchical structure

1. **while** $dimension(M_s) \geq 3 \times 3$ **do**

2.   **for all** i, j such that $i \neq j$ **do**

3. $minimum \leftarrow min(M_s[i, j], Minimum)$

4.   **end for**

5. $MergeRows(M_s, i, j)$

6. $MergeColumns(M_s, i, j)$

7. $M_s[k, ij] \leftarrow avg(M_s[k, i], M_s[k, j])$

8. $M_s[ij, k] \leftarrow avg(M_s[i, k], M_s[j, k])$

9. $node \leftarrow MakeNode(i, j)$

10. $InsertNode(T, node, minimum)$

11. **end while**

The first experiment was conducted over CA Continuous snapshots. The resulting data partition in which eleven clusters have been clearly created is shown in Figure 8.12. An important characteristic to note within this structure is the homogeneity of each of the eleven partitions, i.e. each branch comprises labels of the form cac_pQ where p is fixed and Q varies from 1 to 5. More important, this fact reveals that there is a strong correlation phenotype-fitness since the USM has been capable of detecting similarities among snapshots of the same creational group as well as dissimilarities among those created with different parameter values. Each branch shows a representative snapshot of each image class.
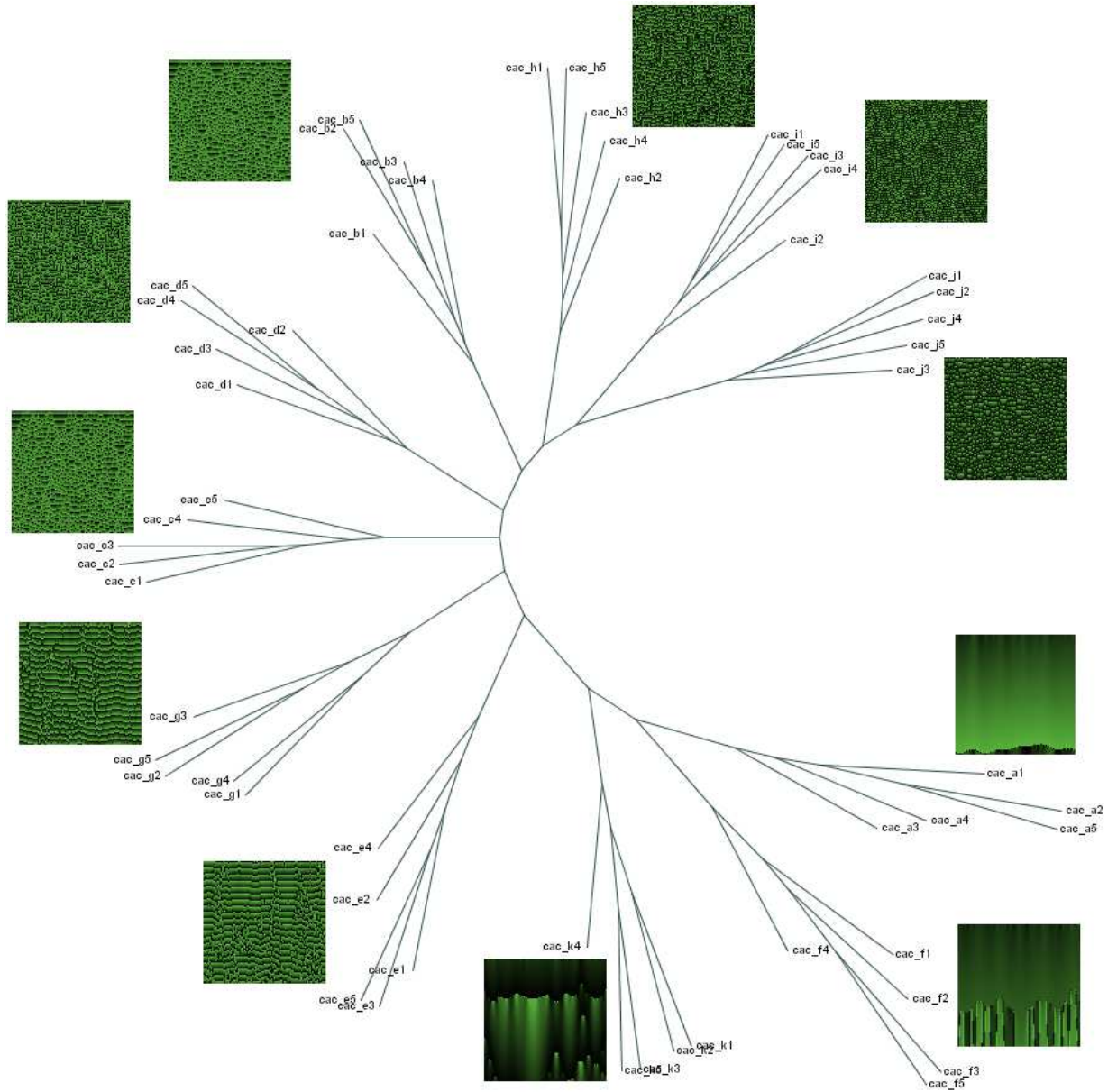
FIGURE 8.12: *Illustration of the logarithmic cluster tree for snapshots belonging to the CA Continuous model. Clustering the fifty-five spatio-temporal patterns of this model have generated the expected clusters each of which corresponds to the eleven creational groups.*

An outcome with similar characteristics was achieved after processing Turbulence snapshots. The resulting partition of data is depicted in Figure 8.13 where a number of ten different clusters is easy to visualize. As previously seen, these groups are also homogeneously constructed, i.e. each cluster contains labels of the form $turb\_pQ$ with group id (p) fixed and occurrence number (Q) varying from 1 to 5. Despite this result, it is interesting to note that there is an outlying snapshot ($turb\_c4$) along the central stem of the dendogram. Although this misplacement may suggest a total dissimilarity to its peers, the assigned location still reflects a certain degree of similarity since the snapshot is not far from the cluster where those of the same creational group are placed.

An additional positive finding was obtained after processing Gas Lattice snapshots. The resulting data partition shown in Figure 8.14 reveals that eleven homogeneous clusters were created with a dendogram topology rather different from the ones obtained in the two previous experiments. In this case, the central axis gives origin to six stems from where binary branches extend the clusters. A remarkable fact to notice is that the clusters are sequentially arranged. That is, a series of clusters lat_aQ, lat_bQ, lat_cQ, lat_dQ, lat_eQ, lat_fQ, lat_gQ, lat_hQ, lat_iQ, lat_jQ, lat_kQ and lat_lQ, for $1 \leq Q \leq 5$, is found in alphabetic order starting from lat_aQ and going clockwise. This interesting distribution resembles indeed the order in which the snapshots of this model have been created.
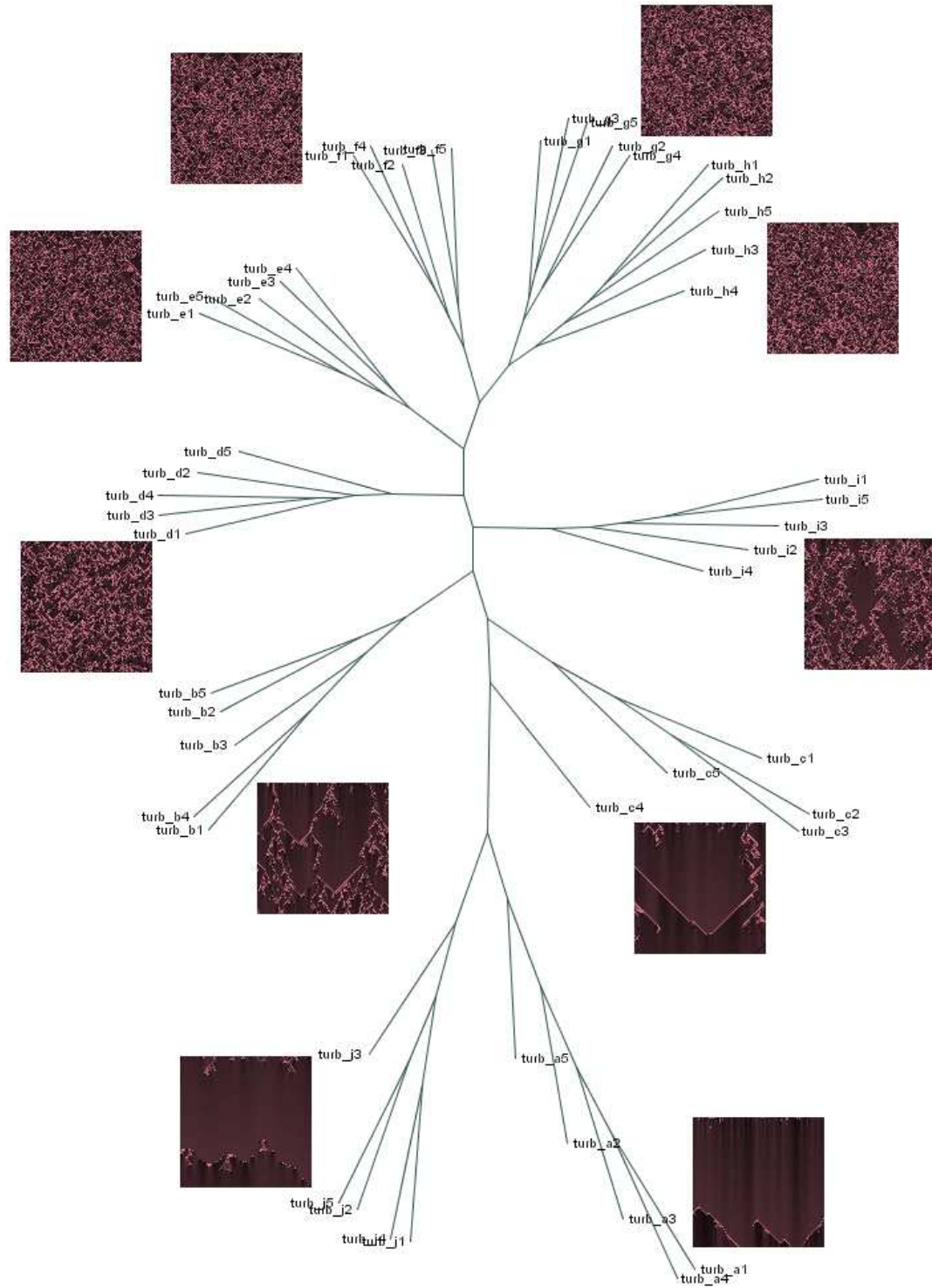
FIGURE 8.13: *Illustration of the logarithmic cluster tree for snapshots belonging to the Turbulence model. Clustering the fifty spatio-temporal patterns of this model have generated the expected clusters each of which corresponds to the ten creational groups.*
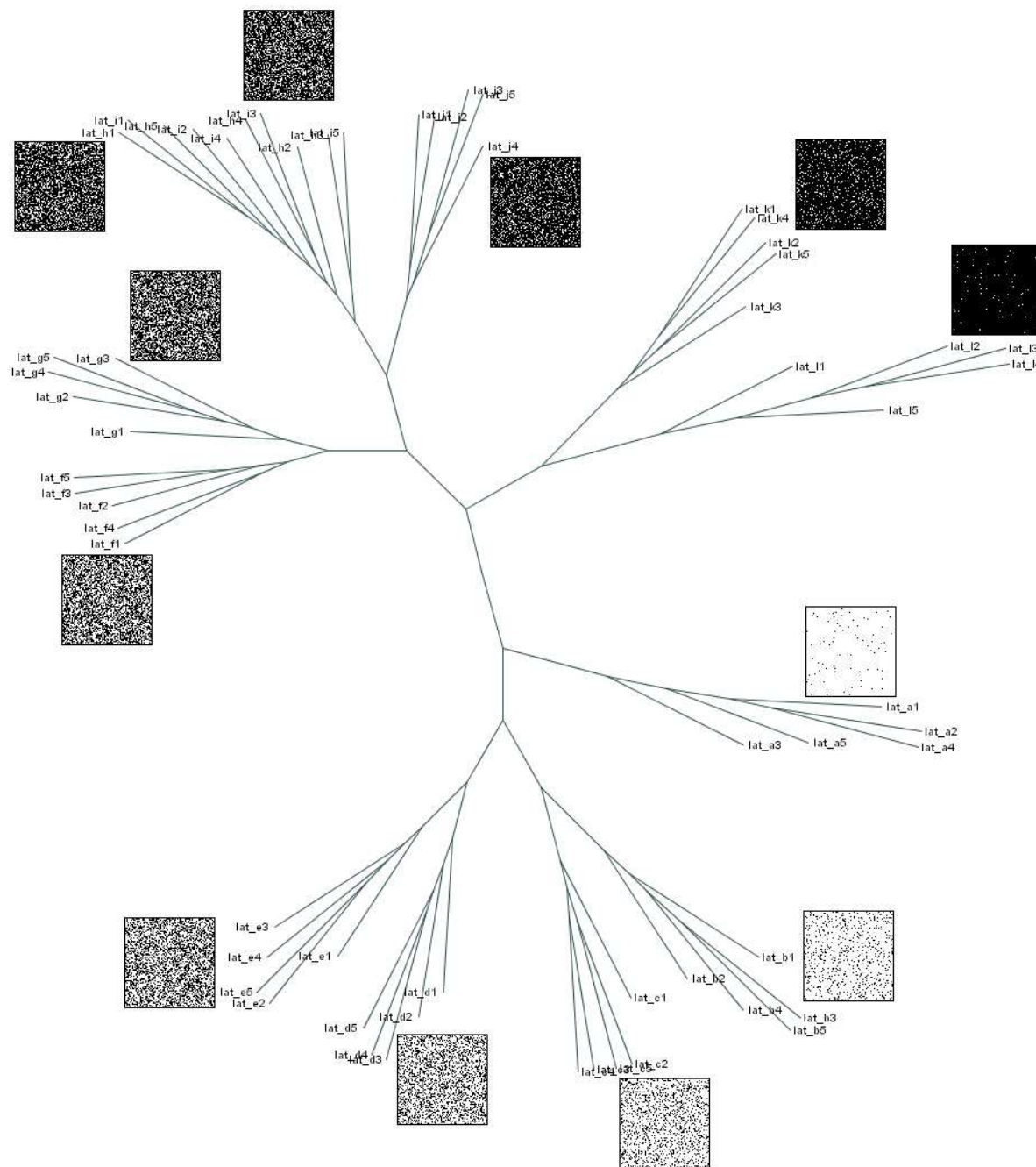
FIGURE 8.14: *Illustration of the logarithmic cluster tree for snapshots belonging to the Gas Lattice model. Clustering the sixty spatio-temporal patterns of this model have generated the expected clusters each of which corresponds to the eleven creational groups.*

In order to subject the USM to a more challenging assessment, the total number of snapshots corresponding to the three systems studied below were taken together for clustering. Hence, considering n snapshots of CA Continuous (T_i^{CAC}), m snapshots of Turbulence (T_r^{TUR}) and l snapshots of Gas Lattice (T_s^{LAT}), the proximity matrix M_s storing inter-snapshot similarities is now generated as defined in Equation 8.7.

$$M_s[i,j] = \begin{cases} USM(T_i^{CAC}, T_j^{CAC}) & \text{if } i \neq j \wedge 1 \leq i, j \leq n \\ USM(T_i^{CAC}, T_r^{TUR}) & \text{if } i \neq j \wedge 1 \leq i \leq n < j \leq n + m \\ USM(T_i^{CAC}, T_s^{LAT}) & \text{if } i \neq j \wedge 1 \leq i \leq n \wedge n + m < j \leq l + n + m \\ USM(T_r^{TUR}, T_t^{TUR}) & \text{if } i \neq j \wedge n < i, j \leq n + m \\ USM(T_r^{TUR}, T_s^{LAT}) & \text{if } i \neq j \wedge n < i \leq n + m < j \leq n + m + l \\ USM(T_s^{LAT}, T_u^{LAT}) & \text{if } i \neq j \wedge n + m < i, j \leq n + m + l \\ 0 & \text{if } i = j \end{cases}$$

$$M_s \text{ is a proximity matrix s.t. } M_s[i,j] = M_s[j,i]$$

$$1 \leq i, j \leq n + m + l \wedge 1 \leq r, t \leq m \wedge 1 \leq s, u \leq l \quad (8.7)$$

Thus, after applying Algorithm 2 to M_s, the resulting cluster analysis reveals that inter-system and intra-system characteristics were very well captured by the USM. On the one hand, the features captured at inter-system level are manifested by the creation of three clear partitions of the data, in which the Turbulence objects (red), the CA Continuous ones (green) and the Gas Lattice instances (blue) are located (Figure 8.15).
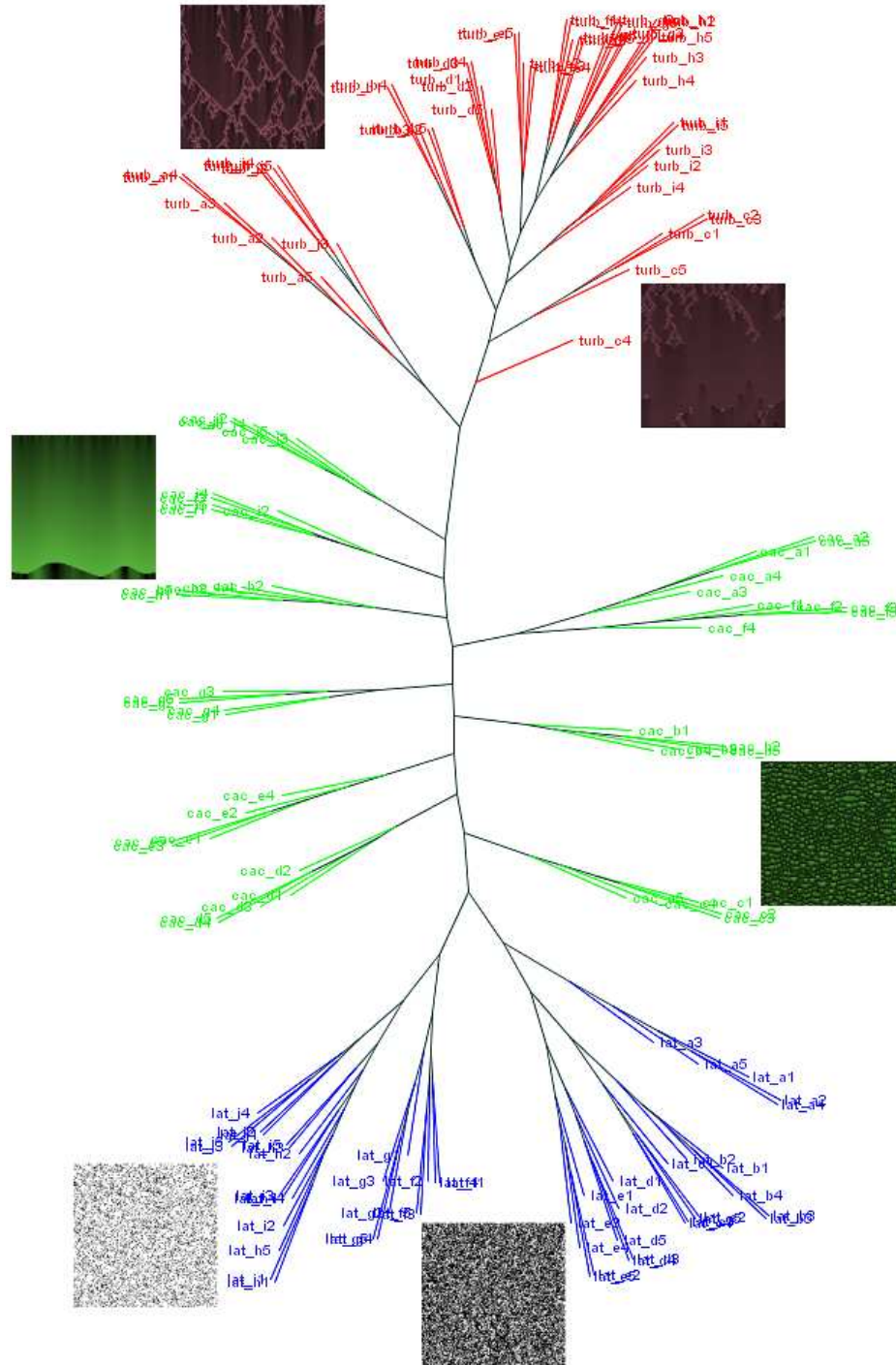
FIGURE 8.15: *Illustration of the logarithmic cluster tree for snapshots belonging to the Turbulence, CA Continuous and Gas Lattice models. The characteristics of the three different collections were detected giving three different logical partitions locating the Turbulence model objects in the top, the CA Continuous in the middle and the Gas Lattice model instances at the bottom.*

These divisions along the dendogram aim to support that the information contained in each of the three CA models is indeed different from one another, but shared among the snapshots belonging to the same system. On the other hand, the features captured at intra-system level matches with the ones found when the systems were analysed individually, i.e. the snapshots within a system belonging to the same creational group have been placed together. For example, the ten clusters containing five snapshots each (a to j) when observing the central stems (green) correspond to the CA Continuous. Both the inter-system and intra-system characterizations suggest that the USM is a successful, direct proximity measure capable of making distinction not only in terms of information content but also, in this case, in terms of colours as the systems' snapshots are colour-based patterns.

Two supplementary experiments employing Meta-automaton snapshots were conducted. In both cases, the Meta-automaton spacial dynamics was divided in two (K-TIMES $= 50$). That is, cells were divided in groups of 50 consecutive cells to which a rule from the pool of 256 elementary rules was applied. In order to conduct the first experiment, an algorithm that receives a pair of elementary rules (r_1, r_2) as input and generates a set of 16-pairs of elementary rules $S = \{(r_a, r_b) \mid r_a \neq r_b \wedge r_a, r_b \in [0, 255]\}$ in terms of mutation and Cartesian product was implemented. The complete pseudocode of this process is shown in Algorithm 3 where $MutateRule(r_i)$ returns an elementary rule in $[0, 255] \setminus \{r_i\}$, $Insert(S, elem)$ inserts an element $elem$ in S and $Get(S)$ returns a copy of an element included in S. In particular, consider that $Get(S)$ in line 16 returns a different element for each of the values that j takes per iteration of the i-loop.

Algorithm 3 RulesCreator

Input: (r_1, r_2) a pair of elementary rules $r_1, r_2 \in [0, 255]$

Output: S a set of pairs of elementary rules

1. $S_a \leftarrow \{\}$

2. $S_b \leftarrow \{\}$

3. $S \leftarrow \{\}$

4. $Insert(S, (r_1, r_2))$

5. **for all** $i \in [1, 3]$ **do**

6. $r_a \leftarrow MutateRule(r_1)$

7. $Insert(S_a, (r_a, r_2))$

8. $Insert(S, (r_a, r_2))$

9. $r_b \leftarrow MutateRule(r_2)$

10. $Insert(S_b, (r_1, r_b))$

11. $Insert(S, (r_1, r_b))$

12. **end for**

13. **for all** $i \in [1, 3]$ **do**

14. $(r_a, r_2) \leftarrow Get(S_a)$

15.   **for all** $j \in [1, 3]$ **do**

16. $(r_1, r_b) \leftarrow Get(S_b)$

17. $Insert(S, (r_a, r_b))$

18.   **end for**

19. **end for**

Hence, considering the arbitrary pair of rules $(60, 102)$ as input of Algorithm 3, 16 pairs of rules shown in Figure 8.16 were obtained. Afterwards, for each $(r_a, r_b) \in S$, groups of five randomly initialised, spatio-temporal patterns were generated assigning r_a and r_b to the first and second 50-cells slot of the Meta-automaton. Each of these patterns was captured in snapshots labelled as $meta\_pQ$ where p refers to the group and Q to the occurrence within that group. Representatives and their rules are shown in Figure 8.16.

meta_p1 / (60, 120)   meta_a1 / (122, 102)   meta_b1 / (167, 102)   meta_c1 / (161, 102)

meta_d1 / (60, 129)   meta_e1 / (60, 105)   meta_f1 / (60, 103)   meta_g1 / (122, 129)

meta_h1 / (122, 105)   meta_i1 / (122, 103)   meta_j1 / (167, 129)   meta_k1 / (167,105)

meta_l1 / (167, 103)   meta_m1 / (161, 129)   meta_n1 / (161, 105)   meta_o1 / (161, 103)
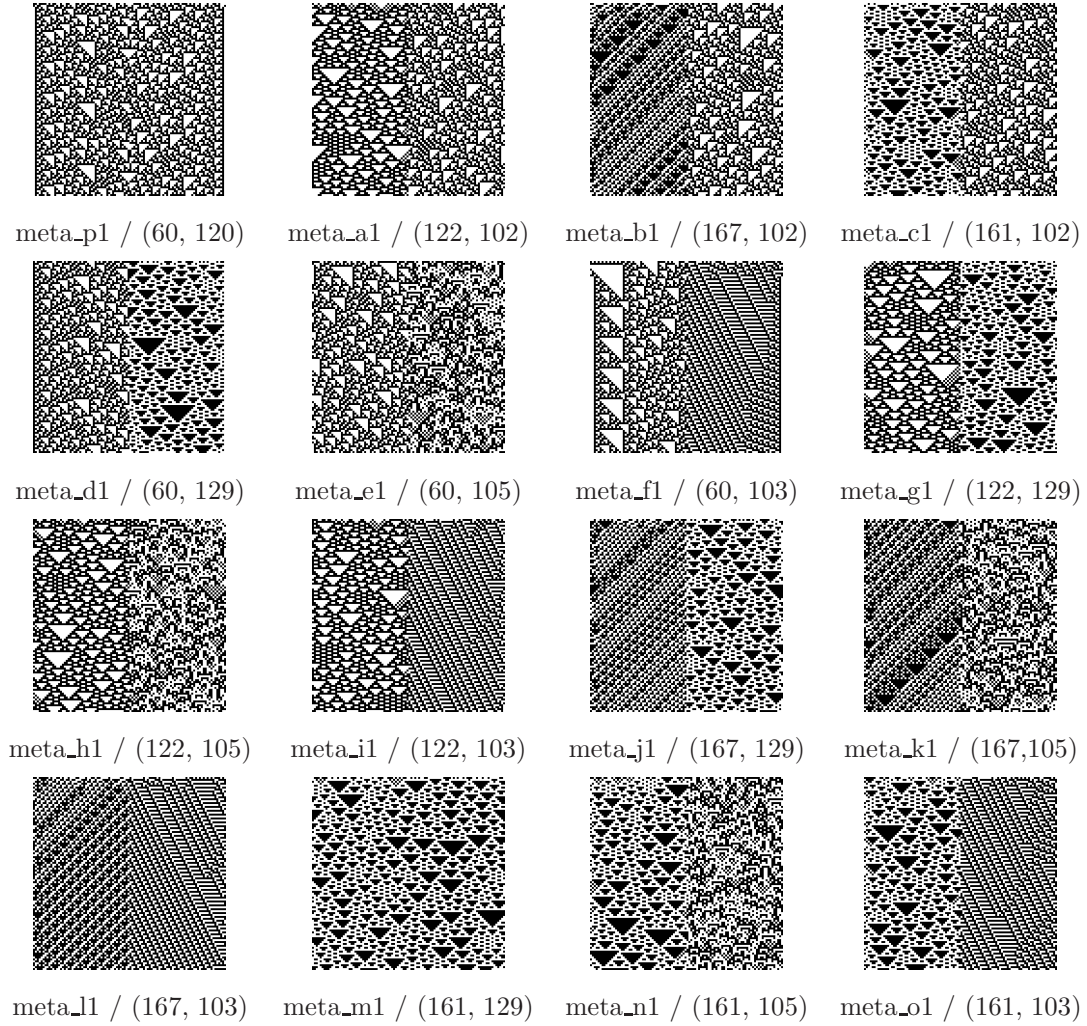
FIGURE 8.16: *Sixteen resulting snapshots after processing* $(60, 102)$ *with Algorithm 3. Labels of the form* xyz_pQ */* (r_a, r_b) *indicate group and associated creational rules.*

As in the first three experiments, an inter-snapshots proximity matrix M_s was built and set as input of the UPGMA clustering algorithm. This two-steps process gave as a result a data partition in Figure 8.17 revealing homogeneous and heterogeneous clusters.

On the one hand, homogeneous clusters are straightforward to observe in the central region of the stem grouping the snapshots of $meta\_eQ$, $meta\_fQ$, $meta\_lQ$, $meta\_mQ$ and $meta\_pQ$ for $1 \leq Q \leq 5$. Other instances on a smaller scale are the grouping of $meta\_o2$ with $meta\_o3$ (magenta) as well as $meta\_i2$ with $meta\_i4$ (red).

On the other hand, the heterogeneous clusters are divided in two types: bipartite clusters and tripartite clusters. Bipartite clusters are observed at the bottom of the dendogram where $meta\_bQ$ (black) with $meta\_kQ$ (green), $meta\_aQ$ (yellow) with $meta\_hQ$ (magenta), or $meta\_cQ$ (cyan) with $meta\_nQ$ (red) populate the same group. In each of these cases, the USM has captured similarity by pattern matching at the left side of the snapshots. This fact is indeed supported by the way the snapshots were constructed. For instance, the ones in $meta\_bQ$ (black) and $meta\_kQ$ (green) were created using the pair of rules $(167, 102)$ and $(167, 105)$ respectively. Yet another particular instance of this type of relationship is also seen in the cluster comprising $meta\_o4$ (magenta) and the snapshots of $meta\_dQ$ (green). Tripartite clusters appear at the top of the dendogram in which the whole $meta\_jQ$ (cyan) plus $meta\_i1$ (red), $meta\_i3$ (red) and $meta\_o3$ (magenta) are grouped together, and at the top-left in which the five snapshots of $meta\_gQ$ (blue) are clustered with $meta\_i5$ (red) and $meta\_o1$ (magenta). In both cases, the linkage between the prominent group and the other snapshots suggests that the USM has captured common structures such as triangular particles or diagonal strips.
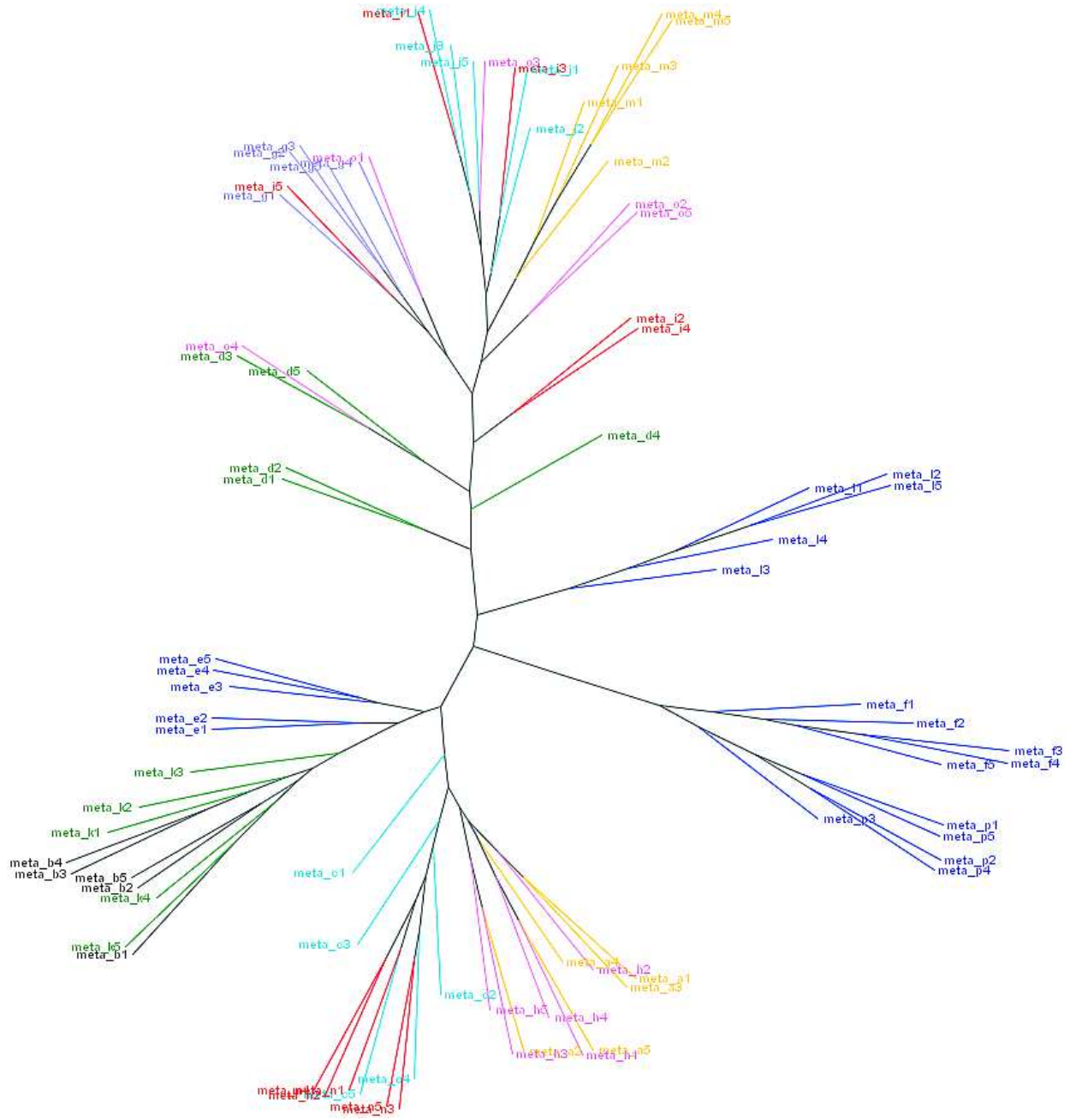
FIGURE 8.17: *Illustration of the logarithmic cluster tree for snapshots belonging to the Meta-automaton model, rules of which were obtained with Algorithm 3. The data partition reveals both homogeneous and heterogeneous clusters.*

In order to perform the second experiment, nine groups of 5 snapshots each were generated from arbitrary chosen pairs of rules. This experiment differs from the latter in the sense that there is no underlying mechanism of creation for establishing a relation among the generated groups of snapshots. A representative snapshot of each group together with its rule is depicted in Figure 8.18.

$meta\_a1/(122, 102)$ $meta\_j1/(167, 129)$ $meta\_q1/(177, 132)$

$meta\_r1/(218, 62)$ $meta\_s1/(111, 42)$ $meta\_t1/(1, 5)$

$meta\_u1/(77, 239)$ $meta\_v1/(145, 26)$ $meta\_w1/(57, 225)$
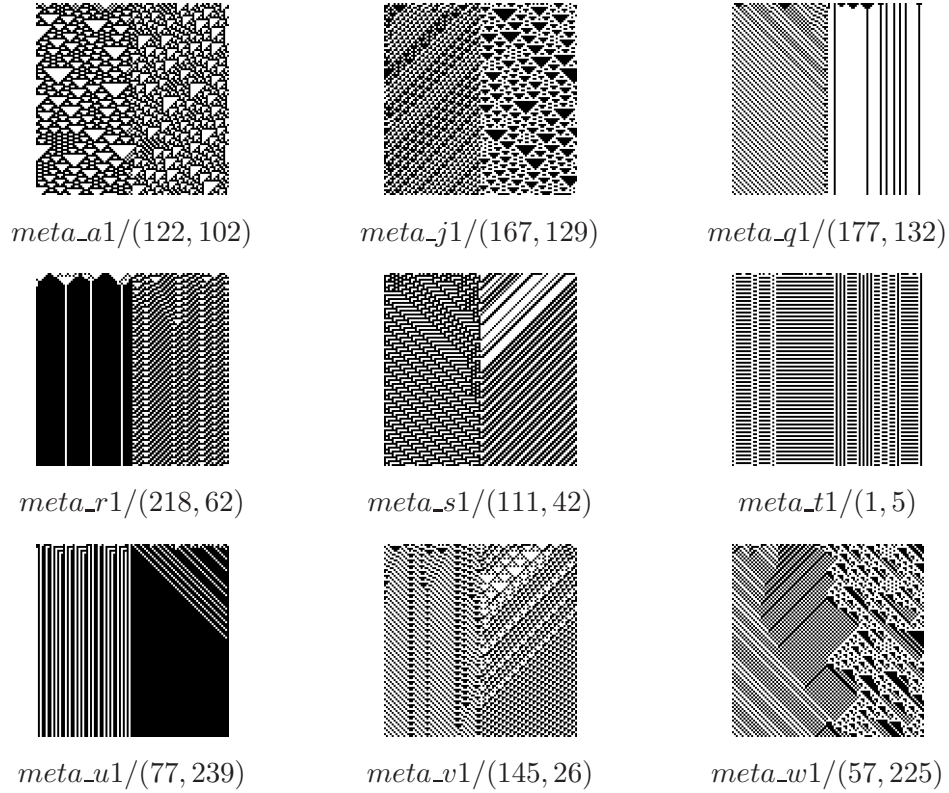
FIGURE 8.18: *Representative snapshots of groups generated with nine arbitrary chosen rules. Labels of the form $xyz\_pQ / (r_a, r_b)$ indicate group and associated creational rules.*

Thus, after creating and processing the inter-snapshots proximity matrix M_s, the resultant data partition is shown in Figure 8.19.
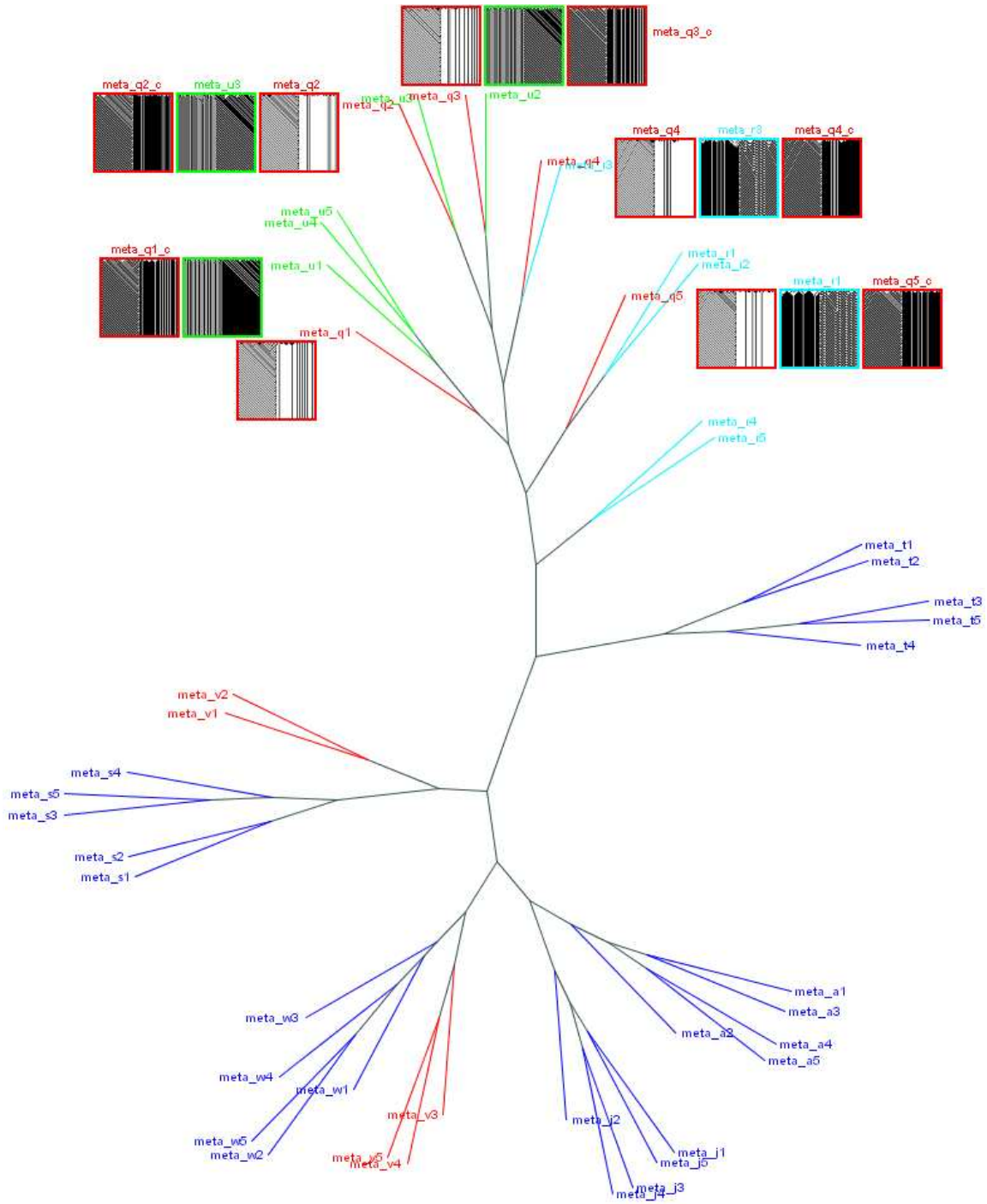
FIGURE 8.19: *Illustration of the logarithmic cluster tree for nine arbitrary groups of snapshots belonging to the Meta-automaton model. The data partitions reveal homogeneous and heterogeneous clusters where some similarities seemed to have been found in terms of complementary mirrored snapshots.*

On the one hand, the achieved dendogram reveals that the USM has detected very well the similarities among the snapshot belonging to $meta\_tQ$, $meta\_aQ$, $meta\_jQ$, $meta\_wQ$ and $meta\_sQ$, resulting from the five homogeneous clusters. On the other hand, due to the differences in number and length of the underlying structures, the snapshots of $meta\_vQ$ were placed in two distinct homogeneous partitions. The first one hosts snapshots $meta\_v1$ and $meta\_v2$ (red) each of which have long, vertical, black strips at the left side and large, triangular, white structures at the right side. The second partition, however, comprises $meta\_v3$, $meta\_v4$ and $meta\_v5$ (red), where the triangular white structures are considerably smaller and the growth of the black strips seems interrupted by diagonal particles "falling" from the right (see Figure 8.20).

$meta\_v1$ $meta\_v2$ $meta\_v3$ $meta\_v4$ $meta\_v5$
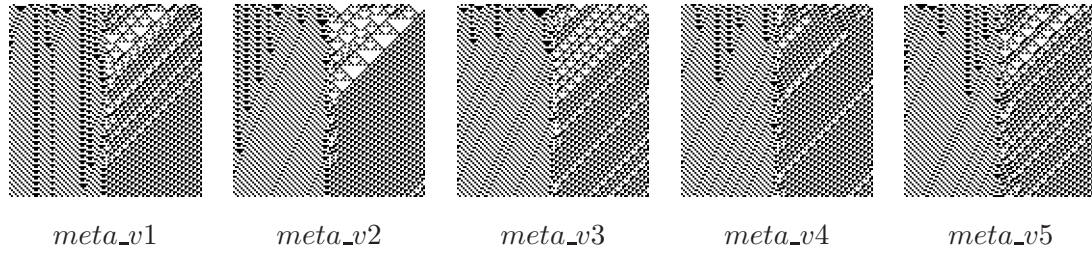
FIGURE 8.20: *A group of snapshots split in two clusters. Snapshots meta_v1 and meta_v2 with long, vertical, black strips (left side) and large, triangular, white structures (right side) were hosted together whereas meta_v3, meta_v4 and meta_v5 were distinguished by their tiny triangular white structures and the black strips broken by diagonal particles "falling" from the right.*

In the same way, snapshots $meta\_r1$, $meta\_r2$, $meta\_r4$ and $meta\_r5$ (cyan) were placed in separate clusters due to their underlying structures. Following an analogous analysis, the long, compact, triangular strips at the right side characterise the first two snapshots. Conversely, the intensified "falling" diagonal particles, which make these strips shorter and less dense, place the last two snapshots in a different partition (see Figure 8.21).

$meta\_r1$ $meta\_r2$ $meta\_r4$ $meta\_r5$
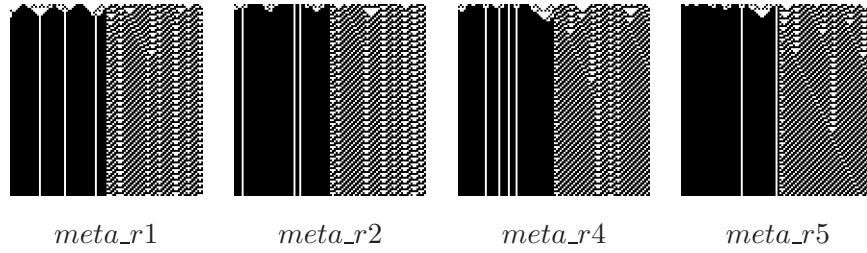
FIGURE 8.21: *Length, size and density of the vertical strips at the right side of the snapshots differentiate meta_r1 and meta_r2 from meta_r4 and meta_r5.*

Nevertheless, snapshots $meta\_u1$, $meta\_u4$ and $meta\_u5$ (green) were also clustered apart from their two other peers. In this case the separation is clearly evident as it is explained by the black trapezoidal area that appears at the bottom right corner of some snapshots, but it is missing in the rest (see Figure 8.22).

$meta\_u1$ $meta\_u4$ $meta\_u5$ $meta\_u2$ $meta\_u3$
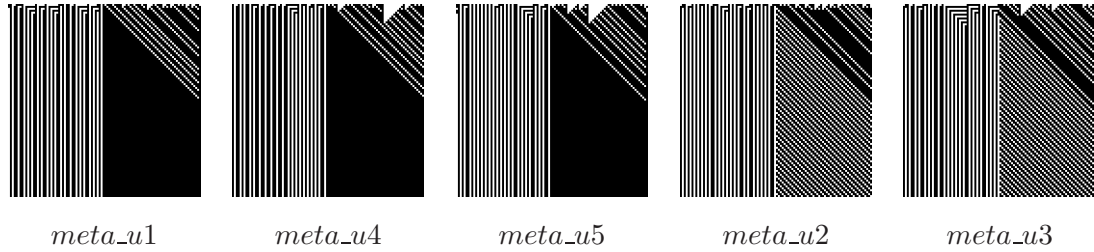
FIGURE 8.22: *A group of snapshots split in two clusters. The separation is clearly evident as it is explained by the black trapezoidal area that appears at the bottom right corner of meta_u1, meta_u4 and meta_u5 and is absent in meta_u2 and meta_u3.*

The remaining clusters in Figure 8.19 consist of snapshots coming from different creational groups. Such is the case of the partitions hosting $meta\_q2$ (red) with $meta\_u3$ (green), $meta\_q3$ (red) with $meta\_u2$ (green), and $meta\_q4$ (red) with $meta\_r3$ (cyan). In each of these pairwise clusters, the USM seemed to have detected similarities in terms of complementary mirrored snapshots, likewise the results seen in Chapter 6. For instance,

complementing and mirroring snapshot $meta\_q4$ results in a snapshot $meta\_q4\_c$ similar to $meta\_r3$ as Figure 8.23 shows. The same line of reasoning may help understand the allocation of $meta\_q1$ and $meta\_q5$ close to $meta\_u1$ and $meta\_r1$ respectively. Figure 8.23 depicts the complementary mirrored snapshots of $meta\_qQ$, for $1 \leq Q \leq 5$, together with their alike counterparts as detected by the USM.

$meta\_q1\_c$ $meta\_u1$

$meta\_q2\_c$ $meta\_u3$ $meta\_q3\_c$ $meta\_u2$

$meta\_q4\_c$ $meta\_r3$ $meta\_q5\_c$ $meta\_r1$
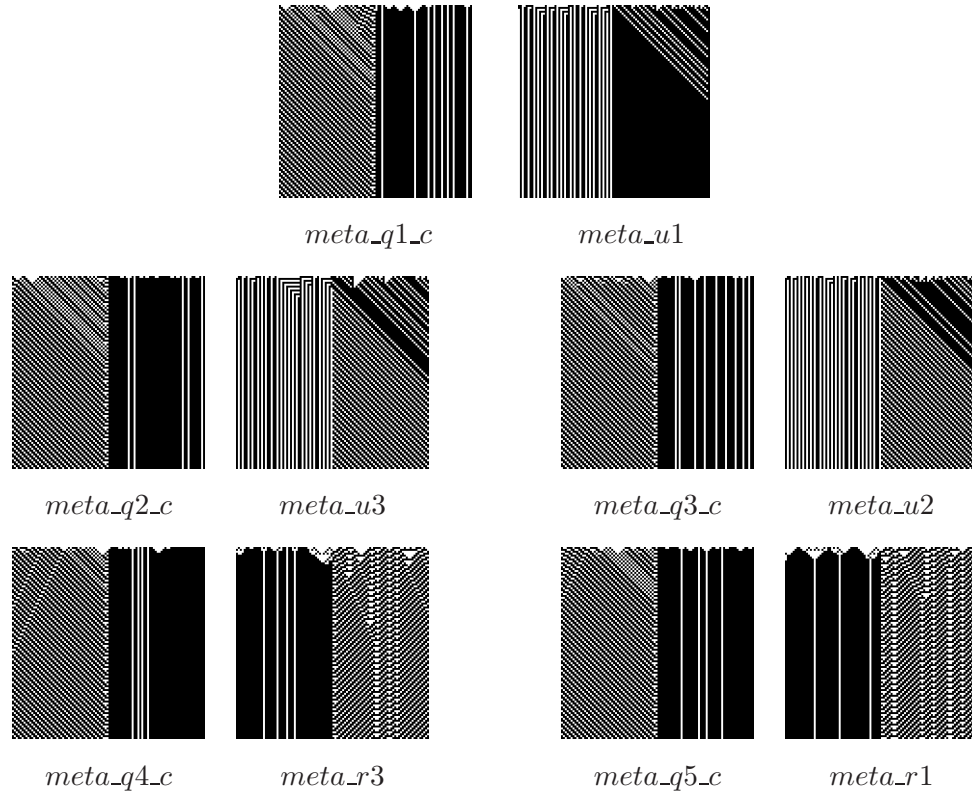
FIGURE 8.23: *Complementary mirrored snapshots of meta_qQ, for $1 \leq Q \leq 5$, together with their alike counterparts as seemed to be detected by the USM.*

Although FDC analyses performed in Section 8.1 say that using USM as fitness function is not always effective in all the experiments performed above the clustering confirms that it is. This is the reason why overall the GAs findings are significant.

8.6 Clustering the Self-Assembly Wang Tiles

As it was seen in Section 8.3, there is also a complex, non-linear, stochastic relationship observed in the mapping from genotype (the tiles) to phenotype and from phenotype (the assembled aggregates) to fitness (numerical value) in the self-assembly Wang tiles evolutionary design optimisation problem.
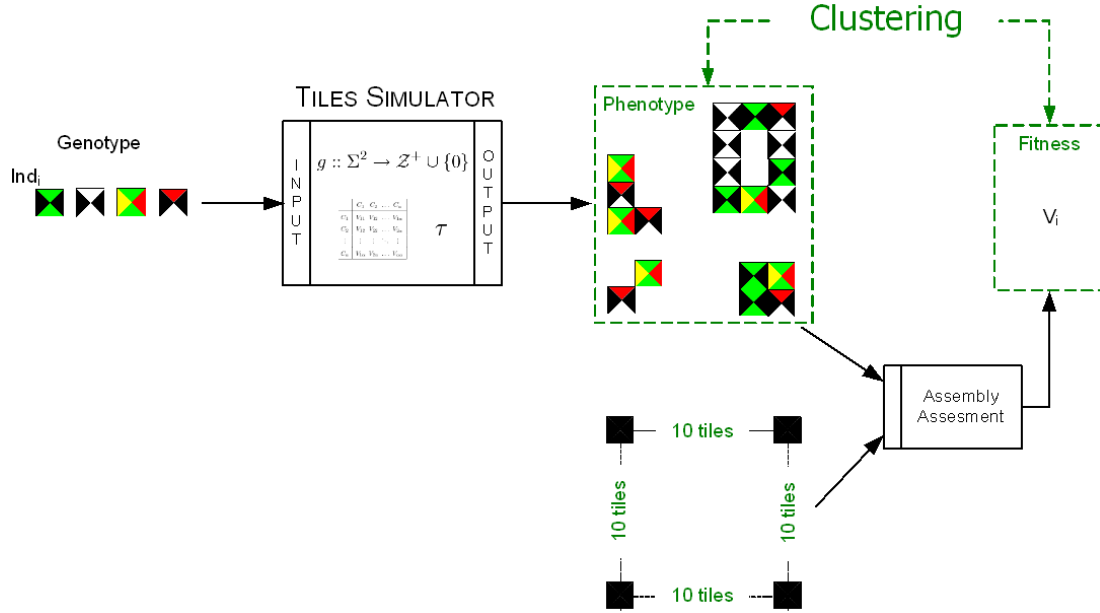
FIGURE 8.24: *Diagram of mappings from genotype onto phenotype and from phenotype onto numerical fitness value, and relationship to clustering analysis.*

As in the previous section, the clustering procedure indicated in Section 2 will be applied at phenotype level with the hope of obtaining a better insight and of verifying the phenotype-fitness relationship (see Figure 8.24). For this to be effective, the cluster analysis is considered over the final configurations obtained by the individuals created with the algorithm of Section 8.3.1 and from where the resulting findings of this methodology will be used as a complementary assessment of the Minkowski functionals as fitness function.

8.6.1   Experiments and Results

In order to undertake the cluster experiments, the whole set of final configurations obtained from the self-assembly Wang tile simulations performed in Section 8.3.1 is employed. As these configurations comprise the actual clustering input data, the feature selection by which the objects are expected to be distinguished is the number of aggregates, their perimeters and the biggest aggregate area. Thereby, the measure of affinity between each pair of configurations $(Conf_i, Conf_j)$ is computed in terms of the evaluation function (see Equation 8.5) and stored in the similarity matrix M_s defined in Equation 8.8. Likewise, for the cluster experiments performed in the previous section, the UPGMA algorithm is employed by which resultant data partitions will be visualized and interpreted over a dendogram representation.

$$M_s[i,j] = \begin{cases} Eval(Conf_i, Conf_j) & \text{if } i \neq j \\ \\ 0 & \text{otherwise} \end{cases}$$

M_s is a proximity matrix s.t. $M_s[i,j] = M_s[j,i]$

$Conf_i, Conf_j$ are final configurations

$$1 \leq i, j \leq 2500 \tag{8.8}$$

Thus, the resultant data partition showing eight clusters labelled as **A**, **B**, **C**, **D**, **E**, **F**, **G** and **H** is depicted in Figure 8.25. By sampling representative configurations from each of these clusters, it is possible to observe that the data has been well partitioned since the distribution and morphology of the aggregations is mostly similar in each of the clusters.
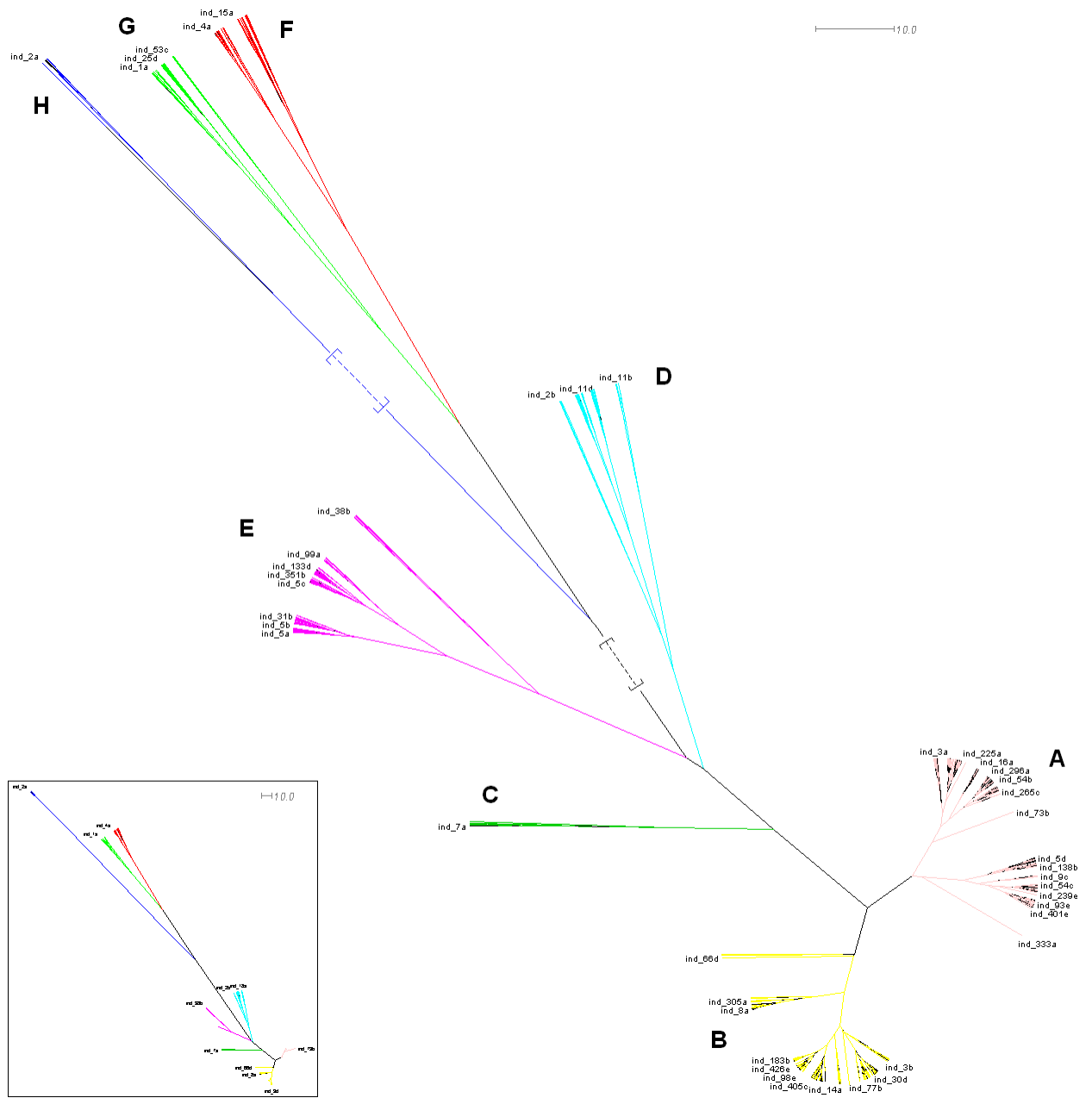
FIGURE 8.25: *Illustration of the logarithmic cluster tree for self-assembly Wang tiles configurations, individuals of which were obtained with Algorithm 1.*

On the one hand, by analysing the partitions located at the top part of the dendogram, the configurations of cluster **H** reveal scattered tiles with no or very few small aggregates like those appearing in the snapshot of Figure 8.26 (a). Conversely, representatives of cluster **G** reveal large-size aggregates with very few unassembled tiles as in the configuration depicted in Figure 8.26 (b). Close to these two types of partitions, the configurations of cluster **F** have achieved assemblies which are either large in size and merged with some small others as shown in Figure 8.26 (c), or medium-size aggregates combined with few others of minor area as shown in the configuration in Figure 8.26 (d).

(a)   (b)

(c)   (d)
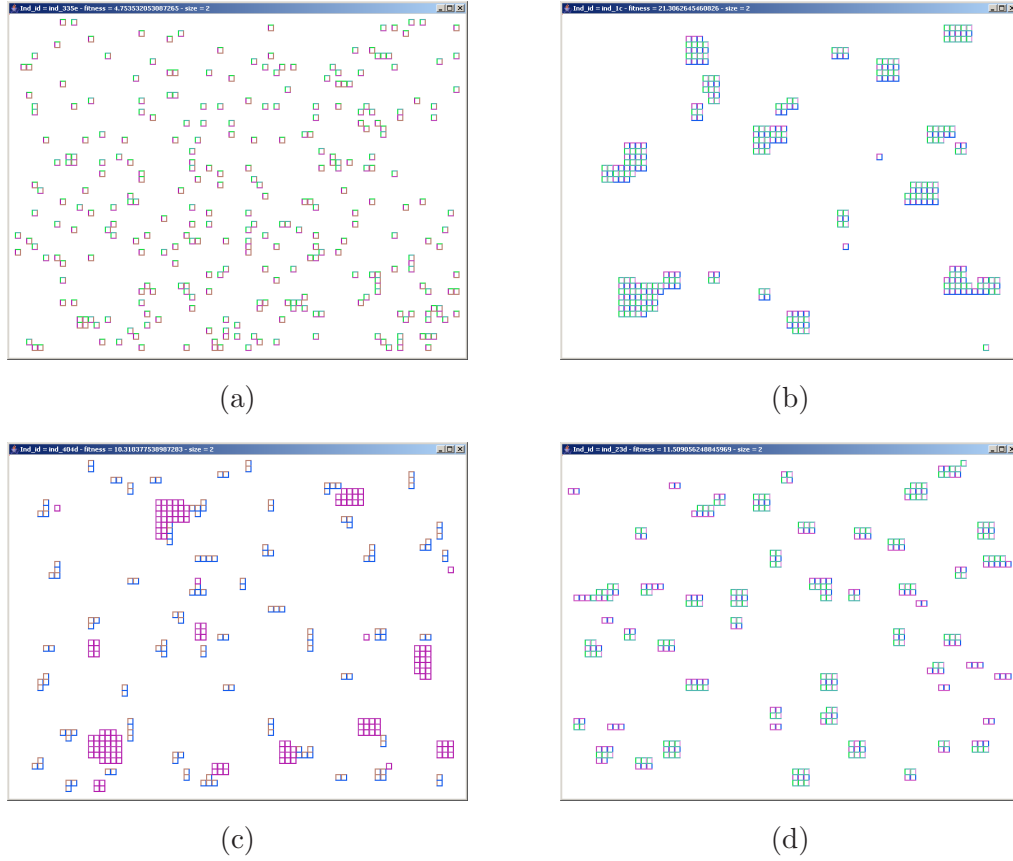
FIGURE 8.26: *Representatives of three clusters: (a) Scattered tiles and small size aggregates characterise partition* **H***; (b) large aggregates feature partition* **G***; (c-d) large and small size aggregates as well as medium and small size aggregates characterise partition* **F***.*

On the other hand, from an analysis of some representatives belonging to the bottom right partitions labelled as **A** and **B**, it is evident that the morphology of the aggregates is contrary to the ones described before. For instance, the configurations observed in **A** comprise either aggregates with dendritic shape along with scattered tiles as shown in Figure 8.27 (a), or small strips also sharing the lattice with few unconnected tiles as depicted in Figure 8.27 (b). Similarly, the aggregations found in cluster **B** also comprise small rectangular aggregates although mixed with T-shaped and L-shaped structures in most of the cases as appears in configurations of Figure 8.27 (c) and Figure 8.27 (d).

(a)

(b)

(c)

(d)
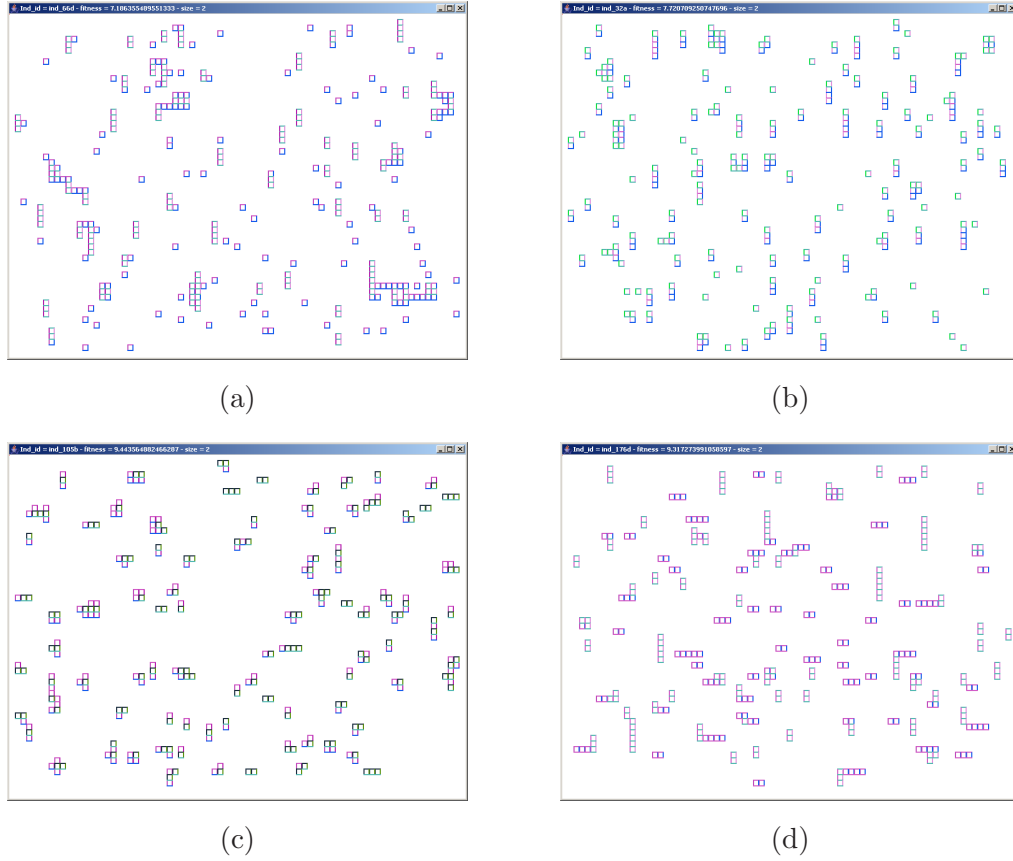
FIGURE 8.27: *Representatives of two clusters: (a-b) dendritic aggregates along with scattered tiles and small strips with few unconnected tiles characterise partition* **A**; *(c-d) the appearance of T-shaped and L-shaped structures characterise partition* **B**.

The three partitions located at the central part of the dendogram seem to represent a transition between the two analyses done above. For instance, the configurations belonging to cluster **D** mostly show medium-size strips together with a vast number of scattered tiles distributed across the lattice as it is shown in Figure 8.28 (a) and Figure 8.28 (b). Configurations with similar morphology and a reduced number of scattered tiles are among those observed in cluster **E** (see Figure 8.28 (c)). Moreover, the same partition also includes some other type of configurations where aggregates are usually large and, in many cases, surrounded by scattered tiles (see Figure 8.28 (d)).

(a)   (b)

(c)   (d)
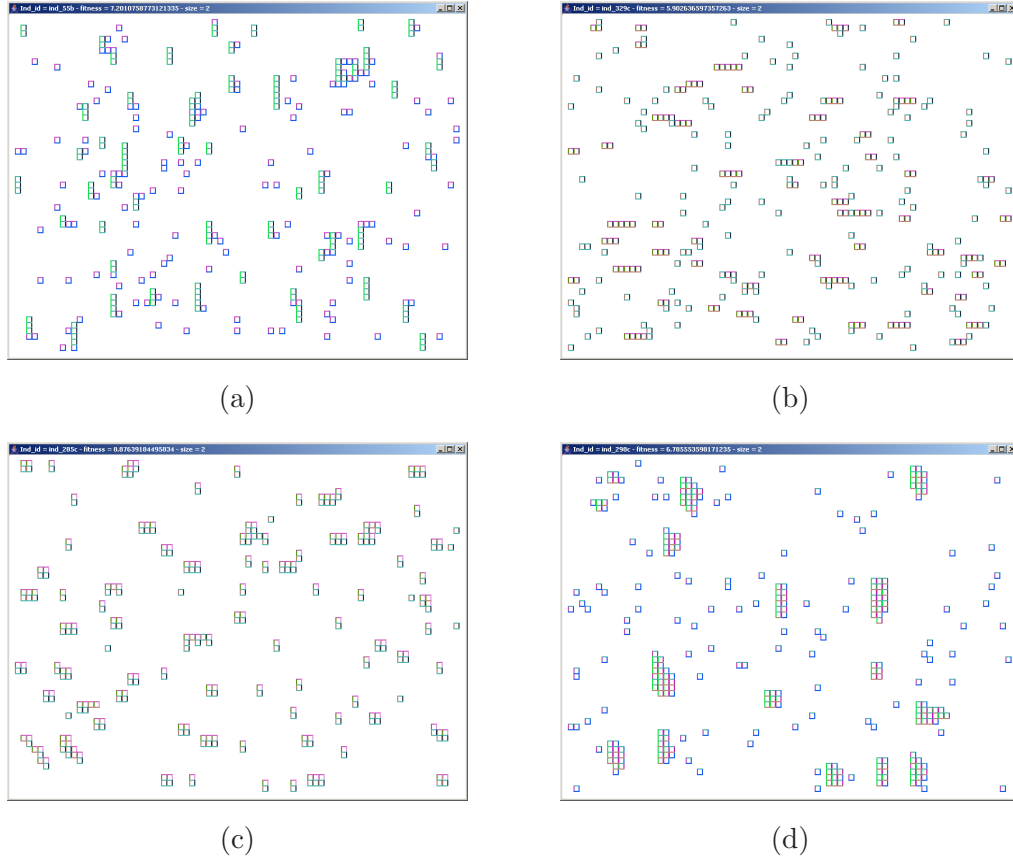
FIGURE 8.28: *Representatives of two clusters: (a-b) medium strips surrounded by a vast number of scattered tiles distributed across the lattice identify partition* **D***; (c-d) a number of scattered tiles approaching to nil and some big aggregates characterise partition* **E***.*

Finally, the occurrence of large- and medium-size aggregates combined with scattered tiles is the common feature that identifies the configurations observed in partition **C** (see Figure 8.29).

<div align="center">(a)   (b)</div>
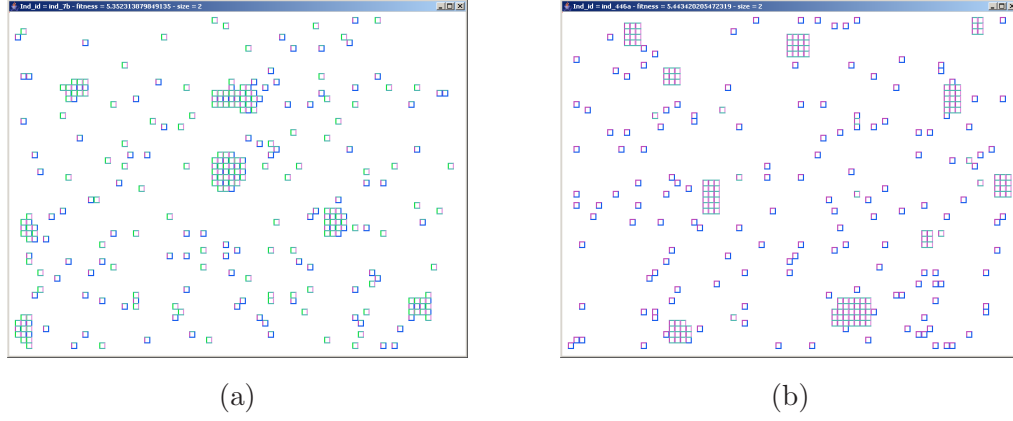
FIGURE 8.29: *Two representatives configurations of partition* **C** *showing large- and medium-size aggregates combined with unassembled tiles distributed across the lattice.*

To summarise, the findings achieved after applying cluster analyses over the 2500 final configurations, i.e. product of simulating 500 individuals, come to support the view that there is in fact an acceptably high correlation between the phenotypes and their fitness values. In other words, these findings verify that Minkowski functionals can effectively differentiate between dissimilar phenotypes and classify similar ones for the purpose of selection.

8.7   Conclusions

This chapter has presented a dual assessment to study the effectiveness of the proposed GAs as methods for the design optimisation of CA parameters and self-assembly Wang tiles shown in Chapter 5, Chapter 6 and Chapter 7 respectively.

Such is the complexity of the genotype-phenotype-fitness mapping, that FDC cannot, alone, be guaranteed to give a completely accurate picture. Indeed, the objective function itself is also only an approximation of two individuals' phenotypic similarity. For these reasons, relying on only FDC or only clustering to validate complex problems as the ones solved here would not be adequate. It is for this reason that both methods were combined with the aim to show whether a given fitness function is a suitable evaluation mechanism for the evolutionary design optimisation problems addressed in Chapter 5, Chapter 6 and Chapter 7.

In the CAs instances presented in Chapter 5 and Chapter 6, the GA findings added some support for the use of compression-based information distance metrics, such as the USM, as fitness functions. However, from the analysis of the FDC and clustering, one could expect (and this was indeed confirmed by the evolutionary runs) that there would be cases where the USM cannot properly inform the evolutionary process. Moreover, an introspective analysis of the cases where the FDC reported poor correlation and where the USM induced bad clustering can shed light on ways on improving the fitness function used. In a similar way, the results obtained with the morphological image analyses in Chapter 7 supports the use of Minkowski functionals as fitness function, although only 5% of the FDC analysis applied to the systematically obtained individuals of length 2 has revealed that the use of GA may successfully solve the problem. On the other hand, the cluster analyses have accurately classified the configurations according to their morphological features, supporting the way in which the fitness function evaluates the self-assembled aggregates.

As a general conclusion, the combination FDC plus cluster analysis presented in Subsection 8.2 and Subsection 8.5 indicates that the use of USM – with both the chosen representation and genotype to phenotype mapping – are amenable for the evolutionary design of complex systems such as CAs. However, the FDC analysis and scatter plots also reveal that some of the target spatio-temporal patterns might be more difficult to evolve than others. So it is expected that the evolutionary algorithm would, in some cases, find it difficult to evolve suitable patterns. Similarly, the application of this dual methodology in Subsection 8.3 and Subsection 8.6 reveals that employing a fitness function in terms of Minkowski functionals for the evolutionary design optimisation of self-assembly Wang tiles results in a complex mechanism of evaluation where, although its success as phenotype evaluator seems to be appropriate, a different type of analysis is needed for an assessment of how effectively an individual correlates to its genotypic distance to a known optimum.

Considering the combination of the results presented in Chapter 5, Chapter 6 and Chapter 7 and those shown in this chapter, it emerges that employing the combination clustering plus FDC is a dual assessment that reveals an accurate indication of the quality of the encoding, i.e. genotype, its mapping to phenotype and USM or Minkowski functionals as fitness functions. Therefore, the application of this methodology before starting long and expensive evolutionary runs should be considered in any problem where the genotype-phenotype-fitness mapping is complex, stochastic, many-to-many and computationally expensive. Thus, this protocol analysis is a contribution of general interest beyond self-assembly and self-organisation optimisation design.

CHAPTER 9

Self-Assembly Dynamics

In this dissertation, a GA approach for the design optimisation of four hierarchical simulation models of self-assembling Wang tiles was presented. After that, a complementary dual assessment combining FDC and clustering in order both to diagnose whether the proposed GA effectively tackles the problem and to analyse how the objective function accurately differentiates phenotypes was introduced. In addition to the significant evolved designs and the indicators collected after studying the complex, non-linear relationship genotype-phenotype-fitness, attention should also be given to the role of the model dynamics and to how the information it provides is processed across evolution. Consequently, this chapter explores the emergent mechanisms of cooperation among tiles and the way in which the strength values encoded in the interaction matrix are employed across the evolutionary process. The first section presents a methodology to explore the underlying cooperative strategies that evolved tiles employ to promote self-assembling, whilst the second part focuses on phenotypic evolutionary activity. This chapter extends a book chapter published in "Systems Self-Assembly: Multidisciplinary Snapshots" [203]

9.1 Emergence of Generalised Secondary Structures

Understanding how autonomous components, equipped with a limited local view of an extensive decentralized environment, find the way of arranging themselves in large aggregates is not a straightforward task. For instance, consider the twelve-steps animation depicted in Figure 9.1 in which a set of self-assembly Wang tiles describes one of the many possible ways to build a square shape when undergoing **Model 2**.
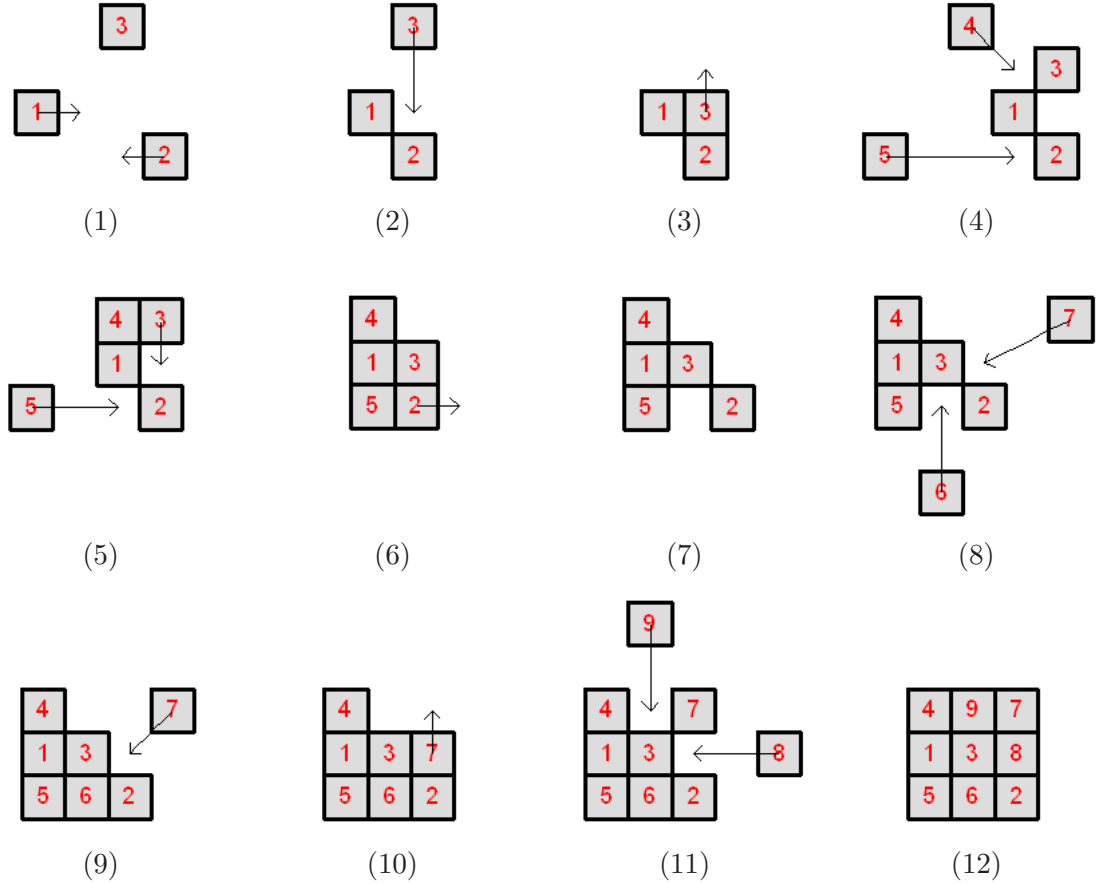
FIGURE 9.1: *Simulation of an individual undergoing* **Model 2**: *(1-3) three tiles self-assembling in a pseudo-corner; (4-5) tile 3 moves up due to probabilistic criteria whilst tile 4 assembles and tile 5 approaches the aggregation; (6-7) tile 5 assembles, tile 3 moves downwards and tile 2 moves to the right; (8-10) tiles 6 and 7 assemble; (11-12) due to the probabilistic critieria, tile 7 moves up as two extra tiles complete the square shape.*

The collective stochastic behaviour among these tiles demands a systematic and careful analysis of their local (edge-to-edge) interactions in order to understand the causes by which large structures emerge. In other words, the interest lays in addressing the following question:

Given an evolved self-assembly Wang tile system, how would a target shape emerge out of tiles' interactions ? i.e. how did the evolutionary process "engineer" self-assembly ?

For this reason, the current section aims to unveil the many co-operative strategies of assembly operating as intermediate steps for the creation of supra-aggregates. In particular, the focus here is centred on the enumeration of all the possible combinations of two, three, four and five tiles in which the resultant strength among the edges to collide is greater than the temperature of the system. These intermediate tile assemblies are referred as *Generalised Secondary Structures (GSSs)* which are deemed to be building blocks for achieving a target shape across evolution. The name is inspired by protein's secondary structures that are thought to be the key milestones for its three-dimensional conformation. GSSs were also referred as self-organised higher order structures which are hyperstructures created under constant external influence [204] [205]. Hyperstructures are attributed to the result of emergence and hierarchy which is a cumulative structure (not necessarily recursive) of primitive agents interacting and forming second order agents with new properties and levels of interaction. It has been observed that it is precisely by this new unfolded levels of interaction that a system is able to produce further and yet more profound complexity.
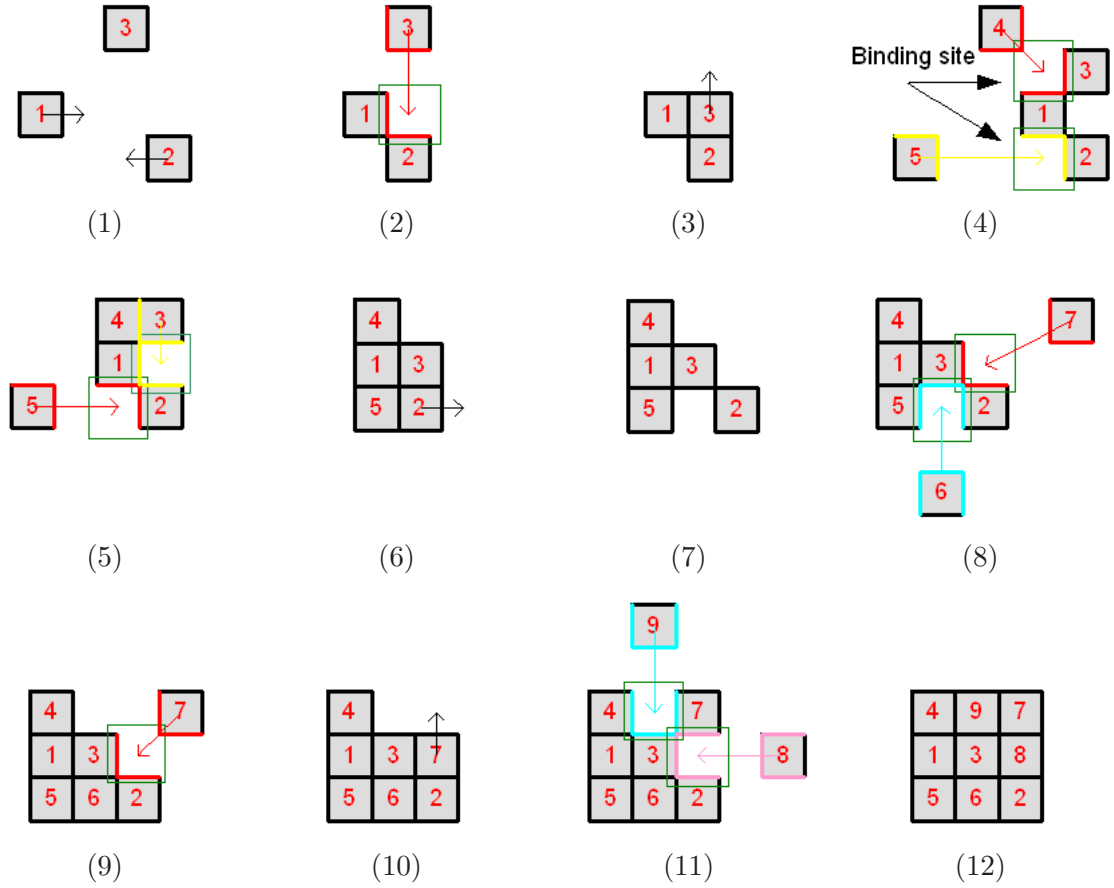
FIGURE 9.2: *GSSs across the simulation of an individual undergoing* **Model 2**.

Figure 9.2 uncovers the participating GSSs of the simulation shown in Figure 9.1. Tiles in which their edges are painted in red or yellow constitute what is called three-tiles self-assembly GSS whilst those painted in cyan or magenta define a four-tiles self-assembly GSS. Later on, the place where the approaching tile is about to collide will be referred as the *binding site* (green box). In order to give a more accurate definition of GSSs, consider the four groups of images depicted in Figure 9.3. The first row characterizes two-tiles self-assembly GSSs. A GSS belonging to this group is an elementary assembly formed by the interaction of only two tiles and it is the most simple and primitive type of GSSs.
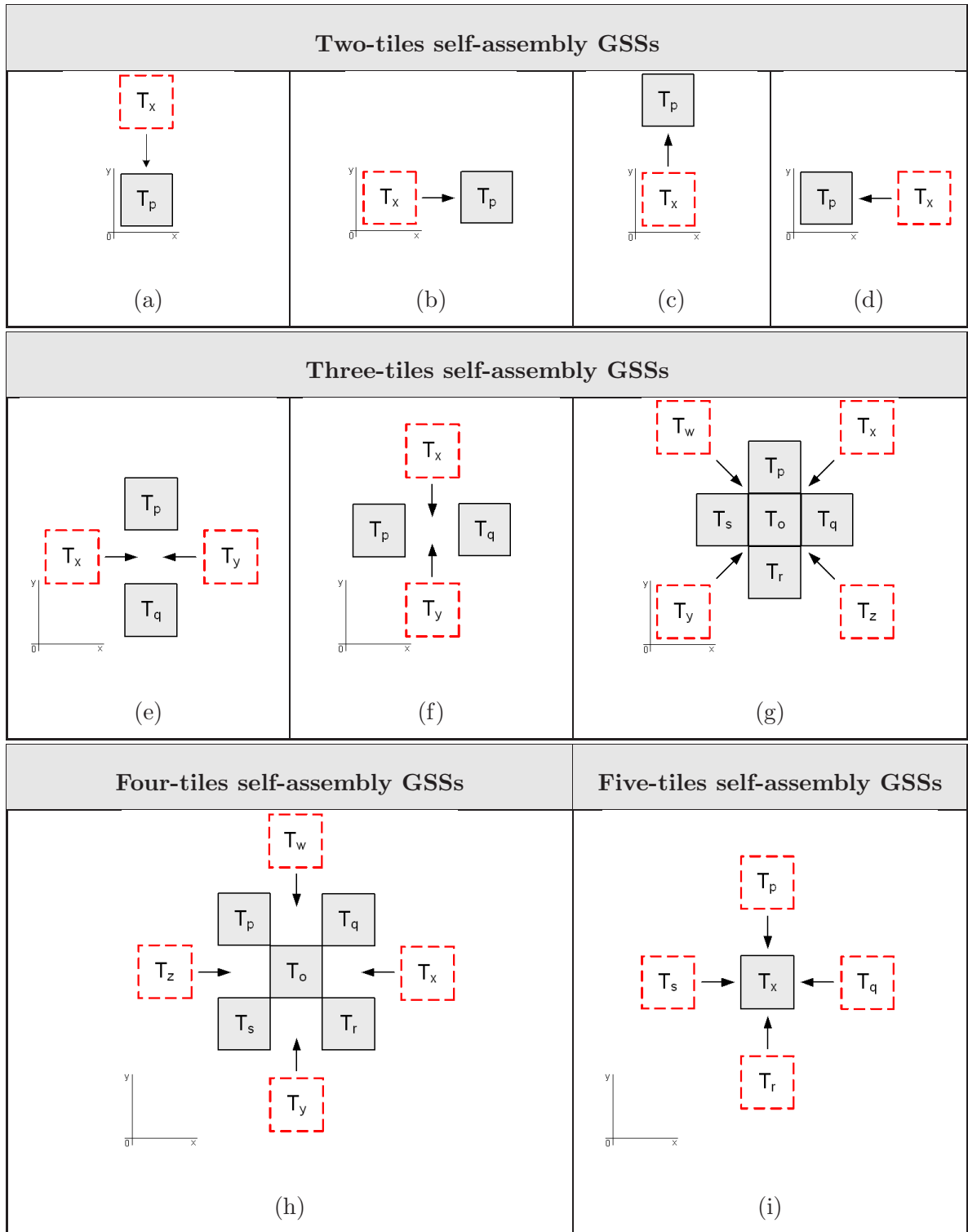
FIGURE 9.3: *Two, three, four and five-tiles self-assembly Generalised Secondary Structures (GSSs) which operate as intermediate steps for the creation of supra-aggregates across the evolutionary process.*

Considering the tiles T_p and T_x, two-tiles self-assembly GSSs comprise the four potential aggregates where T_x attempts to stick on north of T_p (Figure 9.3 (a)), east of T_p (Figure 9.3 (b)), south of T_p (Figure 9.3 (c)) or west of T_p (Figure 9.3 (d)).

The second group characterizes the three-tiles self-assembly GSSs. These are elementary assemblies obtained by the interaction among three tiles. Considering the tiles T_p, T_q, T_x and T_y, these are the six aggregates where T_x (T_y) attempts to assemble T_p and T_q when coming from the west (east) (Figure 9.3 (e)), when T_x (T_y) comes from the north (south) (Figure 9.3 (f)) and when a tile attempts to assemble to any other two forming a pseudo-corner structure. Following Figure 9.3 (g), this last group is formed by aggregates where T_x attempts to assemble T_p and T_q when coming from north-east; T_z to T_q and T_r when coming from the south-east; T_y to T_s and T_r when coming from south-west; and T_w to T_s and T_p when coming from north-west.

Following the similar line of reasoning, four-tiles self-assembly GSSs and five-tiles self-assembly GSSs are defined as the elementary assemblies formed by the interaction of four and five tiles respectively. Considering the tiles T_o, T_p, T_q, T_r, T_s, T_x, T_y, T_w and T_z, the former group comprises the four aggregates where T_x attempts to assemble T_q, T_o and T_r when coming from the east; T_y to T_r, T_o and T_s when coming from the south; T_z to T_p, T_o, and T_s when coming from the west; and T_w to T_p, T_o and T_q when coming from the north (Figure 9.3 (h)).

In the latter, five-tiles self-assembly GSS is the aggregate obtained when tiles T_p, T_q, T_r and T_s attempts to assemble T_x from north, east, south and west respectively (Figure 9.3 (i)).

In order to enumerate the two, three, four and five-tiles self-assembly GSSs, four independent algorithms which receive an individual (Ind_i), a matrix encoding the glue types interaction (M) and returns the GSSs were implemented. The pseudocode to enumerate the two-tiles self-assembly GSSs is shown in Algorithm 4 where N, S, E and W are functions that return the glue type associated to the north, south, east or west edge of a given tile respectively and S_H, S_V are sets of pair of tiles that self-assemble in elementary structures of two tiles as shown in Figure 9.3 (b-d) and Figure 9.3 (a-c) respectively, i.e. in horizontal or vertical fashion.

Algorithm 4 Enumerate_Two-Tiles_GSSs

Input: Ind_i an individual, M interaction matrix

Output: S_H, S_V sets of two-tiles self-assembly GSSs

1. **for all** tiles T_p, T_x in Ind_i **do**

2.   **if** $M[N(T_p), S(T_x)] > \tau$ or $M[S(T_p), N(T_x)] > \tau$ **then**

3. $Insert(S_V, (T_p, T_x))$

4.   **end if**

5.   **if** $M[W(T_p), E(T_x)] > \tau$ or $M[E(T_p), W(T_x)] > \tau$ **then**

6. $Insert(S_H, (T_p, T_x))$

7.   **end if**

8. **end for**

The pseudocode for enumerating the three-tiles self-assembly GSSs is shown in Algorithm 5 where S_H, S_V, S_1, S_2, S_3, S_4 are sets comprising tuples of three tiles self-assembling as shown in Figure 9.3 (e), Figure 9.3 (f) and as in each of the four pseudo-corners depicted in Figure 9.3 (g) respectively.

Algorithm 5 Enumerate_Three-Tiles_GSSs

Input: Ind_i an individual, M interaction matrix

Output: $S_H, S_V, S_1, S_2, S_3, S_4$ sets of three-tiles self-assembly GSSs

1. **for all** tiles T_p, T_q, T_x in Ind_i **do**

2.   **if** $M[S(T_p), N(T_x)] + M[N(T_q), S(T_x)] > \tau$ **then**

3. $Insert(S_V, (T_p, T_q, T_x))$

4.   **end if**

5.   **if** $M[E(T_p), W(T_x)] + M[W(T_q), E(T_x)] > \tau$ **then**

6. $Insert(S_H, (T_p, T_q, T_x))$

7.   **end if**

8.   **if** $M[N(T_p), S(T_x)] + M[W(T_q), E(T_x)] > \tau$ **then**

9. $Insert(S_1, (T_p, T_q, T_x))$

10.   **end if**

11.   **if** $M[E(T_p), W(T_x)] + M[N(T_q), S(T_x)] > \tau$ **then**

12. $Insert(S_2, (T_p, T_q, T_x))$

13.   **end if**

14.   **if** $M[S(T_p), N(T_x)] + M[E(T_q), W(T_x)] > \tau$ **then**

15. $Insert(S_3, (T_p, T_q, T_x))$

16.   **end if**

17.   **if** $M[W(T_p), E(T_x)] + M[S(T_q), N(T_x)] > \tau$ **then**

18. $Insert(S_4, (T_p, T_q, T_x))$

19.   **end if**

20. **end for**

In a similar way, Algorithm 6 draws the pseudocode for enumerating the four-tiles self-assembly GSSs. In this case, S_1, S_2, S_3, S_4 comprises tuples of four tiles capable of self-assembling as in each of the four alternatives shown in Figure 9.3 (h).

Algorithm 6 Enumerate_Four-Tiles_GSSs

Input: Ind_i an individual, M interaction matrix

Output: S_1, S_2, S_3, S_4 sets of four-tiles self-assembly GSSs

1. **for all** tiles T_p, T_q, T_r, T_x in Ind_i **do**

2.   **if** $M[E(T_p), N(T_x)] + M[N(T_q), S(T_x)] + M[W(T_r), E(T_x)] > \tau$ **then**

3. $Insert(S_1, (T_p, T_q, T_r, T_x))$

4.   **end if**

5.   **if** $M[S(T_p), N(T_x)] + M[E(T_q), W(T_x)] + M[N(T_r), S(T_x)] > \tau$ **then**

6. $Insert(S_2, (T_p, T_q, T_r, T_x))$

7.   **end if**

8.   **if** $M[W(T_p), E(T_x)] + M[S(T_q), N(T_x)] + M[E(T_r), W(T_x)] > \tau$ **then**

9. $Insert(S_3, (T_p, T_q, T_x))$

10.   **end if**

11.   **if** $M[S(T_p), N(T_x)] + M[W(T_q), E(T_x)] + M[N(T_r), S(T_x)] > \tau$ **then**

12. $Insert(S_4, (T_p, T_q, T_x))$

13.   **end if**

14. **end for**

Finally, the pseudocode depicted in Algorithm 7 presents the mechanism to enumerates five-tiles self-assembly where S is a set comprising tuples of five tiles capable of

self-assembling GSSs as depicted in Figure 9.3 (i).

Algorithm 7 Enumerate_Five-Tiles_GSSs

Input: Ind_i an individual, M interaction matrix

Output: S set of five-tiles self-assembly GSSs

1. **for all** tiles T_p, T_q, T_r, T_s, T_x in Ind_i **do**

2.   **if** $M[S(T_p), N(T_x)] + M[W(T_q), E(T_x)] + M[N(T_r), S(T_x)] + M[E(T_r), W(T_x)] > \tau$ **then**

3. $Insert(S, (T_p, T_q, T_r, T_x))$

4.   **end if**

5. **end for**

9.1.1 Experiments and Results

With the four algorithms at hand, the following analysis aims to inspect which type and how many GSSs were discovered by the evolutionary process. For this purpose the best individuals achieved by the GA when employing the Minkowski functionals as fitness function (see Chapter 7) was employed. That is, for every combination of two, three, four and five tiles of an individual, the algorithms will explore which possible arrangements in which the resultant strength among the edges to collide are greater than the temperature, that is which intermediate GSSs are stable thus potentially able to "seed" the target structure.

Two-tiles self-assembly GSSs were found in some of the experiments using **Model 1** and **Model 3**. For instance, the best individuals found by experiments M1A and M1B have only one tile capable of sticking to any other tile, thus increasing the chances to form small self-assembly aggregations at the beginning of the simulation. A similar fact is

observed by the individuals found in M3A, M3B and M3C. These analyses are shown in Table 9.1, Table 9.2, Table 9.3 and Table 9.4 where each element of the column **Tiles** is the entry point in the table. The combination of N, S, E and W labels the edges by which the tile binds to another, e.g. NS means instances where the two tile types could assemble by their north and south edges.

Tiles	4 3 □ 4 8	0 9 □ 8 5	0 9 □ 9 9	0 9 □ 8 6
4 3 □ 4 8	EW	NSEW	NSEW	NSEW
0 9 □ 8 5	NSEW	-	-	-
0 9 □ 9 9	NSEW	-	-	-
0 9 □ 8 6	NSEW	-	-	-

TABLE 9.1: *Two-tile self-assembly GSSs analysis on experiment M1A. Tile 4483 sticks to any other increasing the possibilities of obtaining early small self-assembled aggregates.*

Tiles	5 9 □ 9 0	2 9 □ 4 4	5 9 □ 8 0
5 9 □ 9 0	-	NSW	-
2 9 □ 4 4	NSE	NSEW	NSE
5 9 □ 8 0	-	NSW	-

TABLE 9.2: *Two-tile self-assembly GSSs analysis on experiment M1B. Tile 2449 sticks to any other boosting the possibilities of forming early small self-assembled aggregates.*

Tiles	9 [9 9] 9	9 [8 0] 8	8 [8 9] 0	8 [0 8] 9	0 [9 8] 8
9 [9 9] 9	–	–	–	–	–
9 [8 0] 8	–	EW	ES	ESW	SW
8 [8 9] 0	–	NW	SN	SW	SWN
8 [0 8] 9	–	EWN	EN	WE	WN
0 [9 8] 8	–	EN	ESN	ES	SN

TABLE 9.3: *Two-tile self-assembly GSSs analysis for the best individual obtained with experiments M3A and M3B. Tile* 9088 *together with its three alternative rotations is the only one capable of self-assembling to any other.*

Tiles	9 [9 9] 9	6 [6 5] 7	6 [7 6] 5	7 [5 6] 6	5 [6 7] 6
9 [9 9] 9	–	NESW	NESW	NESW	NESW
6 [6 5] 7	NESW	NS	S	E	W
6 [7 6] 5	NESW	N	WE	W	S
7 [5 6] 6	NESW	W	E	NS	N
5 [6 7] 6	NESW	E	N	S	WE

TABLE 9.4: *Two-tile self-assembly GSSs analysis for the best individual obtained with experiment M3C. Tile* 6576, *together with its three alternative rotations, is the only one capable of self-assembling to any other.*

Interestingly, in the experiments using **Model 2** and **Model 4**, there is no possibility of obtaining two-tiles self-assembly GSSs. That is, there are no combinations of two tiles such that the glue types at their sticking edges obtain a strength greater than the temperature in the system. Hence, three-tiles co-operation is an emergent feature of the system where more than two tiles are required in order to initiate self-assembly. This kind of cooperation was recognised by Winfree and Rothemund in [73] as *necessary* for programmable self-assembly and it is a remarkable result that the evolutionary design of self-assembly Wang tiles achieved precisely that – namely, high cooperativity.

Therefore, for those experiments where two-tiles self-assembly GSSs are not observed, the possible combinations of three tiles are considered; i.e. where one tile attempts to stick to any other two as shown in Figure 9.3 (e), Figure 9.3 (f) and Figure 9.3 (g). In order to make this characterization more tractable, a further analysis counting the number of occurrences for each type of GSSs is performed, i.e. according to the direction from where the red tiles approach in Figure 9.3 (e-g). The outcome of the analysis is summarised in Table 9.5. On the one hand, a few three-tiles self-assembly GSSs are employed in those experiments using **Model 1**. For instance, experiment M1A has developed only 1 occurrence for a particular type of GSS whereas experiment M1C has achieved up to 1 occurrence for each of the six possible types of three-tiles self-assembly GSSs. On the other hand, in **Model 2** and **Model 4**, it emerges that a total of 97 occurrences of this type of GSSs were developed by M2A, 96 by M2B, 15 by M2C and 6 by experiments M4A, M4B and M4C. Experiments performed with **Model 3** do not report any occurrence.

	M1A	M1C	M2A	M2B	M2C	M4ABC
	1	1	15	27	1	1
	–	1	16	9	2	1
	–	1	16	15	3	1
	–	1	17	15	3	1
	–	1	14	15	3	1
	–	1	19	15	3	1
Total	1	4	97	96	15	6

TABLE 9.5: *Amount of three-tiles self-assembly GSSs occurrences found for the best individual obtained with experiments M1A, M1C, M2A, M2B, M2C and M4ABC.*

The last analyses concerning GSSs attempt to reveal if four-tiles self-assembly GSSs (Figure 9.3 (h)) and five-tiles self-assembly GSSs (Figure 9.3 (i)) also participate in the formation of supra-aggregates. Certainly, these have been found only in some individuals of **Model 1** and **Model 2**. The figures for experiments M1C and M2A are listed in Table 9.6. In the former, only 2 occurrences of four-tiles self-assembly GSSs and 1 of five-tiles self-assembly GSSs were found. In contrast, M2A registers a total of 26 occurrences of four-tiles self-assembly GSSs and none involving co-operation among five tiles.

It is interesting to note that among the results obtained when analysing the three-tiles self-assembly GSSs (Table 9.5), some of the elementary assemblies are more likely to occur than others. For instance, the figures under M2B reveal that some GSSs will participate with a total of 27 occurrences while others participate with just 9 or 15. In fact,
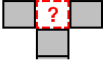
					Total		
M1C	–	1	1	–	2	**M1C**	1
M2A	7	5	9	5	26	**M2A**	–

TABLE 9.6: *Amount of four-tiles and five-tiles self-assembly GSSs occurrences found for the best individual obtained with experiments M1C and M2A.*

these numbers indicate the percentage of participation for a given type of GSS, but they do not really reflect how much strength they contribute. In other words, this fact brings into discussion the following question:

Given a set of GSSs, is there any way to identify which are the most likely to participate in the self-assembly process ?

In order to answer this question, the total strength by which any three tiles are expected to contribute to building a three-tiles self-assembly GSS, is calculated. Inspired by the quantification of free energy contributions taking place in proteins [73] [206], this value receives the name of *Normalised Average Free Energy (NAFE)*. Thus, Equation 9.1, Equation 9.2, Equation 9.3 and Equation 9.4 calculate the NAFE for those binding sites where the T_x tile would bind when coming from north-east, south-east, north-west and south-west respectively (Figure 9.3 (g)) whilst Equation 9.5 and Equation 9.6 compute the NAFE for those where the T_x tile would bind when coming from east (west) (Figure 9.3 (e)) and north (south) (Figure 9.3 (f)). In these equations, G is the glue function where its arguments are

tile edges, e.g. the north edge of the x-tile is T_x^n, and $|Ind_i|$ is the length of the individual.

$$NAFE_{ne}(T_x, T_p, T_q) = \frac{\sum_{x=1}^{|Ind_i|}(G(T_p^e, T_x^w) + G(T_q^n, T_x^s))}{|Ind_i|} \tag{9.1}$$

$$NAFE_{se}(T_x, T_r, T_q) = \frac{\sum_{x=1}^{|Ind_i|}(G(T_r^e, T_x^w) + G(T_q^s, T_x^n))}{|Ind_i|} \tag{9.2}$$

$$NAFE_{nw}(T_x, T_p, T_s) = \frac{\sum_{x=1}^{|Ind_i|}(G(T_p^w, T_x^e) + G(T_s^n, T_x^s))}{|Ind_i|} \tag{9.3}$$

$$NAFE_{sw}(T_x, T_r, T_s) = \frac{\sum_{x=1}^{|Ind_i|}(G(T_r^w, T_x^e) + G(T_s^s, T_x^n))}{|Ind_i|} \tag{9.4}$$

$$NAFE_{e/w}(T_x, T_p, T_q) = \frac{\sum_{x=1}^{|Ind_i|}(G(T_p^s, T_x^n) + G(T_q^n, T_x^s))}{|Ind_i|} \tag{9.5}$$

$$NAFE_{n/s}(T_x, T_p, T_q) = \frac{\sum_{x=1}^{|Ind_i|}(G(T_p^e, T_x^w) + G(T_q^w, T_x^e))}{|Ind_i|} \tag{9.6}$$

The equations given above contribute with an analytical step to identify which of the three-tiles self-assembly GSSs are on average more likely to participate in a self-assembly process. This process is applied to each of the individuals resulting from the experiments summarised in Table 9.5. The obtained results are further partitioned in equivalence classes in which it is said that two GSSs are deemed equivalent if their NAFEs have equal value. The resulting partitions for experiments M1A, M1C, M2A, M2B, M2C, M4ABC – short for M4A, M4B and M4C – are shown in Table 9.7 and Table 9.8. These results reveal that the total amount of GSSs provided by equivalence classes with NAFE lesser than or equal to τ is greater than the quantity provided by equivalence classes with higher NAFE. In other words, GSSs with NAFE smaller than τ are highly likely to participate in the self-assembly process.

Individual 4483-0859-0869-0999-0969 (M1A)

Eq. Class	NAFE	Quantity	Eq. Class	NAFE	Quantity
i	7.0	2	xvii	7.4	2
ii	8.6	4	xviii	3.6	8
iii	3.2	6	xix	15.2	1
iv	2.6	10	xx	12.8	1
v	6.8	2	xxi	13.2	1
vi	8.0	8	xxii	3.8	2
vii	3.4	24	xxiii	8.4	2
viii	7.8	4	xxiv	10.0	4
ix	7.6	6	xxv	7.2	1
x	4.4	6	xxvi	3.0	12
xi	8.8	2	xxvii	9.8	4
xii	9.2	2	xxviii	12.0	1
xiii	2.2	4	xxix	14.4	1
xiv	9.6	2	xxx	2.8	20
xv	8.2	2	xxxi	10.8	1
xvi	14.0	1	xxxii	4.2	4

Individual 6622-9900 (M1C)

Eq. Class	NAFE	Quantity	Eq. Class	NAFE	Quantity
i	2.0	6	iv	2.5	8
ii	1.5	8	v	1.0	1
iii	3.0	1			

TABLE 9.7: *Equivalent classes, normalised average free energy values (NAFE) and amount of binding site configurations for the best individuals found by experiments M1A and M1C.*

Individual Individual 0217-0771-7047 (M2A)

Eq. Class	NAFE	Quantity	Eq. Class	NAFE	Quantity
i	3.0	3	iv	4.66	6
ii	3.33	12	v	4.33	3
iii	3.66	18	vi	4.0	12

Individual 7517-7410-7557 (M2B)

Eq. Class	NAFE	Quantity	Eq. Class	NAFE	Quantity
i	2.0	1	v	3.0	8
ii	5.0	4	vi	4.33	18
iii	5.33	3	vii	4.0	14
iv	3.33	6			

Individual 3550-6467 (M2C)

Eq. Class	NAFE	Quantity	Eq. Class	NAFE	Quantity
i	2.0	5	iv	3.0	8
ii	3.5	4	v	2.5	1
iii	5.0	1	vi	4.5	5

Individual 7777 (M4ABC)

Eq. Class	NAFE	Quantity
i	6.0	6

TABLE 9.8: *Equivalent classes, normalised average free energy values (NAFE) and amount of binding site configurations for the best individuals found by experiments M2A, M2B, M2C and M4ABC - short for M4A, M4B and M4C.*

The analyses shown in this section have contributed with a procedure to quantify and identify key emergent cooperative for understanding the mechanisms employed by the evolved tiles for self-assembling into supra-aggregates. The findings have revealed that generalized secondary structures naturally emerge as a product of artificial evolution acting as milestones across the design optimisation process. From these observations, the selection of the appropriate glue strengths may also play a crucial role for the discovery of those tiles capable of self-assembling into the target shape. This is the reason why the following section expands in a methodology that aims to investigate the activity of the glue strengths in terms of adaptation across generations.

9.2 Evolutionary Activity

In the previous section, the GSSs were introduced as elementary assemblies that characterize the way in which large structures emerge out of local interactions among tiles. Although the identification of these GSSs clearly contributes to a systematic understanding of how cooperativity among tiles takes place, nothing has yet been said about how evolution selects, across generations, the adequate type and number of glue strengths to perform a successful self-assembly of large aggregates. It is therefore a crucial part of the analysis to answer the following question:

Given a GA for the design optimisation of self-assembly Wang tiles, is there any way to measure how the glue strengths participate in the evolutionary process ?

In order to envisage this adaptive phenomena in evolving systems, a general framework for defining and measuring evolutionary activities is defined in [207]. An *evolutionary activity* is a statistic that measures the extent to which components of an evolving system have been resisting selection pressure over its entire history. The general idea is based on the identification of the component to analyse, a way to define its activity and the initial activity values. Once the component is chosen, the focus must be on which instances of these are present in the system at a given time including births and deaths, since the extant components of most of the evolving systems change over time. In most of the cases, the activity of a component is stored in its activity counter. Hence, if new components are added to the evolving system, their activity counters must be initialized either by assigning the same initial value at all times or by subject to the context. Usually, the activity counter is incremented at each moment that the component exists, e.g. when there is exposure to selection, and in terms of its concentration in the population.

In order to visualize an evolutionary activity, the *activity distribution function* and *activity wave* were defined in [208]. The first concept combines information about how the activity of each component changes over time whilst the second is the associated visualization method comprising salient lines called waves. These graphs depict how the evolutionary activity of the components (on the y-axes) varies as function of time (on the x-axis) by the presence of waves. In particular, each wave corresponds to a single component of interest mapping its evolutionary activity over time. For instance, when the birth of a component takes place, a new wave arises from the x-axis increasing its slope as the observed concentration of its associated component grows. When the component goes extinct, its

associated wave falls to zero and ends, reflecting the changes in concentration. Since many components co-exist in the evolving system, it is also possible that multiple waves co-exist in the activity wave diagram, uncovering interactions that may affect the concentrations of the components. Thus dominating components would appear as dominating waves during a given epoch of evolution.

The application of evolutionary activity statistics can be seen in systems undergoing cultural evolution such as is reported in [209] where the aim is to create an empirical picture of the adaptive dynamics in the evolution of patented inventions. More studies of activity distribution functions for genotypes were performed in Holland's Echo model [94], Packard's Bugs model [210], taxonomic families in the fossil record [211], evolution of assembly language programs [208], the Santa Fe artificial stock market [212], Lindgren's model of evolving strategies in the iterated prisoner's dilemma [213], Ray's Tierra model [214] and multimeme algorithms [215], to name but a few. In the following experiments, the activity of interest is considered at the phenotypic level given that it is a more convenient method when genes are highly epistatic. For this reason, the focus of the analysis will be upon the cooperation mechanisms operating among the self-assembly Wang tiles of an individual.

9.2.1   Experiments and Results

Considering the findings achieved by **Model 4** when applying the Minkowski functionals as fitness function (see Chapter 7), the experiments in this section aim to visualize the evolutionary activity over the glue strengths (0...9) employed by the GSSs. Thus the evolutionary activity $a_i(t)$ maps the cumulative usage for a glue of strength i, as instantiated

by some GSS, up to generation t. This is specified in Equation 9.7 where $C_i(t)$ refers to the concentration in the population of glues with strength i at generation t. A cut-off of 0.1 was used for the concentration so as to capture only significant glues, hence when it falls below 0.1 the counting is restarted.

$$a_i(t) = \begin{cases} \sum_{j=k}^{t} C_i(j) & \text{if } C_i(j) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{9.7}$$
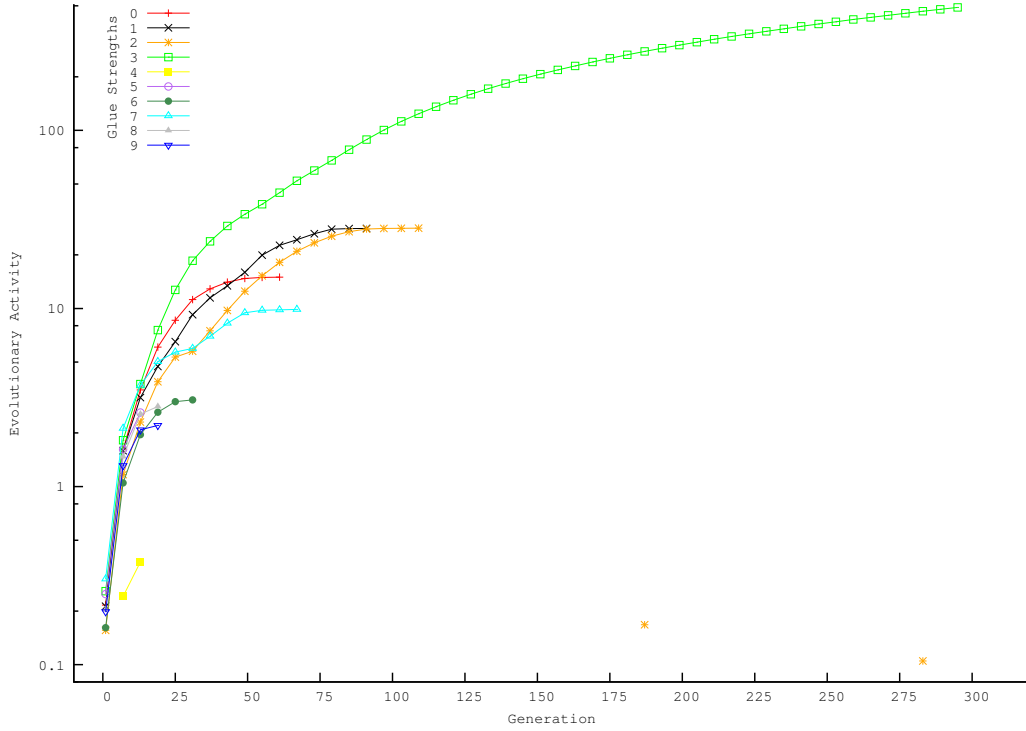
FIGURE 9.4: *Activity waves of the 10 different glue strengths of the system when undergoing* **Model 4**. *Crests depict the degree of participation of the glue strengths along the evolutionary process as two tiles attempt to assemble.*

The chart in Figure 9.4 shows the resulting activity waves of the possible glue strengths for a representative run of the evolutionary algorithm under **Model 4**. The

analysis is applied to all the possible two-tiles assemblies. Since this model is invariant under rotation, glue strengths arising from interactions in the X or Y axis are equivalent. Thus, the evolutionary activity plots in Figure 9.4 show how the algorithm explores several strategies for building two-tiles interactions at the beginning of the run, that is, all the glue strengths are (to a degree) initially represented. As the evolutionary process progresses, several glue strengths are discarded and they become extinct, i.e. their evolutionary waves vanish, only to be replaced with a glue strength of 2, 1 and 3 until generation 80. This happens because strong links – glue strengths bigger than τ – form assemblies too early which are too different, chaotic and small from the target. Therefore, it must preserve glues that by being close to the "edge of chaos", in this instant τ, can co-operatively assemble but also correct early errors by disassembling. From generation 115 onwards, glue strengths 3 drives all the rest to extinction. Since the temperature τ is 4, it is important to notice that this glue strength is the smallest that is closer to τ. Hence, the two-tile assemblies, although not sticking permanently, do have a high chance of staying together. Thus, evolution "tunes" the stickiness of two-tiles generalised secondary structures so as to avoid premature aggregations but still having a high chance of building up higher order GSSs.

9.3 Conclusions

Such is the complexity of the system, that the autonomy, stochastic behaviour and variety of configurations makes it difficult to understand of how large aggregates emerge out of many local interactions that have been fine-tuned by evolution. The first part of this chapter

has presented a methodology to quantify and explore the many self-assembly cooperative mechanisms among tiles. As an overall conclusion, it is important to say that certain types of observed cooperation, defined here as generalized secondary structures, emerge as a product of artificial evolution serving as building blocks for the assembly of greater aggregates. In addition, it is also interesting to note that the less complex the simulation model, the smaller the resulting aggregations as two-tiles self-assembly GSSs become not only common but also frequent building blocks. In contrast, experiments using tiles assisted with probabilistic criteria result in larger aggregations due to the usage of three-tiles, four-tiles and five-tiles cooperativeness. In other words, probabilistic criteria necessitates intermediate larger building blocks for adequate self-assembly to occur. Thus, it seems that a conservation of complexity principle is operating whereby the richer the environment the simpler the individuals and, conversely, the simpler the environment the more complex the individuals.

The second part of this chapter focused on the application of a methodology for the measurement and visualization of evolutionary activity which is a statistic used for analysing adaptive phenomena in evolving systems. In this case, the analysis has been done at the phenotype level in order to track the glue strengths employed by the GSSs across the evolutionary process. In particular, the study has revealed that all the available strength values of the model were found among the first generations of the evolutionary process. Such diversity was reduced to a set of glue strengths with values smaller than τ starting from the third part of the total generations onwards. Thus, the findings revealed that achieving small aggregates is highly likely along the first generations whereas greater assemblies produced by better types of cooperation would emerge towards the last generation.

CHAPTER 10

Conclusions

Self-organisation and self-assembly are both phenomena which are ubiquitous in nature where complex structures and coordinated behaviour emerge as the result of the local exchange of information among the components of the system. Although self-organised structures appear far from any thermodynamic equilibrium and are mostly part of open environments, there are many underlying features common to both concepts such as the lack of master plan, absence of external intervention and the existence of predefined autonomous behaviour at entity level.

Understanding how nature produces, tunes and relies upon these two natural phenomena to manufacture magnificent engineering solutions is of enormous scientific and technical relevance. For this reason, this thesis investigated the automated design of self-assembly and self-organising systems by means of artificial evolution. Towards this goal, this research was principally focused on the use of CA systems, self-assembly Wang tiles and GAs. As a complementary analytical part of the methodology, a dual assessment model based on FDC and clustering as well as the analysis of emergent patterns of cooperativity

and the measurement of adaptive activity across evolution were explored.

These findings support the hypothesis that biologically-inspired methods such as the Darwinian principles of natural selection, reproduction, mutation and survival of the fittest implemented in GAs are suitable mechanisms for the automated design and optimisation of systems where the local interactions among their components play a crucial role for the emergent development of complex structures and coordinated behaviour.

10.1   Contributions

Chapter 5, Chapter 6 and Chapter 7 have contributed strategies for the automated design optimisation of CAs input parameters and self-assembly Wang tiles by means of artificial evolution.

The first approach addresses, in separate studies, a continuous and a discrete design optimisation problem by means of a GA aiming to tune a population of CA input parameters in order to design a spatio-temporal behaviour. For that purpose, individuals have been established as fixed length ensembles of randomly generated values, each of which act as input of a CA system. The iterative application of CA rules produces a two-dimensional pattern captured in an image that, together with a user defined pattern, established as the target, are compared in terms of similarity using USM. Thus, the more similar the generated and the user defined patterns, the closer the collection of parameters to those that gave rise to the target. The results achieved by this approach have been reported in Chapter 5 and Chapter 6 where the input parameter values of Turbulence CA and the input rules of Meta-automaton CA were the subject of design optimisation. In general, visual inspections

of the resulting evolved spatio-temporal patterns obtained from both Turbulence CA and Meta-automaton CA successfully match the targets in most of the cases. This evidence is indeed analytically supported by the USM values associated to the fittest individuals of the experiments. Not withstanding the highly complex, non-linear and stochastic nature of genotype-phenotype-fitness mapping, the findings shown in this dissertation have revealed that in most of the cases it is possible to tune the CA input parameters by means of artificial evolution, hence supporting the evolutionary design optimisation on complex behavioural systems of this kind.

The second approach addresses a discrete optimisation design problem by means of a GA, the aim of which is to design a set of Wang tiles capable of self-assembling in a user defined shape. In this case, the individuals of the population were set up as variable length sets of randomly created Wang tile families, instances of which undergo self-assembly in a fixed time simulation environment. This simulation results in a configuration defined as a group of aggregations which are later compared with a target structure by a lattice scanning function, USM or Minkowski functionals. This comparison returns a value indicating the similarity of the target shape and the resulting self-assembled aggregations. Hence, the more resemblance found, the better designed the collection of tiles families to build the target structure. The findings of this approach were reported in Chapter 7. In this study, three different fitness functions and four increasingly richer simulation environments – classified according to the way in which tiles interact – have been investigated. On the one hand, the use of the lattice scanning as an evaluation function implies not only an expensive computational effort but also uncertainty on whether the embedded tiles enclosed in a certain

scanned position are part of a whole aggregation. Unlike the results obtained in the CA input parameters design optimisation problem, it was also shown that measuring similarities in terms of information distance between the resulting aggregations of the simulator and the target shape is not as effective as when comparing spatio-temporal patterns. On the other hand, the use of a sophisticated fitness function, such as the Minkowski functionals, gives a more appropriate fitness value since a better characterization of the resulting phenotype can be performed. In particular, the computation of Euler, area, perimeter and radius of gyration have contributed to a more accurate evaluation between the self-assembled aggregations and the target shape. With regard to the different models, the experimental results have shown that the use of probabilistic stickiness criteria help the tiles to self-assemble in aggregations greater in size than those achieved when neither deterministic stickiness criteria is present nor rotation is allowed. In addition, considering both the model and the size of the resulting individuals, it was observed that a conservation of complexity principle is operating whereby the richer the environment the simpler the individuals needed to achieve a specific structure and vice versa.

As well as presenting methodologies, Chapter 8 has contributed a complementary dual assessment for validating complex, stochastic, non-linear genotype-phenotype-fitness relationships. The proposed methodology developed in this chapter combines FDC and clustering validation techniques. The results revealed that USM and Minkowski functionals are both suitable fitness functions for the evaluation of spatio-temporal patterns and self-assembled structures. In addition, these indicators have contributed a more general result which is a method for assessing the quality of the encoding and the accuracy of its map-

ping to phenotype in evolutionary systems where genotype-phenotype-fitness is an intricate association.

Finally, Chapter 9 has carried out an analysis of how the evolutionary process "programs" the self-assembly Wang tiles in order to be effective for building a specific target shape. These studies have been carried out by enumerating the generalised secondary structures and by measuring the glue strengths activity in the evolutionary process. The former has revealed that certain types of patterns of cooperation emerge across evolution as natural milestones to achieve the target shape. In addition, the latter has measured and visualised the participation of the glue strengths by means of activity waves which have revealed that, as the evolutionary process progresses, several glue strengths survive to the selective pressure and others become extinct. This last result highlighted a panoramic view where evolution is seen fine-tuning the stickiness of first order generalized secondary structures (i.e. two-tiles elementary assemblies) in order to avoid premature self-assembled aggregations but giving the chance to build up higher order generalized secondary structures (i.e. three, four and five-tiles elementary assemblies).

10.2 Future Directions

The investigations presented in this thesis suggest many directions for further research. For this reason, the following subsections discuss ideas that reinforce current methodologies and outline the potential fields of impact.

10.2.1 Reinforcements

A future study would consider how to improve the operative mechanism in which USM is employed as an evaluation function for the evolutionary design optimisation of self-assembly Wang tiles. One way to do this would be to perform a fine grain comparison rather than a single measure of the whole. Clearly, this comprises many to one calculations that could involve computing the information distances between each aggregation and the target shape separately followed by an average. This alternative way to calculate similarity predicts a thick (in terms of time) layer between the resulting simulation and the evaluation function, and an explosive increase in terms of computational cost. Although current architectures like parallel or multi-thread processing would be responsible for counterbalancing such impact, recognising the aggregates produced by the first generations would still be a very clear bottleneck to speed up. In general terms, this underlying idea could be enriched with the combination of satellite fitness functions capable for strengthening the calculus by measuring the entropy, density or other morphological characteristics of the aggregates such as symmetries.

Since one of the important outcomes of this research was the identification and quantification of generalized secondary structures post experiment, an additional and different evaluation approach would consider the use of a reward-penalization schema in the evaluation stage. That could be implemented as an adaptive evaluation criteria where the contribution of two, three, four and five-tiles self-assembly GSSs would be gradually rewarded. Since individuals are randomly initialised, a first sight of such a scheme suggests predefining milestones, in terms of number of generations, where lower to higher order GSSs

are preferred as the evolutionary process takes place. This in-line enumeration of patterns of cooperativity would inject "small doses" of selective pressure expecting, of course, to improve the quality of the evolved tiles. Notice that integrating such an idea is totally independent of the fitness functions already implemented.

The self-assembly model studied in this thesis is equipped with a very simple physical simulation environment, i.e. a finite size lattice where the glue strengths and the temperature remain fixed. A more realistic version would consider, among others, variable temperature, adaptive glue strength matrix, a non-bounded lattice or the addition of many other physical features such as acceleration, gravity, friction and realistic bouncing. For instance, one can assume that the cooler (warmer) the system becomes, the lower (higher) the acceleration of the tiles certainly affecting the self-assembly dynamics of the system. Similarly, the chosen CA would present a more challenging combination of parameters, dynamic neighbourhood topology, etc.. For instance, it could be considered reasignation of rules across time in the Meta-automaton CA or a combined set of parameters comprising continuous and discrete values. Any combination would offer a different perception of the reality where the proposed evolutionary approach may either outperform the simplistic version or stick in a poor quality local optima. In any case, the stochastic, non-linear intricate relation genotype-phenotype-fitness would still be present undermining the fitness functions. Would the complementary dual assessment (FDC & clustering) still be an appropriate mechanism for analysing the complex mapping ? On the one hand, a brief analysis of the FDC does not project as much positive feedback since the many combinations of the variables estimate a huge search space dimension, i.e. the chosen optimum and the sam-

ples would be parametrically far apart. On the other hand, one could also expect negative feedback from the fitness function due to erroneous distinctions among the phenotypes and, in consequence, meaningless clusters. One of the proposed improvements here would be to change the genotype encoding in order to get a less intricate relationship towards the phenotype. This becomes a problem-dependent approach that would require a thorough analysis of methods of representation and extensive computational time to validate the complexity of the mappings. However, a more tractable and scalable approach would be to perform a variety of independent assessments. That is, in turn, to fix some parameters and to perform studies of FDC & clustering rather than attempt to achieve a "multi-objective" panacea.

10.2.2   Multidisciplinary Impact

Chapter 7 demonstrated the challenge of attempting to design the most adequate set of self-assembling Wang tiles needed to build a certain target shape. This, viewed as an evolutionary design optimisation problem using the representation employed here, is translated in an intricate genotype-phenotype-fitness relationship. Although this requires a more strenuous effort, it is worth considering these questions: what could be the applications where the evolved aggregates would become useful ? What would be the benefits of spending such paramount amount of computation and analysis ?.

Recall that when defining the Tiles Assembly Model, Winfree [12] has shown computation where the south and east edges of a tile work as input ports, the north and west as output ports and the binary message read from bottom to top constitutes output bit strings. With this general idea in mind, one of the inspirations lies in the development of automatic program solving constructions. To be more precise, self-assembly Wang tiles could

be seen as independent heuristics (Figure 10.1 (a)) as their assemblage a solving strategy in full (Figure 10.1 (b)). However, it is still important to discuss the input/output ports and the sequence (if not parallel) to execute the operations. This overall idea can be seen as automatic design of hyper-heuristics feasible for solving NP-hard problems since actual research has proven that tiles are successful [80] [81] [79].

(a)   (b)
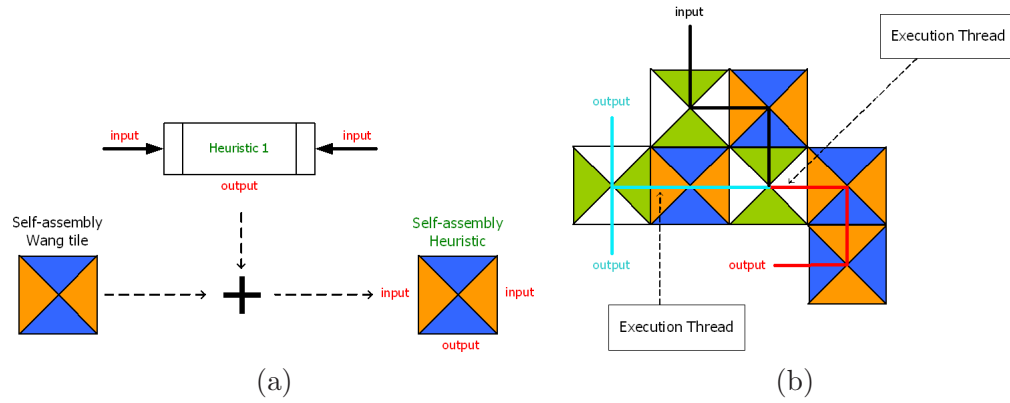
FIGURE 10.1: *A self-assembly Wang tile embedding a heuristic with two inputs and an output (a); an aggregate defining a composition of self-assembly heuristics with two alternative execution threads comprising five heuristics each (b).*

Recent advances in biotechnology, bioinformatics and computational modelling promise deeper understanding of the complexity of biological systems when capturing the internal behaviour in terms of computation. One of the most important mechanisms of this kind that has been discovered is *quorum sensing* (QS) which is defined as a behavioural coordination system that takes place under changing environments [216] [217] [218]. For this coordination to occur, QS relies on the activation of a response regulator protein by a diffusible signal molecule referred as "autoinducer". The concentration level of autoinducers reflects the number of bacterial cells in a particular niche and perception of a threshold concentration of that signal molecule indicates that the population is ready to mount a

unified behavioural decision such as swarming, biofilm formation or antibiotic biosyntesis [219]. A simulation of QS is shown in Figure 10.2.
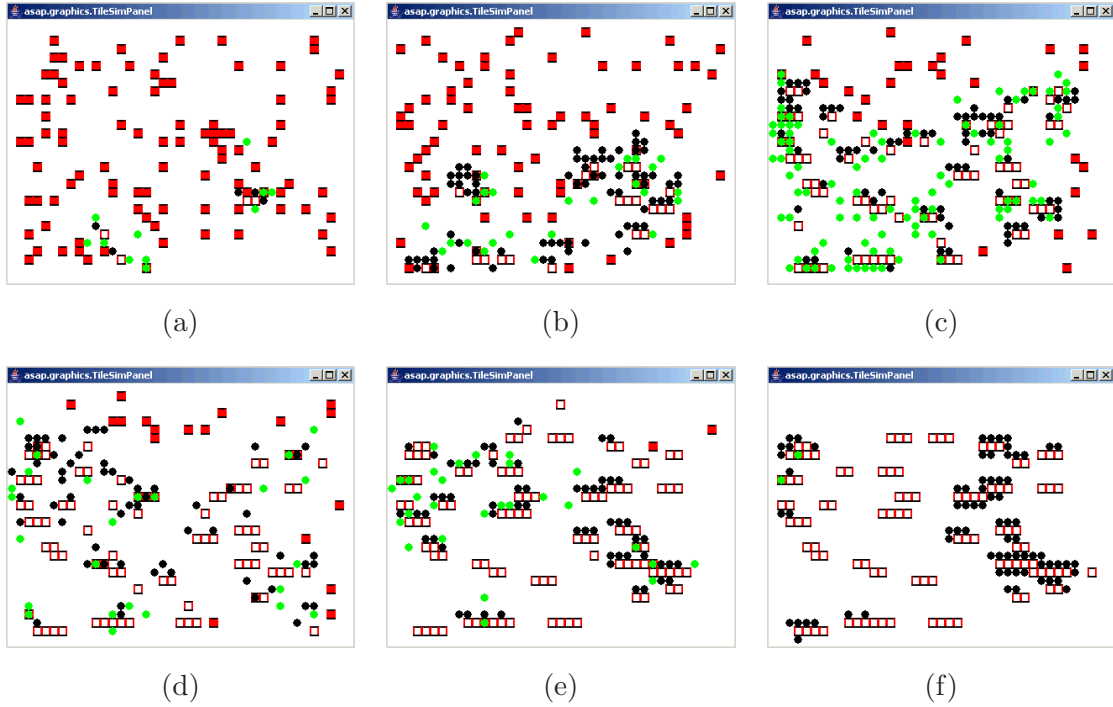
FIGURE 10.2: *A quorum sensing simulation. Signal molecules in the environment (black) induce their replication (green) into the bacteria (red squares). As the simulation progresses from (a) to (f), bacteria aggregate in structures (white squares) interpreted as biofilm.*

This bacterial cell-to-cell communication and coordination system, clearly an instance of self-organisation, was informally proven as computationally complete and modelled in P-systems [220] in some of my initial investigations reported in [221] [222] [223]. Although this findings focused on capturing the main entities involved in QS, i.e. bacteria, autoinducers, environmental rules and topological representation, a substantial expansion capable of simulating richer dynamics of quorum sensing would pursue the grounds for applying any of the evolutionary design optimisation approaches presented in this thesis. Ongoing advances on the field are recently reported in [224] [225].

References

[1] G. Morgenthal and A. McRobie. Millennium Bridge Simulator. University of Cambridge - `http://www.morgenthal.org/MillenniumBridge/`.

[2] B.G. Marcot. Matabele Ants. Ecology Picture of the Week - `http://taos-telecommunity.org/epow/EPOW-Archive/archive_2003/EPOW-030811.htm`.

[3] M. Markus. Pigmentation of the Shells of Molluscs. Max Planck Institute for Molecular Physiology - `http://www.mpi-dortmund.mpg.de/forschungProjekte/AGs/UnabhaengigeAG/swo/Markus/shells/index.html`.

[4] Wikipedia. Polyethylene. Wikipedia, the free encyclopedia - `http://en.wikipedia.org/wiki/Polyethylene`.

[5] Markus Deserno. The Lipid Bilayer. Max Planck Institute for Polymer Research - `http://www.mpip-mainz.mpg.de/~deserno/lipid_bilayer.php`.

[6] Cell Biology Graduate Program. Membrane Structure and Function. University of Texas Medical Branch - `http://cellbio.utmb.edu/cellbio/membrane_intro.htm`.

[7] K. G. Libbrecht. Guide to Snowflakes. Information Management Systems and Ser-

vices CALTECH - `http://www.its.caltech.edu/~atomic/snowcrystals/class/class.htm`.

[8] F. Lacombe. Modeling Swarm Behavior. University of Calgary - `http://www.physorg.com/news11060.html`.

[9] D. S. Jordan. A Book of Natural History. Project Gutenberg - `http://www.gutenberg.org/files/18274/18274-h/18274-h.htm`.

[10] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter. *Molecular Biology of the Cell*. Garland, fourth edition, 2002.

[11] C. Büchen-Osmond. The Universal Virus Database, version 3. International Committee on Taxonomy of Viruses data base - `http://www.ncbi.nlm.nih.gov/ICTVdb/ICTVdB/00.001.htm`.

[12] E. Winfree. Simulations of Computing by Self-Assembly. In *4th Annual Meeting on DNA-Based Computers*, pages 213–242, June 1998.

[13] L. Barone, R. Lyndon While, and P. Hingston. Designing Crushers with a Multi-objective Evolutionary Algorithm. In *Genetic Evolutionary Computation Conference*, pages 995–1002, 2002.

[14] P. J. Bentley and J. P. Wakefield. The Evolution of Solid Object Designs Using Genetic Algorithms. In *Applied Decision Technologies (MDT '95)*, pages 391–400, Apr 1995.

[15] G. S. Hornby and J. B. Pollack. Evolving *L*-systems to generate virtual creatures. *Computers and Graphics*, 25(6):1041–1048, 2001.

[16] G. Poulton, Y. Guo, P. Valencia, G. James, M. Prokopenko, and P. Wang. Designing Enzymes in a Multi-Agent System based on a Genetic Algorithm. In *8th Conference on Intelligent Autonomous Systems*, pages 253–262, Amsterdam, The Netherlands, March 2004.

[17] Y. Guo, G. Poulton, P. Valencia, and G. James. Designing self-assembly for 2-dimensional building blocks. In G. Di Marzo Serugendo et al, editor, *Engineering Self-Organising Systems: Nature-Inspired Approaches to Software Engineering, LNAI 2977*, pages 75–89. Springer Verlag, February 2004.

[18] A. Buchanan, G. Gazzola, and M. Bedau. Evolutionary Design of a Model of Self-Assembling Chemical Structures. In N. Krasnogor, S. Gustafson, D. Pelta, and J. L. Verdegay, editors, *Systems Self-Assembly: Multidisciplinary Snapshots*. Elsevier, 2008.

[19] S. von Mammen, C. Jacob, and G. Kókai. Evolving Swarms that Build 3D Structures. In *IEEE Congress on Evolutionary Computation*, volume 2, pages 1434–1441. IEEE, 2005.

[20] J. M. C. Hutchinson and G. Gigerenzer. Simple heuristics and rules of thumb: where psychologists and behavioural biologists might meet. *Behav Processes*, 69(2):97–124, 2005.

[21] C. Sartiani, P. Manghi, G. Ghelli, and G. Conforti. Xpeer: A Self-Organizing XML P2P Database System. In *9th International Conference on Extending Database Tech-*

nology, volume 3268 of *Lecture Notes in Computer Science*, pages 456–465, Crete, Greece, 2004. Springer.

[22] R. Levinson, D. Helman, and E. Oswalt. Intelligent signal analysis and recognition using a self-organizing database. In *1st international conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, pages 1116–1130, New York, NY, USA, 1988. ACM.

[23] M. Mamei, R. Menezes, R. Tolksdorf, and F. Zambonelli. Case studies for self-organization in computer science. *J. Syst. Archit.*, 52(8):443–460, 2006.

[24] L. Li, N. Krasnogor, and J. Garibaldi. Automated Self-Assembly Programming Paradigm: Initial Investigations. In *3rd IEEE International Workshop on Engineering of Autonomic & Autonomous Systems*, pages 25–36, Washington, DC, USA, 2006. IEEE Computer Society.

[25] S. Stöcker. Models for tuna school formation. *Mathematical Biosciences*, 156(1-2):167 – 190, 1999.

[26] M. Bucolo, S. Carnazza, L. Fortuna, M. Frasca, S. Guglielmino, G. Marletta, and C. Satriano. Self-organization and emergent models in bacterial adhesion on engineered polymer surfaces. *International Symposium on Circuits and Systems*, 3:III–689–92, May 2004.

[27] G. W. Flake. *The Computational Beauty of Nature: Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation*. The MIT Press, Cambridge, Massachusetts, USA, 2000.

[28] F. Heylighen and C. Gershenson. Meaning of Self-organization in Computing. *IEEE Intelligent Systems*, 18(4):72–75, 2003.

[29] J. Kim, Y. Liu, S. J. Ahn, S. Zauscher, J. M. Karty, Y. Yamanaka, and S. L. Craig. Self-Assembly and Properties of Main-Chain Reversible Polymer Brushes. *Advanced Materials*, 17(14):1749–1753, 2005.

[30] R. Klajn, K. J. M. Bishop, and B. A. Grzybowski. Light-controlled self-assembly of reversible and irreversible nanoparticle suprastructures. *Proceedings of the National Academy of Sciences of the United States of America*, 104(25):10305–10309, 2007.

[31] C. Anderson. Self-Organization in Relation to Several Similar Concepts: Are the Boundaries to Self-Organization Indistinct? *Biological Bulletin*, 202(3):247–255, 2002.

[32] A. B. Sendova-Franks and N.R. Franks. Self-assembly, self-organization and division of labour. *Phil. Trans. Roy. Soc. London*, 354(B):1395–1405, 1999.

[33] K. John and M. Bär. Alternative Mechanisms of Structuring Biomembranes: Self-Assembly versus Self-Organization. *Physical Review Letters*, 95(19):198101, 2005.

[34] C. P. Martin, M. O. Blunt, E. Vaujour, A. Fahmi, A. D'Aleo, L. De Cola, F. Vogtle, and P. Moriarty. Self-Organised Nanoparticle Assemblies: A Panoply of Patterns. In N. Krasnogor, S. Gustafson, D. Pelta, and J. L. Verdegay, editors, *Systems Self-Assembly: Multidisciplinary Snapshots*. Elsevier, 2008.

[35] T. Jones. A 2d model for malformations in viral self-assembly. Colorado School of Mines, 2006 - `http://www.mines.edu/Academic/chemeng/courses/chen610`.

[36] B. Berger, P. W. Shor, L. Tucker-Kellogg, and J. King. Local rule-based theory of virus shell assembly. *Proc Natl Acad Sci U S A*, 91(16):7732–7736, 1994.

[37] D. Endres, M. Miyahara, P. Moisant, and A. Zlotnick. A reaction landscape identifies the intermediates critical for self-assembly of virus capsids and other polyhedral structures. *Protein Sci*, 14(6):1518–1525, 2005.

[38] F. Schweitzer. Self-Organization of Complex Structures: From Individual to Collective Dynamics – Some Introductory Remarks. In F. Schweitzer, editor, *Self-Organization of Complex Structures: From Individual to Collective Dynamics*, pages xix–xxiv. CRC, 1997.

[39] F. Schweitzer and J. Zimmermann. Communication and Self-Organisation in Complex Systems: A Basic Approach. In M. M. Fischer and J. Fröhlich, editors, *Knowledge, Complexity and Innovation Systems*, Advances in Spatial Science, pages 275–296. Springer, 2001.

[40] S. Dormann and A. Deutsch. Modeling of Self-Organized Avascular Tumor Growth with a Hybrid Cellular Automaton. In *The German Conference on Bioinformatics*, volume 2, pages 393–406, Oct. 2001.

[41] L. Edelstein-Keshet and G. B. Ermentrout. Models for contact mediated pattern formation: cells that form parallel arrays. *J. Math. Biol.*, 29:33–58, 1990.

[42] G. B. Ermentrout and L. Edelstein-Keshet. Cellular Automata Approaches to Biological Modeling. *Journal of Theoretical Biology*, 160:97–133, Jan 1993.

[43] N. Krasnogor, D. H. Marcos, D. Pelta, and W. A. Risi. Protein Structure Prediction as a Complex Adaptive System. In Cezary Janikow, editor, *Frontiers in Evolutionary Algorithms*, pages 441–447, 1998.

[44] D. G. Green, A. P. N. House, and S. M. House. Simulating Spatial Patterns in Forest Ecosystems. *Mathematics and Computers in Simulation*, 27:191–198, 1985.

[45] M. Droz. Cellular automata approach to pattern formation in reaction-diffusion systems. *Physica A*, 240:239–245, June 1997.

[46] G. A. Giraldi, A. V. Xavier, A. L. Apolinario Jr, and P. S. Rodrigues. Lattice Gas Cellular Automata for Computational Fluid Animation. *Computing Research Repository*, abs/cs/0507012, 2005.

[47] D. Peak, J. D. West, S. M. Messinger, and K. A. Mott. Evidence for complex, collective dynamics and emergent, distributed computation in plants. *Proceedings of the National Academy of Sciences of the United States of America*, 101(4):918–922, 2004.

[48] X. H. Liu and C. Andersson. Assessing the impact of temporal dynamics on land-use change modeling. *Computers, Environment and Urban Systems*, 28(1-2):107–124, 2004.

[49] X.-B. Li, R. Jiang, and Q.-S. Wu. Cellular Automata Model Simulating Complex Spatiotemporal Structure of Wide Jams. *Phys. Rev. E*, 68(1):016117, July 2003.

[50] B. Chopard and M. Droz. *Cellular Automata Modeling of Physical Systems*. Cambridge University Press, 1998.

[51] L. B. Kier, P. G. Seybold, and C. Cheng. *Cellular Automata Modeling of Chemical Systems*. Springer, 2005.

[52] K. Maeda and C. Sakama. Identifying Cellular Automata Rules. *Journal of Cellular Automata*, 2(1):1–20, 2007.

[53] Y. Yang and S. A. Billings. Neighborhood detection and rule selection from cellular automata patterns. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 30(6):840–847, 2000.

[54] Y. Zhao and S. A. Billings. Neighborhood Detection Using Mutual Information for the Identification of Cellular Automata. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 36(2):473–479, 2006.

[55] A. Adamatzky. *Identification of cellular automata*. Taylor & Francis, London, 1994.

[56] A. Adamatzky. Automatic programming of cellular automata: identification approach. *Kybernetes*, 26:126–135, 1997.

[57] S. Wolfram. *A New Kind of Science*. Wolfram Media Inc., Champaign, Ilinois, US, United States, 2002.

[58] W. Sierpinski. Sur une courbe dont tout point est un point de ramification. *C. R. Acad. Sci. Paris*, 160:302–305, 1915.

[59] D. Anagnostou, M. T. Chryssomallis, J. C. Lyke, and C. G. Christodoulou. Reconfigurable Sierpinski gasket antenna using RF-MEMS switches. In *Antennas and*

Propagation Society International Symposium, volume 1, pages 375–378, The University of New Mexico, 2003. IEEE Press.

[60] J. Romeu and Y. Rahmat-Samii. Dual band FSS with fractal elements. *Electronics Letters*, 35(9):702–703, 1999.

[61] R. Holakouei, C. Ghobadi, and J. Nooriniya. A novel design of multi-band FSS with sierpinski gasket tripole elements. In *Antennas, Radar, and Wave Propagation*, Banff, Alberta, Canada, 2005. Acta Press.

[62] H. Wang. Proving Theorems by Pattern Recognition I. *Communications of the ACM*, 3(4):220–234, 1960.

[63] R. Berger. The Undecidability of the Domino Problem. *Memoirs American Mathematical Society*, 66:1–72, 1966.

[64] R. M. Robinson. Undecidability and Nonperiodicity for Tilings of the Plane. *Inventiones Mathematicae*, 12(3):177–209, 1971.

[65] J. A. Pelesko. *Self-Assembly*. Chapman & Hall/CRC, first edition, 2007.

[66] G. J. Chaitin. The halting probability via wang tiles. *Fundamenta Informaticae*, 86(3-4):429–433, 2008.

[67] E. Winfree, F. Liu, L. A. f, and N. C. Seeman. Design and self-assembly of two-dimensional DNA crystals. *Nature*, 394:539–544, 1998.

[68] J. Chen and N.C. Seeman. Synthesis from DNA of a molecule with the connectivity of a cube. *Nature*, 350:631–633, 1991.

[69] H. Yan, T. H. LaBean, L. Feng, and J. H. Reif. Directed nucleation assembly of DNA tile complexes for barcode-patterned lattices. *Proceedings of the National Academy of Sciences of the United States of America*, 100(14):8103–8108, 2003.

[70] S. H. Park, C. Pistol, S. J. Ahn, J. H. Reif, A. R. Lebeck, C. Dwyer, and T. H. LaBean. Finite-size, fully addressable DNA tile lattices formed by hierarchical assembly procedures. *Angew Chem. Int.*, 45(5):735–739, 2006.

[71] P. W. K. Rothemund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440:297–302, 2006.

[72] D. Soloveichik and E. Winfree. Complexity of Self-assembled Shapes. In *10th International Workshop on DNA Computing*, volume 3384 of *Lecture Notes in Computer Science*, pages 344–354, Milan, Italy, 2005. Springer Berlin / Heidelberg.

[73] P. W. K. Rothemund and E. Winfree. The program-size complexity of self-assembled squares (extended abstract). In *32nd ACM symposium on Theory of computing*, pages 459–468, New York, NY, USA, 2000. ACM.

[74] L. Adleman, Q. Cheng, A. Goel, and M.-D. Huang. Running Time and Program Size for Self-assembled Squares. In *33rd annual ACM Symposium on Theory of Computing*, pages 740–748, New York, NY, USA, 2001. ACM.

[75] L. Adleman, Q. Cheng, A. Goel, M. Huang, D. Kempe, P. Moisset de Espanes, P. Wilhelm, and K. Rothemund. Combinatorial optimization problems in self-assembly. In *34th annual ACM symposium on Theory of computing*, pages 23–32, New York, NY, USA, 2002. ACM Press.

[76] D. Soloveichik and E. Winfree. Complexity of Self-Assembled Shapes. *SIAM Journal on Computing*, 36(6):1544–1569, 2007.

[77] G. Aggarwal, M. H. Goldwasser, M.-Y. Kao, and R. T. Schweller. Complexities for generalized models of self-assembly. In *55th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 880–889, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.

[78] S. Sahu, P. Yin, and J. H. Reif. A Self-assembly Model of Time-Dependent Glue Strength. In *11th International Workshop on DNA Computing*, volume 3892 of *Lecture Notes in Computer Science*, pages 290–304. Springer, 2006.

[79] Y. Brun. Solving NP-complete problems in the tile assembly model. *Theor. Comput. Sci.*, 395(1):31–46, 2008.

[80] Y. Brun. Constant-Size Tileset for Solving an NP-Complete Problem in Nondeterministic Linear Time. In *13th International Meeting on DNA Computing*, volume 4848 of *Lecture Notes in Computer Science*, pages 26–35. Springer, 2008.

[81] Y. Brun. Reducing Tileset Size: 3-SAT and Beyond. In *14th International Meeting on DNA Computing*, page 178, Prague, Czech Republic, June 2008.

[82] T. Back, D. B. Fogel, and Z. Michalewicz, editors. *Handbook of Evolutionary Computation*. IOP Publishing Ltd., Bristol, UK, 1997.

[83] A. E. Eiben. Evolutionary computing: the most powerful problem solver in the universe? *Dutch Mathematical Archive (Nederlands Archief voor Wiskunde)*, 5/3(2):126–131, 2002.

[84] A. E. Eiben and M. Schoenauer. Evolutionary computing. *Information Processing Letters*, 80(1):1–6, 2002.

[85] D. E. Goldberg. *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 2002.

[86] P. Merz. NK-Fitness Landscapes and Memetic Algorithms with Greedy Operators and k-opt Local Search. In W. E. Hart, N. Krasnogor, and J. E. Smith, editors, *Recent Advances in Memetic Algorithms*, volume 166 of *Studies in Fuzziness and Soft Computing*, pages 209–228. Springer, 2004.

[87] C. Darwin. *The Origin of Species*. Gramercy, May 1995.

[88] H. Mühlenbein, M. Gorges-Schleuter, and O. Krämer. Evolution algorithms in combinatorial optimization. *Parallel Computing*, 7(1):65–85, 1988.

[89] C. del Valle, R. M. Gasca, M. Toro, and E. F. Camacho. A Genetic Algorithm for Assembly Sequence Planning. In *7th International Work-Conference on Artificial and Natural Neural Networks*, pages 337–344, Menorca, Spain, 2003.

[90] D. B. Jourdan and O. L. de Weck. Multi-objective genetic algorithm for the automated planning of a wireless sensor network to monitor a critical facility. In E. M. Carapezza, editor, *SPIE Defense and Security Symposium*, pages 565–575, 2004.

[91] H. Richter and K. J. Reinschke. Optimization of local control of chaos by an evolutionary algorithm. *Physica D*, 144(3-4):309–334, 2000.

[92] D. B. Fogel. *Evolutionary Computation: The Fossil Record*. Wiley-IEEE Press, 1998.

[93] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[94] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence.* MIT Press, Cambridge, MA, USA, 1992.

[95] M. Mitchell. *An introduction to genetic algorithms.* MIT Press, Cambridge, MA, USA, 1996.

[96] Z. Michalewicz. *Genetic Algorithms + Dta Structures = Eolution Programs (3rd ed.).* Springer-Verlag, London, UK, 1996.

[97] J. R. Koza. *Genetic programming: on the programming of computers by means of natural selection.* MIT Press, Cambridge, MA, USA, 1992.

[98] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution.* John Wiley, New York, USA, 1966.

[99] I. Rechenberg. *Evolutions strategie – Optimierung technisher Systeme nach Prinzipien der biologischen Evolution.* Frommann-Holzboog, Stuttgart, GER, 1973.

[100] H. G. Beyer and H. P. Schwefel. Evolution Strategies a Comprehensive Introduction. *Natural Computing: An International Journal,* 1(1):3–52, 2002.

[101] J. H. Holland and J. S. Reitman. Cognitive systems based on adaptive algorithms. In D. Hayes-Roth and F. Waterman, editors, *Pattern-directed Inference Systems,* pages 313–329. Academic Press, New York, 1978.

[102] S. W. Wilson. Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175, 1995.

[103] S. F. Smith. Flexible Learning of Problem Solving Heuristics Through Adaptive Search. In *8th International Joint Conference on Artificial Intelligence*, pages 421–425, Los Altos, CA, 1983. Morgan Kaufmann.

[104] T. Bäck and H.-P. 'fel. An Overview of Evolutionary Algorithms for Parameter Optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

[105] J. Bacardit, M. Stout, J. D. Hirst, K. Sastry, X. Llorà, and N. Krasnogor. Automated Alphabet Reduction Method with Evolutionary Algorithms for Protein Structure Prediction. In *Genetic and Evolutionary Computation Conference*, pages 346–353, New York, NY, USA, 2007. ACM.

[106] E. K. Burke, M. R. Hyde, G. Kendall, and J. Woodward. Automatic Heuristic Generation with Genetic Programming: Evolving a Jack-of-all-Trades or a Master of One. In *Genetic and Evolutionary Computation Conference*, pages 1559–1565, New York, NY, USA, 2007. ACM.

[107] N. Pillay and W. Banzhaf. A Genetic Programming Approach to the Generation of Hyper-Heuristics for the Uncapacitated Examination Timetabling Problem. In *13th Portuguese Conference on Aritficial Intelligence*, pages 223–234. Springer, 2007.

[108] C. Gondro and B.P. Kinghorn. A simple genetic algorithm for multiple sequence alignment. *Genetics and Molecular Research*, 6(4):964–982, 2007.

[109] G. Ochoa, M. Villasana, and E. K. Burke. An Evolutionary Approach to Cancer Chemotherapy Scheduling. *Genetic Programming and Evolvable Machines*, 8(4):301–318, 2007.

[110] C. Schoreels and J.M. Garibaldi. Genetic Algorithm Evolved Agent-based Equity Trading using Technical Analysis and the Capital Asset Pricing Model. In *6th International Conference on Recent Advances in Soft Computing 2006*, pages 194–199, Canterbury, UK, 2006.

[111] U. Markowska-Kaczmar, H. Kwasnicka, and M. Szczepkowski. Genetic Algorithm as a Tool for Stock Market Modelling. In *Artificial Intelligence and Soft Computing*, pages 450–459, 2008.

[112] R. da S. Torres, ao A. X. Falc M. A. Gonçalves, J. P. Papa, B. Zhang, W. Fan, and E. A. Fox. A genetic programming framework for content-based image retrieval. *Pattern Recognition*, 42(2):283–292, 2009.

[113] A. D. Parkins and A. K. Nandi. Genetic programming techniques for hand written digit recognition. *Signal Processing*, 84(12):2345–2365, 2004.

[114] A. Oduguwa, A. Tiwari, S. Fiorentino, and R. Roy. Multi-objective Optimisation of the Protein-Ligand Docking Problem in Drug Discovery. In *Genetic and Evolutionary Computation Conference*, pages 1793–1800, New York, NY, USA, 2006. ACM.

[115] C. Fayad and S. Petrovic. A Genetic Algorithm for the Real World Fuzzy Job-Shop Scheduling. In *International Conference on Industrial and Engineering Applications*

of Artificial Intelligence and Expert Systems, Lecture Notes in Computer Science. Springer-Verlag, 2005.

[116] P. J. Bentley. *Evolutionary Design by Computers*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[117] P. J. Bentley. Aspects of Evolutionary Design by Computers. In *3rd On-line World Conference on Soft Computing in Engineering Design and Manufacturing*, 1998.

[118] P. J. Bentley and J. P. Wakefield. The Table: An Illustration of Evolutionary Design using Genetic Algorithms. In *1st International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, pages 412–418, Sheffield, UK, 1995. IEEE Press.

[119] J. Rieffel and J. Pollack. Evolving Assembly Plans for Fully Automated Design and Assembly. In *NASA/DoD Conference on Evolvable Hardware*, pages 165–170, Washington, DC, USA, 2005. IEEE Computer Society.

[120] P. J. Funes. *Evolution of complexity in real-world domains*. PhD thesis, School of Arts and Sciences, Brandeis University, 2001.

[121] H. Takagi. Interactive Evolutionary Computation: Fusion of the Capabilities of EC Optimization and Human Evaluation. *Proceedings of the IEEE*, 89(9):1275–1296, 2001.

[122] Y. Semet. Interactive Evolutionary Computation: a survey of existing theory, 2002.

[123] F. Boschetti. Controlling and investigating cellular automaton behavior via interactive

inversion and visualization of the search space. *New Gen. Comput.*, 23(2):157–169, 2005.

[124] H. Takagi. Auditory and Visual-based Signal Processing with Interactive Evolutionary Computation. In *2nd Euro-International Symposium on Computational Intelligence*, Kosice, Slovakia, 2002.

[125] A. M. Brintrup, J. Ramsden, H. Takagi, and A. Tiwari. Ergonomic Chair Design by Fusing Qualitative and Quantitative Criteria Using Interactive Genetic Algorithms. *Transactions on Evolutionary Computation*, 12(3):343–354, 2008.

[126] J.R. Smith. Designing Biomorphs with an Interactive Genetic Algorithm. In R. K. Belew and L. B. Booker, editors, *4th International Conference on Genetic Algorithms*, pages 535–538, 1991.

[127] J. A. Pelesko. Self-Assembly – Promises and Challenges. In *International Conference on MEMS, NANO and Smart Systems*, pages 427–428, Washington, DC, USA, 2005. IEEE Computer Society.

[128] S. Camazine, J. L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E.Bonabeau. *Self-Assembly.* Princeton University Press, 2003.

[129] Y. Guo, G. Poulton, C. Murray, and G. James. Designing self-assembly DNA-based structures in a multi-agent system. In *7th Asia-Pacific Conference on Complex Systems*, pages 37–648, Cairns, Australia, December 2004.

[130] M. Theis, G. Gazzola, M. Forlin, I. Poli, Ma. Hanczyc, and M. Bedau. Optimal

Formulation of Complex Chemical Systems with a Genetic Algorithm. In *European Confedence on Complex Systems Society*, Oxford, UK, 2006.

[131] A. Tversky. Features of Similarity. *Psychological Review*, 84(4):327–352, 1977.

[132] U. Hahn, N. Chater, and L. B. Richardson. Similarity as transformation. *Cognition*, 87(1):1–32, 2003.

[133] R. L. Goldstone and J. Y. Son. Similarity. In K. Holyoak and R. Morrison, editors, *Cambridge Handbook of Thinking and Reasoning*, pages 13–36. Cambridge University Press, 2005.

[134] M. Li and P. M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications (Texts in Computer Science)*. Springer, 1997.

[135] A. Gammerman and V. Vovk. Kolmogorov Complexity: Sources, Theory and Applications. *The Computer Journal*, 42(4):252–255, 1999.

[136] M. Li, X. Chen, X. Li, B. Ma, and P. Vitányi. The similarity metric. In *14th ACM-SIAM Symposium on Discrete Algorithms*, pages 863–872, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[137] R. Cilibrasi and P. M. B. Vitányi. Clustering by compression. *Information Theory, IEEE Transactions on*, 51(4):1523–1545, 2005.

[138] M. Li, J. H. Badger, C. Xin, S. Kwong, P. Kearney, and H. Zhang. An Information Based Sequence Distance and its Application to Whole Mitochondrial Genome Phylogeny. *Bioinformatics*, 17(2):149–154, 2001.

[139] R. Cilibrasi, P. Vitanyi, and R. de Wolf. Algorithmic clustering of music. In *4th International Conference on Web Delivering of Music*, pages 110–117, 2004.

[140] X. Chen, B. Francia, M. Li, B. McKinnon, and A. Seker. Shared information and program plagiarism detection. *IEEE Transactions on Information Theory*, 50(7):1545–1551, 2004.

[141] C. Bennett, M. Li, and B. Ma. Linking chain letters. *Scientific American*, pages 77–81, June 2003.

[142] D. A. Pelta, J. R. Gonzales, and N. Krasnogor. Protein Structure Comparison through Fuzzy Contact Maps and the Universal Similarity Metric. In *4th Conference of the European Society for Fuzzy Logic and Technology and 11 Recontres Francophones sur la Logique Floue et ses Applications*, pages 1124–1129, Barcelona, Spain, September 2005. EUSFLAT-LFA.

[143] D. Barthel, J. D. Hirst, J. Blacewicz, and N. Krasnogor. Procksi: a Metaserver for Protein Comparison Using Kolmogorov and Other Similarity Measures. In *Late Breaking Papers. European Conference on Computational Biology*, Eilat, Israel, 2006. Peer reviewed CDROM publication.

[144] J. D. Hoheisel. Microarray technology: beyond transcript profiling and genotype analysis. *Nat Rev Genet*, 7:200–210, March 2006.

[145] M. Nykter, O. Yli-Harja, and I. Shmulevich. Normalized compression distance for gene expression analysis. In *3rd IEEE Workshop on Genomic Signal Processing and Statistics*, page 2, 2005.

[146] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. *Science*, 286(5439):531–537, 1999.

[147] E. J. Wood. Applying Fourier and Associated Transforms to Pattern Characterization in Textiles. *Textile Research Journal*, 60(4):212–220, 1990.

[148] T. Matsuyama, S.I. Miura, and M. Nagao. Structural Analysis of Natural Textures by Fourier Transformation. *Computer Vision, Graphics, and Image Processing*, 24(3):347–362, 1983.

[149] T. I. Hsu, A. D. Calway, and R. Wilson. Texture Analysis Using the Multiresolution Fourier Transform. In *8th Scandinavian Conference on Image Analysis*, pages 823–830. IAPR, July 1993.

[150] D. Gibson and P. A. Gaydecki. Definition and application of a Fourier domain texture measure: Applications to histological image segmentation. *Computers in Biology and Medicine*, 25(6):551–557, November 1995.

[151] A. Graps. An Introduction to Wavelets. *IEEE Comput. Sci. Eng.*, 2(2):50–61, 1995.

[152] M. Unser. Splines and Wavelets: New Perspectives for Pattern Recognition. In B. Michaelis and G. Krell, editors, *25th Pattern Recognition Symposium*, volume 2781 of *Lecture Notes in Computer Science*, pages 244–248, Magdeburg (Sachsen-Anhalt), Germany, September 2003. Springer.

[153] G. Van De Wouwer, B. Weyn, P. Scheunders, W. Jacob, E. Van Marck, and D. Van Dyck. Wavelets as chromatin texture descriptors for the automated identification of neoplastic nuclei. *Journal of Microscopy*, 197 (Pt 1):25–35, January 2000.

[154] W. Sweldens. Wavelets: What Next? *Proceedings of the IEEE*, 84(4):680–685, 1996.

[155] V. Steinberg, E. Moses, and J. Fineberg. Spatio-temporal complexity at the onset of convection in a binary fluid. *Nuclear Physics B Proceedings Supplements*, 2:109–123, November 1987.

[156] K. R. Mecke. Morphological characterization of patterns in reaction-diffusion systems. *Phys. Rev. E*, 53(5):4794–4800, 1996.

[157] K. Michielsen and H. De Raedt. Morphological image analysis. *Computer Physics Communications*, 1:94–103, 2000.

[158] K. Michielsen and H. De Raedt. Integral-geometry morphological image analysis. *Physics Reports*, 347:461–538, July 2001.

[159] D. Stoyan, W. S. Kendall, and J. Mecke. *Stochastic Geometry and its Applications*. Akademie Verlag, Berlin, 1989.

[160] J. S. Kole, K. Michielsen, and H. De Raedt. Morphological Image Analysis of Quantum Motion in Billiards. *Physical Review*, 63(1), 2001.

[161] J. Schmalzing, M. Kerscher, and T. Buchert. Minkowski Functionals in Cosmology. In *Proceedings of Int. School of Physics Enrico Fermi*, 1995.

[162] A. V. Mordvinov, I. I. Salakhutdinova, L. A. Plyusnina, N. G. Makarenko, and L. M. Karimova. The Topology of Background Magnetic Fields and Solar Flare Activity. *Solar Physics*, 211:241–253, 2002.

[163] M. S. Barbosa, L. da Fontoura Costa, and E. de Sousa Bernardes. Neuromorphometric characterization with shape functionals. *Phys. Rev. E*, 67(6):061910, June 2003.

[164] L. Karimova and N. Makarenko. Analysis of spatiotemporal data by Minkowski functionals. *EGS - AGU - EUG Joint Assembly*, April 2003.

[165] K. Michielsen, H. De Raedt, and J. G. E. M. Fraaije. Morphological Characterization of Spatial Patterns. *Progress of Theoretical Physics*, Suppl(138):543–548, 2000.

[166] G. J. Agur Sevink. Mathematical Description of Nanostructures with Minkowski Functionals. In Andrei V. Zvelindovsky, editor, *Nanostructured Soft Matter*, pages 269–299. Springer Netherlands, New York, 2007.

[167] P. Siepman, G. Terrazas, and N. Krasnogor. Evolutionary Design for the Behaviour of Cellular Automata-based Complex Systems. In *7th International Conference of Adaptive Computing in Design and Manufacture*, pages 199–208. The Institute for People-centred Computation, 2006.

[168] U. Wilensky. Netlogo. 1999 - `http://ccl.northwestern.edu/netlogo`. Center for Connected Learning and Computer-Based Modeling. Northwestern University, Evanston, IL.

[169] K. Kaneko. *Theory and Applications of Coupled Map Lattices*. Wiley, New York, 1993.

[170] D. Goldberg. Real-coded Genetic Algorithms, Virtual Alphabets, and Blocking. *Complex Systems*, 5:139–167, 1991.

[171] G. Sywerda. Uniform crossover in genetic algorithms. In *3rd International Conference on Genetic algorithms*, pages 2–9, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[172] Heinz Mühlenbein and Dirk Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm i. continuous parameter optimization. *Evolutionary Computation*, 1(1):25–49, 1993.

[173] D. E. Goldberg and K. Deb. A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. In G.J.E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann, 1991.

[174] R. Dawkins. *Climbing Mount Improbable.* W. W. Norton & Company, New York, USA, 1996.

[175] G. Terrazas, P. Siepman, G. Kendal, and N. Krasnogor. An Evolutionary Methodology for the Automated Design of Cellular Automaton–Based Complex Systems. *Journal of Cellular Automata*, 2(1):77–102, 2007.

[176] M. Sipper, M. Tomassini, and M. S. Capcarrere. Designing Cellular Automata Using A Parallel Evolutionary Algorithm. *Nuclear Instruments & Methods in Physics Research, Section A*, 389(1–2):278–283, 1997.

[177] N. Krasnogor, G. Terrazas, D.A. Pelta, and G. Ochoa. A Critical View of the Evolutionary Design of Self-assembling Systems. In *2005 Conference on Artificial Evolution*,

volume 3871 of *Lecture Notes in Computer Science*, pages 179–188, Lille, France, 2005. Springer.

[178] G. Terrazas, N. Krasnogor, G. Kendall, and M. Gheorghe. Automated Tile Design for Self-Assembly Conformations. In *IEEE Congress on Evolutionary Computation*, volume 2, pages 1808–1814. IEEE Press, 2005.

[179] G. Terrazas, M. Gheorghe, G. Kendall, and N. Krasnogor. Evolving Tiles for Automated Self-Assembly Design. In *IEEE Congress on Evolutionary Computation*, pages 2001–2008. IEEE Press, 2007.

[180] S. J. Gould. *The Structure of Evolutionary Theory*. Harvard University Press, 2002.

[181] T. Jones and S. Forrest. Fitness Distance Correlation as a Measure of Problem Difficulty for Genetic Algorithms. In *6th International Conference on Genetic Algorithms*, pages 184–192, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[182] T. Jones. *Evolutionary Algorithms, Fitness Landscapes and Search*. PhD thesis, University of New Mexico, May 1995.

[183] M. Clergue, P. Collard, M. Tomassini, and L. Vanneschi. Fitness Distance Correlation and Problem Difficulty for Genetic Programming. In *Genetic and Evolutionary Computation Conference*, pages 724–732, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[184] L. Vanneschi, M. Tomassini, P. Collard, and M. Clergue. A Survey of Problem Difficulty in Genetic Programming. In S. Bandini and S. Manzoni, editors, *AI*IA 2005:*

Advances in Artificial Intelligence, volume 3673/2005 of *Studies in Fuzziness and Soft Computing*, pages 66–77. Springer Berlin / Heidelberg, 2005.

[185] R. J. Quick, Victor J. Rayward-Smith, and G. D. Smith. Fitness Distance Correlation and Ridge Functions. In *5th International Conference on Parallel Problem Solving from Nature*, pages 77–86, London, UK, 1998. Springer-Verlag.

[186] J. Koljonen. On fitness distance distributions and correlations, GA performance, and population size of fitness functions with translated optima. In T. Honkela, J. Kortela, T. Raiko, and H. Valpola, editors, *9th Scandinavian Conference on Artificial Intelligence*, pages 68–74, Espoo, Finland, 2006. Finnish Artificial Intelligence Society.

[187] L. Altenberg. Fitness Distance Correlation Analysis: An Instructive Counterexample. In *7th International Conference on Genetic Algorithms*, pages 57–64, San Francisco, CA, USA, 1997. Morgan Kaufmann.

[188] L. Vanneschi and M. Tomassini. Pros and Cons of Fitness Distance Correlation in Genetic Programming. In Alwyn M. Barry, editor, *Bird of a Feather Workshops, Genetic and Evolutionary Computation Conference*, pages 284–287, Chigaco, 11 July 2003. AAAI.

[189] M. Tomassini, L. Vanneschi, P. Collard, and M. Clergue. A Study of Fitness Distance Correlation as a Difficulty Measure in Genetic Programming. *Evolutionary Computation*, 13(2):213–239, 2005.

[190] L. Vanneschi, M. Tomassini, P. Collard, and M. Clergue. Fitness Distance Correlation in Structural Mutation Genetic Programming. In C. Ryan, T. Soule, M. Keijzer,

E. Tsang, R. Poli, and E. Costa, editors, *Genetic Programming, Proceedings of EuroGP*, volume 2610 of *Lecture Notes in Computer Science*, pages 455–464, Essex, 2003. Springer-Verlag.

[191] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Intelligent Information Systems*, 17(2-3):107–145, 2001.

[192] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, September 1999.

[193] P. Berkhin. A Survey of Clustering Data Mining Techniques. In J. Kogan, C. Nicholas, and M. Teboulle, editors, *Grouping Multidimensional Data*, pages 25–71. Springer Berlin / Heidelberg, 2006.

[194] P. Berkhin. Survey of Clustering Data Mining Techniques. Technical report, Accrue Software, San Jose, CA, USA, 2002.

[195] S. Kotsiantis and P. Pintelas. Recent Advances in Clustering: A Brief Survey. *Transactions on Information Science and Applications*, 1(1):73–81, 2004.

[196] P. Hansen and B. Jaumard. Cluster analysis and mathematical programming. *Math. Program.*, 79(1-3):191–215, 1997.

[197] S. D. Kamvar, D. Klein, and C. D. Manning. Interpreting and Extending Classical Agglomerative Clustering Algorithms using a Model-Based approach. In *19th International Conference on Machine Learning*, pages 283–290, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[198] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids.* Cambridge University Press, 1998.

[199] S. R. Henz, D. H. Huson, A. F. Auch, K. Nieselt-Struwe, and S. C. Schuster. Whole-genome prokaryotic phylogeny. *Bioinformatics*, 21(10):2329–2335, 2005.

[200] N. Krasnogor and D. A. Pelta. Measuring the similarity of protein structures by means of the universal similarity metric. *Bioinformatics*, 20(7):1015–1021, 2004.

[201] P. Merz. Advanced fitness landscape analysis and the performance of memetic algorithms. *Evol. Comp.*, 12(3):303–325, 2004.

[202] J. Brzustowski. Clustering Calculator. `http://www2.biology.ualberta.ca/jbrzusto/cluster.php`.

[203] L. Li, P. Siepmann, J. Smaldon, G. Terrazas, and N. Krasnogor. Automated Self-Assembling Programming. In N. Krasnogor, S. Gustafson, D. Pelta, and J. L. Verdegay, editors, *Systems Self-Assembly: Multidisciplinary Snapshots.* Elsevier, 2008.

[204] N. A. Baas. Self-Organization and Higher Order Structures. In F. Schweitzer, editor, *Self-Organization of Complex Structures: From Individual to Collective Dynamics*, pages xix–xxiv. CRC, 1997.

[205] N. A. Baas, M. W. Olesen, and S. Rasmussen. Generation of Higher Order Emergent Structures. Working Papers 96-08-057, Santa Fe Institute, 1996.

[206] K. Saraboji, M. M. Gromiha, and M. N. Ponnuswamy. Importance of main-chain hy-

drophobic free energy to the stability of thermophilic proteins. *International Journal of Biological Macromolecules*, 35(3-4):211–220, 2005/4.

[207] M. J. Raven and M. A. Bedau. General Framework for Evolutionary Activity. In W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler, editors, *7th European Conference in Artificial Life*, volume 2801 of *Lecture Notes in Computer Science*, pages 676–685. Springer, 2003.

[208] M. A. Bedau and C. Titus Brown. Visualizing evolutionary activity of genotypes. *Artificial Life*, 5(1):17–35, 1999.

[209] A. Skusa and M. A. Bedau. Towards a comparison of evolutionary creativity in biological and cultural evolution. In *8th international conference on Artificial life*, pages 233–242, Cambridge, MA, USA, 2003. MIT Press.

[210] M. A. Bedau and N. H. Packard. Measurement of Evolutionary Activity, Teleology, and Life. In C. G. Langton, C. Taylor, J. Doyne Farmer, and S. Rasmussen, editors, *Artificial Life II*, pages 431–461. Addison-Wesley, Redwood City, CA, 1992.

[211] M. J. Benton. *The fossil record 2*. Chapman and Hall, London, 1993.

[212] R. G. Palmer, W. B. Arthur, J. H. Holland, B. LeBaron, and P. Tayler. Artificial economic life: a simple model of a stockmarket. *Phys. D*, 75(1-3):264–274, 1994.

[213] K. Lindgren. Evolutionary Phenomena in Simple Dynamics. In C. G. Langton, C. Taylor, J. Doyne Farmer, and S. Rasmussen, editors, *Artificial Life II*, pages 295–312. Addison-Wesley, Redwood City, CA, USA, 1992.

[214] T. S. Ray. An Approach to the Synthesis of Life. In C. G. Langton, C. Tayler, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II*, pages 371–408. Addison-Wesley, Reading, MA, USA, 1992.

[215] N. Krasnogor and J. Smith. Emergence of Profitable Search Strategies Based on a Simple Inheritance Mechanism. In L. Spector, E. Goodman, A. Wu, W.B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. Garzon, and E. Burke, editors, *Genetic and Evolutionary Computation Conference*, pages 432–439, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[216] K. Winzer, K. H. Hardie, and P. Williams. Bacterial cell-to-cell communication: sorry can't talk now – gone to lunch! *Current Opinion in Microbiology*, 5(2):216–222, 2002.

[217] S. Busby and V. de Lorenzo. Cell regulation – Putting together pieces of the big puzzle. *Current Opinion in Microbiology*, 4(2):117–118, 2001.

[218] S. Swift, J. A. Downie, N. A. Whitehead, A. M. L. Barnard, G. P. C. Salmond, and P. Williams. Quorum sensing as a population-density-dependent determinant of bacterial physiology. *Advances in Microbial Physiology*, 45:199–270, 2001.

[219] P. Williams, M. Camara, A. Hardman, S. Swift, D. Milton, V. J. Hope, K. Winzer, B. Middleton, D. I. Pritchard, and B. W. Bycroft. Quorum sensing and the population dependent control of virulence. *Phil. Trans. Roy. Soc. London B*, 355(1397):667–680, 2000.

[220] G. Păun. *Membrane Computing. An Introduction.* Springer-Verlag, Berlin, 2002.

[221] F. Bernardini, M. Gheorghe, N. Krasnogor, and G. Terrazas. Membrane Computing – current results and future problems. In B. Cooper, B. Lowe, and L. Torenvliet, editors, *Computability in Europe*, volume 3526 of *Lecture Notes in Computer Science*, pages 49–53, Amsterdam, The Netherlands, June 2005. Springer-Verlag.

[222] N. Krasnogor, M. Gheorghe, G. Terrazas, S. Diggle, P. Williams, and M. Camara. An Appealing Computational Mechanism Drawn from Bacterial Quorum Sensing. *Bulletin of the European Association of Theoretical Computer Science*, (85):135–148, February 2005.

[223] G. Terrazas, N. Krasnogor, M. Georghe, F. Bernardini, S. Diggle, and M. Camara. An Environment Aware P-Systems model of Quorum Sensing. In B. Cooper, B. Lowe, and L. Torenvliet, editors, *Computability in Europe*, volume 3526 of *Lecture Notes in Computer Science*, pages 479–485, Amsterdam, The Netherlands, June 2005. Springer-Verlag.

[224] L. Bianco, D. Pescini, P. Siepman, N. Krasnogor, F. J. Romero-Campero, and M. Gheorghe. Towards a P-Systems Pseudomonas Quorum Sensing Model. In G. Păun, editor, *7th International Workshop on Membrane Computing*, volume 4361 of *Lecture Notes in Computer Science*, pages 197–214. Springer, 2006.

[225] F. Romero-Campero, H.Cao, M. Camara, and N. Krasnogor. Structure and Parameter Estimation for Cell Systems Biology Models. In *10th annual conference on Genetic and Evolutionary Computation*, pages 331–338, New York, NY, USA, 2008. ACM.

Appendix A

CA Continuous

This appendix lists a collection of spatio-temporal snapshots correspondent to CA Continuous. For this one-dimensional model, all the parameters, except ADD-CONSTANT, were fixed and a number of 11 groups comprising 5 snapshots each were generated. The groups are defined as the variable parameter is altered taking values [0.004, 0.1, 0.201, 0.301, 0.402, 0.502, 0.603, 0.703, 0.803, 0.9, 0.996]. The notation used to identify a particular snapshot is of the form xyz_pQ where xyz refers to the model itself, p to the group and Q to the pattern occurrence within that group.


| cac_a1 | cac_a2 | cac_a3 | cac_a4 | cac_a5 |

FIGURE A.1: *Group **a** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.004 and PRECISION-LEVEL = 16.*

cac_b1 cac_b2 cac_b3 cac_b4 cac_b5
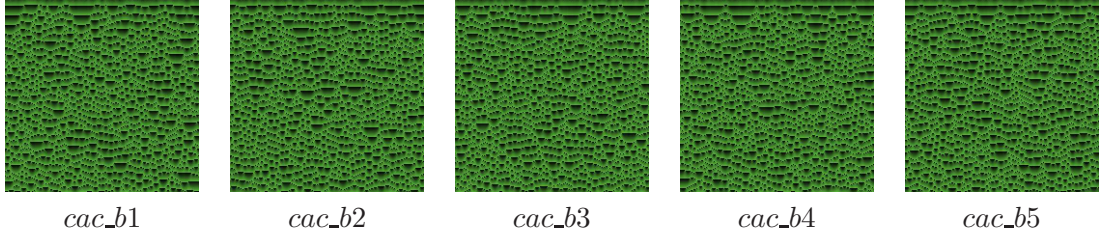
FIGURE A.2: *Group **b** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.1 and PRECISION-LEVEL = 16.*

cac_c1 cac_c2 cac_c3 cac_c4 cac_c5
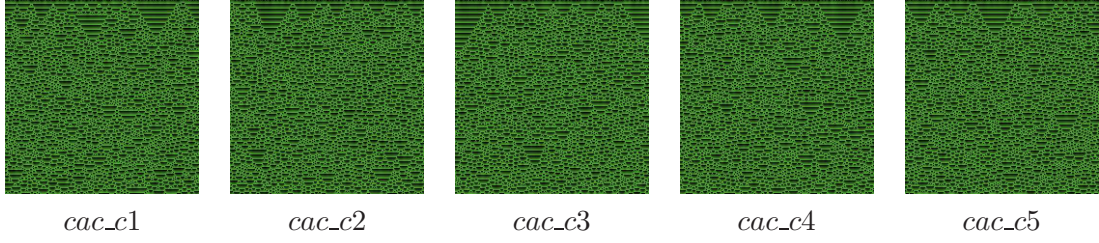
FIGURE A.3: *Group **c** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.201 and PRECISION-LEVEL = 16.*

cac_d1 cac_d2 cac_d3 cac_d4 cac_d5
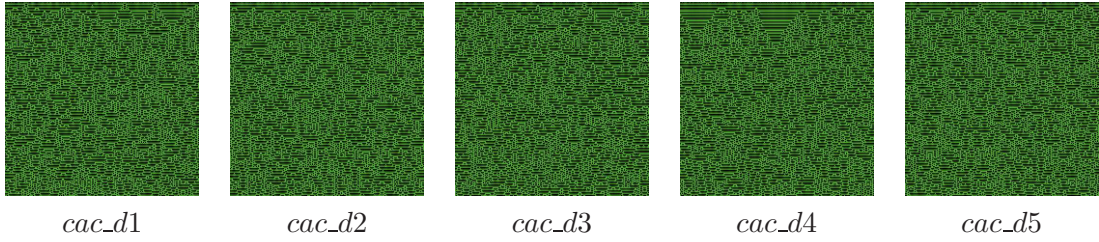
FIGURE A.4: *Group **d** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.301 and PRECISION-LEVEL = 16.*

cac_e1 cac_e2 cac_e3 cac_e4 cac_e5
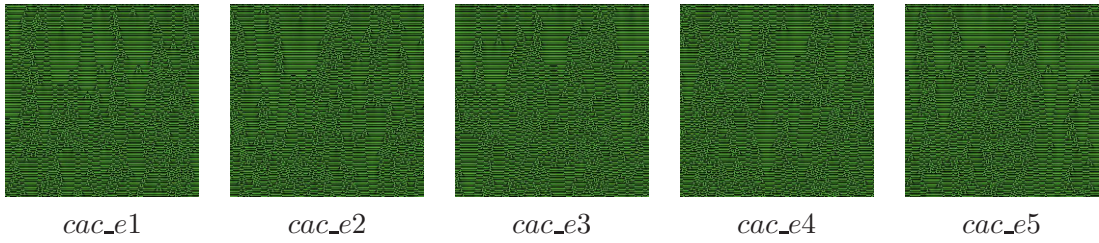
FIGURE A.5: *Group **e** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.402 and PRECISION-LEVEL = 16.*

cac_f1   cac_f2   cac_f3   cac_f4   cac_f5
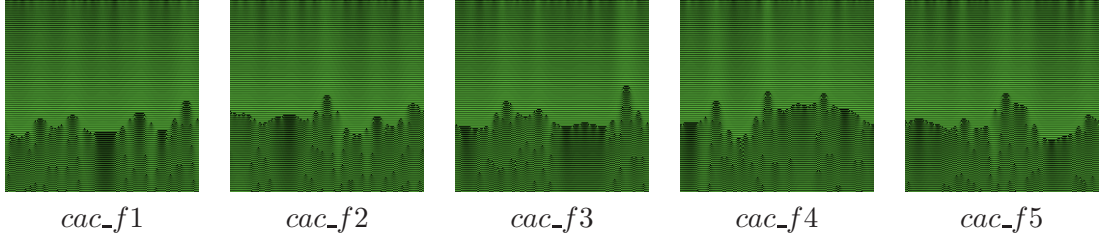
FIGURE A.6: *Group **f** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.502 and PRECISION-LEVEL = 16.*

cac_g1   cac_g2   cac_g3   cac_g4   cac_g5
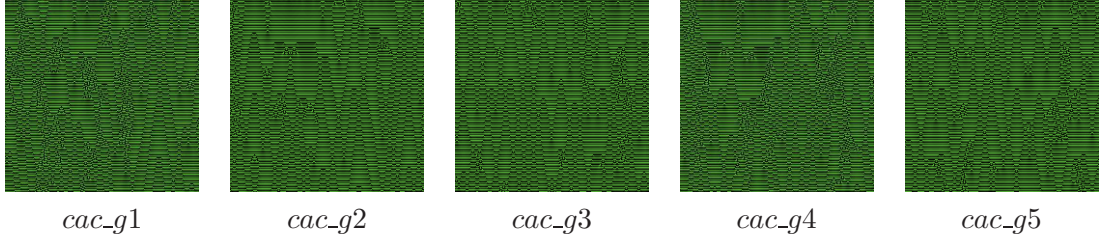
FIGURE A.7: *Group **g** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.603 and PRECISION-LEVEL = 16.*

cac_h1   cac_h2   cac_h3   cac_h4   cac_h5
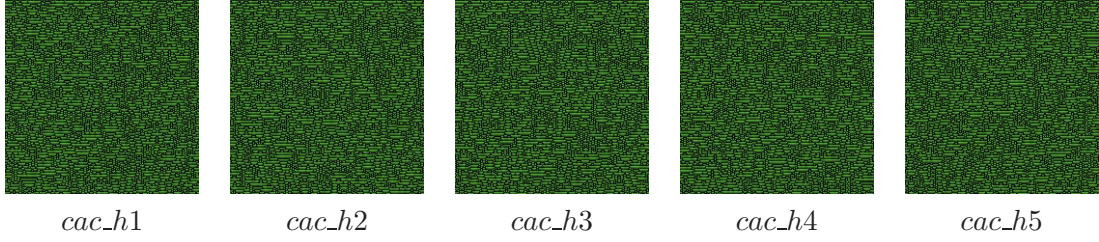
FIGURE A.8: *Group **h** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.703 and PRECISION-LEVEL = 16.*

cac_i1   cac_i2   cac_i3   cac_i4   cac_i5
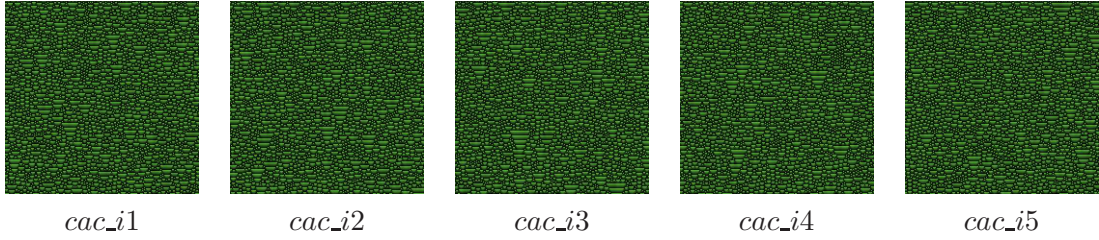
FIGURE A.9: *Group **i** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.803 and PRECISION-LEVEL = 16.*

cac_j1 cac_j2 cac_j3 cac_j4 cac_j5
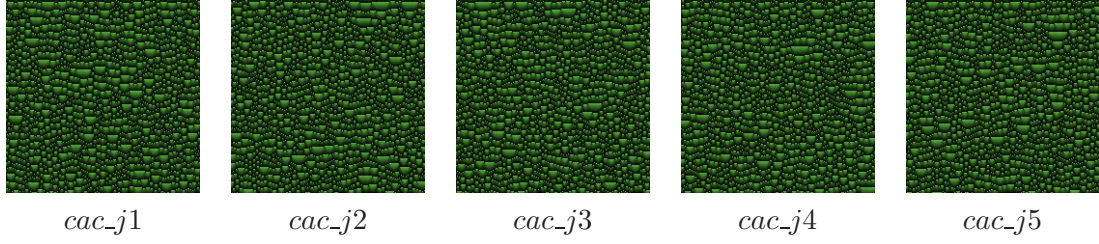
FIGURE A.10: *Group **j** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.9 and PRECISION-LEVEL = 16.*

cac_k1 cac_k2 cac_k3 cac_k4 cac_k5
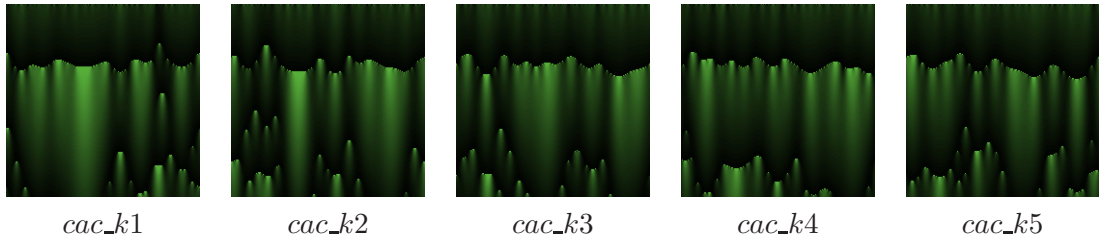
FIGURE A.11: *Group **k** of CA Continuous generated with random initialisation, ADD-CONSTANT = 0.996 and PRECISION-LEVEL = 16.*

Appendix B

Turbulence CA

This appendix lists a collection of spatio-temporal snapshots correspondent to Turbulence CA. For this one-dimensional model, all the parameters, except COUPLING-STRENGTH, were fixed and a number of 10 groups comprising 5 snapshots each were generated. The groups are defined as the variable parameter is altered taking values [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]. The notation used to identify a particular snapshot is of the form xyz_pQ where xyz refers to the model itself, p to the group and Q to the pattern occurrence within that group.

$turb\_a1$ $turb\_a2$ $turb\_a3$ $turb\_a4$ $turb\_a5$

FIGURE B.1: *Group $\boldsymbol{a}$ of Turbulence CA generated with random initialisation, COUPLING-STRENGTH $= 0.1$ and ROUGHNESS $= 0.001$.*

turb_b1   *turb_b2*   *turb_b3*   *turb_b4*   *turb_b5*

FIGURE B.2: *Group **b** of Turbulence CA generated with random initialisation, COUPLING-STRENGTH = 0.2 and ROUGHNESS = 0.001.*

turb_c1   *turb_c2*   *turb_c3*   *turb_c4*   *turb_c5*
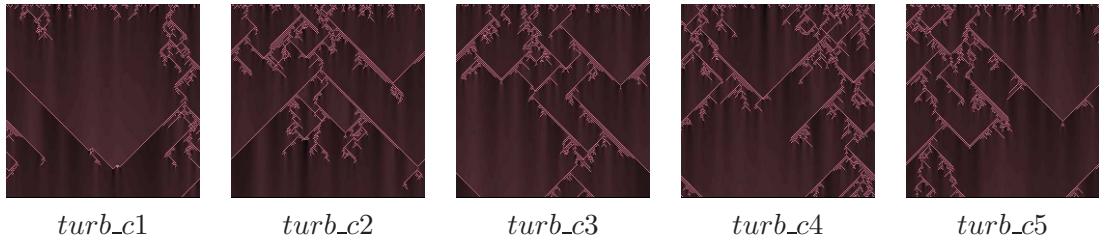
FIGURE B.3: *Group **c** of Turbulence CA generated with random initialisation, COUPLING-STRENGTH = 0.3 and ROUGHNESS = 0.001.*

turb_d1   *turb_d2*   *turb_d3*   *turb_d4*   *turb_d5*
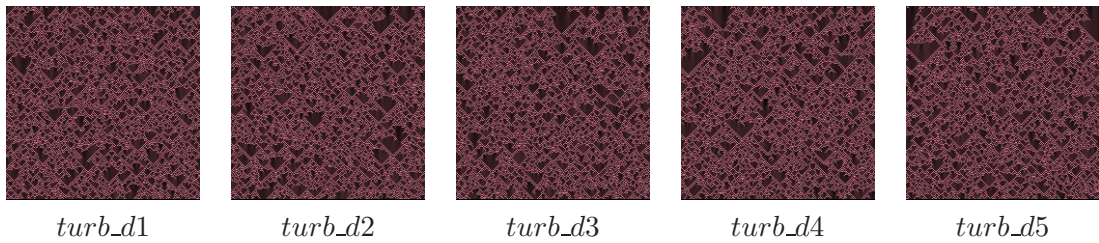
FIGURE B.4: *Group **d** of Turbulence CA generated with random initialisation, COUPLING-STRENGTH = 0.4 and ROUGHNESS = 0.001.*

turb_e1   *turb_e2*   *turb_e3*   *turb_e4*   *turb_e5*
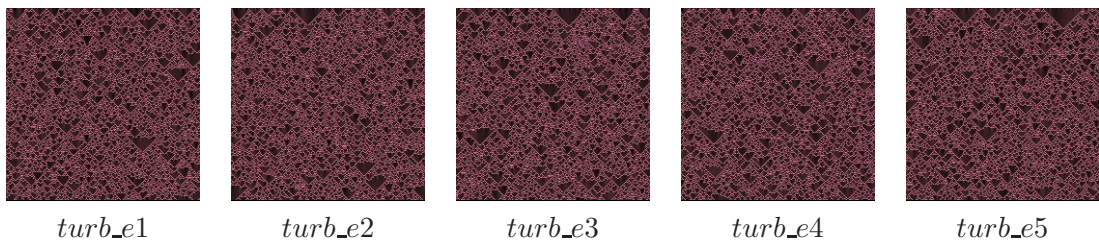
FIGURE B.5: *Group **e** of Turbulence CA generated with random initialisation, COUPLING-STRENGTH = 0.5 and ROUGHNESS = 0.001.*

| $turb\_f1$ | $turb\_f2$ | $turb\_f3$ | $turb\_f4$ | $turb\_f5$ |
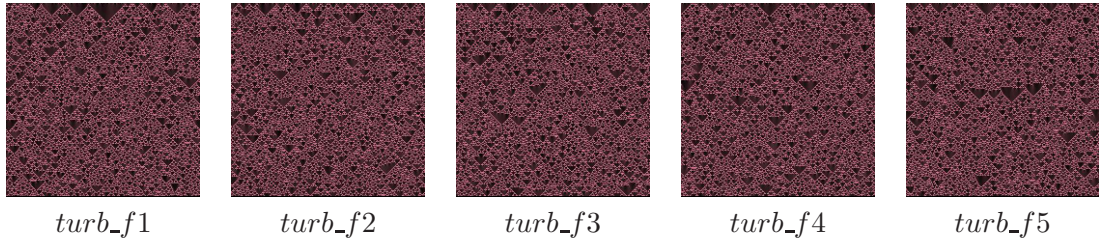
FIGURE B.6:   *Group* ***f*** *of Turbulence CA generated with random initialisation, COUPLING-STRENGTH = 0.6 and ROUGHNESS = 0.001.*

| $turb\_g1$ | $turb\_g2$ | $turb\_g3$ | $turb\_g4$ | $turb\_g5$ |
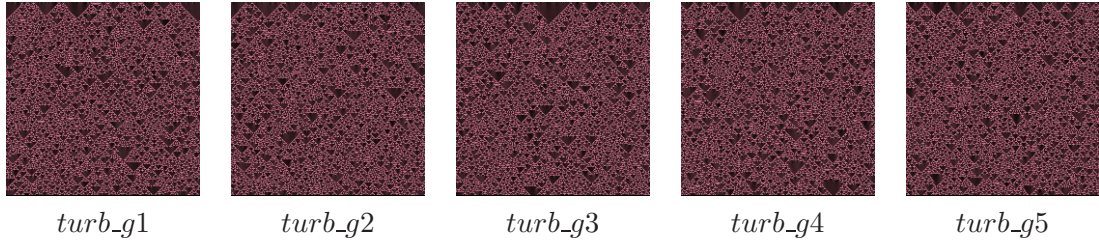
FIGURE B.7:   *Group* ***g*** *of Turbulence CA generated with random initialisation, COUPLING-STRENGTH = 0.7 and ROUGHNESS = 0.001.*

| $turb\_h1$ | $turb\_h2$ | $turb\_h3$ | $turb\_h4$ | $turb\_h5$ |
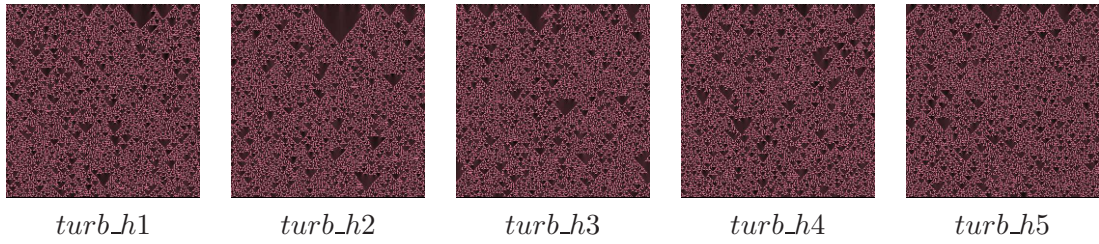
FIGURE B.8:   *Group* ***h*** *of Turbulence CA generated with random initialisation, COUPLING-STRENGTH = 0.8 and ROUGHNESS = 0.001.*

| $turb\_i1$ | $turb\_i2$ | $turb\_i3$ | $turb\_i4$ | $turb\_i5$ |

FIGURE B.9:   *Group* ***i*** *of Turbulence CA generated with random initialisation, COUPLING-STRENGTH = 0.9 and ROUGHNESS = 0.001.*

| $turb\_j1$ | $turb\_j2$ | $turb\_j3$ | $turb\_j4$ | $turb\_j5$ |

FIGURE B.10:   *Group $\boldsymbol{j}$ of Turbulence CA generated with random initialisation, COUPLING-STRENGTH = 1.0 and ROUGHNESS = 0.001.*

Appendix C

Gas Lattice CA

This appendix lists a collection of spatio-temporal snapshots correspondent to Gas Lattice. For this two-dimensional model, all the parameters, except RADIUS, were fixed and a number of 12 groups comprising 5 snapshots each were generated. The groups are defined as the variable parameter is altered taking values [1, 10, 20, 30, 40, 50, 60, 70, 70, 80, 90, 100]. The notation used to identify a particular snapshot is of the form xyz_pQ where xyz refers to the model itself, p to the group and Q to the pattern occurrence within that group.

gas_a1 gas_a2 gas_a3 gas_a4 gas_a5
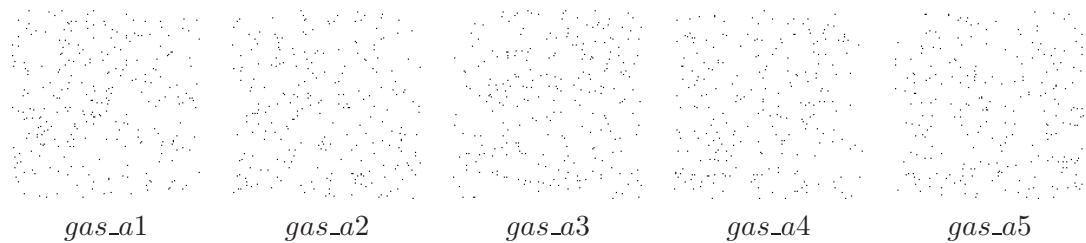
FIGURE C.1: *Group **a** of Gas Lattice CA generated with random initialisation, RADIUS = 1 and DENSITY = 0.*
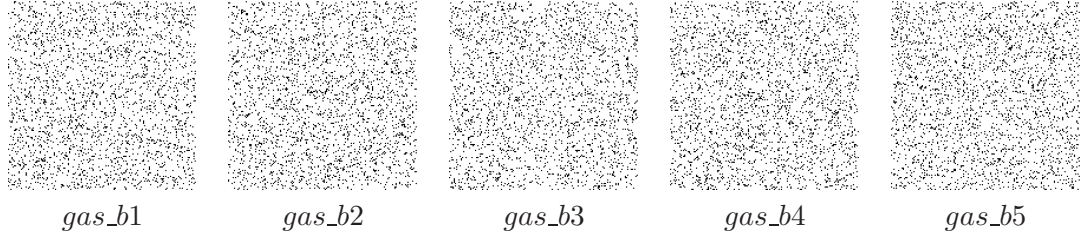
FIGURE C.2: *Group **b** of Gas Lattice CA generated with random initialisation, RADIUS = 10 and DENSITY = 0.*
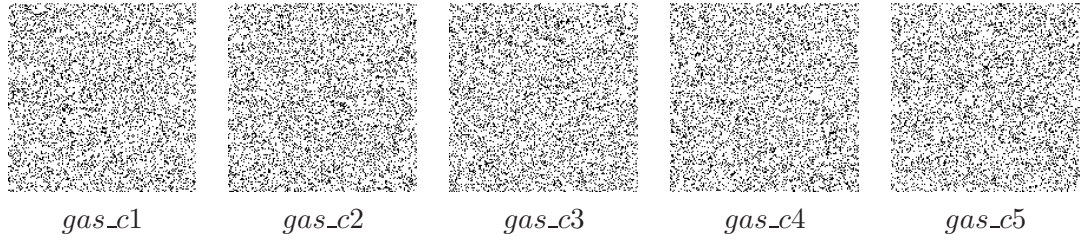
FIGURE C.3: *Group **c** of Gas Lattice CA generated with random initialisation, RADIUS = 20 and DENSITY = 0.*
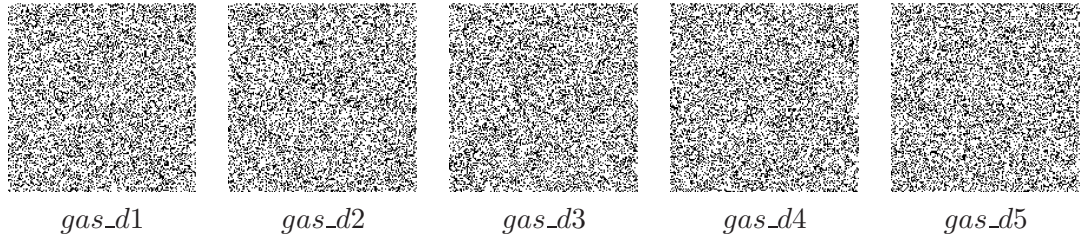
FIGURE C.4: *Group **d** of Gas Lattice CA generated with random initialisation, RADIUS = 30 and DENSITY = 0.*
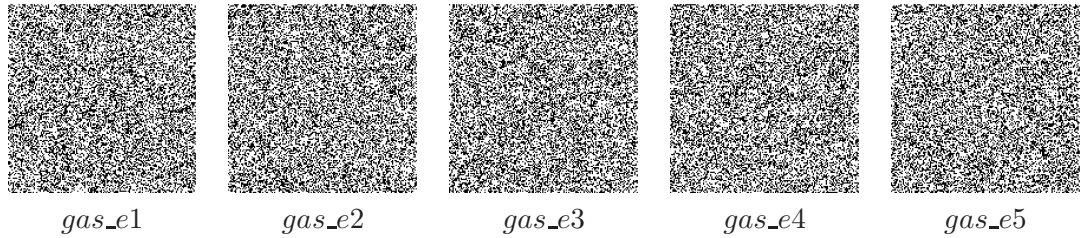
FIGURE C.5: *Group **e** of Gas Lattice CA generated with random initialisation, RADIUS = 40 and DENSITY = 0.*

gas_f1 $\quad\quad$ gas_f2 $\quad\quad$ gas_f3 $\quad\quad$ gas_f4 $\quad\quad$ gas_f5
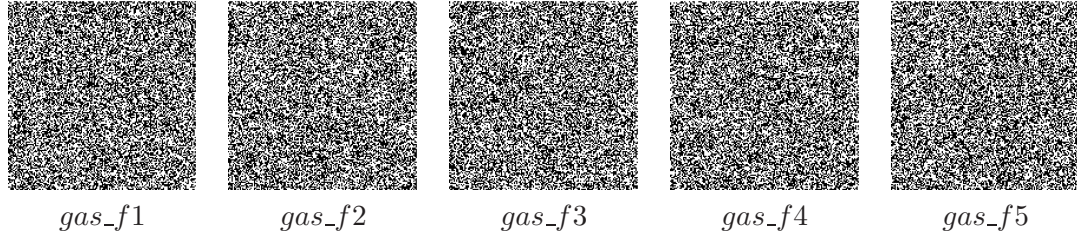
FIGURE C.6: *Group **f** of Gas Lattice CA generated with random initialisation, RADIUS = 50 and DENSITY = 0.*

gas_g1 $\quad\quad$ gas_g2 $\quad\quad$ gas_g3 $\quad\quad$ gas_g4 $\quad\quad$ gas_g5
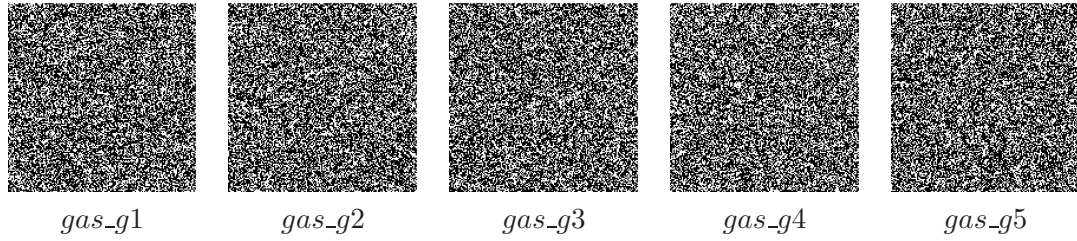
FIGURE C.7: *Group **g** of Gas Lattice CA generated with random initialisation, RADIUS = 60 and DENSITY = 0.*

gas_h1 $\quad\quad$ gas_h2 $\quad\quad$ gas_h3 $\quad\quad$ gas_h4 $\quad\quad$ gas_h5
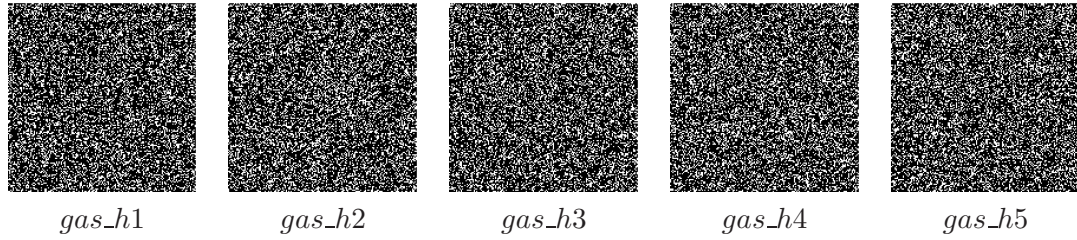
FIGURE C.8: *Group **h** of Gas Lattice CA generated with random initialisation, RADIUS = 70 and DENSITY = 0.*

gas_i1 $\quad\quad$ gas_i2 $\quad\quad$ gas_i3 $\quad\quad$ gas_i4 $\quad\quad$ gas_i5
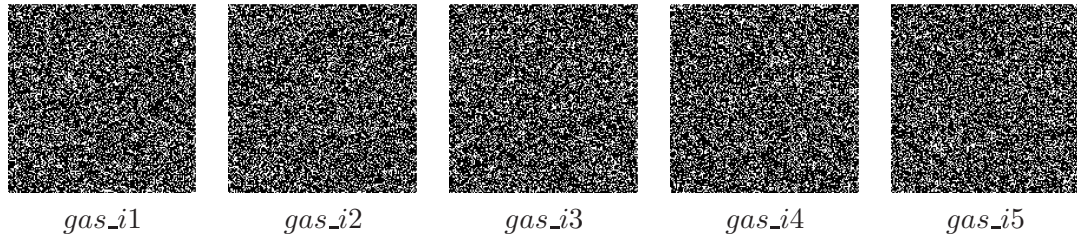
FIGURE C.9: *Group **i** of Gas Lattice CA generated with random initialisation, RADIUS = 70 and DENSITY = 0.*

gas_j1 gas_j2 gas_j3 gas_j4 gas_j5
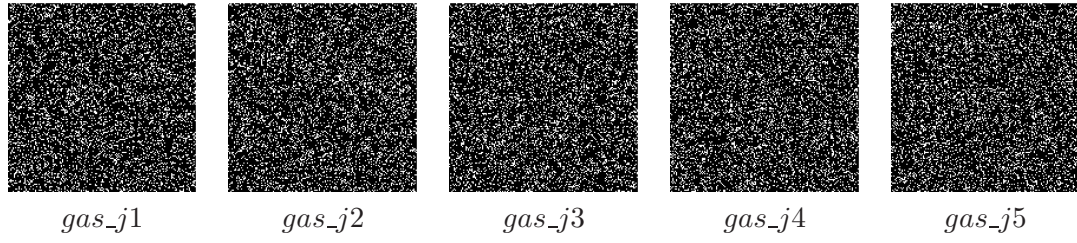
FIGURE C.10: *Group* **j** *of Gas Lattice CA generated with random initialisation, RADIUS = 80 and DENSITY = 0.*

gas_k1 gas_k2 gas_k3 gas_k4 gas_k5
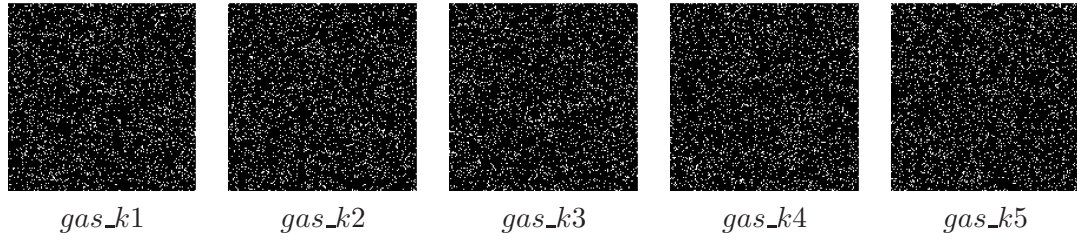
FIGURE C.11: *Group* **k** *of Gas Lattice CA generated with random initialisation, RADIUS = 90 and DENSITY = 0.*

gas_l1 gas_l2 gas_l3 gas_l4 gas_l5
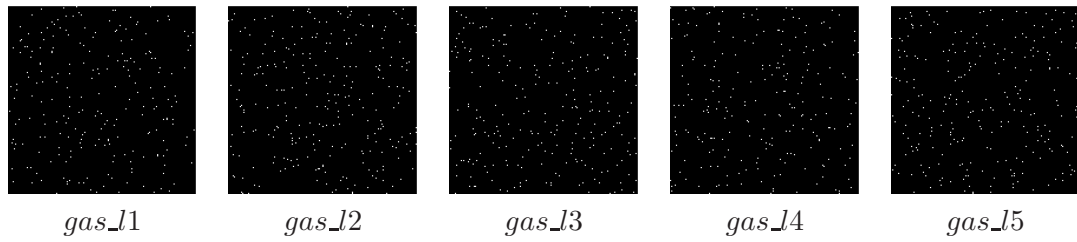
FIGURE C.12: *Group* **l** *of Gas Lattice CA generated with random initialisation, RADIUS = 100 and DENSITY = 0.*

APPENDIX D

Meta-automaton CA

This appendix lists a collection of spatio-temporal snapshots correspondent to Meta-automaton CA. For this one-dimensional model, all the parameters, except RULES, were fixed and a number of 10 groups comprising 5 snapshots each were generated. The groups are defined as the variable parameter is altered taking values [122, 148, 181, 120, 97, 135, 229, 131, 154, 133]. The notation used to identify a particular snapshot is of the form xyz_pQ where xyz refers to the model itself, p to the group and Q to the pattern occurrence within that group.

$metaK1\_a1$ $metaK1\_a2$ $metaK1\_a3$ $metaK1\_a4$ $metaK1\_a5$
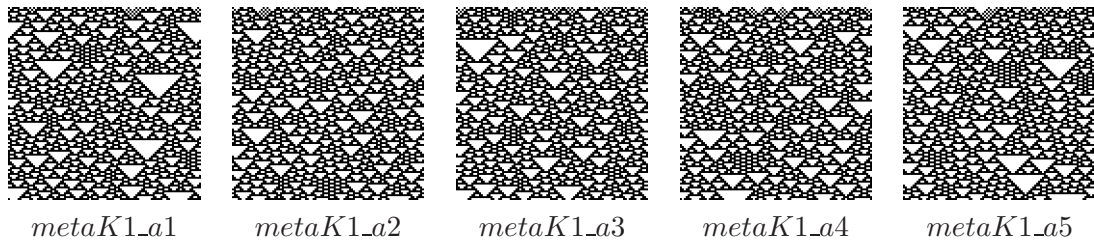
FIGURE D.1: *Group **a** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 122.*

metaK1_b1   metaK1_b2   metaK1_b3   metaK1_b4   metaK1_b5
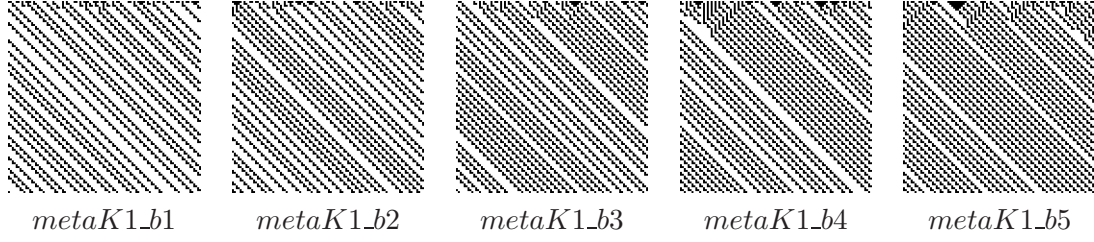
FIGURE D.2: *Group **b** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 148.*

metaK1_c1   metaK1_c2   metaK1_c3   metaK1_c4   metaK1_c5
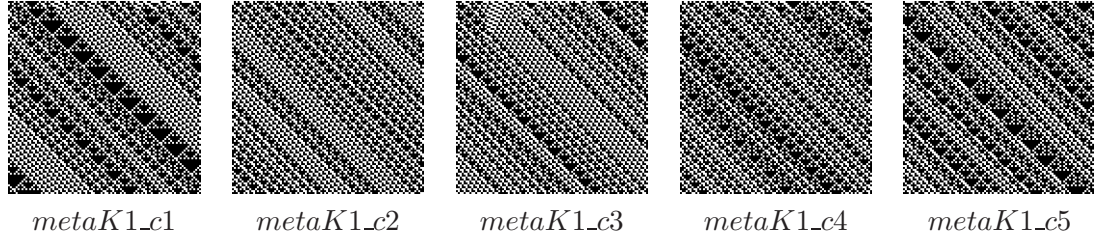
FIGURE D.3: *Group **c** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 181.*

metaK1_d1   metaK1_d2   metaK1_d3   metaK1_d4   metaK1_d5
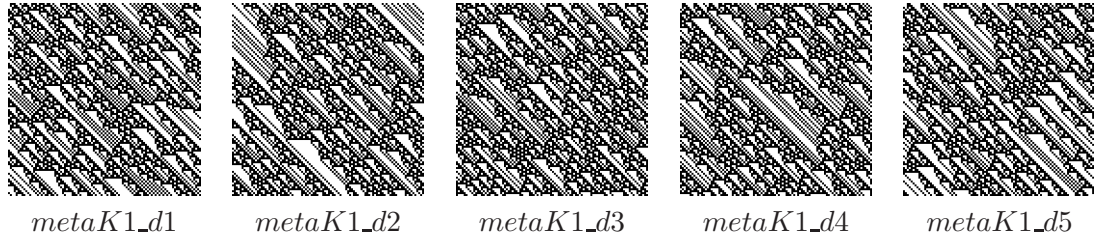
FIGURE D.4: *Group **d** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 120.*

metaK1_e1   metaK1_e2   metaK1_e3   metaK1_e4   metaK1_e5
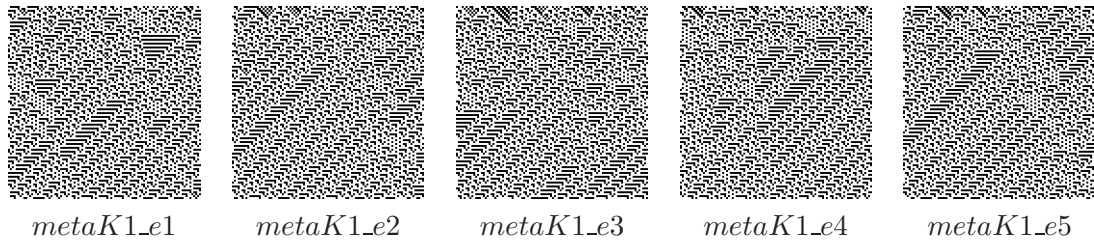
FIGURE D.5: *Group **e** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 97.*

$metaK1\_f1 \qquad metaK1\_f2 \qquad metaK1\_f3 \qquad metaK1\_f4 \qquad metaK1\_f5$
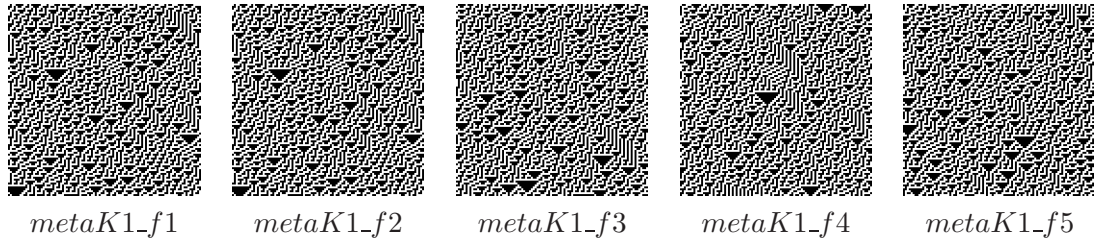
FIGURE D.6: *Group **f** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 135.*

$metaK1\_g1 \qquad metaK1\_g2 \qquad metaK1\_g3 \qquad metaK1\_g4 \qquad metaK1\_g5$
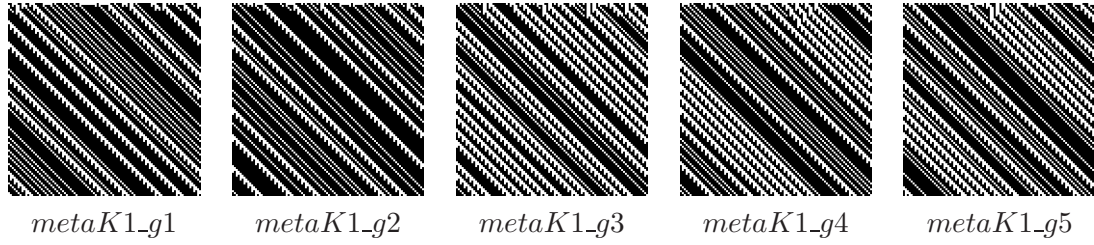
FIGURE D.7: *Group **g** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 229.*

$metaK1\_h1 \qquad metaK1\_h2 \qquad metaK1\_h3 \qquad metaK1\_h4 \qquad metaK1\_h5$
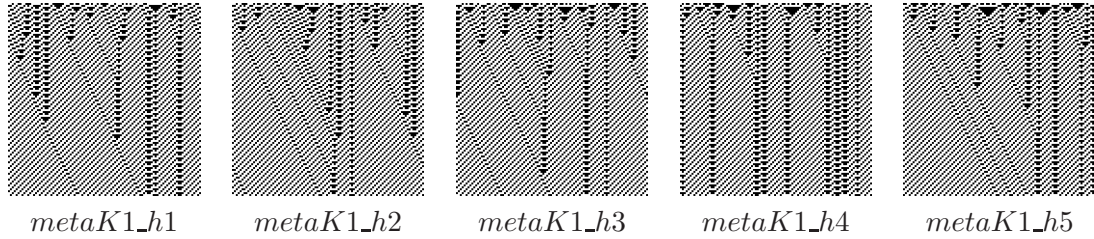
FIGURE D.8: *Group **h** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 131.*

$metaK1\_i1 \qquad metaK1\_i2 \qquad metaK1\_i3 \qquad metaK1\_i4 \qquad metaK1\_i5$
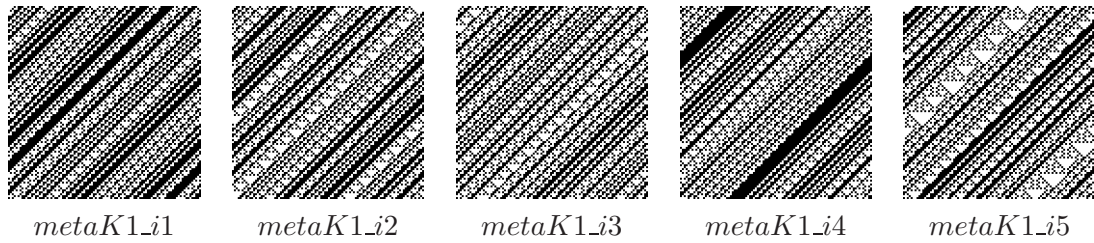
FIGURE D.9: *Group **i** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 154.*

metaK1_j1   metaK1_j2   metaK1_j3   metaK1_j4   metaK1_j5
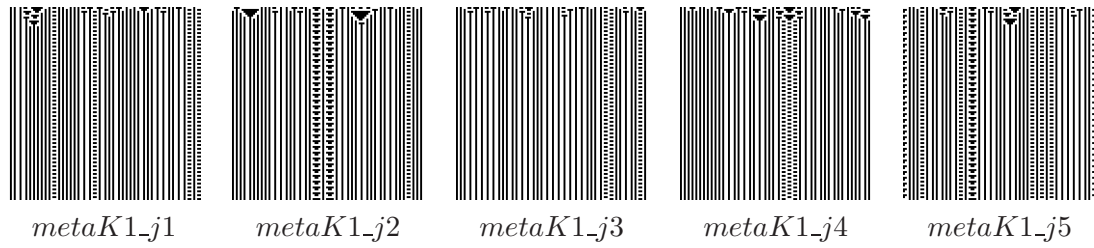
FIGURE D.10: *Group **j** of Meta-automaton CA generated with random initialisation, K-TIMES = 100 and RULES= 133.*

Appendix E

CA Continuous Scatter Plots

This appendix lists a collection of scatter plots correspondent to the fitness distance correlation experiments performed over CA Continuous in Section 8.2.1 of Chapter 8. In turns, each of the CA snapshots was considered as a target (T) to which the remaining snapshots of all the groups (T_i) were evaluated on fitness (f_i) using the USM and on distance (d_i) using Euclidean difference among the values of their associated creational parameters. Equation E.1 shows the calculation of FDC where n is the number of samples, $\overline{f}$ and S_F are the mean and standard deviation of the fitness values, and $\overline{d}$ and S_D are the mean and standard deviation of the distances.

$$f_i = f(T_i) = USM(T_i, T)$$

$$d_i = (\sum_{k=1}^{n}(par_k^i - par_k^T)^2)^{1/2}$$

$$FDC = \frac{(1/n)\sum_{i=1}^{n}(f_i - \overline{f})(d_i - \overline{d})}{S_F S_D}$$

$$T \text{ is a target snapshot}$$

$$par_k^i \text{ is the creational } k\text{-parameter} \tag{E.1}$$
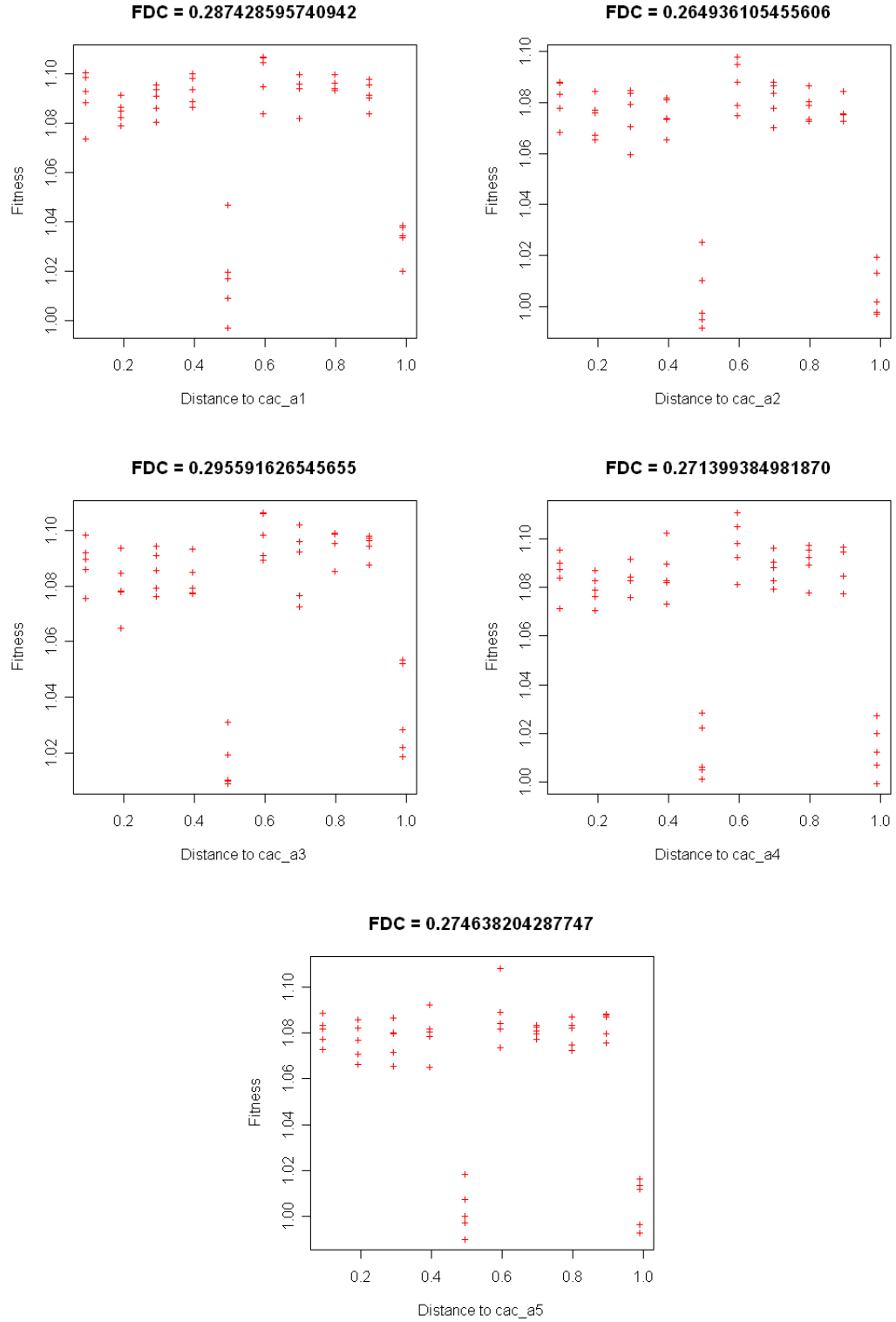
FIGURE E.1: *Graphics of the resultant scatter plots and correlation coefficients for the group **a** of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
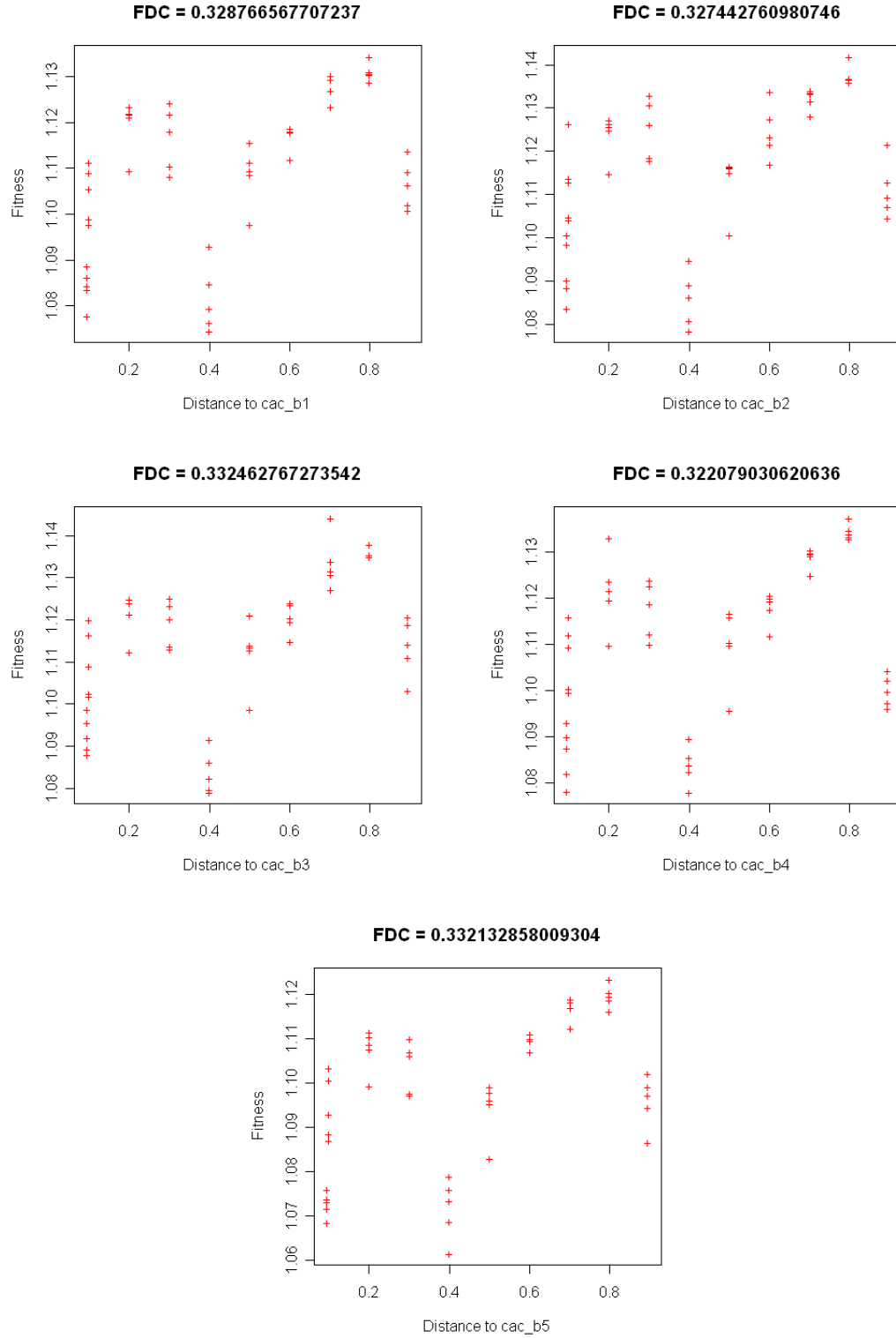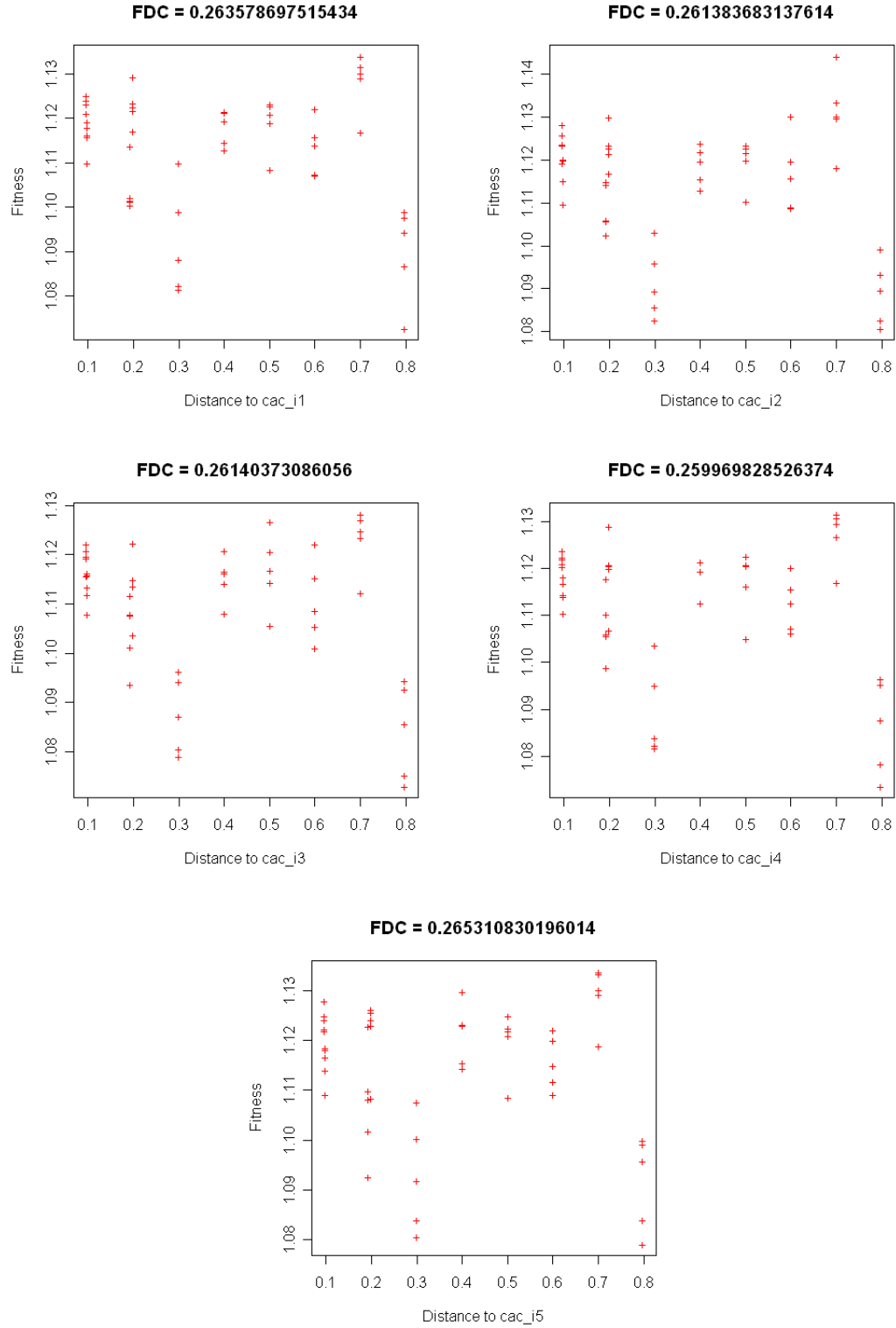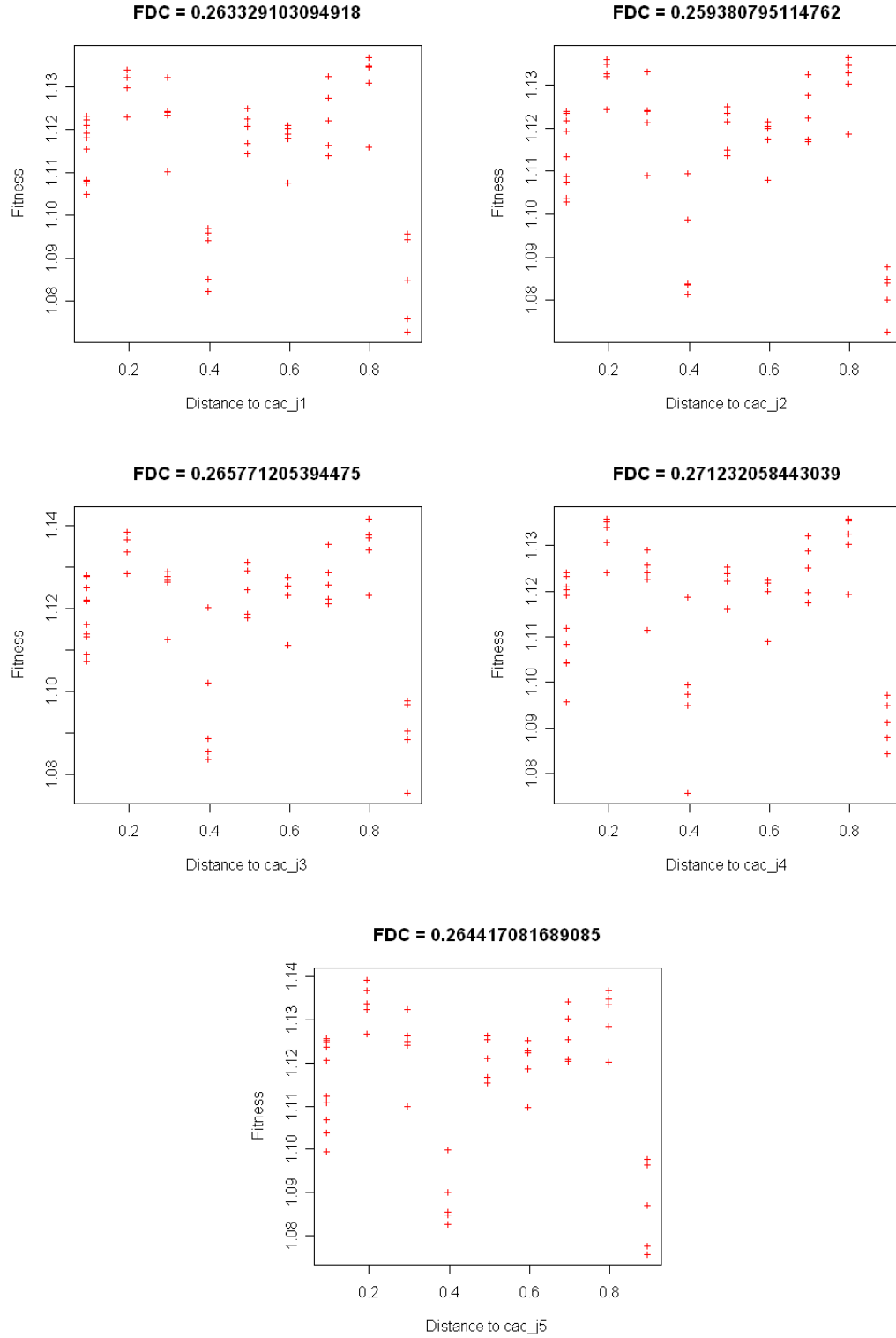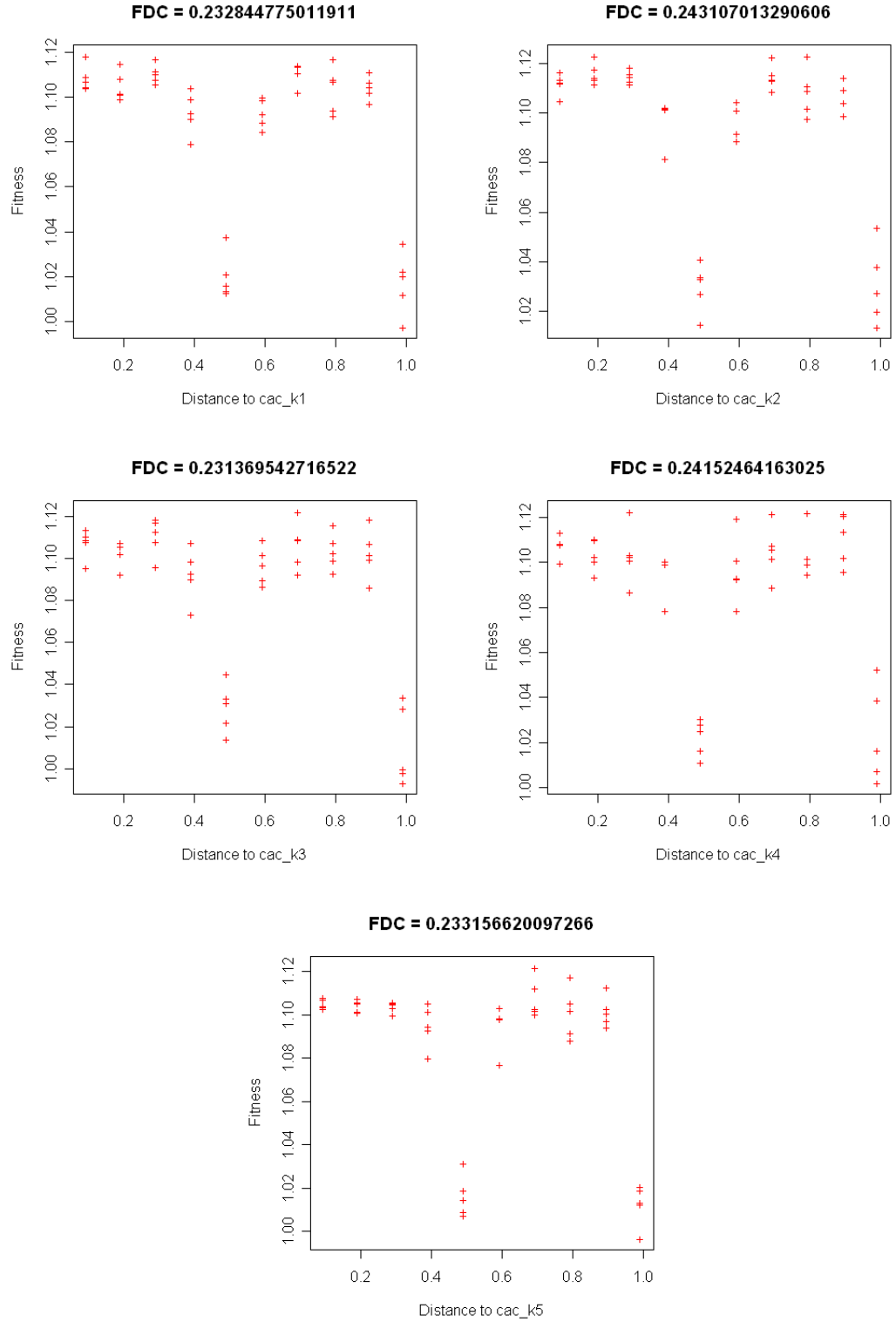
FIGURE E.2: *Graphics of the resultant scatter plots and correlation coefficients for the group **b** of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
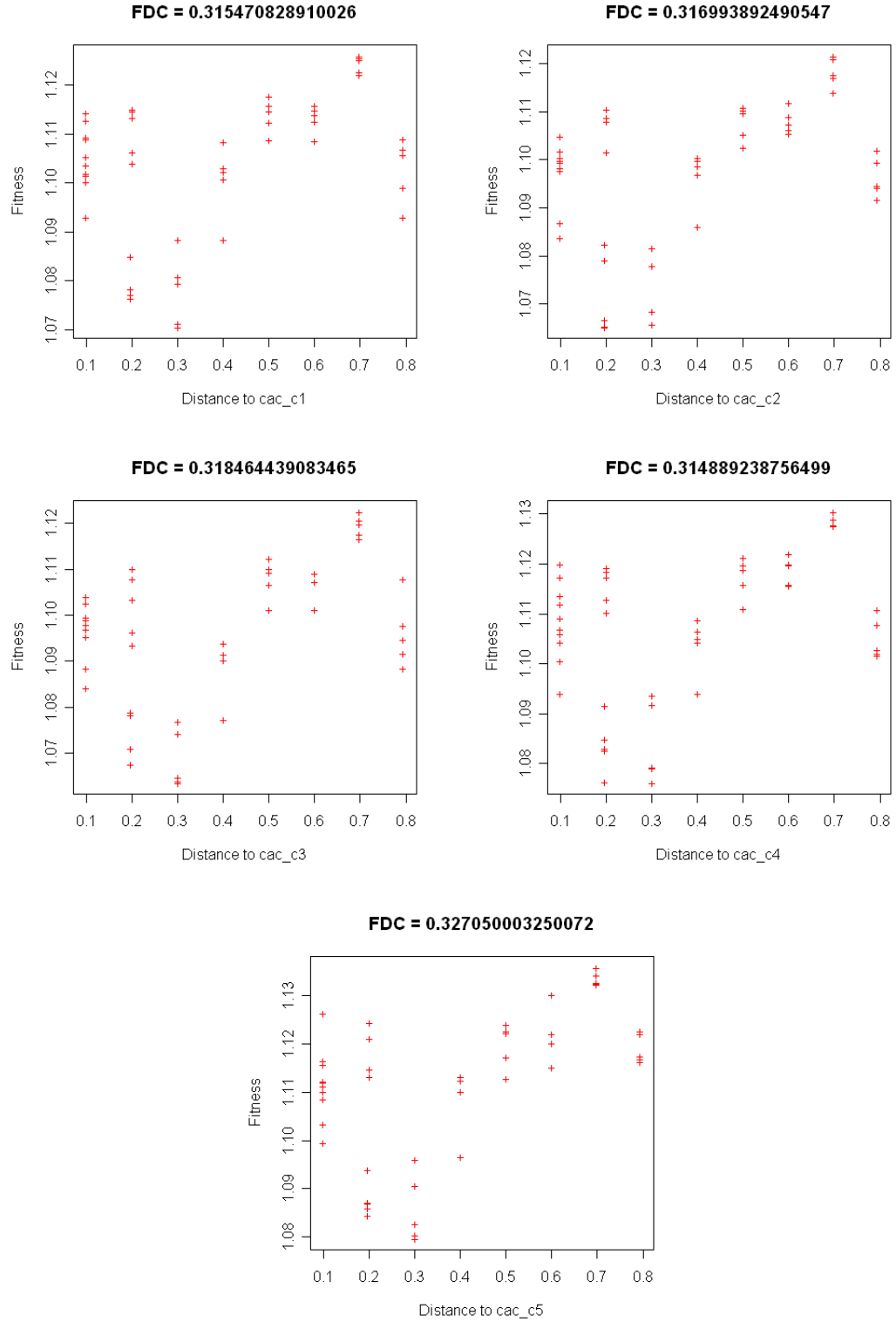
FIGURE E.3: *Graphics of the resultant scatter plots and correlation coefficients for the group **c** of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
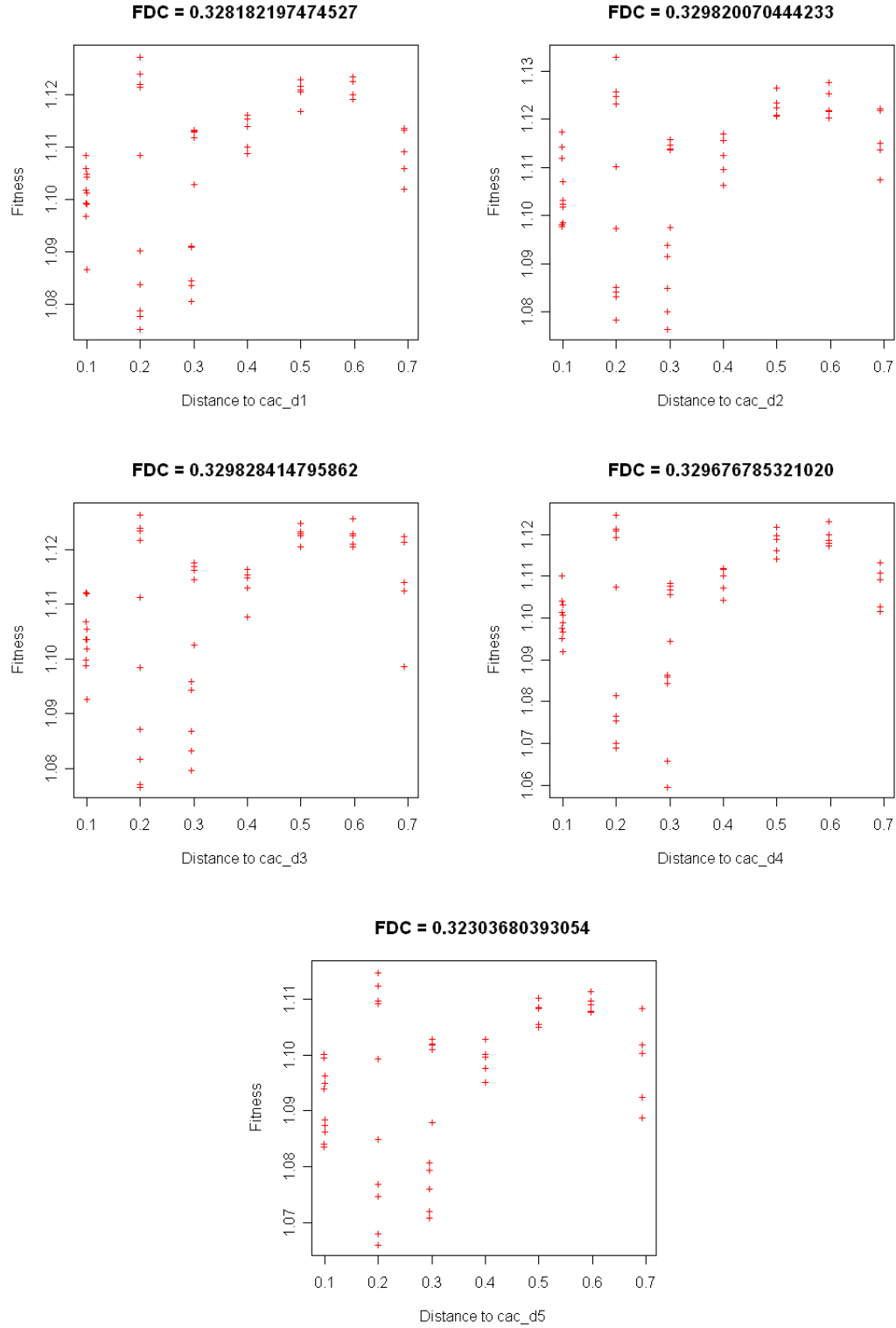
FIGURE E.4: *Graphics of the resultant scatter plots and correlation coefficients for the group* ***d*** *of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
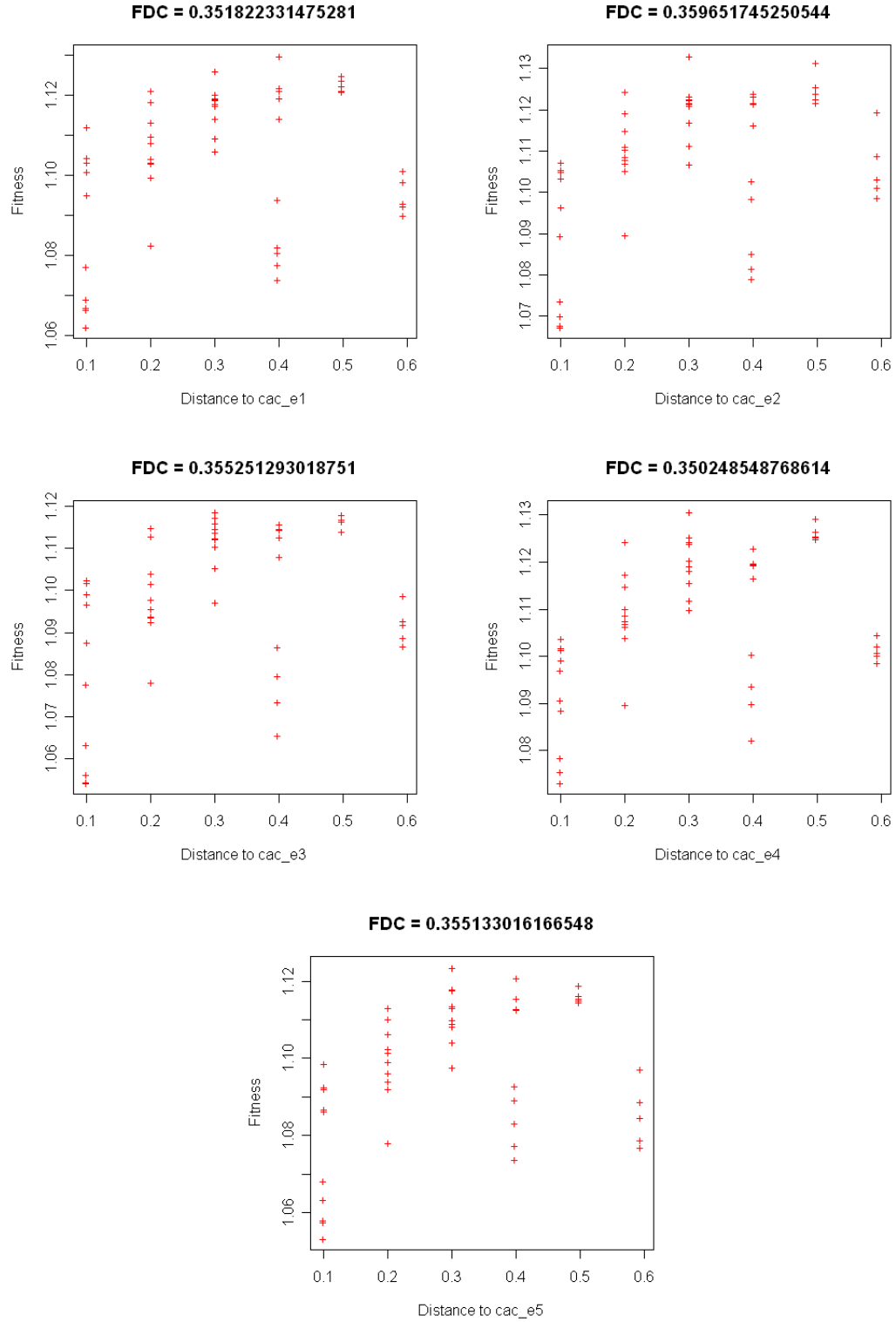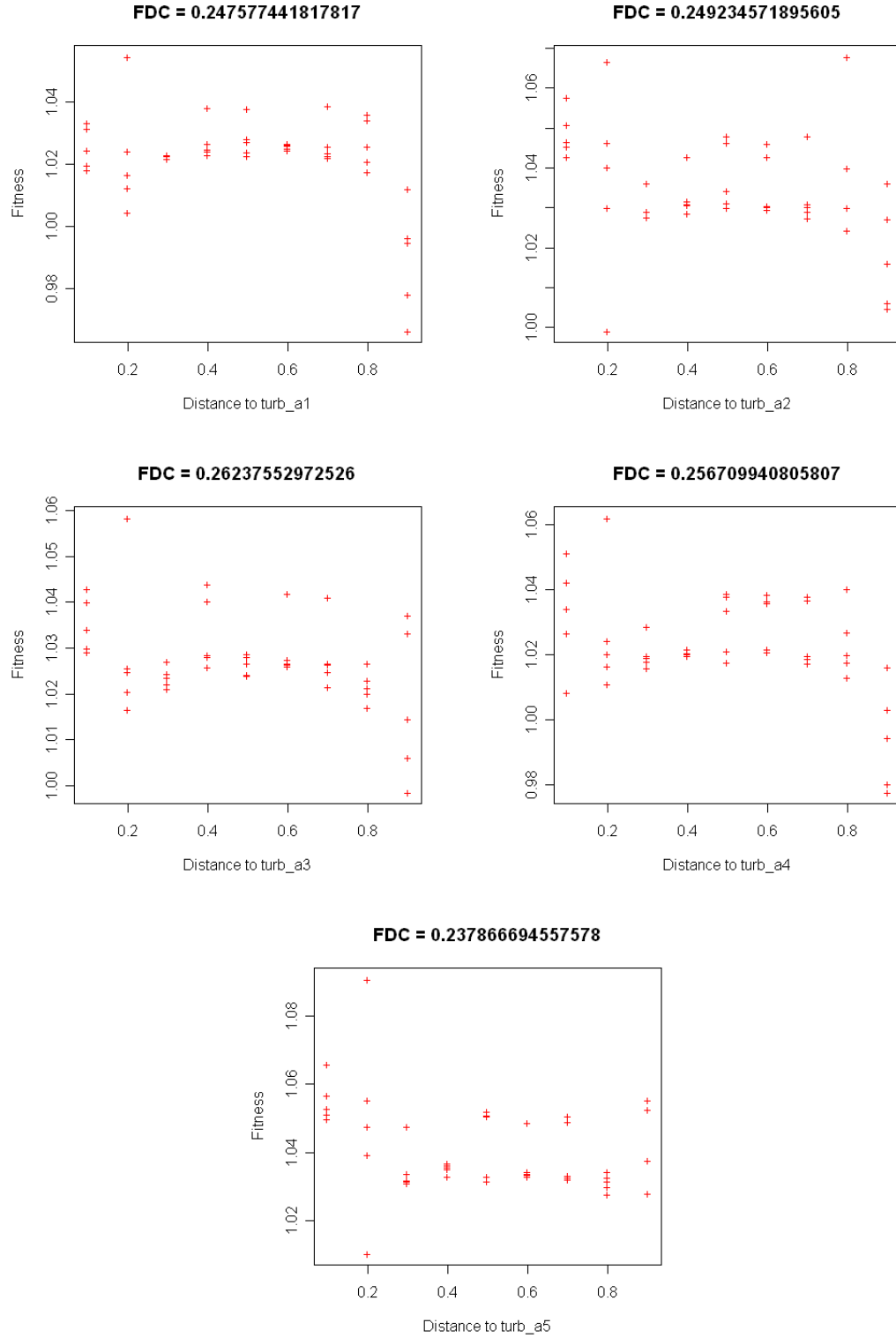
FIGURE E.5: *Graphics of the resultant scatter plots and correlation coefficients for the group **e** of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
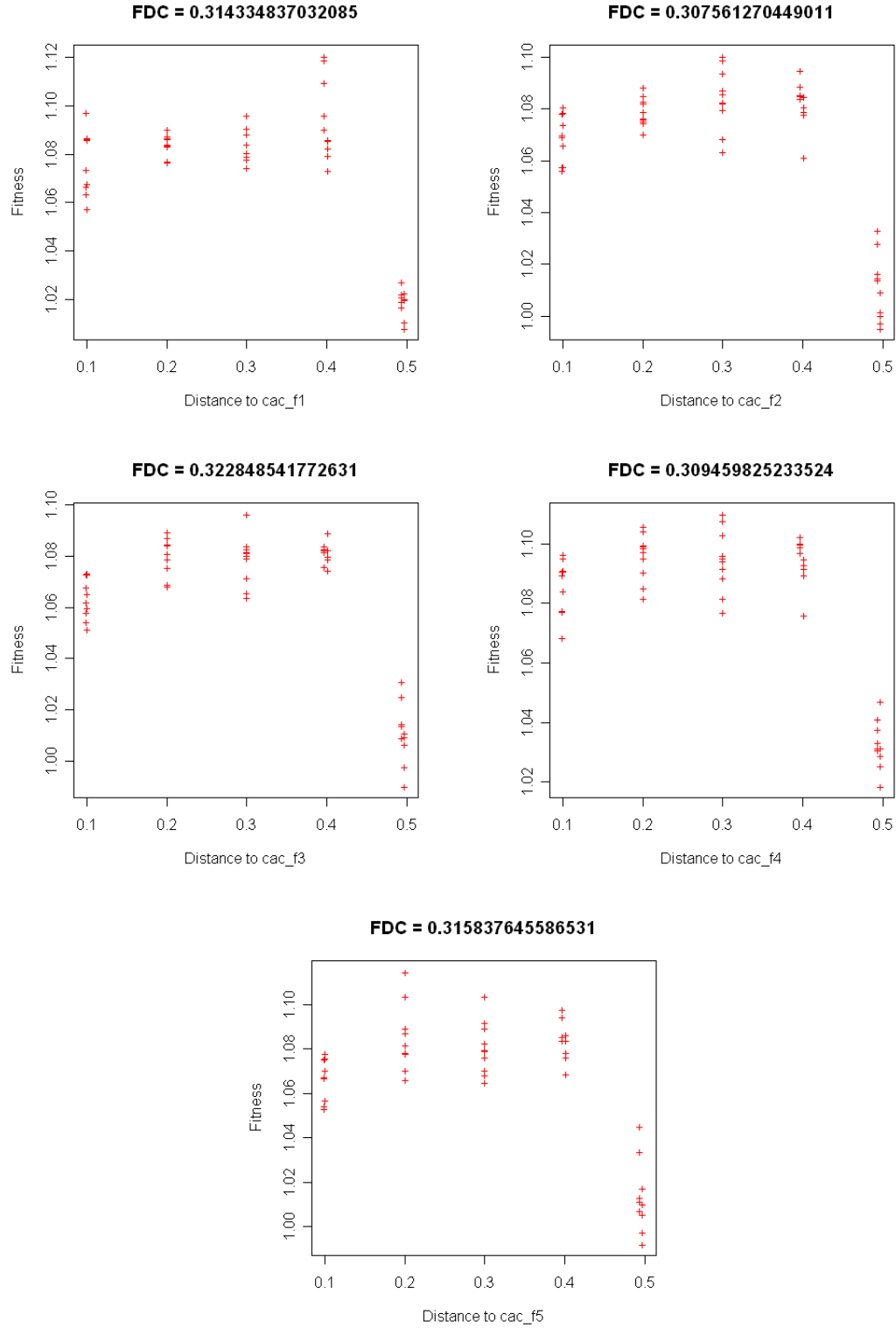
FIGURE E.6: *Graphics of the resultant scatter plots and correlation coefficients for the group **f** of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
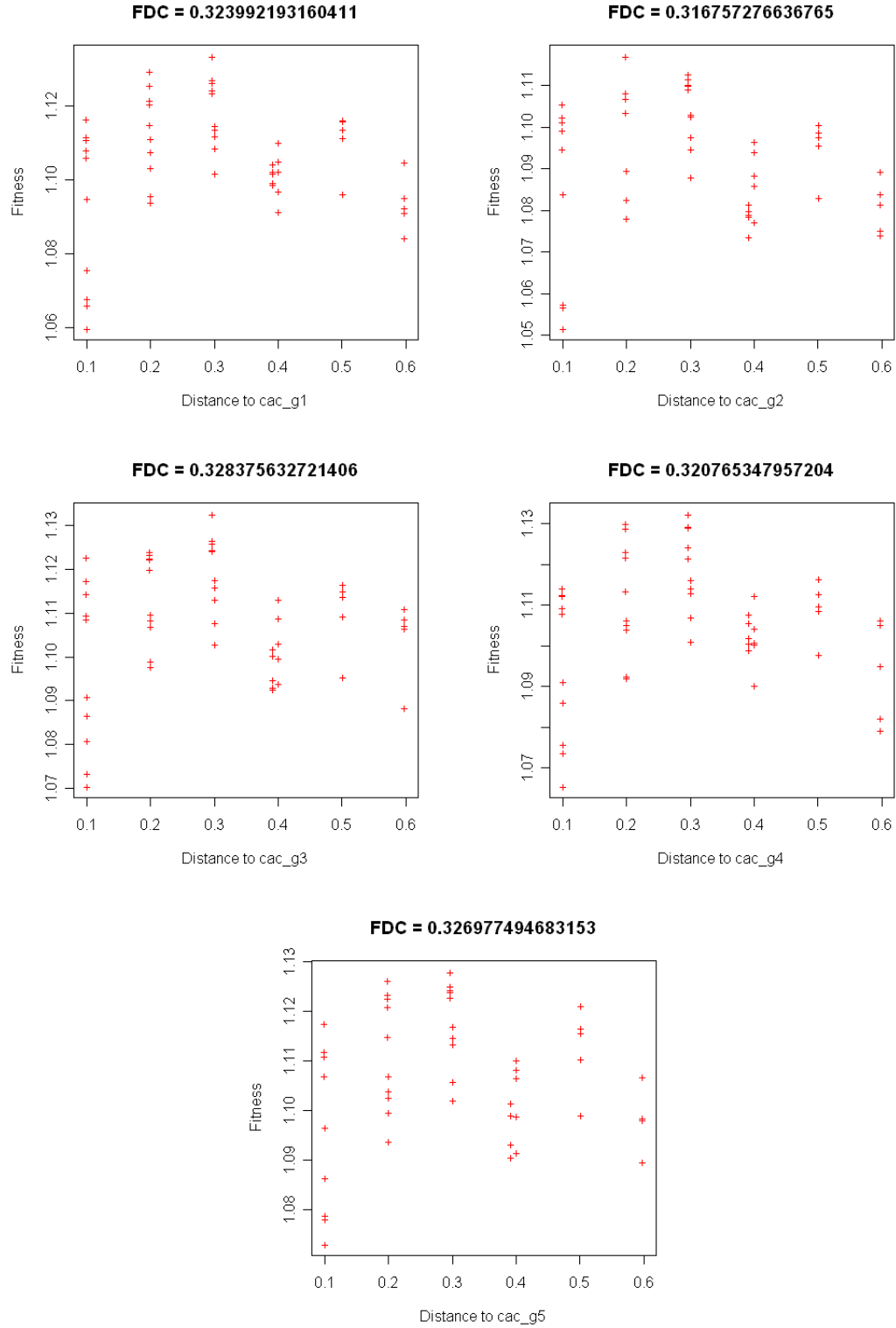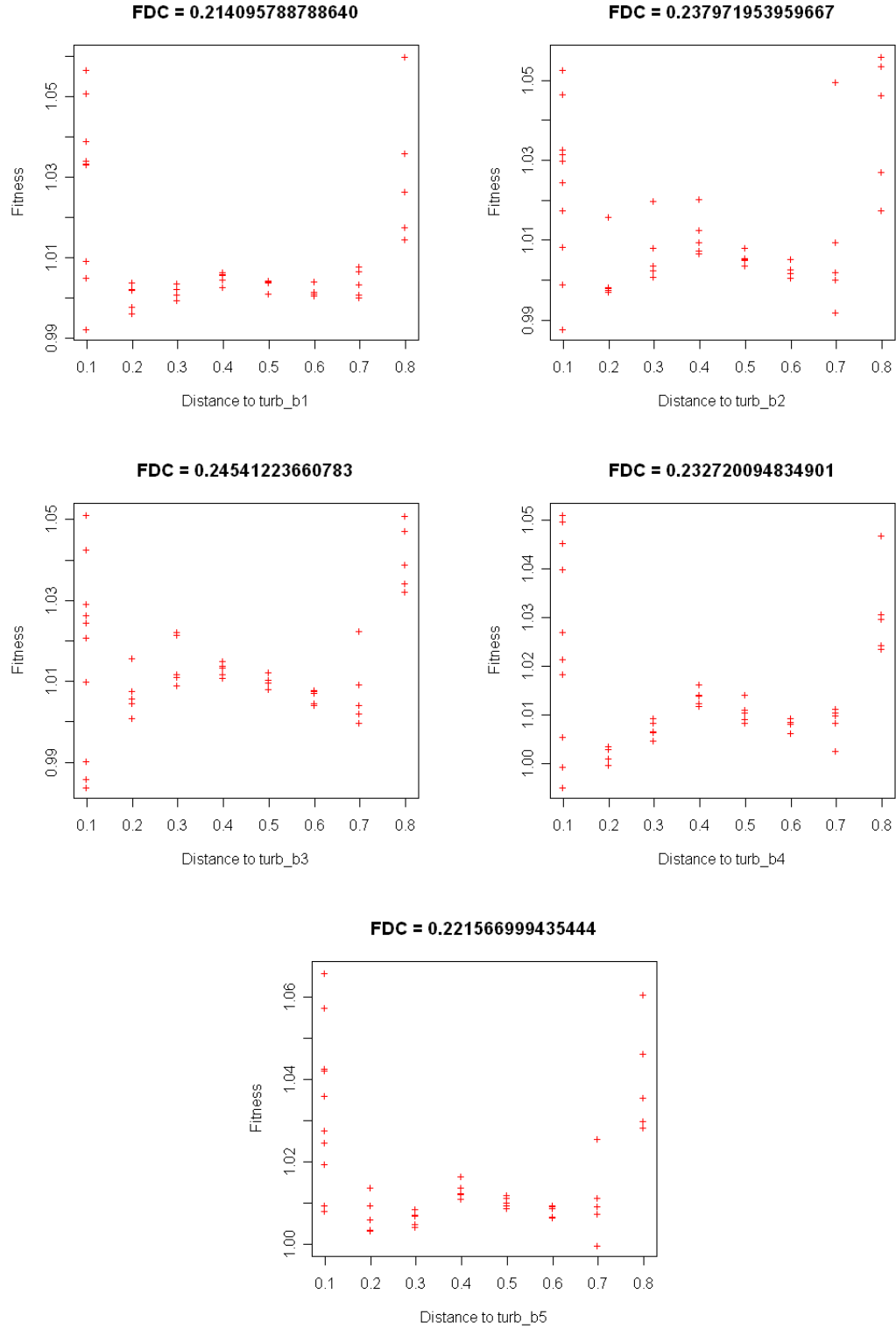
FIGURE E.7: *Graphics of the resultant scatter plots and correlation coefficients for the group **g** of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
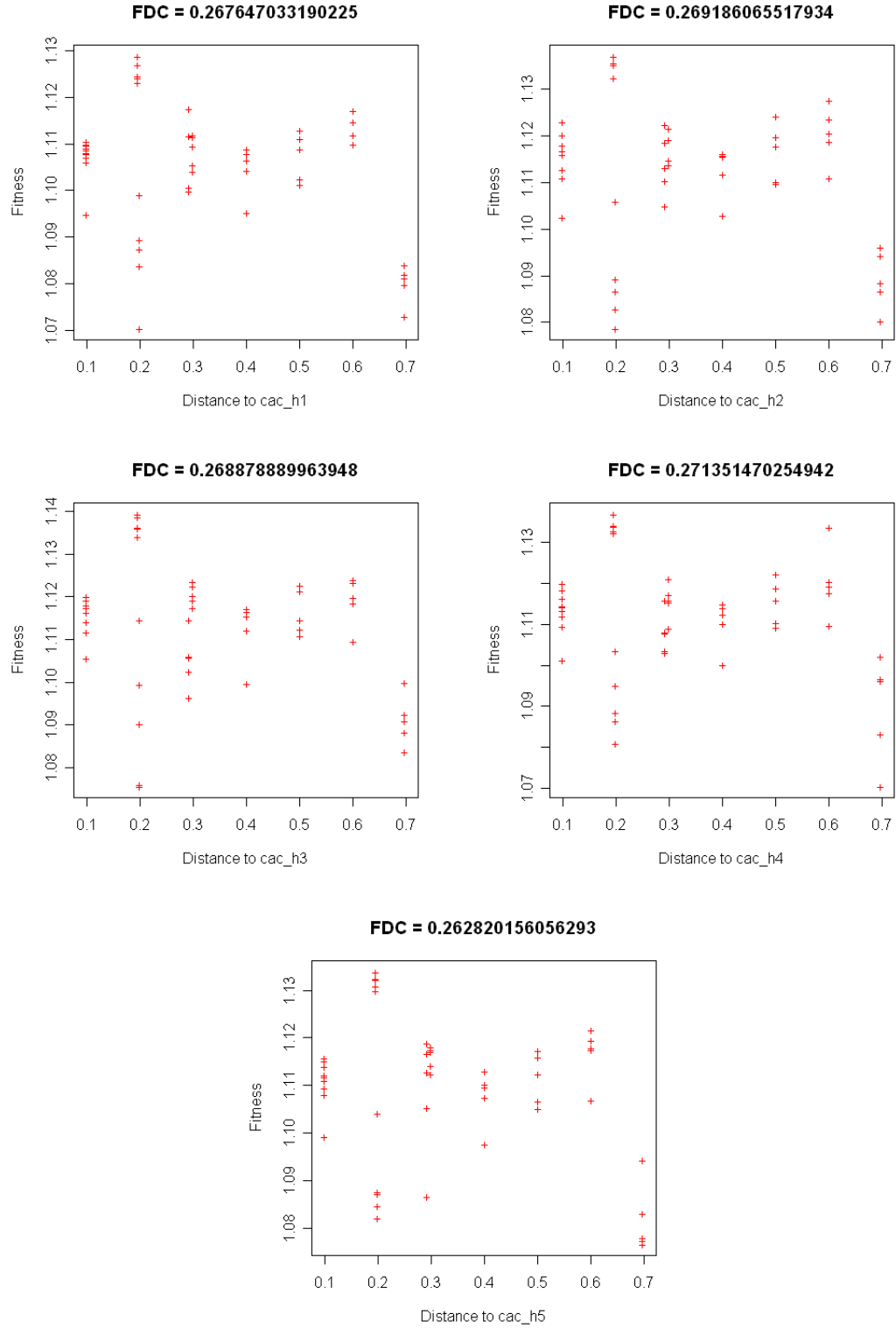
FIGURE E.8: *Graphics of the resultant scatter plots and correlation coefficients for the group **h** of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
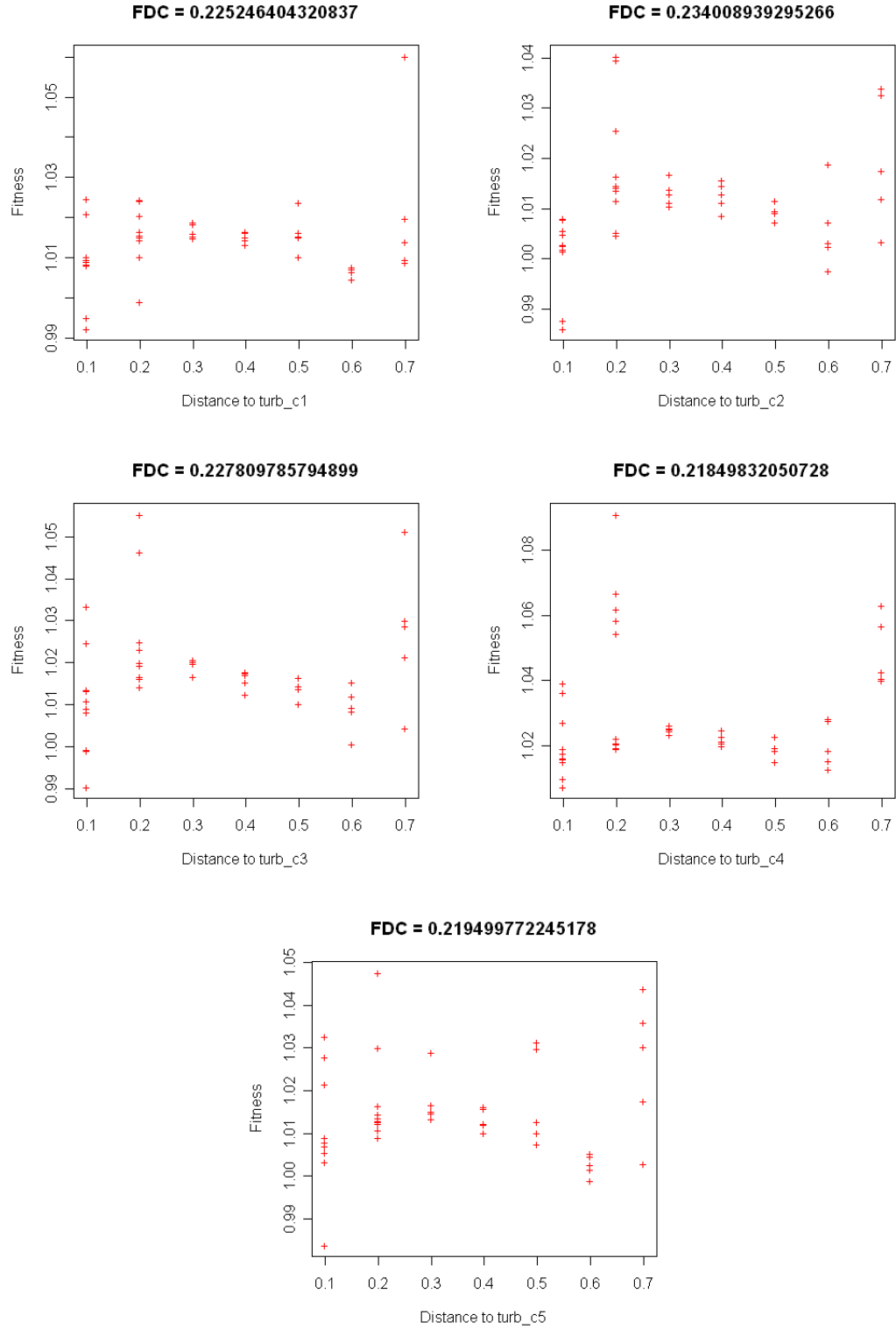
FIGURE E.9: *Graphics of the resultant scatter plots and correlation coefficients for the group i of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
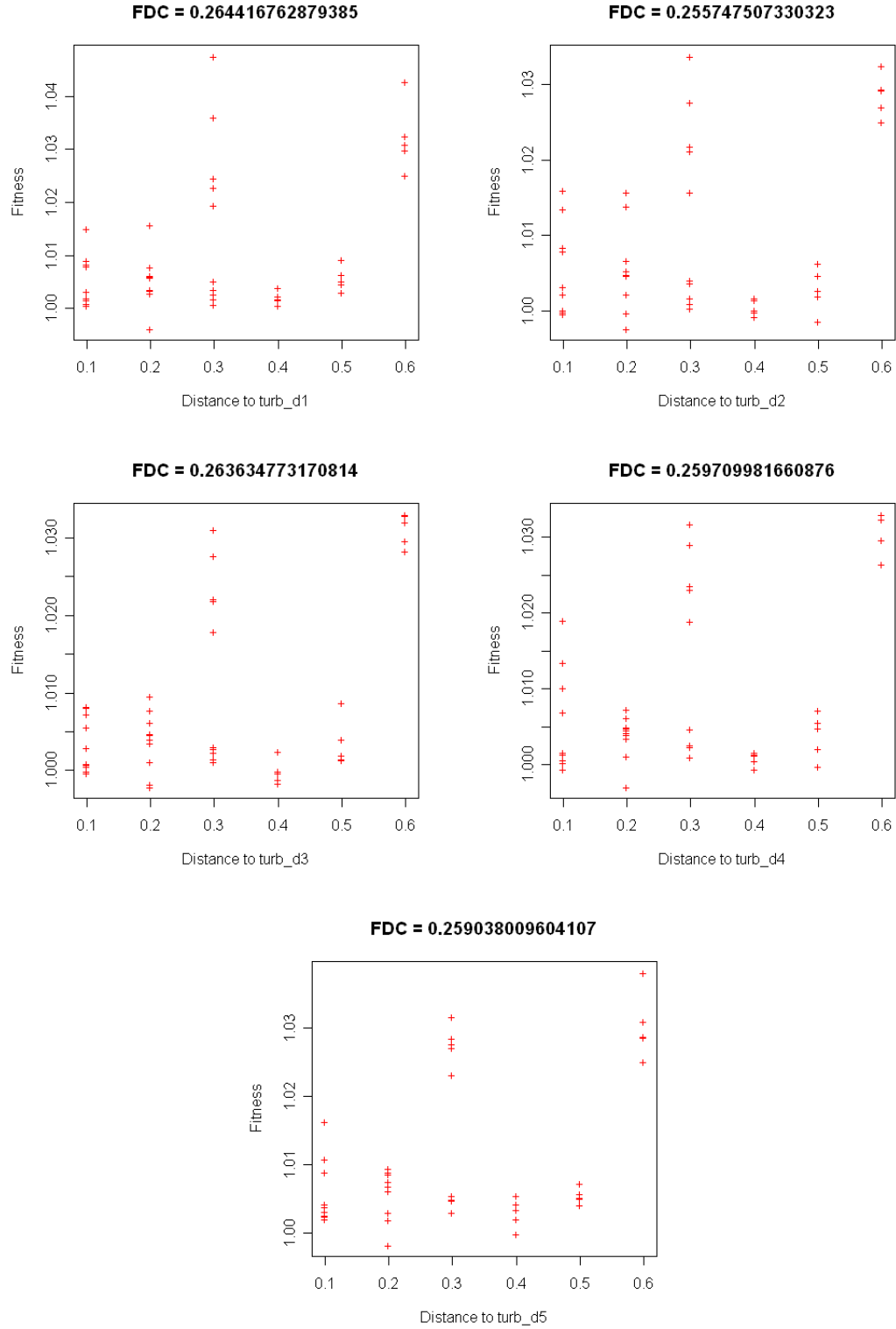
FIGURE E.10: *Graphics of the resultant scatter plots and correlation coefficients for the group **j** of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE E.11: *Graphics of the resultant scatter plots and correlation coefficients for the group **k** of CA Continuous showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

Appendix F

Turbulence CA Scatter Plots

This appendix lists a collection of scatter plots correspondent to the fitness distance correlation experiments performed over Turbulence CA in Section 8.2.1 of Chapter 8. In turns, each of the CA snapshots was considered as a target (T) against which the remaining snapshots of all the groups (T_i) were evaluated on fitness (f_i) using the USM and on distance (d_i) using Euclidean difference among the values of their associated creational parameters. Equation F.1 shows the calculation of FDC where n is the number of samples, $\overline{f}$ and S_F are the mean and standard deviation of the fitness values, and $\overline{d}$ and S_D are the mean and standard deviation of the distances.

$$f_i = f(T_i) = USM(T_i, T)$$

$$d_i = (\sum_{k=1}^{n}(par_k^i - par_k^T)^2)^{1/2}$$

$$FDC = \frac{(1/n)\sum_{i=1}^{n}(f_i - \overline{f})(d_i - \overline{d})}{S_F S_D}$$

$$T \text{ is a target snapshot}$$

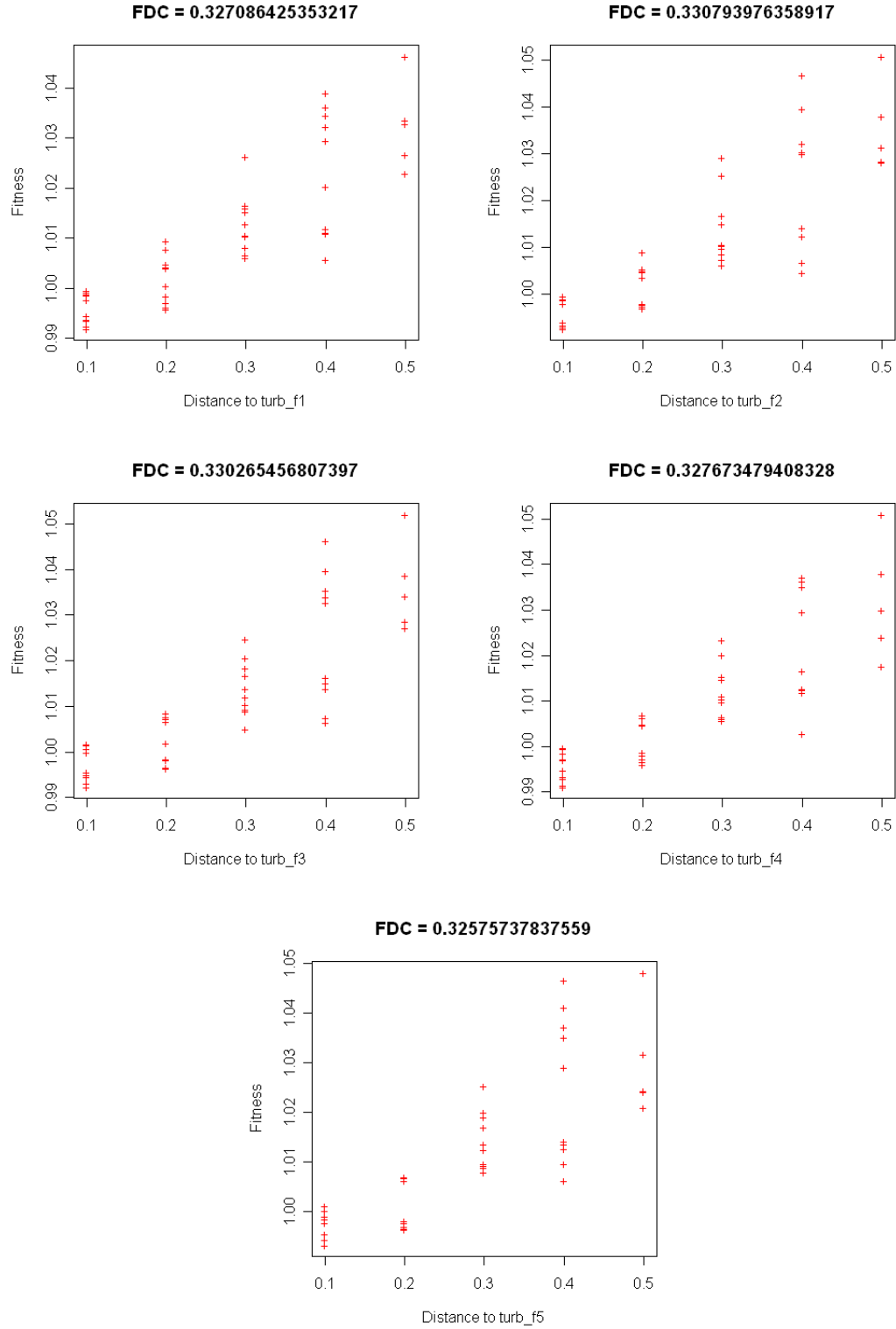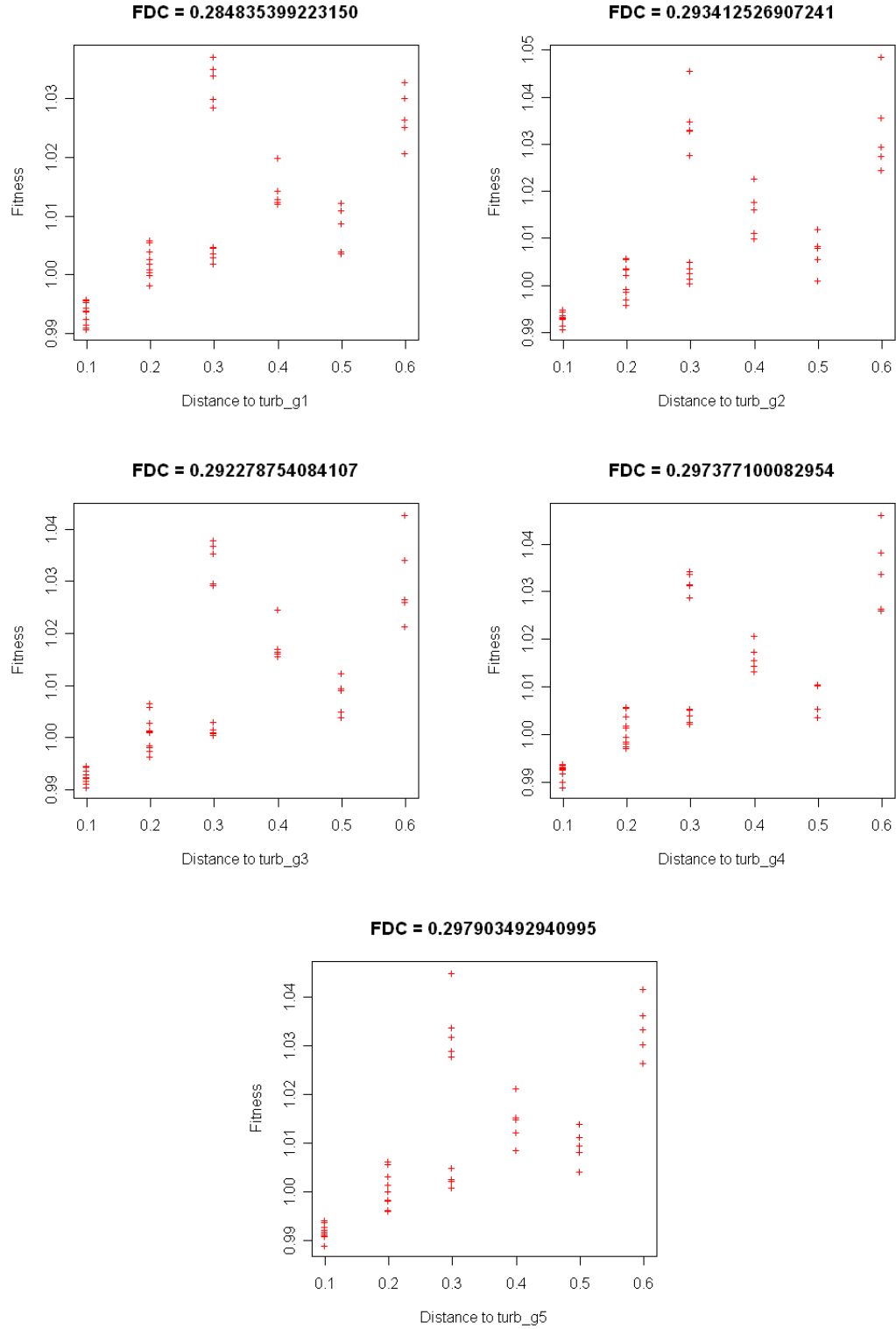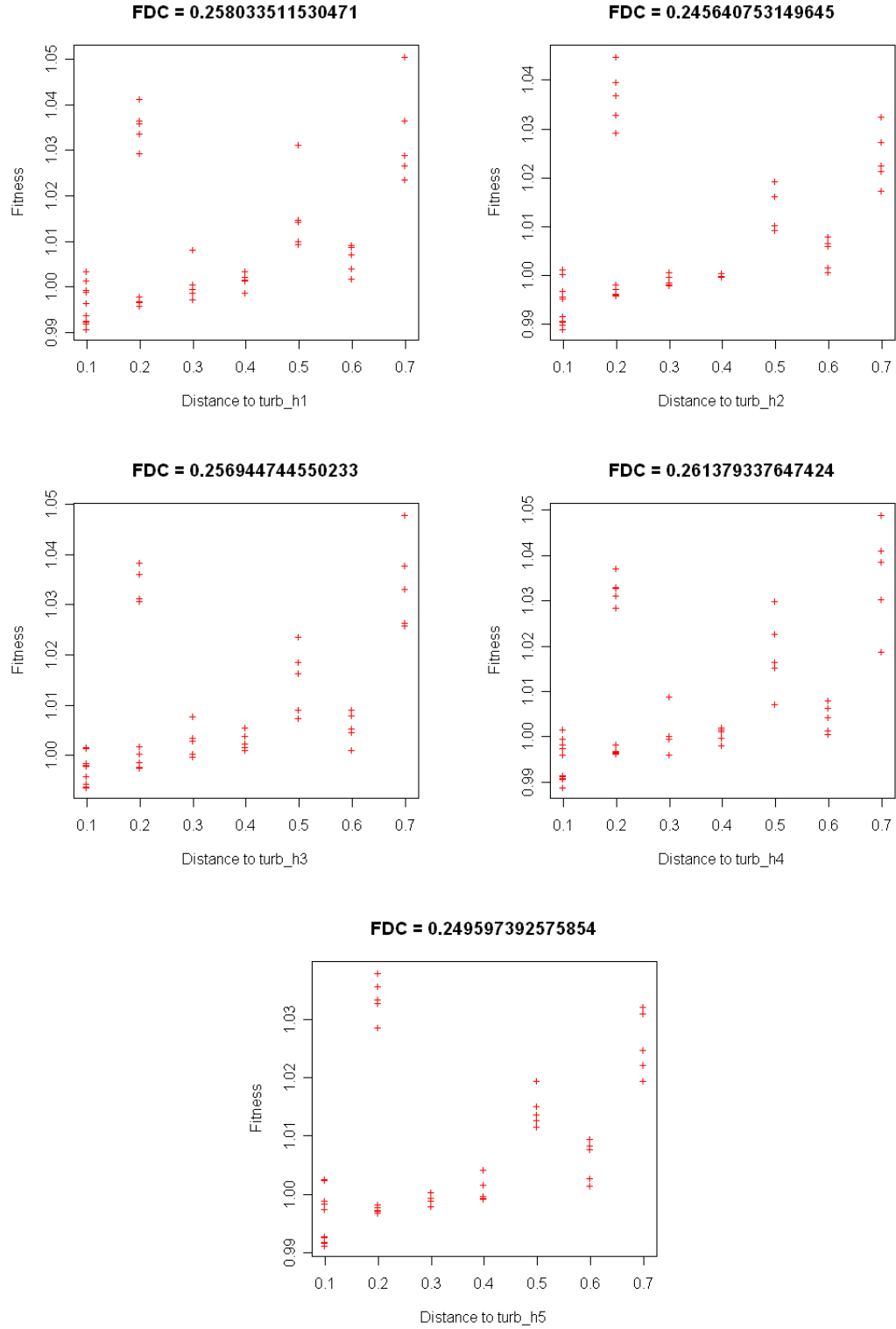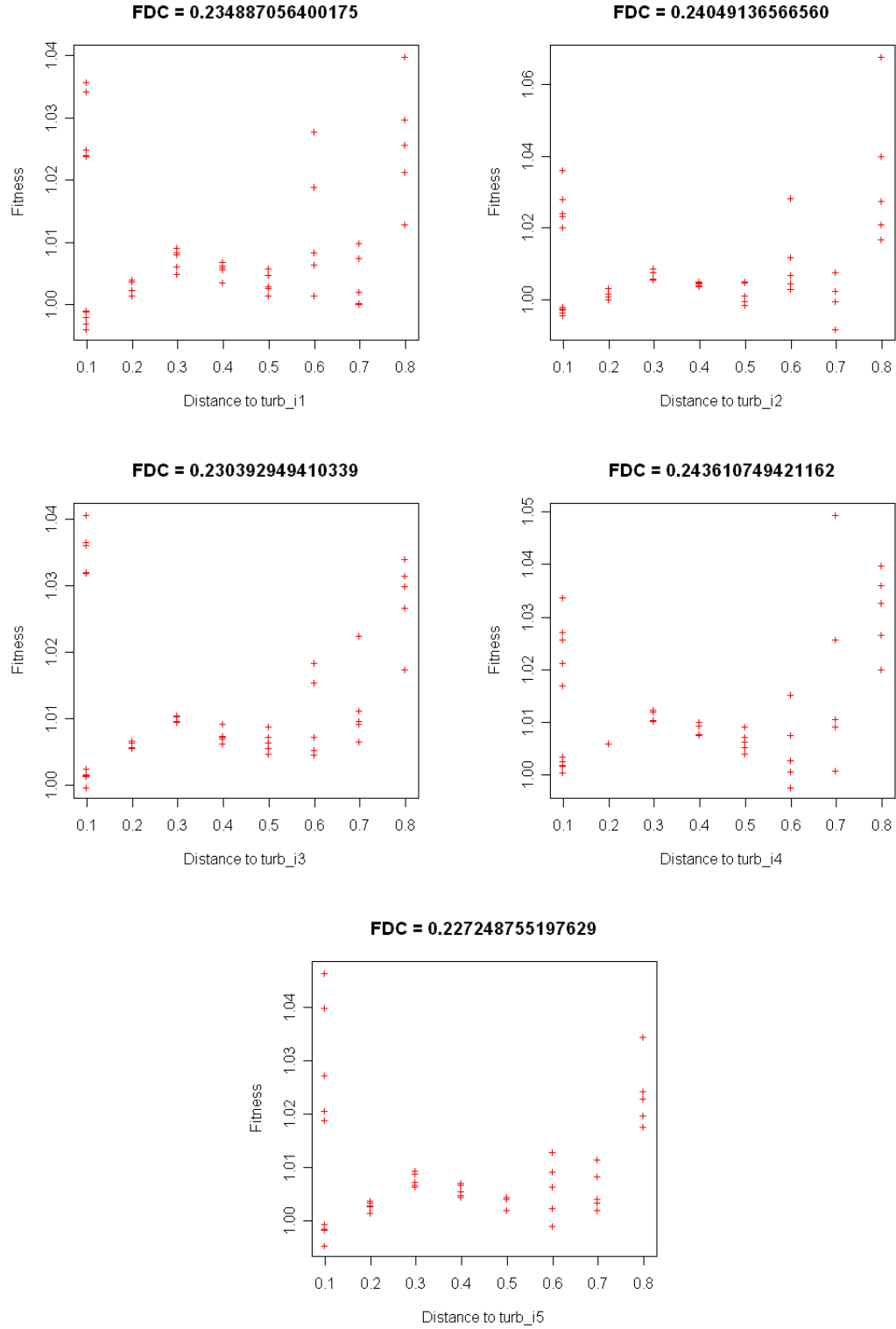$$par_k^i \text{ is the creational } k\text{-parameter} \tag{F.1}$$

FIGURE F.1: *Graphics of the resultant scatter plots and correlation coefficients for the group* **a** *of Turbulence CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE F.2: *Graphics of the resultant scatter plots and correlation coefficients for the group **b** of Turbulence CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
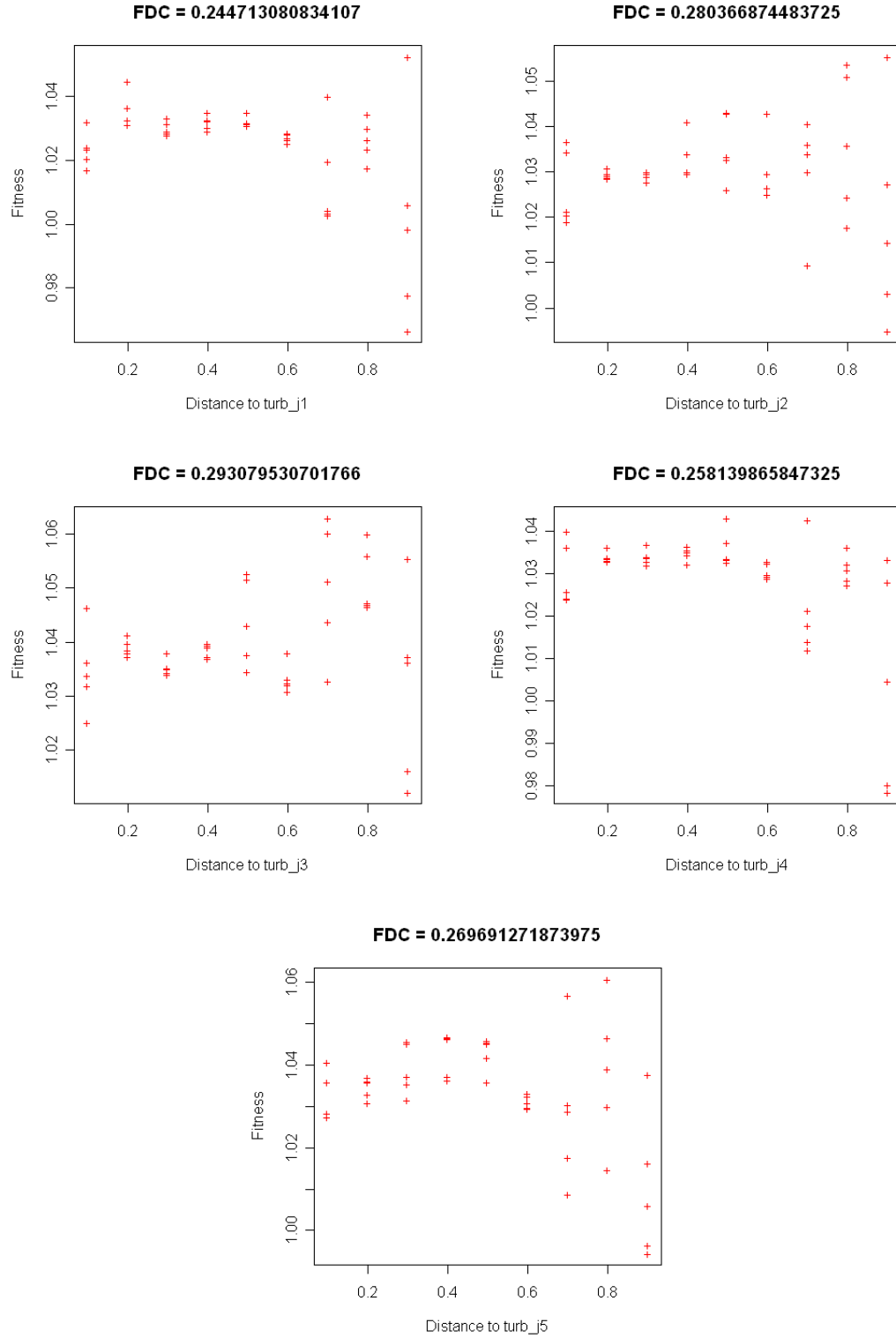
FIGURE F.3: *Graphics of the resultant scatter plots and correlation coefficients for the group **c** of Turbulence CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE F.4: *Graphics of the resultant scatter plots and correlation coefficients for the group **d** of Turbulence CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
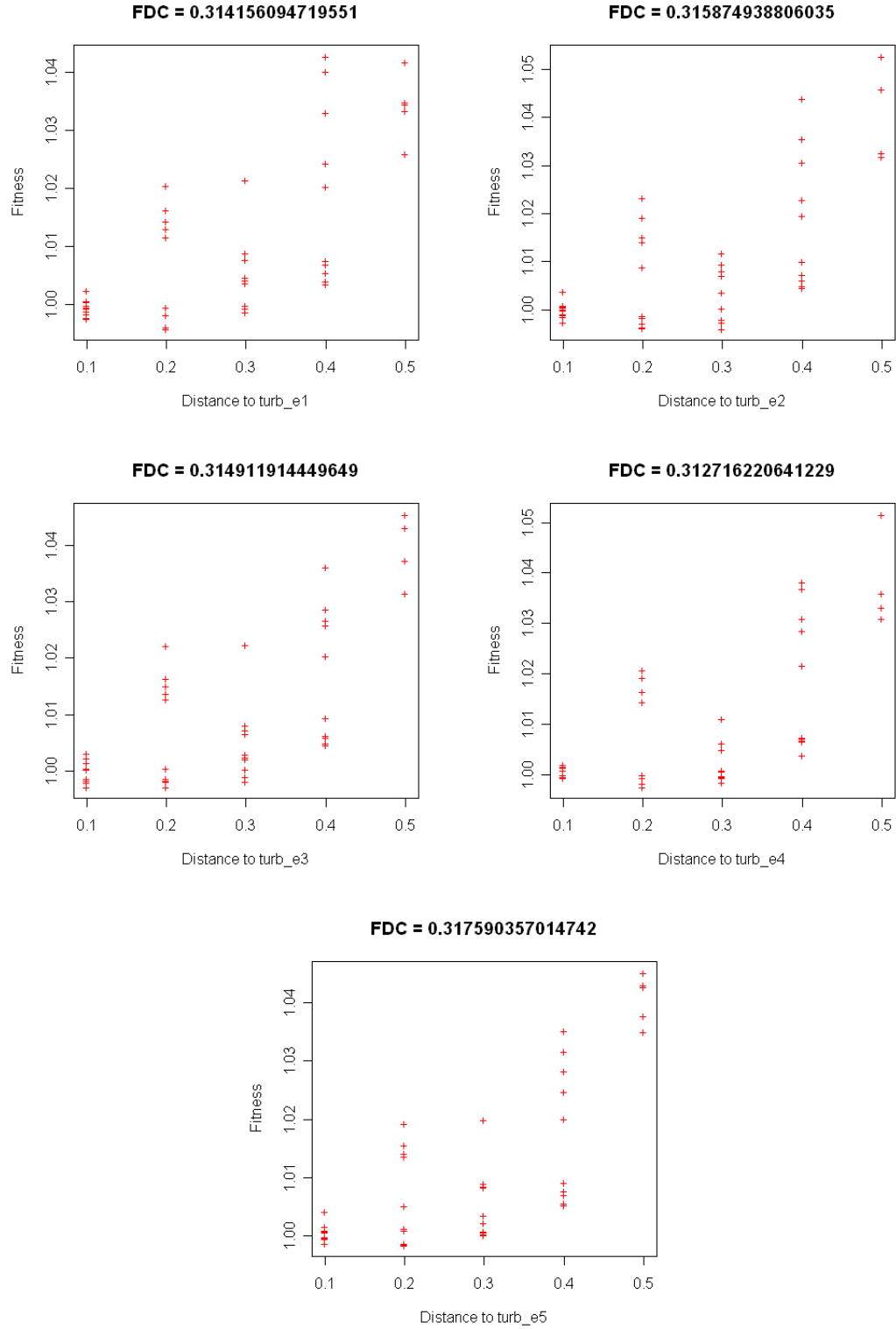
FIGURE F.5: *Graphics of the resultant scatter plots and correlation coefficients for the group **e** of Turbulence CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE F.6: *Graphics of the resultant scatter plots and correlation coefficients for the group **f** of Turbulence CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE F.7: *Graphics of the resultant scatter plots and correlation coefficients for the group **g** of Turbulence CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*
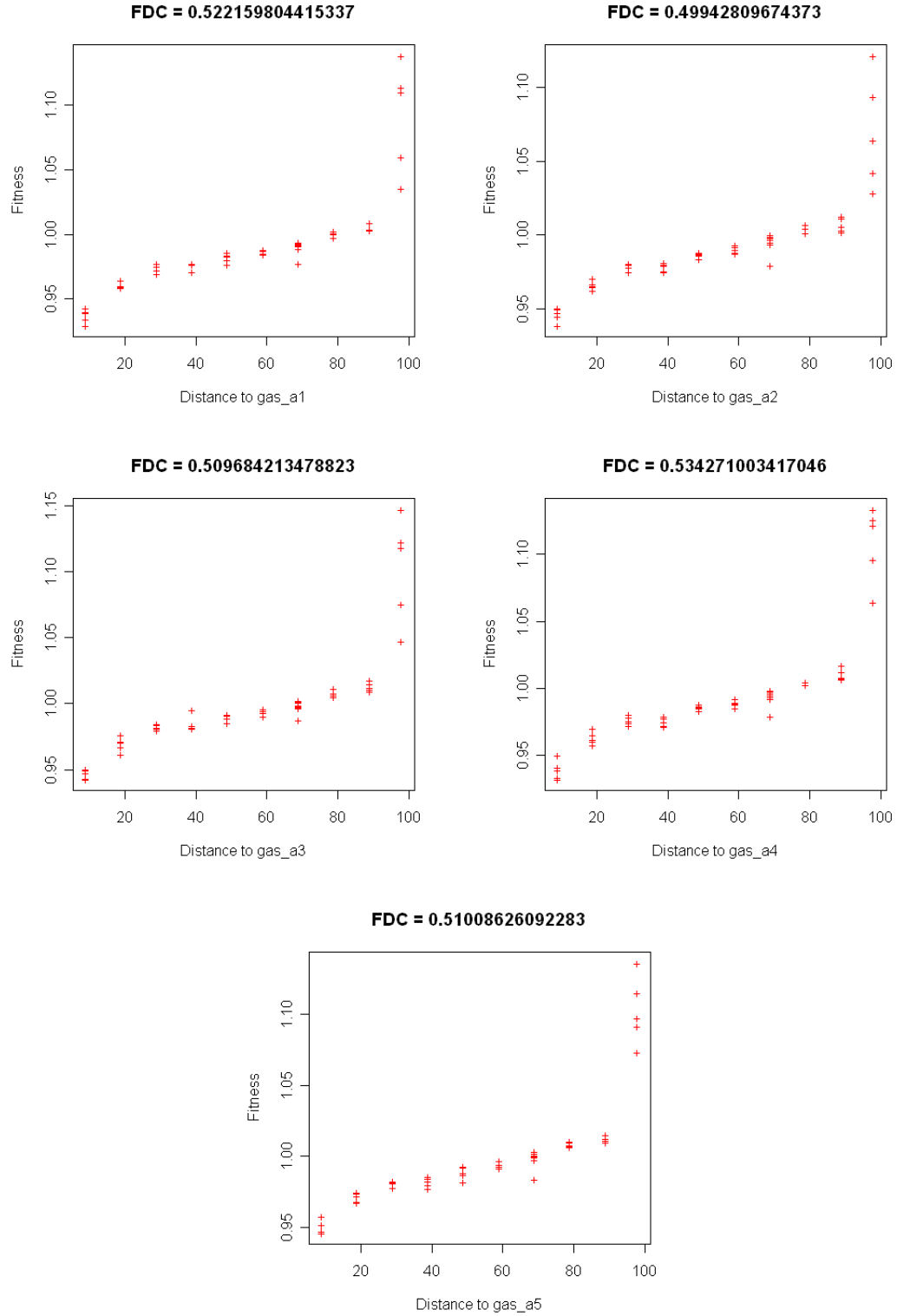
FIGURE F.8: *Graphics of the resultant scatter plots and correlation coefficients for the group **h** of Turbulence CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*
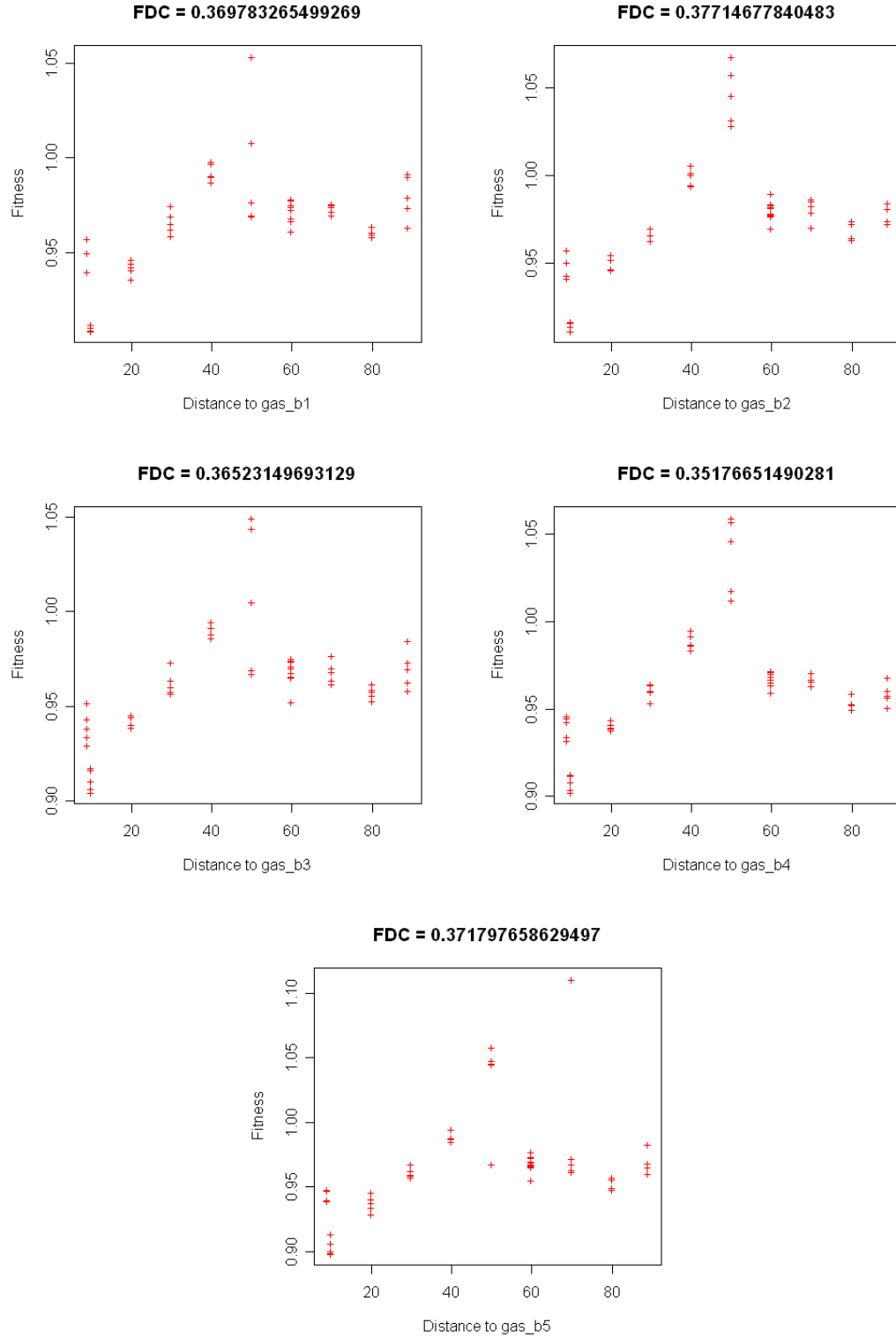
FIGURE F.9: *Graphics of the resultant scatter plots and correlation coefficients for the group **i** of Turbulence CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE F.10: *Graphics of the resultant scatter plots and correlation coefficients for the group **j** of Turbulence CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

APPENDIX G

Gas Lattice CA Scatter Plots

This appendix lists a collection of scatter plots correspondent to the fitness distance correlation experiments performed over Gas Lattice CA in Section 8.2.1 of Chapter 8. In turns, each of the CA snapshots was considered as a target (T) against which the remaining snapshots of all the groups (T_i) were evaluated on fitness (f_i) using the USM and on distance (d_i) using Euclidean difference among the values of their associated creational parameters. Equation H.1 shows the calculation of FDC where n is the number of samples, $\overline{f}$ and S_F are the mean and standard deviation of the fitness values, and $\overline{d}$ and S_D are the mean and standard deviation of the distances.

$$f_i = f(T_i) = USM(T_i, T)$$

$$d_i = (\sum_{k=1}^{n}(par_k^i - par_k^T)^2)^{1/2}$$

$$FDC = \frac{(1/n)\sum_{i=1}^{n}(f_i - \overline{f})(d_i - \overline{d})}{S_F S_D}$$

$$T \text{ is a target snapshot}$$

$$par_k^i \text{ is the creational } k\text{-parameter} \qquad\qquad (G.1)$$

FIGURE G.1: *Graphics of the resultant scatter plots and correlation coefficients for the group $\boldsymbol{a}$ of Gas Lattice CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*

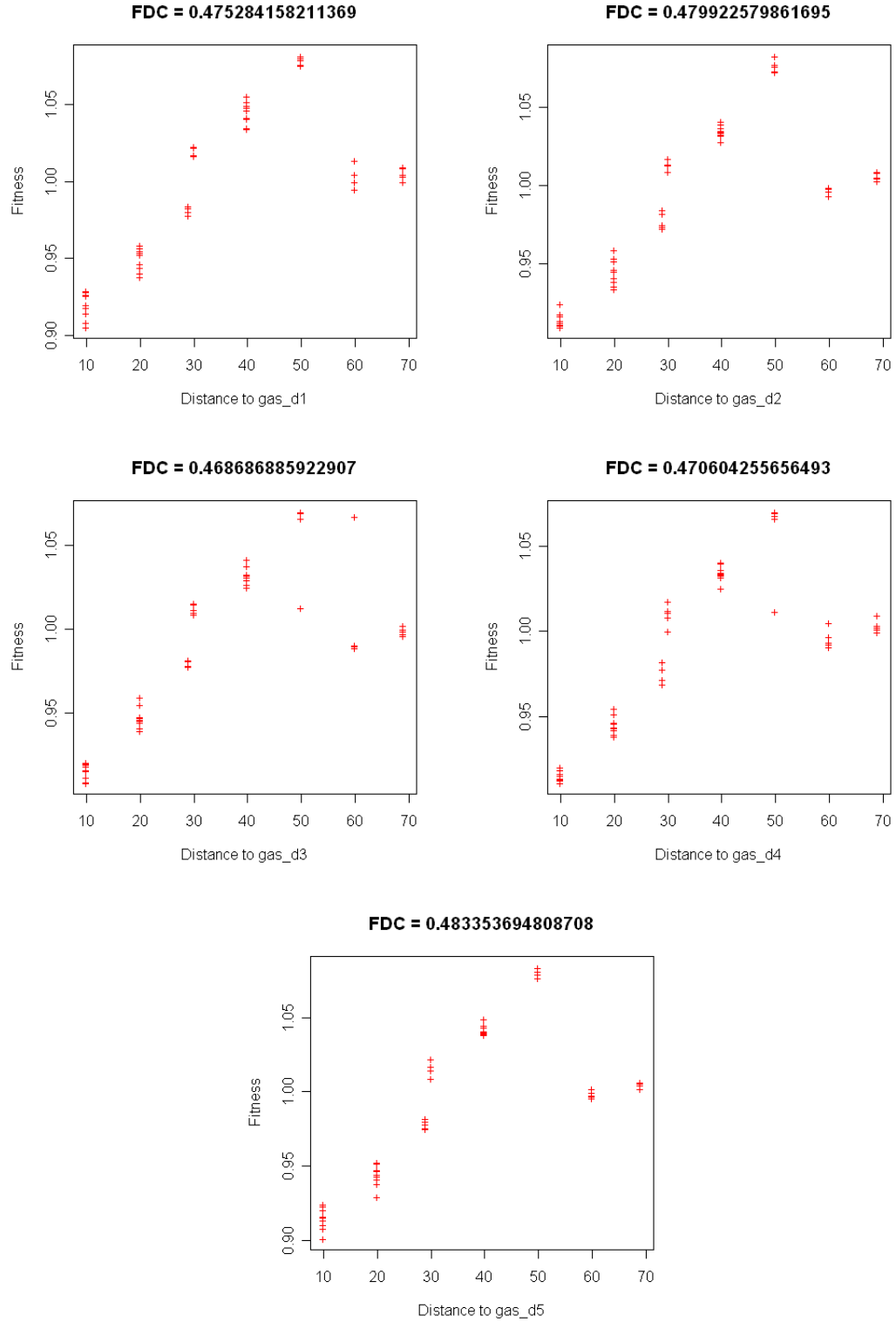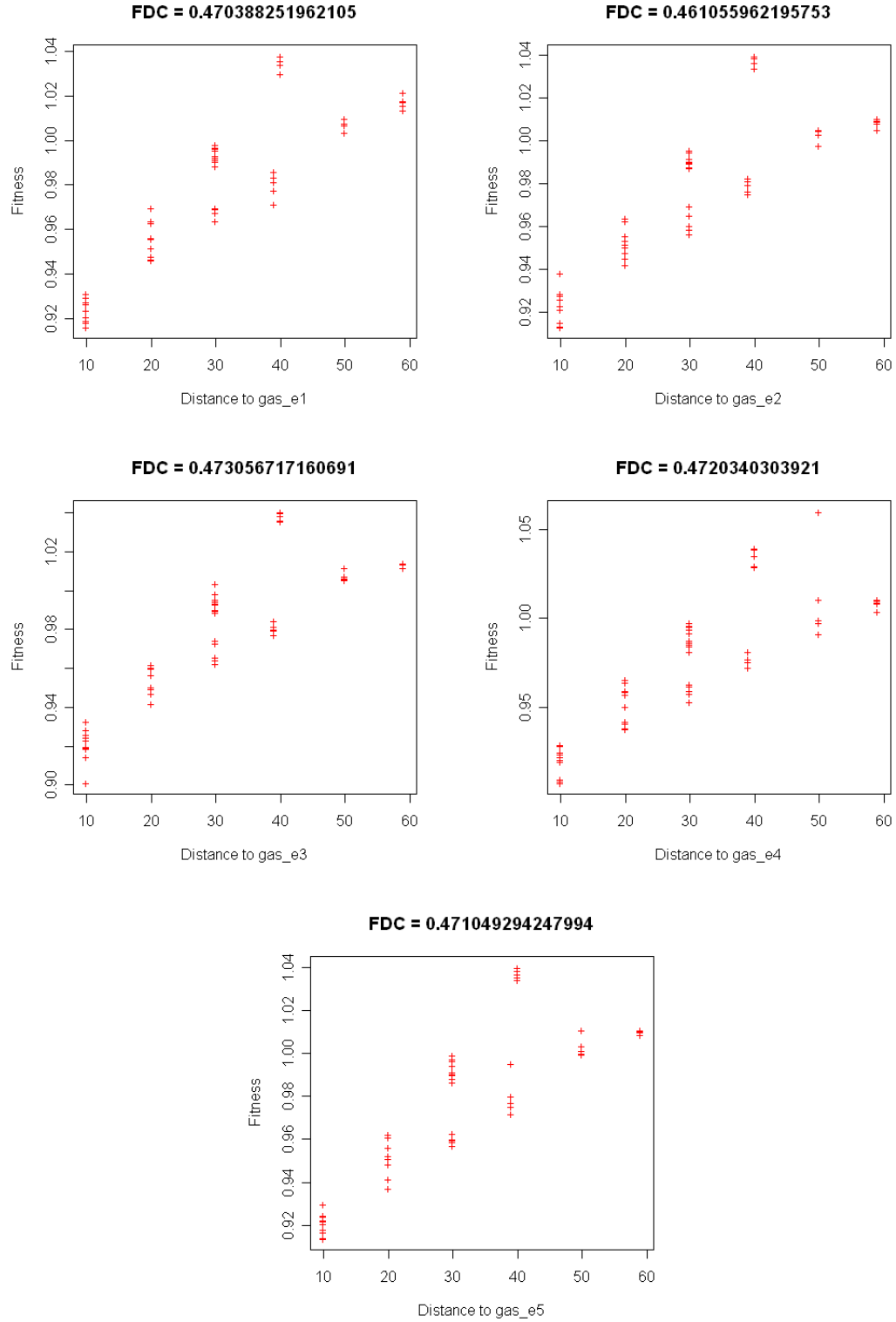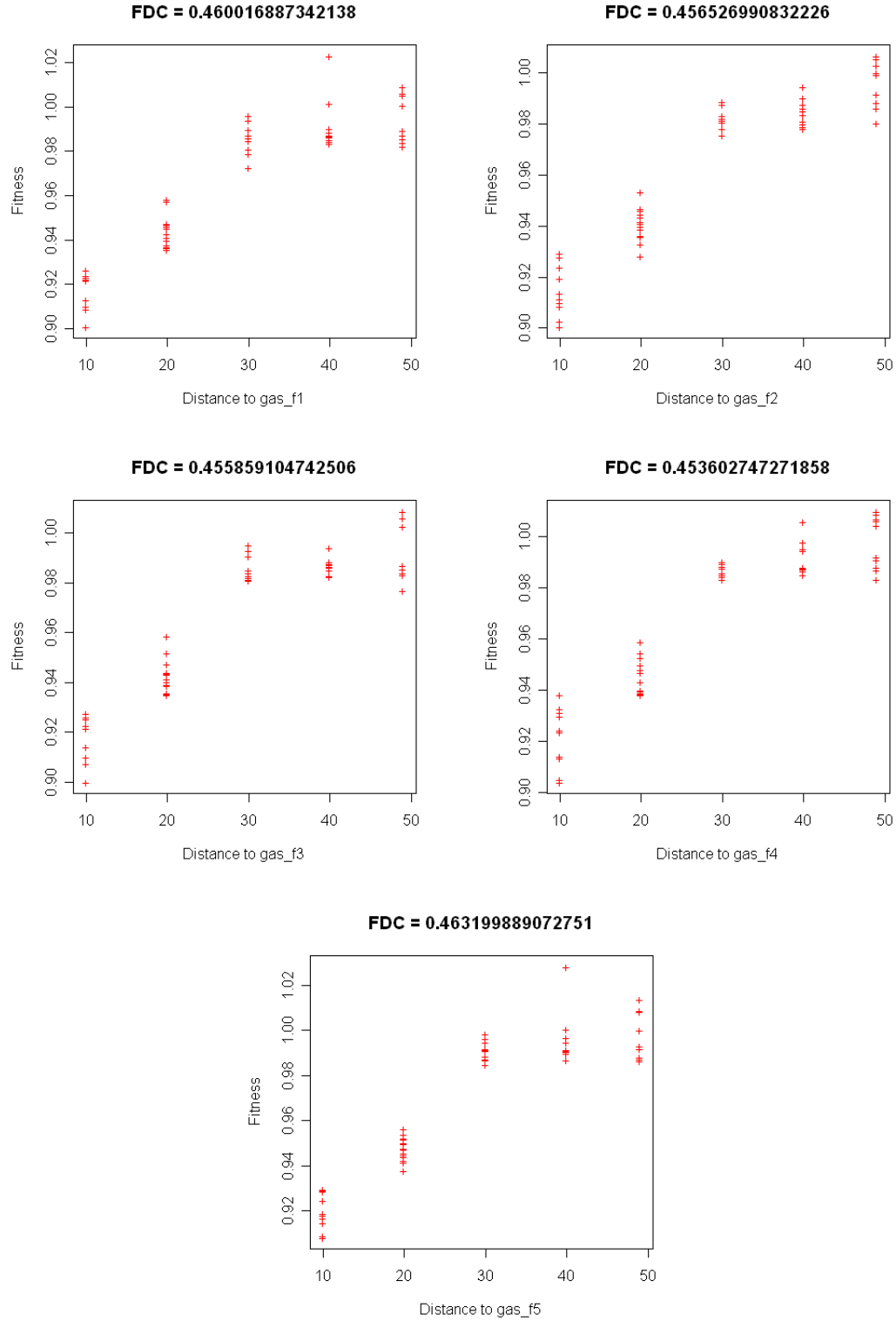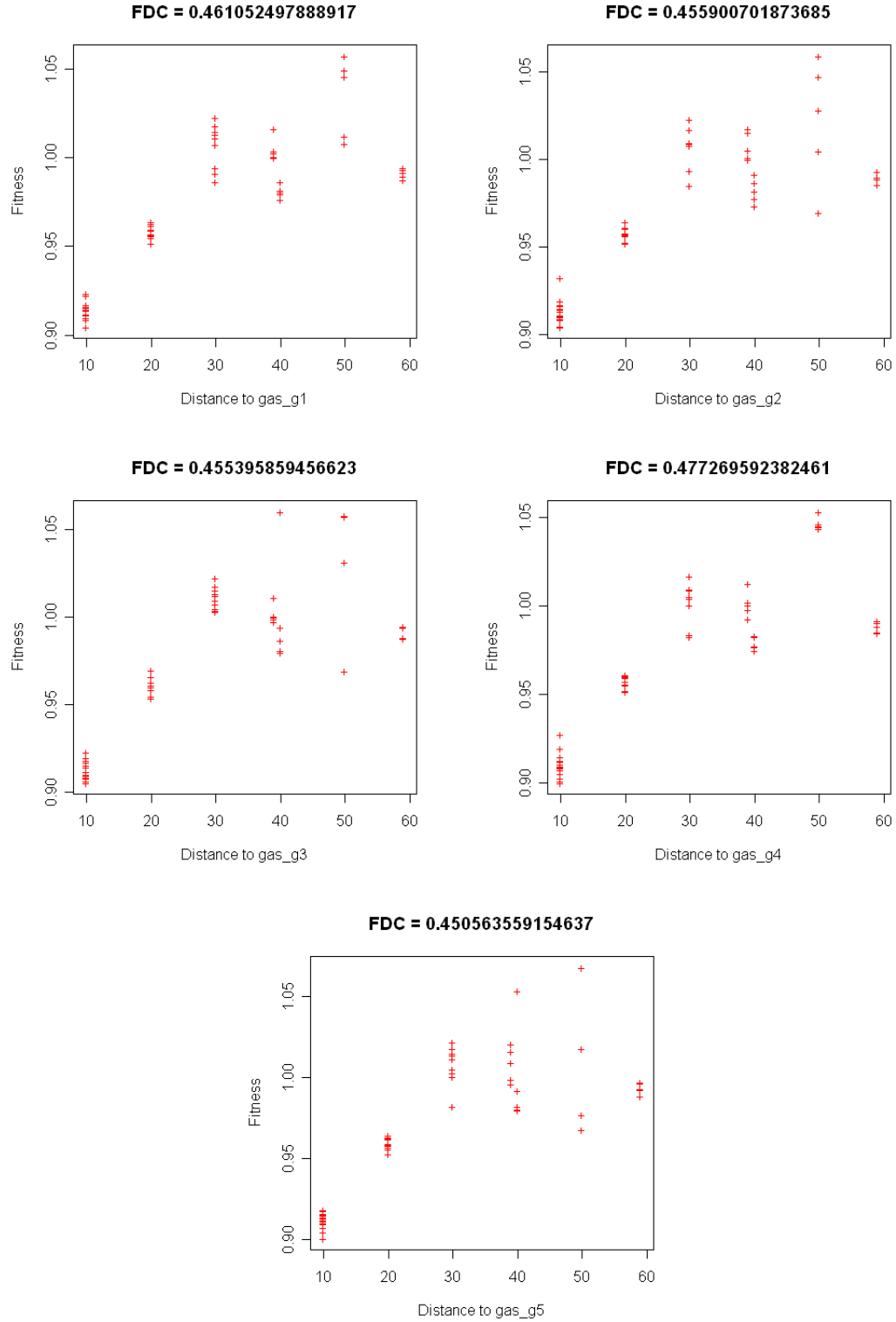FIGURE G.2: *Graphics of the resultant scatter plots and correlation coefficients for the group **b** of Gas Lattice CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
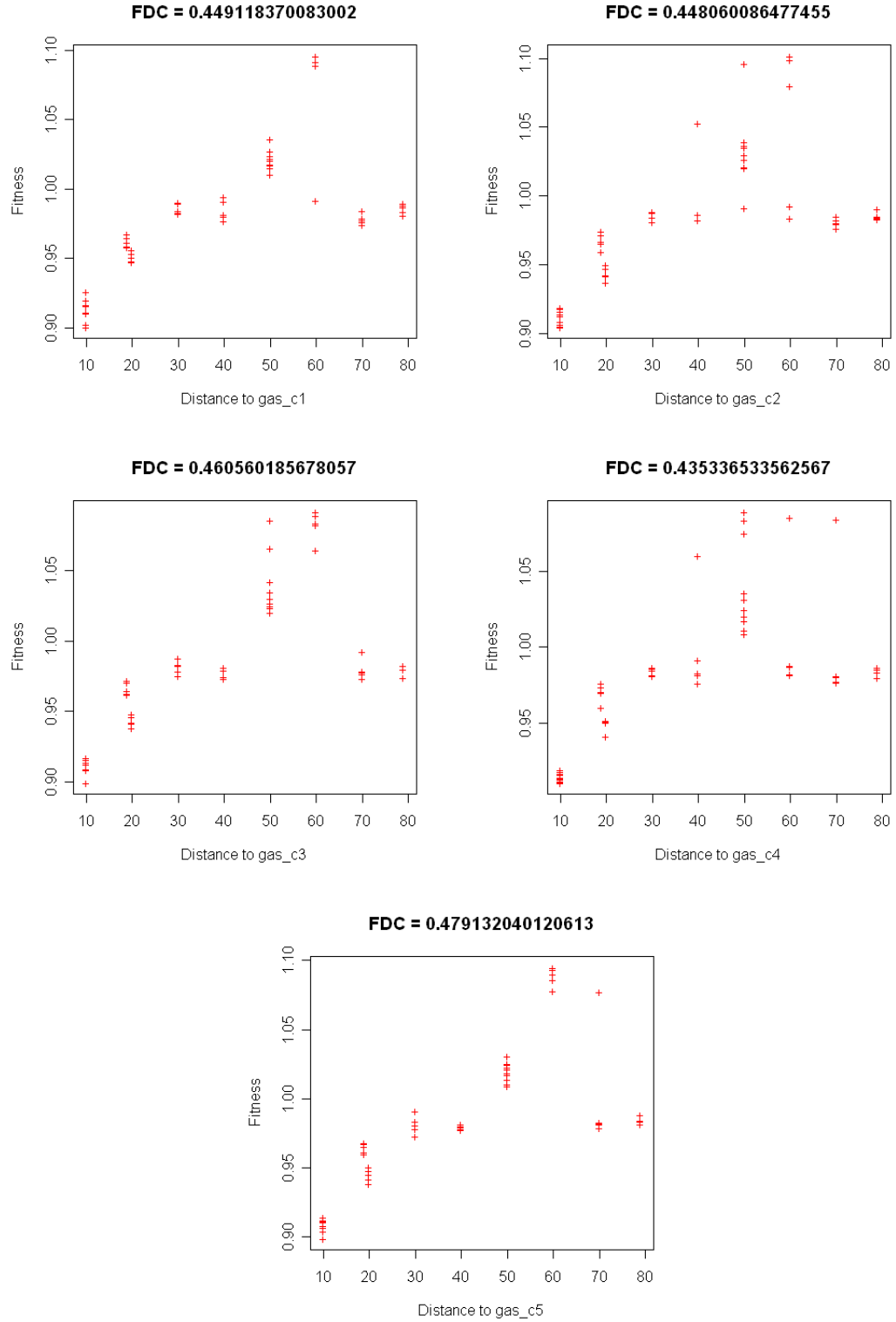
FIGURE G.3: *Graphics of the resultant scatter plots and correlation coefficients for the group **c** of Gas Lattice CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*
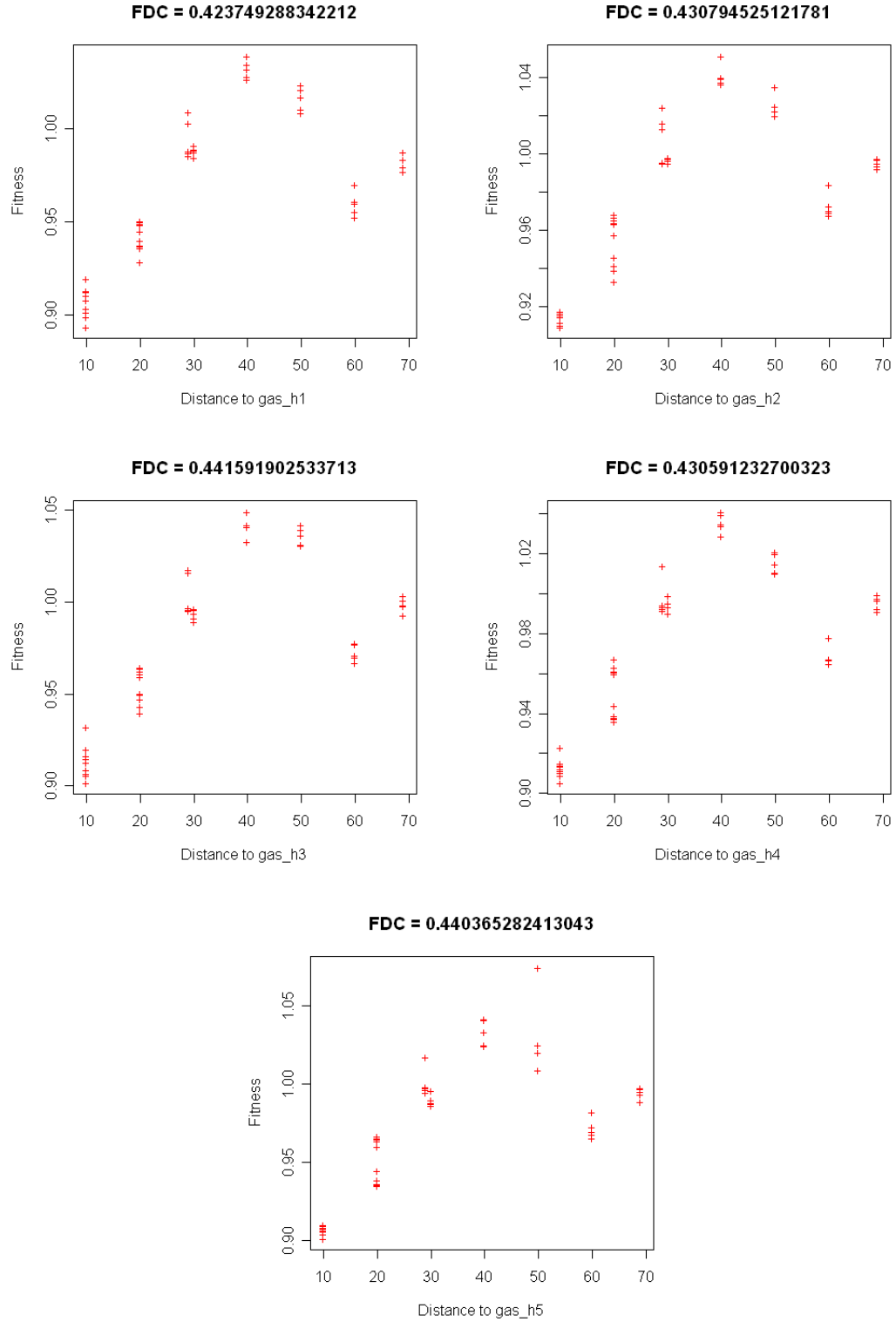
FIGURE G.4: *Graphics of the resultant scatter plots and correlation coefficients for the group **d** of Gas Lattice CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*
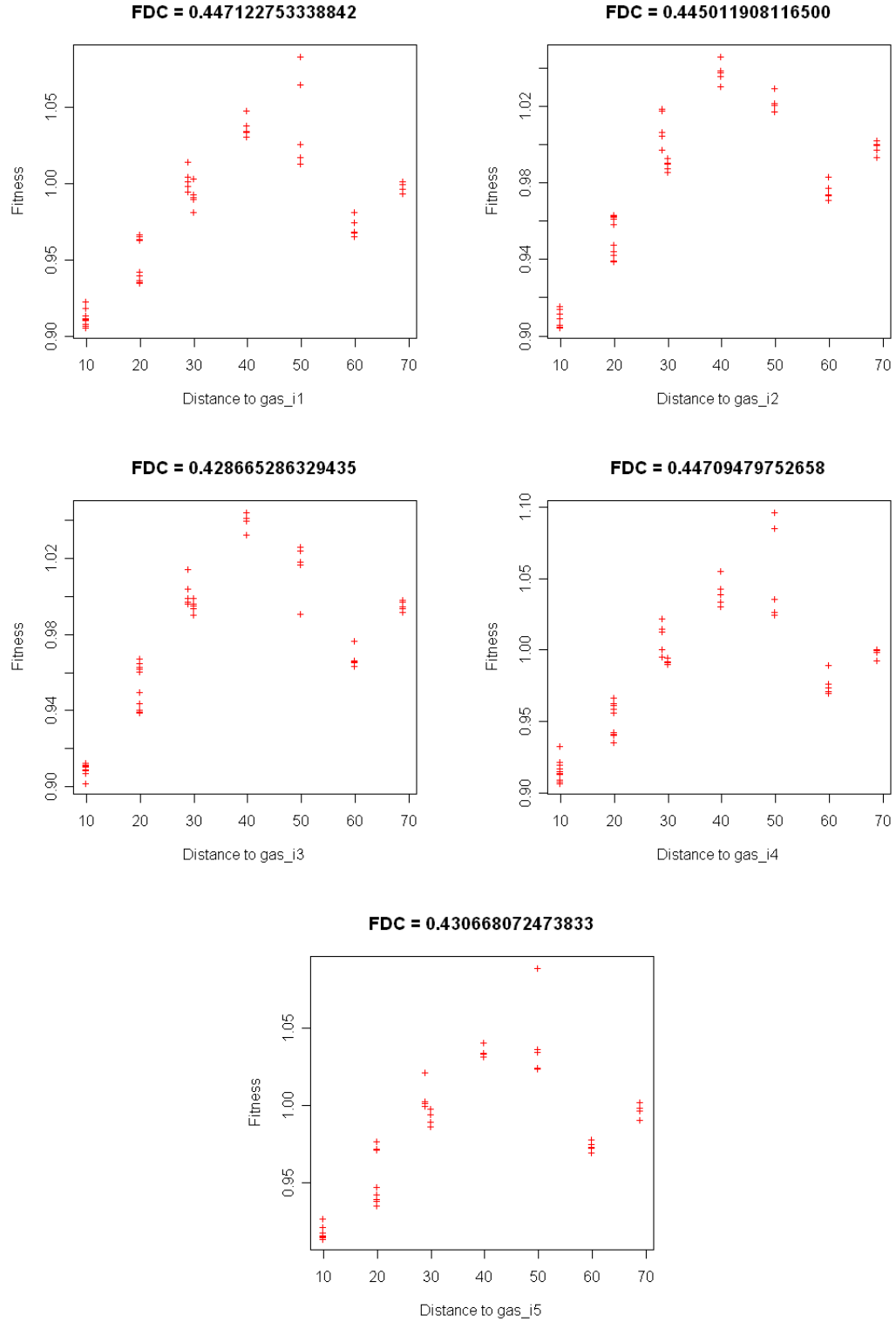
FIGURE G.5: *Graphics of the resultant scatter plots and correlation coefficients for the group **e** of Gas Lattice CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*
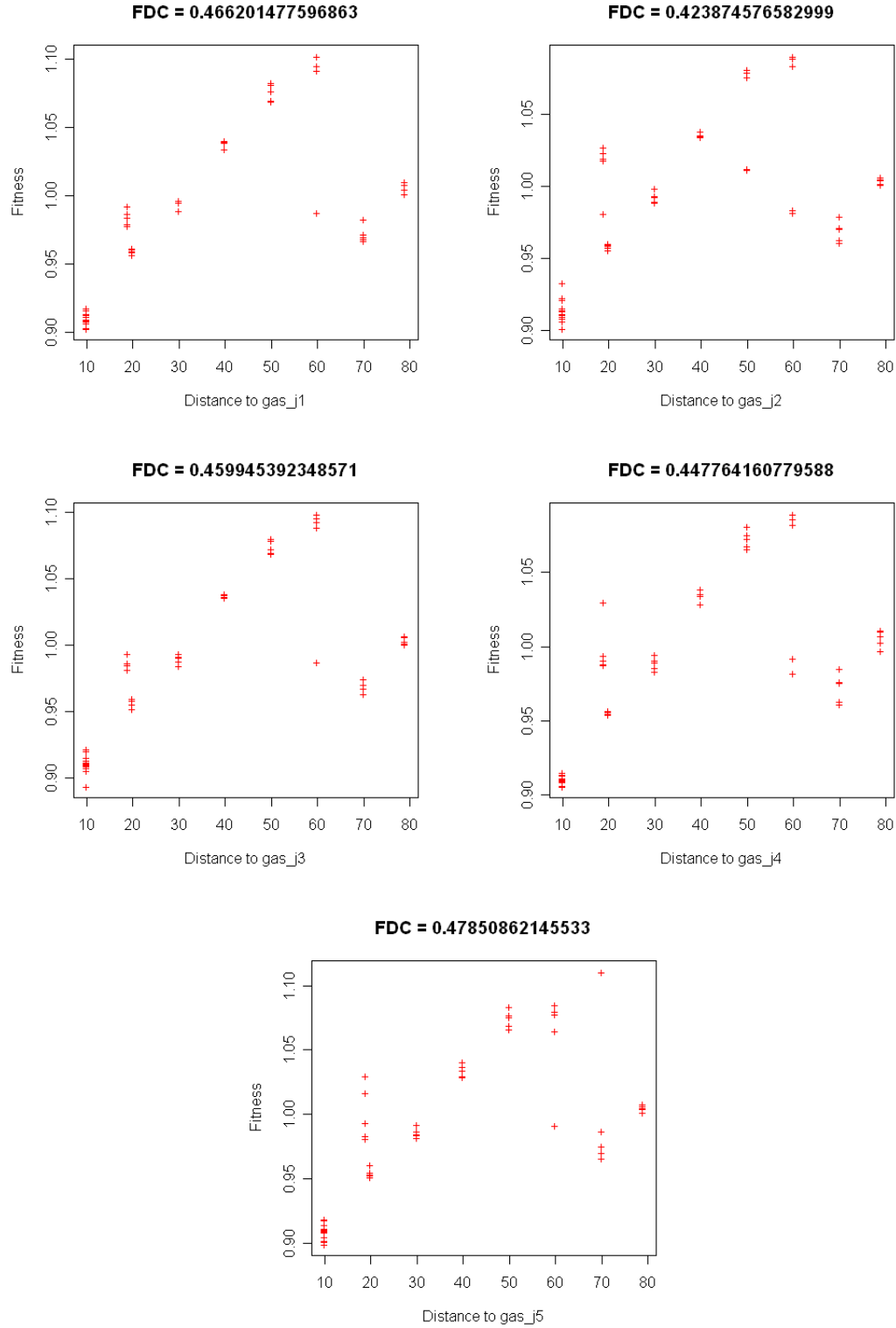
FIGURE G.6: *Graphics of the resultant scatter plots and correlation coefficients for the group f of Gas Lattice CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*
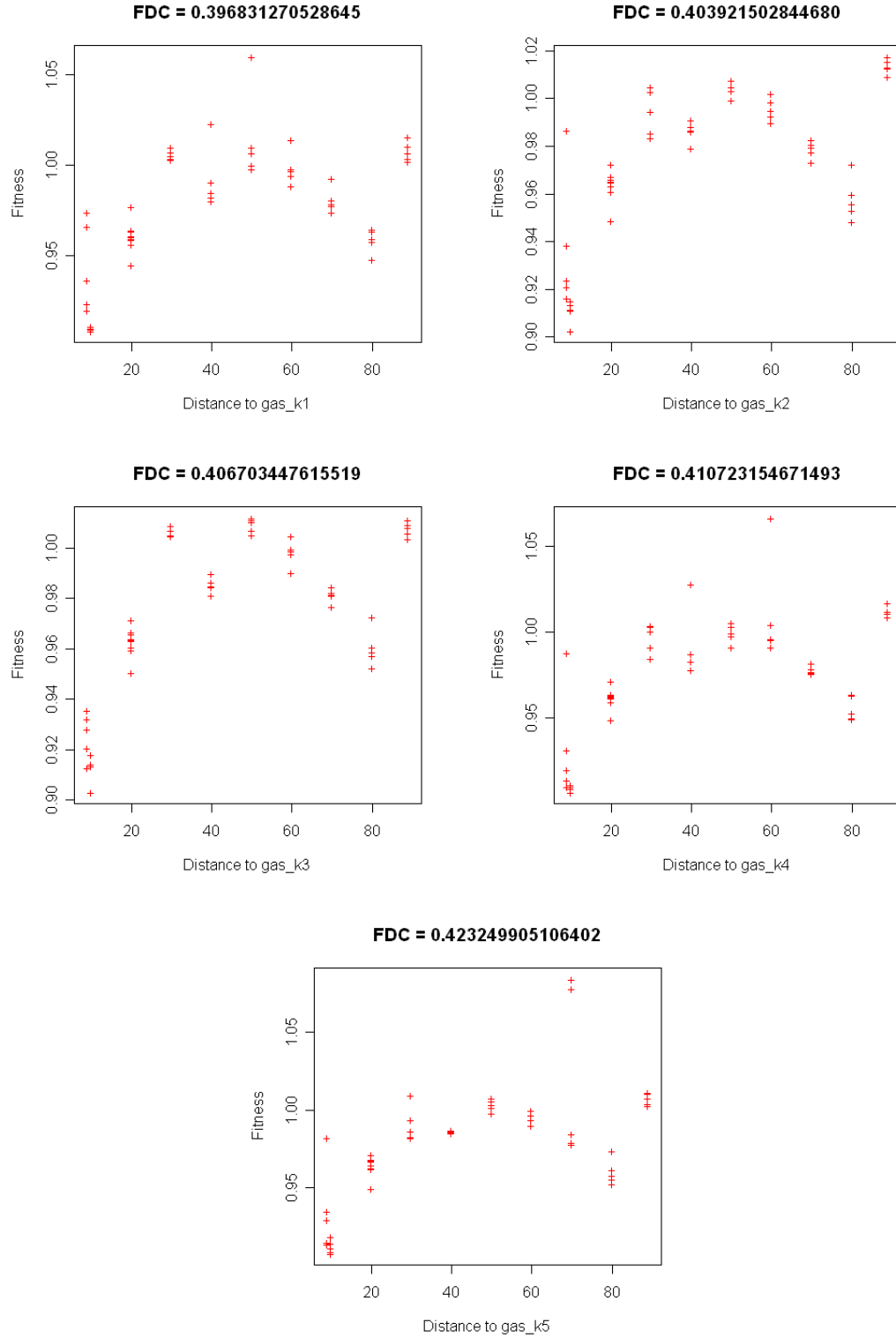
FIGURE G.7: *Graphics of the resultant scatter plots and correlation coefficients for the group **g** of Gas Lattice CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*
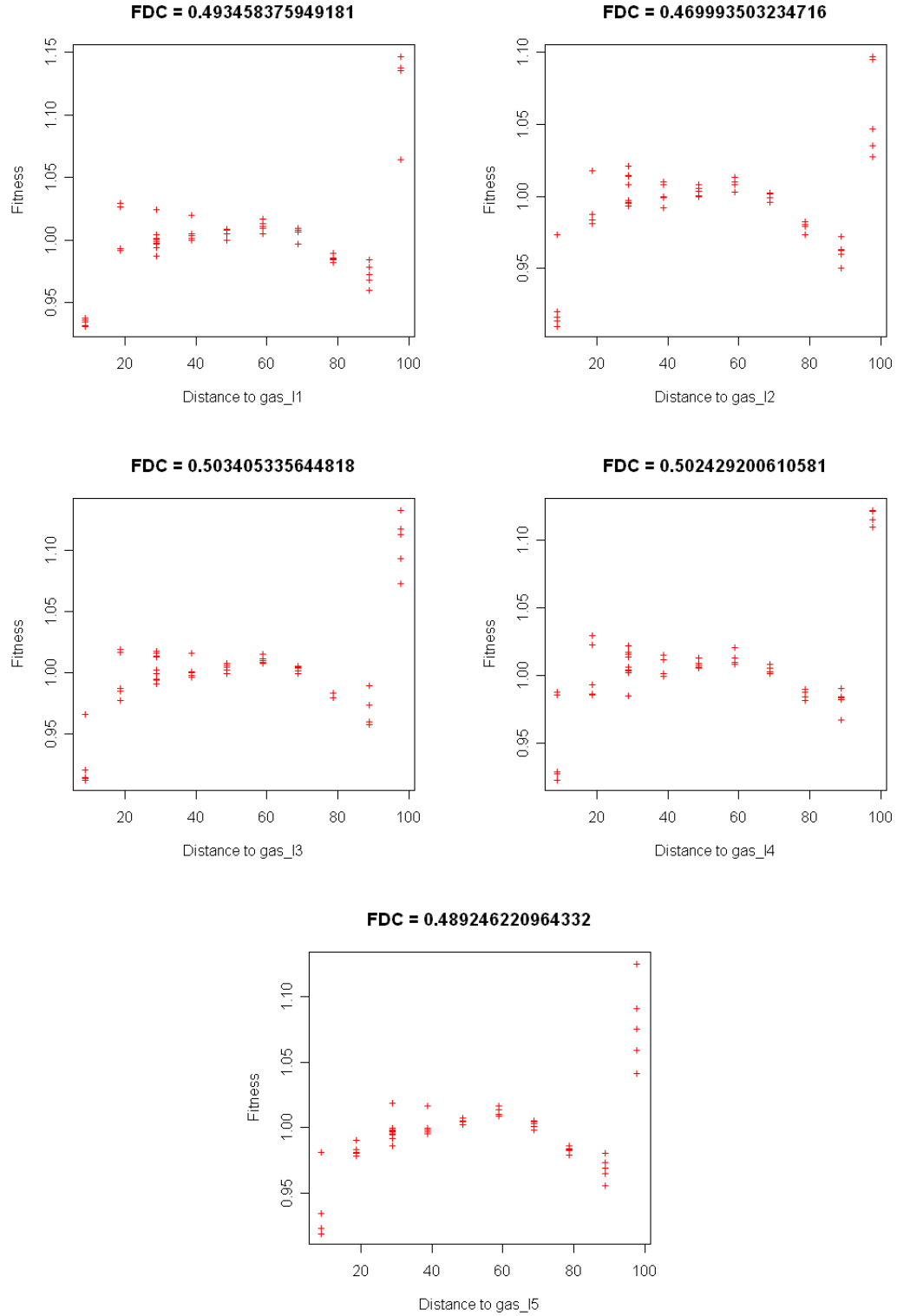
FIGURE G.8: *Graphics of the resultant scatter plots and correlation coefficients for the group **h** of Gas Lattice CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE G.9: *Graphics of the resultant scatter plots and correlation coefficients for the group **i** of Gas Lattice CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE G.10: *Graphics of the resultant scatter plots and correlation coefficients for the group **j** of Gas Lattice CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE G.11: *Graphics of the resultant scatter plots and correlation coefficients for the group **k** of Gas Lattice CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE G.12: *Graphics of the resultant scatter plots and correlation coefficients for the group **l** of Gas Lattice CA showing that the USM has a relatively high correlation with the genotype of the spatio-temporal behaviour pattern.*

Appendix H

Meta-automaton CA Scatter Plots

This appendix lists a collection of scatter plots correspondent to the fitness distance correlation experiments performed over Meta-automaton CA in Section 8.2.1 of Chapter 8. In turns, each of the CA snapshots was considered as a target (T) against which the remaining snapshots of all the groups (T_i) were evaluated on fitness (f_i) using the USM and on distance (d_i) using Euclidean difference among the values of their associated creational parameters. Equation H.1 shows the calculation of FDC where n is the number of samples, $\overline{f}$ and S_F are the mean and standard deviation of the fitness values, and $\overline{d}$ and S_D are the mean and standard deviation of the distances.

$$f_i = f(T_i) = USM(T_i, T)$$

$$d_i = (\sum_{k=1}^{n}(par_k^i - par_k^T)^2)^{1/2}$$

$$FDC = \frac{(1/n)\sum_{i=1}^{n}(f_i - \overline{f})(d_i - \overline{d})}{S_F S_D}$$

$$T \text{ is a target snapshot}$$

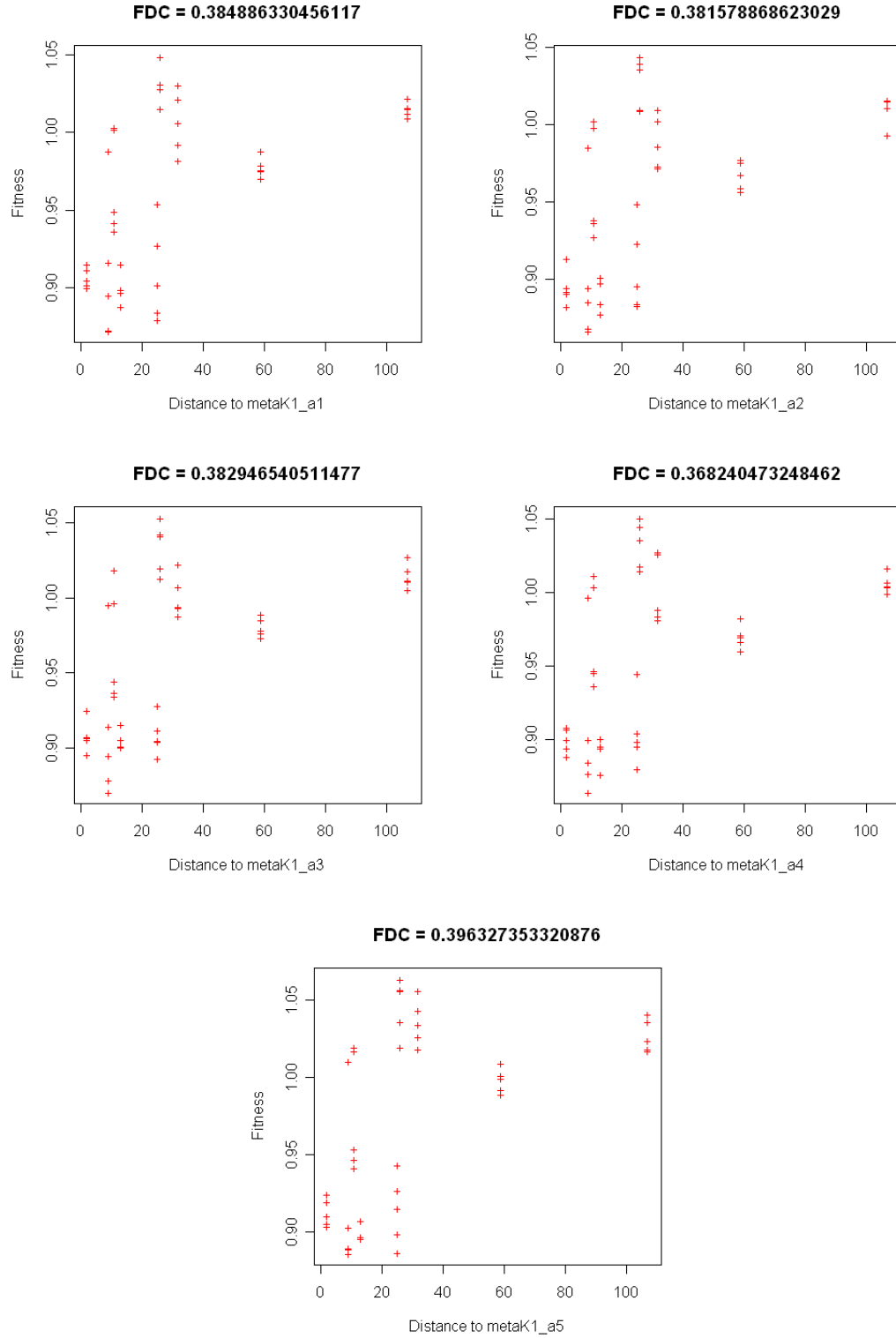$$par_k^i \text{ is the creational } k\text{-parameter} \tag{H.1}$$

FIGURE H.1: *Graphics of the resultant scatter plots and correlation coefficients for the group **a** of Meta-automaton CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
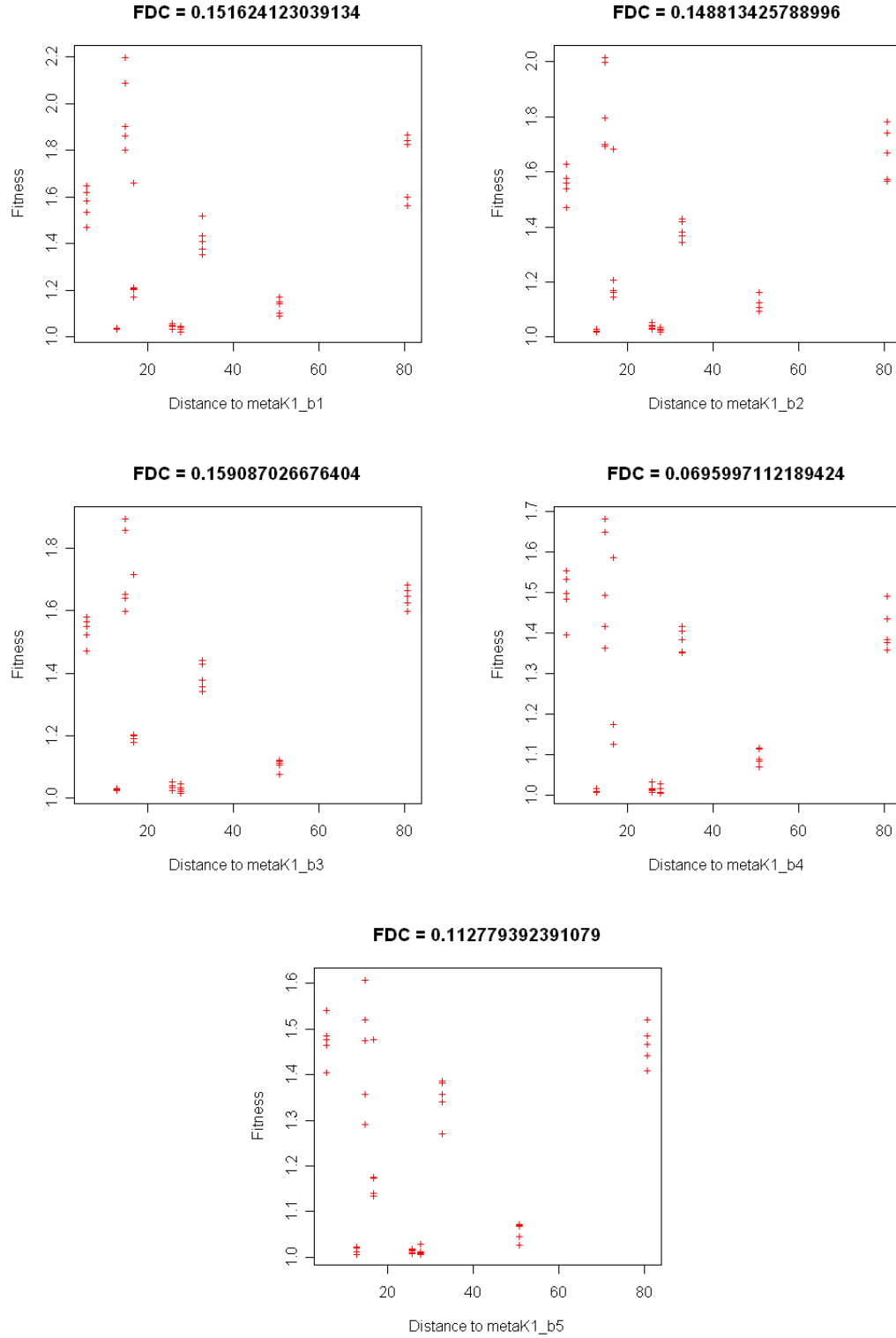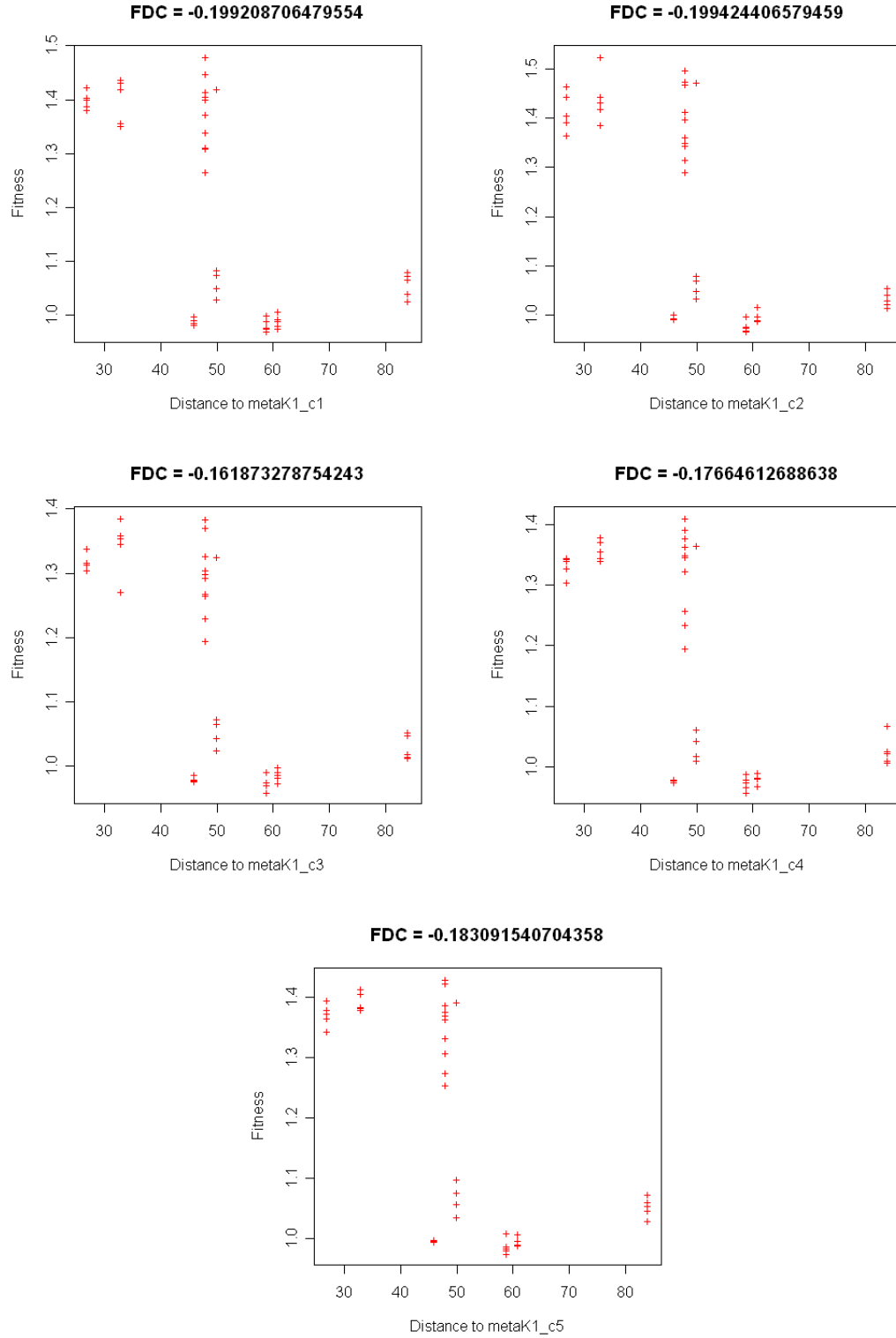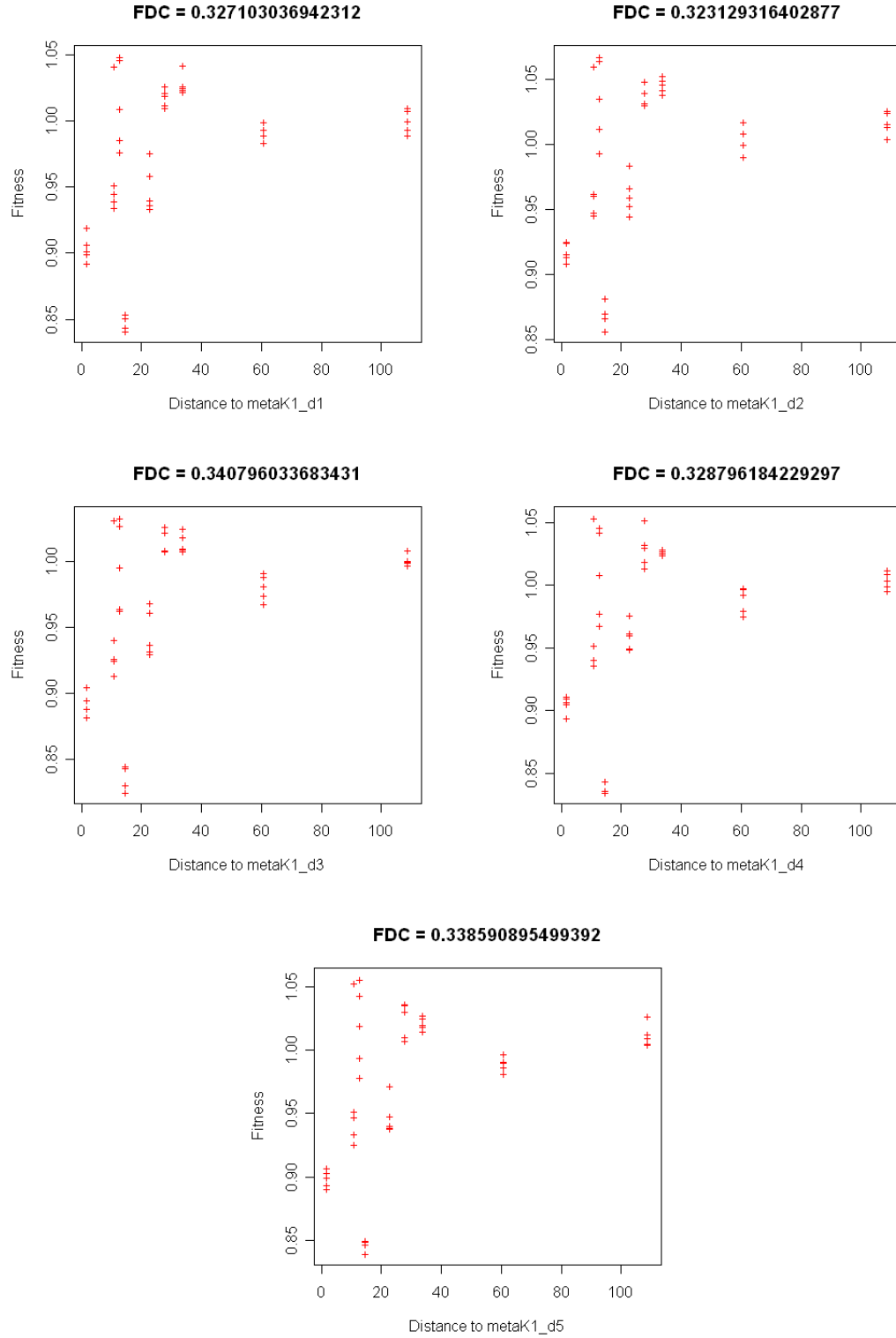
FIGURE H.2: *Graphics of the resultant scatter plots and correlation coefficients for the group **b** of Meta-automaton CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE H.3: *Graphics of the resultant scatter plots and correlation coefficients for the group **c** of Meta-automaton CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
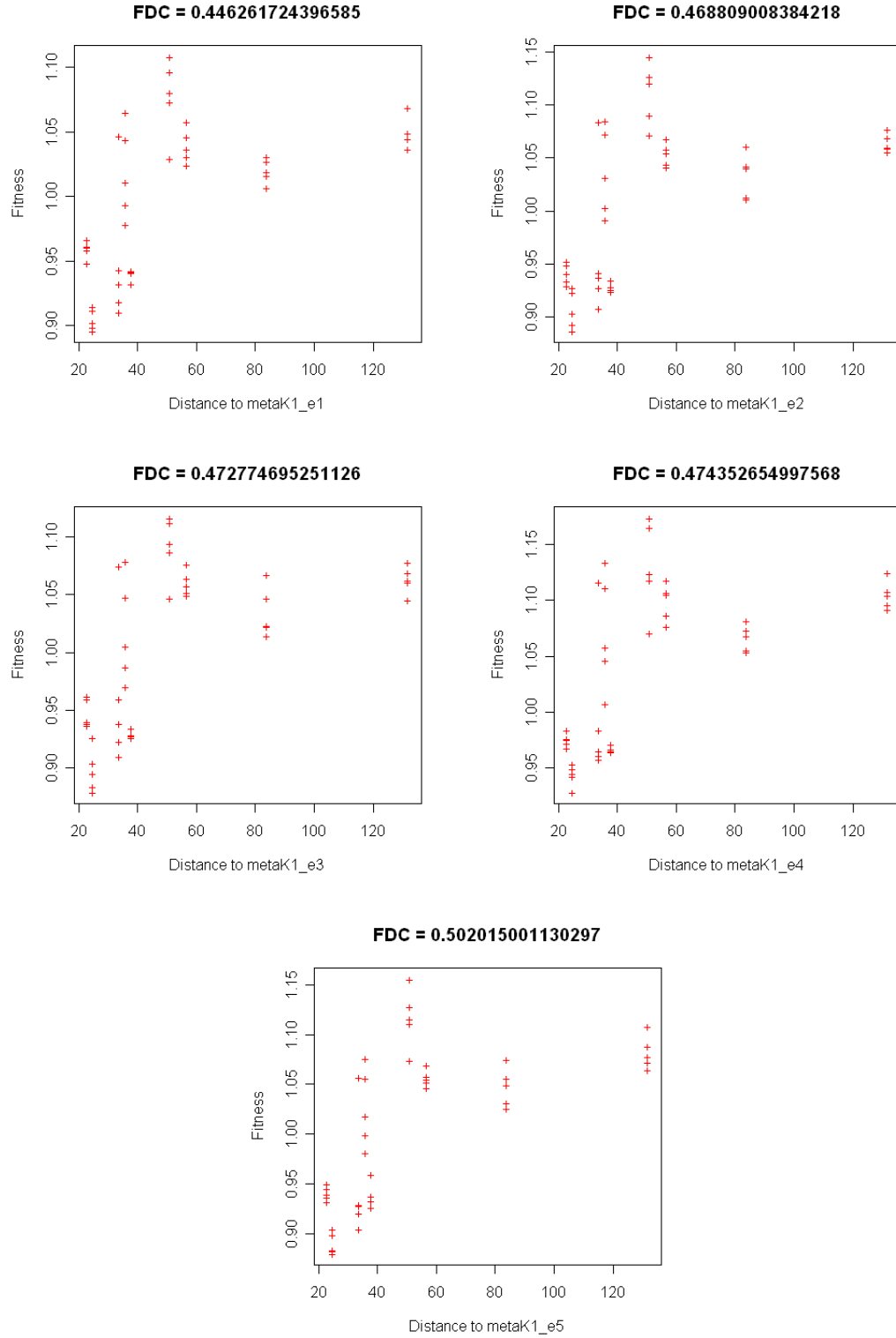
FIGURE H.4: *Graphics of the resultant scatter plots and correlation coefficients for the group **d** of Meta-automaton CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*

FIGURE H.5: *Graphics of the resultant scatter plots and correlation coefficients for the group **e** of Meta-automaton CA showing that the USM has a significant correlation with the genotype of the spatio-temporal behaviour pattern.*
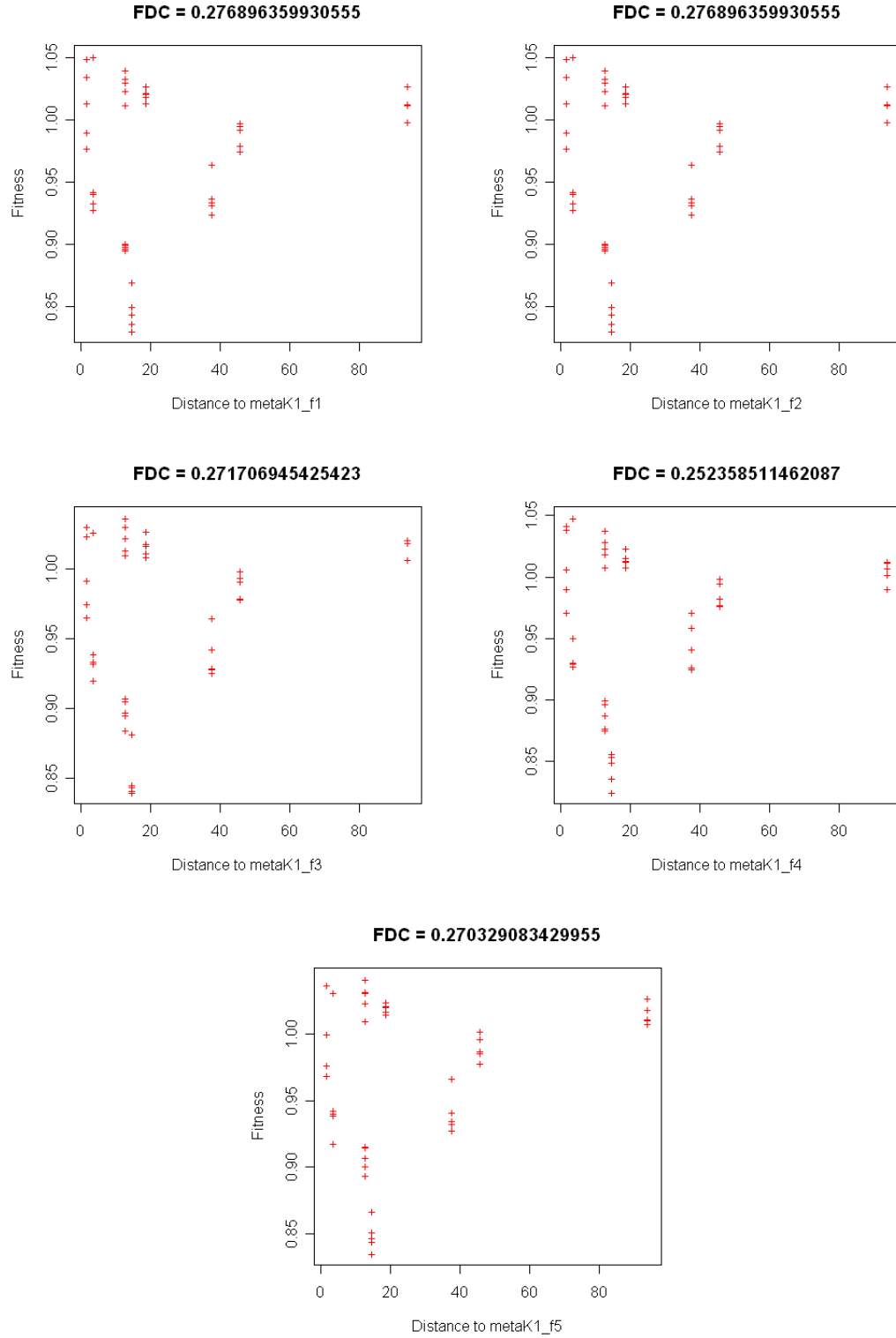
FIGURE H.6: *Graphics of the resultant scatter plots and correlation coefficients for the group* **f** *of Meta-automaton CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
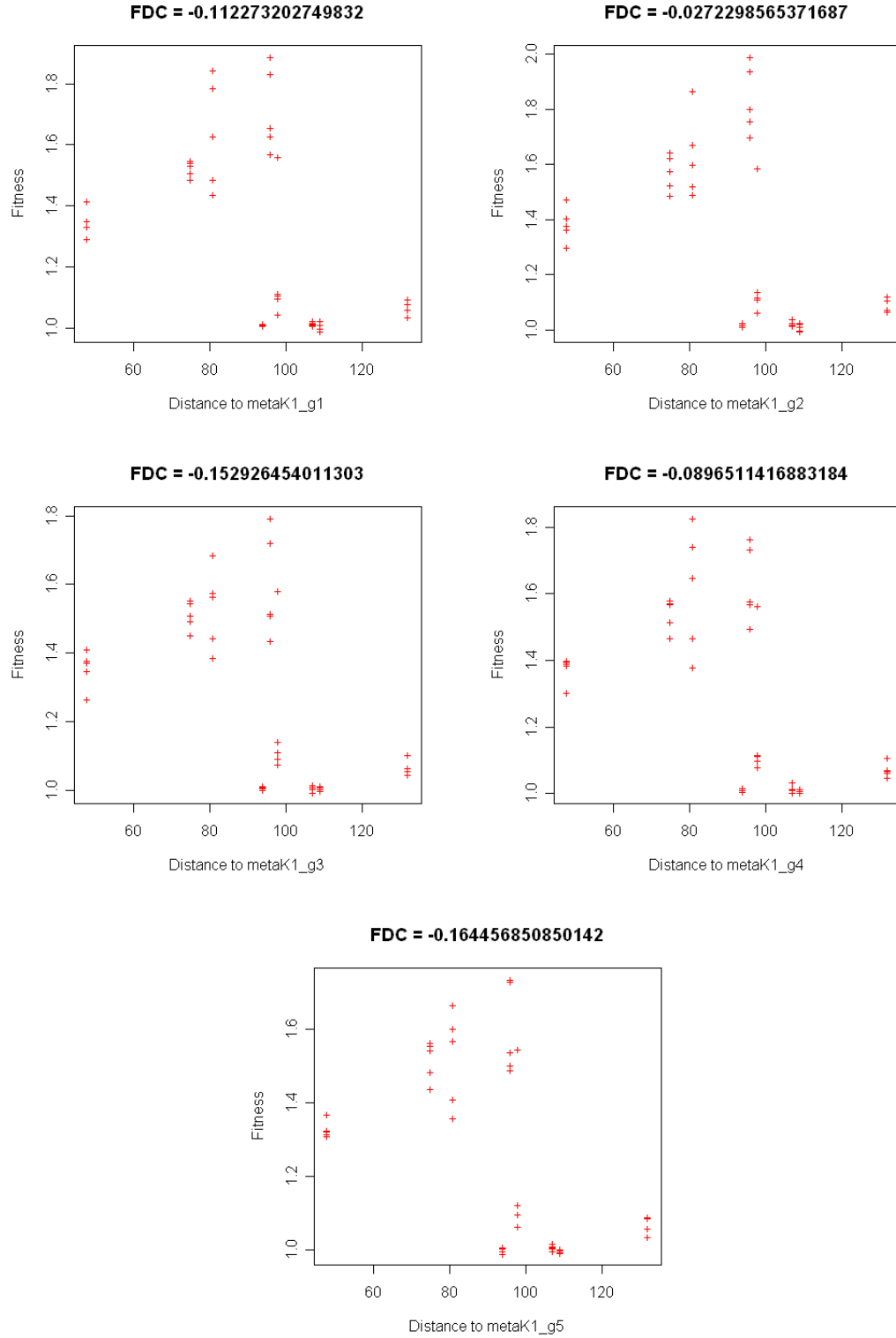
FIGURE H.7: *Graphics of the resultant scatter plots and correlation coefficients for the group **g** of Meta-automaton CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
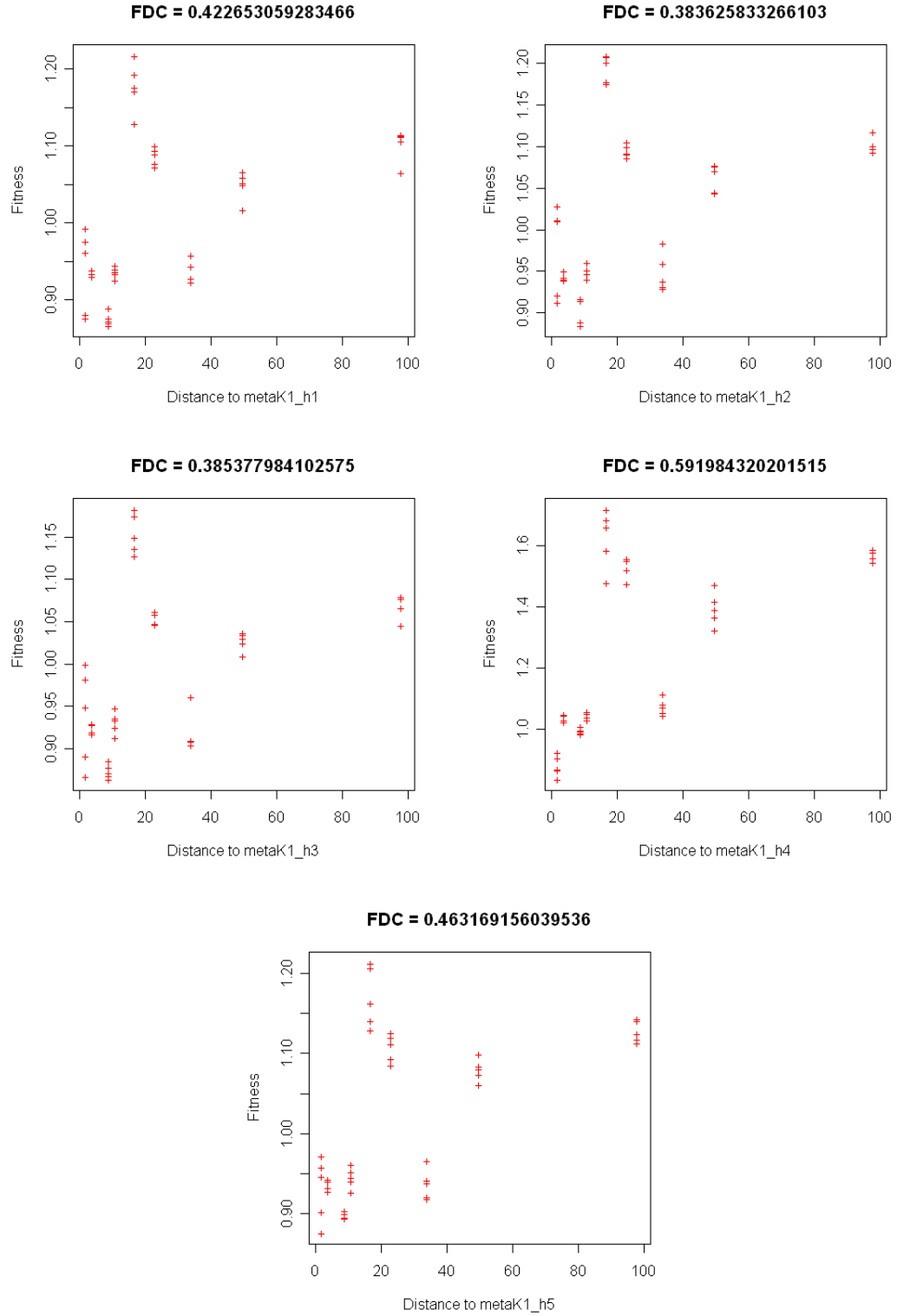
FIGURE H.8: *Graphics of the resultant scatter plots and correlation coefficients for the group* **h** *of Meta-automaton CA showing that the USM has a significant correlation with the genotype of the spatio-temporal behaviour pattern.*
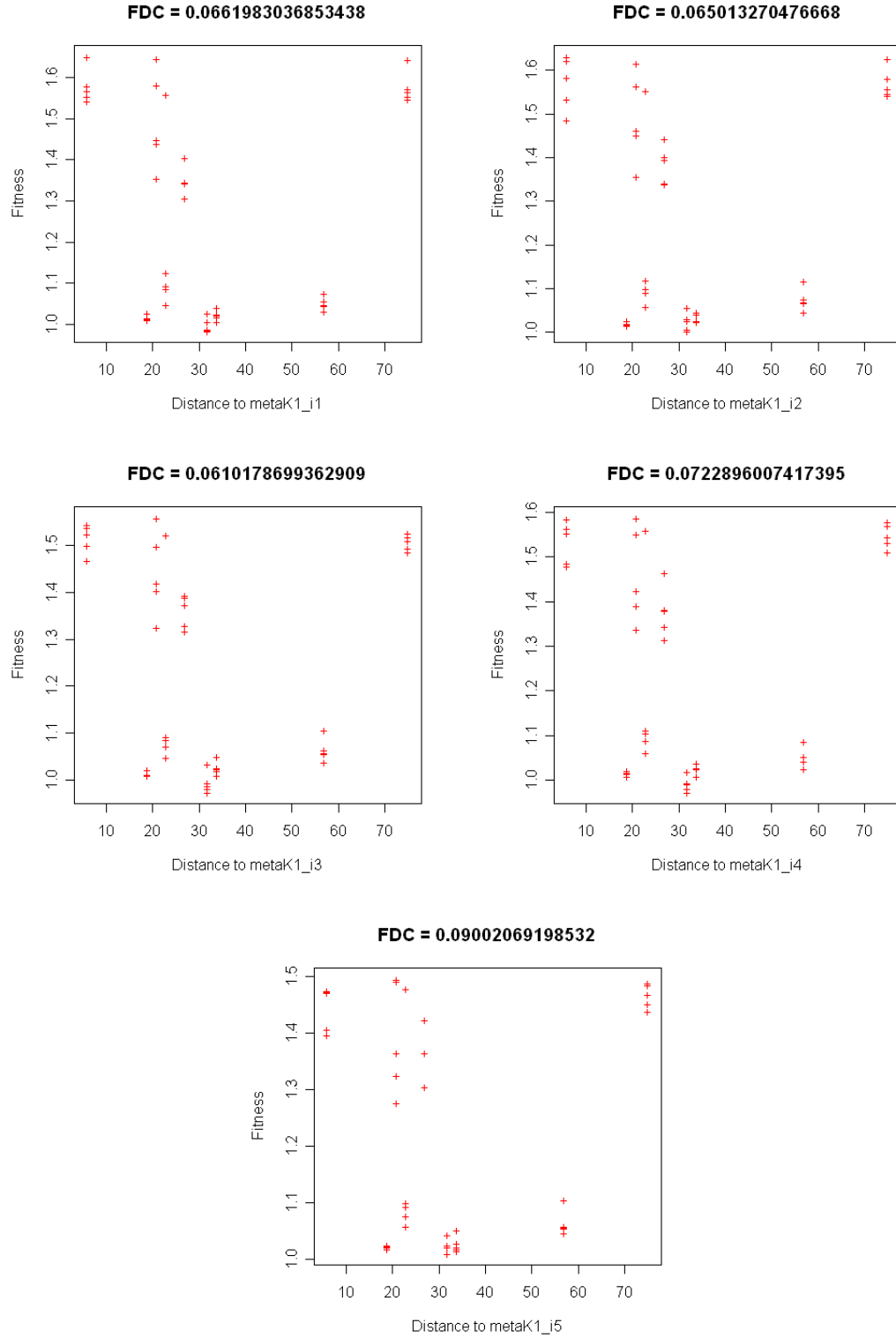
FIGURE H.9: *Graphics of the resultant scatter plots and correlation coefficients for the group $\boldsymbol{i}$ of Meta-automaton CA showing that the USM has a low correlation with the genotype of the spatio-temporal behaviour pattern.*
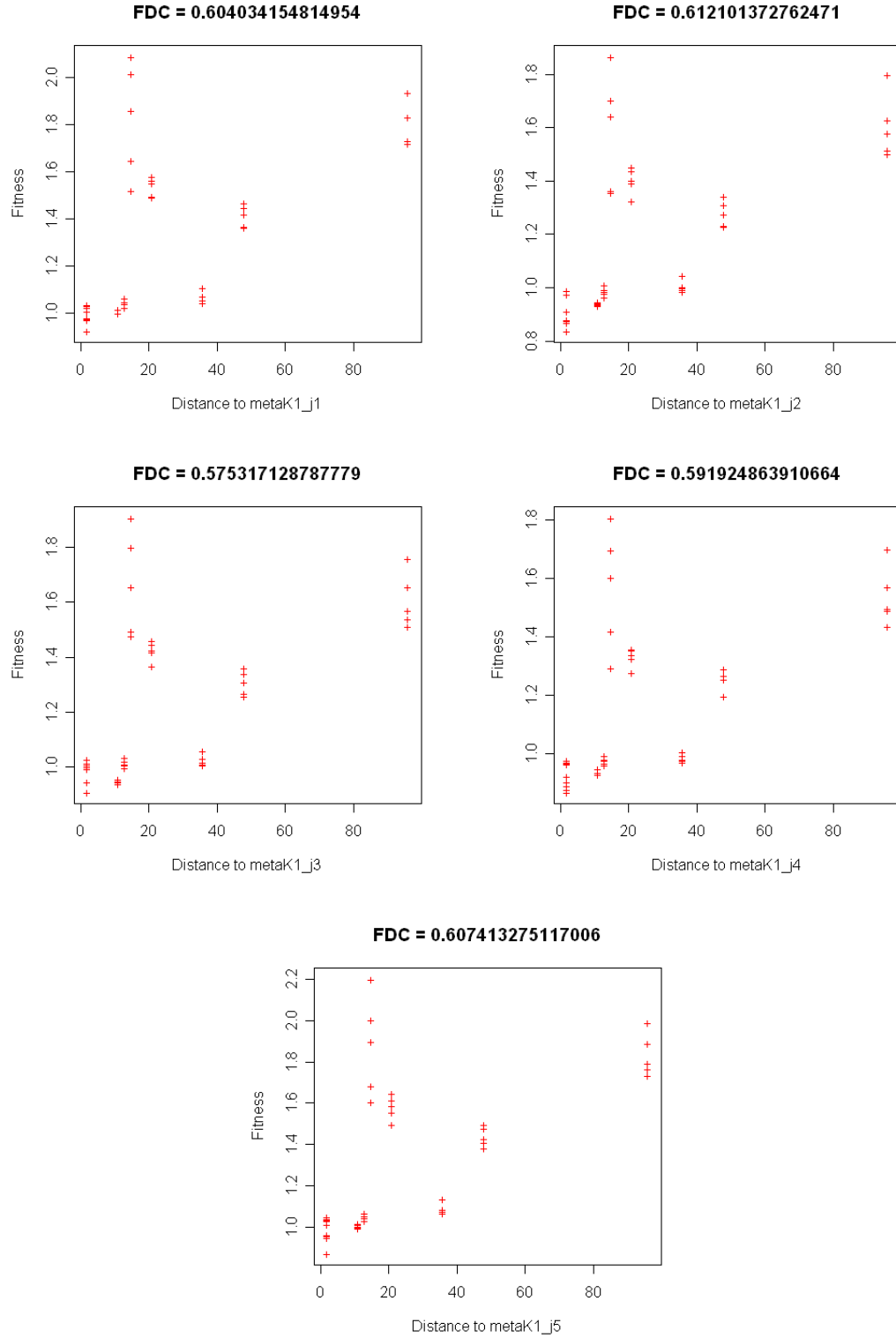
FIGURE H.10: *Graphics of the resultant scatter plots and correlation coefficients for the group **j** of Meta-automaton CA showing that the USM has a significant correlation with the genotype of the spatio-temporal behaviour pattern.*