Common metrics for cellular automata models of complex systems

by

C.W.D. Johnson

A Doctoral Thesis

Submitted in partial fulfilment
of the requirements for the award of

Doctor of Philosophy
of
Loughborough University

27 October 2015

Abstract

The creation and use of models is critical not only to the scientific process, but also to life in general. Selected features of a system are abstracted into a model that can then be used to gain knowledge of the workings of the observed system and even anticipate its future behaviour. A key feature of the modelling process is the identification of commonality. This allows previous experience of one model to be used in a new or unfamiliar situation. This recognition of commonality between models allows standards to be formed, especially in areas such as measurement. How everyday physical objects are measured is built on an ingrained acceptance of their underlying commonality.

Complex systems, often with their layers of interwoven interactions, are harder to model and, therefore, to measure and predict. Indeed, the inability to compute and model a complex system, except at a localised and temporal level, can be seen as one of its defining attributes. The establishing of commonality between complex systems provides the opportunity to find common metrics. This work looks at two dimensional cellular automata, which are widely used as a simple modelling tool for a variety of systems. This has led to a very diverse range of systems using a common modelling environment based on a lattice of cells. This provides a possible common link between systems using cellular automata that could be exploited to find a common metric that provided information on a diverse range of systems. An enhancement of a categorisation of cellular automata model types used for biological studies is proposed and expanded to include other disciplines. The thesis outlines a new metric, the C-Value, created by the author. This metric, based on the connectedness of the active elements on the cellular automata grid, is then tested with three models built to represent three of the four categories of cellular automata model types. The results show that the new C-Value provides a good indicator of the gathering of active cells on a grid into a single, compact cluster and of indicating, when correlated with the mean density of active cells on the lattice, that their distribution is random. This provides a range to define the disordered and ordered state of a grid. The use of the C-Value in a localised context shows potential for identifying patterns of clusters on the grid.

Acknowledgements

I would like to thank my two supervisors. Firstly to Prof. David Parish for giving me the opportunity to carry out my research in the High Speed Networks lab. Secondly to Prof. Ron Summers who brought this opportunity to my attention and over the last few years has given me support and guidance that has been instrumental in the completion of this thesis.

I would also like to thank all my colleagues in the HSN lab. But a special thank you must go to Dr. John Whiteley for his early encouragement and for introducing me to LaTeX and Perl. The support from Dr. Konstantinos Kryiakopoulos has been invaluable, along with that of my fellow PhD researcher, Dr. Francisco Aparicio Navarro.

My friends and family have been extremely supportive and understanding over recent years and I thank them all for putting up with me, especially during the writing up process. But I have to give a huge thank you to my wife Glenys, who has supported and encouraged me throughout this process. Without her understanding, tolerance and selfless support I would not have been able to contemplate undertaking this task, let alone finish it.

In memory of my brother-in-law, Stephen Fredrick Joseph Woodhouse

29.01.1952 – 23.12.2013

with whom I was planning to celebrate the completion of this thesis.

Contents

List of Figures

List of Tables

Glossary

(M,R)	metabolism repair.
1D	one dimensional.
2D	two dimensional.
3D	three dimensional.
ADP	active network date package.
AN	active network.
ANode	active network node.
BBR	bounding box ratio.
CA	cellular automata.
cAMP	cyclic adenosine monophosphate.
CAS	complex adaptive system.
COAST	Complex Automata Simulation Technique.
CRBN	classical random boolean network.
csv	comma-separated values.
CxA	complex cellular automata.
DARBN	deterministic asynchronous random boolean network.
DGARBN	deterministic generalised asynchronous random boolean network.
ECA	elementary cellular automata.
ECM	extracellular matrix.
GARBN	generalised asynchronous random boolean network.

GoL	game of life.
GST	general system theory.
H	threshold value.
LAN	local area network.
MUSCLE	Multi-scale Coupling Library and Environment.
MxRBN	mixed-context random boolean network.
N_s	spatial noise.
N_t	temporal noise.
p_A	agitation rate.
p_c	percolation threshold or critical percolation point.
p_E	emission rate.
p_T	transmission rate.
p_value	probability value.
PDE	partial differential equation.
R-theory	relational theory.
RBN	random boolean network.
ROA	random order asynchronous (also known as *random new sweep*).
RSA	random selection asynchronous (also known as *uniform choice*).
SO	self-organisation.
SSM	scale separation map.

Chapter 1

Introduction

1.1 Background context

Models are used across a range of systems and for a number of purposes. They can be used to show how a system will work, or how something will look. A model can be used to test the suitability or limitation of, for example, a new material or a design modification. In physics the emphasis is on a model being quantitative, with mathematics used to define the quantitative relationships. The output of such models can be compared with real systems. In this way a model can also be applied as a method of hypothesis of how a system works (deductive), or will work (predictive). As a system becomes more complicated it becomes necessary to abstract key features of the system, rather than model the entirety. This process of abstraction can be viewed as a practical form of reductionism used to simplify the complicatedness, before applying reductionism to ascertain how a system fits together and functions. Such reductionism has proved the backbone of scientific advancements made over the centuries since Newton.

But some sciences, such as biology, have areas that are not so susceptible to the use of mathematical models.

> "[A]ll biological systems are based on the same elemental matter as everything else, so why can't physics and chemistry fully explain biology? A significant difference between biology and more fundamental sciences is that in biology elementary particles combine to form 'complex agents' – machines that perform tasks – and the behavior of these agents is often difficult to capture mathematically."

[Tamulonis, 2013, p.7]

Complex systems present severe challenges for modellers and reductionists. They are, by their very nature, extremely hard, if not impossible, to de-construct, analyse and understand. They often exhibit unexpected high level structures, sometimes referred to as 'emergence', although the actual origins and definition of such emergence are still widely disputed. The greater the level of abstraction used to model complex systems the more the process moves towards simulations where the rules used should not be seen as a direct expression of the rules governing the complex system. Instead they represent potential insights into the working of the system.

Indeed, it could be held that everything observed is in fact a model or abstraction of the real world or a mental model that influences how we interact with the world [Forrester, 1971; Rosen, 1991; Senge, 2006].

> "Each of us uses models constantly. Every person in his private life and in his business life instinctively uses models for decision making. The mental image of the world around you which you carry in your head is a model. One does not have a city or a government or a country in his head. He has only selected concepts and relationships which he uses to represent the real system. A mental image is a model. All of our decisions are taken on the basis of models. All of our laws are passed on the basis of models. All executive actions are taken on the basis of models. The question is not to use or ignore models. The question is only a choice among alternative models."
>
> [Forrester, 1971]

Mental models are also used in the process of anticipation where an organism anticipates how something works by correlating the action of the model of a previously observed system with a newly encountered one. This ability can be seen as a key element of complex organic systems [Nadin, 2012; Rosen, 2012], while it can be seen in the *tagging* and *internal models* included in the basic elements of complex adaptive system (CAS) [Holland, 1995]. General Systems Theory seeks to identify system isomorphisms and uncover isomorphic laws. In a similar way the forming and utilising of mental models can be transposed onto the concept of sharing a formal model between analogous systems [Rosen, 1991], thus suggesting the possibility of identifying some common metrics between systems through their shared model. This would be useful in any search for common indicators in complex systems, such as for when it moved from or to a high-ordered state. This search can be started by first selecting a commonly used modelling environment.

The use of computers and computational models has greatly facilitated the simulation of not only mathematical based models, but also complex systems capturing the interaction of multiple elements. Parts of the latter may involve mathematical equations, but the overall working of the model is not open to a single mathematical definition. Thus, the predictions that can be extrapolated from a mathematical model have to be induced with computational models through the running of the simulations [Tamulonis, 2013]. Lattice based models, or cellular automata (CA), have been used as the basic framework for many computational models studying complex systems, including within biology [Chopard et al., 2002; Ermentrout and Edelstein-Keshet, 1993; Hegselmann and Flache, 1998; Kroc et al., 2010; Mitchell, 1996; Packard and Wolfram, 1985; Schiff, 2007; Tamulonis, 2013; Torrens, 2000; Wolfram, 1984b, 1994, 2002]. The CA lattice of cells, or grid, is usually one, two, or three dimensions, with each cell in one of a finite number of states. The rules determining the new state of a cell uses the current state of its designated neighbours. The cells on the grid are updated in discrete time steps; traditionally this has been a synchronously update of all the cells, although asynchronous updating schemes that do not always update all the cells within a time step have become commonplace. Two dimensional CA (2D CA) models provide a better visual tool than 1D CA when representing 3D events such as traffic or pedestrian flow, or the development of skin pigmentation. Although 3D CA mirrors the world we live in, this simpler realisation of 2D CA, coupled with the greater ease of programming and running 2D simulations, make it a widely used format. Applied CA research has been conducted with specific systems in mind, although there has been a move towards providing a formal setting for CA modelling. This use of CA can be seen in attempts to bring the concept of emergence into the design of engineered systems methodology [Fromm, 2006]. Ermentrout and Edelstein-Keshet [1993] proposed three categories of 2D CA models within biology, (deterministic, particle and growth). If these types are applied to systems outside the discipline of biology, then there is the possibility of seeing analogy between diverse systems modelled by the same type, if not across the types. Such analogy would increase the chance of discovering common metric or means of measuring the output space of lattice-based models from diverse system domains.

1.2 Motivation

While models and simulations of complicated or complex systems are only likely to give an abstracted and partial view of a system, they can have the benefit of showing the state of the system within a localised context over a set period of time. The modeller chooses a specific modelling technique and the abstractions

to be modelled with a view of what they are hoping to observe, especially when the model is designed as a form of monitoring. Cell-based models, including two dimensional CA, have become very popular and are used to model aspects of different types of non simple systems. Although they are not as closely coupled with real systems in the way that some models are, nor are they a panacea for all modelling, 2D CA do model diverse systems using the same technique and using a common output format. How the output space is analysed does depend on what is being modelled and what is being sought. But the analysis of the cell-based output space has the potential of giving metrics that could be applied to the changing state of the output space over time for a diversity of systems. Various measuring techniques have been used, but they have generally been employed to identify specific, sought after patterns or behaviour within the output space, rather than a common metric of the overall state of the space. Those measures that have been considered and proposed have their limitations. The aims and objectives of this thesis listed below outline the approach being taken in this thesis to propose a metric that provides a common means of identifying the state of a 2D CA output space.

Aims:

1. to extend the three classes of biologically 2D CA outlined in Ermentrout and Edelstein-Keshet [1993] to include non-biological motivated models and models using probability in either or both their updating schemes and rules (enhancement of existing work)

2. to design, implement and evaluate a simple method based on connectedness between the active cells to identify the state of 2D CA grid space (novel work)

3. to identify a general means of identifying the changing state of a 2D CA grid space across different domain models (novel work).

Objectives:

1. *to classify systems using 2D CA modelling by means of the type of model used and the characteristics of the observed output (aim 1)*

   The behaviour and output expected from each of the three types of CA models of biology can be used to distinguish them. Deterministic automata focus on the synchronous, non probabilistic changing state of each cell on the grid; particle automata deal with agents' movement across the grid; and

growth automata track the gradual spread and population of cells across the grid. Non biological extensions to deterministic automata, can include forest fires, browning motion and various power-law models, such as earthquakes. But there are also a large group of automata that use probability in an asynchronous updating scheme or in the rules used to update a cell. This group, labelled in the thesis as "randomised automata", extends the modelling of systems traditionally covered by deterministic automata and needs to be viewed as a separate type of CA. Growth automata, can encompass models of snow crystals, snow avalanche and slime mould. Particle automata arise out of the lattice gas models, which feature the changing state of the particles when they collide. But an increasing number of particle or *agent* based automata can be classified as non collision models, such as pedestrian and vehicle and swarm models. Within the non collision based models there are basic flow models, threat models, (including evacuation where humans can exhibit herd behaviour), obstacle and navigation models. While there is a temptation to split this type into two types, particle and agent, the similarities are strong enough to keep them as one type.

2. *to establish any analogy between different systems using 2D CA models (aim 1)*

   The abstraction of a system into a model simulated by a 2D CA can be classified by the type of CA model used. The similar grid behaviour and output expected within each type would indicate that a level of analogy exists between the models of the different systems within each type, which in turn has the possibility of some common metrics associated with the output of the models.

3. *to develop the basis of a measuring technique that can be used with 2D CA output from different domains and with different expected output (aim 2)*

   Visualisation of the CA model is a key part of its appeal as a modelling technique. But other measuring techniques have been employed, including density measures, the tracking of agent movements, the measurement of 'entropy' and the identification of patterns and clusters. Clustering algorithms are utilised both in the modelling rules used in the running of the CA and in determining the presence of clusters of a specific type of cells. Any generic metric needs to move away from the identification of specific features and instead focus on the changing state of the CA grid as a whole.

4. *to evaluate the suitability of the measurement technique against various 2D CA output scenarios (aim 2)*

Any metric has to first distinguish between different states of the CA grid, such as between whether the active cells are randomly spread across the grid or show any level of clustering. This can be related to the density and number of clusters on the grid, as well as any pattern created on the grid or the formation of just a single cluster. The visualisation enabled through the CA modelling technique makes it simple to create a range of scenarios to test against. The new proposed technique needs to be compared to other existing measurements of 2D CA output.

5. *to construct models to simulate and measure the output of three types of CA models, deterministic, particle and randomised (aim 1, 2 and 3)*

   The use of scenarios can be used to test and tune any metric. The purpose of this objective is to test the metrics against the changing state of the output of actual models of three different types of 2D CAs. The changing state illustrates the movement to or from a random distribution of active cells on the grid to one that has some discernible order or grouping of the active cells.

6. *to evaluate the effectiveness of the measure and metric (aim 2 and 3)*

   The effectiveness of any metric is not only how well it works, but also how it compares to other metrics.

7. *to establish if the new metric provides the basis of tracking changes in the overall state of the CA output from different domain models (aim 3)*

   The final objective is to see if the new metric can be used to track the state of the lattice of different domain models as it changes between a random and a highly structured state.

1.3 Plan of thesis

The closing section of this chapter outlines the plan of the thesis (*see also Figure 1.1*).

In chapter 2 some of the ideas on the central topics of the thesis are reviewed, starting with the classification of systems and the role of reductionist and non-reductionist approaches in the analysis of systems. This leads into an exploration of isomorphism, analogous systems and modelling. The epistemic value of computer simulation is then considered. This is followed by an explanation of the computational modelling technique of cellular automata (CA) and examples of

the variety of systems modelled with it are outlined. The types of CA defined in [Ermentrout and Edelstein-Keshet, 1993] are reviewed and a couple of enhancements proposed. The task of measuring 2D CA output is reviewed prior to the summary of the chapter.

The simulation methodology adopted and methods used are outlined in chapter 3. The four approaches to the measuring of 2D CA output space are examined, including connectedness and the history of its concept and previous manifestations and usage. The means of testing them with scenarios is explained and the three types of CA models used are described. The four metrics are then defined in formal terms.

In chapter 4 the scenarios and models used are outlined and an overview of the program suite created to run and analyse the scenarios and models is given. Then the three types of scenarios selected to test the metrics are explained. This is followed by details of the design of each of the three models and their implementation and validation against the original work from which they were drawn. A test plan is presented for each model.

The results from the scenarios and from the simulations of the deterministic, randomised and particle models are analysed in chapter 5. This incorporates a comparison of the performance of the three existing metrics and the connectedness method proposed in this thesis.

In chapter 6 there is a discussion of the impact and relevance of the findings that are highlighted throughout the thesis, beginning with the enhancement of the classification of CA types. This is followed first by an evaluation of the scenarios and models used, and then by a review of modelling and the role of the three models used within the context of the simulation methodology. Next the performances of the four metrics are considered; this leads into an evaluation of the connectedness metric proposed in chapter 3. The chapter concludes with a reappraisal of the modelling process and the feasibility of a common means of evaluating the state of a 2D CA output space across different system domains.

The main body of the thesis ends with chapter 7, where the objectives from chapter 1 are reviewed with reference to the research carried out in the thesis. This is followed by a statement on the contribution made by the thesis. The chapter ends with suggestions for future work.

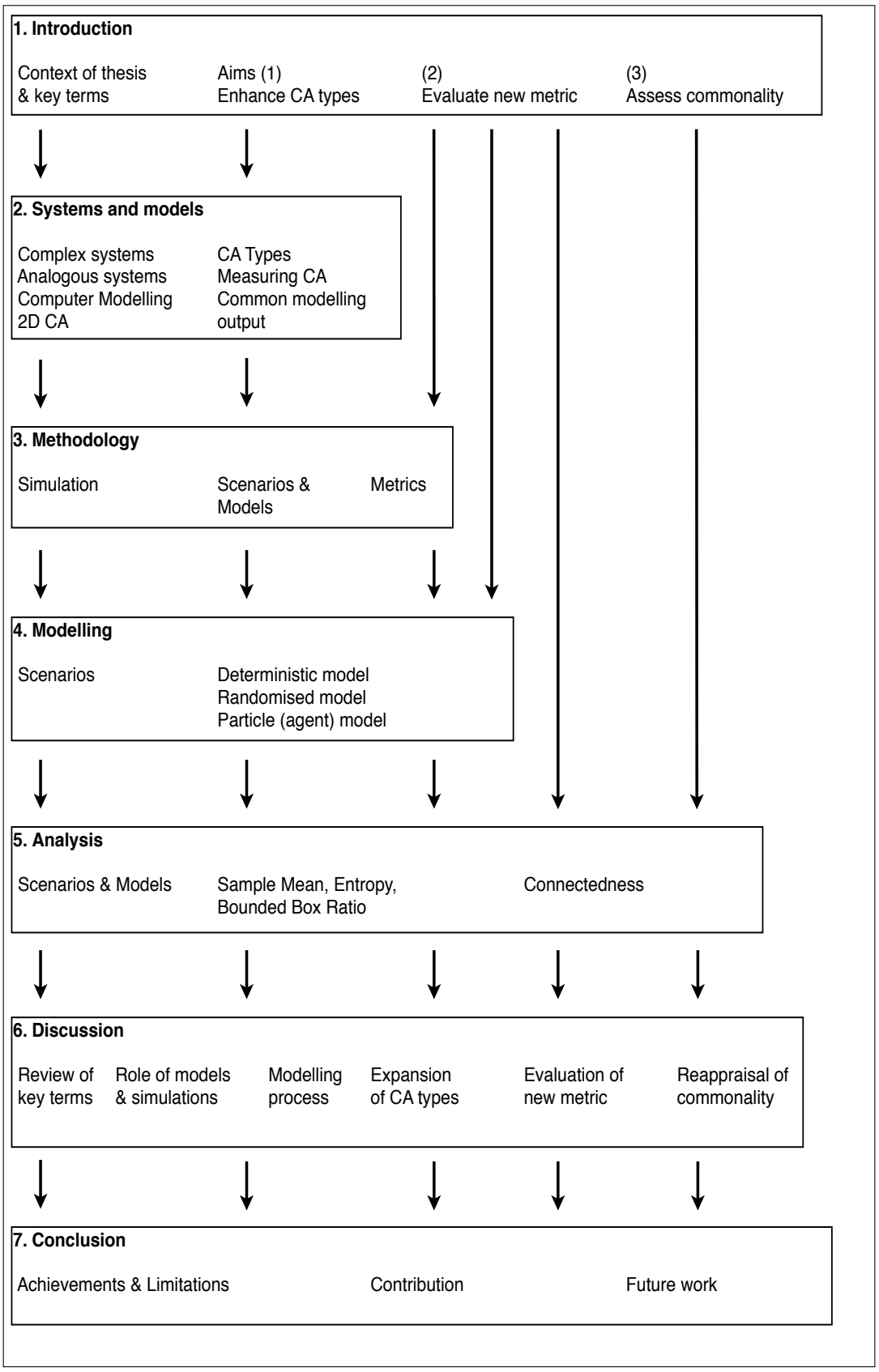


Figure 1.1: An outline of the thesis

Chapter 2

Systems, models and simulations

2.1 Introduction

Systems and models form an integral part of the scientific method. The operation of a system, comprising of two or more interacting elements within a defined boundary, is observed and a hypothesis formed to explain the observed phenomenon. The system can consist of subsystems that interact and the relationship between its boundary and its surrounding environment can be either closed, isolated or open. Models, based on the abstraction of features of the system perceived to be pertinent, are used to refine the hypothesis and provide the basis of what an experiment then needs to validate. In this formal environment the experiment provides epistemic value, rather than the model. The model itself can range from a partial extraction of a real life system, to a representation of the target system governed by a globally defined mathematical equation.

In everyday life the experience and information gained from observing the behaviour of one 'system' can be used in mental models to predict how a newly encountered system might behaviour. The effective mapping of similarities or analogies between source and target systems is also fundamental to how a simulation, (where a model is 'run'), provides any epistemic value, opaque or otherwise. Simulations provide a representation of the target system, but unlike an experiment they have no material link with it. In cases where the interactions within a system are such that its behaviour cannot be reduced to a mathematical equation, then experiments can be difficult to construct and run. A recourse to understanding and unravelling such 'complex' systems is to simulate their behaviour. This can be seen in the computer simulations used in astrophysics and climatology. While such large simulations are testament to the power of modern computers, a considerable amount of computer simulation is conducted at a much simpler

and a much less resource consuming level. Cellular automata (CA) provide such a framework. A cellular automaton is made up of a lattice of cells. Each cell is in one of a finite number of states, and the lattice can be in any finite number of dimensions. The cells are updated in discrete time steps using the values of a cell's neighbours to determine the value of that cell at the next time step. CA have been used to model complex behaviour [Kroc et al., 2010; Wolfram, 1984a], including self organisation [Fatès et al., 2008; Wolfram, 1994]; a high ordered state that is seen as emergent as it cannot be attributed directly to the known characteristics of the elements within a system, in this case the cells of the cellular automaton, their state and the local rules governing their update. Emergence within a system is seen by many as a key characteristic of a complex system.

A 'dumb hole' uses nearly perfect fluids, such as Bose-Einstein condensate, to create a sonic black hole that is theoretically similar to a gravitational black hole [Dardashti et al., 2014]. This example of an analogous simulation shows how a model of one system can be used to understand another system that is extremely difficult to observe. This similarity in the behaviour of two diverse systems is demonstrated through a shared model. The running of the simulation produces output that is then shown to exhibit the common behaviour. Computer simulations have a problem in that the operation of the computer and the program governing the simulation are opaque. There is always the possibility that anything observed is an artefact, rather than a true representation of the target system. Despite this, many CA simulations utilise abstracted characteristics of real life systems to solve problems in different system domains; such as chemotaxis for modelling pedestrian movement [Schadschneider, 2001] or the gathering of agents [Fatès et al., 2008]. In this way an analogy is being drawn between two systems. But like analogous simulations, these computer simulations use measurements that are obviously specific to the simulation and the motivation driving the simulation.

Two-dimensional CA have been used as simple simulations of many different systems (*see subsection 2.6.5*). Each simulation shares a similar modelling framework, including the visual output space. But even if a varied selection of CA simulations are grouped into different types of CA, (such as in [Ermentrout and Edelstein-Keshet, 1993]), within each type there is no guarantee that one measure used to analyse the output of one cellular automaton could be used across that type of CA. The identification of a common measure across the CA output from the simulation of models of diverse systems would be of interest, not least in establishing a common indicator of the state of the output space that is independent of any measure specific to a simulation. This forms the central part of the thesis

and is introduced in chapter 3.

This chapter is split into seven further sections. A review of how systems have been classified as *simple, complicated* and *complex* is carried out in section 2.2. Special attention is given in subsection 2.2.3 to what makes a system complex, including emergence in subsubsection 2.2.3.2. Comparisons are explored between complex systems and both chaotic systems in subsubsection 2.2.3.3 and complex adaptive systems in subsubsection 2.2.3.4. The subsection on complex systems concludes with a working definition of a complex system in subsection 2.2.4.

Although reductionism continues to be an extremely useful and effective way of analysing isolated and closed systems, it does have limitations when facing the openness associated with complex systems. These limitations are considered in section 2.3 and then a non-reductionist view of systems is presented in section 2.4. The latter looks at similarities between second order cybernetics of von Foerester and Robert Rosen's relational theory. This features (a) the role of the observer in subsection 2.4.1, (b) how the interactions between the system's constituent parts is explained in terms of relations, anticipation and feedback in subsection 2.4.2, (c) the importance of context when modelling in subsection 2.4.3, and (d) role of isomorphism and analogous systems in modelling in subsection 2.4.4. .

The latter section concludes with the idea of looking for some analogy, and thus common indicator, in a common modelling technique and the measurement of the output space. This leads into the appraisal of computer simulation in section 2.5, including their epistemic worth compared to experiments in subsection 2.5.1 and their use and epistemic value as opaque thought experiments in subsection 2.5.2. A brief outline of the advantages of computer simulation is given in subsection 2.5.3. This is followed by a review of two types of simulations in subsection 2.5.4; those based on global mathematical equations and those using local rules. The latter features cellular automata, which are explored in section 2.6.

The thesis uses two-dimensional CA for its simulations (see chapter 4). The use of different dimensions is considered in subsection 2.6.1 and two-dimensional CA are explained in subsection 2.6.2. The impact of synchronous and asynchronous updating schemes is evaluated in subsection 2.6.3. The review of CA closes in subsection 2.6.5 with an outline of some of the different areas of investigation that have been modelled by CA. The areas of *physics & chemistry, social science, biology & medicine, earth science, traffic flow* and *general applications* are then regrouped using the three broad categories of CA labelled in [Ermentrout and

Edelstein-Keshet, 1993] as *deterministic, particle* and *growth automata*. A fourth category of *randomised automata* is then proposed and a revised table of CA simulations is presented.

The penultimate section, section 2.7, looks at the issues of measuring different types of CA. In subsection 2.7.2 a outline is given of how various entropic measurements have been used to measure for emergence in CA, such as by tracking any loss of randomness. The section ends with an overview of Kolmogorov Complexity in subsection 2.7.3. This offers a relatively easy way of measuring an output space if it can be represented in a string format. The chapter closes with a summary in section 2.8.

2.2 Systems - a linear view

The definition of *system* in the Oxford English Dictionary includes:

> "1. An organized or connected group of objects
>
> 2. A set or assemblage of things connected, associated, or interdependent, so as to form a complex unity; a whole composed of parts in orderly arrangement according to some scheme or plan; rarely applied to a simple or small assemblage of things (nearly = 'group' or 'set')
>
> 3. In various scientific and technical uses: A group, set, or aggregate of things, natural or artificial, forming a connected or complex whole."
>
> Simpson et al. [1989]

The key feature is the connectedness of interrelated and interacting components, grouped, by implication, within a defined boundary. A basic system is generally referred to as *simple*. The terms *mechanistic* and *mechanical* are also used. From a reductionist's stance systems are on a linear sliding scale of simple to complicated to complex, (*see Figure 2.1*).

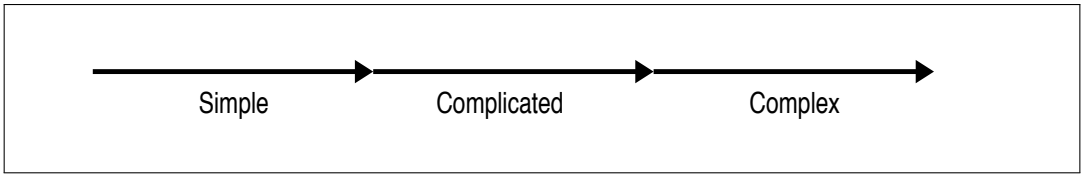


Figure 2.1: Linear scalar of systems

In this reductionist or mechanistic view a system's position on the scale is dependent on our knowledge of how it works. The position can change as our under-

standing grows or, if the system boundaries expand, lessens. But the definitions and demarcation between simple, complicated and especially complex can be confusing.

2.2.1 Simple systems

Simple systems conform to the reductionism ideal. They are reducible to their component parts, from which their function can be ascertained. These parts can then be reassembled to reproduce a working system. Their alternative titles of *mechanistic* and *mechanical* signifies the way that they are generally constructed systems, built with a discernible purpose in mind. In this way the models associated with simple systems range from detailed design drawings and scale models to modelling of the performance of the system as a whole, or of component parts. The latter is often carried out as part of a system's test programme, such as stress tests of an aircraft wing in a wind tunnel.

A simple system is often an artefact, constructed with a purpose in mind. Consequently, the relational aspects of the system can be subsumed into the general attributes of the system and its components. An example of this is the relatively simple system of a combustion engine where the principal operation is the ignition of a gas so that it expands to apply force to a component of the engine, for example a piston. Both the process of causing the gas to expand and the working of the piston can be tested separately. It is only when they are integrated together that the relationship between the chemical reaction of the gas and the machined structure of the piston results in the creation of mechanical energy. This mechanical energy is not part of the individual components, but a product of their interaction as a result of their engineered relationship. This relationship in a simple system is usually taken for granted as it is designed and explainable.

The various models of a simple system can lead to the sharing of new techniques with other modelling processes and systems, such as new construction methods, or uses for new materials. Measuring methods can also be shared across models and actual systems. Load bearing factors, CO_2 omissions, fuel consumption, engine power, and heat efficiency are just a few examples of measurements that can be used in multiple models and systems. Such measures can be used with a model to monitor the performance of a system, or to analyse production performance issues. Consequently, a key feature of a simple system is the ability to measure, quantify and predict its actions.

2.2.2 Complicated systems

A complicated system can be seen as a larger simple system; still predictable and following discernible rules, but with more components. The components can even contain one or more simple systems. In his use of broken symmetry to reappraise reductionism and point out its limitations, Anderson et al. [1972, p.396] stated that "[a]t some point we have to stop talking about decreasing symmetry and start calling it increasing complication". This follows the concept that as things grow and move up the scales, then the interaction will change the symmetry and potentially involve new laws. As with a simple system, a complicated system can be reduced to and reassembled from its parts [Ottino, 2004], and as he elaborates elsewhere:

> "*Très compliqué* is used to describe the most elaborate mechanical watches. They are, as their French name implies, complicated. A Star Caliber Patek Phillipe has 103 pieces. A Boeing 747-400 has, excluding fasteners, 3106 parts. In complicated systems, parts have to work in unison to accomplish a function. One key defect (in one of the many critical parts) brings the entire system to a halt. This is why redundancy is built into designs when system failure is not an option (e.g., a nuclear submarine)."
>
> [Ottino, 2003, p.292. author's italics]

This view of a complicated system involves not just the ability to reduce and rebuild a system, but also an understanding of the integrity of the relationship between the components of the system. Kier and Witten [2005, p.9] concur that "Complicated systems also have the property that one key defect can bring the entire system to its knees". Bak [1999] observes that large dynamic systems are so complicated that it is impossible to construct a full-sized model. Also, he contends that it has to be modelled as it is too complicated to collect and analyse data from real life. "In the final analysis, the quality of the model relies on its ability to reproduce the behavior of what it is modeling!" [Bak, 1999, p.42]. Furthermore, he believed that better insight could be gained by starting with simple models and then building on them, rather than from an over complicated model [Bak, 1999, p.132].

The distinction between simple and complicated systems has to be more than a subjective judgement. The size, or rather number of interacting components is obviously a factor. But when considering Bak's comment above on the use of simple models, the concept of some of the components of a complicated system

being made up of interconnected simple systems could be useful. The predictability of the simple systems would make any relationship with other sub-systems or components equally predictable within the context of the complicated system. Prediction in a complicated system would be harder to achieve than in a simple one.

2.2.3 Complex systems

If simple and complicated systems are essentially on the same spectrum then on a linear sliding scale of systems so are complex systems. Consequently, they would be classified as a higher level of complicatedness. But this mechanistic view of systems would seem to be limiting, especially when biology and other natural systems are brought into consideration [Abbott, 2009; Anderson et al., 1972; Hosseinie and Mahzoon, 2011; Mikulecky, 2001; Rosen, 1991; Schrödinger, 1992]. In contrast to the potential fragility of a complicated system, adaptivity to prevent critical failure in the face of any component breaking is held as a feature of complex systems and distinguishes it from simple and complicated systems [Rickles et al., 2007]. Indeed, the range of systems that are held to exhibit complex behaviour extend past biological and non-biological natural systems to include man made and socio-economic systems [Çambel, 1993]. But it is not absolutely clear what distinguishes a complex system not only from simple and complicated ones, but also from chaotic systems and complex adaptive systems (CAS).

2.2.3.1 Linear characteristics of complex systems

If we take the assessment from subsection 2.2.1 and subsection 2.2.2 that simple and complicated systems are made up of components and relatively simple rules, and consequently are predictable, then unpredictability would be the key feature of complexity. But reductionism would see this unpredictability as a result of a lack of knowledge that could, at least theoretically, be resolved. This once again leads back to a linear scale of systems and one that is bound up in some measure with the observer's ability to understand and predict the workings of the system in focus. Edmonds [1999b] attempts to get around this 'knowledge' problem by defining complexity as an attribute of the model being used to investigate and understand a system. His comprehensive study of the syntactic measures of complexity led him to offer a general definition of complexity that could be reinterpreted in different contexts:

> "*Complexity is that property of a model which makes it difficult to formulate its overall behaviour in a given language, even when given*

> *reasonably complete information about its atomic components and their inter-relations*"
>
> [Edmonds, 1999b, p.72. author's italics]

But it could be argued that not only is the building of 'good' models a skilful task, but there is a certain amount of knowledge and experience required when selecting the attributes of a system to model.

The results of Edmonds' survey are reflected in the extensive investigation of complexity and chaos in the context of complex adaptive systems (CAS) carried out by Couture [2007a], and in the general review of complexity by Mitchell [2009]. The measures of complexity documented can generally be grouped in one of four categories:

1. more of a result of the complexity than a condition for it (e.g., processing time and resources)
2. too specific to a limited set of types of complexity (e.g., algorithmic information content)
3. too subjective (e.g., algorithmic information content; degree of hierarchy)
4. hard to actually apply (e.g., statistical complexity; entropy; thermodynamic depth)

So apart from the ability for a complex system to survive component failure, the only defining characteristic of a complex system would seem to be the unpredictability of the system, albeit maybe only temporarily. Mitchell [2009] holds that randomness and probabilities are essential properties of a complex system. But both of these can be seen as complementing unpredictability.

The ability to adapt and to self-organise is associated with complex systems, but this is an aspect of unpredictability that is aligned with a feature of complexity termed *emergence*, where the characteristics of a system could not be directly traced back to the attributes of the physical components that made up the system [Couture, 2007b; Fromm, 2005a; Prokopenko et al., 2007]. So this is again a complementing part of unpredictability. Gershenson [2005, p.3] adopts a practical notion of the theoretical aspects of self-organisation, "[a] system described as self-organizing is one in which elements interact in order to achieve dynamically a global function or behavior". This would suggest a link between the interaction of components, unpredictability, self-organisation, adaptability and emergence.

Ottino [2004, p.399] summarises that "[t]he hallmarks of complex systems are adaptation, self-organization and emergence". But there is as much debate around emergence as there is about complexity.

2.2.3.2 Emergence

The nature of emergence can be described from what could be termed a strong reductionist point of view, as a redundant term created merely to gloss over whatever we have no current causal explanation for. On the other extreme of the debate, emergence implies completely novel structures and system characteristics that 'emerge' and can not be traced back to the attribute of the components of the system. There are examples of previously emergent properties that are subsequently explained as new scientific knowledge and techniques are developed, such as in solid state physics and molecular biology [Kim, 2006]. But there are as many properties in the universe that still appear surprising and irreducible,

> "How do proteins work their wonders? Why do magnetic insulators superconduct? Why is ^{3}He a superfluid? Why is the electron mass in some metals stupendously large? Why do turbulent fluids display patterns? Why does black hole formation so resemble a quantum phase transition? Why do galaxies emit such enormous jets? The list is endless, and it does not include the most important questions of all, namely those raised by discoveries yet to come."
>
> [Laughlin and Pines, 2000, p.30]

Some supporters of emergence have separated it into an observer related feature and an actual natural phenomenon. Abbott [2006] splits the debate between functionalism and reductionism, likening his 'static emergence', which is not time dependent, and 'dynamic emergence', where emergence is observed as the model changes over time, to Weinberg's petty and grand reductionism. Walliser [2009] has a similar view, seeing epistemological emergence as covering the intended outcome of the modeller's conceptualisation of the system; while ontological emergence indicates that the emergent phenomenon actually exists as a characteristic of the macroscopic level. Hosseinie and Mahzoon [2011] argue that a better metaphysical framework (Transcendentalism) is needed to understand emergence, as reductionism and holism explain different perspectives of the functional levels and scope of a system. Ronald et al. [1999] goes so far as to put the observer at the centre of any notion of emergence; it is visualised in the mind of the observer and as such has a nebulous quality to it. The role of the observer is considered in subsection 2.4.1.

Various attempts have been made to give substance to emergence by listing its characteristics. Couture [2007b, p.73] lists (a) supervenience - dependence on existence of a lower level, (b) downward causation - the effect is not epiphenomenal, but has a causal effect on the levels below , and (c) irreducibility - it is not the aggregate of component parts. De Wolf and Holvoet [2005] place supervenience, downward causation and irreducibility in their *micro-macro effect*, *two-way link* and *radical novelty* respectively. They then propose another five: (a) coherence (pattern formation), (b) interacting parts (local interaction micro-level elements), (c) dynamical (system evolution), (d) decentralised control (self-organisation), and (e) robustness and flexibility (adaptability).

In what can be seen as a step on from characteristics, various classifications and frameworks have been proposed to clarify the nature of emergence, resulting in even more discussion. Fromm [2005b], on his part, outlined a taxonomy of four types of emergence: (a) simple/nominal emergence without top-down feedback, (b) weak emergence including top-down feedback, (c) multiple emergence with many feedbacks, and (d) strong emergence. The idea itself of weak and strong emergence has led to much debate about their definition. Davies [2006] suggests that weak emergence is where the system and its environment could be explained theoretically through reduction, but in practice needs close analysis or simulation. Bedau [2008] focuses on weak emergence, which depends on the system being open and the downward causation being diachronic; with the latter, the macroscopic can affect the microscopic conditions for a future state, but it is not a disorderly or vicious cycle and represents a degree of equifinality. Emergence, especially strong emergence, is usually associated with downward causation, where the macrolevel affects the microlevel [Clayton and Davies, 2006]. Ryan [2007] argues that emergence is understandable only if it is explained with reference to the relationship between the scope of macro and micro level descriptions, rather than through levels of observations; he defines an emergent property as something only seen at the macro level and it is 'weak' when the levels differ only in resolution, and novel when the difference between the macro and micro levels is only in their scope. This view is expanded to propose an information-theoretical framework for complexity science [Prokopenko et al., 2007]. Deacon [2006] takes the circular causality demonstrated by feedback in dynamic systems, and argues that emergent dynamics arise out of the stochastic nature of a system, which amplifies and dampens the feedback between different dynamic levels; the circular process provides a degree of system closure and he places emergent phenomena into three subcategories (teleodynamics, morphodynamics and thermodynamics), which he arranges in increasing topological complexity.

Boschetti and Gray [2008] take what seems a less philosophical view with their list of three types of emergence, (1) pattern formation, (2) intrinsic emergence and (3) causal emergence. They question to what extent emergence can actually be modelled or simulated - especially causal emergence. They caution against taking models too seriously, "[n]o actual information about the real world is produced by a simulation" [Boschetti and Gray, 2008, p.4]. They also espouse the idea that as things change existing information is used; new information is not created, instead the original rules and states are reviewed and whatever is needed is extracted, "[t]heorems are transformations of information, not new information" [Boschetti and Gray, 2008, p.4]. Rickles et al. [2007] also take a practical view concerning how there can be a hierarchy of levels producing emergent properties. They state that:

> "Emergent properties may also be universal or multiply realisable in the sense that there are many diverse ways in which the same emergent property can be generated. For example, temperature is multiply realisable: many configurations of the same substance can generate the same temperature, and many different types of substance can generate the same temperature."
>
> [Rickles et al., 2007, p.934]

If it is agreed that emergence cannot be explained by the aggregation of the components of a system, and that something cannot arise out of nothing, then the only other system activity mentioned is the interaction or relationship between the system components; unless the system is so open that factors outside the designated system boundary are having an impact on the system. There are still the potential problems of (a) unpredictability and its association, if any, with the lack of knowledge and (b) whether everything that could have an effect on the system has been taken into account such that "out of nothing" would have to be qualified by the limitations of what was known and modelled; such that it was 'out of something unknown or not modelled'. There is also the question of whether any emergence remains emergent once it is satisfactorily explained. If this was the case, then the example in subsection 2.2.1 of a combustion engine creating mechanical energy would not be held as emergent. Kim [1999] refers to emergent and resultant. The former indicates something new and unpredictable that has no inductive or no theoretical predictability, whereas the latter is additive or subtractive and could still be a complex calculation, but its predictability is the key difference with an emergent characteristic. This could work for the creation of mechanical energy in a combustion engine if it was seen as an additive result of

a spark, a combustible gas and a container that could expand with the explosion of the gas. However, even at this simple system level it seems a relational process, rather than a physical component one.

Whatever decision is taken on emergence in simple systems and the classification of emergence once its causation is explained and a level of predictability achieved, the issue of unpredictability and knowledge remains. It may be that the desire for a more fixed definition of complexity and emergence is fruitless. But there is potential if unpredictability is coupled with non-computable. Laughlin and Pines [2000, p.28] point out that the equation of conventional non-relativistic quantum mechanics "cannot be solved accurately when the number of particles exceeds about 10. No computer existing, or that will ever exist, can break this barrier because it is a catastrophe of dimension". But, reductionism, with its goal of a theory of everything [Barrow, 2008], and the ideas of a digital world espoused by Zuse [1970] and Fredkin [1990], would dispute the existence of anything non-computable [Brodu, 2007, p.17]. However, there are other views on the nature of systems that will be discussed in section 2.4.

2.2.3.3 Complex and chaotic systems

There has been a lot of interest in chaotic systems, strange attractors and the unpredictable nature of such systems, even though they are deterministic [*see* Crutchfield et al., 2008; Gleick, 1987; Kellert, 1993; Lam, 1998; Newman, 1996; Strogatz, 1994; Wuensche, 1998]. Chaos has been associated with complexity [*see* Çambel, 1993; Couture, 2007a; Holland, 1998; Langton, 1990; Prigogine et al., 1985]. Indeed, complex systems can be chaotic, and vice versa. But in terms of systems, chaos is not synonymous with complex.

> "Chaos is the generation of complicated, aperiodic, seemingly random behaviour from the iteration of a simple rule. This complicatedness is not complex in the sense of complex systems science, but rather it is chaotic in a very precise mathematical sense. Complexity is the generation of rich, collective dynamical behaviour from simple interactions between large numbers of subunits. Chaotic systems are not necessarily complex, and complex systems are not necessarily chaotic (although they can be for some values of the variables or control parameter; [...])."
>
> [Rickles et al., 2007, p.934]

A complex system is open to its environment and subject to the variants that can result from a large collection of interacting components. While sensitivity to initial conditions is seen as a key feature of chaotic systems, the non-linearity usually inherent in complex systems results in them sharing this sensitivity. But an obvious extension to this is that the complexity of the many components and their interaction within a complex systems means that this sensitivity can arise from any or all of the various points of interaction; the input from a subsystem or from the interaction of components can induce a sensitivity that is not confined to the initial start up of a complex system. This sensitivity, whether from initial conditions for chaotic or complex systems or from any internal input point of a complex system, means that the behaviour of the system under observation is extremely hard, or impossible to predict with any degree of certainty.

2.2.3.4 Complex and complex adaptive systems

It is not always clear what difference is intended between the use of the label complex system and complex adaptive system (CAS). Some literature use complex system for natural, but not biological systems, such as for Bak's sand pile experiment. Holland [1998] uses the term CAS to cover both natural systems such as societies and the immune system, and also engineered systems such as distributed computing and artificial intelligence systems. A CAS is usually associated with a collection of interacting autonomous or semi-autonomous agents; the ability of these agents can range from relative simple behaviour to very complex systems or complex adaptive systems in their own right - human beings are an obvious example of the latter type. On a more simplistic level, the modelling of the self-organisation of cells on a grid can be seen as the simulation of a CAS. In the same way as in cybernetics, there are often layers of systems within systems. The system containing the agents that is the focus of an investigation (the system in focus) is the microsystem or microscopic level; any high level structure that arises from the microscopic levels manifests itself in the macrosystem or macroscopic level [Ruhl, 2006b]. Such a view tends to identify multiple microscopic levels below the macroscopic level. But as discussed above, while there are rules governing the agents and their interaction, the ability to predict how they will behave and how the system will evolve is virtually impossible as even minute changes to the composition of the system, or perturbation from the environment will alter the flow and evolution of the system. None of this makes any great distinction between a complex system and a CAS, nor provides any new insight into what makes a system complex. There is no real discernible difference in the elements attributed to a CAS from those of a complex system. Ruhl [2006a] comments on

the "co-evolution" and "radical openness", but such behaviour is not precluded from complex systems.

2.2.4 Complex systems revisited

The previous subsections have reviewed what makes a system complex. A number of characteristics have been mentioned. A complex system tends to be more open to its environment, making its context very relevant. The interaction between its many components leads to unpredictable behaviour that it is not possible to attribute to the physical components that make up the system. This unpredicted behaviour can manifest itself, for example, as adaptation and self-organisation that emerges unexpectedly. This gives the concept of order arising out of disorder and, in the same way, emergence being linked to a loss of randomness.

Ladyman et al. [2013, p.26] argue that randomness in a complex system should be associated with the "source of interaction" between the components. They also highlight that emergence is a required feature of a complex system, but it is not all that is needed to define it:

> "Certainly we must say that emergence in all epistemological senses is necessary for complex systems. If a system doesn't exhibit higher-level order as discussed above, then it is not complex. However, emergence is not sufficient because, for example, an ideal gas exhibits emergent order but is not a complex system."
>
> [*ibid*, p.9]

They stress the need for the system to have "many elements", although this once again introduces a degree of vagueness as to how many is sufficient; they consider a provisional definition:

> "**Complex System (physical account)** A complex system is an ensemble of many elements which are interacting in a disordered way, resulting in robust organisation and memory."
>
> [*ibid*, p.27. authors' bold font]

Memory is inferred from the robustness; and robustness means that any emergence, such as a pattern or self-organisation, persists even though the system's many elements continue to interact. This robustness also reflects a complex system's ability to suffer perturbations or survive component failure, unlike a simple

or complicated system. This definition features interacting elements and order emerging from disorder. Although not explicit, unpredictability is implied in the robust organisation arising from the disordered interaction between the many components in the system.

A more explicit version of the above definition is proposed as a working definition:

> *A complex system is a collection of many elements whose random interaction leads to the unpredictable emergence of robust organisation.*

2.3 Reductionism and its limitations

If the extreme reductionist's view is taken that everything in a system is theoretically reducible to its component parts, then an increase in the number of components in a system and the complicatedness of their interactions above that of a complicated system would be a definitive feature of a complex system.

However, if, for example, unpredictability is taken as the core feature of a complex system, then non-computability, non-reducibility, openness and the increase in system permutations through the interaction of components and context could all be seen as contributing to the level of unpredictability. It is worth looking a little more at why reductionism is considered by many to have limitations.

Even those highlighting the shortfalls of reductionism have pointed out the fact that it has been fundamental to the scientific advances made over recent centuries [Edmonds, 1999b; Ellis, 2006; Hosseinie and Mahzoon, 2011; Kineman, 2011b; Laughlin and Pines, 2000; Weinberg, 2008]. It is still seen as a mainstay of scientific analysis in the scientific method [Barton and Haslett, 2007]. The act of reduction is central to everyday activity, providing, as it does, a means of breaking something into manageable parts. In this way reductive analysis is fundamental to general problem solving. Indeed, the abstraction we employ in our mental modelling is itself a part of reductionism. Reductionism abstracts or breaks a system into smaller parts that are easier to analyse and then explain in an unambiguous syntactic language. In this way a sense of objectivity is presented. The reductionist world is composed of mechanistic systems that can be recursively de-constructed, segment by segment, layer by layer, like a matryoshka doll, until we arrive at the fundamental formulae and building blocks of everything. Consequently, any system that cannot be seen in mechanistic terms is specific and non-generic.

Reductionism focuses on the components within a system and the structural organisation of those physical components. Its models are generally expressions of systems closed to the external environment. But this overlooks the system's interactions between the components internally and with the external environment. The latter, which can be seen as the context of the observed system and its model, can influence and potentially change the functions, interactions and relationships within a system. And despite all the scientific success brought by reductionism and the scientific method over recent centuries, what we actually know is very limited; for example, the standard model is an explanation of the fundamental structure of matter, but "the model only describes the 4% of the known universe, and questions remain" [CERN, 2013].

> "For the biologist, evolution and emergence are part of daily life. For many physicists, on the other hand, the transition from a reductionist approach may not be easy, but should, in the long run, prove highly satisfying. Living with emergence means, among other things, focusing on what experiment tells us about candidate scenarios for the way a given system might behave before attempting to explore the consequences of any specific model. This contrasts sharply with the imperative of reductionism, which requires us never to use experiment, as its objective is to construct a deductive path from the ultimate equations to the experiment without cheating. But this is unreasonable when the behavior in question is emergent, for the higher organizing principles - the core physical ideas on which the model is based - would have to be deduced from the underlying equations, and this is, in general, impossible."
>
> [Laughlin and Pines, 2000]

Questions over the central and singular role of reductionism are not new. Rosen [1987, 1991, 2012], in his argument for his modelling relations, proposed that the prevalent state of systems is complex, which reductionism is not capable of properly explaining; furthermore, the non-porous boundary between a simple system and a complex one was defined by a complex system having at least one of its parts that was non-computable. Kineman [2011b, 2012] synthesised modelling relations into R-theory, with reductionism complementary to it. Davies [2006] held that a complete reductive analysis of a system is only possible if the system is closed. He argued that as novel behaviour is partly a result of a complex system being open, then reductionism would be inadequate to explain everything. Synthesis and holism hold that a system cannot be reduced or studied in isolation, but has

to be observed as a complementary part of an active, open and dynamic system [Hitchins, 2003b]. Çambel [1993] advocated a dual approach that applies reduction to the details of complexity, while also maintaining an holistic viewpoint. Koestler, as highlighted by Corning [2002], shared a similar view, believing that living systems could only be explained through both reductionism and holism. The openness means that unpredictable or novel behaviour in a system can be explained by causes outside the system boundary. The system boundary can be expanded to include the causes, thus making the behaviour predictable within the system. But Chu et al. [2003] explained that there is a state of radical openness's where the boundary can no longer be extended and the system is at its most complex. As the resources needed to reduce a complex system far exceed what is available to us, Edmonds concentrated on the models and formal languages used to investigate and explain complex systems to then arrive "at an analytical useful conception of complexity" [Edmonds, 1999b, p.134]. He later suggested a pragmatic approach to the reductionism verses holism debate [Edmonds, 1999a]

Consequently, there are a number of characteristics of complexity that reductionism fails to, or has difficulty in addressing, including openness, component interaction, emergence, and non-computability.

2.4 A non-reductionist view of systems

The consideration of interaction between system components and the consequence of opening a system up to external perturbations is not a recent occurrence, although modern day complexity theory and chaos theory would be the first focus of much research.

> "Complexity theory is an approach to the modelling of highly complicated and interconnected systems using techniques derived from the physical sciences, with a focus on self-organisation, emergence and nonlinearity. It takes inspiration both from general systems theory and cybernetics."
>
> [Ramage and Shipp, 2009]

While it is interesting that the authors talk of "highly complicated and interconnected systems" rather than complex ones, the main point for the current discussion is the reference to general system theory (GST) and cybernetics.

> "GST studies systems at all levels of generality, whereas Cybernetics focuses more specifically on goal-directed, functional systems which have some form of control relation."
>
> [Heylighen and Joslyn, 2001]

GST, founded by von Bertalanffy, believed in the importance of an interdisciplinary approach that considered open systems, emergence, system boundaries and hierarchies [Ramage and Shipp, 2009]. The intentions of GST were:

> " 1. To investigate the isomorphy of concepts, laws, and models from various fields, and to help in useful transfers from one field to another
>
> 2. To encourage development of adequate theoretical models in fields which lack them
>
> 3. To minimize the duplication of theoretical effort in different fields
>
> 4. To promote the unity of science through improving communications among specialists."
>
> [Adams et al., 2013]

For their part, cybernetics focused on the idea of feedback and information within a system, and the similarities between human and machine behaviour. Von Foerster and second-order cybernetics moved cybernetics from a quasi objective and mechanistic view of systems, to one that emphasised "autonomy, self-organisation, cognition, and the role of the observer in modelling a system" [Heylighen and Joslyn, 2001]. Glanville [2002] saw this development as comparable to the progress made between the "Newtonian view of the universe, and the Einsteinian". Cybernetics distanced themselves from the linear causality of reductionism, seeing causality as circular and self referential, without any defined primary cause. Thus cybernetics modelled systems with feedback loops and feed-forward loops. This circularity can be seen in the organisational closure of a system, such as the concept of autopoiesis that Maturana and Varela [1980] used to describe life.

The basis of relational theory and the modelling relation [Rosen, 1991, 2012] was formulated from a similar multi-disciplinary melting pot of ideas as cybernetics, although it has only recently been synthesised into R-theory [Kineman, 2011a,b, 2012]. Core to the theory is the idea that complex systems predominate and mechanistic or simple systems are a specific category of systems. R-theory has similarities with cybernetics. The role of observation plays a key role, as it does in second order cybernetics. The model of anticipation (*see figure Figure 2.2(a)*)

bears, according to Nadin [2010, pp.24-25], a "striking analogy" to "von Foersters concept of non-trivial machines" (*see Figure 2.2(b)*). Its metabolism repair, or (M,R) system has comparisons with autopoiesis [Nomura, 2007] and the concept of anticipation has been coupled with autopoiesis in an algorithm to show autopoietic properties in a cellular automata model [Dubois and Holmberg, 2010]. At its heart is the view that the organisation of relations between the components of a system within a context is more important than the organisational structure of the physical components. In this way everything observed can be seen as a model, even our view of our surroundings and the natural world. The model or realised view of a system is bound to the continual process of abstracting and modelling the system within a context. The context can be modified or changed and represents the potential that a system can realise. A mechanistic view of a system can be taken by reducing the realised system to a fixed, single context, with no potential outside of that context. Organisation within R-theory and cybernetics should not be confused with thermodynamic order and disorder; instead it is how a system is organised in terms of the relationship between components and also between the organisation and the components.

The following subsections look at some of areas that can be seen as limited by reductionism, but catered for by GST, cybernetics, R-theory, or a combination of them.

2.4.1 The observer

Consideration of the role of the observer has already been referred to in subsubsection 2.2.3.2 when looking at emergence and in the section above when referring to Von Foerster and the second order cybernetics and the importance they placed on the observer. In second-order cybernetics the focus moves from the detachment of observed systems to an integration of the observer and an 'observing systems' where the aim of the model is replaced by that of the modeller. In this way the system becomes less controlled and more autonomous and the focus is on the interaction between observer and observed [Umpleby, 2001].

R-theory makes the forming of models an essential part of the relationship that you have with the environment, or the 'ambiance' external to your inner self. In essence, our modelling of the natural world is the forming of a mental model, making the relationship between modelling and observation pervasive.

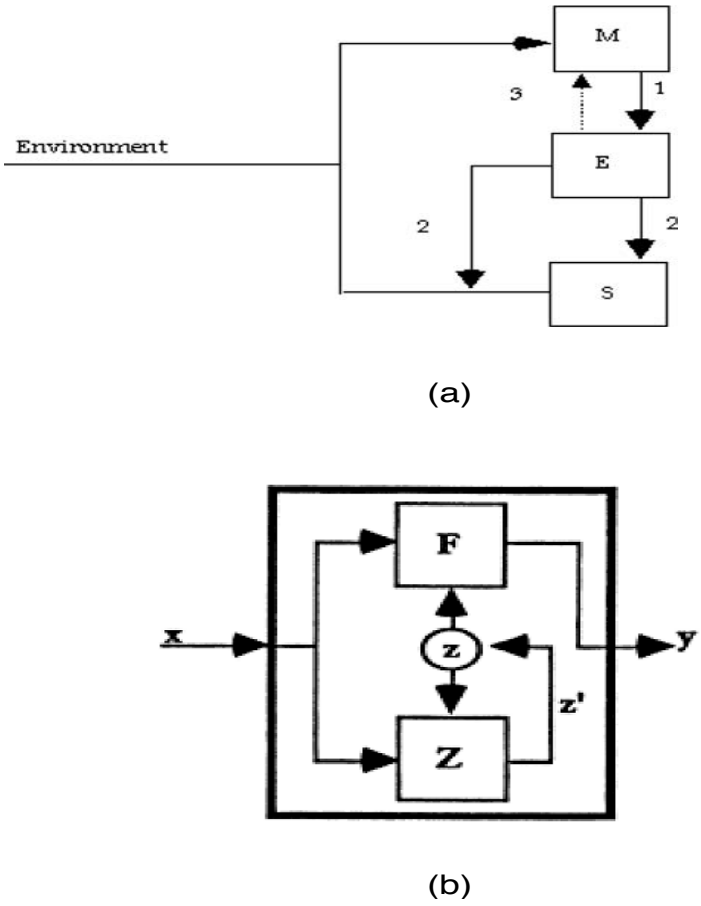


Figure 2.2: (a) Rosen's anticipatory model [Louie, 2008, p.299], with the model (M), the effector (E) and the object system (S). The output goes into the encoding for 'current time' model of the natural system; thus the environment entails both the model and the scenario of the object system (S); M entails E and can be entailed by E; and E entails S both directly and through influences the environment's entailment of S. (b) von Foerester's non-trivial machines (*adapted from* Nadin [2010, p.25]), with internal feedback and a similarity to a black box where it is synthetically determined, history dependent, analytically undetermined and unpredictable.

The impact of observation and measurement within a supposedly closed environment was first clearly demonstrated in quantum physics. But unexpected emergence can also be seen as a result of observation and analysis; "The scientist interacts with the system in two ways; through setting the actual experiment up to be observed and through measurement probing of the system" [Kier and Witten, 2005]. A link between observation and emergence is also highlighted by Kitto [2008], who suggests that the observer creates an epistemological emergence, while novel complex system behaviour is 'real' emergence and can be termed as ontological emergence. The impact of the interaction between the observer and the system can be more profound. Gershenson and Heylighen [2003, p.610] reason that, "[a]gain, the purpose of the system is not an objective property of the system, but something set by an *observer*" (authors' italics).

It is understandable that despite all attempts to make science as objective as possible, inevitable anthropocentrism means everything has to be defined in terms that we can understand and, potentially, observe. Bonabeau and Dessalles [1997] connect emergence with the observer and the methods of measurement used; these can be the observer's own senses or created ones, such as models, but they are all supplementary to the system and not intrinsic to it. They argue that,

> "Nothing would emerge in the absence of human observers and of their conceptual constructs."
>
> [Bonabeau and Dessalles, 1997, p.10]

Observing does not equate with subjectivity, but it does belie any concept of true objectivity being really achievable.

2.4.2 Relations, anticpation and feedback

The importance of interaction between components was key in GST, cybernetics and R-theory.

> "Systems are nets of *relations* which are sustained through *time*. The process by which they are sustained are the process of *regulation*. The limits within which they can be sustained are the conditions of their *stability*"
>
> [Vickers, 1983, p.17. author's italics]

The feedback and feed-forward mechanisms of the cybernetics and the concept of anticipation in R-theory are examples of relational influences within open systems.

In R-theory the anticipatory system of an organism, which is seen as a subset of a complex system, allows it to model situations in faster than conscious time, thus allowing possible scenarios to be analysed and used in the making of a decision [Rosen, 2012]. The process could be far reaching, as in evolutionary characteristics:

> "In terms of material structure, the evidence of organization-based anticipation of future conditions can be seen in the existence of such things as reproductive organs at birth in human babies, long before any considerations of reproducing become a biological imperative for that individual organism."
>
> [Rosen and Kineman, 2005, p.407]

Reductionism holds with a linear concept of time, where the cause of something precedes the event. As anticipation and feed-forward mechanisms bring in the concept of the 'future' having a causal effect on the present, they can be seen as problematic for reductionism. On the other hand, feedback is something that fits into the modelling of systems, simple or more complicated. A feedback mechanism can be seen as expressing the relationship between components in a system, as well as with the immediate environment. R-theory is held as a better or as good a way to model simple systems [Rosen, 1991], but there is no evidence of a working example.

The relational view is that when two complex systems are combined, the resulting combined system or systems have their own context and attributes; so any new features or behaviour that might be viewed as emergent, are explainable as part of the interaction and combination of the systems, which is more than just the sum of the combined systems. Kineman [2007, pp.64-65], with reference to the article by Ulanowicz [2007] *Emergence, naturally!*, highlights how the third system appears naturally, leading to $1 + 1 => 3$; stating that:

> "This result supports the view that the functions of each system interaction must be considered uniquely and recorded with respect to their original context. Furthermore, the concept of 'emergence' is clearly an artefact of the mechanistic/quantitative analysis; whereas the 'third' system exists quite naturally in a relational analysis Ulanowicz [2007]."
>
> [Kineman, 2007, p.65]

The important of context is considered in the next section.

2.4.3 Context and modelling

Cakar et al. [2007] define degrees of self-organisation (SO) where the term self depends on the context of the system's boundaries and the view point of the observer. Second-order cybernetics was focused on the importance of the observer in the modelling process. This took the model away from a closed, mechanical representation, to one that was open and subject to perturbations. The observer provided the context or setting of the model, and as such the actions of the observer and the context influences what was being modelled. Likewise, the why or purpose of a system is linked to the context of a system. The context illuminates the purpose of a system and facilitates or impedes the achieving of any system goals. So the context that the system is observed in is crucial to how the system is observed.

Edmonds [2007] argues that modelling with a view to the context is essential, terming this as 'context-dependency'. In a later work Edmonds [2010] makes the observation that unlike complex systems, simple systems can be modelled from the perspective of a single context. He goes on to take a pragmatic view where the inability to compute the full nature of complexity can be aligned with the concept that complexity can only be modelled in a localised and temporal way [Edmonds, 1999a]. In this way the constraint of a localised context is imposed to allow a limited observation of a complex system to be modelled.

Chu et al. [2003] proposed the concept of 'radical openness', where systems are at their most complex. This would appear to correlate with the 'high end complexity' of Kitto [2008]. She goes on to point out it is not just that a complex system is open, but that there is a 'blurring' of the boundary between a system and its environment; she argues that the dynamics of the system with its environment, or its 'contextuality', is vitally important. Kitto proposes a scalar view of systems where there are bands of 'simple', 'contextual' and 'observer driven', but there is no clear distinction between what is or is not a complex system (*see Figure 2.3*).

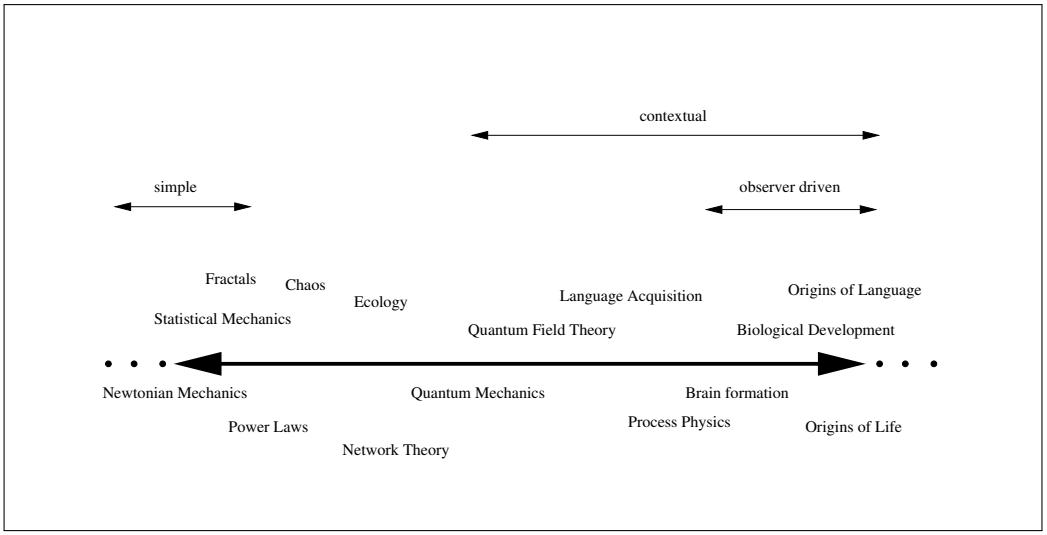


Figure 2.3: A linear complexity scale proposed by Kitto [2006, p.21]. All systems belong to a scale of complexity, where some are more complex than others. The scale stretches from Newtonian mechanical systems exhibiting simple 'complexity', to systems displaying high end complexity

2.4.4 Isomorphism, analogous systems and modelling

Traditionally the methodology of science has followed either the experimental or theoretical paradigms, with the latter relying on the former to verify or falsify their theories [Kroc et al., 2010]. This approach does not open itself up to abstraction at a level that can find and exploit such phenomena as emergence [Beautement, 2001], nor to the modelling of component interactions and relations. The benefit of such an approach can be seen in the application of chaos in short-term prediction on the behaviour of a complex system or the use of small perturbation to stabilise a system without even fully understanding the dynamics of the system [Lam, 1998]. Hitchins [1996] states that complexity can be part real and part perception; he sees models and an external view of the system as two effective ways of cutting through the complexity. As mentioned in subsection 2.2.2, Bak [1999] maintains that large dynamic systems are both too complicated to model in their entirety, and produce too much data in real life for any comprehensive collection and analysis of their output; he advocated the use of models, starting with simple ones that could then be built on to discover insights into the working of the system. Natural systems, such as the swarming behaviour of bees or fish, or the Nile perch in Lake Victoria discussed by Chu et al. [2003], are open systems and have no unique pattern. Therefore any models used to simulate them cannot be verified or validated. A prediction can be made and a model applied and observed, but the confirmation can only be partial and the main worth of the model is heuristic [Oreskes et al.,

1994]. In this way a speculative formulation, serving as a guide in the investigation or solution of a problem (often the most appropriate solution of several found by alternative methods), is selected at successive stages of a program for use in the next step of the program. Bonabeau et al. [1999] argue that a model of a swarm should be consistent with what is known about the natural system as you are trying to show how the natural world works.

In R-theory the modelling relation has to be in congruence, or else it is not a suitable model [Louie, 2007; Rosen, 1991]. If the congruence or modelling relation is between just the input and output, then you have a *simulation* as you have no way of learning how the natural system functions internally or is organised. A modelling relation has to be formed between the causality of the natural system and the inference of the formal system; this enables you to gain insight into the organisation and workings of the natural system. In contrast to this, cybernetics has the concept of a 'black box' whose internal workings cannot be explained, except for the inputs and outputs; in some ways this can be seen as a simulation within R-theory. Two important concepts arise out of the implication of the modelling relationship, which have similarities in cybernetics:

1. a natural system, or the system being modelled, can have multiple models; Edmonds [2007] refers to this as 'clusters of models' that can be combined in many ways; cybernetics also have the concept of one to many, as well as many to many, which can increase the variety of a system

2. the same formal system or model could hold for more than one natural system; this would make the natural systems analogous; in cybernetics and GST a key objective is to find isomorphisms in the modelling of systems, although allowance has to be made for von Bertalanffy caution against the inappropriate application of isomorphism

Analogy is a relation between natural systems which arises through the models of their causal entailment and not directly from their material structures. This can be seen in the way we apply and modify mental models to different systems and different contexts. As such, analogy and its cognates offer a more powerful and physically sound alternative to reductionism; they 'share a common model' and therefore are 'analogous systems', as opposed to 'one encompasses the other' with a separate model for each [Louie, 2009, pp.102-103].

However, Godfrey-Smith [2006] in his argument that the 'semantic view' was distorting the philosophical understanding of models, argues that:

> "For one scientist, the model might operate merely as a predictive device. All that matters is its input - output profile. For another, the model is not just a device for predicting what the system will do, but a causal map of the target system that tells us something about why it does what it does. This more realist attitude does not present itself as a "yes or no" choice, or even as a single gradient. There is usually something more like a multidimensional space of different ways in which a model system might resemble a target. Following Giere (especially 1999), I reject attempts to give a uniform formal analysis of these resemblance relations using a concept of isomorphism, or some relative of that concept."
>
> [Godfrey-Smith, 2006, p.733]

The subjectivity associated with the application of a semantic view of a system is a common argument levelled at the alternatives to reductionism. But, reductionism attempts to facilitate objectivity through a syntactic approach. In doing this it encloses and limits itself, especially in terms of complex system modelling and analysis.

> "For better or worse we are now witnessing a transition from the science of the past, so intimately linked to reductionism, to the study of complex adaptive matter, firmly based in experiment, with its hope for providing a jumping-off point for new discoveries, new concepts, and new wisdom."
>
> [Laughlin and Pines, 2000, p.30]

The modelling process is pervasive in the natural world,

> "At one extreme, bacteria hardwire simple models in their biological structure, while at the other extreme humans employ conscious mental processes and store formal mathematical tools in books and computers. Nonetheless, they both 'model' and 'predict'."
>
> [Boschetti et al., 2012, p.108]

They further qualify that models are not solely for prediction, but also for examining, comprehending and even controlling a system; but any prediction is (1) not a

prophecy, (2) only valid within the context of the model, (3) is scale-dependent, and (4) the building of the model itself is a catalyst for further conjecture and experimentation, rather than end target of the modelling process [Boschetti et al., 2012, pp.109-111].

A model of a system can be at a level of abstraction that renders it specific to that system. This would logically give the potential for more to be learnt about the system being modelled, but it would negate the possibility of discovering any analogous systems and through them any comparable and transferable features or facts. This implies that the degree of abstraction would need to be more generalised in order to facilitate the identification of analogous systems. A mental model can be formed of the result of dropping a brick on your foot. That model can be abstracted in different ways and levels to suggest the consequence of, for example, (a) a brick hitting another part of your body, (b) a heavy object hitting you, (c) the level of pain and damage caused by the fluctuation in the weight and velocity of the object hitting you, (d) the pain experience by someone else hit by an object, and onto (e) the likely damage resulting from the impact of two inanimate objects, based on additional knowledge such as the weight of the objects and the velocity at the point of impact. The potential of using the mental model in a different modelling scenario is increased as the degree of abstraction becomes looser. In the last example the focus has moved from pain and damage, to just potential damage, which may be informed by another set of models more related to the specific inanimate objects under observation; such as ones that took account of the physical composition of the objects. And as can be seen, the relevance of the weight and velocity involved in the impact has been extracted as things that can be used in the analysis and prediction of the consequence of two objects colliding.

A cybernetic inspired model showing a basic feedback system could be labelled to show a number of basic systems, where an activity is turned on or off depending on the feedback received. This involves measuring the feedback, but it might be the temperature in a room, the level of water in a cistern or the amount of energy being used to maintain a vehicle at a set cruise control speed. In this very basic level of abstraction the state of the feedback mechanism is a general indicator of the state of the system. As the model becomes more detailed, then the link between the commonality of the model and analogous systems becomes harder to maintain, and any general indicator becomes more obtuse. This would suggest that the generality of an indicator would lessen as the model became more detailed and less abstract. A slightly different approach would be to consider a common modelling environment that is used for the modelling and simulation of

a variety of different systems. So the analogy would be through the output, or output space being created or used. This would restrict the generality of any indicator to systems that could be modelled via the common modelling technique, and possibly to groups within that range of systems. Any generality would be in the measurement of that output space.

2.5 Computer simulation

Although computer simulation, (sometimes referred to as digital simulation), is the focus of this section, it is worthwhile first to look at an analogue simulation. A celebrated example is the "dumb hole" simulation using Bose-Einstein condensates and fluid flow as an analogue for the behaviour of black holes [Schützhold and Unruh, 2002]. Dardashti et al. [2014] hold that this simulation of a dumb hole goes further than the common use of analogy in philosophy and science. They propose that analogue simulation should replace the nomic isomorphism of analogical reasoning, which in the case of the dumb hole example would lead to the impractical task of establishing a full nomic connection between the source and target system. Instead "[a]nalogue simulation, as we understand it, can occur even when the syntactic isomorphism one can identify does not hold between the laws governing the two systems in generality" [Dardashti et al., 2014, p.10]. They contend that:

> "[T]here *is* a syntactic isomorphism to be exploited in the dumb holes case, and we think it is best understood as holding between two very particular *modelling frameworks*, each with narrower scope than genuine laws. The question is not of an isomorphism between the laws of fluids and the laws of quantum gravity on the other. Rather, there is an isomorphism between a particular adequate way of modelling a special class of fluid setups and a particular adequate way of modelling the behaviour of quantum fields near a black hole horizon."
>
> [*ibid*, authors' italics]

The analogue simulation has a material model and experiment as its source system that is analogous to the target system, whereas a computer simulation runs a model that represents the target system. While both types of simulations share no material similarities with the target system[1], Durán [2014] points out that analogue simulations, as opposed to computer simulations, are causally related.

[1]The concept of silicon based target system being simulated on a computer with silicon chips is looked at by Winsberg [2009, p.5], who holds that "[o]ne problem is that, in this case, it seems quite clear that the relevant similarities are not material".

A succinct definition of a model used as the basis for a computer simulation is given as:

> "[A] hypothetical system that is claimed to represent a certain target-system."
>
> van der Grient [2011, p.5]

Thus a computer simulation uses a hypothetical system as its model, which might be a representation of an abstraction of either the target system or a system analogous to an abstraction of the target system. The latter can be seen in the use of abstractions of natural biological systems in the computer simulation of diverse systems (for example, cockroach aggregation and collective decision making in a group of micro-robots [Garnier et al., 2008]; swarm theory and the control of unmanned aerial vehicles [Hart and Craig-Hart, 2004]; the aggregation of the Dictyostelium discoideum cellular slime mould and decentralised gathering [Fatès et al., 2008; Girau et al., 2009]; chemotaxis and pedestrian dynamics [Schadschneider, 2001]).

Kier and Witten [2005] draw a distinction between two types of simulation along different lines. They see the two types as both being "active imitations of real things, [...] but with different aims" [*ibid*, p.5]. The purpose of the first is to replicate a specific behaviour of the target system, but usually with very little contextual relevance; such as a mechanical bird whistle or a hologram. "Such a simulation reveals little or nothing about the features of the original system, and is not intended to do so" [*ibid*]. But the second type of simulation has much greater ambitions, achieved by running it:

> "It attempts to mimic at least some of the key features of the system under study, with the intent of gaining insight into how the system operates. In the context of our modeling exercise, a simulation of this sort means letting our model 'run'. It refers to the act of letting the parts of our model interact and seeing what happens. The results are sometimes very surprising and informative."
>
> [*ibid*]

But there is debate about the worth of simulations within scientific enquiry and the nature of their epistemic value.

2.5.1 Epistemic value of computer simulation

Di Paolo et al. [2000] show a general cycle of scientific enquiry where *theory* is refined by constructing a *model* or a series of models from which arises *predictions* that are then compared to *observations*. In a more specific way the scientific method has been used in the natural sciences to derive, collate or amend knowledge using a cycle of *theory*, *analysis*, *hypotheses* and *experimentation*. Experimentation has traditionally been seen of greater epistemic value than a simulation [Giere, 2009; Guala, 2002; Morgan, 2005; Winsberg, 2014]. The principal argument is based on materiality, where an experiment "bears a material similarity to the target of interest, but in a simulation, the similarity between object and target is merely formal" [Winsberg, 2014, p.15]. This material connection between source and target systems promotes a physical connection that cannot be established through a simulation:

> "The epistemological payoff of a traditional experiment, because of the causal connection with the target system, is greater (or less) confidence in the fit between a model and a target system. A computer experiment, which does not go beyond the simulation system, has no such payoff."
>
> [Giere, 2009, p.61]

So the example of an aircraft wing being tested in a wind tunnel is an experiment that has a high degree of materiality and, consequently, of epistemic worth. In this process of experimentation, models can be seen as ways to both formulate a suitable experiment and to encapsulate and communicate the findings:

> "Models not only explain why certain phenomena occur, they also serve as a way of summarizing knowledge. A good model compresses and organizes large amounts of experimental data into a succinct description of the system."
>
> [Tamulonis, 2013, p.4]

This lack of material connection can call into question the validity and epistemic worth of computer simulations when compared directly against an experiment. But Parke [2014, p.17] argues that in the debate over epistemic privilege between simulation and experimentation, "[t]he methodological difference between experiment and simulation is not purely pragmatic. It matters for making judgements about epistemic value - but only in a context-sensitive way". She sees experiments as usually having privilege when little background knowledge is known; but

in some contexts there is both "enough [information] to build reliable simulations to answer certain sorts of questions" and issues of scale making it infeasible to construct accurate physical experiments [*ibid*]. Indeed there are situations, such as in astrophysics and cosmology, where experiments are either not possible, or have inferior output to computer simulations [Morrison, 2009].

Gershenson [2007, p.54] argues that the model cannot be understood until it is run; so the computer simulation implements the model. He sees the process as progressing from the abstract to the particular and the level of simulation moving from a simple proof of concept to an extensive run that provides data on the performance of the system. So in some ways a simulation is itself a model of the system, but it is not the same as a model.

> "To summarize, simulations are, like models, autonomous both from theories and from the real world. They differ from models mainly in their temporal expansion (and sometimes also in their representation of a temporal process) as well as in their epistemic opacity."
>
> [Grüne-Yanoff and Weirich, 2010, p.26]

The epistemic opacity is where "[t]he use of a computer to tackle numerical problems causes the process between input and output of the simulation to become opaque" [van der Grient, 2011, p.9]; this refers to the inability to follow the steps of the process both because the process itself is complicated and because it is obscured as part of the internal working of the computer. This aligns computer simulation with the idea of a black box, although Beisbart [2012, p.415] would argue that "[a] program that predicts the behavior of a target system using a black box would probably not count as a computer simulation". Also, epistemic opacity supports the premise that nothing can be learnt about the causal links of a system through a simulation. This would make simulations of little or substandard use in understanding the working of a system when compared to models [Giere, 2009; Rosen, 1991].

But while the full working of a computer simulation may be epistemically opaque, they can provide a means of investigating how a system works [van der Grient, 2011], or as a guide to modifications to a model or an experiment [Peschard, 2012]. Rasmussen et al. [2001, p.307] maintain that "[t]hrough simulation we have come a long way in understanding the nature of evolutionary process". Some see simulations as capable of producing new knowledge. Barberousse et al. [2009] investigate whether simulations can be seen as experiments and conclude that they

"do generate new data about empirical systems, just as field experiments do" [ibid, p.573]. In contrast, they are also seen not as experiments, but as arguments:

> "I conclude that the epistemic force of computer simulations does not derive from the fact that they are experiments (if they are so at all), and that the argument view is superior since it brings to the fore the assumptions and models upon which simulations are built."
>
> [Beisbart, 2012, p.425]

Morrison [2009] argues that computer simulation has epistemic status equivalent to experimental measurement and sometimes provides knowledge in areas where traditional experiments are of little use. But Giere [2009] counters that even in models that give us an insight into the phenomena of the target system that cannot be obtained through traditional methods, such as in astrophysics, meteorology and climate change, any output has to be qualified "within the confines of the model" and what is obtained is "only a simulated measurement, not a real measurement" [Giere, 2009, p.61]. Winsberg [2009] argues against both sides of the debate over the epistemic value of experiments and simulations, while concluding that there is some justification to the intuitive feeling that there is a difference. He feels that "it is true that experiments are not *intrinsically* more epistemically powerful than simulations. But there may still be important epistemological differences between experiments and simulations" Winsberg [2009, p.584 author's italics].

His final argument is based on the 'quality' of the background knowledge:

> "How trustworthy or reliable an experiment or simulation is depends on the quality of the background knowledge, and the skill with which it is put to use, and not on which kind it belongs to."
>
> [Winsberg, 2009, p.591]

He earlier qualifies this background knowledge and the difference between simulation and experimentation as follows:

> "The conceptual distinction between experiment and simulation is now clear: when an investigation fundamentally requires, by way of relevant background knowledge, possession of principles deemed reliable for building models of the target systems, and the purported reliability of those principles, such as it is, is used to justify using the object to

> stand in for the target, when a belief in the adequacy of these principles is used to sanction the external validity of the study, then the activity in question is a simulation. Otherwise, it is an experiment."
>
> [Winsberg, 2009, p.588]

Consequently, the reliability of the modelling assumptions or principles used are employed for the *internal validation* of an experiment, but for the *external validation* of a simulation.

Durán [2014] sees computer simulations, through their ability to process large amounts of information, as cognitive enhancers. But he sees this as a lower level epistemic characteristic compared to the ability to "describe *patterns of behavior* of the target system"; where he takes "the notion of *patterns* as one which reflects the structures, the performance, and the *behavior* of the target system" [*ibid*, p.85. author's italics]. He argues that:

> "The advantage of conceptualizing computer simulations in this way is that the physical features of the computer are no longer their primary epistemic virtue, but rather, it is their capacity to represent or describe *patterns of behavior* of the target system that entrenches computer simulations as epistemically powerful."
>
> [*ibid*. author's italics]

Thus, his evaluation of the epistemic power of computer simulation is not focused on its mechanistic capabilities, but rather on "the analysis on the kind of scientific activities that a computer simulation can perform" [*ibid*].

If a computer simulation cannot realise epistemic value through any material connection to the target system, then its worth has to be seen in terms of its ability to represent the observed behaviour of the target system. This can also be realised through the alignment of a simulation with, or as the conceptual model of the target system, much as in the role of thought experiments in traditional science.

2.5.2 Computer simulations as thought experiments

Mäki [2005, p.309] uses the term 'thought experiment' to refer to models established on theoretical isolation, as a comparison to 'material experiments'. He sees theoretical modelling as being able to impose control on the level of isolation that is beyond that of a material experiment. He concludes with the suggestion

that "models=experiments insofar as theoretical models and material experiments share the characteristics of representations that are manipulated in order to effect isolations" [Mäki, 2005, p.311]. He goes on to argue that:

> "We can talk about two kinds of models: theoretical and material models; about two kinds of experiments: theoretical experiments (or thought experiments) and material experiments; and more generally about two broad classes of representations: theoretical and material representations."
>
> [Mäki, 2005, p.312].

While thought experiments can be compared with computer simulations in the aspect of their content, form and usage [Beisbart, 2012, p.426], by its very nature the processes outlined in a thought experiment need to be transparent and understandable, which is contrary to the epistemic opaqueness of computer simulations [van der Grient, 2011, p.18]. But van der Grient [2011, pp.18-20] goes on to argue that computer simulation can raise epistemological questions and "discovers in the literal sense of the word. It makes visible what was previously hidden or inaccessible to us". Di Paolo et al. [2000, p.1] hold the view "that although simulations can never substitute empirical data collection, they are valuable tools for re-organising and probing the internal consistency of a theoretical position". They contend that the model verses simulation model debate can be seen as between the two extremes where, (1) models are general, whereas simulations are specific and gain validity based on the amount of the real system they accurately capture, as opposed to (2) "simulation models [appear] to be more like thought experiments: unrealistic fantasies which nevertheless shed light on our theories of reality" [*ibid*, pp.3-4]. They disagree with both extremes and conclude that:

> "[I]t is reasonable to understand the use of computer simulations as a kind of thought experimentation: by using the relationships between patterns in the simulation to explore relationships between theoretical terms corresponding to analogous natural patterns" .
>
> [*ibid*, p.9]

Their comparison is to opaque thought experiments owing to the explanatory opaqueness of computer simulations; but like 'transparent' thought experiments, the insights and theoretical challenges they engender then need systematic enquiry and testing in the real world.

2.5.3 Advantages of computer simulation

A number of advantages gained by the use of computer simulations have been outlined. Kier and Witten [2005, p.7] explain how multi-scale simulations in biology have been "termed '*in silico*' modeling and simulation"; they go on to promote the usefulness of both modelling and simulation within this environment:

> "Modeling and simulation provide the scientist with two very useful tools. The first of these is validation of the theoretical understanding and its model implementation. The second of these tools is that, the more complete the model, the more it provides an experimental laboratory for further research on the very system being modeled. Thus, "*in silico*" models can both validate current viewpoints/perspectives of the dynamical evolution of a system and can provide an environment in which the scientist can explore potential new theories and their consequences."
>
> [Kier and Witten, 2005, p.8]

Humphreys [2011, p.9] holds that the epistemic opacity of computer simulations gives rise to many features that were out of reach prior to the widespread use of computers. He believes, in contrast to reductionism, that computer science and the simulation tools it provides offers a means of seeing commonality between different domains:

> "Reduction suggests to us that we can better understand higher level systems by showing how they can be reduced to, how they can be explained in terms of, lower level systems. Computational templates suggest that we can gain understanding of systems without pursuing reduction by displaying the common structural features possessed by systems across different subject domains."
>
> [Humphreys, 2011, p6]

Giere [2009, p.59] agrees with the underlying premises of Morrison [2009] "that computer simulation is a qualitatively new phenomenon in the practice of science. It is the major methodological advance in at least a generation".

The focus of the debate within the philosophy of science arena has been mainly about where computer simulations based on mathematical equations fit within the accepted model of scientific practice. However, a large body of work has been built around computer simulations utilising local rules, rather than general overarching mathematical equations.

2.5.4 Types of simulations

Computer simulation can be split into two types where the simulation is based on; (1) the application of a global mathematical equation, such as a partial differential equation (PDE), and (2) the use of local rules to determine the behaviour of individual elements or agents within the simulation [Winsberg, 2014].

Durán [2014] focuses on equation-based computer simulations, specifically excluding cellular automata, agent-based simulations and complex systems from the class of computer simulations used in his defence of the epistemic power of computer simulations. He gives a number of reasons for the exclusion, concluding with:

> "Perhaps the best reason for excluding cellular automata, agent-based, and complex systems from this study stems from the minimal requirements needed for a successful explanation by computer simulations. In other words, in equation-based simulations the explanans can be reconstructed directly from the simulation model since its computation does not add value to the results. In any of the other classes of computer simulations, the interplay of the various elements during the computation must be considered as part of the explanans, for they are part of the success of an explanation."
>
> Durán [2014, p.96]

In the same vein, Keane [2011b] argues that within combat modelling a PDE model can be more easily modified and explained than CA. She also holds that:

> "There is a danger in the anthropomorphization of agents, insinuating agents have a reasoning and planning ability when this is obviously not the case. Also these types of wargames concentrate heavily on the addition of extra communication ability between agents, shifting the emphasis to global or increasingly complex nonlocal features."
>
> [Keane, 2011b, p.12]

Although CA are used in many combat simulations, she believes that the so called emergent behaviour is difficult to explain, especially when 'intelligence' is assumed for the agents; whereas, she contends, her mathematical model explained more about what influences the behaviour of the system and agents. She recommends the complementary use of PDE; especially "as modern warfare now takes on a manoeuvrist approach" [Keane, 2011a, p.2735] that has to include nonlocal communication.

Bryden and Noble [2006] review the difference between explicit mathematical modelling and computer simulation, where the latter has implicit steps within it. The explicit mathematical treatment are held as simpler to understand than some "complex and unwieldy computer simulations" [*ibid*, p.2]. They refer to Chomsky [1986] and his concept of *competence* and *performance*; the former is our intrinsic linguistic ability and is superior to the latter, which is our actual production of language. In this way a computer simulation is seen as a performance of a scientific explanation, whereas an explicit mathematical model can be viewed as having some inherent ability or competence as a scientific explanation. They explain how computer simulations can still have relevance:

> "At this point, we are left with a conundrum. If computer simulation models are viewed as mere instances (performances) rather than as systematic explanations (having competence), how can they be of use to science? The answer is that there are many areas, identified especially in the ALife field, which do not yet yield to mathematical modelling but in which simulation models can already be produced. Such simulation models not only have scientific power as proofs of concept and for generation of insights for performing empirical science, but they can also have some explanatory power (Di Paolo et al., 2000)."
>
> [Bryden and Noble, 2006, p.3]

The concept of computer simulation as a performance echoes both the lack of materiality between the source and target systems, the representational nature of the model and the idea that a simulation needs to be run. This could be seen as a positive and important feature of computer simulations, especially when studying complex systems and facilitating the investigation of emergence where the actual background knowledge of the target system is restricted. Bedau [2008] draws an analogy between computer simulation and weak emergence and argues that:

> "Derivation by simulation is the process by which causal influence typically propagates in nature. [...] Thus, derivation by simulation and weak emergence apply to natural systems just as they apply to computer models".
>
> [Bedau, 2008, p.164]

He goes on to claim that:

> "Computer simulations allow weak emergence to extend reductionism into new territory, but they do so by embodying the idea that something's nature can depend on its genesis".
>
> [Bedau, 2008, *ibid*, p.184]

Thus, while reductionism is essentially non context dependent, computer simulation and the study of weak emergence relies on context in the form of its initial conditions. Tamulonis [2013] follows a similar line of thought and states that:

> "It is often not clear, for example, how a certain tissue functions as a whole based on observations of individual cells. To understand complex systems, we need models that can link emergent properties with the underlying constituents."
>
> [*ibid*, p.3]

He argues that a cell-based model can therefore act as a virtual lab, where experimentalists and theoreticians alike can play with the system and generate targeted ideas for new experiments [*ibid.* p.10]. He holds that the sheer computational power of computer models offers the only way to model complex systems, although any predictions have to be induced rather than deduced:

> "By conducting many simulations for different system parameters, we can explore how the system behaves under different conditions. This is a major difference between between computational and mathematical models. Predictions from computational models can only be made through induction from running simulations, whereas predictions from purely mathematical models can be deduced from the equations, and may not require running any simulations at all. Computational models may therefore seem to be a more brute force approach, but for most complex systems there is no other way."
>
> [Tamulonis, 2013, pp.9-10]

Wolfson incited criticism by suggesting that his in depth study of CA [Wolfram, 2002] heralded, as the title proclaimed, "[a] new kind of science". He puts forward the idea of models based on simple programs that offer, through their discrete operations, an easier way to discover basic phenomenon of complexity than a mathematical equation modelling a continuous system [Wolfram, 2002, pp.161-168]. The idea of materiality between source and target system is not relevant within this modelling world:

> "Whenever one makes a model of something, what one's trying to do is to capture certain essential features, and then idealize everything else away, [...] [T]he whole point of a model is that it's supposed to be an abstract way of reproducing what a system does; it's not supposed to be the system itself."
>
> [Wolfram, 2003]

Such models are just run. He suggests that finding the appropriate underlying model could be achieved in a number of ways:

> "Usually that's sort of a creative act. Or perhaps one has some well-defined class of models, and one can just use statistics to tweak their parameters. But what about models based on simple programs? Well, if a model is simple enough, there's a bizarre possibility: one can just search through possible models to try to find it."
>
> [*ibid*]

Wolfram's claims for his 'discoveries' are extensive, but Gray [2003] illustrates the type of criticism Wolfram has received; his review ends with a response to the claim that the book was "introducing a major generalization of mathematics" [Wolfram, 2002, p.7]:

> "In this he is entirely mistaken, but there are at least two ways in which he has benefited mathematics: he has helped to popularize a relatively little-known mathematical area (CA theory), and he has unwittingly provided several highly instructive examples of the pitfalls of trying to dispense with mathematical rigor."
>
> [Gray, 2003, p.17]

Mitchell [2009] likewise welcomes the publicity that Wolfram has brought to CA, but observes that "[r]eactions to the book were bipolar: some readers thought it brilliant and revolutionary, others found it self-aggrandizing, arrogant and lacking in substance and originality" [*ibid*, pp.158-159].

Chopard and Droz [1998] hold that the simple rules used with CA to model the microscopic level of some phenomenon was more intuitive and effective than such traditional approaches as differential equations. Indeed, the potential and usefulness of CA as an alternative to equation-based simulations is put forward by Toffoli

[1984], who argues that CA can act as an alternative to differential equations in the modelling of physics. Vichniac [1984] holds a similar view. He argues that CA can be used successfully in three different, though linked approaches; the "second approach aims at what Stan Ulam has wonderfully called 'imaginary physics' as opposed to 'real physics,' [which is] the object of [the first and third] approaches" [*ibid*, p.97]:

(1) CA as mere computational tools.

(2) CA as fully discrete dynamic systems.

(3) CA as original models for actual physical phenomena, possibly competing with existing continuum models.

A CA defines its own discrete environment, which he sees as having a lot of similarity with the abstract and theoretical space of a physicist [*ibid*]. He concludes that CA are an enhanced form of simulation:

> "In contrast with standard simulations, cellular automata do not only seek a mere numerical agreement with a physical system, but they attempt to match the simulated system's own structure, its topology, its symmetries, in short its 'deep' properties."
>
> [*ibid*, p.113]

An example of a PDE based model leading to a CA version is illustrated by Cohen et al. [2011, 2010]. In [Cohen et al., 2010] the generation of patterns of bristles on the *Drosophila* notum is explored using "a set of coupled differential equations" to simulate the cell signalling process of Delta-Notch mediated lateral inhibition. This work is then built on in [Cohen et al., 2011] by constructing a CA model that they use to explore and analyse the role of noise in the organisation of the bristles, as well as the generation of a range of complex patterns (*see chapter 4 for a detailed outline*).

The next section looks at CA and some of the applications it has been used with.

2.6 Cellular Automata

The birth of CA is attributed to John von Neumann in the early 1950s, following a suggestion of Stanislaw Ulam on how to approach his work on an abstract model of self-reproduction in biology [Berto and Tagliabue, 2012; Chopard and Droz, 1998; Mitchell, 2009; Schiff, 2007; Wolfram, 2002]. The cells on von Neumann's

large two-dimensional grid could each be in one of twenty-nine states [Chopard and Droz, 1998]. Subsequent research using a lattice framework, but not always the term CA, was carried out, but it was not until Wolfram's seminal paper on the "Statistical mechanics of cellular automata" [Wolfram, 1983] that "the first serious study of cellular automata" was published; according to Wolfram, his paper has been cited by over ten thousand articles [Schiff, 2007].

CA consists of a discrete lattice of one to n dimensions made up of cells. Each cell can be in one of a number of defined states at any discrete time step. Although von Neumann used a range of twenty-nine possible states per cell, the move has generally been to simplify it down to a much smaller range, often just two, which can then be seen as analogous to a binary on / off, or active / inactive state. Generally the cells are synchronously updated each discrete time step using simple rules that use the state of neighbouring cells as their input. But the use of asynchronous cell updating is now also used.

The attraction of CA can be found in their relatively simple application to complex phenomena:

> "The CA paradigm is very appealing and its inherent simplicity belies its potential complexity. [...] It has been found that this is an excellent way to analyze a great many natural phenomena, the reason being that most physical processes are themselves local in nature - molecules interact locally with their neighbors, bacteria with their neighbors, ants with their, and people likewise. Although natural phenomena are also continuous, examining the system at discrete time steps does not really diminish the power of the analysis. So in the artificial CA world we have an unfolding microcosm of the real world."
>
> [Schiff, 2007, p.xii]

The order that emerges from underlying complexity in biology and nature can be represented by CA using simple local rules; such as the pattern formation on the combs of honey bee colonies [Camazine, 1991], the behaviour of genes networks [De Sales et al., 1997], and self-organisation [Fatès et al., 2008; Girau et al., 2009]. Chopard and Droz [1998] reflect that our interest is in observing the macroscopic and there is a distinct advantage in the "much simpler microscopic reality" presented by a CA representation of the "complexity [that] comes from a collective behavior rather than some distinctive aspects of the microscopic interactions" [*ibid*, pp.27-28].

The growth in the use of CA to model complex system has also seen an increase in the variety of CAs employed and the formal language used to describe them. One and two dimensional CA have been the predominant CA modelling environments[2]; but there is no theoretical limit on the number of dimensions a CA lattice of cells can have. CA modelling now includes a range of hybrid and coupled environments, which broadens and enhances its efficaciousness, while retaining as much simplicity as possible. The rules used to update the cellular grid can range from very simple, such as used by elementary cellular automata (ECA), to complicated computation used to remap the cellular space; although the former could be viewed as a simulation as opposed to the computation of the latter [Baldwin, 2002].

2.6.1 Dimensions

As the world is three dimensional then it might seem obvious that any model hoping to reflect that world should also be three dimensional. However, the majority of existing CA models are one or two dimensional. Von Neumann's original CA was built on a large two dimensional lattice. Wolfram's work has been principally concerned with one dimensional CA, or ECA as they are often referred to. Indeed, Wolfram [2002, pp.169-221] looks at the use of CA with different dimensions and the correlation with the level of complexity exhibited. He states that traditional science tends to suggest that adding another dimension will increase the complexity of the observed behaviour. But he holds that his examples of ECAs show as much complexity and that ECAs in fact underpin whatever dimension is being used:

> "[O]n a two-dimensional grid one can certainly imagine snaking backwards and forwards or spiralling outwards to scan all the elements. But as soon as one defines any particular order for elements - however they may be laid out - this in effect reduces one to dealing with a one-dimensional system."
>
> [Wolfram, 2002, p.192]

Certainly the output of a one dimension CA or ECA shows the evolving effect of time on the cells within one image, whereas with two and more dimensions a selection of time steps need to be used to show the evolving state of the simulation. But while this can make it easier to show the changing state of an ECA, it is

[2]See the work of Wolfram [1994, 2002] for examples, especially 1D CA; Schiff [2007] is one of the many useful general introductions to CA

harder to link it to a natural phenomenon. In some of our main forms of visual communications, such as drawings and writing, we generally use a two dimensional platform. A leopard is and is seen as a three dimensional object, but when the distribution of the spots on its skin are studied it is usually reduced to a two dimensional representation. This is not only easier and less resource intensive for our own visual observation and evaluation, but also for any computationally representation.

In the modelling, for example, of traffic or pedestrian movement, the rendering of the system into a two dimensional space greatly simplifies the model. Likewise, the modelling of robots or unmanned vehicles moving across the ground and self organising is easier to create rules for, model and compute in a two dimensional representation. Lowe [1987] reflects on how research into computer recognition assumes that human visual recognition matches three-dimensional objects against reconstructed three-dimensional data. Instead, he argues that this is not necessarily "the primary pathway used for recognition in human vision and that practical applications of computer vision could similarly be performed without bottom-up depth reconstruction" [*ibid*]. He suggests "it seems likely that the role of depth recovery in common instances of recognition has been overstated", and that:

> "While it is true that the appearance of a three-dimensional object can change completely as it is viewed from different viewpoints, it is also true that many aspects of an object's projection remain invariant over large ranges of viewpoints (examples include instances of connectivity, collinearity, parallelism, texture properties, and certain symmetries)"
>
> [Lowe, 1987, p.356]

The LEGO construction problem is focused on the development of a computer program that will generate the LEGO building instruction for any real-world object. In his research into the performance of two difference approaches to solving the problem, one of which was a CA based approach, Smal observed that:

> "The traditional approach to solving the LEGO construction problem is to virtually cut a digital representation of the 3D object into horizontal two-dimensional (2D) layers. The problem then reduces to a series of 2D solutions which can be joined together to produce the final 3D LEGO sculpture."
>
> [Smal, 2008, p.4]

Line drawings, sketches and 2D images have been used as a basis for creating 3D objects (see [Lau et al., 2010; Lee et al., 2008; Lipson and Shpitalni, 2007; Tian et al., 2009]). This gives some credence to Wolfram's view that a 3D lattice can be made up of 2D lattices, and a 2D lattice of 1D lattices [Wolfram, 2002].

Computer recognition systems still rely on 2D images, often owing to performance issues:

> "In general, 3D methods are slow and limited in the scale of problems they can address."
>
> [Nicholls et al., 2004, p.452]

and,

> "However, most approaches are still either limited with respect to the degree of 3D modeling, or can not provide competitive performance in terms of 2D B[ounded] B[ox] localization" .
>
> [Pepikj et al., 2015, p.1]

While the visualisation of a 3D object or lattice can be represented at any point in time with 2D lattices, 3D modelling has clear advantages when modelling the movement and interaction of elements or agents operating within 3D space, such as representation of unmanned aerial vehicles swarming or self organising. Although such modelling could seek to learn from the experience of building two dimensional models of, for example, unmanned ground vehicles. Decaestecker et al. [2007], in their review of *in vitro* screening of anti-migratory drugs, also consider the increasing role of 3D substrates compared to 2D ones. These tests are experiments, rather than models, but they illustrate that while 2D tests are easier to run and observe, there are behaviours that can only be seen in 3D tests:

> "The vast majority of cell locomotion experiments are performed for convenience's sake on 2D substrates so that they can be easily observed. [...] Observations of 3D cell cultures have previously shown that when compared to cells cultured on a rigid (possibly ECM-coated) support (i.e., 2D cell culture), certain cell types cultured in a 3D gel exhibit completely different types of behavior in terms of gene expression, proliferation, shape, locomotion, and multicellular organization."
>
> [Decaestecker et al., 2007, pp.156-158]

But, as was stated in section 2.5, computational models are representations of their target systems. So while there are systems that involve interaction within a 3D space, and thus are best represented by 3D models, there are also many systems or their abstractions that can be observed and modelled within a 2D environment.

The simulations used in this thesis have been created with two dimensional models because of (a) their better representation of natural phenomenon than ECA, (b) their easier representation and computation than three dimensional CA models, and (c) there is a much larger body of work using two, rather than three dimensions.

2.6.2 Two dimensional CA

In a two dimensional cellular automaton the state of each cell on a lattice is governed each time step by a rule involving a neighbourhood of cells. The two best known neighbourhoods used with 2D CA are the *von Neumann* which consists of the four cells directly above, below and to either side of the cell in focus; the cell in focus can be included making this a 4 or 5 cell neighbourhood; the second is the *Moore* neighbourhood that also includes the four cells diagonally aligned with the cell in focus, making this an 8 or 9 cell neighbourhood. In 1970 Conway defined a simple rule based on an 8 cell Moore neighbourhood where a cell was either alive (black) or dead (white) based on:

1. birth - exactly 3 live neighbours,
2. survival - 2 or 3 live neighbours,
3. loneliness - live cell with $<$ 2 live neighbours dies, and a dead cell with $<$ 3 live neighbours stays dead, and
4. overcrowding - a live or dead cell with $>$ 3 live neighbours dies or stays dead.

Conway's Game of Life (GoL) is a totalistic rule system, with the boundaries wrapping around to form a torus shape. GoL has received considerable interest, with variations of the rules and the discovery of innumerable 'lifeforms' created by the application of a rule on an initial formation within a two dimensional grid (*see for example Sigmund [1993]*). GoL has been classified as a class IV (complex) system when updated synchronously; asynchronous updating tends to lead to a static state [Schiff, 2007].

Modelling with two dimensional CA is not confined to the GoL; most of the models referred to in Table 2.1 are 2D CA. It has been used to model the movement or flow of particles, traffic and pedestrians, spatial clustering, and the the movement of autonomous agents such as Langton's ant [Chopard and Droz, 1998; Schiff, 2007]. Stuart Kauffman from the 1960s has worked on Random Boolean Networks (RBN) [Kauffman, 1991], which can be seen as an extension of CA. They are sometimes referred to as Classical Random Boolean Networks (CRBNs). The nodes and links in an RBN are comparable to a network or a spread out, disjointed lattice that is updated in discrete time steps, just like a CA. Apart from the randomness of the connection of the nodes, RBN also differ from traditional CAs by each node having its own set of rules, which can be different to each other node [Mitchell, 2009]. RBNs can have two dynamic phases - ordered and chaotic [Gershenson et al., 2010]. The deterministic, discrete nature of an RBN means that eventually over a series of time steps the network will evolve towards either a point or a cyclic attractor [Gershenson et al., 2004]. Gershenson has outlined a series of RBNs that have different combinations of other characteristics [Gershenson, 2002, 2004; Gershenson et al., 2003].

1. Asynchronous RBNs (ARBN) have asynchronous and non-deterministic updating - same characteristics as CRBNs, but their updating is asynchronous and random,

2. Generalised ARBNs (GARBN) differ from ARBNs as it is semi-synchronous and non-deterministic in their updating - same as ARBNs that can update more than one node at a time,

3. Deterministic ARBNs (DARBN) have asynchronous, non-deterministic updating - same as ARBNs except the nodes are not updated randomly,

4. Deterministic GARBNs (DGARBN) is the semi-synchronous, deterministic updating - same as ARBNs except more than one node can be updated at any one time and the updates are not random, and

5. Mixed-context RBNs (MxRBN) are non-deterministic in a particular way - they are DGARBNs with a selection of contexts of which one is randomly selected and applied at a regular time steps.

Gershenson et al. [2004] looked at the different types of RBNs and their updating schemes when different initial conditions were used (i.e. sensitivity to initial conditions). Their findings were that "the phase transition between 'ordered' and 'chaotic' regimes of the networks" were unaffected by updating scheme used.

Gershenson has looked at some factors that can be exploited to manipulate the various strains of RBNs:

1. through using the probability and connectivity in moving the phase space, and

2. through topology, modularity, redundancy and degeneracy in the broadening of the critical regime [Gershenson et al., 2010].

2.6.3 Updating

The standard way of updating a CA between time steps is to synchronously update in parallel each cell on the grid. This means that the updating rule is run for each cell within the context of the existing time step and the cell in focus is not aware of any new updates made to cells in its neighbourhood. Synchronous updating of the cells usually has a fixed order, such as top left cell of the grid and then incrementally column by column, row by row. However, as the new state of a neighbouring cell is not used in the updating of a cell the order of cell update is irrelevant.

The main difference with asynchronous updating is that the updated state of a cell is immediately available for use in any subsequent cell update. In this way the order of updating has significance. In their reflection on asynchronous CA Bandini et al. [2010] outline a number of update schemes. There are two themes to the updating: (a) the order of the updating - either random each time step or a fixed, cyclic order; and (b) the range of cells updated in a time step - either all or a random selection. In the case of random selection there is no guarantee that every cell will be updated in each time step.

Schönfisch and de Roos [1999] categorise asynchronous updating into four types, the first three of which have already been mentioned above:

- random fixed sweep - this is equivalent to a *fixed cyclic order* involving the asynchronous updating of all the cells on the grid during a time step. The order is randomised before the first time step, but then remains constant through each subsequent time step driven update;

- random new sweep - this matches a *random order* update, where each cell is asynchronously updated in a time step, but the order is randomised prior to each time step;
- uniform choice - this is the same as a *random selection* of cells for asynchronous update each time step. This is often illustrated as randomly updating one cell per time step. After a series of time steps equal to the number of cells on the grid each cell will have been updated once, or multiple times, or not at all; and
- Time driven - this is not mentioned above. It involves each cell having, in effect, an alarm clock that has a randomly selected setting for when it will go off and the cell will be updated. Once it has been updated the wake up time is randomly set again.

Schönfisch and de Roos [1999] considered that the choice of asynchronous updating method was important. They argued that the exponential time driven updating method was "most satisfying from a theoretical and from a biological point of view as it is justified by a deviation from continuous time processing and introduces least undesirable structure" [Schönfisch and de Roos, 1999, p.140]. However, from the three other methods, uniform choice is the most similar to time driven updating, and a lot less complicated to implement. This would also tie in with the idea of "random interaction" used in the definition of complexity given in subsection 2.2.4.

There seems to be varying views on whether synchronous or asynchronous updating should be used. The argument that asynchronous updating is intuitively more similar to how physical phenomena operate in real life is made on the basis that "there is no universal clock in Nature" [Schiff, 2007, p.106]. Wolfram comments on the same issue:

> "Yet just as it seems unreasonable to imagine that the universe consists of a rigid grid of cells in space, so it also seems unreasonable to imagine that there is a global clock which defines the updating of every element in the universe synchronized in time."
>
> [Wolfram, 2002, p.486]

It is not guaranteed that a different output will be created by changing from one scheme to the other. Boerlijst and Hogeweg [1991] modelled spiral wave structure in pre-biotic evolution and found:

> "It turns out that the precise definition of the cellular automaton does not affect the development of the spirals. We examined for instance asynchronous updating of the cells, a non-toridal field and other neighbour cells that can give catalytic support".
>
> [Boerlijst and Hogeweg, 1991, p.25]

But generally, the choice of asynchronous updating produces significant difference in the output of a cellular automaton [Schiff, 2007, pp.118-121]. In his review of asynchronous cellular automata Fatès [2013] notes that if asynchronous updating is used the GoL can stabilise on a fixed pattern. Wolfram refers to sequential updating as a synonym of asynchronous updating and also highlights the potential difference in output from the two schemes, "[t]aking the rules for an ordinary cellular automaton and applying them sequentially will normally yield very different results" [Wolfram, 2002, p.1035]. Although his main focus is synchronously updated ECAs, he does look at sequential updating. One of these forms is his mobile automata in which one active cell is updated per time step. These mobile automata can produce complex behaviour, but it is very rare compared to the complex output of ECA. [Wolfram, 2002, pp.71-77].

Although asynchronous updating might seem more natural, there is an argument that the choice of update scheme depends on the scale that is applied to the observation of the natural phenomena:

> "the difference between synchronous and asynchronous update is a question of how we look at the (real) process."
>
> [Schönfisch and de Roos, 1999, p.140]

If the time scale is large, then on average all the cells will be updated and it can be seen as a synchronous process. But if the time scale is reduced so that only a single or small number of cells can be updated, then an asynchronous updating scheme will better represent the phenomenon. The difference produced by the scheme used can be the difference between a model that works and has relevance and one that is an artefact and has no epistemic worth. Fatès et al. [2008] created a CA simulation based on the the aggregation of the Dictyostelium discoideum cellular slime mould using synchronous updating with the intention of looking at the gathering of agents by combining reaction-diffusion and chemotaxis (*see chapter 4 for a detailed outline*). His choice was based on the fact that asynchronous updating produced unwanted types of excitation waves:

> "The advantage of reaction-diffusion over classical diffusion is that waves propagate without attenuation and over arbitrary large distances; the drawback [...] is that messages have to be relayed synchronously and perfectly in order to prevent the creation of self-entertained excitation waves."
>
> [Fatès et al., 2008, *ibid*, p.24]

A collection of works [Grilo and Correia, 2008; Huberman and Glance, 1993; Newth and Cornforth, 2009; Saif and Gade, 2009] investigating the historic use of synchronous updating in models of the Prisoner's dilemma, compared to a range of asynchronous schemes, all concluded "that many of the previous equilibrium states are mere artifacts of a synchronous updating on a regular lattice" [Fatès, 2013, p.22]. It should be remembered that a CA is not a real system, but a representation, or an imaginary system encapsulating essential and relevant abstractions of a real phenomena where the CA "are a caricature of the real world rather than its portrait" [Chopard and Droz, 1998, p.28]. Synchronous updating does have the potential advantage of being simpler to code. It can also produce noticeable manifestations in the CA output more quickly [Fatès, 2013]. But establishment that simple programs with simple rules can produce interesting or even complex artefacts is of no real value apart from supporting the concept that just as order can arise from complexity, so complexity can emerge from simplicity.

But just as there is no clock guiding nature, it is equally true that events can happen synchronously, rather than always asynchronously [Fatès, 2013]. In this way it is possible that a mixture of synchronous and asynchronous updating should be used to model natural phenomenon, rather than one or the other:

> "Probably neither a completely synchronous nor a random asynchronous update is realistic for natural systems."
>
> [Radicchi et al., 2007, p.1]

The degree and depth of representation and the level of epistemic value desired will determine the composition of a CA simulation. If the image created on the lattice is the sole focal point of interest, then how it is created is of little, if any relevance; even if it is an artefact of the modelling process. But if the CA simulation is intended as an opaque thought experiment, or as part of the modelling and experimental scientific process, then the basis of the CA model, how it is updated, the nature and origin of its rules, and the elimination of any unwanted artefacts are all part of what contribute to the appropriateness and worth of the simulation.

2.6.4 Coupling CA, Hierarchical CA and Hybrid models

There has been an increasing amount of research into modified CAs including,

1. **Coupled CA** that link together one dimensional CA [Alonso-Sanz and Bull, 2008; Bull and Alonso-Sanz, 2008; Grassberger, 1999; Maignan and Gruau, 2008a]

2. **Higher order cellular automata**, arising from work on hyperstructure [Baas, 2009a,b; Baas et al., 2004], have been proposed where second order CAs (2-CA) can use more than one rule within a one dimensional CA grid [Helvik, 2006].

Mori et al looked at a CA based on Rule 22 and where each cell had eight receptors; the binary representation of Rule 22, 00010110, is used so that each one of the binary units was one of the receptors for a cell, with the '1's being masters or independent and the '0's being slaves that take on the values of the master receptor based on a probability formula, such that the first receptor (f000) could be changed to a 1, making the rule enacted on the cell into rule 23 (00010111) [Mori et al., 1998].

Zuse introduced the "net automaton", which is built on a grid system where each cell contains a complete calculating system. These single cell "calculating systems contain both information-processing and information-storing elements" [Zuse, 1970, p.91]. In the net automaton the role of a cell is to process information. Sarkar [2000] uses the term *hybrid* for a CA where each cell has its own rule; a cell can change these rules at each time step - this is called programmable CA or tessellation automata. In the latter, a cell can be seen to have a finite set of rules and the input determines which one gets selected. Sarkar outlined a one-way CA where the rule depends on the cell itself and either the left or the right cell - but not both. So the flow of information is one sided. This has been used "in the development of many easy-to-implement systolic algorithms" [Sarkar, 2000, p.88]. Tissera et al. [2007] have a different take on hybrid. Their evacuation simulation was based on models of traffic and pedestrian usage, but they decided that while CA was very good in modelling the local interaction and environment, a further, maybe different model was needed to show the higher system perspective. Their approach was to use CA to model the dynamic spread of fire and smoke, while adopting a goal oriented intelligent agent model to simulate the reaction of people within the environment.

2.6.5 Modelling and simulating diverse systems using Cellular Automata

Tamulonis [2013, p.9] considered the modelling of biological cells, pointing out that within biology

> "Cell-based models have become an important theoretical tool for understanding and predicting the behavior of cellular scale systems."

As he demonstrates, the range of cell-based models is extensive, including non-lattice based models, as well as lattice based models of different dimensions and shape. One category of lattice based models has been used across a wide range of systems. As already mentioned, CA are a discrete abstraction of a physical system in which the physical elements are represented by a finite set of values, often revealing dynamics that are lost in the continuous system such as a differential equation [Chopard and Droz, 1998]. The initial focus was on using CA to model physics, but the range of their application has grown over recent years, (*see Table 2.1*).

Table 2.1: Some examples of the range of domains modelled with CA

Area	**Domain**
General applications	Cryptography [Oliveira et al., 2008; Wuensche, 2008]; DDoS detection [Lawniczak et al., 2008]; data clustering algorithms [Chen et al., 2004; de Lope and Maravall, 2013; Moere and Clayden, 2005; Moere et al., 2006]; computing [Adachi et al., 2004; Mamei et al., 2005; Mitchell et al., 1994]; LEGO construction problem [Smal, 2008]; image processing [Adamatzky, 1996]; robotics [Behring et al., 2000; Ioannidis et al., 2008; Magg and te Boekhorst, 2006; Scheidler et al., 2006]; robot path planning [Behring et al., 2000]; agricultural price volatility [Chen and Wang, 2007]; anticipation [Kier and Cheng, 2000]; differential equations [Toffoli, 1984]; heat transfer [Burzyński et al., 2004]; self-replicating structures [Reggia et al., 1998]; distance detection [Maignan and Gruau, 2008b]

continued on next page

Area	Domain
Biology & Medicine	Cardiology [Makowiec, 2008]; immune system and viruses [Bernaschi et al., 2000; Ermentrout and Edelstein-Keshet, 1993; Sloot et al., 2002; Zorzenon dos Santos and Bernardes, 1995]; tumour growth [Naumov et al., 2011; Wcisło et al., 2009]; genetics [De Sales et al., 1997; Xiao et al., 2006]; in-stent restenosis [Caiazzo et al., 2009]; complex biochemical systems [Kier and Witten, 2005], nervous system [Luthi et al., 1998]; cell mechanics [Tamulonis, 2013]; delta-notch signalling [Cohen et al., 2011; Ghosh and Tomlin, 2001]; formation of ocular dominance stripes [Swindale, 1980]; oscillator synchronisation [Murray et al., 2013]; swarm systems [Adamatzky and Holland, 1998; Wright et al., 2000]; biofilm growth structure [Picioreanu et al., 1999]; skin patterning [Young, 1984]; self-organisation in bee combs [Camazine, 1991]; pheromones and biological traffic phenomena [Chowdhury et al., 2005; van Dyke Parunak and Brueckner, 2001]; ant based clustering algorithm [Chen et al., 2004]; myxobacteria [Stevens, 2000]; growth of branching in fungi [Edelstein-Keshet and Ermentrout, 1989]; vessel morphogenesis [Markus et al., 1999]
Traffic flow	Traffic flow [Benjaafar et al., 1997; Dupuis and Chopard, 2001; Gershenson and Rosenblueth, 2009; Nagel and Rickert, 2001; Schaefer et al., 1998]; pedestrian flow [Burstedde et al., 2001; Dijkstra et al., 2000; Franca et al., 2009; Makarenko et al., 2008; Schadschneider, 2001; Schultz and Fricke, 2010; Schultz et al., 2010]; evacuation simulation [Poudel et al., 2009; Tissera et al., 2007]; transportation systems [Topa et al., 2006]
Social science	Social dynamics [Hegselmann and Flache, 1998; Helvik, 2006]; urban planning and growth [Al-Ahmadi et al., 2009; Aljoufie et al., 2013; Barredo et al., 2003; Cheng and Masser, 2004; Clarke and Gaydos, 1998; Torrens, 2000; Tsompanas and Sirakoulis, 2012; Yang et al., 2013]

continued on next page

Area	Domain
Earth science	Earthquakes [Georgoudas et al., 2007; Malamud and Turcotte, 2000]; forest fires [Malamud and Turcotte, 2000]; snow avalanche [Avolio et al., 2010]; strength properties of heterogeneous coal-beds [Zavsek et al., 2005]
Physics & Chemistry	Lattice Boltzmann [Chopard et al., 2002; Hatzikirou and Deutsch, 2010; Kusumaatmaja and Yeomans, 2010]; neural activity [Marro et al., 2007]; snow crystals [Reiter, 2005]; Browning motion [Lee and Peper, 2008]; Laser dynamics [Guisado et al., 2005]; diffusive and reactive processes [Pintus et al., 2011]

How it is applied has also grown significantly; classical local rule based CAs have been supplemented with hybrid CAs [Janssens, 2010; Zavsek et al., 2005], complex CAs (CxA) [Kroc et al., 2010], agent based CAs [Chen and Wang, 2007; Franca et al., 2009; Gruner, 2010; Poudel et al., 2009], and dissipative CAs [Zambonelli et al., 2003], to list some of the variations that have been created. Its potential for modelling and simulating local interactions in a relatively simple way makes it a very useful modelling tool. The concept of a clusters of models can be seen in how the various facets of, for example, pedestrian movement is abstracted into models of flow, obstacle and collision avoidance, quickest path, bottleneck and evacuation management. The teasing out of analogous systems involves more thought. Obviously traffic flow and pedestrian flow have many basic level similarities; although traffic flow follows more explicit rules in its use of lanes, overtaking and general traffic regulations. A possible approach is to classify the CA models in a classification of model types that removes the emphasis on domains.

Ermentrout and Edelstein-Keshet [1993] proposed three broad categories of CA models within biology.

1. Eulerian models (deterministic automata)
    - discrete lattice of points (cells), usually two-dimensional.
    - has completely determined rules with no use of probability to determine the state of the cell
    - updates synchronously

- the focus is on the changing state of each cell, rather than the tracking of a cell or agent across the CA grid
- also referred to as "Eulerian" due to analogy with fluid mechanics.
- example: skin patterns [Young, 1984];

Young [1984] uses a grid with its rules based on an activator-diffusion theory. The grid is made up of a mixture of coloured and uncoloured cells. Each cell can diffuse inhibitor and activator morphogen to its neighbourhood. The morphogen, at the next time step, can be dispersed to the next set of neighbours. The strength, and thus range, of the inhibitor morphogen is greater than the activator, thus giving short-range activation and long-range inhibition, resulting in connected groups of coloured cells. In Figure 2.4 the results of decreasing the range of the inhibitor is shown. As it decreases the spot patterns seen on the left of the figure begin to connect up to form a pattern of stripes, as seen on the right of the figure.

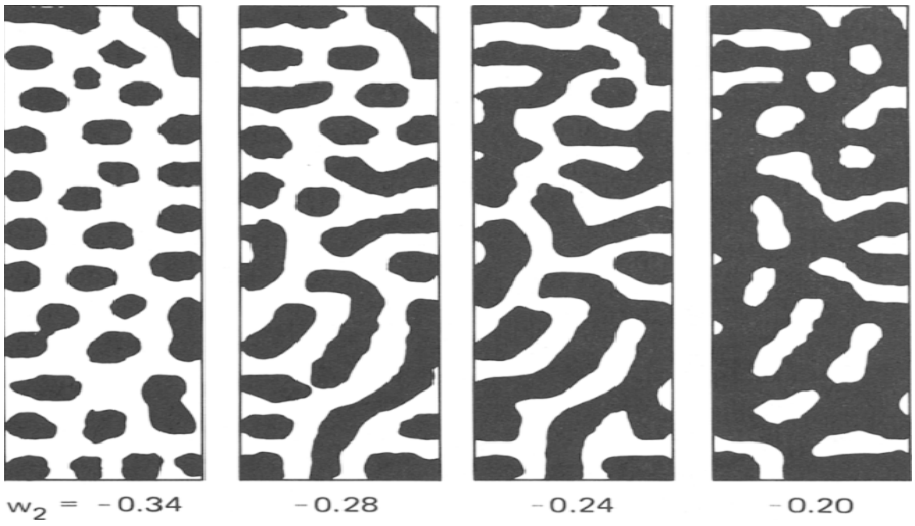


Figure 2.4: An example of a deterministic automata: skin patterns created via a CA using a activator-inhibitor scheme. The activation parameter is set to 1.0, while the range of the inhibitor parameter starts at -0.34 on the left and gradually 'decreases' towards zero. This results in more cells being activated and becoming coloured pigment cells; consequently the spots seen on the left gradually turn into a pattern of stripes on the right of the figure (from [Young, 1984, p.54]).

2. Lattice gas models (particle automata)

   - particles move randomly or deterministically across a discrete lattice
   - they undergo state changes if they collide
   - a cell might be empty; a particle moves into and binds to a cell, but in this type it is reversible
   - example: pedestrian flow in room evacuation [Burstedde et al., 2001]

   In this example of a particle automaton the grid represents the floor space and each cell can either be empty or occupied by exactly one particle (pedestrian). Burstedde et al. [2001] set the size of the cell to 40 *by* $40cm^2$, which they use to represent the typical space that a pedestrian takes up in a crowd. The pedestrian can move one cell at a time in a non-diagonal move, so a von Neumann neighbourhood is used. The update is completed in parallel for all pedestrians. The basic rules give a directional preference to each pedestrian and at each time step evaluates the probabilities for a move of each pedestrian. Once the desired move of each particle is decided, then if the target cell is unoccupied and no other cell is contending for the target cell, the pedestrian moves to the target cell. If there is any contention between two or more pedestrians then the choice of which one make the move, while the other stays where they are, is based on the relative probabilities used to make the original choice of target cell. The model can also use a *dynamic floor field* where the pedestrians leave a trail that decays over subsequent time steps; this allows for interaction between pedestrians, especially in situations like the evacuation of a room, as shown in Figure 2.5.

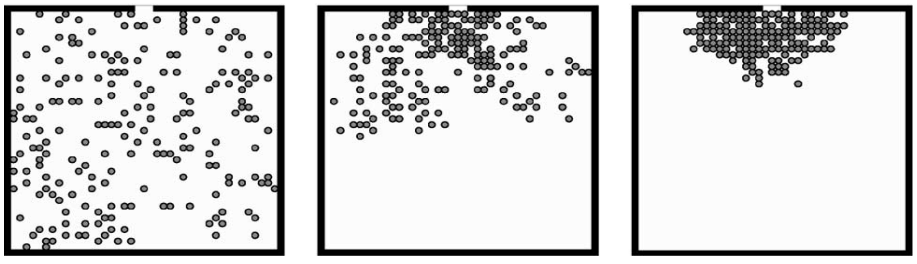


Figure 2.5: Particle automata example: the simulation shows three stages in the evacuation inspired movement of a crowd within a room with one exit (from [Burstedde et al., 2001, p.517]).

3. Solidification models (growth automata)

   - only sites adjacent to pre-existing 'cells' can be populated by new cells
   - once occupied a cell stays occupied
   - models are local (i.e "has a finite velocity of propagation which is proportional to the rate of growth of the cells" [Ermentrout and Edelstein-Keshet, 1993, p.126]
   - example: vessel morphogenesis [Markus et al., 1999]

   The growth automata example is the simulation of morphogenesis in [Markus et al., 1999]. They employ an algorithm made up "of a list of simple rules describing the essential biophysical features, permitting comfortable programming and fast computations" [*ibid*]. In Figure 2.6 a simulation is shown representing the lateral branching vessels observed in insect trachea. The authors' used simple rules with four variable parameters - u (activator) and v (inhibitor) that represents a morphogenetic activator-inhibitor scheme; g acting as a genetic switch; and s that allows the type of substrate being simulated to be set, thus reflecting different growth patterns. The results were held to be better than those produced by PDE driven automata, (for example [Edelstein, 1982; Meinhardt, 1976]), as they not only showed anastomosis where separate parts of a branching system is connected, but also "the isotropic, quasi-disordered growth, that is observed in nature" [Markus et al., 1999, p.204], but was not produced by the PDE based automata.

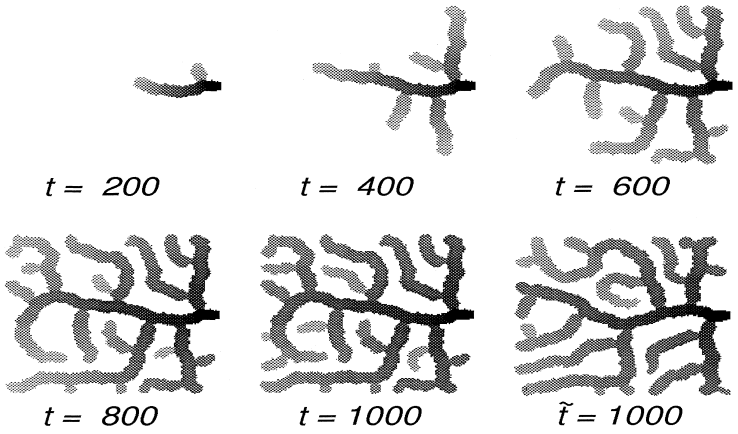


Figure 2.6: Growth automata example: this shows the simulation of lateral branching vessels, such as those observed in insect trachea; a square lattice of 150 by 150 cells is used. The last two images are both after 1000 time steps, but $\tilde{t}$ is from a second simulation using different initial settings (from [Markus et al., 1999, p.199]).

Some examples, such as in urban growth and usage models, while they might seem to be growth models, do not fully match a strict interpretation of the criteria; the status of cells are often driven by other rules, such as the existence of roads, and their usage can change. Consequently, as the state of a cell is not irreversible and can be changed, they do not belong in the growth classification. It might be that the examples given of the characteristics of a growth model are too simplistic and needs expanding to take into account the varied combinations and types of CAs used nowadays. But if they are left as they are, then it is understandable, even when based on the small selection of 2D CA examples in table Table 2.2, that there is a greater variety of scenarios that open themselves up to the other CA types rather than growth automata.

In their second category, *particle automata*, Ermentrout and Edelstein-Keshet [1993, p.116] initially refer to the collision based model of lattice gas. This would preclude many non biological CA that are based on the movement of one or more agents across the CA grid, such as pedestrian and vehicle models, which could be more appropriately termed *collision avoidance* models, or *agent* models. However, they go on to give examples that include self-organisation of ant trails. Consequently, the type is not split, but includes non-collision based particle models such as vehicle (including unmanned vehicles), pedestrian models and swarm models, basic flow models, threat models, (including evacuation where humans can exhibit herd behaviour), obstacle and navigation models.

The wide usage of CA as a modelling and simulation tool means that the categorisation can be applied to models outside of the biological domain (*see table Table 2.2*). Key to the definition of a deterministic CA are synchronous updating, deterministic rules and the absence of randomisation either in the form of an asynchronous updating mechanism or in some level of probability used to influence the setting of the state of the cell. The latter should not be confused with the deterministic setting of a probability value for a cell, such as when determining possible land use based on a cell's neighbours (*e.g. Barredo et al. [2003]*). This presents a slight dilemma when considering, for example, the delta-notch signalling used by Cohen et al. [2011] or the the fuzzy based urban dynamics model devised by Al-Ahmadi et al. [2009]. The former models the changing state of the cells, determined by their neighbourhood, but uses an asynchronous updating system and also probability to mirror noise in the update rules (*see chapter 4*). Such CA that update the state of their cells either using an asynchronous update method, or a probabilistic updating rule, or both, is grouped in a new type called *randomised* (*see Table 2.2*).

Deterministic automata has been widely used to model a range of excitable media. The modelling of spiral wave structure pre-biotic evolution in [Boerlijst and Hogeweg, 1991] would have been grouped within in this category, except for the use of probability in its rules. Boerlijst and Hogeweg [1991] illustrate how the outcome of evolutionary processes can be greatly altered by the spatial self-structuring in partially mixed media. They use a 300 by 300 square grid with each cell either occupied by a molecule of a certain species or empty. The rules use a cell's neighbourhood to determine on one of three processes that decide whether (1) a cell occupied by a molecule *decays* and becomes empty, (2) an occupied neighbour of certain species *replicates* into an empty cell, or (3) an empty cell becomes occupied through a process of *catalysis* where an occupied neighbour of any species replicates into the cell if a catalytic molecule also occupies one of the adjacent neighbouring cells. A *diffusion* process is also used between time steps. The grid is divided into subfields of 2 by 2 cells; probability is used to rotate each subfield 90° clockwise or anti-clockwise at each diffusion step. The subfileds are shifted one cell diagonally after the diffusion process. This use of probability in the second part of the update process places the model in the new randomised category. The wiping out of a parasite is shown in Figure 2.7, as well as where a parasite persists like a cyst.

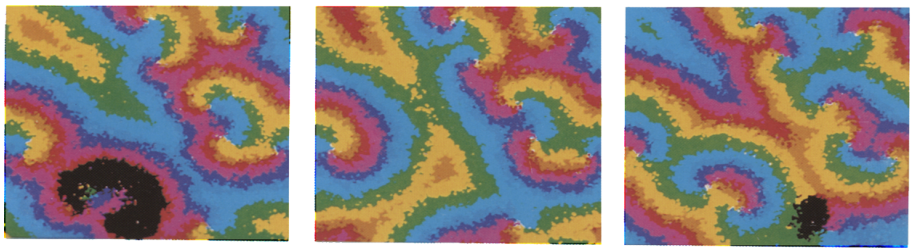


Figure 2.7: An example of a randomised automata: the effect of self-structuring spiral wave on the outcome of evolutionary processes. In plate 4a, at time step 110, the centre of a double spiral is invaded by parasites (black cells); in plate 4b, at time step 550, the parasites have been wiped out by other spirals. In plate 5 the parasitic invasion at the centre of a single spiral still remains as a cyst after 600 time steps (from [Boerlijst and Hogeweg, 1991, p.23]).

Table 2.2: Categories of 2 D CA types using entries in table Table 2.1

CA Type	Domain
Deterministic	excitable media [Adamatzky and Holland, 1998; Gerhardt et al., 1990; Weimar et al., 1992; Yan and Yuan, 2000]; instent restenosis [Caiazzo et al., 2009]; delta-notch signalling [Ghosh and Tomlin, 2001]; formation of ocular dominance stripes [Swindale, 1980]; Immune system and viruses [Bernaschi et al., 2000; Zorzenon dos Santos and Bernardes, 1995]; skin patterning [Young, 1984]; Browning motion [Lee and Peper, 2008]; lattice gas [Pomeau and Frisch, 1986]; urban land use [Aljoufie et al., 2013; Barredo et al., 2003; Cheng and Masser, 2004]; earthquakes [Georgoudas et al., 2007]; natural hazards (forest fires, earthquakes and landslides) [Malamud and Turcotte, 2000]; snow avalanche [Avolio et al., 2010]; strength properties of heterogeneous coal-beds [Zavsek et al., 2005]; LEGO construction problem [Smal, 2008]; agricultural price volatility [Chen and Wang, 2007];
Randomised	excitable media [Boerlijst and Hogeweg, 1991]; Cardiology [Makowiec, 2008]; delta-notch signalling [Cohen et al., 2011]; oscillator synchronisation [Murray et al., 2013]; immune system and viruses [Sloot et al., 2002]; genetics [De Sales et al., 1997]; nervous system [Luthi et al., 1998]; neural activity [Marro et al., 2007]; laser dynamics [Guisado et al., 2005]; diffusive and reactive processes [Pintus et al., 2011]; urban land use and growth [Al-Ahmadi et al., 2009; Clarke and Gaydos, 1998]; evacuation simulation [Tissera et al., 2007]; transportation systems [Topa et al., 2006]; computing [Adachi et al., 2004; Mamei et al., 2005]; myxobacteria [Stevens, 2000]; anticipation [Kier and Cheng, 2000]; selz-organisation in bee combs [Camazine, 1991]; ant based clustering algorithm [Chen et al., 2004]
Growth	growth of branching in fungi [Edelstein-Keshet and Ermentrout, 1989]; biofilm growth structure [Picioreanu et al., 1999], snow crystals [Reiter, 2005] ; vessel morphogenesis [Markus et al., 1999]

continued on next page

CA Type	Domain
Particle	Lattice Boltzmann [Chopard et al., 2002; Hatzikirou and Deutsch, 2010; Kusumaatmaja and Yeomans, 2010]; pheromones and biological traffic phenomena [Chowdhury et al., 2005; van Dyke Parunak and Brueckner, 2001]; swarm systems [Adamatzky and Holland, 1998; Wright et al., 2000]; traffic & pedestrian flow [Burstedde et al., 2001; Dijkstra et al., 2000; Franca et al., 2009; Makarenko et al., 2008; Nagel and Rickert, 2001; Schadschneider, 2001; Schaefer et al., 1998; Schultz and Fricke, 2010; Schultz et al., 2010]; aircraft evacuation simulation [Poudel et al., 2009]; data clustering algorithms [Chen et al., 2004; Moere and Clayden, 2005]; urban land use and growth [Tsompanas and Sirakoulis, 2012; Yang et al., 2013]; robotics [Behring et al., 2000; Ioannidis et al., 2008; Magg and te Boekhorst, 2006; Scheidler et al., 2006]

It should also be noted that a system can be modelled under more than one type, especially when considering the cell updating types of deterministic and randomised. This is understandable as the randomised automata are deterministic automata that also used probability and / or apply rules randomly. Delta-notch signalling, for example, has models in both CA types. The simulation of delta/notch in [Ghosh and Tomlin, 2001] is based on the state of a cells neighbours and is classified as a deterministic automata; whereas [Cohen et al., 2011] also involves the use of probability, both in the asynchronous updating scheme and in the perturbation of noise in the signalling process, and is therefore classified as a randomised automata. This both exhibits the versatility of CA modelling and illustrates the role of the modeller in deciding how a system will be modelled and, subsequently, measured.

2.7 Measurement

It has already been mentioned that the process of measuring a system can impact on the system (*see subsection 2.4.1*). It can also be a subjective process, especially if a substantial part of any argument is based on a visual pattern output by a computer simulation. The choice of modelling technique can either determine the measuring technique or be determined by it. While the results of an experiment can be unexpected, the process is usually embarked upon with some idea of what is being sought and the best means of representing the anticipated output.

This expectation influences the choice of modelling environment and measuring technique. Ermentrout and Edelstein-Keshet [1993, p.97] describe CA as, "[m]ore properly, a cellular automaton consists of a simulation which is discrete in time, space, and state". A CA provides a very visual representation of the changing state of the CA grid, based on localised interactions or rules. This immediately suggests the observation and measuring of density, clustering and pattern formation or dissipation, and the tracking of particles or agents across the CA grid. While this is not a definitive list, it gives a clear example of why CA might be chosen, apart from the general ease of modelling with CA compared to other techniques.

But there seems to be no specific alignment of different measuring techniques to the three categories of models outlined above. Even though the 'tracking of particles or agents across the CA grid' refers specifically to particle models, in a more general sense such things as path-finding problems can be accomplished via a growth model, such as the plasmodium of the slime mould *Physarum polycephalum* [Tsompanas and Sirakoulis, 2012]. The role of measuring is not just to feed information into a predictive process as observed input into a black box. It can also be the technique that provides an indicator of the present and impending state of the system, through the measuring and comparison of the linear system output or state. There is obviously an observer interest in 'why' a system is measured, which in turn can influence what and how something is modelled and measured. This is tied into what the observer wants to predict, whether it is the throughput of a network and any indication of bottlenecks in the flow of traffic through any of the nodes on the network; or the rate of flow of traffic on a motorway; or the spread of a virus, or a forest fire. It might be a relatively straight forward indication of a growth prediction based around the consumption of available resources; or the more complicated tracking of indicators, whether it is labelled as emergent, self-organising or in some other way to signify a significant state change. The latter is often linked to 'unexpected' behaviour or change, so modelling is a means of unravelling, or at least finding a way to predict to some degree, the unexpected.

CA models are often used to gain information for use in other domains; such as the formulation of data clustering algorithms by modelling ant behaviour [Chen et al., 2004; Moere and Clayden, 2005; Moere et al., 2006]; or the modelling of *Physarum's* behaviour to create a "powerful low-cost virtual laboratory" that can be used in "solving the path-planning problem by guiding the development of adaptive networks as in the case of the actual rail network of Tokyo, Japan" [Tsompanas and Sirakoulis, 2012, p.17. author's italics]. These provide examples of a single model being used for two different, but analogous systems. This ana-

logy is centred on the algorithm extracted from the model of one system and employed in another. So it is not based on any predictive measurements, but on the behaviour and efficaciousness of the model. Well known measures, such as Shannon's or Boltzman's entropy, or slight variants of them, have been used (*see subsection 2.7.2*). This can be to provide a specific measure of the system being modelled, or to propose a general indicator of, for example, complexity or entropy, both of which can best be described as compound terms, having many definitions. But the analysis of a single model for a selection of analogous systems is more likely to provide any possible generic indicators. Such models need to be at a low enough level to include as many systems as possible, without becoming meaningless as a predictive tool. A sensible starting point is within a category or set of models.

2.7.1 Measuring CA output

The measurement of a 2D CA can evaluate a number of features, such as density, clustering and patterns. Density is relevant not only in the size of the active cells within a growth automaton, but also in setting the significance of any active cells in terms of the overall occupied and unoccupied CA cells. Clustering algorithms are used to organise data as well as to identify clusters of data. Various algorithms exist (e.g., k-means clustering, hierarchical clustering). Pattern identification can range from a subjective visual evaluation to one involving concepts of the level of randomness in the data structure under observation. Different methods, or different applications of measuring methods can have more relevance to particular CA types. Mean density can have greater relevance to the tracking of any changes in the number of active cells in deterministic automata. Clusters can be identified in deterministic automata, such as with spanning clusters in percolation theory [Essam, 1980; Hoshen et al., 1997; Kier et al., 1999; Wilkinson and Willemsen, 1983]. Likewise, clustering techniques can be used in the classification of the state of agent based CA and self-organising systems. Patterns forming and dissipating can be associated with the degree of randomness in data representation of the CA output space.

So the choice of simulation and modelling technique is linked to the preference of measuring method. There obviously has to be something considered worthy of measuring. That measurement will have limited worth if it just tells us something about the model. Ideally it will provide some information related to the system under focus or help in some predictive process related to the system.

2.7.2 Entropy and measuring emergence in CA

The characteristics of complexity and emergence have been discussed in previous sections. There are a number of views of how emergence manifests itself to the observer. Standish [2001] cites the view that surprise is a good test for emergence (*see Ronald et al. [1999]*), but this is very much a subjective view from the observer and not very measurable. The loss of randomness has been held up as an indication of emergence, in terms of high level structure or stability [Boschetti et al., 2005; de Silva, 2004]. The proposed measures of complexity, (see section 2.7), offer different scales to measure for emergence, providing one holds that a lessening in the complexity of a system can be attributed to the emergence of order, or a loss of randomness. Coe et al. [2008] looked at producing random CA with probabilistic and deterministic rules not only to give a better perspective on random behaviour, but also to facilitate the analytical solution of CA models. Crutchfield argued that randomness was not by itself the most useful indicator; he argues that the interaction of order and randomness is what makes a system either complex, or merely complicated. He used the example of "ordered" ice and "random" water both being constituted from the same matter (H_2O) that is at its most complex when it is a mixture of both; this suggests that phase transition is possibly one way of identifying complex thermodynamic states [Crutchfield, 1994].

An accepted measure of the state of a thermodynamic system is its level of entropy. The 2nd law of Thermodynamics dictates that entropy will increase with time in a closed system [Hitchins, 2003a]. This involves a balance of energy, therefore for something to emerge, there has to be some activity; so in order to move to some stability, you need to push things out of balance [Åm, 1994]. This raises the question of where the energy needed to create order / emergence goes in order to satisfy the 2nd law. A view put forward proposes the 2nd law of Thermodynamics is satisfied by order (loss of entropy) in the macro level being balanced by an increase in entropy in the micro level, which acts as a "sink" [Van Dyke Parunak and Brueckner, 2001]. Lambert states how, given the opportunity, "energy spontaneously tends to flow from being concentrated in one place to becoming diffused or dispersed or spread out"; this is not inevitable, it is a tendency rather than a prediction and introduces a measure of bounded stability [Lambert, 2010]. Lambert [2010] goes on to point out that "[o]ur psychological sense of time is based on the second law. It summarizes what we have seen, what we have experienced what we think will happen", leading to the 2nd law sometimes being referred to as "time's arrow".

The change in a system as a consequence of the second law, resulting in energy being dispersed, can be measured in the entropy change [Lambert, 2010]. In this sense what is needed is an understanding of what represents stability and randomness in a system, and a way to map the phase transition of the system over a period of time. Hitchins offered a definition of stability:

> "A set of interacting systems, itself constituting an open system, may be said to be stable when, over a period of interest, its net configuration entropy tends to a constant value" [Hitchins, 2000].

The term *configuration* refers to "the set of every states at a given time" [Maignan and Gruau, 2008a]. There are different interpretations of entropy, depending on the system view being taken by the observer. Hitchins [2000] focuses on configuration entropy, which "is the degree of disorder in pattern, organization and structure" and is represented by the formula $S = k\ ln\ W$, where S is entropy, k is Boltzmann's constant and W is the number of ways things can be arranged.

Mitchell [2009, p.168] refers to Crutchfield's "particles" that are the boundaries between the areas of simple patterns that can be identified in a space-time diagram. The image alters as the particles interact with the borders of the image. "This can be seen as information-processing at the particles". The idea of identifying "information" flow and entropy within a CA model has produced a number of formulae based on Shannons's information entropy, which uses certain probabilities. If an event is unlikely to occur, thus having a low probability, then a high amount of information is gained if it does occur; whereas if there is a high probability that an event will occur, then there is reactively little information gained when it does occur. If P is the probability of an event occurring, then the amount of information (I) gained is:

$$I = log_2(1/P) \tag{2.1}$$

If the focus is the amount of information gained from a series of events, then the total amount of information is the weighted sum of all the probabilities of each event occurring, giving:

$$I = \sum_{i=1}^{N} P_i log_2(1/P_i) \tag{2.2}$$

Van Dyke Parunak and Brueckner [2001] look at a formula based on Shannon's information entropy requiring the measurement of (1) the set of states accessible to the system and, (2) the probability of finding the system in each of those states,

$$S = -\sum_i p_i log p_i \tag{2.3}$$

where i is the range of possible states and p_i is the probability of finding the system in state i. Pan [2010] researched a large collection of models found within the GoL and introduced three parameters to assist in the measuring of the transition between order and chaos,

1. order parameter
2. complexity index $\phi(t)$, based on sum of the different local environments encountered in a generation
3. entropy also based on Shannon's information entropy,

$$\mathcal{H} = -\sum_i p_i log_2 p_i \tag{2.4}$$

As can be seen, the difference between the Equation 2.3 and Equation 2.4 is that the latter uses log base 2.

Wuensche looked at the idea of measuring the input-entropy of a one dimensional CA by monitoring the frequency of the look-ups in the rule table and displaying them for each time-step.

> "Each rule produces a characteristic cloud of points which lie within a parabolic envelope because high entropy is most probable at medium density, low entropy at either low or high density. Each complex rule produces a plot with its own distinctive signature, with high input-entropy variance. Chaotic rules, on the other hand, will give a flat, compact cloud at high entropy (at the top of the parabola). For ordered rules the entropy rapidly falls off with very few data points because the system moves rapidly to an attractor" [Wuensche, 1998].

The identification of patterns and the repeat cycle is a relatively standard exercise, but Crutchfield does not see it as a definite proof of emergence. Consideration of the role of the observer and the biases they might bring is important. This also applies in distinguishing the difference between discovery and emergence. He holds that emergence is dynamical and usually evolves over time and appears as something new; whereas discovery "is atemporal: the change in state and increased knowledge of the observer are not the focus of the analysis activity; the products of model fitting and statistical parameter estimation are" [Crutchfield, 1994, p.9]. The plotting of the various elements of the system, such as the configuration of the agents and the rules, into a fitness landscape reveals a lot about the state of the system and the possible affect of the system elements on the development of the system. A system adapts to maintain a high level of fitness and a non-smooth fitness landscape indicates such system characteristics as feedback and emergence [Ruhl, 2006b].

Block entropy is a way to compute the presence of hidden structures in a string of symbols. Sloot et al. [1999] outline the need for some quantity to measure in order to identify phase transition within a system. They take Shannon entropy and associate it with the degree of uncertainty that exists in a complex system, defining the probability of sequences occurring in the model and formulating a spatial block entropy and a temporal entropy formula [Sloot et al., 1999, pp.209-210]. They point out that the Kolmogorov-Sinai entropy per unit time can also be used to evaluate the random space-time processes that are seen in deterministic CA [*ibid*]. Theoretically, if key words were identified as indicating a system trying to establish order, (pre-emergent state), through its communication channels, then the frequency could be used as an indicator of the level of order (emergence) or randomness (complexity). In a model this could be expressed as the amount and, or size of messages between two linked models or the amount of change between cycles or over a period of cycles. A decrease or absence of any messages or changes would indicate that the system has reached some form of ordered state, albeit maybe a deadlock or frozen one.

2.7.3 Kolmogorov Complexity

Li and Vitányi [2008] state that the "notion of Kolmogorov complexity has its roots in probability theory, information theory and philosophical notions of randomness, and came to fruition using the recent development of the theory of algorithms" [ibid, p.47]. The most common definition of Kolmogorov complexity concerns compressibility and the shortest programme to describe a string. The premise is that

any object could be describable by a binary string [Bortolussi, 2011; Spracklin and Saxton, 2007]. Like the Hurst exponent, Kolmogorov complexity is an estimate - it is impossible to compute all the possible programmes to describe a string and, therefore, to know if they would all halt [Bortolussi, 2011; Li and Vitányi, 2008; Nannen, 2010]. "It is nevertheless essential for proving existence and bounds for weaker notions of complexity" [Nannen, 2010].

Kolmogorov complexity has developed in other areas, such as with Algorithmic prefix complexity, random finite sequences and information distance. GE's research into Active Networks [Bush, 2002] looks at using Kolmogorov Complexity in the network management prediction system. Spracklin and Saxton [2007] evaluated a binary string representation of words in an email to apply a spam filter. Landauer held that there is an amount of heat that can be associated with the processing of every bit of information in a computer (see Li and Vitányi [2008, pp.629-631]); in this way the heat generated by each node could be assessed for the amount, even complexity of the information being processed. If the relevant information is represented in a string format, then Kolmorogov Complexity and associated theorems can be applied to it. But a key component is the context and as such this research intends to follow the principle that:

> "If x is an element of a 'simple' (in the sense of Kolmogorov complexity) finite set A, then the Kolmogorov complexity K(x) of x cannot be much greater than the binary logarithm log$|A|$ of the size of A. This simple upper bound on K(x) allowed Kolmogorov to define the notion of randomness; x is random in A if K(x) is close to its upper bound log$|A|$. Thus randomness essentially means the closeness of the K(x) to some upper bound."
>
> [Gammerman and Vovk, 1999]

K(x) refers to the prefix Kolmogorov complexity where x is self-delimiting. It is important to establish both the lower and upper bounds being used within the context of what is being modelled and measured.

2.8 Summary

This chapter reviewed areas that were considered pertinent to the thesis. This started with the investigation of systems and the distinction between different classes of systems. It was suggested that a linear approach to classifying systems based on the number of components would overlook the true relevance of the interactions between the components of a system. A simple or mechanistic system is either designed for a purpose, or the purpose is discernible. In this situation, the purpose becomes the single context of the system and any system behaviour is traceable and attributed to the design and workings of the system; even if it is not a direct attribute of any system component, but a result of their designed interaction. The production of mechanical energy was given as a simple example of this (*see subsection 2.2.1*). The term *complicated system* is not a separate classification, but a grading within the category of simple systems. The number of components within a complicated system make it hard to understand and unravel, but it is made up of components and interactions that are traceable and capable of being modelled. As such, simple systems can be reduced to their component parts and predictive models used to help design, monitor and even exert control over them. R-theory proposes two separate categories of systems,*complex systems* and *simple systems* or *mechanisms*, with the former as the common state of systems. But whether a linear view is taken where a system can be added to or subtracted from in order to alter its classification, or a R-theory approach where a complex system has at least one non-computable component, there is an element of judgement that has to be applied. A classification under the linear method relies on the level of knowledge available; this is especially relevant when any emergence is experienced though the interactions within a system or between a system and its immediate environment or context. Classifying complexity through terms of it being currently unexplainable or unpredictable cannot be fixed. Likewise, the halting problem means that non-computability is hard, if not impossible, to establish. A working definition of a complex system was proposed in subsection 2.2.4: "a complex system is a collection of many elements whose random interaction leads to the unpredictable emergence of robust organisation".

Modelling is used in a wide variety of forms to help design, test and monitor systems, as well as an aid to tease out the workings and future behaviour of a system. While experiments have more epistemic value than simulations, there is epistemic worth in simulations, especially as opaque thought experiments and in situations where experiments are not a realistic option. The many interacting components in a complex system means that, by its very nature, it can be difficult

to capture the system in its totality. Although it relies on selecting appropriate attributes and rules, a limited model of a complex system is perfectly acceptable, as it is an abstraction of the system and is intended as a representation of the system, not a complete replication. The simulation of a model of a complex system cannot show the true potentials of the contexts open to the system, but has to collapse it down to a single contextual view, as is the case with a simple system. In the case of a complex system this means taking a localised and temporal contextual view of the system. Reductionism is invaluable in the process of modelling and simulation, but it is limited as a tool for explaining complexity and emergence. For its part, computer simulation provides a platform for modelling complexity, where the interaction of elements using local rules can demonstrate the emergence of, for example, patterns and self-organisation.

While the modelling and simulation of highly complicated or complex systems is limited or at the very least localised, it still has value. Not least as a means for testing out a hypothesis; a simulation's output can be compared with other observations. And while such models and simulation cannot be taken as direct representation of a system, they can be used to monitor or reflect situations where a system changes state, or to give insight into situations that appear to lead to such changes. A corporate LAN can be seen as an extremely complicated, (some would say complex), network of interactive components, handling massive amounts of data and open to unwanted and disabling perturbations. Various metrics are linked to a combination of models to monitor the state of the network and to alert to any 'known' problems. Such models and metrics then become generalised across different network systems, but they are not of any real direct use in completely different types of systems.

Mental models are used to transfer acquired information and metrics across different systems' scenarios and contexts. The level of abstraction affects the diversity of system for which the mental model is of use; the higher the level of abstraction, the more widespread the range of systems the model can be applied to. But as the level of abstraction becomes higher, the model becomes less specific. The same identification of commonality that makes mental models so effective, can be applied to a formal model of a system. In this way analogous systems, or subsystems can be identified through shared models.

The search for any common indicators across different systems would be facilitated through the analysis of any shared models that could be found. Following the example of mental models, the level of abstraction would determine how diverse the

systems would be. But this is very dependent on the modeller and the abstraction and modelling technique chosen. A simpler means of attempting to identify any common indicators of system state across different systems is to select a commonly used modelling technique, such as cellular automata. Cellular automata have been used to model and simulate a variety of domains, as shown in the categorisation of 2D CA in Table 2.2. The range is evident within as well as between the categories. While what is being modelled and the rules being used are different, all the 2D CA share a common 2D grid output space. The modeller usually designs the model to look for specific events, but there are details about the general state of the output space that are ignored and could, instead, be used to track the overall state of the model's representation of the system. The next chapter will look at methods of measuring the general state of the output of two dimensional cellular automata.

Chapter 3

Methododology

3.1 Introduction

When investigating non simple systems there is a certain inevitability that any approach will involve abstraction. As was discussed in the previous chapter, reductionism traditionally seeks to reduce a system to its component parts in order to understand how each component works and with what and how it interacts to form the system under examination. This can be seen as a process of vertical reduction. As was mooted in the previous chapter, abstraction can be viewed as a form of reduction where the identification of analogies between different systems can lead to common models. This process, both the analogies and common models, can be conceptualised as a form of horizontal reductionism in which the abstraction of key features of a system is akin to reducing the scope of what is examined of a system in order to better unravel and understand some system phenomenon. As well as being used in an analogous simulation, this process of abstraction and modelling can be employed in computer simulation to show how the observed behaviour of one system can be abstracted and modelled to simulate the behaviour of a diverse system.

This abstraction or horizontal reduction is more often than not used to form a model of a non simple system. A model of a complicated or complex system will, therefore, represent a designated part of the whole system, which can broadly be seen as a reductionist approach. Even if an open and holistic view is taken where the interactions and relationships within the system and between the system and its environment are the focus, rather than just the physical parts and structure of the system, the modelling of the system, especially any computational modelling, cannot directly reflect the complexity of the interactions. Indeed, R-theory contends that the main indication of a complex system is that at least one of its parts

is non-computational. So any computer modelling and simulation of a complex system has to be seen also in the context of a localised and temporal abstraction of the system. These localised and temporal abstraction can be combined to give a wider view of a complex system, such as with weather forecasting. Over the decades improved methods have been developed for collecting and collating more precise and timely information for input into weather models, such as the use of satellites to gather weather information from above oceans. But even with an immense increase in the amount of computer resources used and in the quantity, quality and global spread of information gathered, the accuracy of any forecast is lessened by any increase in the period between the forecast and the time for which it was made. As was stated in the previous chapter, a computer model is only a representation of a real system.

This can mean that at times the connection between a computer simulation and the system in focus can be tenuous, albeit that the system is itself a model abstracted from the real world. Consequently, although the search for isomorphic and analogous systems through shared mental models or common models can be productive, it is, potentially, very much dependent on individual observation and selection and thus very limited in scope. In order to obtain any common indicators of a system's state across different domains, a series of factors have to be agreed on, such as (a) the commonality of the abstractions selected, (b) the modelling framework used, (c) the commonality of the model, (d) the metrics used and (e) the process of analysis of the output. An alternative approach is to investigate a common modelling and simulation technique that is used across a wide and diverse range of systems. The question can then be moved to whether the commonality of the output space can be used to tell us anything relevant across the range of systems being modelled; even though each model is set up and its output space analysed for specific features, rather than for any generic statements about the output space.

The rest of the chapter starts in section 3.2 with consideration of simulation methodology. This picks up the discussion on the epistemic worth of computer simulations in the previous chapter, (*see subsection 2.5.1*), and looks at the verification and validation process in computer simulation. In section 3.3 the research approach of analysing the output space of diverse 2D CA simulations is outlined; including some of the ways of traditionally measuring 2D CA grids, such as mean density (subsection 3.3.2), randomness and clusters (subsection 3.3.3), entropy (subsection 3.3.4) and a new approach to measuring the connectedness of the active elements on a 2D grid (subsection 3.3.5).

In section 3.4 the research approach adopted to build, run and analyse the simulations used in the thesis is summarised. The use of scenarios to verify the analysis programs written in Perl is explained in subsection 3.4.1. The three types of CA to be simulated are outlined in section 3.5 and their origins and backgrounds, against which they are externally validated, are introduced in subsection 3.4.2. The metrics selected to measure the output of the simulations are described in section 3.6, which also provides a formal definition of the CA output space. Three existing metrics are outlined, mean density (subsection 3.6.1), bounding box rate (subsection 3.6.2) and an entropic metric (subsection 3.6.3). The new proposed measure of the connectedness between individual elements on the 2D grid is expounded in subsection 3.6.4. The chapter closes with a summary in section 3.7.

3.2 Simulation methodology

The pervasiveness of modelling was mentioned in the previous chapter. Human perception of and interaction with the world can be viewed as a process of modelling. In this way the actual interaction using those models could be seen as a process of simulation, where the models are 'run'. Thought experiments can be seen as the process of forming mental models and hypotheses, and then mentally running them. The intention here is not to suggest that the world and life is digital, but rather that often the context that we place things in for evaluation is. The scientific approach is to break things down into units that we can understand, preferably to the lowest level of a two state option, such as with a straightforward choice between either|or, on|off, or 0|1. Such topics as fuzzy logic and quantum theory have shown that it is not always as clear cut as a choice between two options; there are shades of grey and the process of observation can influence and change the perception of things. But even with such uncertainties and vagueness, as well as complex systems that cannot be fully explained, models are formed and simulations run and some epistemic value gained, as discussed in section 2.5.

Although simulation should not be seen as synonymous with computer simulation, the increase in the use and power of computers means that computer simulation can be viewed as the main method of simulation within research and industry. In this sense computer simulation is better approached from an application and practical viewpoint, rather than purely from unbounded philosophical speculation. Ulgen et al. [1994] provide a practitioner's perspective to using simulation methodology in industry. In practical terms they see the simulation methodology as the "the process of applying the simulation technique" [*ibid*]. As part of the process they highlight the need for conceptual validation between the real system

and the conceptual model; verification between the conceptual model and the simulation model; and operational validation between the simulation model and the real system [Ulgen et al., 1994, p.11]. This is shown in the section of Figure 3.1 above the dotted line.

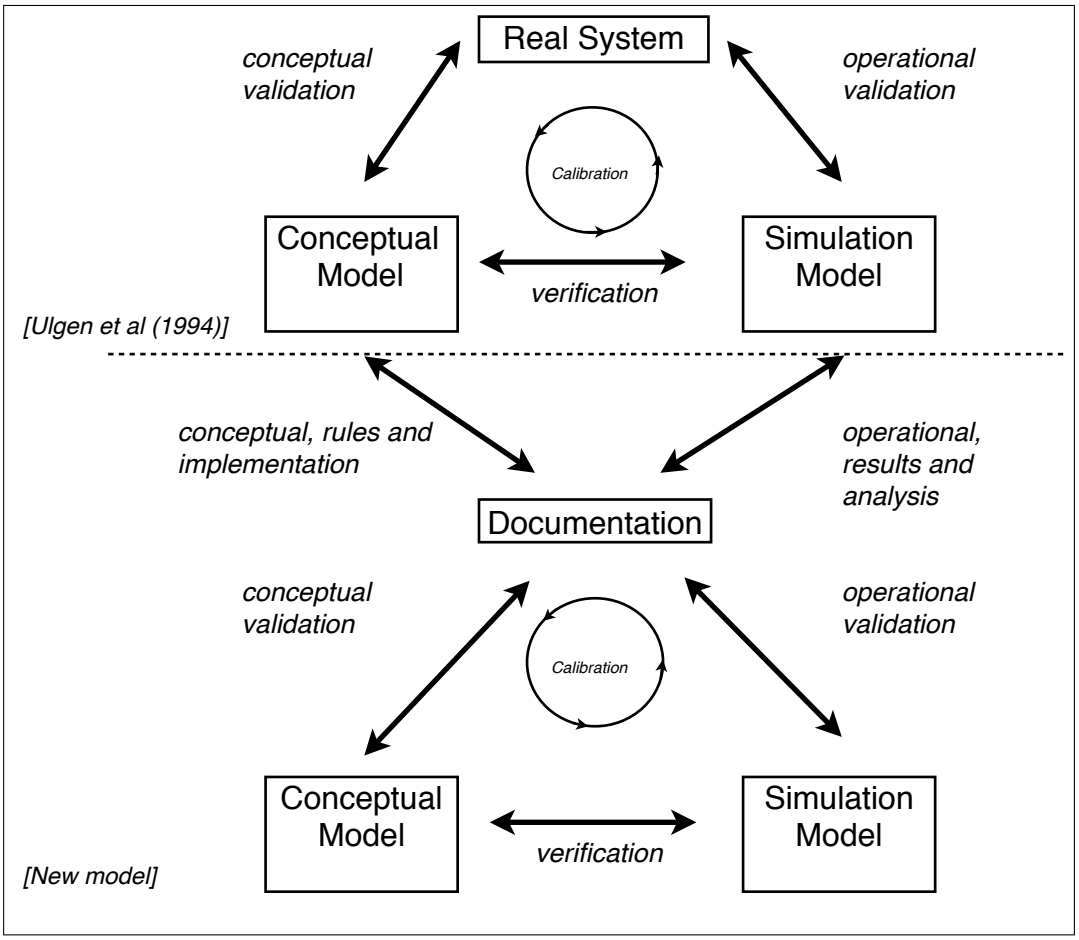


Figure 3.1: The top diagram above the dotted line is from [Ulgen et al., 1994]. The bottom part of the diagram represents how the current work is validated against the documentation of the conceptual and simulation models of the previous studies. This includes the rules, implementation, results and analysis, as part of the conceptual and operational information. Thus it can be said that the new models are being validated against the 'real system' of the original work by proxy.

This thesis replicates three previous studies, (*see section 3.5*). Any independent reproduction of the original work that successfully corroborates the findings adds credence to the validity of that original work. But rather than validating the conceptual and simulation models against a real world target system, they are validated against the documentation outlining the relevant original work and its results. Consequently, the new simulation and conceptual models have to be verified against each other, but the conceptual and simulation models are validated

against their documented counterparts in the original simulations. In this way the new work, if it successfully replicates the original work, can be seen as both validating the original work and being validated against a 'real system' by proxy through the original work, in so far as the original work was successfully validated (*see Figure 3.1*).

The new simulations then provides a variety and range of 2D outputs from three different types of CA. These outputs map the progression of a cross selection of the states of the simulations through time and, preferably, noticeable state change. The outputs are measured to assess whether each metric is of use to specific types of CA, or whether any of the metrics generate worthwhile general information from all three of the CA types simulated.

3.3 Research approach

The discussion of computer simulation, (section 2.5), and CA, (section 2.6), in the previous chapter illustrated the usefulness and wide usage of 2D CA models to simulate system's phenomenon. Although the real world is 3D, 2D CA models were considered a better choice than either 1D or 3D ones because:

- they gave a better visual representation of natural phenomenon than 1D CA; such as patterns and traffic / pedestrian movement,
- they provided a simpler output space to analyse than 3D CA,
- they were easier to program than 3D CA,
- they required less computer resources than 3D CA; allowing them to be more easily run on laptops and desktop computers, and
- there is a much larger body of work using 2D, rather than 3D.

In subsection 2.6.5 three categories of CA proposed in [Ermentrout and Edelstein-Keshet, 1993] were expanded to four by defining randomised automata as CA that were similar to deterministic automata except for the inclusion of probability in their updating process. This provides a useful means of selecting diverse systems that use a common output space, but have particular features they are interesting in measuring through the use of specific metrics or types of metrics. This facilitates the research approach of analysing the output space of simulations conducted using 2D CA. For the purpose of the current discussion a two state output is considered, which can be termed inactive and active. The means of generating different outputs and how it influences the measuring technique will be looked

into below; the first consideration is the potential range of different output space, regardless of any intended or hoped for state. The two extremes are an output space that is either populated with all inactive cells or all active cells.

The rest of this section continues on from the discussion in subsection 2.7.1 and considers some of the ways measuring CA 2D grid space can be approached.

3.3.1 Measuring 2D grids

This work follows on from previous research that calculated the Hurst exponent of the resource usages of each node on a 2D grid output from a Petri-net simulation of an Active Network model [de Silva, 2004]. In this thesis a different approach to measurement is taken from that in the previous study for two reasons. Firstly, the Hurst exponent generally requires a sizeable range of data to work with and the algorithmic process can be laborious, although Hagiwara et al. [2000] have looked at a scaled down method for use in real network traffic analysis. Consequently, a key consideration is that the rescaled range and subsequently the Hurst exponent does not work well with data that is non-changing, such as a node that has no call on its resources over a period of time. Steps can be taken to try and limit this kind of impact, such as by ignoring negative rescaled range values or defaulting them to a minimum positive value; but this can reduce the value of the resulting Hurst value, which is in itself an estimate. The second issue is that it does not show anything about the state of the grid and, therefore, how it has changed and might potentially change. This is not to say the cell or node view is not of use, but rather that the overall grid view is more relevant in this thesis.

The context in which the output space is viewed is tied into the context of what is being modelled and, in turn, helps decide how it is measured. Three views represent the way a 2D CA output space is considered; an agent's as it traverses across the grid, a cell's as its state fluctuates, and the grid's overall state after each time step. A specific agent view or cell view can be broken down to show the activity of individual agents over the grid, or the fluctuating state of a cell. A grid view obviously puts the behaviour of the agent or state of the cell into context both with other agents or cells and with that of the dimensions of the grid itself. Although the grid view presents the best overall perspective, analysis of any of these will provide information or insight into the overall changing state of the grid; such as any congestion or blockages on the grid (grid view) through the movement of packages across the grid (agent view) or the consumption of resources (cell view). Obviously, the choice is tempered by the attributes of the system that

are being modelled and its context. In more detail:

- *an agent view* includes examining the shortest path, obstacle avoidance and self-organisation; in the case of modelling a network, then any potential congestion or blockage can be shown by an agent not moving or taking a round about route;
- *a cell view* can be seen as a different take of an agent view, such as when the tracking of resource usage across a grid reflects the path taken by an agent; the tracking of a network resource of a network router in a cell, such as processing power or memory, can suggest possible congestion, blockages, slow response, and under or over utilisation; and
- *a grid view* provides the bigger picture and reveal patterns across the whole grid or the randomness of any grid activity; giving a more comprehensive perspective on the nature of the functioning of the modelled network.

Whatever measure and method of measurement is adopted one of the first steps is to establish how the key states are registered for each domain. A pattern may indicate a better grid utilisation than random usage, or vice versa. Likewise a full grid could indicate gridlock, or a highly utilised network, or a virus dominated grid; and a nearly empty grid could reflect the opposite. In this way the choice of what to measure reflects the objectives of the simulation process and the context of the modelled system, as well as the often subjective 'interests' of the researcher [Iordache, 2011]. It also influences the choice of metric employed, which is specific to the simulation. The issue then becomes whether a single metric or a combination of metrics can tell us anything general about the state of different types of CA simulations.

If a 2D grid output from a CA simulation using two states to represent either an active / occupied or an inactive / unoccupied cell is considered, then there a number of obvious approaches to how to measure the state of the output space, albeit that they do not work as well with all CA types.

3.3.2 Mean density

If the mean, or average density is given as the number of active cells divided by the number of cells in the output space, then the mean density of the space will range from 0 for one with no active cells, to 1 where all the cells are active. This would indicate that there is a relatively easy measure of the output space, but only if density is the main characteristic being measured. In the case of a

deterministic CA this would be of interest; such as tracking the contagiousness of a virus or, as in percolation theory, ascertaining at what percentage of active cells a cluster spanning between two opposite sides of the grid occurs. But mean density would appear to offer little of value in an agent CA, where the focus is likely to be more on the placement of the agents in the output space in terms of clusters or self-organisation. This would suggest that the value of such an indicator would be limited to domains within the same CA type, where the purpose of the measure would be the same or similar. It may be, in order to attempt to gain any commonality across CA types, that mean density can be used as an indication of the overall state and, thus, what subsequent measure to employ.

3.3.3 Randomness and clusters

A common object of interest in the analysis of a 2D CA output is how random any active cells are. This can be expressed as the randomness or lack of randomness in any data representation of the output space, or in the clustering of the the active cells. If an algorithm related to Kolmogorov complexity (*see subsection 2.7.3*) is used, then the former depends a lot on how the data are unpacked from the 2D grid; the extraction of the cells into a single 1D array will only relate to a small part of the potential connection between cells. A cellular automaton being assessed within a Moore neighbourhood of all its 8 surrounding cells and, usually, itself would have eight possible connections, as opposed to the 2 that a 1D representation would provide. Chen and Sundaram [2005] use it to estimate the complexity of 2D shapes, but this again relies on extracting the outer shape into a 1D linked chain of points.

Clusters or the lack of clusters are a better way to signify the randomness of the active cells within a 2D grid. The movement of singleton cells, (i.e. single cells not in a cluster), into a cluster can also be used to highlight self-organisation (SO), especially within an agent based CA. The density of the 2D grid, as well as the number of different cluster sizes has been suggested as a measurement of complexity, as well as an indication of when a percolation or spanning cluster has been formed [Tsang and Tsang, 1999]. As the density increases, the diversity of cluster sizes decreases and eventually a cluster that spans two opposite sides of the grid is formed. In percolation theory the focus is on looking for the percolation threshold or critical percolation point, p_c, given at the level in a grid's density where a spanning cluster occurs. This can also be given by the probability value required to produce a grid with a spanning cluster, where the value is the probability that a cell is active. But just as at one end of the density scale a grid with a

high percentage of active cells will congeal into a single cluster, at the other end a grid with a handful of agents can likewise self organise into a single cluster. The former generally reflects a movement from a range of clusters of different sizes to a single cluster, while the latter from 'random' singletons to a single cluster. The end result is the same, but the evaluation of the preceding time steps is different. This can present a problem with using the same metric to track the progression to a single cluster for both a sparsely and a densely populated grid.

3.3.4 Entropy

The use of entropy to measure CA was reviewed in subsection 2.7.2. In many ways it has become a term that is used to indicate a metric employing some form of probability relationship between the current state of a system and its next measured state. This originates in the principles brought in by Boltzmann and statistical mechanics, which deal with the number of possible microsystems of a system. In this way entropy becomes the measure of how many potential microsystem exist to explain the next macrosystem state of the grid. This has led some people to form a the link between order/disorder and the level of entropy in a system, where high entropy is associated with a high degree of randomness. Thus a system incorporating the change from a solid to a liquid and then to a gas is seen as increasing its entropy. In terms of the microsystems, the possible positions, velocity and direction of each molecule are much more varied for a gas than a liquid, and for a liquid than a solid.

The classical view of entropy is engrossed with an isolated or closed views of systems and their macrosystems. Here the entropy is associated with the amount of energy **not** available for work. Traditionally it is measured in terms of temperature and as the system increases in entropy it will eventually reach a point of equilibrium where the system has a uniformed temperature. If a temperature map is taken of the system then as maximum entropy and equilibrium is reached, the macrosystem temperature map reaches a stable condition. This lends some confusion as maximum entropy occurs as the system settles down into a stable state, at least in terms of the temperature that is being measured. However, this needs to be viewed also in terms of the microsystems. Statistical entropy focuses on the number of potential microsystems that could could represent the state of the macrosystem at the next time step. As a system increases in entropy, so does the number of microsystems that could describe the next macrosystem state. It is this that creates the idea of an increase in entropy being associated with a rise in disorder and randomness. As the isolated macrosystem reaches equilibrium it also

reaches maximum entropy, even though the macrosystem has the appearance of a stable, ordered and non-random state. But this is only in terms of the temperature being used to measure the macrosystem. If the microsystems show, for example, the potential position and velocity of the atoms that will make up the macrosystem at the next time step, then although the macrosystems is now at a constant temperature, the number of potential microsystems will not have decreased and may have increased. This aligns with the assertion that entropy will remain the same or increase within an isolated system.

This approach also holds with Shannon's information entropy, which links the amount of information about an event to the probability that it will occur. The more likely it is to occur, the less information we will gain if it does. Information in this context is best seen as the level or measure of uncertainty at the receiving end of a communication, transmission or, in this case, the completion of a CA time step. In a grid with two possible states, the maximum uncertainty is when both states have equal probability. The setting of the state under this probability setting would give us the most information. As the probability increases for one state and decreases for the other, the uncertainty over which state will prevail becomes less and the subsequent information value equally diminishes. This can be applied to just the state of the macrosystem.

In viewing the output space of a 2D CA the initial consideration is the macrolevel one and the mapping of the change between the quantity and location of active cells within the context of the bounded parameters of the CA grid. Such an entropic evaluation of the macrolevel 2D CA grid looks at the probability or frequency that an active cell is within some other sub-structure within the grid. This substructure can be a structural subdivision of the grid space (see Guisado et al. [2005]), or the result of some other analysis of the grid, such as cluster analysis (see Juwono [2012], Tsang and Tsang [1999], Essam [1980]). But there could be a problem as the entropy value moves to its maximum or minimum as it can provide little information on any more subtle changes within the grid and its macrolevel view.

3.3.5 Connectedness

The concept of the connectivity, connectedness, compactness and similar notions have been used in a number of ways. Connectedness and connectivity between nodes or data items is related to how the members of a cluster are found. Dunne et al. [2002] consider the role of connectance and size in food-web structures, "which depict networks of trophic relationships in ecosystems, provide complex yet

tractable depictions of biodiversity, species interactions, and ecosystem structure and function" [*ibid*, p.12917]. Trophic species are the taxonomic categories of species "that share the same set of predators and prey within a food web" [*ibid*, p.12918]. They define connectance as equal to L/S^2, where L is the directed trophic links between S nodes or trophic species. This is related to clustering in random networks:

> "The linear relationship of the clustering ratio to network size arises because clustering in random networks should be equal to connectance ($C = L/S^2$), because the likelihood that two neighbors of a node are connected."
>
> [*ibid*, p.12919]

Ding and He [2004] take the nearest neighbour consistency that is a key concept in statistical pattern recognition and extend it to "data clustering, requiring that for any data point in a cluster, its k-nearest neighbors [kNN] and mutual nearest neighbors [kMN] should also be in the same cluster". This kNN relationship is not a symmetrical one. If y is the nearest neighbour to x it does not mean that x is the nearest neighbour to y, but it is sometimes convenient to use kMN to define a symmetric nearest neighbours relationship. They define Cluster kNN consistency as "[f]or any data object in a cluster, its k-nearest neighbors should also be in the same cluster" [Ding and He, 2004, p.1].

Handl and Knowles [2007] look at the evaluation of multiobjective clustering solutions, including an evolutionary approach to the problem. They separate clustering algorithms into three major groups, of which one is:

> "Methods based on a concept of *connectedness* make up the second group. They employ a more local concept of clustering based on the idea that neighbouring data items should share the same cluster. [....] These are well-suited to detect clusters of arbitrary shapes, however, they can lack robustness when there is little spatial separation between clusters."
>
> [*ibid*, p.58. author's italics]

They assess the cluster connectedness with a measure of the connectivity that is "conceptually similar to the criterion of nearest-neighbor consistency" introduced by Ding and He [Handl and Knowles, 2007, p.60]. The more compact a cluster is the less its connectivity values are.

On one level connectedness is related to how a cluster is formed based on the connectivity between data items or elements. On another level connectedness can also be used to express the connectivity between clusters of data items within the dataset. The sorting of data items into clusters involves establishing the connectivity level or rule that is used to decide whether a particular date item is included in a specific cluster. The degree of that connectivity can also be used to then set the connectedness or compactness of the resulting cluster, or if the clusters are seen as data objects, then the connectedness between the data objects within the data set. So within a cluster of data items it can act as an indicator of how closely related the items are and within a dataset it can point to the nearest neighbour or related dataset.

The above studies show that the concept of connectedness pre-exists as an indicator of the degree of connectivity or compactness used or achieved in forming a cluster. However, this work takes its inspiration from graph theory, a branch of topology. Three types of connectedness in topology are considered in [Mendelson, 1990, pp.112-118]; (a) connectedness when it is impossible to decompose the subspace of a topological space "into two disjoint non-empty open sets", meaning it is connected, (b) path connectedness, which involves the connection of two points, and (c) "simple connectedness" where the topological space is simply connected "if there are no holes in it to prevent the continuous shrinking of each close arc to a point" [*ibid*, p.112]. The third type is reflected in the 'simply connectedness' that Cook uses in his model of pseudo still life, where he considers whether he can take a given pattern and "decompose its islands into two stable sets. This is equivalent to finding a boundary that separates the two sets from each other" [Cook, 2003, p.97]. In order to accomplish this he sees the cells as representing either islands that consist of 'land' cells, or empty cells that can be viewed as 'water'. The boundary has to go through the water to separate "the islands in a stable way" [*ibid*]. Water cells that are connected to each other constitutes a sea. He uses his concept of 'simply connectedness' to determine if a cell should be added to a sea or not; this is based on the state of its neighbouring cells. Thus his idea of connectedness is related to a rule determining the state of a cell.

> "If any sea is not simply connected (that is there is land both inside and outside it), then we can draw a boundary dividing land through that sea alone, and so the pattern is a pseudo still life."
>
> [Cook, 2003, p.99]

Graph theoretical models tend to use a two dimensional representation, using vertices (sometimes referred to as points or nodes) to signify actors within the network and edges (sometimes referred to as lines) showing direct ties between a pair of vertices. A pair of vertices is also referred to as a dyad. A path on a graph is defined as a set of specific vertices and edges that connect a defined dyad. There may be more than one path between the defined dyad. The length of a path is the number of edges in it. The graph itself can be standard, where the edge between a dyad is symmetrical, or it can be a directed graph, digraph, which shows whether the link between the two vertices is one way or bidirectional. The graph can also be assigned values signifying, for instance, the strength of the relationship between the dyads, making it a valued graph, or *vgraph*.

Graph theoretical models have often been linked to describe social networks.

> "For many centuries ideas now embodied in graph theory have been implicit in lay discussions of networks. The explicit linking of graph theory and network analysis began only in 1953 and has been rediscovered many times since. Analysts have taken from graph theory mainly concepts and terminology; its theorems, though potentially valuable for the analysis of real data, are generally neglected."
>
> [Barnes and Harary, 1983, p.235]

In an earlier work Barnes shows how these "concepts and terminology" included connectivity and connectedness, the latter being "one of the central notions of topology" [Barnes, 1969, p.218]. In graph theory connectivity and connectedness are associated to ideas of joining dyads and the reachability of one vertex from another:

> "Applied to social networks, the word 'connectedness' and 'connectivity' may refer to properties of the distance between persons, the number of paths between them, whether there is a path at all, or the proportion of possible paths actually in existence."
>
> [Barnes, 1969, p.215]

Barnes [1969] goes on to discuss how connectivity and connectedness had been used to describe (a) the join or lack of a join, or possibility of a join between a dyad, (b) the shortest path between a dyad, (c) the number of genuinely different paths between a dyad, and (d) the orientation and direction of existing paths. Barnes also looks at the strength that can be associated with cliques within a social

network, proposing the calculation of the local density of the vertices attributed to each clique. The clique can be seen as a subgraph of the graph representing the whole network. The method he proposes "for determining local density is based on a series of undirected subgraphs formed of any given point arbitrarily chosen as root" [Barnes, 1969, p.225]. He calculates the density of the clique by expressing the number of lines in the subgraph, including those making up a path, as a percentage of the number of lines in a completed graph with the same number of points. Although he bemoaned, in a later work [Barnes and Harary, 1983], the general failure to apply the theories of graph theory to the analysis of social networks, he suggests that the calculation of local density is:

> "[A]ppropriate to the study of rumours, the application of diffuse sanctions, the establishment and maintenance of standards of evaluation, and the like in communities where the social network stretches indefinitely in all directions from any given individual."
>
> [Barnes, 1969, p.225]

Peay argues that connectedness "is trivially indistinguishable from reachability in ordinary graphs" [Peay, 1980, p.387]. But when a digraph is used the joining has different definitions, thus the triviality is removed and the terms joining and connectedness "gain substance in the conceptual elaboration required by digraphs" [*ibid*, p.390]. He concludes that:

> "Described in the most general terms, the idea of joining and connectedness has to do with systematically characterizing pairs of points in a structure of directed relationships in terms of the paths and semipaths by which they may be linked."
>
> [*ibid*, p.403]

In their tome on graph theory and its applications, Gross and Yellen [2006] devote a chapter to connectivity. The focus is on edge-connectivity and vertex-connectivity and how a connected graph can become unconnected through the removal of vertices or edges [*ibid*, pp.217-246]. In this way the connectedness is associated with the number of vertices or edges that need to be removed before a graph becomes disconnected, and with the number of alternative paths available:

> "Determining the number of edges or vertices that must be removed to disconnect a given connected graph applies directly to analyzing the vulnerability of existing or proposed telecommunications networks,

> road systems, and other networks. Intuitively, a network's vulnerability should be closely related also to the number of alternative paths between each pair of nodes."
>
> [Gross and Yellen, 2006, *ibid*, p.217]

While this work borrows some of the concepts and terms of graph theory, it is not concerned with the direction or 'path' related calculations of connectivity; nor with the vulnerability of a network. Although developed separately, it uses a concept of connectedness that can be seen as extending the relevance of the link between the number of existing connections and the possible number of connections, which was part of the underlying idea of Barnes' local density; but applied in a different context to another environment and without any connotation of paths and the passage of information.

In graph theory one of the values used is the number of edges, or connection, that a vertex has. If the vertex is seen as a cell on a CA grid, then an edge exists on the boundary between an active cell and any active cells in its neighbourhood. In terms of establishing a cluster of active cells, then it can be based on groups of cells that are connected via an edge with a neighbouring cell in the cluster. Thus some information relating to that cluster can be obtained, such as the number of cells in the cluster, or the number of clusters on the grid, or even the number of active cells not in a cluster (singletons). But identifying clusters is a specific measure that will give an indication of whether, for example, the active cells or elements on the grid have formed, or self-organised, into a group. What would be useful is a measure that provides a better understanding of the state of the whole grid and the relationship of all the active cells on that grid to the grid's state. Therefore, if the number of edges that every active cell has with a neighbouring cell is evaluated, then the connectivity of each active cell is established. This can then be used to give you the overall connectedness of those active elements, based on the number of connections and the number of active cells. If the optimal connectedness based on the size of a cell's neighbourhood for that number of active cells is calculated, you can then calculate the ratio between the actual connectedness and the optimal connectedness. This provides the potential to handle both small and large number of cells, as well as small and large groupings of cells and also to differentiate between a grid with a random spread of cells compared to highly connected groupings. In addition, this process maintains the concept of keeping the CA framework and environment as simple as possible, whilst retaining its usefulness and power as a tool. The idea of measuring the connectedness of a 2D CA output grid is explained in subsection 3.6.4.

3.4 Research strategy

The research strategy is carried out using a suite of Perl and Java programs (*see section 4.3*). Data are passed between the programs in textual format (*see Figure 3.2*). The aim is to provides a range of 2D CA output grids for measuring

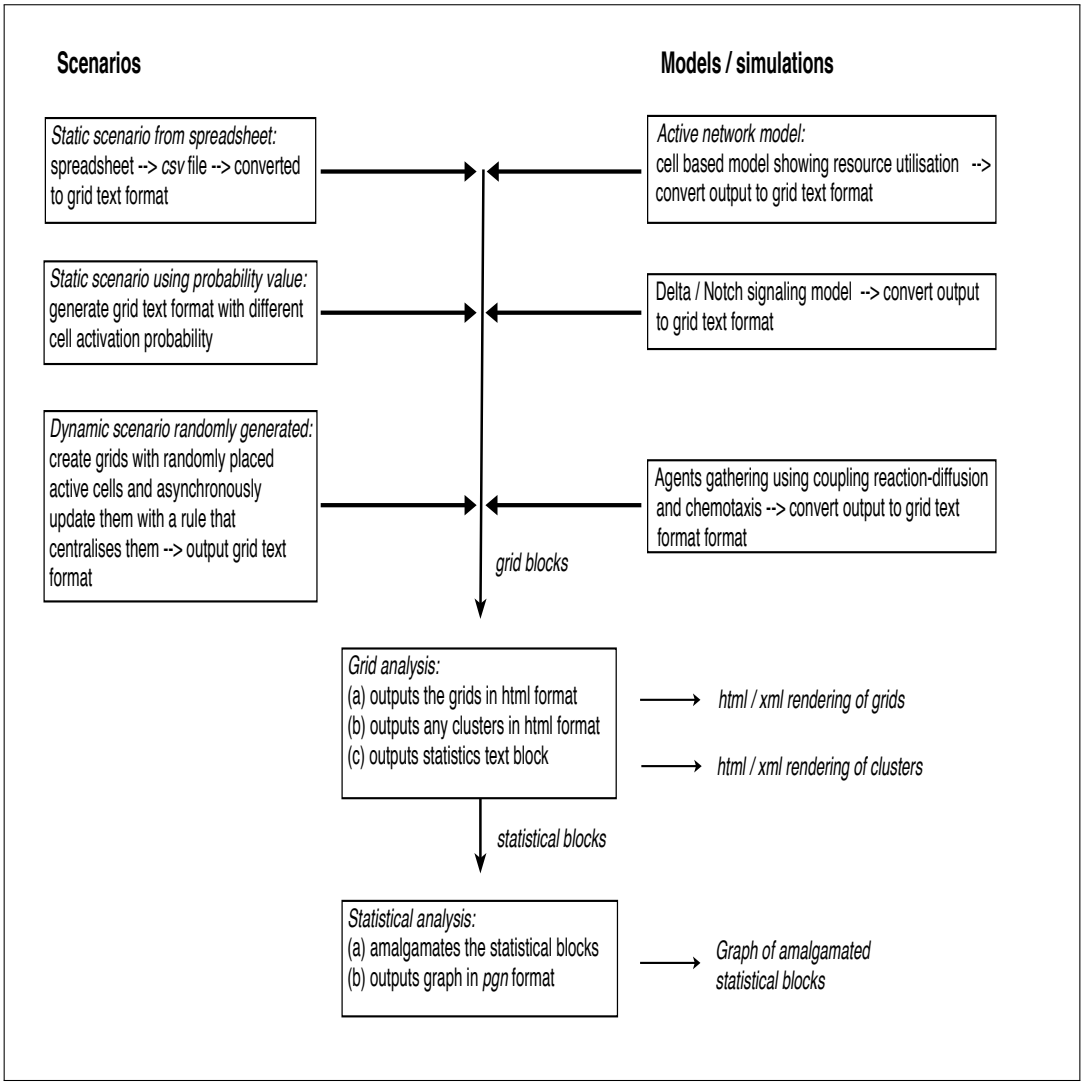


Figure 3.2: Outline of research strategy

with a series of metrics. The grids are produced in two ways, (a) the creation of scenarios 2D CA grids using Perl and (b) the modelling of three of the types of CA; one using Java within the CxA formalism and the MUSCLE code base [Hegewald, 2009; Hoekstra et al., 2010a], the others using Perl. The metrics and analysis programs are written in Perl, with data passed between the suite of programs in a text based format. The tasks are separated to both facilitate flexibility and to break down the overall processing needed into more manageable parts. The analysis can be performed either with a Moore or a von Neumann (4 cells -

north, south, east and west) neighbourhood, but with no wrap around on any of the sides. The analysis is carried out using a Manhattan distance of 1; this can be extended to 2 or 3. The Manhattan distance uses only the horizontal and vertical cells when expanding the range of a search beyond a cell's immediate neighbourhood. In cases where no clusters are found in the distance specified, it is increased until a cluster is identified, or the distance is equal to half the width or half the height of the grid, whichever is the smallest.

The second program collates and averages the statistical output and is used to create graphs of the metrics being used.

3.4.1 Scenarios

Scenarios are used to carry out the verification of the metrics and analysis programs in identifying three basic grid states: (1) connected groups or clusters of active cells, (2) the density of active cells on a grid, and (3) the gathering of active agents. This verifies that the metrics function as expected in these basic tests and also provide a baseline test for the metrics. The scenarios are split into two types: (a) a single static scenario where the output space is created with a mixture of active and inactive cells; this allows the metrics to be tested across a diverse range of possible outputs with varying numbers of active cells and (b) a dynamic scenario where an initial grid with a number of randomly located active cells is taken through a series of time steps in each of which every active cell is asynchronously updated with a simple rule that attempts to move the active cell towards the approximate centre of the grid; this allows the fluctuation state of a fixed number of cells to be observed as they move toward a central cluster.

The static scenarios are created in two ways that reflect one of the main differences in the modelling and analysis of 2D CA. The first set of static scenarios is created in a spreadsheet with a grid populated with zeros and ones. This is exported as a comma separated variable file and then converted into the text format required for input into the statistical evaluation program. This manual method allows the creation of grids with a set number of active nodes and fixed patterns or random active node distribution, which provides a test for the discovery of clusters of connected active cells.

The second set is generated by a program that uses a probability value to decide whether a cell is active. Thus a grid created with a probability value of 0.1 will have very few active cells, whereas one using 0.9 probability will mainly consist of

active cells (see Tsang and Tsang [1999] for details of this use in models testing percolation theory). The main distinctions between each of the probability values is a mean density that reflects the probability value and a decrease in the range of sizes of clusters as the increase in the probability value activates more and more cells.

The dynamic scenarios, as already mentioned above, are formed by populating a grid with a number of active cells randomly placed on the grid. The grid is then updated with a rule that attempts, within each time step, to move asynchronously each active cell towards the approximate centre of the grid. The size of the grid, number of active cells on the initial grid and the number of time steps can all be specified. If no moves are possible before the supplied time steps are completed then the process stops. This allows the metrics to be tested against a fixed number of agents on a grid and across a series of time steps, meaning that the mean density will not register any difference in the state of the grid across the time steps. It can also be used to provide a further set of single static scenarios with a fixed set of active cells that are randomly generated.

3.4.2 Model types

The simulations are based on existing work . Thus the validation of each simulation is against the results of the work it is seeking to reproduce (*see chapter 4*). Once this is completed, the test are run for the evaluation of the selected metrics (*see chapter 5*). The models simulated reflect three of the CA types outlined in the previous chapter. A growth automaton is not simulated as it reflects a simple expansion of active cells. Thus we have the following three CA model types, with their relevant system of interest:

- Particle - a model using a coupling of reaction-diffusion and chemotaxis to gather agents.
- Randomised - a protein delta-notch model of juxtacrine signalling; and,
- Deterministic - an active network model;

3.5 Models

The subject areas of the three models used as the basis for the simulations in this thesis are outlined in this section. Details of the actual design, implementation, verification and validation of the simulations against the original work is described in chapter 4. Biology provides the inspiration for the first two CA models. The first model represents the movement and gathering of 'particles' (agents) on the 2D grid. The second model involves the changing state of cells within a CA grid. It incorporates probability in the use of noise to 'control' the formation of patterns. This puts it in the new category of randomised CA type that was proposed in subsection 2.6.5 as an extension to the three proposed by Ermentrout and Edelstein-Keshet [1993]. The third model represents the deterministic change of resource usage in the nodes of a theoretical active network.

3.5.1 Particle model using raction-diffusion and chemotaxis

Slime mould *Dictyostelium discoideum* provides the basis for the particle model. *Dictyostelium* usually exists as a mono-cellular organism, but it has the ability to change into a multi-cellular organism if necessary. Its amoebae reproduce by separating into two identical sets of chromosomes, each in its own nucleus. The amoebae gather together to form a large coordinated group or *mound* when there is a scarcity of food. The gathering process is one of signalling waves of cyclic adenosine monophosphate (cAMP). The waves follow reaction-diffusion patterns that couple with the attracting phenomenon or *chemotaxis* generated by the cAMP. An amoeba builds up internal cAMP until it reaches a level where it begins to emit it. Other amoeba are attracted to the strong chemotaxis trail, much like an ant following a pheromone trail, (as in [Van Dyke Parunak and Brueckner, 2001]). As the amoeba is drawn along the trail, it dissipates its own internal cAMP until it becomes immune to the draw of the chemotaxis for a fixed period. This process gives the signal or dispersion of the chemotaxis its reaction-diffusion pattern.

The particle model follows the design devised by Fatès et al. [2008] in their coupling of chemotaxis and reaction-diffusion to provide a solution to the decentralised gathering problem. The work can be related to the controlling of robots when there is no visible link between them. The aim of the model is for scattered agents on a grid to form compact clusters. The agents have no location information regarding themselves or the other agents on the grid and are limited to either altering the state of the grid they occupy or moving to a neighbouring cell. Their perception

is restricted to their neighbourhood and their only form of communication is to send out information. Any information is passed on by the cells. In this process the information can be corrupted and its strength is dissipated at each time step; noise can also be introduced to effect the process. Fatès et al. [2008] lay out a key objective as the determination of "what are the simplest ingredients involved to achieve a decentralised gathering with these constraints".

The original work measures the gathering of the amoebae with the *bounding box ratio* (BBR), where the area of the box containing all the amoebae is divided by the total area of the grid.

> "This parameter is rather simplistic since it captures only a small part of the system's organisation into clusters. However, note that it is a 'strong' criterion in the sense that a low values of BBR is attained only if no amoeba is forgotten from the gathering."
>
> [Fatès et al., 2008, p.13]

BBR is outlined in subsection 3.6.2.

3.5.2 Randomised model using delta-notch signalling

The randomised model does not use a biological system as a means to solve an unrelated problem. Instead it is an actual representation of how the delta-notch protein signalling system can determine cell growth. Different cell types emerge from a structure that is initially composed of cells of the same type. This differentiation can be seen in all animals and plant tissue. The range of topics covered has included how butterfly wings form [Evans and Marcus, 2006; Reed and Serfas, 2004]; the role of the delta-notch in cardiac development and disease [Abdulla et al., 2012]; and various studies of the fruit fly *Drosophila melanogaster* [Crozatier et al., 2004; Luthi et al., 1998; Radtke et al., 2005], including how a cell is selected to develop into a sensory bristle [Cohen et al., 2011, 2010; Nagle, 2011]. The trigger for the differentiation is largely attributed to gene activity, which can be tracked by measuring the protein concentration in a cell.

> "Genes control cell fate by controlling the type and amount of proteins made in a cell. Proteins in turn affect gene activity by turning 'on' or 'off' gene expression thereby affecting the production of proteins themselves."
>
> [Ghosh and Tomlin, 2001, p.232]

There is general acceptance of the concept that what helps to determine a cell's fate is the lateral signalling between the cells via the delta-notch protein pathway. Delta and notch are transmembrane proteins. The delta ligand interacts with notch receptors in neighbouring cells. How the cells interact affects the production of delta ligands and the development of the cell [Chitnis, 1995; Greenwald and Rubin, 1992]. If interaction results in the activation of the notch receptor, then the gene expression of the cell is immediately affected. The activated cell also influences its own and its neighbours production of delta ligands. This produces lateral inhibition where a cell increasing its production of ligands compels its neighbours to cut back their output of ligands. The opposite can also occur, where lateral induction encourages the creation of ligands in neighbouring cells, leading to clear distinction between groups of cells. In this way delta-notch signalling is seen by many as the mechanism for skin pigmentation and tissue patterns [Chen et al., 2014; Cohen et al., 2011, 2010; Collier et al., 1996; Ghosh and Tomlin, 2001; Kicheva et al., 2012; Savill and Sherratt, 2003; Shaya and Sprinzak, 2011].

The modelling of delta-notch signalling attempts to help with insight into how this cell to cell communication and development works; an area that is still not fully understood. This model mirrors a simple, but very effective simulation of the role of structured noise and delta-notch signalling in self-organising patterns [Cohen et al., 2011]. In an earlier work Cohen et al. [2010] used a mathematical model to show that the organisation of the bristles on a fruit fly's notum did not conform to the traditional models of delta-notch signalling. They found that the forming of the pattern was dependent on introducing lateral inhibition from non-neighbouring cells. Their later work [Cohen et al., 2011] uses a CA to simulate this lateral inhibition as a form of structured noise that could marshal cells into a well ordered pattern.

The underlying principle of delta-notch signalling used in the model is that cells compete to change to a delta state and once they have done this they inhibit their neighbouring cells from making the same change by activating their notch signalling. This results in a discernible spaced pattern of delta, or active cells. This visible pattern is stable and static, but is set by the order of the asynchronous updating of the cells. The perturbation of this process with structured noise breaks up this stability, leading the cells to develop over time a much tighter patterning.

3.5.3 Deterministic model using a theoretical active network

The particle model, (*see subsection 3.5.1*), shows agents manoeuvring and gathering together across a grid, whereas the randomised model, (*see subsection 3.5.2*), is based on the changing state of cells on the grid. The deterministic model simulates an active network (AN) and is based on the Petri-net simulation of a theoretical AN in [de Silva, 2004]. In an AN the data packets can be processed at a server node, thus using more sever resources than just required for the receiving and forwarding of the packet. The CA model representation of the original model can be seen as a hybrid of the other two models, containing moving agents in the form of the Active Data Packets (ADP), moving across a grid of cells representing network servers with changing states. Although a key difference between the abstracting of moving agents in the particle model and this model is that in the latter the ADPs are moving across and off of the grid, whereas in the former amoebae are seeking to gather.

However, the focus of the analysis is not on the movement of the ADP, but rather the deterministic change to the resource usage of the network servers that is reflected in the changing state of the cells on the grid. An ADP is set with an individual resource requirement, which a server will provide if it has enough to spare; otherwise the ADP is passed on. If an ADP has been processed, then subsequent cells will forward it on until it exits the grid (network). The original work used the Hurst exponent to measure the fluctuations server usage in each cell. But as explained in subsection 3.3.1, this metric is not considered suitable for measuring the 2D CA output space.

3.6 Metrics

Four approaches to measuring the output grid of a 2D cellular automaton were considered in section 3.3; (1) mean density, (2) randomness and clusters, (3) entropy, and (4) connectedness. Four metrics are selected that reflect these approaches.

1. A standard mean density metric is selected to provide a basic measurement of the number of active cells on the output grid (*see subsection 3.6.1*).

2. The BBR measure used in [Fatès et al., 2008] is used to provide a relatively simple calculation of the cluster size of all the active cells (*see subsection 3.6.2*). Apart from being another relatively simple metric, it provides

a link to the work used as the base for the particle simulation, thus giving the means to validate the particle automaton against the original work.

3. The entropy metric chosen is used in [Tsang and Tsang, 1999]. This involves the identification of clusters within the output space, using the decrease in the variety of clusters of different sizes to indicate a move towards a single cluster spanning the grid (*see subsection 3.6.3*).

4. The last metric proposes a new measure based on the calculation of the number of connections, or edges that the active cells have, divided by the optimum connectedness that the same number of cells would have (*see subsection 3.6.4*).

The first three are existing metrics and are used to compare with and validate the fourth, as well as to see if there is any potential in combining them. The last three metrics feature the identification of clusters or cell connectivity; this allows a view of how different clustering approaches can be used. The formation of clusters can be seen as the emergence of order or a loss of randomness, but randomness is not necessarily easy to establish. A nearly empty or nearly full grid represents the two extremes of mean density; but they should also represent noticeable stages in the entropy of a grid. As a grid fills up it should also correspond with the point where a spanning cluster is created. Consequently, none of the metrics can be seen as so specific that they exclude any similarities with any of the other metrics. This cross coverage of the four approaches, as well as the relatively easiness of understanding and implementing the metrics, helped in the selection. Although there are a rich variety of metrics, especially relating to clustering and entropy that could have been considered, the number was kept at four to correlate with the four approaches outlined above.

The following provides a formal definition of the CA output space. The grid or lattice is formally defined as,

$$L = \{1, ..., I\} * \{1, ..., J\}$$

where I is the number of rows and J is the number of columns. The number of cells is given by,

$$N_L = I * J$$

A cell is located on the grid by,

$$c_{i,j}, \text{ where } i \in I \text{ and } j \in J$$

The range of possible states is given by, for example a two state system, as,

$$R = \{0, 1\}$$

The state of a cell is,

$$\sigma_{i,j}, \text{ where } \sigma \in R$$

The number of active cells on a grid is statistically given by,

$$N_{L_{\sigma=1}}$$

The neighbourhood of a cell $c_{i,j}$ is denoted by,

$$V_{c_{i,j}}$$

A cluster is labelled with its size, k_s and the number of clusters of size s in a grid is N_{k_s}

In examples of the models (*e.g., see chapter 4*) the specific cell location is often dropped in general discussions about a cell, its neighbourhood and the rules being applied to a cell.

3.6.1 Mean density

The mean density is a simple calculation of the number of active sites in a grid divided by the total number of cells in the grid.

$$MD = \frac{N_{L_{\sigma=1}}}{N_L} \tag{3.1}$$

This has already been mentioned as a way of distinguishing the growth in the number of active cells in a grid over a series of time steps. Its limitation as a potential common measure across CA types has been raised (subsection 3.3.2), but it acts as a base line as well as a basic indicator that could influence the application of other metrics.

3.6.2 Bounding box ratio

Fatès et al. [2008, pp.13-16] propose measuring the rectangle that encompasses all the active cells on the grid. The difference between the highest and lowest value of i of an active cell on the grid is multiplied by the highest and lowest j value; the sum is divided by the total number of cells on the grid. The bounding box

ratio (BBR) represents the number of cells within its border in relationship to the total number of cells,

$$BBR = \frac{(i_{max} - i_{min}) * (j_{max} - j_{min})}{N_L} \tag{3.2}$$

This simple metric would appear to act as a very rough clustering of all the active cells. In a grid with a spanning cluster the value will approach one. A possible issue is that while the BBR might indicate if the active cells are grouping more closely together, it tells us very little about the actual grouping within the rectangle.

3.6.3 Entropy

Entropy is a term that has been used to describe a range of associated, but different measures (*see subsection 2.7.2 and subsection 3.3.4*). The analysis is focused on the 2D CA output space. This can be viewed as a macrospace, which would lend itself to an entropic measurement that takes a more information or uncertainty approach. The one selected is used by Tsang and Tsang [1999] in their study of cluster size diversity, percolation and complex systems.

> "The entropy of the system can be defined as a function of the probability that an occupied site is part of an s-site cluster."
>
> [Tsang and Tsang, 1999, p.2686]

The sum of s-clustered per grid is calculated by dividing the number of clusters of size s by the number of cells on the grid. This is also known as the normalised cluster number,

$$n_s = \frac{N_{k_s}}{N_L} \tag{3.3}$$

where N_{k_s} is the number of clusters of size s and N_L gives the number of cells in the grid.

The probability that a cluster of s cells contains an arbitrarily selected occupied cell is given by,

$$w_s = \frac{sn_s}{\sum_{s}^{L} sn_s} \tag{3.4}$$

Thus the entropy of the system is,

$$H = -\sum_{s} w_s \, ln \, w_s \tag{3.5}$$

This measure is used to help identify the critical percolation point, p_c, where a spanning cluster is formed. This means the curve of the graph can tail off after this point, giving little consistent information about the state of the grid once p_c has been reached and most of the active sites belong to a spanning cluster.

3.6.4 Connectedness

The measuring of a CA grid can focus on the comparison of the changing states of the individual cells over a series of time steps. However, the very visual nature of the CA grid lends itself to the evaluation of the changing nature of the whole grid over time. Clustering algorithms have been used both in the rules governing the action of agents on a CA, (see Chen and Wang [2007] - a novel ant clustering algorithm based on cellular automata), and to identify clusters in a CA. But this identified a cluster of cells, it did not provide any real information concerning the general state of the CA grid. This distinction between cluster identification as opposed to grid or dataset state was observed in subsection 3.3.5.

Taking the terms of graph theory, the cells are vertices and cells are connected to other cells in their neighbourhood via edges. This research uses levels of connectedness (edges) between active cells (vertices) to form connected groups, or clusters of cells. The number of edges in relationship to the number of vertices in a grouping indicates its density. The number and nature of the cell clusters, including singletons, within the CA grid provide an indicator of the degree of order; singletons, with no clusters, signify randomness, whereas a full or virtually full grid would correlate with a single, dense cluster and an ordered state. Moreover, this can be used to identify (a) similar areas within deterministic automata, (b) self-organisation and the loss of randomness in both agent and particle automata, and (c) the expansion, (increase in vertices and edges, against the decrease in the clusters of non-active cells), in all types of automata.

So in a grid the size of a Moore neighbourhood and where all the cells are active, there would be 9 vertices and 20 connecting edges (*see Figure 3.3 (a)*). This could be viewed as an optimally connected cluster. In the context of a larger grid the cluster might not be so tightly connected (*see Figure 3.3 (b)*); if the 9 cells, or

vertices, were in a straight line, the number of connecting edges would be 8.

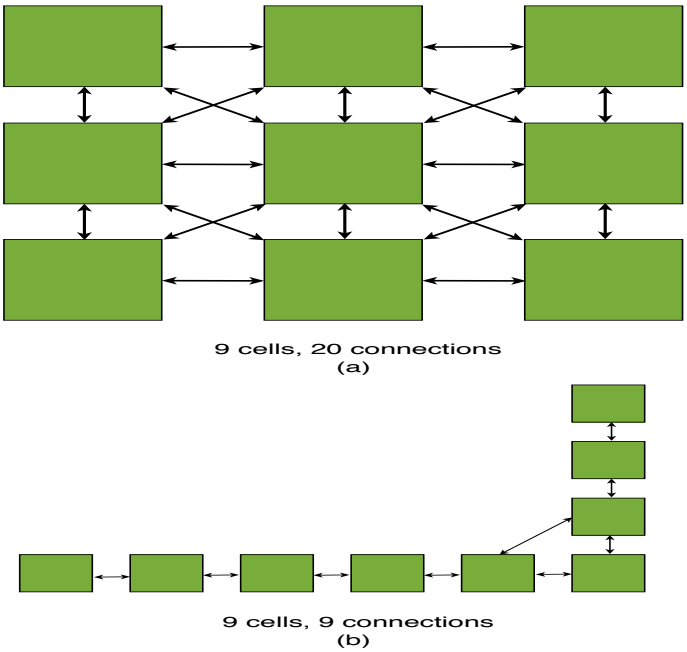


Figure 3.3: Connectedness between 9 cells using a Moore neighbourhood: (a) maximum connectedness of 20, (b) a more linear formation reduces the connectedness to 9

This gives another evaluation of the state of the active cells in the 2D CA output space, especially in the context of changes to the organisation of the active cells over a series of time steps. A cluster of 9 cells is a cluster, whatever their formation. So a measure of the mean density would register the same for any formation. Likewise, if all the active cells were in a single cluster, then the entropy would be the same. But measuring the number of edges provides the potential to say something about the changing state of that cluster as its number of edges show it moving from or towards its optimal connectedness. If this is used for all the elements on a grid then there is potential to provide a metric that provides information on the state of the whole grid.

The aim is to find clusters of cells connected at different depths using the Manhattan distance. The CA grid is scanned first for active cells or vertices that are in the same neighbourhood; each pair of connected vertices is deemed to have an edge between them; this is repeated until the whole CA grid has been scanned. This can then be repeated for cells that are one cell apart. The Manhattan distance can be specified up to a limit of 3, unless no clusters have been found, in which case it can be increased to one less of half the width, and the same for the height

if the grid is not a square.

The optimum number of edges for a cluster is given by,

$$O_{k_s} = \varphi(s) \tag{3.6}$$

where φ is the relevant calculation laid out in pseudo code (*see Figure 3.4*).

The number of connection or edges in the site is given by,

$$N_e = \sum_{k}^{L} k_e \tag{3.7}$$

where k_e is the number of connections in a cluster.

The optimum number of edges for a cluster with a size equal to all the active cells in a grid is,

$$O_{L_{\sigma=1}} = \varphi(N_{\sigma=1}) \tag{3.8}$$

The metric calculates the number of edges in the site and divides it by the optimal connectedness for a cluster of all the active nodes (including singletons),

$$C\text{-}Value = \frac{N_e}{O_{L_{\sigma=1}}} \tag{3.9}$$

The C-Value offers a possibility of distinguishing differences both in low and high occupied grids. The question then becomes whether that provides any useful information across the different types of CA.

Figure 3.4: Pseudo code for calculating the optimum number of edges for nodes in a Moore or a von Neuman neighbourhood

```
# a cluster must have at least 2 nodes;  treat 2 and 3 nodes as known
# cases with one and 3 connections respectively
# set up the variables:
rows = square root of the number of nodes, rounded up
columns = the quotient of the number of nodes divided by the rows
remainder = the modulo of the number of nodes divided by the rows

# (A) calculation for a Moore neighbourhood.
# get the connections of the first column, optimum is 6 for the first
# 4 nodes, and 5 for each additional completed square:
starter_block = 6 + ( 5 * ( rows - 2 ) )
# the rest of the columns of squares have 5 for the first completed
# square and 4 for each of the rest in each column:
other_blocks = ( columns - 2 ) * ( 5 + ( 4 * ( rows - 2 ) ) )
# calculate the remaining connections
# corner nodes have 2 when it is part of an incomplete row.
# inner nodes can have 4 when added to an incomplete row, or 3 if
# it is the first node in an incomplete row with no corners:
if the remainder is greater than zero
  if it is one less than competing another row
    set surplus to 2 for one corner node
                                plus 4 for each subsequent node
  else
    set surplus to 3 for the first node
                         plus 4 for each subsequent node

the optimum number of edges = inner_block + other_blocks + surplus
-----------------------------------------------------------------------
# (B) calculation for a von Neumann neighbourhood.
# get the connections of inner rectangle, optimum is 4 for 4 nodes:
inner_block = ( (rows - 1) * columns ) + ( (columns - 1) * rows )
# calculate the remaining connections
# corner nodes have a maximum of 2 connections; inner nodes have
# 2 when added to an incomplete row. The first node in
# any incomplete row has only 1 connection:
if the remainder is greater than zero
  surplus = 1 for the first node and 2 for each subsequent node

the optimum number of edges = inner_block + surplus
```

3.7 Summary

This chapter outlined the methodology and methods used in the thesis. It opened with the adoption of a simulation methodology. As discussed in subsection 2.5.1, a simulation is externally validated, as opposed to the internal validation of an experiment. In this thesis the three simulations are based on previous work, meaning that their validation is against the recorded results in the original works. In effect, the simulations are validated against the 'real world' by proxy through the documented results of the original work. Successful validation signifies that the original work has been reproduced and also contributes to validity of the original work itself.

A CA can be used for a variety of modelling situations, including ones where the state of the cells is of interest and others where the focus is on the movement of agents across the whole grid. Each focus has a different approach to what is of specific significance in terms of measuring and analysing. This can range from a simple assessment of the density of active cells, to the identification of clusters on the grid. A metric is chosen with the specific focus on the context of the simulation. The output grid of a 2D CA is a very visual space that has a limited scope for representation; although the breadth of domains modelled with CA is extensive. The simplicity of the 2D CA environment means that many attributes of one metric can be seen in another. Thus, the density or clustering of active cells have mutual points of interest, such as a spanning cluster that will generally signify a high density of active cells. Likewise, high entropy can also be seen as high density, and both low entropy (in terms of the macrolevel view presented by the grid), low density and low levels of clustering could be construed as indicating a higher chance that any active cells are randomly spaced on the grid.

Analogue simulation can be used to show an experiment of one system that can be used as a simulation of a diverse system. In this way the specific metrics used in the experiment also have relevance in the simulation. But such generality between diverse systems is difficult to establish, although the drawing of analogies between different observed contexts and systems is seen in how humans interact with new situations and experiences. A more accessible and useful approach in establishing some concept of generality between diverse systems is the measurement of a common modelling output space. The 2D CA output space provides an ideal common area used for the modelling of diverse systems. A focus on the state of the whole grid can be used to establish a common metric that is useful across different types of system.

Models covering three of the CA types described in subsection 2.6.5, *particle, randomised* and *deterministic* were outlined, as well as the original work they are based on. Four metrics, *mean density, BBR, entropy* and a newly proposed *C-Value* were then explained. These were selected to cover the four approaches listed as appropriate for measuring a 2D CA output space, *density, clustering (including randomness), entropy* and *connectedness*. The three establish metrics are all perceived to have a problem with measuring all three of the models, whereas the new C-Value metric is felt to have the potential to provide information on all three of the models, either individually or combined with one of the other metrics.

Chapter 4

Modelling

4.1 Introduction

The methodology and methods employed in this thesis were outlined in chapter 3, including the scenarios and models used in the thesis. This chapter gives further details of the implementation of the models and their validation against the original work they replicate. The scenarios used are explained and the process of using them to verify the working of the code implementing the metrics and analysis of the simulations is started.

The chapter begins with an explanation in section 4.2 of the types of neighbourhoods, their range and the boundary conditions used in the three simulations. Relatively simple fixed boundary conditions are used with the rectangular grids that feature in the particle and deterministic models. Whereas the randomised model employs a more complicated hexagonal grid with a wrap around toroidal boundary, making the 2D grid effectively a torus; this makes it a more realistic representation of, for example, the development of skin pigmentation on an animal. The choice of the development and programming environments are outlined in section 4.3, including examples of the how information is stored and passed between the various programs.

In section 4.4 the scenarios are explained. The hand-crafted scenarios show how the neighbourhood chosen can effect how the active cells are grouped. The test plans outlined for the probability and dynamic scenarios are implemented in chapter 5. The three models, their rules and implementation are discussed in section 4.5. Simulation examples are validated against the output of the original work from which the models are derived. A test plan for each model shows the simulations that will be run and analysed in chapter 5. The chapter concludes

with a summary.

4.2 Neighbourhood

There are two types of neighbourhood used in the models with rectangular grids, and one for the model with a grid of hexagonal cells:

- the preferred nine cell Moore neighbourhood incorporating the cell in focus and the eight cells bordering it;
- the five cell von Neumann neighbourhood using the cell in focus and the cells bordering it to the north, east, south and west ; and
- the seven cell hexagonal neighbourhood using the cell in focus and the six cells bordering it.

The neighbourhood also has a range or depth as well as boundary conditions.

4.2.1 Boundary conditions

The boundary conditions determine how the update rules are applied to a cell on the edge of the grid. The four main boundary conditions are usually given in terms of 1D CA (*see Chopard and Droz [1998, pp.15-16]*):

- *fixed* where the missing neighbouring cell is assumed to to have a predefined fixed value. Obviously in a two state CA this would be one of the two states;
- *reflection* where the missing cell has the value of an existing neighbouring cell. So a cell on the east border of a CA grid will reflect the value in its existing western neighbour into the missing eastern neighbour;
- *adiabatic* where the missing cells are given the value of the border cell ([Schiff, 2007, pp.47-49] refers to this as reflective); and,
- *periodic* where the grid wraps around and is treated as a torus; this is also referred to as a *toroidal* boundary condition.

The particle and deterministic models employ a null fixed boundary condition for the rectangular grids. This in effect means that only the existing neighbours are taken into consideration. The randomised model uses a grid of hexagonal cells and a periodic boundary condition. These choices reflect the more obvious options for a 2D CA grid, where any agent either leaves the grid and enter the surrounding environment to have no further effect on the grid (null fixed boundary condition), or leaves the edge of a grid to reappear on the opposite edge (periodic or toroidal boundary condition).

4.2.2 Range

The range is relevant to both the depth of the neighbourhood and the depth between cells in a cluster. In both cases a Manhattan distance is used with a minimum range of 1 and a maximum of 3. In the simulations using a rectangular grid and a Moor neighbourhood, a cell with coordinates of *i,j* (row, column) has an expanding neighbourhood where the the cell in focus has eight neighbours when the range is 1, a further sixteen when it is 2 and an additional twenty-four when it is 3 (*see Table 4.1*). The cells are stored in nested arrays reflecting their

Table 4.1: Cartesian coordinates for Moore neighbourhood of *i,j* for a range of 1, 2 and 3.

i-3, j-3	i-3, j-2	i-3, j-1	i-3, j	i-3, j+1	i-3, j+2	i-3, j+3
i-2, j-3	i-2, j-2	i-2, j-1	i-2, j	i-2, j+1	i-2, j+2	i-2, j+3
i-1, j-3	i-1, j-2	i-1, j-1	i-1, j	i-1, j+1	i-1, j+2	i-1, j+3
i, j-3	i, j-2	i, j-1	**i, j**	i, j+1	i, j+2	i, j+3
i+1, j-3	i+1, j-2	i+1, j-1	i+1, j	i+1, j+1	i+1, j+2	i+1, j+3
i+2, j-3	i+2, j-2	i+2, j-1	i+2, j	i+2, j+1	i+2, j+2	i+2, j+3
i+3, j-3	i+3, j-2	i+3, j-1	i+3, j	i+3, j+1	i+3, j+2	i+3, j+3

coordinates. The location is made up of $(i * number\ of\ columns\ in\ a\ row) + j$, and is used to build and move around temporary lists of cell coordinates.

The calculation of the neighbourhood and range of an hexagonal grid follows similar principles, but is slightly more complicated in its implementation. The grid used has cells rotated so that a flat edge is on top (*see Figure 4.1*). The cells are stored in location order; this facilitates the drawing of the grids, as well as storage and retrieval. But any manipulation is performed on the unpacked row, column coordinates.

The layout of each row of the grid is split into an upper and lower level. The cells in column zero and the even columns are in the 'upper' half, the odd numbered columns in the 'lower'. Any upper cells will have cells to the south east and south west on the same row, but will have to go up a row for their north east and north west cells. Conversely, lower cells have north neighbours on the same row, but need to go down a row for their southern ones. This becomes more involved when taking into account the wrapping around effect of a toroidal boundary where a cell might be looking for its neighbours from both the row and column on the other visual side of the grid. Consequently, an increase in the range is far from straight forward, (*see Figure 4.1*). The listing of sample code, which include the bases of

the algorithm used to work out neighbouring cells in the delta-notch hexagonal program suite, can be found in the appendix C.

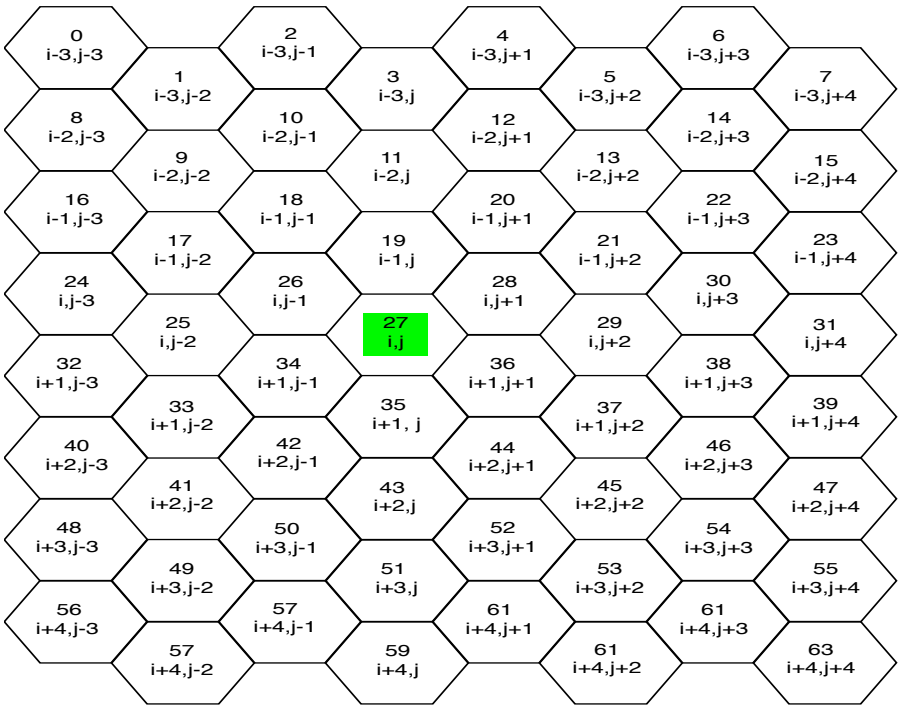


Figure 4.1: Cell location with row (i) and column (j) relevant to the cell in focus at location 27. The grid is stored in a location based array to save storage, but the application of rules using a cell's neighbourhood and range is based on unpacking the location into row and column coordinates.

The packing of the row, column (i,j) coordinates into a location is achieved by:

$$\begin{aligned} \text{location} = \quad & (\ (\ \text{integer from } i \ /\ 2\ )\ *\ \text{number of columns}\ ) \\ & + j \end{aligned}$$

The unpacking is move involved:

$$\begin{aligned} i = \quad & ((\ \text{integer from } \textit{location}\ /\ \text{number of columns}\ )\ *\ 2) \\ & + \textit{location} \bmod 2 \\ j = \quad & \textit{location} \bmod \text{number of columns} \end{aligned}$$

A range of one, two and three are shown for a cell in a central position on an eight

by eight hexagonal grid in Figure 4.2. However, the difference can be seen when the boundary condition is brought into play. In the rectangular grids a null fixed condition is used, meaning that if i, j is on the right hand, or eastern border of the grid, then all the two leftmost columns in table Table 4.1 would be treated as if they were inactive. However, the hexagonal simulation uses a periodic or toroidal condition, so it wraps around the grid edge as if it is joined to the opposite side (*see Figure 4.3*). One of the advantage of this is that it better models the 2D representation of the outer layer of a 3D object, such as pattern formation on skin.

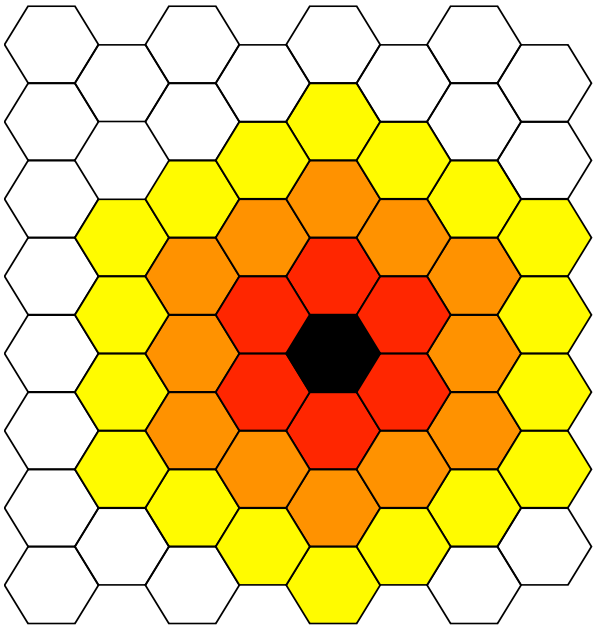


Figure 4.2: Range and boundary conditions for a hexagonal grid. This shows the range around the cell in focus (black) of 1 (6 red cells), 2 (6 red + 12 orange cells) and 3 (6 red + 12 orange + 18 yellow cells) for an eight by eight grid

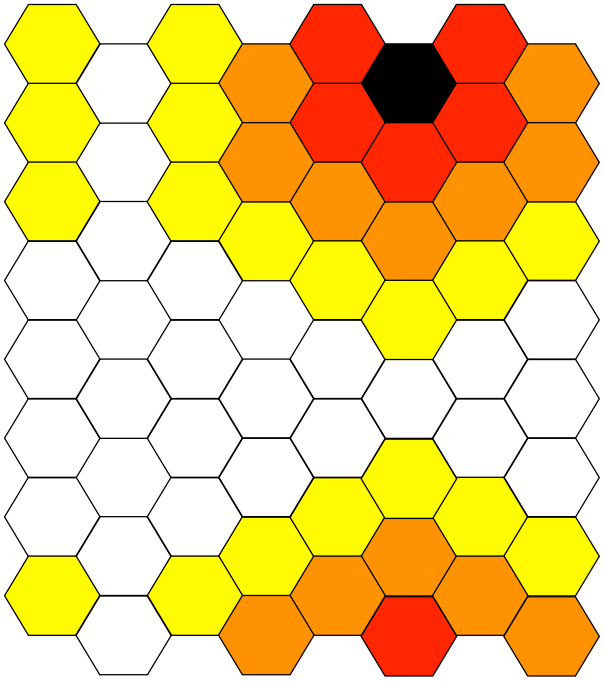


Figure 4.3: Range and boundary conditions for a hexagonal grid. The toroidal boundary conditions causes the range radiating from the (black) cell in focus to wrap around the grid.

The search for clusters follows the same neighbourhood, boundary conditions and can have the same range. All clusters are found within a range of 1. If it is specified, clusters of two cells apart can then be collated and then three, providing the cell has not already been allocated to a cluster of a smaller Manhattan size. The range is limited to three as this is the highest range that precludes cells being clustered across existing clusters. The only time that that the range is expanded to find a cluster is if no cluster has been found within the range specified. Any expanded search is stopped as soon as two nodes are found and the expanded range has been checked. This conforms with the underlying principle that a cluster must consist of at least two nodes.

4.3 Program suite

The programs to run the models and analyse the output using the four metrics outlined in chapter 3 were written in Java and Perl. The choice to program the models rather than utilise an existing tool set, such as Matlab or Wolfson's

Mathematica, was initially based on the intention to use the spatial and temporal features that was part of the MUSCLE modelling and simulation framework, (*see appendix A*), for the Active Network simulation used as the deterministic model. MUSCLE provides either a C^{++} or Java code base. The Java base proved easier to implement on the Ubuntu Linux operating system used to run the AN model. The analysis of the AN model's output and the programming and simulation of the two other models was carried out on the Apple OSX operating system, which has a Unix core and shares many run line utilities with the Ubuntu platform. All data was saved in a textual, record based format so that it could be easily passed between programs and manipulated by the various utilities available.

When considering the two other models and the measuring and analysis programs, it was felt that continuing to write programs, rather than take recourse in a third party tool, allowed greater control and flexibility. However, it was felt there would be greater benefit from using another programming language. Perl was chosen because of a number of reasons; it is (a) free, (b) a general-purpose scripting language, (c) an interpreted language and (d) available on most platforms. Although as an interpreted language Perl will not execute as fast as a compiled language, its development cycle is much faster. It is possible to develop and test in incremental and iterative cycles that are much reduced in time as the code does not have to be compiled. It also provides a succinct and flexible object orientated programming environment.

A number of utilities, both programmed and scripted, are used to facilitate the handling of the data output, such as extracting specific time steps. These utilities make use of the tools supplied as part of the Unix / Linux core. In general, all output from the programs are collected in the appropriate text or display file and relevant sub-directory (for example, *logs*, *displays*, *stats*, *graphs* and *thesis_files* directories). The output type is text based, including display files in *xml* and *html* format, except for graphs which are created as *png* files. The general procedure is for an output file to start with a date and time stamp consisting of the day of the month, hyphen, hour, minute, seconds and the ID, bounded by two underscores, of the running program, such as '24-153938_RDC_' or '24-153938_DNH_' for output from the particle (**R**eaction-**D**iffusion and **C**hemotaxis) and randomised (**D**elta-**N**otch **H**exagonal) models respectively. The tail of the output file name will reflect information about its running option followed by the relevant file type. An output file might have more than one date-time stamp and program ID if it has been run by more than one program.

Each Perl simulation or analysis program is invoked by a *pl* file which processes the runtime parameters and calls the relevant object modules. The objects, stored in Perl *pm* files and using the *Moose* extension to the Perl object oriented system, generally consist of a kernel or controlling object that contains an array of grid objects, that in turn contain an array of node objects, the latter being synonymous with the cells on a grid. The Perl kernel object will initially hold just the initial grid in their array, but this will increase to one grid per subsequent time step of the simulation. When a simulation is measured an array of grids showing each 'connected' group of cells is created for each grid in the array created by the simulation. The dynamic scenarios and models are invoked by calling the following programs with the relevant parameters[1]:

dynamic scenarios	TSM_Kernel.pl,
randomised model	RDC_Kernel.pl,
particle model	DNH_Kernel.pl and
deterministic model	ActiveMode.java

An outline of the main programs and scripts can be found in appendix D, and the actual programs and documentation are on the disk supplied with the thesis. The basic process for each model can be viewed in four steps:

(1) **initialisation** - create and populated a grid:

Any program creating an initial grid for use in a simulation will save the grid in a file starting with its ID and then information about the size of the grid, the number of active cells and any relevant creation rules. An initial grid is created and stored for repeated use (*see example in listing 4.1*).

The obvious advantage of saving the initial grid is not only to allow testing and assist program development, but also to show the different effect of changed program parameters, such as the neighbourhood type or the Manhattan distance used.

Each of the main simulations programs can create and save an initial grid, with the size of the grid and the number of occupied cells specified at runtime. They can also be supplied a file with the initial grid in. In the case of the particle model there are some simulations where the grid has blocked cells on the gird.

[1] the deterministic model is a series of Java modules that need to run in the MUSCLE framework and the simulation is invoked using a Ruby program. See appendix D for details.

Listing 4.1: Example of a stored initial 10 by 10 grid with 36 active cells set to '1'

```
GridSize=[10,10]
Grid=[0:0:0:0:0:1:1:0:1:0:1:0:0:0:0:0:0:0:0:0:0:0:1:0:
0:1:1:0:0:1:1:0:0:0:0:1:1:1:1:1:1:0:0:1:0:0:1:0:0:0:1:1:
1:0:0:0:1:0:1:1:0:0:1:0:0:0:0:0:0:1:0:1:1:0:0:0:0:1:1:
0:0:1:0:0:0:0:1:1:0:1:0:1:0:0:1:0:0:0:1:1:0]
```

(2) **simulation** - apply the model's rules over a number of time steps using the initial grid:

The simulations using rectangular grids pass their output to a program that runs the metrics; whereas the simulation based on the hexagonal grid with a toroidal boundary also contains the routines used to calculate the metrics as the hexagonal grid and toroidal boundary required different algorithms. The simulations using hexagonal grids can be run once to create a grid, run the simulation, save the state as a text for future processing, save the metrics as text and save the simulation as *xml*; or it can be run in a series of separate operations, providing the output of the simulation is saved as a textual representation of the state of the grid during each time step. The facility to re-process a grid, or simulation output, or to split the process from a single to a multi-stepped one is also available in the other core programs. A saved simulation will consist of a text file with multiple grids, each identified by a unique ID, such as the relevant time step. Each file starts with a record indicating the origin of the initial grid; the example below in listing 4.2 points to a simulation of the particle 'RDC' model , using a 10 by 10 grid with 40 active cells (*see section 4.6 for further details of this model*).

The rectangular grids can be saved and displayed as *html*; *xml* is used for the hexagonal grids as it enabled, through Perl, the construction of the hexagonal cells. This provides a means of easily conducting a visual inspection of the output. The saving of the grids in textual form also allows grids from specific time steps to be extracted and either rendered as *html* or *xml*, or measured and analysed. Indeed, as both *html* and *xml* are text and record based, selected time steps can be extracted directly from the displays files themselves and in the case of large grids, the display parameters can be adjusted to render the grids so that they can be observed and printed more easily. This is useful both from a testing and analysis perspective, and as a means of selecting key time steps for use within the thesis without having to re-process a full simulation.

Listing 4.2: Example of a stored simulation with the source file at the top and the first two time steps

```
Origin[input_grids/RDC_10by10a40.txt]
GridSize=[10,10]
TS=0
Grid=[0:0:0:0:0:1:1:0:1:0:1:0:0:0:0:0:0:0:0:0:0:1:0:
0:1:1:0:0:1:1:0:0:0:0:1:1:1:1:1:0:0:1:0:0:1:0:0:0:1:1:
1:0:0:0:1:0:1:1:0:0:1:0:0:0:0:0:0:1:0:1:1:0:0:0:0:1:1:
0:0:1:0:0:0:0:1:1:0:1:0:1:0:0:1:0:0:0:1:1:0]
TS=1
Grid=[1:0:1:1:1:0:0:1:0:0:0:1:0:0:0:0:0:0:0:0:0:0:1:0:
1:0:1:0:0:1:1:0:0:0:0:0:0:0:0:0:0:1:0:0:1:0:0:0:1:1:
1:1:0:0:1:0:1:1:0:1:1:0:1:0:1:0:1:0:1:1:0:0:0:1:0:0:0:
1:0:1:0:0:0:0:1:1:0:1:0:0:0:0:0:0:0:1:0:1:1]
```

(3) **measurement** - apply selected metrics to simulation's output:

The measurement of the grids created from a simulation is saved in a *STATS* text file. An example of the first five time steps of a the simulation of the particle model is shown in listing 4.3.

Listing 4.3: Example of the stored block of the metrics (STATS) for five time steps; more information is provided in the text above

```
Origin=[input_grids/rdc_20by20w80m2ts40ehS_grids.txt]
x_title=[time step]
LAYOUT=[MeanDensity, Entropy, SampleMean, BBR, C-Value]
STATS[0,400,1,M,3,0.2000,1.5509,0.1825,0.9025,0.2268]
STATS[1,400,1,M,3,0.1925,1.6124,0.1775,0.9025,0.2364]
STATS[2,400,1,M,3,0.1775,1.1787,0.1625,0.9025,0.2034]
STATS[3,400,1,M,3,0.1675,1.3697,0.1525,0.9025,0.2308]
STATS[4,400,1,M,3,0.1725,1.5160,0.1575,0.9025,0.2227]
```

In the first line the *Origin* shows the text file that contained the grids from the simulation and was subsequently processed. The *x_title* specifies the title to be used on the x-axis of any graph. The names of the metrics that were used are listed in the *LAYOUT*; these match respectively the end of each *STATS* line, so that the last value in the *STATS* lines shown above is the *C-Value*, the penultimate is the *BBR*, and so on. The *LAYOUT* names are also used in any graphs or tables created from the file The first five settings in a *STATS* line represent:

1. an ID - this is used to combine *stats* together in the analysis program. In the example this shows the time step of the relevant grid. It could be the probability value used to create the grid or the number of active cells, for example, where later analysis can calculate the average of the combined *STATS* with the same ID,
2. the number of cells in the grid; in the *Origin* used in the example the grid is 20 by 20 equalling, as shown, 400 cells,
3. the cell state value used to extract data for the statistical analysis; this is usually a '1' signifying an active cell,
4. the neighbourhood used - M = Moore, V = von Neumann, H = Hexagonal grid and,
5. the Manhattan range of 1, 2 or 3 used in the cluster analysis.

All simulations are stored in the textual grid format so that the metrics can be rerun and re-analysed.

(4) **analysis** - compare and analyse the results from the metrics:

The analysis of the *STATS* from a simulation or a collection of simulations is carried out for all the models by the S_Data modules, invoked by calling S_Data.pl. The main purpose of the analysis program is to output comparisons of the metrics and different simulations in the form of a graph or a table. The graphs are saved as *png* files and the tables are stored in the LaTeX *longtable* format, which facilitates their inclusion in the thesis. Examples of the graphs and tables created from the *STATS* can be seen in chapter 5.

4.4 Scenarios

An outline of the scenarios was given in subsection 3.4.1. They provide a relatively simple way of testing different output scenarios and verifying that the programs implementing the selected metrics and subsequent analysis are functioning as expected. The verification can be split into two key areas, (a) an identification of groupings of connected active cells, which will be referred to below as clusters, and (b) benchmarking the metrics. The ability to identify clusters is verified using hand-crafted stater grids and can be seen in subsection 4.4.1. Probability (subsection 4.4.2) and dynamic (subsection 4.4.3) scenarios are used to verify and benchmark the metrics in chapter 5.

4.4.1 Hand-crafted

The hand-crafted scenarios are one off grids used to verify the analysis program, especially the identification of clusters of connected active cells on a grid. They are created using a spreadsheet. Using a spreadsheet, a grid of cells of a selected size is populated with 0 or 1, the latter indicating an active cell. The grid is then exported as a *csv* file. This is then converted by a Perl program into the required grid layout for either further processing or analysis. Although a small series of grids could be created to represent a brief time series of agents moving in or out of a cluster, such a sequence is dealt with by the dynamic scenario (*see subsection 4.4.3*).

Test plan

A sample of the hand-crafted grids were created to test the identification of clusters using both the Moore 8 cell neighbourhood and the von Neumann 4 cell neighbourhood. The results of using the Moore neighbourhood to extract clusters of neighbouring cells sharing an edge from fairly simple patterns is shown in the next three figures. In the first a grid with five nested clusters of square outlines of connected active cells are extracted in Figure 4.4.

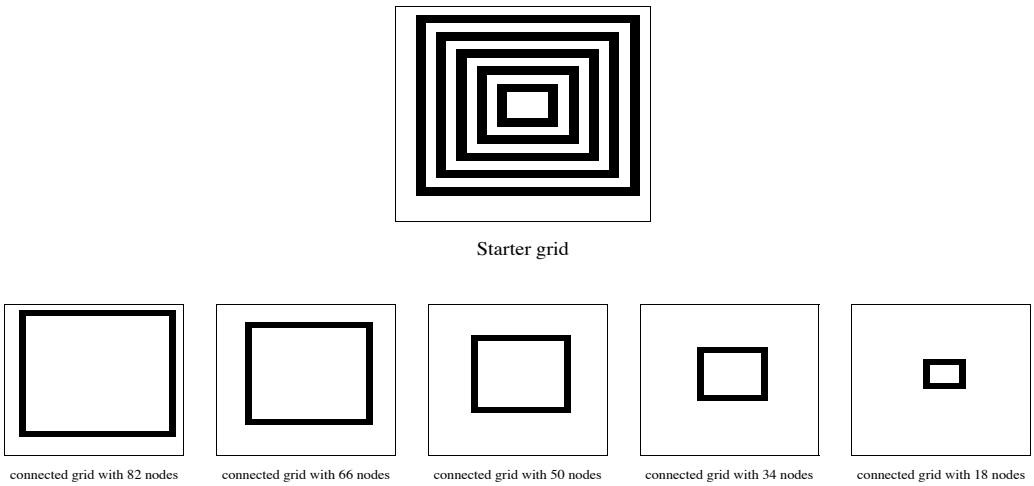
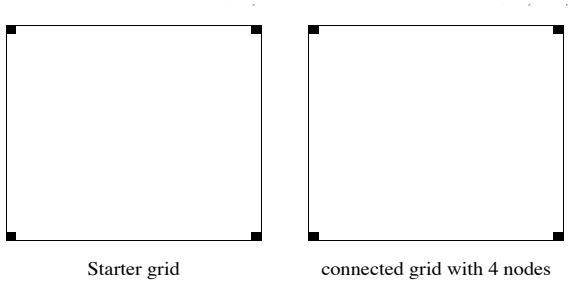


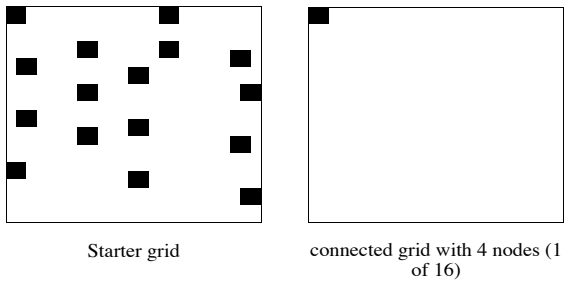
Figure 4.4: The extraction of clusters using the Moore eight cell neighbourhood. The active cells are shown as black. The starting grid is at the top. Five clusters are extracted.

In the next, Figure 4.5, three starting grids are shown; (a) has four active cells, one at each corner of the square grid, which yields a single cluster of connected cells, (b) has a starter grid with a scattering of four cells clusters, all of which

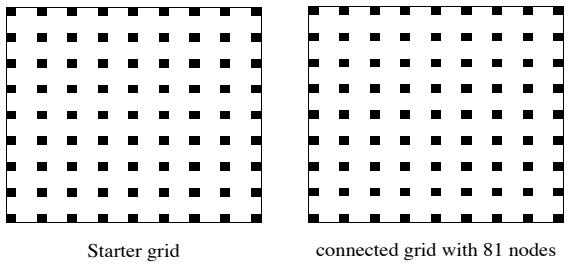
were found, and (c) has eighty one active cells evenly spaced across the starter grid, which is then extracted as a single cluster. The majority of the clusters found so far have a Manhattan distance of one, although the cluster in example Figure 4.5(c) has a Manhattan distance of 3.



(a)



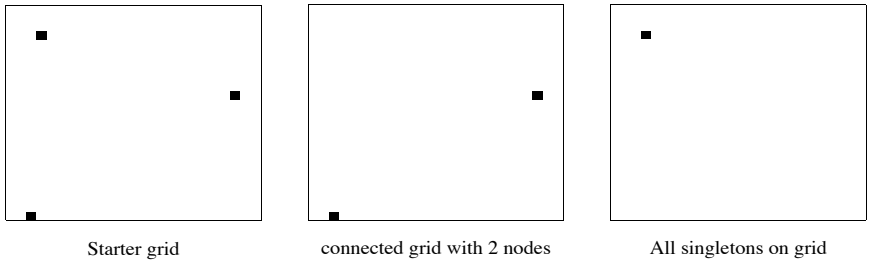
(b)



(c)

Figure 4.5: Three simple examples of hand-crafted grids and the identification of their respective groupings of connected cells using the Moore eight cell neighbourhood. The active cells are shown as black. The starting grids are on the left. (a) the four active cells are of equal distance apart on the square grid and form a single cluster; (b) the starter grid is made up of groups of 4 cell; just one of the 16 clusters is shown; (c) all of the cells on the starter grid form a single eighty one cell cluster

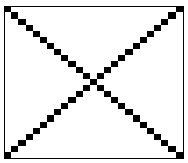
The last of the simple examples in Figure 4.6 are used to verify the ability of the written Perl program suite to identify clusters using a Moore neighbourhood uses a starter grid with just three active cells, spaced unevenly apart and at more than a Manhattan distance of three. As no clusters are found within a Manhattan distance of three, the search distance is increased, as was the case with Figure 4.5(b), until either a cluster is found, or the search exceeds the limits of the grid. In this example the nearest two active cells are grouped together and the remaining active cell is a singleton.



(a) (b) (c)

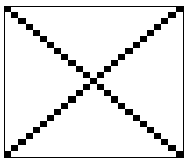
Figure 4.6: The three active cells (coloured black) are of unequal distance apart. Using a Moore eight cell neighbourhood, (a) the starter grid produces (b) a one two cell cluster and (c) a one singleton.

The next three figures show more intricate patterns and the difference in what is extracted by the two neighbourhoods. In Figure 4.7 the starter grid in (a) yields a single forty nine cell cluster in (b) using a the Moor neighbourhood. But in (c) a von Neumann neighbourhood does not include the diagonal cells next to the cell in focus, so one cluster of four cells at a Manhattan distance of 2 is found, while the other forty five cells are singletons.



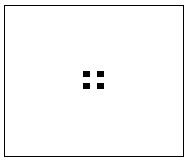
Starter grid

(a)

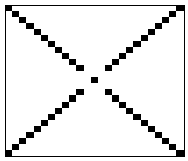


connected grid with 49 nodes

(b)



connected grid with 4 nodes



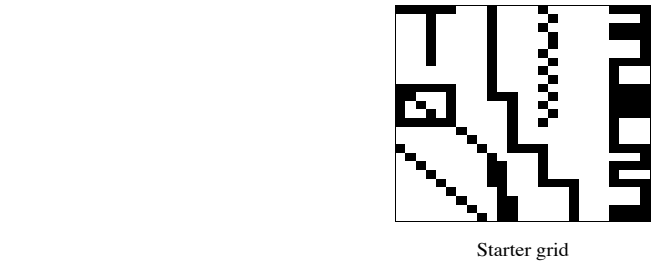
All 45 singletons on grid

(c)

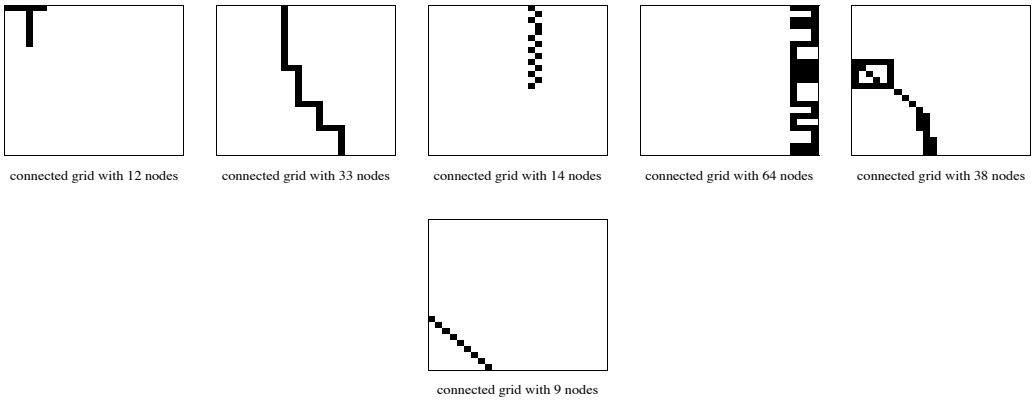
Figure 4.7: This shows the difference in cluster identification in (a) the starter grid at the top when using (b) a Moore eight cell neighbourhood where the starter grid at the top produces one cluster of all the active (black) cells and (c) a von Neumann neighbourhood where a cluster of four cells is found at a Manhattan distance of 2, see left hand grid, and 45 singletons, see right hand grid.

The contrast is even more evident in Figure 4.8. In (b) the Moor neighbourhood search of the starter grid in (a) groups all the active cells into six separate clusters. In (c) the first three clusters found with a von Neumann neighbourhood match three found in (b). But after that the other three clusters shown in(b) are replaced in (c) with six smaller clusters and fourteen singletons.

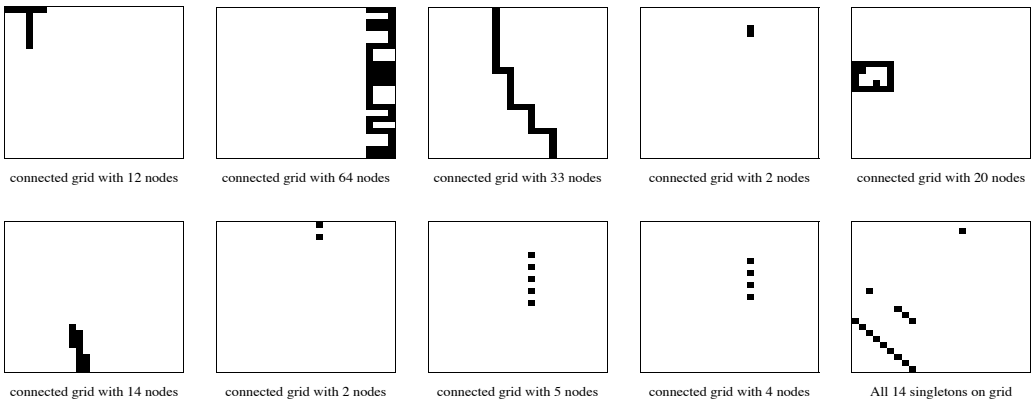
The final verification test using a hand-crafted scenario shows how the starter grid in Figure 4.9(a) is made up in (b) of two clusters using a Moor neighbourhood, but becomes in (c) one identical cluster, five smaller clusters and eleven singletons when using a von Neumann neighbourhood.



(a)

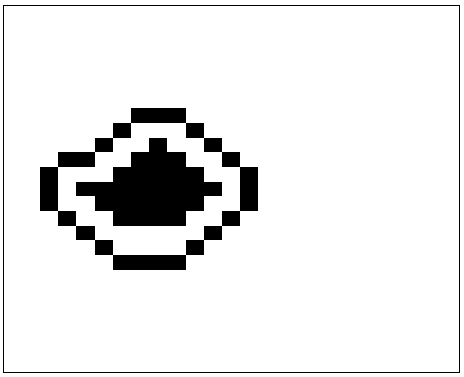


(b)



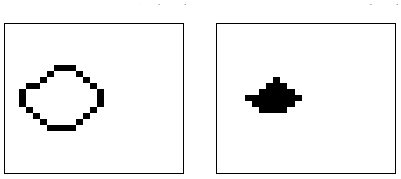
(c)

Figure 4.8: Another example of the difference in cluster identification in (a) between an eight cell Moore neighbourhood and a four cell von Neumann neighbourhood. (b) six clusters are found using the Moor neighbourhood; (c) a von Neumann neighbourhood identifies nine clusters and fourteen unconnected singletons that are displayed in the last grid



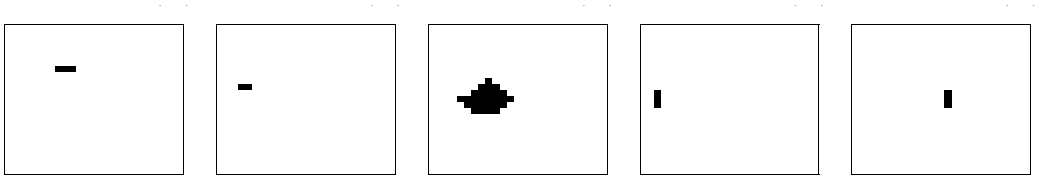
Starter grid

(a)

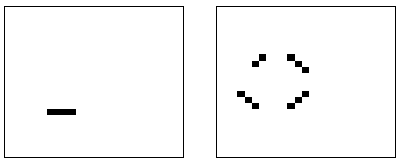


connected grid with 26 nodes connected grid with 27 nodes

(b)



connected grid with 3 nodes connected grid with 2 nodes connected grid with 27 nodes connected grid with 3 nodes connected grid with 3 nodes



connected grid with 4 nodes All 11 singletons on grid

(c)

Figure 4.9: A final example of the difference in cluster identification between an eight cell Moore neighbourhood and a four cell von Neumann neighbourhood. (a) the starter grid at the top produces the two clusters in (b) when a Moor neighbourhood is used ; in (c) the use of a von Neumann neighbourhood yields six clusters and eleven singletons (see the last grid).

4.4.2 Probability

The probability scenarios are created by a Perl program. The size of the grid is supplied as well as a probability value where $0 < p\_value < 1$. A random number between zero and one is generated for each cell in the grid; if the random number is equal to or less than the p_value the cell is set to one (active), otherwise it is set to zero (inactive). This means that a low p_value will result in a grid with a low number of active cells, whereas a p_value of 0.9 will create a grid with a large percentage of active cells (*see Figure 4.10*). This method is used by Tsang and Tsang [1999] in their investigation of cluster size diversity and percolation threshold, and the measurement of complexity.

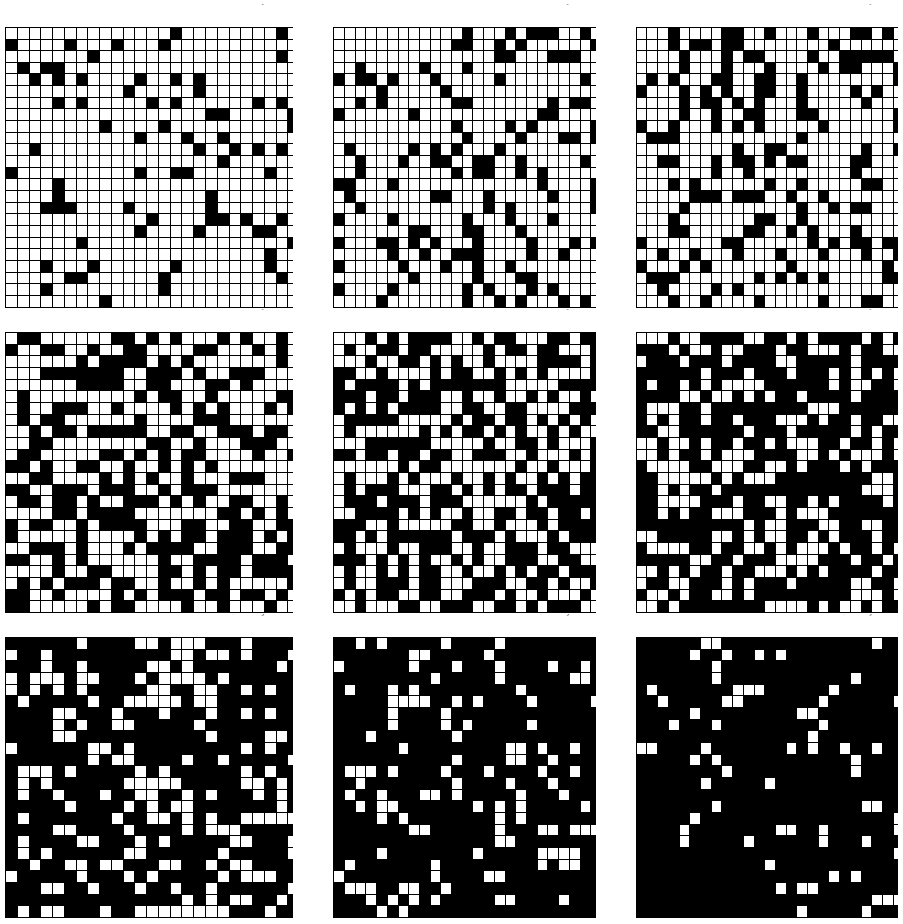


Figure 4.10: Examples of nine different probability grids. The active cells are shown as black. The first grid in the top left corner was created with a p_value of 0.1. The p_value of the subsequent grids going across the page and down increment by 0.1 up to a value of 0.9 in the grid in the bottom right corner. As the probability increases, so too does the density of active cells on the grid. This creates scenarios where the metrics and analysis programs can be tested against variable densities and the impact of the density on clustering identification can be seen.

Test plan

Output files can be combined to consist of grids formed from a mixture of p_values and grid sizes before being analysed. These scenarios are used in chapter 5 to benchmark the metrics against different percentages of active cells on the grid. In this way the ability of the metrics and analysis programs to distinguish between different grid densities can be verified. The correlation between the number of different cluster sizes and density, as well as the basic identification of clusters can also be confirmed.

4.4.3 Dynamic

The dynamic scenarios are created using a Perl program with parameters that can be used to set the grid size and the number of active cells. The specified active cells are randomly spaced on the grid. The size of grid and number of agents are varied as, although the scenario is predictable, a key distinction is the number of agents or active cells on the grid. A densely packed grid will soon become a single cluster, as opposed to a handful of active cells spaced out on a large grid. Therefore, the main purposes of this scenario is to (a) baseline the metrics against a range of output grids that show scattered agents gradually grouping together and (b) ascertain how the metrics performed when there was little change in the output grid to be identified, such as when the initial grid was very densely packed with active cells. The created grid is updated for a supplied number of time steps where the active cells are asynchronously processed with a simple rule that moves them, if there is an empty cell, towards the centre of the grid. The metrics can then be tested against a series of output grids that shows a move towards a single dense cluster (*see Figure 4.11*).

Test plan

Dynamic scenarios are created to investigate the performance of the metrics in a number of situations, including (a) how much distinction can be seen in the information obtained from the time steps of a densely packed grid as they quickly stabilise into a single cluster and (b) what discernible range of information can be obtained from the time steps involved in moving a few active cells on a large grid into a cluster. A series of simulations were conducted and metrics compared (*see chapter 5*).

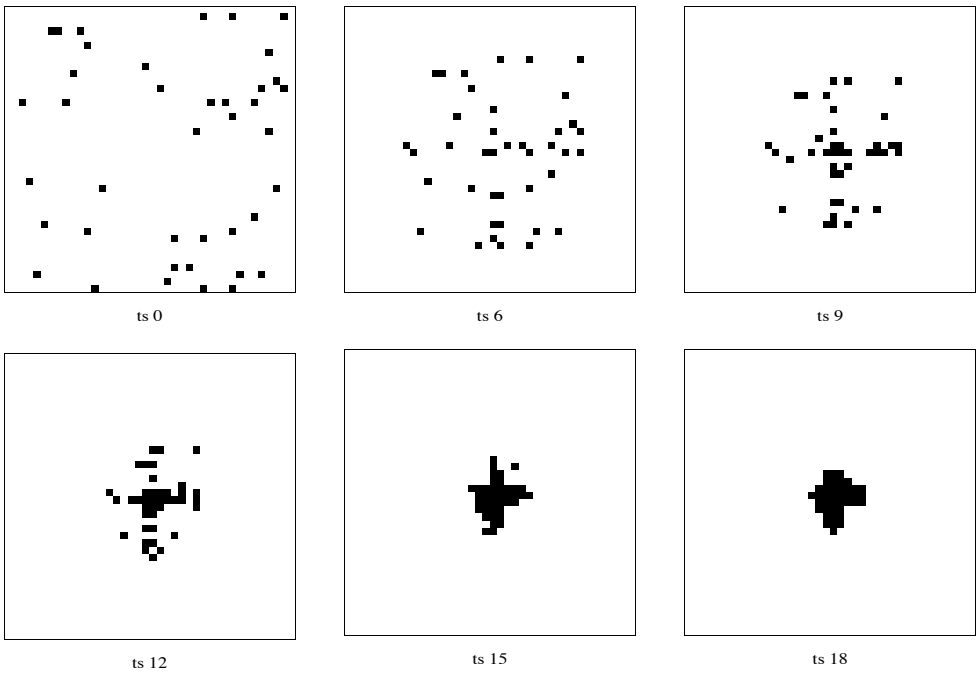


Figure 4.11: Example of dynamic scenario with a 40 by 40 grid and 40 agents (black cells). The agents cluster around the centre of the grid within eighteen time steps. Time steps (ts) 0, 6, 9, 12, 15 and 18 are shown.

4.5 Models

Three models have been built to reflect three of the four types of CA outlined in Table 2.2, *particle, randomised* and *deterministic*. The growth type is not modelled as it consists of the simple expansion or growth of the number of active cells that is easily measured; although the direction, pattern and speed of growth are of interest in some specific areas. All three models are based on existing work. The purpose of the following sections is to outline and validate the design of the three models and their operation against the relevant original work. The models were selected to provide an example of each of the three CA types. As will be seen below, the three models also have different features that ensure the variety tested is not confined to CA type, but incorporates different boundary conditions and updating rules.

4.6 Particle model

The particle model follows the design devised by Fatès et al. [2008] in their coupling of chemotaxis and reaction-diffusion to provide a solution to the decentralised gathering problem Their modelling of slime mould *Dictyostelium discoideum* gath-

ering in mounds in the event of food scarcity is considered in subsection 3.5.1. In the model the amoebae cells of the slime mould are synonymous with 'agents' or 'particles'. The model represents the amoebae signalling waves of cyclic adenosine monophosphate (cAMP) in reaction-diffusion patterns coupled with their chemotaxis in response to the cAMP. The rest of this section looks at the design of the particle model and validates the new simulations of the model against the original ones.

4.6.1 Particle model design

The model uses a synchronous update method, otherwise the reaction-diffusion waves are not formed and the chemotaxis is not sufficiently established to generate any meaningful gathering of the agents (*see the discussion on CA updating in subsection 2.6.3*). This can be seen in Figure 4.12 where the left hand image is a dynamically created 40 by 40 grid with 200 randomly placed active 'amoebae' (black cells) and a series of arrays of blocked (blue) cells of random length and distribution; it is updated for 3500 time steps asynchronously (middle grid) and synchronously (right hand grid). The rules and working of the model are explained below, but the figure shows how the desired wave pattern, represented by the red excited cells and the orange refractory ones, and resultant gathering of the amoebae is only achieved with the synchronous updating method. In the asynchronous grid displayed in the middle of the figure the amoebae are still scattered across the gird, and no evidence of any reaction-diffusion patterns can be identified.

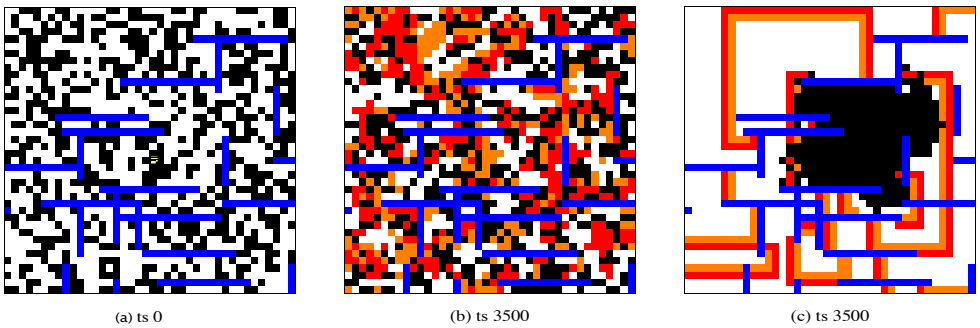


Figure 4.12: Example of asynchronous and synchronous updating in a 40 by 40 grid with 200 agents and with the excited value M set to 2. Cells containing amoebae are coloured black; red cells are in an excited state, orange ones are refractory; blue indicates a blocked cell. (a) initial grid configuration; (b) after 3500 time steps of asynchronous updating the agents are still scattered on the grid and the reaction-diffusion is patchy; (c) after 3500 time steps with synchronous updating the agents have gathered together in one region of the grid and the wave pattern of the reaction-diffusion is evident.

The design and rules of the model follow those documented in [Fatès et al., 2008]. An eight cell Moore neighbourhood is used, so the state of the cell in focus is not used. A null fixed boundary condition is applied, so cells on the edge of the grid have a reduced neighbourhood. Three probability settings have a potential impact on the execution of the update rules: (a) the transmission rate p_T; (b) the agitation rate p_A; and (c) the emission rate p_E. The model has two layers, the amoebae (or agent) and the environment. The update rules involve the linking of the two layers. The layers and their interaction are outlined in the following subsections.

4.6.2 Environment layer

The environment layer deals with the state of the cells, which ranges from a neutral state of 0, through to an excited state of M that is defined at run time. After one time step a cell in an excited state enters a refractory state where its state is lessened by 1 each subsequent time step until it reaches a neutral state. Thus, if M is set to 3, the possible state would consist of R={0, 1, 2, 3} and the refractory state of {1, 2} (*see Figure 4.13*).

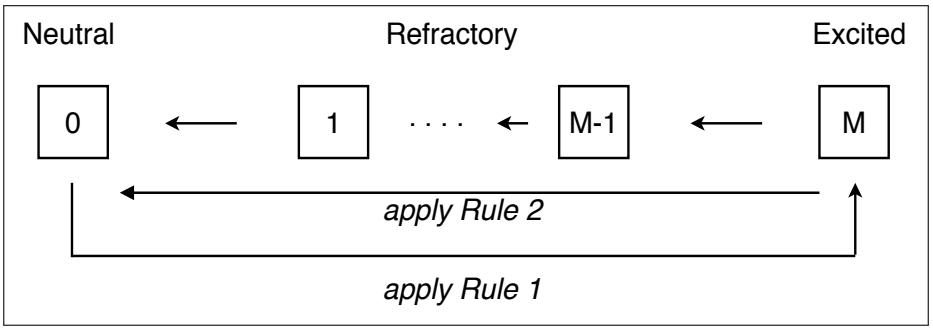


Figure 4.13: Environment layer with a simple reaction-diffusion process. A cell becomes excited if it is neutral ($\sigma_c^t = 0$) and there is at least one excited neighbour ($card\{E_{v_c}^t\}$, where V_c is the neighbourhood of the cell in focus c), and with probability of Bernoulli (p_T) = 1. A cell dissipates from an excited state through a refractory state by 1 each time step until it returns to a neutral state (*adapted from Fatès et al. [2008, pp.5-6] and Girau et al. [2009, p.3]*).

The rules governing any change to the state of a cell between time steps are:

Rule 1: a neutral cell becomes excited if it has at least one cell in its neighbourhood that is in an excited state and a randomly generated number is less than

or equal to p_T (*transmisson rate*);

$$\sigma_c^{t+1} = M \qquad \text{if } \sigma_c^t = 0 \text{ and } card\{E_{v_c}^t\} > 0 \text{ and } B(p_T) = 1 \tag{4.1}$$

Rule 2: a cell in an excited or refractory state dissipates by 1 every time step until it reaches a neutral state;

$$\sigma_c^{t+1} = \sigma_c^t - 1 \qquad \text{if } \sigma_c^t \in \{1, ..., M\} \tag{4.2}$$

Rule 3: a cell is neutral if none of the previous rules are applied.

$$\sigma_c^{t+1} = 0 \quad \text{otherwise} \tag{4.3}$$

These rules match R1, R2 and R3 outlined in [Fatès et al., 2008, pp.5-6].

4.6.3 Amoeba layer

The amoeba layer controls the movement and location of each amoeba at each time step. An amoeba is a single cell organism, but in the model it is not a one amoeba to one grid cell correlation as more than one amoeba can occupy a cell. The model represents the real life situation where there is a food shortage and the amoebae stop spreading out and instead form into a mound [Bretschneider et al., 1997]. As the update method used is synchronous, an amoeba wanting to move to a cell only knows the state of that cell at the last time step, so at the next time step multiple amoebae can occupy a cell. If a cell has one amoeba in it, after the synchronous update all eight neighbouring cells could, theoretically, have moved an amoeba into the cell. This would mean that the cell would have nine amoebae in it after the update. It would be unable to accept any more until at least eight further time steps had elapsed; more if it failed to move an amoeba off during a time step. A consequence of this, within a modelling tool such as CA, is that while there is a fixed number of amoebae on a grid, with no births or deaths, visually the number of amoebae (represented by black cells) might appear to fluctuate.

A cell is available for an amoeba to move to if it is *empty* or has less than two amoebae in it; the latter is termed *free*. The rules deals with free cells that are not in an excited state ($\tilde{F}_c^t$) and free cells that are in an excited state ($\tilde{E}_c^t$). A randomly selected cell from any free cells in the neighbourhood is denoted by $R(\tilde{F}_{v_c}^t)$ and a target cell is $\oplus_c^t$. While more than one amoeba might end up in a cell, only one amoeba can move from an occupied cell during the time step update. The movement of an amoeba is controlled by:

Rule 4: if a randomly generated number is less than or equal to the agitation rate then the amoeba is moved to a randomly selected free cell if any exists;

$$\text{if } B(p_A) = 1 \quad \text{then} \quad \oplus_c^t = R(\tilde{F}_{v_c}^t) \tag{4.4}$$

Rule 5: if the random number is greater than the agitation rate, then if the cell is neutral and there is at least one free excited cell in the neighbourhood the amoeba moves to a randomly selected excited free neighbouring cell;

$$\begin{aligned} \text{if } B(p_A) \neq 1 \quad \text{then} \quad & \text{if } \sigma_c^t = 0 \text{ and } card\{\tilde{E}_{v_c}^t\} > 0 \\ & \text{then } \oplus_c^t = R(\tilde{E}_{v_c}^t) \end{aligned} \tag{4.5}$$

Rule 6: otherwise no move occurs.

These rules correspond to R4, R5 and R6 given in [Fatès et al., 2008, p.6].

4.6.4 Layer interaction

The two layers interact following one rule:

Rule 7: if a cell in a neutral state is occupied by one or more amoebae ($A_c^t > 0$) then the cell is set to excited if a random number is less or equal to the emission rate.

$$\sigma_c^{t+1} = M \quad \text{if } \sigma_c^t = 0 \text{ and } A_c^t > 0 \text{ and } B(p_E) = 1 \tag{4.6}$$

The two rules that excite a cell (Rule 1 and Rule 7) can be combined together:

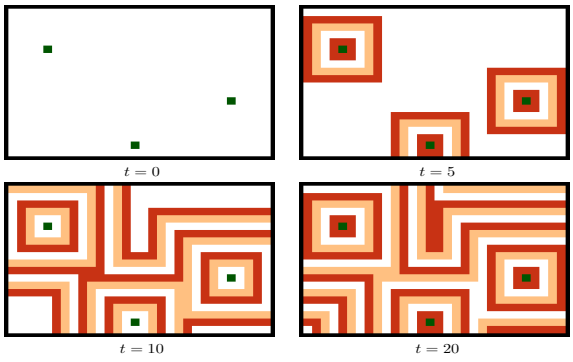
Rule 8: A cell becomes excited based on the transmission rate and a neutral state as well as there being either at least one excited neighbouring cell or that the cell is occupied and a randomly generated number is less than or equal to the emission rate.

$$\begin{aligned} \sigma_c^{t+1} = M \quad & \text{if } B(p_T) = 1 \text{ and } \sigma_c^t = 0 \\ & \text{and } (card\{E_{v_c}^t\} > 0 \text{ or } (A_c^t > 0 \text{ and } B(p_E) = 1)) \end{aligned} \tag{4.7}$$

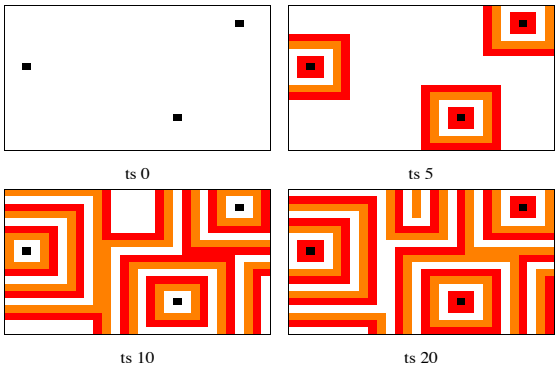
These rules replicate R7 and R1' given in [Fatès et al., 2008, p.6]. The model also follows their setting of $M = 2$ with a state range of R=$\{0, 1, 2\}$ [*ibid*, p.7]; this keeps the rules and states of the model as simple as possible.

4.6.5 Simulation examples

This section provides validation that the model and simulation, written in Perl, provides similar results to those published in [Fatès et al., 2008]. Their first simulation established that transmission and emission rates of 1 and an agitation rate of zero produced reactive-diffusion waves that met and then dissipated or merged, rather than interacting with any amoebae and producing chemotaxis. This meant that no gathering occurred and the amoebae remained static. The simulation was run on a 20 rows by 30 columns lattice with 3 amoebae. The number of amoeba is kept low so as to show the working of the model as clearly as possible. The results concur with the original work and they can be seen in Figure 4.14 along with the original results from [Fatès et al., 2008, p.8].



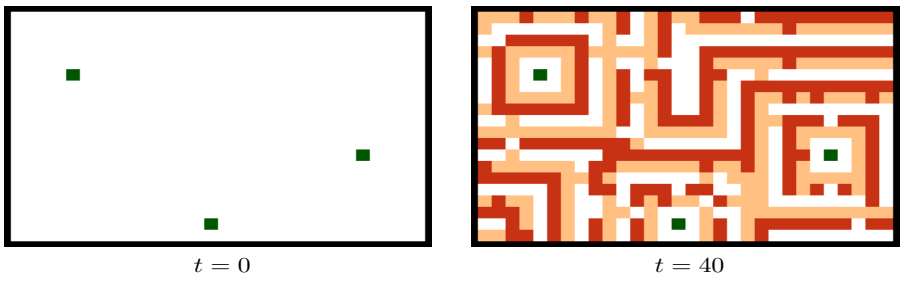
(a)



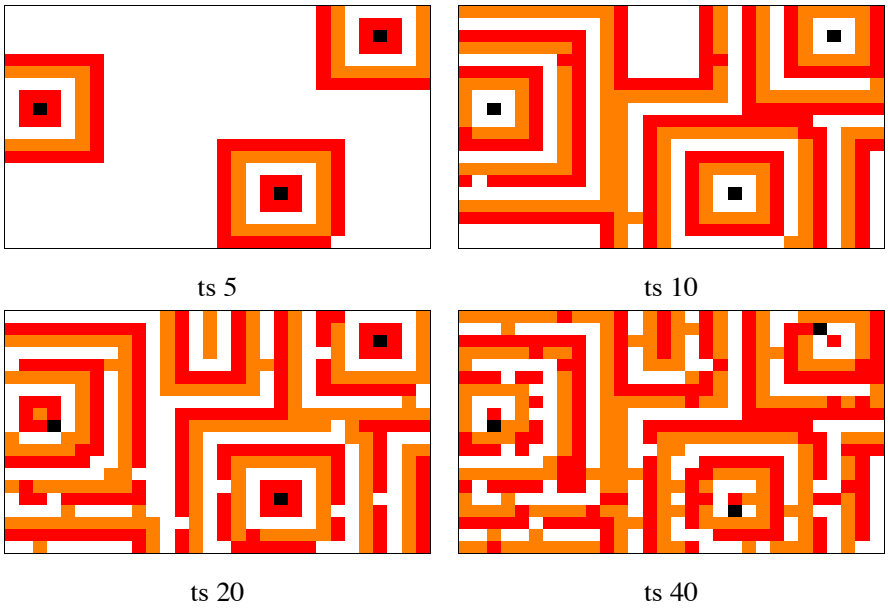
(b)

Figure 4.14: Fully deterministic static evolution with a 20 rows by 30 columns grid, 3 amoebae and rates of $(p_T, p_E, p_A) = (1, 1, 0)$. The cell state range is R={0, 1, 2}, with M=2; the resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as green cells in the original work and as black cells in the new simulations; the white cells are neutral. (a) time steps 0, 5, 10 & 20 from the original work in [Fatès et al., 2008, p.8]; (b) the output for time steps 0, 5, 10 & 20 from the simulation using the new Perl program; this concurs with the original work in (a).

In their second simulation Fatès et al. [2008] showed the effect of introducing a 1% possibility that neighbouring cells would not set the cell in focus to an excite stage. The setting of $p_T = 0.99$ for this non-coherent regime resulted in small transmission errors that caused waves independent of the amoebae and hindered any gathering process. The results are shown in Figure 4.15 and mirror the original ones.



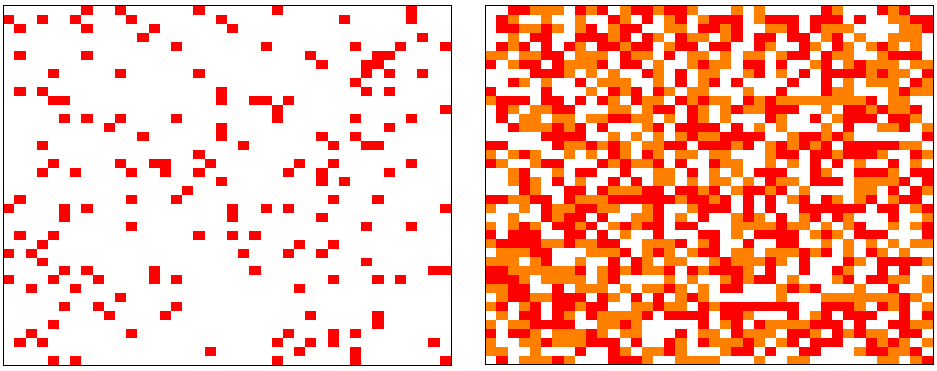
(a)



(b)

Figure 4.15: Non-coherent regime with same starting grid as the previous figure, but with rates of $(p_T, p_E, p_A) = (0.99, 1, 0)$. The reduction in the transmission rate effects the even spread of the reaction-diffusion wave pattern by introducing additional waves independent from the amoeba, thus impeding any gathering process. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as green cells in the original work and as black cells in the new simulations; the white cells are neutral. (a) this shows the initial and the fortieth time steps of the original work [Fatès et al., 2008, p.10]; (b) time steps 5, 10, 20 & 40 from the new model match those of the original work in (a).

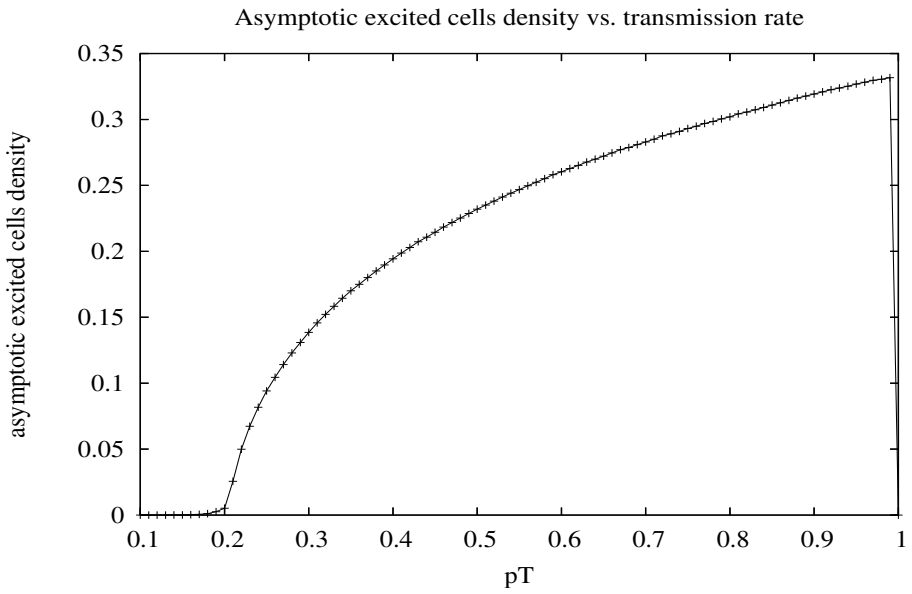
Fatès et al. [2008] also looked more closely at the effect of varying the transmission rate in their extinction regime simulation. In the simulation the initial grid has no amoebae on it. Instead an average of 10% of the grid is populated with a randomly located excited cells, all the other cells are left in a neutral state. The simulation is carried out for 0.1 increases of p_T from 0.1 to 1 inclusively; and with $(p_E, p_A) = (0, 0)$. As was seen in the non-coherent simulation above (*see Figure 4.15*), a setting of $p_T = 0.99$ led to a loss of coherence in the reactive-diffusion wave pattern (*see Figure 4.16*). In this simulation the final grid of each run is evaluated for the mean density of excited cells. Although the simulation was run with smaller grids and for less time steps, the results matched the original simulation (*see Figure 4.17*), and show that the "transition from non-coherent regime to the extinction regime is sharp and occurs for p_T 0.20" [Fatès et al., 2008, pp.10-11]. They see the graph's shape as an indication of second-order phase transition. Although there are no amoebae involved, the lack of any wave patterns in the end grid of $p_T = 0.3, ..., 0.9$, (*see Figure 4.16(b)*), resembles the results found when using an asynchronous updating method (*see Figure 4.12(b)*). This re-emphasises the lack of cohesion in such an updating method.



(a) ts 0 (b) ts 400

Figure 4.16: The extinction regime simulation. Red indicates an excited cell, and orange a refractory one. The white cells are neutral. There are no amoeba placed on the grid for this simulation. Results gained from a 40 by 40 grid over 400 time steps with $(p_E, p_A) = (0, 0)$ and a p_T range of {0.1, 0.2,...,0.9,1}. (a) an example of an initial grid consisting of neutral cells except for about 10% randomly located excited cells; (b) an example of the state of the grid with $p_T = 0.9$ after 400 time steps.

(a)



(b)

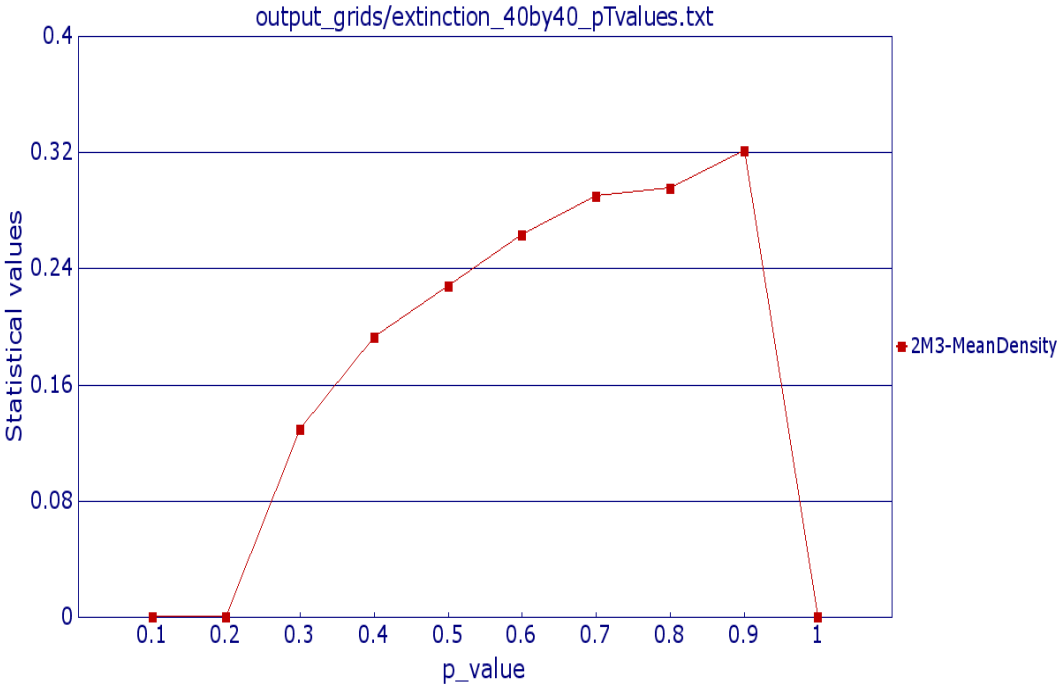


Figure 4.17: Mean density of excited cells as a function of p_T. (a) this shows the results from the original work [Fatès et al., 2008, p.12] where a 100 by 100 grid depleted of amoebae was initially set with 10% of its cells in an excited state; (b) this graph shows the mean density of excited cells for each of the p_T settings after 400 time steps and the transition at p_T 0.20 from non-coherence to extinction; a 40 by 40 grid was used. The results from the new model shown in (b) match those of the original work in (a).

In their final simulation Fatès et al. [2008, pp.13-22] looked at how the probability rates could be used to gather the amoebae into clusters, with and without perturbation. The measuring of the clusters is the focus of chapter 5, so the validation of this part of the new agent gathering model against the original work is an example of the gathering process using the last two settings laid out in Table 4.2 for quick self-organisation.

Table 4.2: Suggested settings for the static, non-coherent, extinction and self-organising regimes (*adapted from [Fatès et al., 2008, p.22]*)

p_T	p_E	p_A	**qualitative behaviour**
1	1	0	static
$[0.2, 1]$	any	0	non-coherent
< 0.2	any	0	extinction
1	0.10	0	self-organising (slow)
1	0.01	0	self-organising (quick)
1	0.01	0.2	self-organising (quick)

The first validation test used a 40 by 40 grid with 600 randomly located amoebae with a pure transmission rate and no agitation: $(p_T, p_E, p_A) = (1, 0.01, 0)$. The simulation was run for two thousand time steps. The results can be seen in Figure 4.18 and reflect the behaviour seen in the original simulation in [Fatès et al., 2008], which used a grid with $(X, Y) = (30, 20)$, around 60 amoebae and the same ratio settings (Figure 4.19). In all the remaining figures in this section the amoebae in the original work are shown as green cells, while in the new simulations they are depicted by black cells.

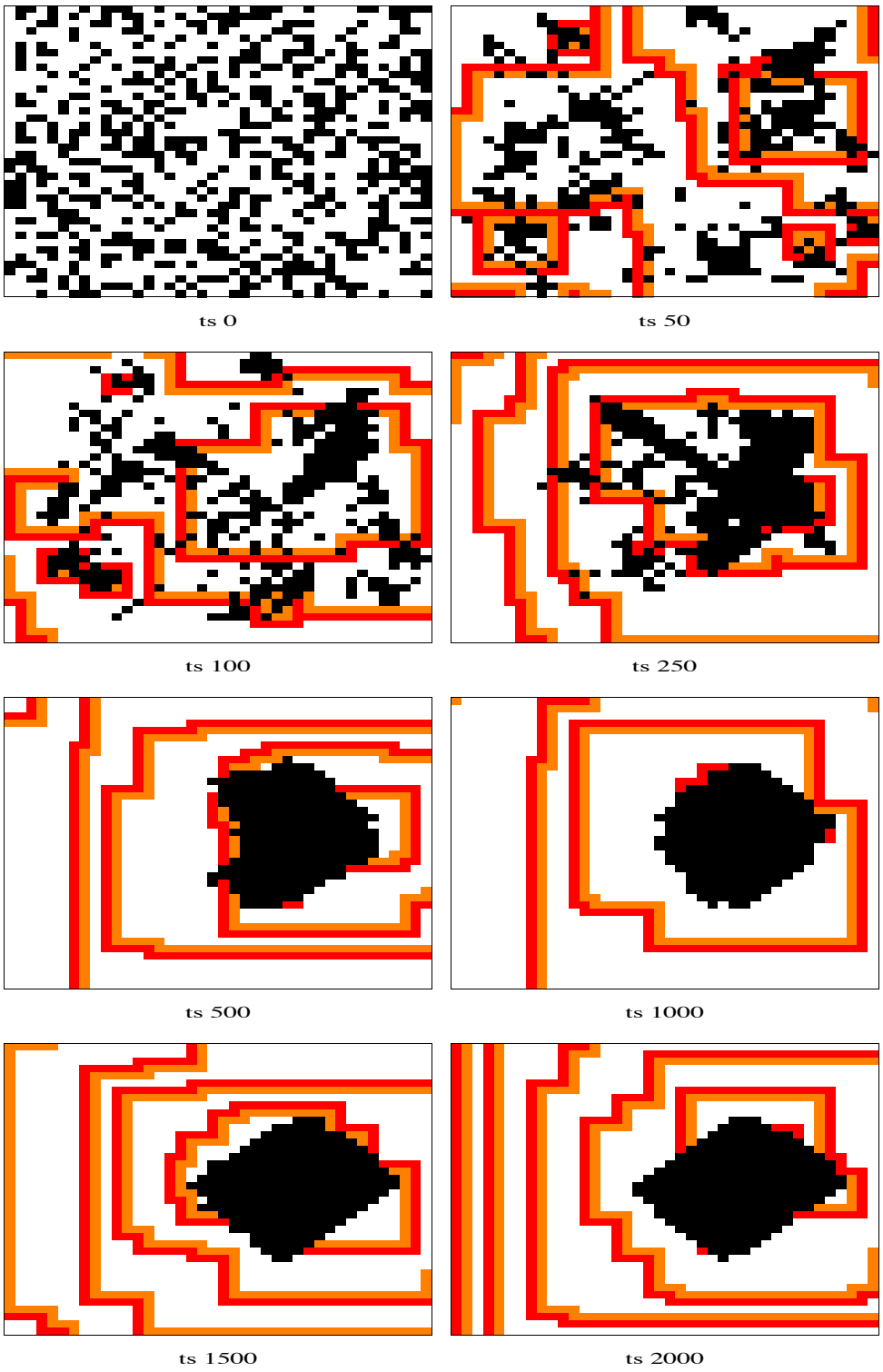


Figure 4.18: Example of the process of amoebae gathering through a simulation of reaction-diffusion and chemotaxis. A 40 by 40 grid was used with a starting grid of 600 amoebae and a probability ratio setting of $(p_T, p_E, p_A) = (1, 0.01, 0)$. The samples shown are, from the top left, time steps 0, 50, 100, 250, 500, 1000, 1500 and 2000. This reflects the findings of the original study shown in Figure 4.19.

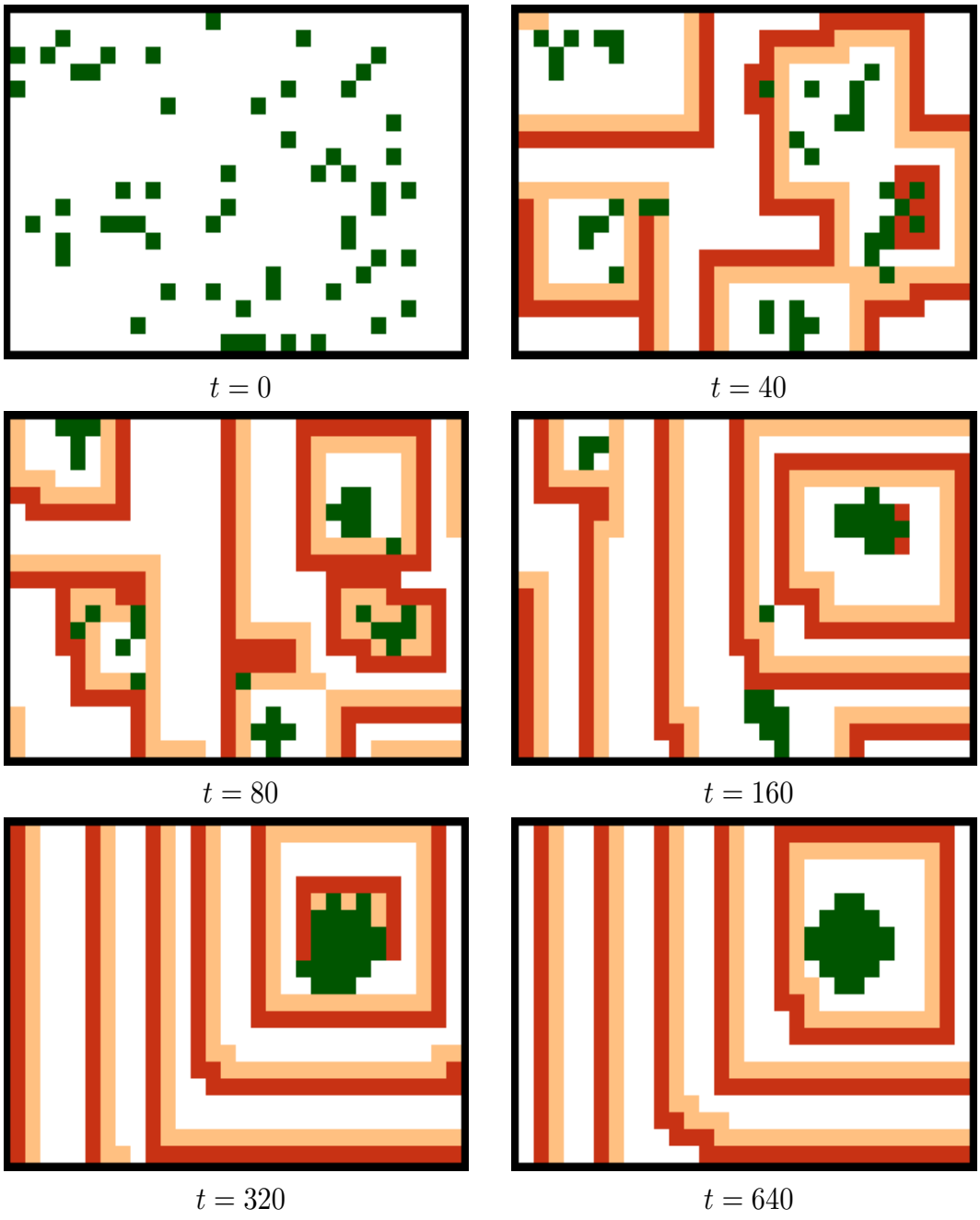


Figure 4.19: Process of amoeba gathering from the original work [Fatès et al., 2008, p.14]. A grid with $(X, Y) = (30, 20)$ is used with a probability ratio setting of $(p_t, p_E, p_A) = (1, 0.01, 0)$. A probability of 10% was used to initially populate a cell with an amoeba. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as green cells and the white cells are neutral. The samples shown are, from the top left, time steps 0, 40, 80, 100, 320, and 640.

The next three simulations use a much larger grid of $(X, Y) = (150, 100)$. The original work has a probability of 10% that a cell on the initial grid is populate with an amoeba; the new simulation randomly places an amoeba on 10% of grid. All three simulations are matched against output from the original work and show corresponding behaviour. The first also uses a pure transmission rate and no agitation. The results of $(p_T, p_E, p_A) = (1, 0.01, 0)$ on a large grid can be seen in Figure 4.20 and the compactness of the groups can be seen in both the new and the original results (Figure 4.21). The second simulation introduces a small agitation rate of 10%: $(p_T, p_E, p_A) = (1, 0.01, 1.0)$. The new results (Figure 4.22) again confirms the original findings (Figure 4.23), with both simulations showing groupings that appear less compact and show the effects of the low agitation rate. The third simulation use the same settings of the previous example, but places around 1500 blocked cells in vertical and horizontal lines onto the initial grid. In the final grid displayed of both the new simulation (Figure 4.24) and the original work (Figure 4.25), two decentralised groupings have formed with both showing the effect of the low agitation rate. In both simulations some amoebae have been blocked in and prevented from joining either of the groups. The three simulations concur with the original work in [Fatès et al., 2008] and show the gathering process, as well as the ability of the amoebae to negotiate around obstacles; something that "can be seen as an emergent property since at no time was it explicitly coded in the local rules governing the system" [*ibid*, p.22].

The new simulations in this section have replicated those from the original work, thus validating the new model against the original one in [Fatès et al., 2008].

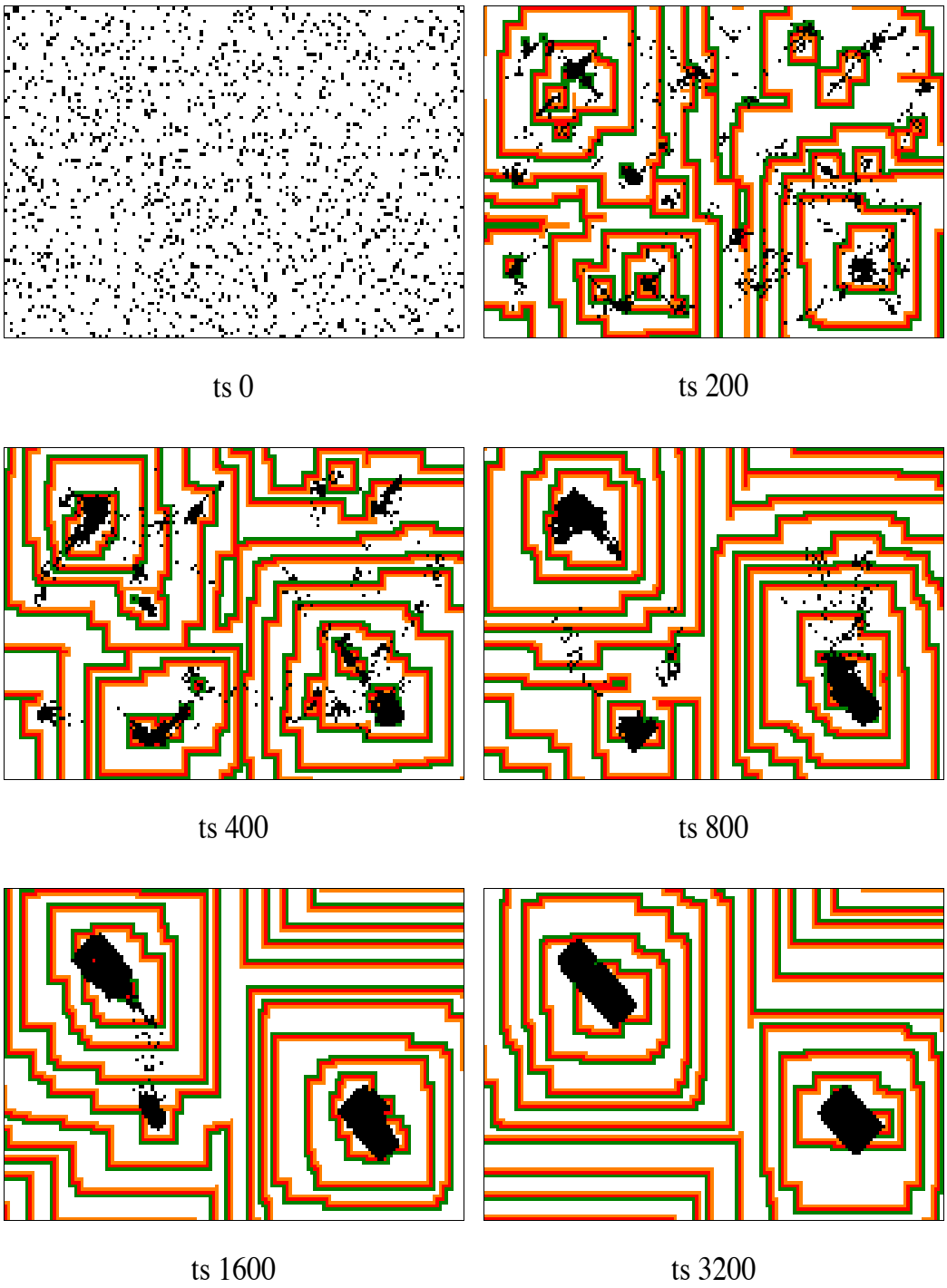


Figure 4.20: Example of the process of amoebae gathering through a simulation of reaction-diffusion and chemotaxis, and with a pure transmission rate and no agitation. A 150 by 100 grid with was used with a starting grid of 1500 amoebae and a probability ratio setting of $(p_T, p_E, p_A) = (1, 0.01, 0)$. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as black cells and the white cells are neutral. The samples shown are, from the top left, time steps 0, 200, 400, 800, 1600, and 3200. This reflects the original work shown in Figure 4.21.

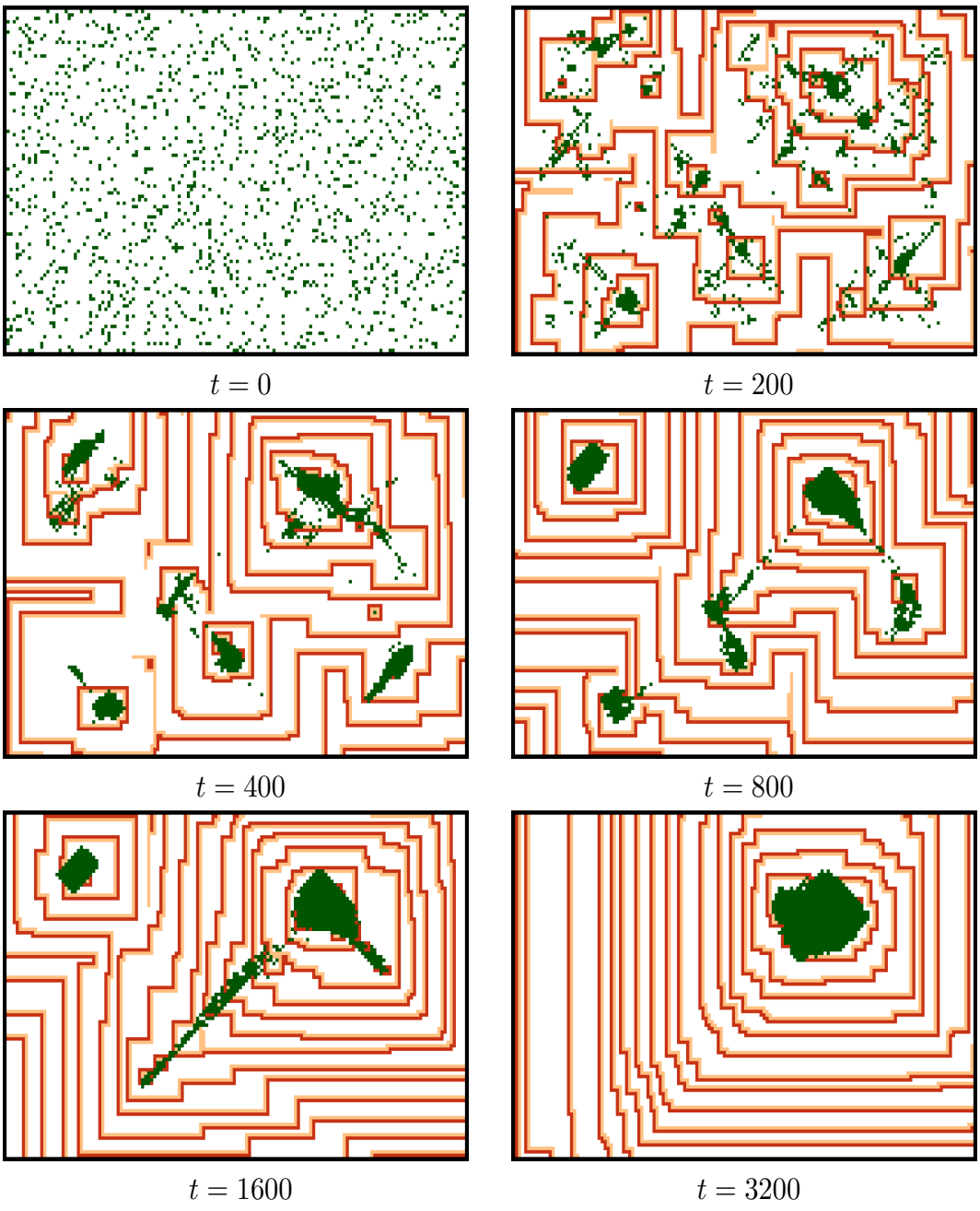


Figure 4.21: Process of amoeba gathering with perfect transmission and no agitation from the original work [Fatès et al., 2008, p.17]. A grid with $(X, Y) = (150, 100)$ is used with a probability ratio setting of $(p_T, p_E, p_A) = (1, 0.01, 0)$. A probability of 10% was used to initially populate a cell with an amoeba. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as green cells and the white cells are neutral. The samples shown are, from the top left, time steps 0, 200, 400, 800, 1600 and 3200.

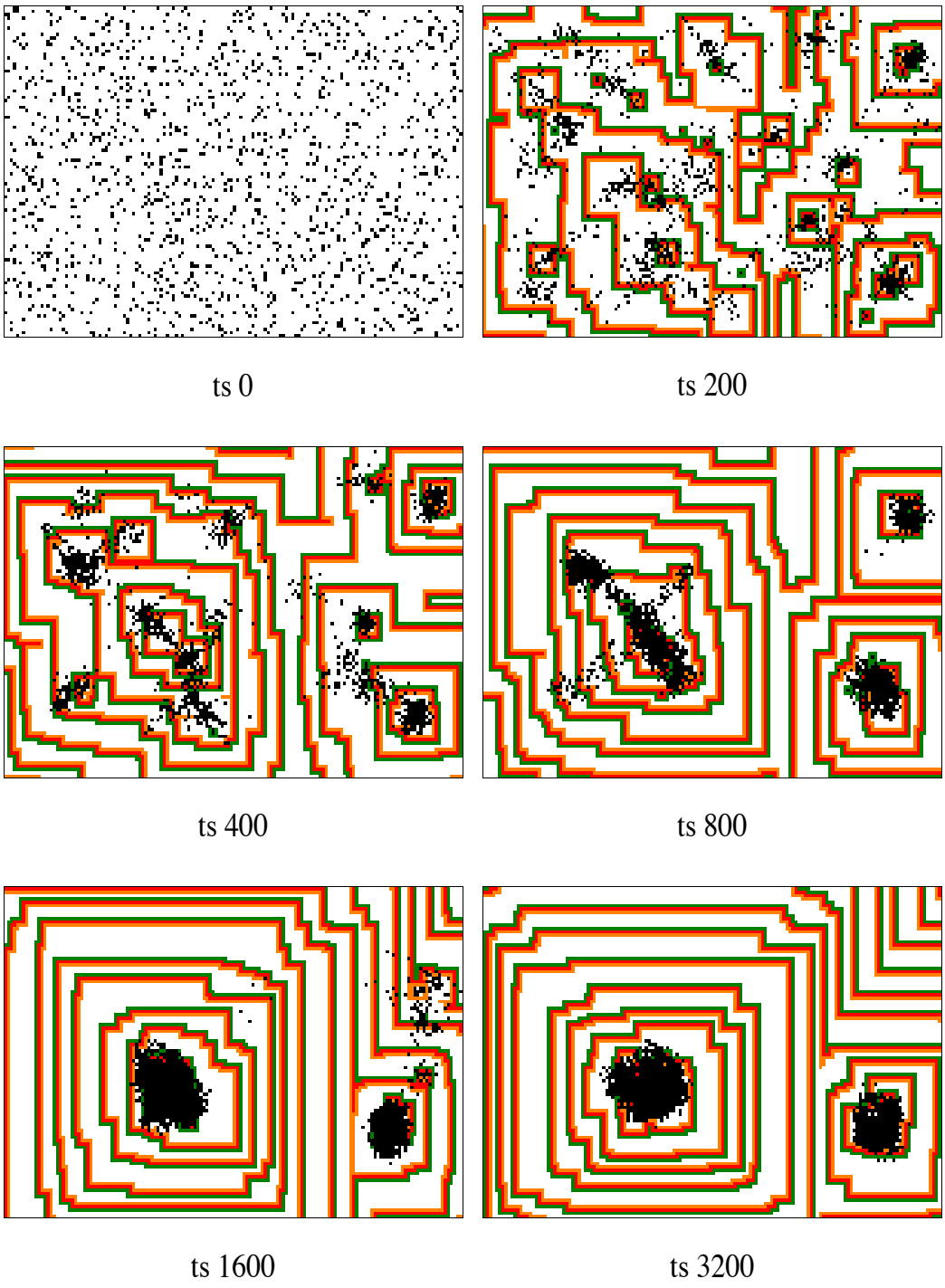


Figure 4.22: Example of the process of amoebae gathering through a simulation of reaction-diffusion and chemotaxis, and with a pure transmission rate and a low agitation rate of 10%. A 150 by 100 grid with was used with a starting grid of 1500 amoebae and a probability ratio setting of $(p_T, p_E, p_A) = (1, 0.01, 0.1)$. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as black cells and the white cells are neutral. The samples shown are, from the top left, time steps 0, 200, 400, 800, 1600, and 3200. This reflects the original work shown in Figure 4.23.

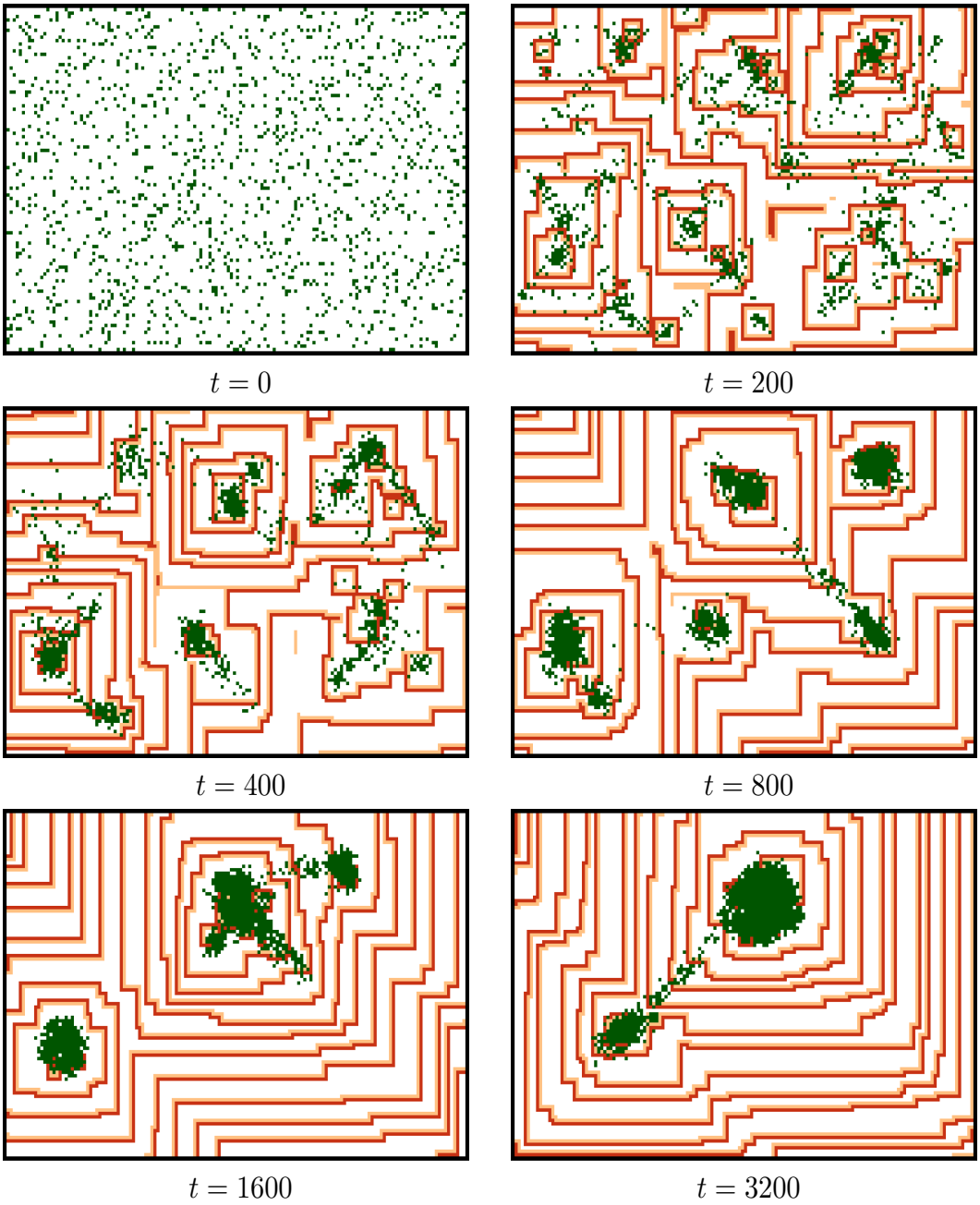


Figure 4.23: Process of amoeba gathering with perfect transmission and a low level agitation rate of 10% from the original work [Fatès et al., 2008, p.19]. A grid with $(X, Y) = (150, 100)$ is used with a probability ratio setting of $(p_T, p_E, p_A) = (1, 0.01, 0.1)$. A probability of 10% was used to initially populate a cell with an amoeba. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as green cells and the white cells are neutral. The samples shown are, from the top left, time steps 0, 200, 400, 800, 1600 and 3200.

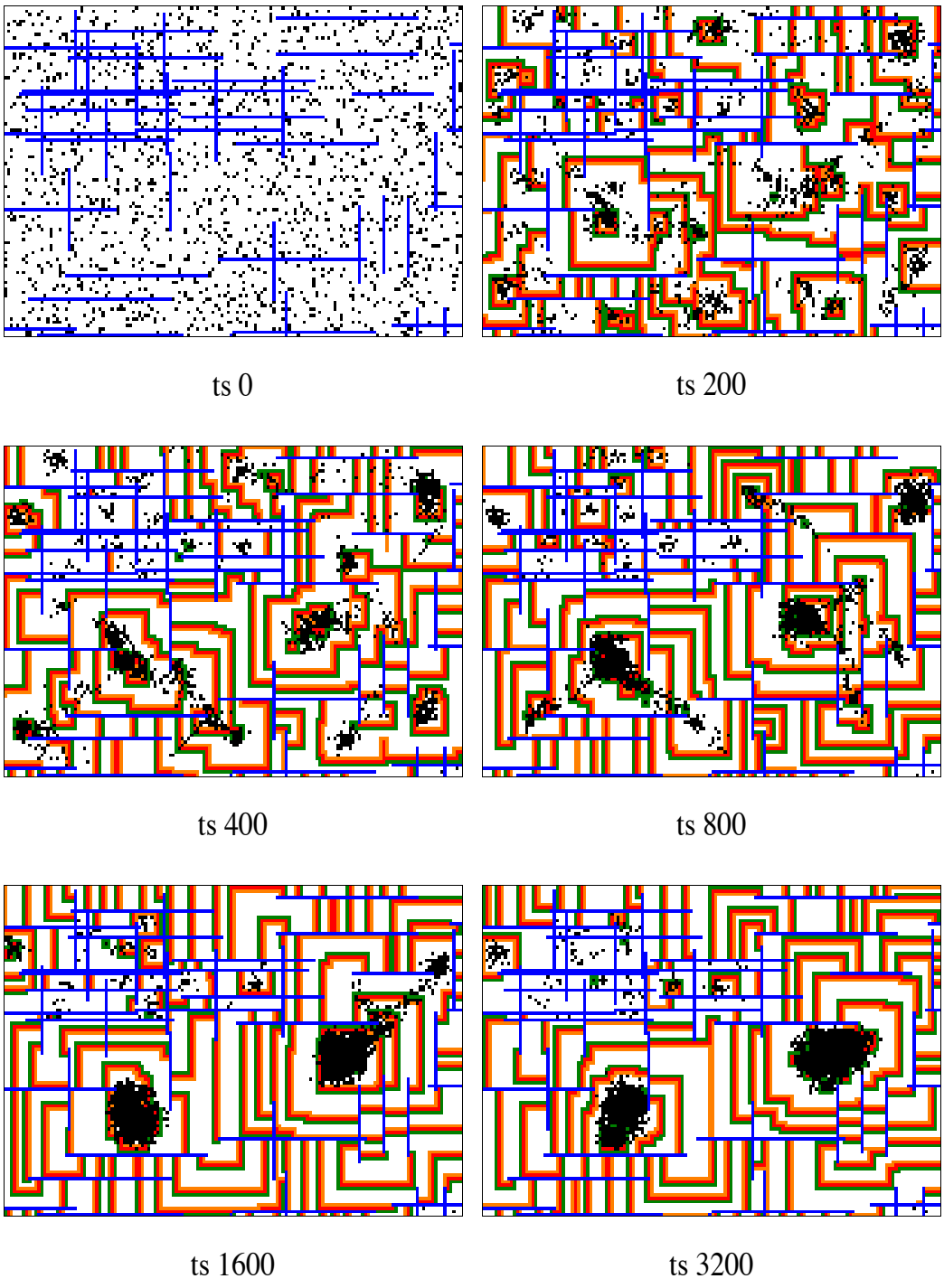


Figure 4.24: Example of the process of amoebae gathering through a simulation of reaction-diffusion and chemotaxis, and with obstacles on the grid. A 150 by 100 grid with was used with a starting grid of 1500 amoebae, 1500 blocked cells and a probability ratio setting of $(p_T, p_E, p_A) = (1, 0.01, 0.1)$. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as black cells and the white cells are neutral. The blue cells are the obstacles on the grid. The samples shown are, from the top left, time steps 0, 200, 400, 800, 1600, and 3200. This reflects the original work in Figure 4.25, although on a smaller grid.)

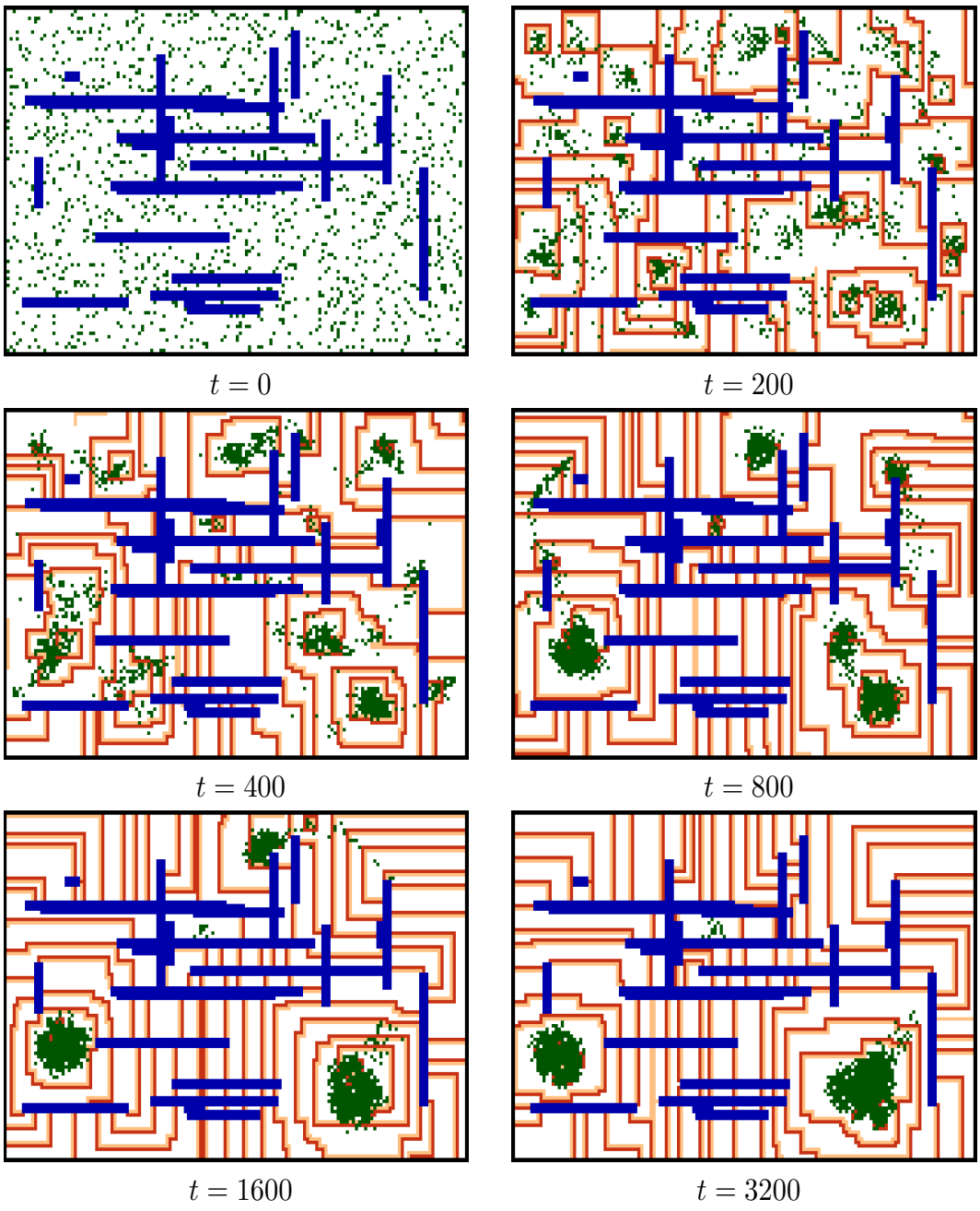


Figure 4.25: Process of amoeba gathering with a perfect transmission rate and a small agitation rate and obstacles from the original work [Fatès et al., 2008, p.19]. A grid with $(X, Y) = (150, 100)$ is used with a probability ratio setting of $(p_T, p_E, p_A) = (1, 0.01, 0.1)$. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as green cells and the white cells are neutral. The blue cells are the obstacles on the grid. The samples shown are, from the top left, time steps 0, 200, 400, 800, 1600, 3200.

In Figure 4.26, as in Figure 4.18, a smaller grid of 40 by 40 is used to show how the gathering process is well illustrated by a smaller grid. A pure transmission rate was used with no agitation: $(p_T, p_E, p_A) = (1, 0.01, 0)$. The grid was randomly populated with 600 amoeba and 222 blocked cells. The run time of a large 100 by 150 grid can be four hours or more, while a 40 by 40 takes considerably less and gives comparable results. Consequently, the thesis uses a 40 by 40 grid in the simulations carried out in chapter 5.

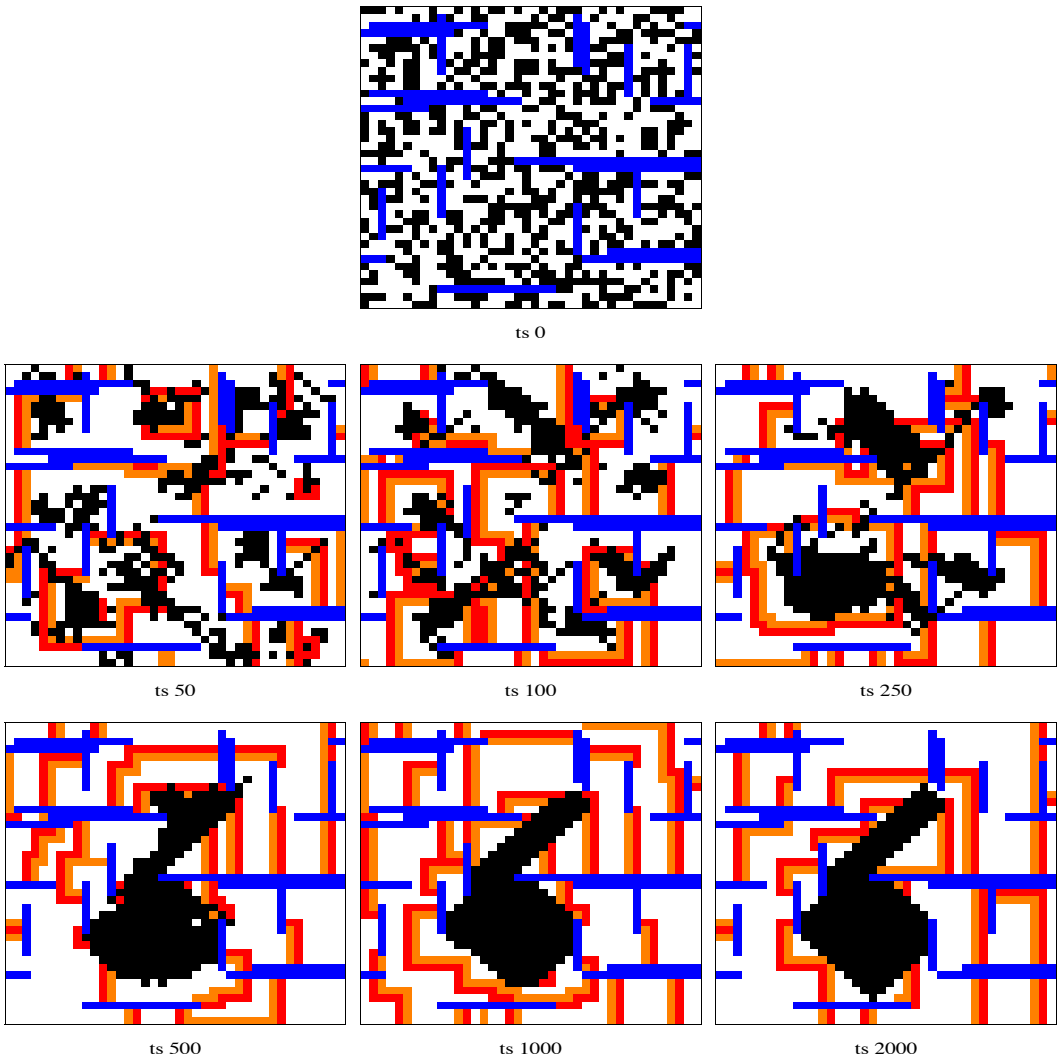


Figure 4.26: Example of the process of amoebae gathering through a simulation of reaction-diffusion and chemotaxis, and with obstacles on the grid. A 40 by 40 grid was used with a starting grid of 600 amoebae and a probability ratio setting of $(p_T, p_E, p_A) = (1, 0.01, 0)$. 222 blocked cells were arranged randomly in lines and bars on the grid. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as black cells and the white cells are neutral. The blue cells are the obstacles on the grid. The samples shown are, from the top left, time steps 0, 50, 100, 250, 500, 1000 and 2000.

4.6.6 Test plan

The reactive-diffusion and chemotaxis model, which is used as an example of a particle CA, gives a range of CA grid outputs through the altering of the probability rates (*see Table 4.2*). Two regimes are modelled and analysed, the extinction and the self-organising regimes. The extinction regime is interesting both in the general interpretation of the grid space where $0.2 < p_T < 1$ and in the randomness of the distribution of excited cells. The self-organising regime, as opposed to the extinction regime, features clustering. Consequently this model not only provides a series of outputs that fits in with the idea of measuring clusters and connectedness, but also provides a series of outputs that test the suitability of measuring results that do not exhibit obvious visual clusters. A series of tests covering both these regimes are analysed in chapter 5.

4.7 Randomised model

The second model involves the changing state of cells within a CA grid. It incorporates probability in the use of noise to 'control' the formation of patterns. This puts it in the new category of randomised CA type that was proposed in subsection 2.6.5 as an extension to the three proposed by Ermentrout and Edelstein-Keshet [1993].

The randomised model replicates the work in [Cohen et al., 2011]. This work simulates the role of structured noise and delta-notch signalling in self-organising patterns. The modelling of delta-notch signalling to help understand how cells communicate and develop, an area that is still not fully understood, is considered in subsection 3.5.2.

The underlying principle of delta-notch signalling is that cells compete to change to a delta state and once they have done that they inhibit their neighbouring cells from making the same change by activating their notch signalling. This results in a spaced pattern of delta, or active cells. This pattern is stable and static, but is set by the order of the asynchronous updating of the cells. The perturbation of this process with structured noise breaks up this stability, leading the cells to develop over time a much tighter patterning. Cohen et al. [2011] implement two types of structured noise, (a) temporal noise that occurs when each signal sending cell generates an inhibitory signal with an irregular strength and (b) spatial noise that originates from a spatial fluctuation of the strength of cell to cell signals.

4.7.1 Randomised model design

The model uses two types of asynchronous updating schemes outlined in subsection 2.6.3. The random order asynchronous (ROA) update is used in the simulations that do not include noise. This is also known as a random new sweep where all the cells are updated in one time step, but the order the cells are updated in is decided at the start of each time step. In their simulations with signal noise Cohen et al. [2011, p.792] define a single time step "as a number of random selections equal to the number of cells in the array". Consequently, a random selection asynchronous (RSA) updating is used when the model incorporates the probability of noise affecting the delta-notch process. In this method, also known as uniform choice, each time step updates the same number of cells that exist on the grid, but each individual update randomly selects which cell to update; thus at the end of the updating sequences for a time step a cell might have been updated once, multiple times, or not at all.

A cell is either active or neutral, the former signifies that it has changed to a delta signalling state, the latter that its notch signalling is active and it has been inhibited from changing state. In the output grids active cells are represented by the black cells. Non active cells are white although, as will be discussed below, in other simulations they are coloured to represent how many cells are active within the specified range. The number of active cells that can exist in a cell's neighbourhood is set by the threshold value. The scope of the neighbourhood is defined by the range, which can be one cell incorporating eight neighbours, or two with eighteen, or three with thirty-six neighbours (*see Figure 4.2(a)*). The grid properties differ from the other two models as it is hexagonal and has toroidal boundary conditions (*see Figure 4.2(b)*).

The following sections outline the rules, (based on the original work in [Cohen et al., 2011], governing the simulations of the model, the validation of the new simulation against the original work and the test focus adopted.

4.7.2 Lateral inhibition without noise

Although this can be defined with a probability, as the probability value used is set to one it is in fact the asynchronous update that makes this a randomised CA type. The state of each cell is determined by the state of its neighbours, with the threshold set to one. This means that only one cell can be active in a hexagonal based Moore neighbourhood of six cells. Thus the cells selected early on for update have a higher probability that none of their neighbours have already

been activated.

The rule governing a lateral inhibition without noise is:

Rule 1: A cell will become active and start transmitting an inhibitory signal if non of its neighbours are active ($\sigma_{v_c}^t < 1$):

$$\sigma_c^{t+1} = 1 \quad \text{if } card\{\sigma_{v_c}^t < 1\}, \quad \text{else } \sigma_c^{t+1} = 0 \tag{4.8}$$

As was mentioned above, this simulation establishes a fixed spatial pattern of delta signalling cells. The introduction of spatial and temporal noise into the cell update rules removes this static state and establishes a much more compact pattern. This rule equates with the probabilistic rule set in [Cohen et al., 2011, p.789, *figure 2(b)*].

4.7.3 Temporal noise

Temporal noise (N_t) has the ability to prevent a cell from becoming active, or to make an active cell inactive. In effect, a cell that should have its state changed to active or should remain active has a probability that it will stay or become inactive.

The rule governing the application of temporal noise is:

Rule 2: a cell will have an active state and transmit an inhibitory signal if the number of active cells in the neighbourhood of the cell are below the threshold limit and if a generated random number is greater than N_t value (H is the threshold value);

$$\sigma_c^{t+1} = 1 \quad \text{if } card\{\sigma_{v_c}^t < H\} \quad \text{and } p(N_t) = 0, \quad \text{else } \sigma_c^{t+1} = 0 \tag{4.9}$$

4.7.4 Spatial noise

Spatial noise (N_s) represents a lessening in the strength of the signal from neighbouring cells for the cell in focus to be inhibited. In the model it is used when the number of active cells in the neighbourhood are equal to or above the threshold value. In the case where the cell in focus would be inhibited from changing state, the spatial noise probability setting allows it a chance to ignore the signals and to become active. The rule is set so that as the number of active neighbours becomes greater than the threshold the chances of ignoring the state and becoming active lessens.

The rule is expressed as:

Rule 3: a cell with the number of active neighbours equal to or greater than the threshold setting can still become active and transmit an inhibitory signal if a randomly generated number is less than or equal to the N_s value to the power of the sum of 1 plus the number of active cells in the neighbourhood and less the threshold value;

$$\sigma_c^{t+1} = 1 \quad \text{if } card\{\sigma_{v_c}^t >= H\} \quad \text{and } p(N_s)^{[1+n-H]} = 1, \quad \text{else } \sigma_c^{t+1} = 0 \tag{4.10}$$

Rules 2 and 3 are run in conjunction based on number of active cells and the threshold setting. They both represent the rule table in [Cohen et al., 2011, p.790, *figure 3(a)*].

4.7.5 Simulation examples

Figure 4.27 shows four examples of the lateral inhibition model with no inhibition using Equation 4.8 and random order synchronous updating. The static spatial disorderly spread of active cells is produced from an initial grid of inactive cells after 1 or 2 time steps. This is in line with the original simulation shown in Figure 4.28.

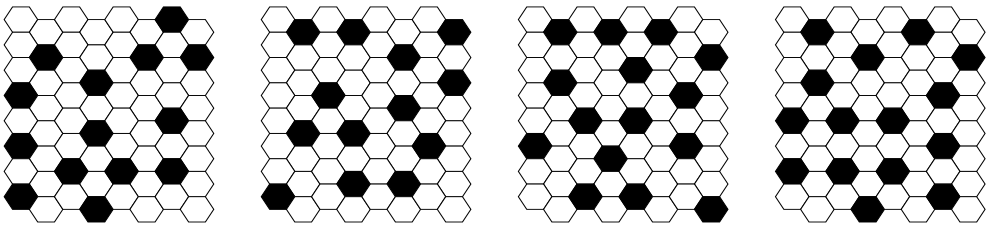


Figure 4.27: Four examples of lateral inhibition with no noise using an eight by eight hexagonal grid. The random order asynchronous updating method is used with rule 1. The active (black) cells quickly form into a disordered pattern that concurs with the original simulation shown in Figure 4.28.

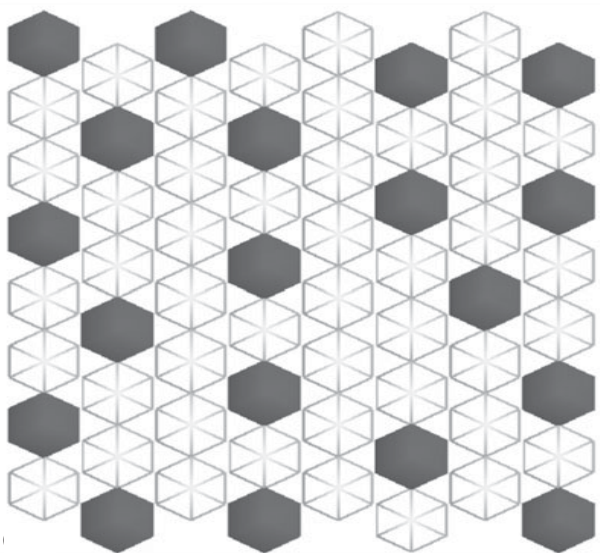


Figure 4.28: A CA simulation of lateral inhibition from [Cohen et al., 2011, p.789]. The cells in an 8 by 8 hexagonal grid are updated using random order asynchronous updating. The disorder of the active (dark grey) cells reflects the inhibitory signal.

The second simulation involves using first $N_s = 0.1, N_t = 0$ and then $N_s = 0, N_t = 0.1$, and using a threshold of $H = 1$. Figure 4.29 shows some of the output grids, which agree with the original results shown in Figure 4.30. The output grids shows the active cells as black; the inactive cells are colour coded to indicate how many active neighbours they have. The result is that the pattern of the active cells became more compact. This is reflected by the colour of the inactive cells gradually showing a higher rate of active neighbours. The final grids are more compact and the neutral cells mainly have three signalling cells around them, as shown by the increase in grey cells. There would seem to be a difference in the application of the toroidal boundary conditions, as the inactive cells on the border of the grids in the original results do not seem to reflect the active cells on the opposite sides of the grid, which in a 'wrap around' toroidal boundary condition should be taken into account.

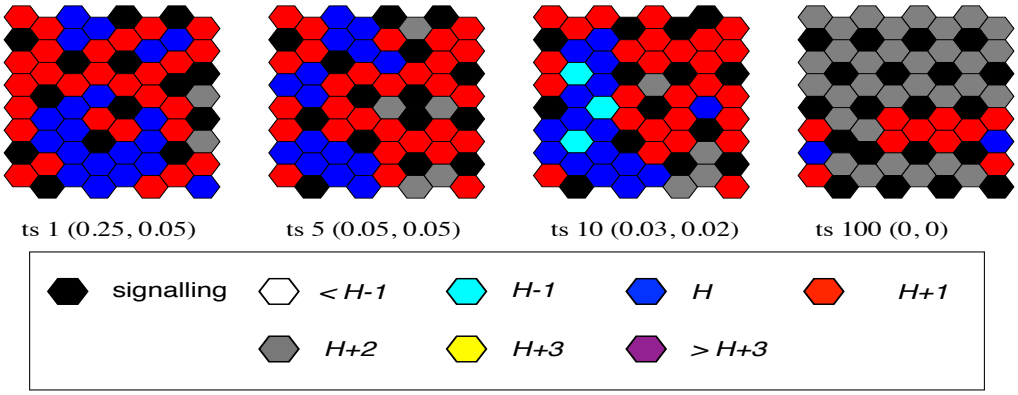


Figure 4.29: Lateral inhibition with spatial noise probability set to 0.1. The threshold is set to $H = 1$, the neighbourhood range is 1 and a toroidal boundary is used. Rather than settling into a static, disordered pattern, by the hundredth time step the spatial noise has perturbed the active cell into a more compact a pattern. This is also indicated by the increase in grey coloured cells signifying an increase in the number of signalling cells around the inactive cells. Time steps (ts) 1, 5, 10 and 100 are shown. The results concur with the original results displayed in Figure 4.30.

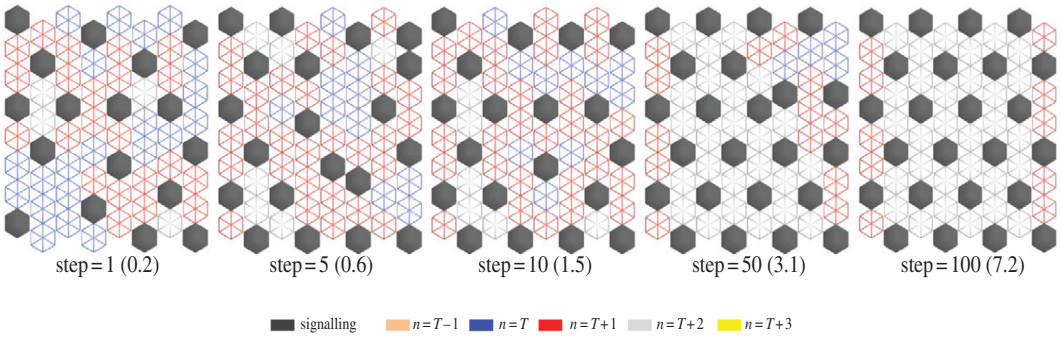


Figure 4.30: A CA simulation of signal noise leading to pattern optimisation from [Cohen et al., 2011, p.791]. The cells in an 8 by 8 hexagonal grid are updated using random selection asynchronous updating. The threshold is defined as T and is set to 1. The colour key is at the bottom of the figure, with signalling cells shown as dark grey and the colour cells indicating the number of active neighbours around an inactive cell. The number of grey cells indicating three signalling cells around an inactive cell increase as the pattern becomes more compact. A selection of time steps are shown, increasing from left to right.

Figure 4.31 shows the last output grid of a series of simulations of a twenty by twenty hexagonal grid over a thousand time steps. The simulations varied in the neighbourhood range, threshold, noise and asynchronous update types. The neighbourhood ranges of one, two and three were tested. Two setting were used for each range:

1. random selection asynchronous update with no noise ($N_t = 0, N_s = 0$) and
2. random selection asynchronous update with some temporal noise ($N_t = 0.01, N_s = 0$).

Simulations with a variety of threshold (H) settings were performed for each range, noise/updating combination, including where the threshold is set to the number of neighbours. Thus the first grouping, Figure 4.31(a), has six cells in a neighbour hood and the last grid shows the result of a simulation with the threshold set to six (H=6). The threshold settings of the last simulations displayed in Figure 4.31(b) and (c) also mirror their respective neighbourhoods of eighteen (H=18) and thirty-six (H=36).

The results show a similar range pattern formation shown in the original work (*see Figure 4.32*), which further validates the new model. In the simulations using a range of one, Figure 4.31(a), the addition of noise in the lower row shows how the signalling (black) cells have formed a compact pattern across the grid. The only exception is the last grid where the threshold is the same as the number of cells in a neighbourhood and the signalling (black) cells predominate; this is also seen in the results from the original work, Figure 4.32(a). In the other two groupings, Figure 4.31(b) and (c), a spotted pattern can be discerned in the first three threshold settings of each, with the lower rows showing how the addition of noise produces a more compact pattern. The next two thresholds settings in both (a) and (b) suggest a move towards a striped pattern, although in the lower row of (b) H=28 the noise has produce an increase in the number black cells such that the inactive, non-black cells form a spotted pattern. The last grid of both (b) and (c) reflects that of (a) where there is a prevalence of the signalling cells. This supports Cohen et al. [2011, p.794] conclusion that "[a]t different signal ranges and inhibitory thresholds, a broad scope of patterns can be achieved" and that noise plays an important part in the development of patterns and compressed self-organisation.

The new simulations in this section replicate those carried out in the original work. Consequently, the new model can be seen as validated against the original model outlined in [Cohen et al., 2011].

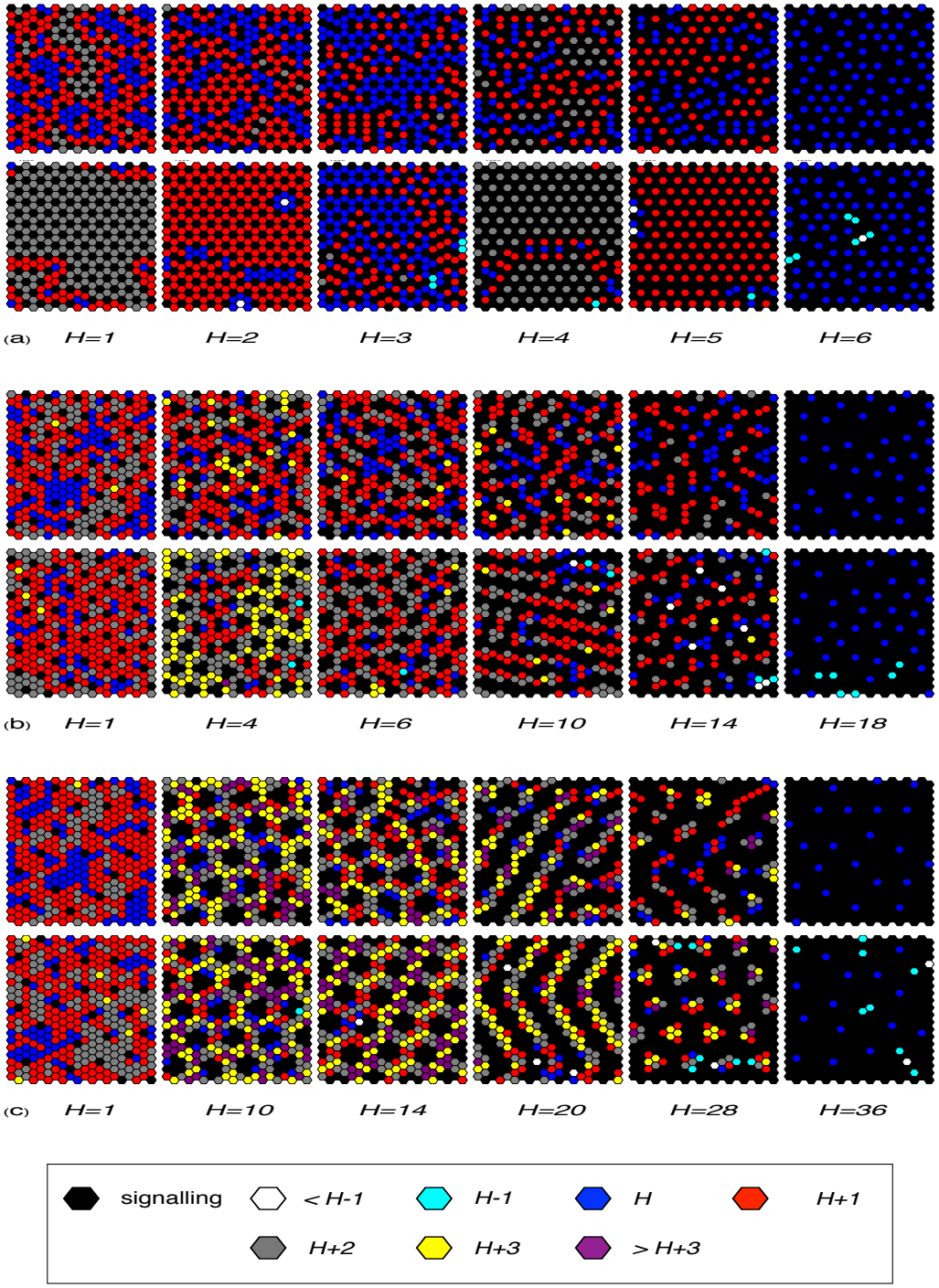


Figure 4.31: Pattern formation using lateral inhibition with different thresholds (H), using a 20 by 20 grid over 1000 time steps (last time step shown). (a) range=1; (b) range =2; and (c) range=3. The upper row in each grouping used a RSA update with no noise settings; the lower row used RSA with $N_t = 0.1, N_s = 0$. The two rows of each grouping corresponds to the output of the original work shown in Figure 4.32. As in the original, patterns of stripes and spots can be seen and noise can also appear to realign a disorder spread of signalling (black) cells into a pattern, such as in the lower row of (a) H=1, 2, 3, 4 and 5 (b) H=4, 6 and 10, (c) H=14, 20 and 28.

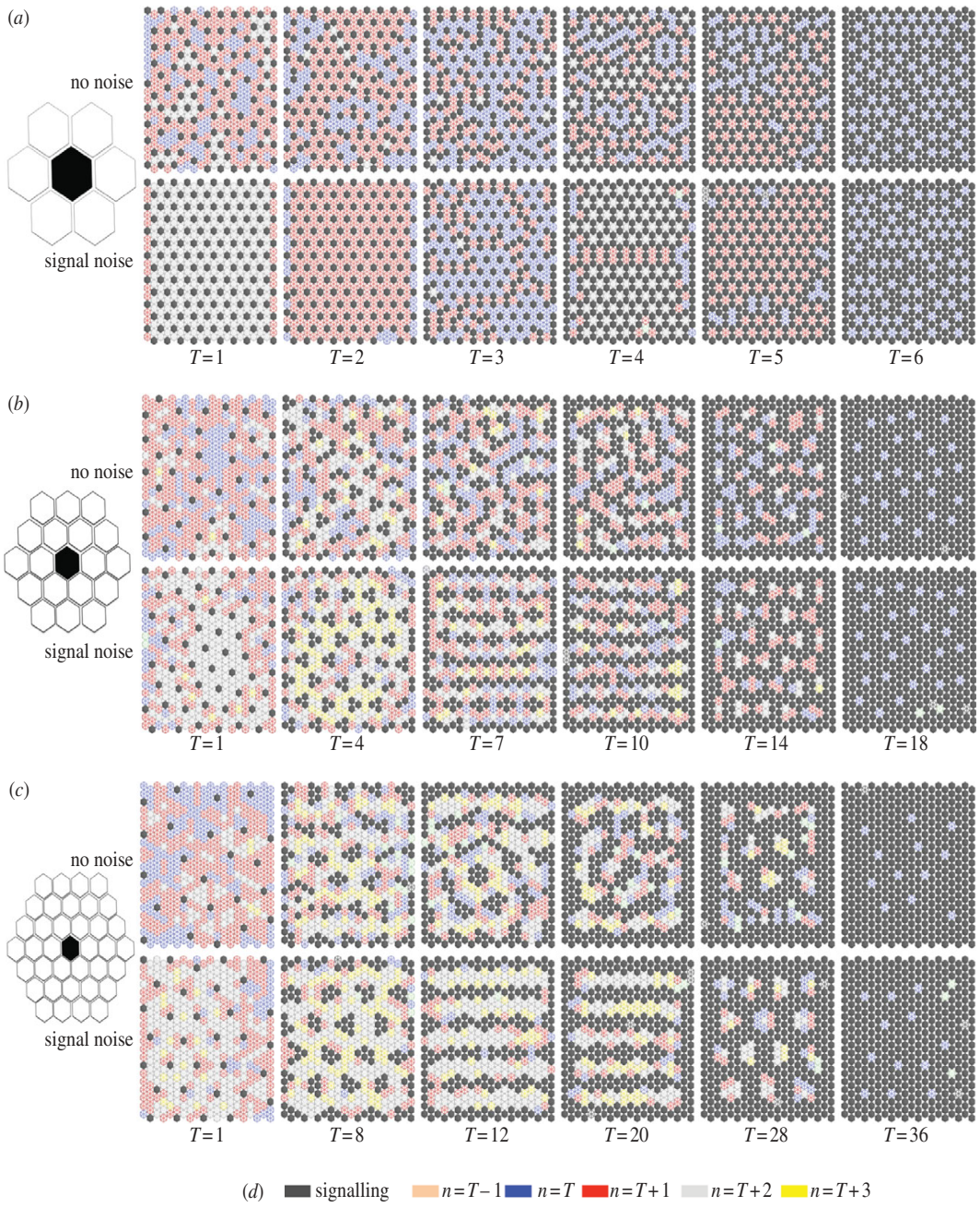


Figure 4.32: Pattern formation using different signal ranges and inhibitory thresholds from [Cohen et al., 2011, p.152]. Each group (a-c) uses a 20 by 20 hexagonal grid and shows the results after 1000 time steps. The signal range is shown by the hexagonal diagram to the left of each grouping; (a) one cell, (b) 2 cells and (b) a range of 3 cells. The top row of each grouping has no noise, while the lower row has a temporal noise setting of $N_t = 0.1$. A selection of threshold is shown, ranging from $T = 1$ to $T = 36$, are shown. The key at the bottom shows colour used to indicate an active signalling cell (dark grey) and the number of active cells surrounding an inactive cell. A range of patterns can be seen, as well as the effect of noise in realigning a pattern (e.g., lower row (b) T=10).

4.7.6 Test plan

The delta-notch model, which is used as an example of a randomised CA, provides a different challenge to the other models as the difference between the first populated grid and the final formation, based on the simulation examples above, is often minute. Tests are run and analysed in chapter 5 against the above simulations, but also against simulations using a variety of noise settings that cause a greater fluctuation of the number and placements of the signalling cells.

4.8 Deterministic model

The focus of the deterministic model is the resource usage of the network servers represented by the state of each cell, rather than the tracking of the Active Packets (AP) as they move across an Active Network (AN), delineated by the grid. Each AP can require processing at a server and has a resource requirement. It is this that creates a greater fluctuation in the resource usage across the grid than would be expected merely with the passage of data packets across a network. A brief outline of the model is given in *see subsection 3.5.3*. This section looks at the design of the deterministic model.

4.8.1 Deterministic model design

The Active Network model consists of Active Packets (APs) that traverse across a network of Active Nodes (ANode). For the sake of the model abstraction the network is configured to only include active nodes and there is no distinction over whether the Active Application is conveyed by the AP or already resides in the ANode. The attributes abstracted into the model are:

1. each ANode has resource capabilities;
2. each AP has resource requirements;
3. an AP can only be processed at an ANode if there are adequate resources available. The allocation is done on a first come first served basis;
4. an AP has the capability to reserve resources at an ANode for up to 50 time cycles from when it is processed;
5. an AP is set to perform one of the following actions at the end of each time cycle:
   (a) Forward - the AP is forwarded to the next node;

(b) Replicate - the AP is forwarded to the next node and if it was processed in the current time cycle two replica APs are created, one either side of the forwarded AP; each replication inherits the settings of the original AP;

(c) Merge - if there is sufficient processing resource, the AP is retained at the ANode for a period of five cycles. Any subsequent AP that is processed at the same ANode that is set with a post-processed status of 'merge', and has the same 'merge ID' is merged / consumed. At the end of the 5 cycles a non-consumed Merge AP is forwarded to the next node; and

(d) Consume - if the AP is processed, it is terminated at the end of the time cycle;

6. each AP has two direction indicators. The first is the key direction (1 = north, 2 = north-east, 3 = east .... 8 = north-west); the second flips between the two directions needed to achieve a diagonal move within the more restrictive Von Neumann environment, such that a first indicator of 2, would have a 2nd indicator that flipped between 1 and 3 - i.e. to achieve a diagonal move two lateral movements have to be made;

7. each AP has a lifespan; and

8. more than one AP can occupy an ANode during a time cycle.

4.8.2 Active Network Domain

The approach taken in this model is to retain the concept of the process status of the AN and its ANodes, but also to add the flow of the Active Data Packets (ADPs) across the AN. A key temporal difference is observed in the update rate of a network (a network cycle) and the processing of an ADP at an ANode. But the processing cannot be as a time scale separation process as the process time will vary and some ADPs will be flagged for passing to the next ANode during one network cycle, while others may take greater than one network cycle to process at the current ANode. This means that while the active processing of an ADP at an ANode may take longer than one network cycle, the processor is checked every cycle for any processed ADPs, hence the temporal overlap. The scale separation map (SSM) for the AN Domain show the ADP Flow and ANode occupying the same temporal scale, with the Processor stretching from the same scale to a larger one. The three kernels are shown in the AN Domain SSM, see figure Figure 4.33. Consequently, the model is a single domain one (*see appendices subsection A.1.3*),

with temporal overlap between the three sub-models / kernels and coupling occurring inside the inner iteration loop, (*see Figure 4.34 for the coupling template*).

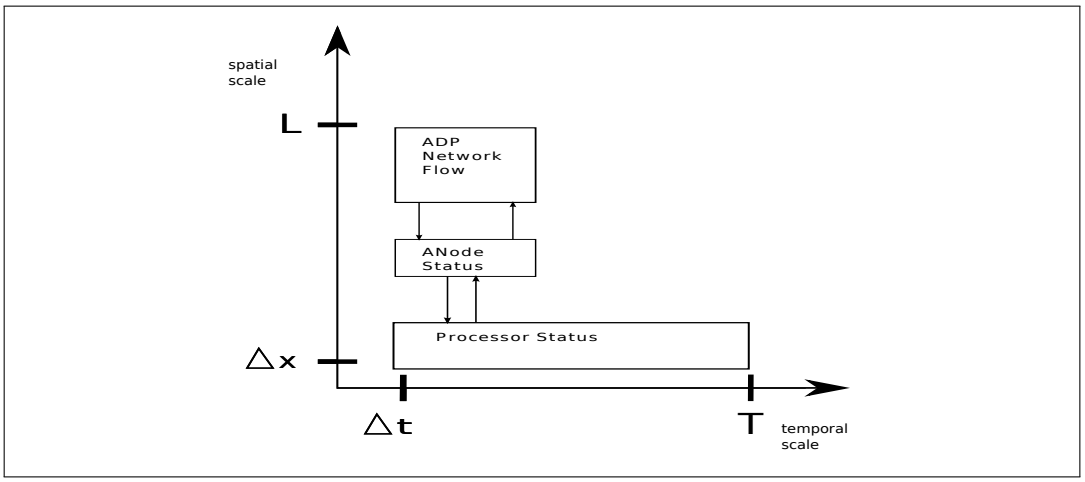


Figure 4.33: AN Domain SSM

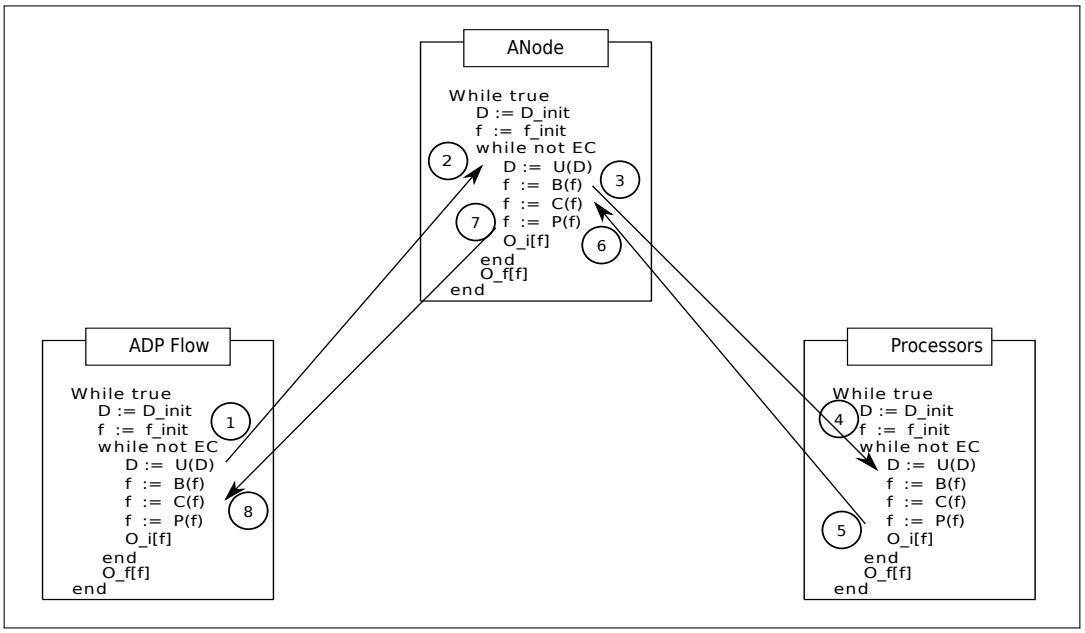


Figure 4.34: AN Domain coupling template

The processing sequence of the kernels (sub-model) is laid out in figure Figure 4.35. The basic functioning of the kernels is:

1. **ADP Flow kernel** - this models the flow of the ADPs across the network. Its first task is to update the ANode kernel with details of the movement

of any ADPs - in the event of none a message indicating this will be sent to resolve the blocking state around `receive`(). It will then receive details from the ANode kernal of all the ADPs that are ready to move from their current locations. The calculation of an ADPs movement is governed by:

(a) a message from the ANode kernel flagging which ADPs are ready to move;

(b) one or more of its four (van Neumann) neighbours having not already been visited by it, (in the event that all four have, then, excluding the last ANode visited, the ADP will treat the other three options as open, providing they satisfy the other rules);

(c) a space in at least one of the three nodes; and

(d) if there are more than 1, then either the ANode with the most resources available, or the route opposite to the previous ANode, or random choice.

If there are no routes open the ADP will remain where it is.

2. **ANode Kernel** - this models the overall status of the ANodes and the ADPs located at them over each network cycle. The ANode kernel receives a list of ADPs to move from one ANode (cell) to another. The number of ADPs that can be located at an ANode during one network cycle is set at initialisation. The main role of this sub-model is to 'process' the ADPs passed to it and then pass a list of ADPs ready to move back to the ADP Flow kernel. The outline of the processing is:

   (a) The ADPs are moved as requested in the list from the ADP Flow kernel; any unprocessed ones that match the processing type of the ANode are added to the list to be sent to the Processor;

   (b) The list is sent to the Processor Kernel;

   (c) The list of processed ADPs is received from the Processor Kernel; and

   (d) A list of ADPs ready to move is created based on:

      i. the ADP has been processed - either it will have just been returned by the Processor Kernel, or it is 'passing through' on the way to an 'end terminal'. If the current ANode is an end terminal it sets the ADP status flag accordingly and the ADP Flow Kernel will terminate the ADP;

      ii. the ADP is unprocessed, but the current ANode does not match its processing type; and

iii. the ADP has passed its Life Span - the status is flagged accordingly and the ADP Flow kernel terminates the ADP.

(e) The 'ready to move' list is sent to the ADP Flow Kernel - in the event of none a message indicating this will be sent to resolve the blocking state around `receive`().

3. **Processor Kernel** - this models the processing that takes place when an ADP activates code at the ANode (i.e. is unprocessed, has a matching processing type and there is processing capacity). The number of ADPs per ANode that can be processed at the Processor Kernel is the same as the number of ADPs that can be located at an ANode during a network cycle and is set at initialisation. The number of cycles required to process an ADP is initially set by the number of cycle requirements of the ADP and the processor scale factor of the ANode - so an ADP requiring 2 network cycles will have its processing time set to 2 by a 'standard' ANode, but to 1 by an ANode twice as fast (scale factor of 0.5) and to 4 by an ANode half as fast (scale factor of 2). This allows for *slow* or *fast* ANodes to be placed in the network and for ADPs to range from processor *light* to *heavy*. Each cycle within the Processor Kernel starts with receiving a list of ADPs to process from the ANode Kernel and ends with sending a list of processed ADPs back to the ANode Kernel - in the event of none a message indicating this will be sent to resolve the blocking state around `receive`(). The reduction of the time set by the Processor Kernel for an ADPs to be processed will normally be one per cycle, but an additional factor can be brought in by building a dependency on the collective amount of processing being undertaken by an ANode; this could be used to reflect an ADP becoming process bound.

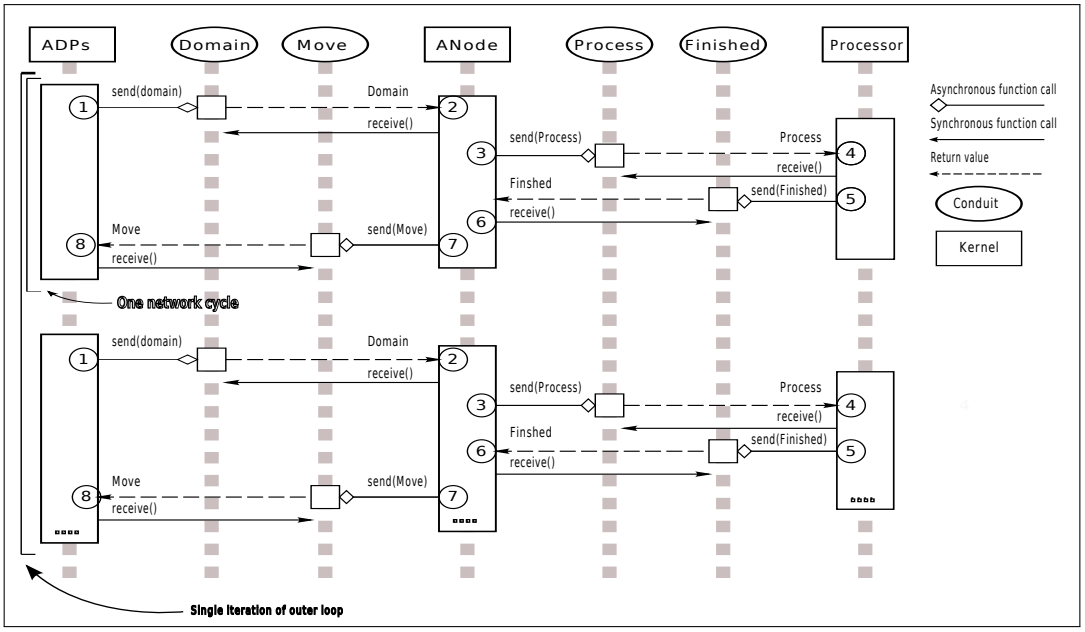


Figure 4.35: AN Domain Sequence Diagram

4.8.3 Attributes

The following is the outline of the attributes of an ADP, an ANode and a Processor:

- **ADP**
  - ID (*nnn*)
  - CurrentLocation (A1)
  - Status (**U***nprocessed* | **P***rocessed* | **D***ied* | **F***inished*)
  - ProcessType (x | y | z | ....)
  - LifeSpan (n network cycles)
  - VistedANodes (A1, B2, B3, C3...)
  - ProcessingCyclesRequired (n network cycles)
- **ANode**
  - ID (A1)
  - MaxADPs (0 - n)
  - ProcessTypes (x + y + z...)
  - Status (**A***ctive* | **D***ead*)
  - Type (**T***erminus* | **N***ormal*)
- **Processor**
  - ID (A1)

- ProcChannels ($n \equiv$ ANode.MaxADPs)
- ProcessorScaleFactor (0.5 | 1 | 1.5 | 2)

4.8.4 Message structure

There are four `send(data)` conduits; these are ①, ③, ⑤, and ⑦ in Figure 4.35 and Figure 4.34. Each one passes a list of ADPs and their relevant attributes.

① ADP $\Longrightarrow$ ANode:

$< MsgNum = 0 - n >< ID = ADP.ID, FROM = ADP.VistedANodes[last], TO = ADP.CurrentLocation, STATUS = ADP.Status[, PTYPE = ADP.ProcessType, PCR = ADP.ProcessingCyclesRequired] >< ID = ... >$

③ ANode $\Longrightarrow$ ADP:

$< MsgNum = 0 - n >< ID = ADP.ID, STATUS = ADP.Status >< ID = ... >$

⑤ ANode $\Longrightarrow$ Processor:

$< MsgNum = 0 - n >< ID = ADP.ID, PCR = ADP.ProcessingCyclesRequired >< ID = ... >$

⑦ Processor $\Longrightarrow$ ANode:

$< MsgNum = 0 - n >< ID = ADP.ID >< ID = ... >$

4.8.5 Initialisation and perturbations

Each kernel reads from a text file on initialisation. The following is set up: (a) the initial ADPs, with their start location and attributes; (b) the number of ADPs that can be at an ANode over one network cycle, and (c) the Processor Scale Factor for each ANode; (d) the number of processing channels per ANode; and (e) the type of ANode - i.e. is it an End Terminal, (these will be set up on the border of the grid), or a normal ANode. Perturbations are set up at initialisation with an attribute that determines after how many cycles they are 'introduced' into the environment. Perturbations are in the form of ADPs with a special status - e.g., an ADP with a Status of k will 'kill' an ANode (set ANode.Status to D).

4.8.6 CAs and observables

Each kernel is a sub-model and has a range of data. Some data can be extracted from more than 1 kernel, (such as the number of ADPs per ANode per network

cycle); only one source per data item is shown below. The following outlines the modelling method and available data at each kernel:

- ADP Flow Kernel - this uses a two dimensional grid. An ADP can move 1 cell to left, right, above, or below (von Neumann neighbourhood). The basis of the movement is on a non-repeating random walk [Chopard and Droz, 1998; Flake, 1999], with a maximum of n ADPs per cell. The data available include:
  - The movement of an ADP is recorded[2] and can be mapped individually, or collectively (e.g. all, or just ADPs of one Process Type);
  - The number of cycles / moves an ADP makes before it finishes or is terminated;
  - The number / percentage of ADPs finished and terminated is recorded;
  - The time taken to process an ADP is recorded - both finding a suitable ANode, at the Processor, and reaching an end terminal; and
  - The number of messages sent to the ANode Kernel each network cycle.
- ANode Kernel - this uses a 2 dimensional grid with each cell representing an ANode. This kernel is essentially a control / monitoring point. But it does record information that can be displayed and analysed including:
  - The state of each ANode in terms of number of ADPs located at it per network cycle;
  - The number of messages sent to the Processor Kernel per network cycle; and
  - The number of messages sent to the ADP Flow Kernel per network cycle.
- Processor Kernel - this model uses an n column one dimensional CA grid to represent the processing channels available at an ANode ($\equiv$ maximum ADPs per ANode). The updating of the grid depends on the value set in the cell, the values in the other cells (such as in a Margolus neighbourhood, but in a one dimensional environment), and the setting of the ProcessorScaleFactor. The data that can be collected from this kernel includes:
  - The data from the processing channels of each ANode; these can be amalgamated to then be represented as a single value per ANode;
  - the number of messages sent to the ANode Kernel; and

[2] 'recorded' also covers data that are extractable from the data logged

– the number of messages sent to the ADP Flow Kernel Kernel.

The data are logged per cycle and at the termination of a kernel. The intention is to capture the data so that visualisation of the sub-models changing states can be recreated and data extracted for analysis as required.

4.8.7 Simulation examples

The AN model, which is used as an example of a deterministic CA, was developed following the core criteria of a Petri net simulation of the properties of "a high-level abstract definition of an Active Network" [de Silva, 2004, p.123]. The model looked at the memory performance of the nodes (*servers*) as agents (*packages*) are replicated across the grid (*network*), using the Hurst parameter to analyse the state of the nodes over 500 time steps. The results showed evidence of a "Cascading Effect" when the network exhibited "high levels of Self-Similarity (i.e. with Hurst values above 0.9). [...] The threshold value of 0.9 was based on the empirical evaluations of several predetermined simulation scenarios" [de Silva, 2004, p.118]. The results were generated by injecting a mixture of replicating and non-replicating packages into the grid; if a node could process the replicating package it would replicate, otherwise it would move onto the next node in search of the required resources. The packages would move across the grid until they entered an end node and exited the grid.

The Hurst value was developed to assess the optimum dam size along the Nile delta to cater for the extreme conditions of drought and flooding experienced there. As such it relates to the analysis of time series and any autocorrelations that can be found in them and any resulting long-term dependencies. The Hurst value is used to indicate whether the next value is likely to be lower or higher or whether there is a developing pattern of fluctuating low / high values, or a sustained period of high values, or whether there is no evident autocorrelation. This makes it a very linear evaluation, that requires something to measure; the Hurst calculation does not like zero values, so a nominal value needs to be fed in, such as if a node has not had any of its memory utilised during a time step. The Hurst exponent is calculated on the memory usage of each node over the time steps of the simulation. This means that any observed graphical characteristic is subject to the order in which the Hurst exponent of each node is displayed. The order is critical; change it and a cascading or oscillating pattern could change to one showing evidence of two plateaued collections of node values. This can be mitigated by keeping the same order for each display, which is the case in the original work. But it highlights a potential problem with how output from abstract models are produced and

assessed. This is something this research cannot claim immunity from, but that it has attempted to address by measuring the state of the output grid across the time steps, rather than the state of individual cells.

In this way it is hard to verify the workings of this model against the former, as the focus is on the state of the grid across the time steps, rather than individual nodes. A suggestion of the "Cascading Effect" found in the previous work [de Silva, 2004, pp.122-147] was observed but it was far from conclusive (*see Figure 4.36*).

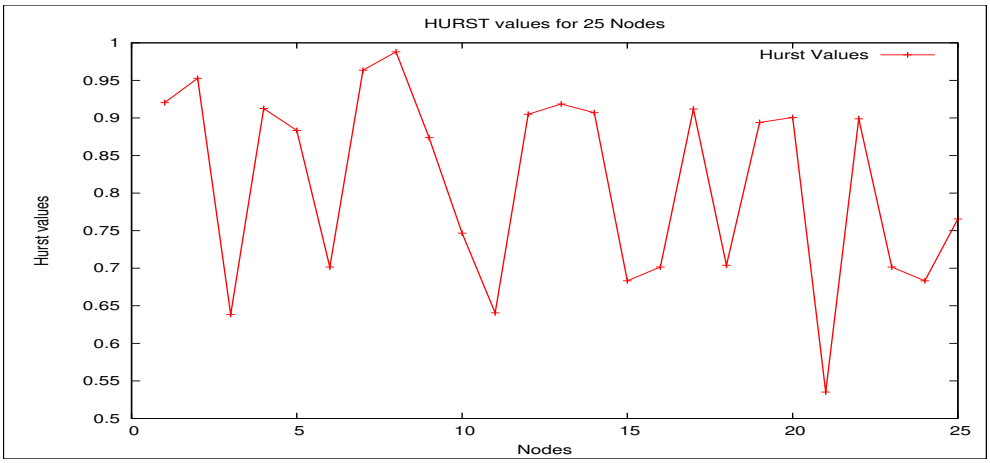


Figure 4.36: A measure of 25 nodes over 1024 time steps using the Hurst parameter

Also, there is no direct comparison with the actual working or performance of an active network that either model can be measured against. The role of this model in comparison with the scenarios and models outlined above will be discussed in chapter 6.

4.8.8 Test plan

The tests focus, as did the previous work, on the memory usage at the nodes. This is observed by utilising the replication of APs as they move across the grid.

4.9 Summary

This chapter described the scenarios and models used in the thesis. The hand-crafted scenarios verified the ability of the code written to identify and extract groupings of connected active cells. These scenarios also illustrated the effect of changing the neighbourhood used when searching for groups of active cells. The explanation of the different neighbourhoods, Manhattan range and boundary conditions indicates how they can impact the analysis of the output grids of a simulation. The details of two other scenarios, the probability and dynamic scenarios, were also outlined, although their usage is shown in chapter 5.

The model first investigated and then built was based on the work in [de Silva, 2004]. The decision to use the MUSCLE framework was founded in an initial desire to explore the potential of the model from both a temporal and spatial perspective. This meant that the first model was coded in Java on top of the MUSCLE code base. The control and freedom that coding gave had a greater appeal than utilising an existing tool, such as Matlab. Consequently the other two models were coded, although the advantage of using a different language was explored and Perl was selected. This interpreted language provided, with its Moose object oriented (OO) framework, a very versatile and extremely powerful environment for handling and manipulating data. One of Perl's strength in manipulating text is its ability to integrate with the tools available within the Unix / Linux core. This was especially important as all data stored and transferred between programs were in text format, which allowed the ability to rerun simulations, the metrics and the analysis. It also enabled the selection of specific records for processing, giving obvious advantages when wanting to extract key time steps from a resource intensive simulation. Although the deterministic model was programmed in Java, the text based output was easily converted into the required layout for measuring and analysing by the Perl programs that were written later.

The details of the three models used in the thesis showed the range of domains and modelling approaches used. The particle model presented an example of the representation of a real life biological phenomenon being used to solve a non biological problem. The modelling of slime mould *Dictyostelium discoideum* gathering in mounds in the event of food scarcity as a way of addressing the decentralised gathering problem also illuminated the effect the updating scheme can have. The required reaction-diffusion waves were only evident when a synchronous updating system was used. So the use of a scheme that could be seen as contrary to what happens in real life was necessary to represent a real life phenomenon. The new

model replicated the rules used in Fatès et al. [2008] and the simulations of the new model were successfully validated against the original work.

The randomised model also involved the representation of a biological system. But instead of using it as an analogous system to explain how a different system could work, it was a CA model of how the delta-notch signalling and 'noise' could form patterns, such as seen in skin pigmentation and the distribution of hair follicles. The model used two types of asynchronous updating schemes. The first updated each cell on the grid in a sequential order randomly set before each time step. But the modelling incorporating noise and producing patterns involved a scheme that is seen by some as more representative of real life, (*see subsection 2.6.3*). This system updated in each time step the same number of cells that existed on the grid, but they were randomly selected with no guarantee whether a cell would be updated once, many times or not at all during a time step. The new randomised model recreated the work done in [Cohen et al., 2011] and the simulations run were successfully validated against the original work.

The deterministic model was an attempt to represent a theoretical system. This, as well as the choice in the original work in [de Silva, 2004] of the Hurst parameter to measure the output of the simulation, meant that of all the models used this was the most difficult to replicate and validate. The problems of modelling a theoretical model is considered later in the thesis.

Chapter 5

Analysis

5.1 Introduction

This chapter presents the key results of the simulations carried out with the various scenarios and models outlined in chapter 4. Special attention is given to the performance of the new connectedness metric proposed, the C-Value (*see subsection 3.3.5*), across three CA types, including the randomised category proposed as an extension to those outlined by Ermentrout and Edelstein-Keshet [1993]. It should be noted that the graphs displayed in this chapter and appendix B are self scaling, based on the values being displayed. This allows as clear a display of the data presented as possible, but it does mean the scaling can vary.

The first section, section 5.2, looks at the scenario tests run to bench mark the performance of the selected metrics. The probability scenarios in subsection 5.2.1 create a sequence of grids with different percentages of randomly dispersed active cells. The grids were measured with both a Moore and a von Neumann neighbourhood. The established metrics, (entropy, BBR and mean density), perform as expected, with the BBR reflecting how the random placement of the active cells stretches across the whole grid. The entropy metric shows how the increase in the number of occupied cells produce a higher number of clusters with diverse sizes until a critical point is reached and a single cluster becomes dominant and spans the grid. The new C-Value keeps abreast of the mean density, indicating that the increase in the level of connectedness is down to compression induced by the increase in the number of active cells on the grid, rather than any actual non-random ordering of the cells.

The dynamic scenarios in subsection 5.2.2 simulate the gathering of the active agents on a grid into a centralised cluster. This is used to evaluate how well the

metrics indicated the gathering process of only a few active agents, as well as for a more densely populated grid. The C-Value successfully shows the gathering process, providing information on the states of the grids for all the tests; although the difficulty for all the metrics in registering changes for only a few active agents is clearly seen. A more extensive comparison of the two neighbourhoods is conducted, along with the three range settings of 1, 2 and 3. A very marginal benefit is discernible when using the Moore neighbourhood and a range of 3 with a low number of active agents. Consequently, this neighbourhood and range setting was adopted as the default for the subsequent simulations.

The probability and dynamic scenarios are then combined in subsection 5.2.3, with the initial grids for a dynamic scenario simulation being set up by the probability scenario. This allows a controlled range of populated grids to be tested, especially where the grids are filled up to 90% with active cells; identifying changes on such grids poses comparable challenges to those of sparsely populated grids. The C-Value provides useful information about the changing state of the grid as the active agents gathered together across all probability scenarios used as initial input; whereas the other metrics register less and less information as the initial grid of the simulation becomes more crowded with active agents.

In section 5.3 the reactive-diffusion and chemotaxis model is used to simulate a decentralised gathering of active agents on a grid. This represented the category of particle model, as defined in [Ermentrout and Edelstein-Keshet, 1993], (*see subsection 2.6.5*). Two tests, as outlined in the original work [Fatès et al., 2008], were run, (a) the extinction regime were there are no active agents, instead a setting of $(p_E, p_A) = (0,\ 0)$ and $p_T = \{0.1,\ 0.2,\ ....,\ 0.9,\ 1\}$ creates a series of simulations consisting of excited and refractory cells where the random placement of the excited cells are measured; and (b) the gathering regime where the decentralised gathering of the amoeba on the grid space is captured and analysed. The extinction regime ties in with the probability scenario as the distribution of the excited cells is random. The similarity is shown by the C-Value keeping abreast of the mean density, confirming that no gathering or ordering of the excited cells has taken place. The gathering regime is tested with a series of simulations with and without barriers on the grid, and with sparely and densely filled grids. As with the dynamic scenario, the C-Value gives a better indication of the gathering process than the other metrics.

The second category of CA model tested was the randomised model that was newly proposed in subsection 2.6.5. This recreates the original work in [Cohen et al.,

2011] that simulates a delta-notch model. There are two core tests in section 5.4, one without any noise perturbation and one with. The simulation without any noise creates a static array of signalling cells. The C-Value shows that while the connections of the signalling cells could be higher, they were far from disorderly. The test that incorporated noise uses two settings; the first used a spatial noise setting of $N_s = 0.1$, the second of $N_s = 0.9$. The former results in a distinctive pattern of forty clusters, each made up of three signalling cells; something that the C-Value does not identified. The latter produces a large cluster, containing virtually all of the signalling cells, spread without any discernible pattern across the grid. In this simulation the C-Value indicates that there was a level of connectedness between the signalling cell, but as it keeps abreast of the mean density it also implies that there is a certain level of randomness to the structure.

The two simulation using noise are then re-analysed in subsection 5.4.1 with a localised C-Value being calculated. This involves the averaging of the C-Value of each cluster found on the grid. This successfully identifies the ordered pattern that emerged with $N_s = 0.1$. Although it shows the same general behaviour of the C-Value in the simulation using $N_s = 0.9$, the LC-Values reveals that the calculation also needs to take into account the imbalance that a few small clusters or singletons can have when averaged with a single cluster holding almost all of the signalling cells.

The final simulation in section 5.5 involves a reworking of the Active Network used in [de Silva, 2004]. This simulation of a theoretical environment proves to be the least productive of all the tests. The main main benefit from the one test run is the example of how a computational model can produce an interesting artefact, where the oscillating cycle of two grids was observed.

The chapter closes with a summary in section 5.6.

5.2 Scenarios

A series of tests were run to provide an overall bench mark for the performance of the metrics against grids with different percentages of cell activation, (probability scenarios, *see subsection 4.4.2*), and against output grids showing the gathering of agents together over a series of time steps, (dynamic scenarios, *see subsection 4.4.3*). The two scenarios were then combined with the probability scenario providing a range of initial grids with different levels of active cell occupation for use in the dynamic scenario simulation.

5.2.1 Probability scenarios

The purpose of this test was to see how the four metrics performed when measuring grids with different numbers of active cells randomly located on them. The test created forty grids of size 40 by 40 for each probability value range of *p_value*={0.1, 0.125, 0.150, 0.175, 0.2,...., 0.875, 0.9}. The program creating the grid iterates through the cells and at each one compares a program generated random number against the *p_value*; if the random number is less than or equal to the *p_value* then the cell is flagged as active / occupied. Thus, the number of active cells on a grid is set by the probability value, with 0.1 correlating with 10%, 0.125 with 12.5% and up to 0.9 where on average 90% of the cells on the grid will be active. A sample of the type of grids created can be seen in Figure 5.1. Each grid was measured and the results then averaged for the grids of each probability value. The tables listing the averages for a Moore neighbourhood are listed in Table 5.1, and for a von Neumann neighbourhood in Table 5.2. The performance of the metrics is shown in the graphs in Figure 5.2.

The overall shapes of the graphs for the two neighbourhood are similar, although the apex of the curve of the entropy metric, for example, is achieved using a Moore neighbourhood on the grid with a probability value of 0.3 with a value of 2.7147 (*see Table 5.1*); whereas the entropy metric using a von Neumann neighbourhood peaks on the grid with a probability value of 0.475 with a value of 2.9205 (*see Table 5.2*). This difference reflects how the Moore's eight cell neighbourhood groups the active cells into fewer cluster more quickly than the von Neumann four cell neighbourhood. The entropy value reflects the reduction of the number of different sized clusters as the number of active cells increases. It has little it can say about the general state of the output grid once it has peaked. The mean density and BBR metrics do not take into account the type of neighbourhood used, so they have the same value in both tables. The BBR produces almost a horizontal line in the graph (*see Figure 5.2*), with all the grids, except the one with the lowest probability, having a value of 0.9506. This reflects how the randomly placed active cells are distribute across the whole grid, including to the outer edges (*see Figure 5.1*).

The new C-Value metric keeps in step with the mean density. This can be seen as reflecting how the gradual rise in the number of agents on the grid is also reflected in an inevitable increase in the compactness, and thus the connectedness of the agents within the fixed dimensions of the grid. What is important to note is that the cells have been randomly placed on the grids. Therefore, the matching of

the C-Value with the mean density has potential significance. The connection between randomness, mean density and the C-Value will be discussed further in the analysis of the results from the tests in subsection 5.2.2 and subsection 5.2.3, which also start with randomly generated grids, but then simulate the gathering of the active cells into a single cluster.

Table 5.1: Probability test using a Moore neighbourhood. Values from analysis of probability scenarios *p_value*={0.1, 0.125, 0.150, 0.175, 0.2, ...,0.875, 0.9}, where 0.1 correlates with a 10% chance that a cell will be active and 0.875 with an 87.5% chance. Forty grids of 40 by 40 size were created for each probability value and then measured using the four metrics. The metric of each group of probability values were then averaged. The ID is made up of the probability value used to set the number of active cells, and then a three character block representing the value indicating an active cell, the neighbourhood type used in any cluster search and the maximum range depth used. Thus, in this table 1M3 means the cell value used was 1, the neighbourhood was a Moore (M) one and the range depth was 3. The entropy metric peaks at a probability setting of 0.3. The BBR is static apart from the lowest probability setting. After a very slight gap at the start, the C-Value and mean density keep in step with each other.

ID	**C-Value**	**Entropy**	**BBR**	**MeanDensity**
0.1 1M3	0.1686	1.3531	0.9482	0.1000
0.125 1M3	0.1827	1.5405	0.9506	0.1250
0.150 1M3	0.1981	1.7581	0.9506	0.1488
0.175 1M3	0.2133	1.9619	0.9506	0.1744
0.2 1M3	0.2289	2.1614	0.9506	0.1991
0.225 1M3	0.2499	2.3694	0.9506	0.2245
0.250 1M3	0.2683	2.5065	0.9506	0.2497
0.275 1M3	0.2911	2.6397	0.9506	0.2758
0.3 1M3	0.3099	2.7147	0.9506	0.2944
0.325 1M3	0.3331	2.6668	0.9506	0.3218
0.350 1M3	0.3632	2.4713	0.9506	0.3528
0.375 1M3	0.3854	2.1585	0.9506	0.3755
0.4 1M3	0.4048	1.6819	0.9506	0.3981
0.425 1M3	0.4326	1.2622	0.9506	0.4232
0.450 1M3	0.4574	0.7180	0.9506	0.4511
0.475 1M3	0.4816	0.3611	0.9506	0.4740
0.50 1M3	0.5073	0.2161	0.9506	0.5014
0.525 1M3	0.5312	0.1552	0.9506	0.5254
0.550 1M3	0.5561	0.0728	0.9506	0.5493

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
0.575 1M3	0.5773	0.0478	0.9506	0.5731
0.6 1M3	0.6026	0.0316	0.9506	0.5985
0.625 1M3	0.6297	0.0185	0.9506	0.6252
0.650 1M3	0.6491	0.0094	0.9506	0.6465
0.675 1M3	0.6750	0.0079	0.9506	0.6717
0.7 1M3	0.7036	0.0078	0.9506	0.7007
0.725 1M3	0.7296	0.0026	0.9506	0.7275
0.750 1M3	0.7524	0.0017	0.9506	0.7506
0.775 1M3	0.7800	0.0003	0.9506	0.7769
0.8 1M3	0.7990	0.0000	0.9506	0.7986
0.825 1M3	0.8271	0.0000	0.9506	0.8259
0.850 1M3	0.8531	0.0000	0.9506	0.8525
0.875 1M3	0.8740	0.0000	0.9506	0.8737
0.9 1M3	0.8961	0.0000	0.9506	0.8970

Table 5.2: Probability test using a von Neumann neighbourhood. Values from analysis of probability scenarios p_value={0.100, 0.125, 0.150, 0.175, 0.200, ...,0.875, 0.900}, where 0.1 correlates with a 10% chance that a cell will be active and 0.875 with an 87.5% chance. Forty grids of 40 by 40 size were created for each probability value and then measured using the four metrics. The metric of each group of probability values were then averaged. The ID is made up of the probability value used to set the number of active cells, and then a three character block representing the value indicating an active cell, the neighbourhood type used in any cluster search and the maximum range depth used. Thus, in this table 1V3 means the cell value used was 1, the neighbourhood was a von Neumann (V) one and the range depth was 3. The entropy metric peaks at a probability setting of 0.475. The BBR is static apart from the lowest probability setting. After a very slight gap at the start, the C-Value and mean density keep in step with each other.

ID	C-Value	Entropy	BBR	MeanDensity
0.1 1V3	0.1842	0.7997	0.9482	0.1000
0.125 1V3	0.2062	0.9642	0.9506	0.1250
0.150 1V3	0.2286	1.1461	0.9506	0.1488
0.175 1V3	0.2502	1.2698	0.9506	0.1744
0.2 1V3	0.2650	1.4272	0.9506	0.1991
0.225 1V3	0.2858	1.6098	0.9506	0.2245
0.250 1V3	0.3040	1.7840	0.9506	0.2497

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
0.275 1V3	0.3244	1.9465	0.9506	0.2758
0.3 1V3	0.3400	2.1090	0.9506	0.2944
0.325 1V3	0.3575	2.2599	0.9506	0.3218
0.350 1V3	0.3825	2.4562	0.9506	0.3528
0.375 1V3	0.4036	2.5939	0.9506	0.3755
0.4 1V3	0.4203	2.7284	0.9506	0.3981
0.425 1V3	0.4443	2.8079	0.9506	0.4232
0.450 1V3	0.4689	2.8938	0.9506	0.4511
0.475 1V3	0.4860	2.9205	0.9506	0.4740
0.50 1V3	0.5120	2.9133	0.9506	0.5014
0.525 1V3	0.5360	2.7458	0.9506	0.5254
0.550 1V3	0.5594	2.4989	0.9506	0.5493
0.575 1V3	0.5796	2.0029	0.9506	0.5731
0.6 1V3	0.6039	1.5459	0.9506	0.5985
0.625 1V3	0.6314	1.1017	0.9506	0.6252
0.650 1V3	0.6510	0.6243	0.9506	0.6465
0.675 1V3	0.6755	0.2977	0.9506	0.6717
0.7 1V3	0.7043	0.1708	0.9506	0.7007
0.725 1V3	0.7296	0.0917	0.9506	0.7275
0.750 1V3	0.7532	0.0495	0.9506	0.7506
0.775 1V3	0.7810	0.0241	0.9506	0.7769
0.8 1V3	0.7998	0.0105	0.9506	0.7986
0.825 1V3	0.8287	0.0074	0.9506	0.8259
0.850 1V3	0.8545	0.0045	0.9506	0.8525
0.875 1V3	0.8757	0.0014	0.9506	0.8737
0.9 1V3	0.8982	0.0006	0.9506	0.8970

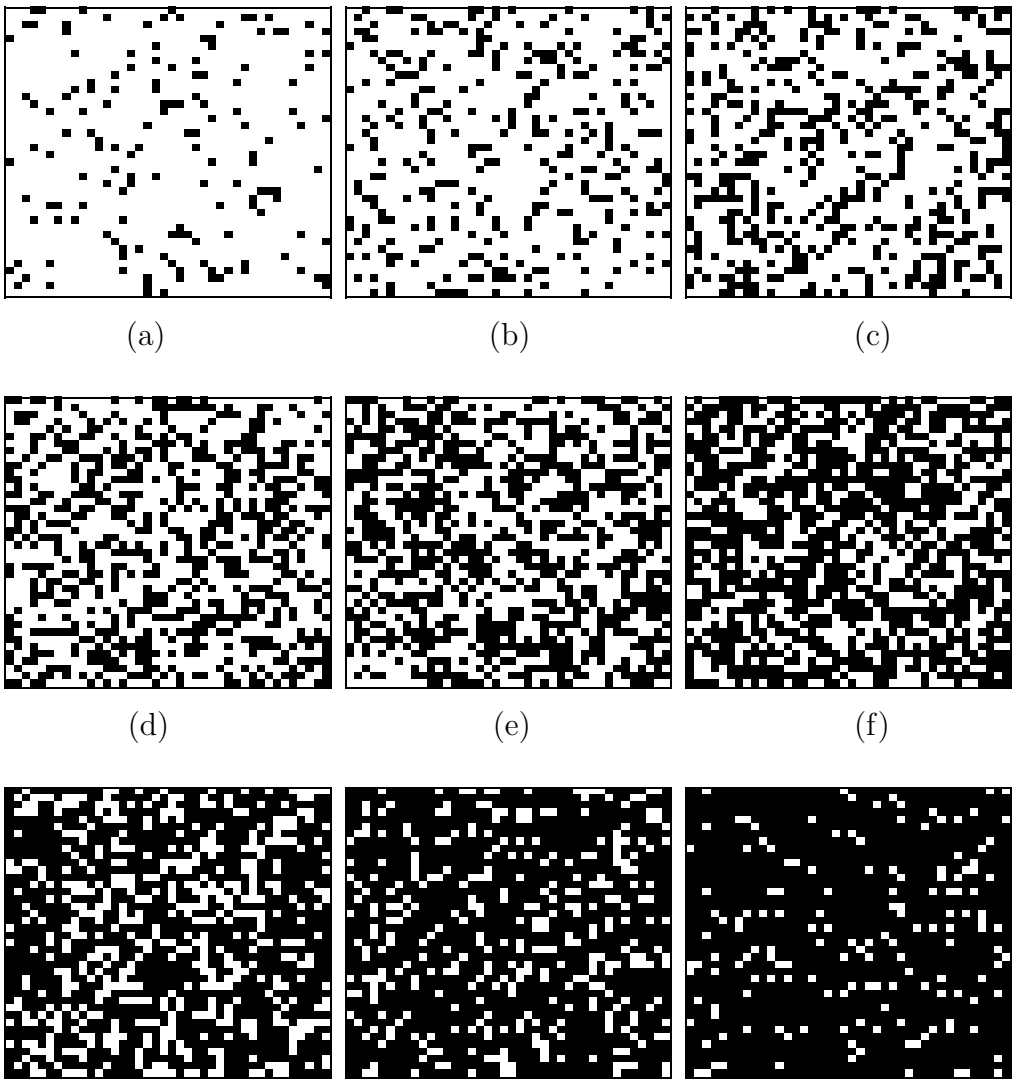
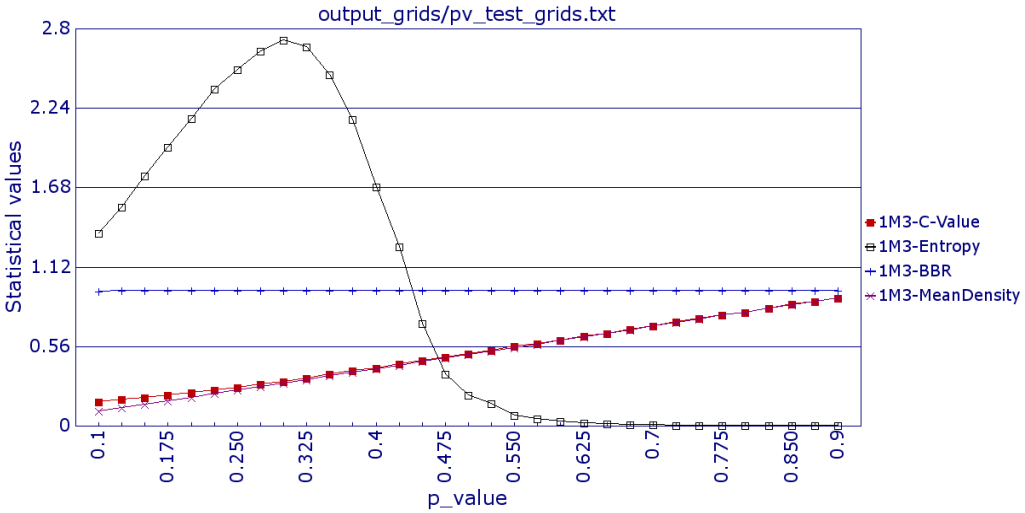
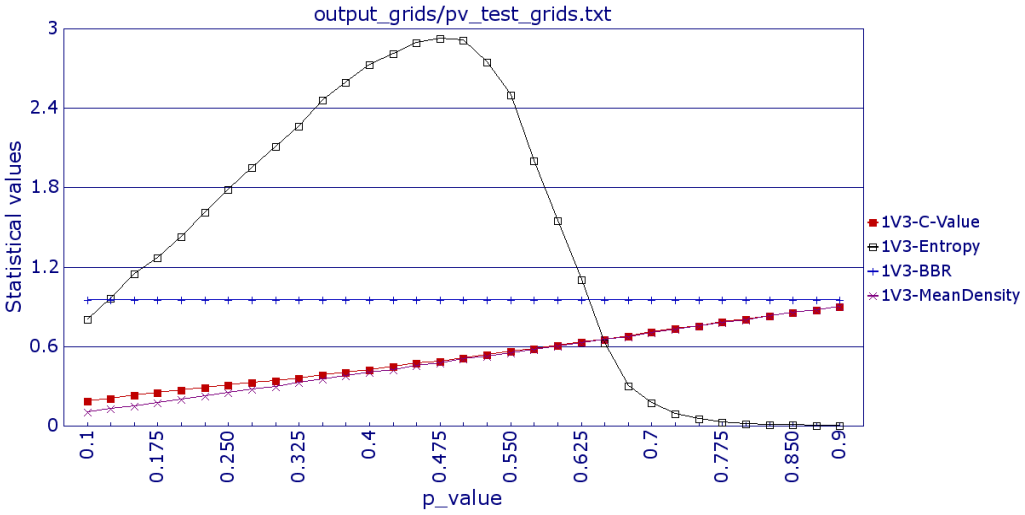


Figure 5.1: Examples of nine different probability grids. The 40 by 40 grids were created with an increasing probability that a cell would be active, where a probability values of 0.1 means around 10% of the cells on the grid will be activated and 0.9 correlates with 90%. The probability values displayed are (a) 0.1, (b) 0.2, (c) 0.3, (d) 0.4, (e) 0.5, (f) 0.6, (g) 0.7, (h) 0.8 and (i) 0.9. The active cells are shown as black and are distributed randomly across the whole grid. The number of active cells increases as the probability value increases. The random dispersal of the active cells leads to them occurring on the fringes of even the least populated grid of (a); this results in a relatively static BBR value for all the probability simulations.



(a)



(b)

Figure 5.2: The results of the four metrics testing the state of output grids across a range of grids created with probability values of p_value={0.100, 0.125, 0.150, 0.175, 0.200, 0.225,...., 0.875, 0.900}. Forty grids of 40 by 40 size were created for each probability value and then measured using the four metrics. The metric of each group of probability values were then averaged. A range of 3 was used with each neighbourhood; the legends on the graphs refer to the value that identifies an occupied cell on the grid (1), the neighbourhood used (M for Moore, or V for von Neumann, and the range used, followed by the relevant metric. (a) shows the use of an eight cell Moore neighbourhood and (b) the uses of a four cell von Neumann neighbourhood. The entropy metric peaks at a lower value probability grid in (a) as the eight cell Moore neighbourhood clusters the occupied cells into a single cluster sooner than the von Neumann neighbourhood. The BBR and mean density metrics are the same for both neighbourhood. The C-Value is virtually the same for both (a) and (b) and keeps step with the mean density.

5.2.2 Dynamic scenarios

This test looks at how the metrics perform when measuring the gathering of active agents towards the centre of the grid. Each cell can be occupied by a single agent, so each cell at a time step is either occupied or empty. Thus the gathering of the agents is in effect represented by a gathering of the occupied cells. The gathering algorithm, (*see Appendix C.4*), is based on an agent having three possible moves it can make to manoeuvre closer to the centre of the grid. If all three options are blocked, then the agent remains where it is. The orientation of the agent is based on an x, y axis through the centre point of the grid, hence the cross like shape of the cluster that forms in Figure 5.3. The gathering process terminates when either the specified number of time steps has been used up, or when no agent can move. A random ordered asynchronous (ROA) updating scheme was used where the order of the agents to be asynchronously moved was randomly created for each time step (*see subsection 2.6.3 and subsection 4.7.1*).

A series of grids were created with a selection of agents randomly placed on them. Each test was run for fifty time steps, or less if movement on the grid halted. Four grid sizes were used, 25 by 25, 50 by 50, 75 by 75 and 100 by 100. The range of the number of agent tested on a each grid was $Agents$={2, 8, 12, 16, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200, 250, 300, 350}. This provided the opportunity to compare what the metrics registered from when there was a very small number of active agents through to when the grid was more crowded. The latter is shown in Figure 5.3, which depicts a selection of the gathering process of 350 agents randomly placed on a 100 by 100 grid. The gathering process is clearly displayed. The graph relating to this specific test can be found in Figure 5.4(d).

The graphical analysis of the simulation of 350 agents across the four grid sizes and using a Moore neighbourhood with a maximum range of 3 are laid out in Figure 5.4. The relevant tables and graphs for all the permutations run for this test can be found in appendix B.1.1. As expected, the mean density is shown as a straight line in all four graphs as the number of agents is consistent across the time steps. The entropy value behaves in a similar way to that shown in the probability scenario tests in subsection 5.2.1. This reflects the fact that the entropy metric is based on the number of different cluster sizes, and the gathering of the agents towards a central cluster mirrors the move towards a single cluster as the probability value increases. In Figure 5.4(b)-(d) the BBR can be described as a reflection of the C-Value, crossing at a time step after the point where the entropy metric is at its apex. The C-Value starts at or just above the mean density value

and then increases as the agents gather together and the connectedness between them becomes greater. The BBR value moves in the other direction as the distance between the row and column positions of the most outlying agents decreases. The effective shrinkage of the grid area containing active agents can be seen clearly in the grid time steps shown in Figure 5.3.

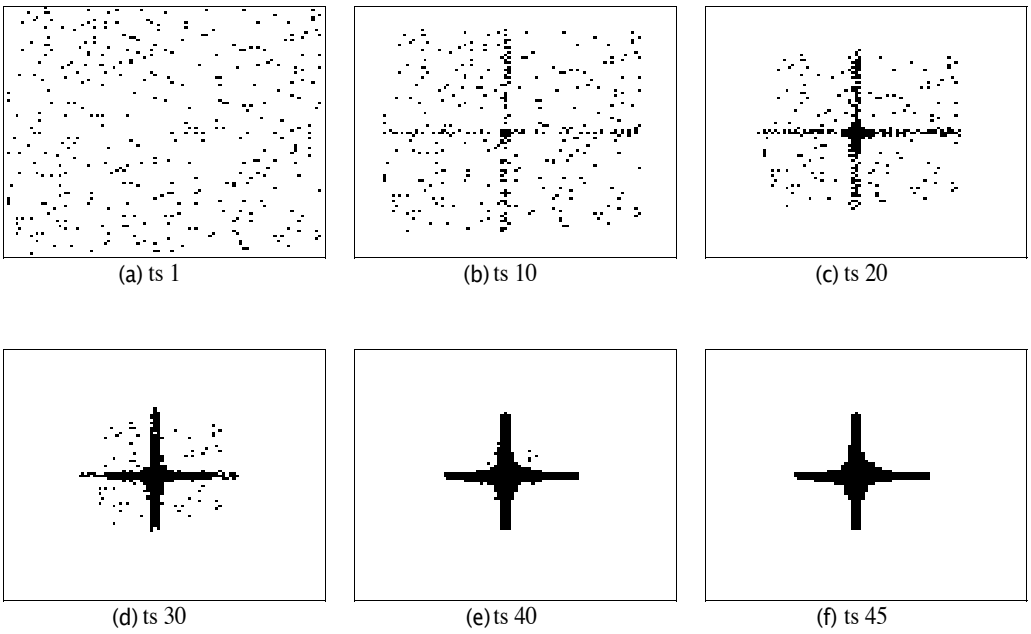


Figure 5.3: Examples of the gathering of 350 agents into the centre of an 100 by 100 grid. (a) The active agents are randomly placed on the initial grid. The gathering process can be seen in action in (b)-(e), time steps 10, 20, 30 and 40 respectively. The gathering process has completed by time step 45, see (f).

The shape of the lines plotting the metrics becomes more pronounced as the grid size increases. The amount of activity needed to complete the gathering process is indicated by the number time steps (*see Figure 5.4 and Figure 5.5*). In Figure 5.5(a) the 350 active agents occupy 56% of the cells on the 25 by 25 grid, making it the most crowded grid used in this test. The gathering process is completed after just eight time steps, as opposed to the forty-five time steps in the simulation using the 100 by 100 grid. In the graph relating to the 25 by 25 grid, (*see Figure 5.4(a)*), the C-Value provides the most pronounced indicator of the change from the initial random grid to the final grid showing the active agents gathered together towards the centre of the grid. The curve of the BBR line is very shallow, which reflects the small amount of change in the outer limits of the active agents locations before the gathering process concludes. In all four

graphs the C-Value starts on the initial randomly generated grid at or around the same value as the mean density. This is consistent with the results from the previous section, subsection 5.2.1. But once the agents begin to gather the C-Value, as would be expected, increases. This move from an initial grid with randomly placed active cells to one where the active cells are ordered in a single centralised group is marked on all four simulations by how the C-Value increases and moves away from the mean density.

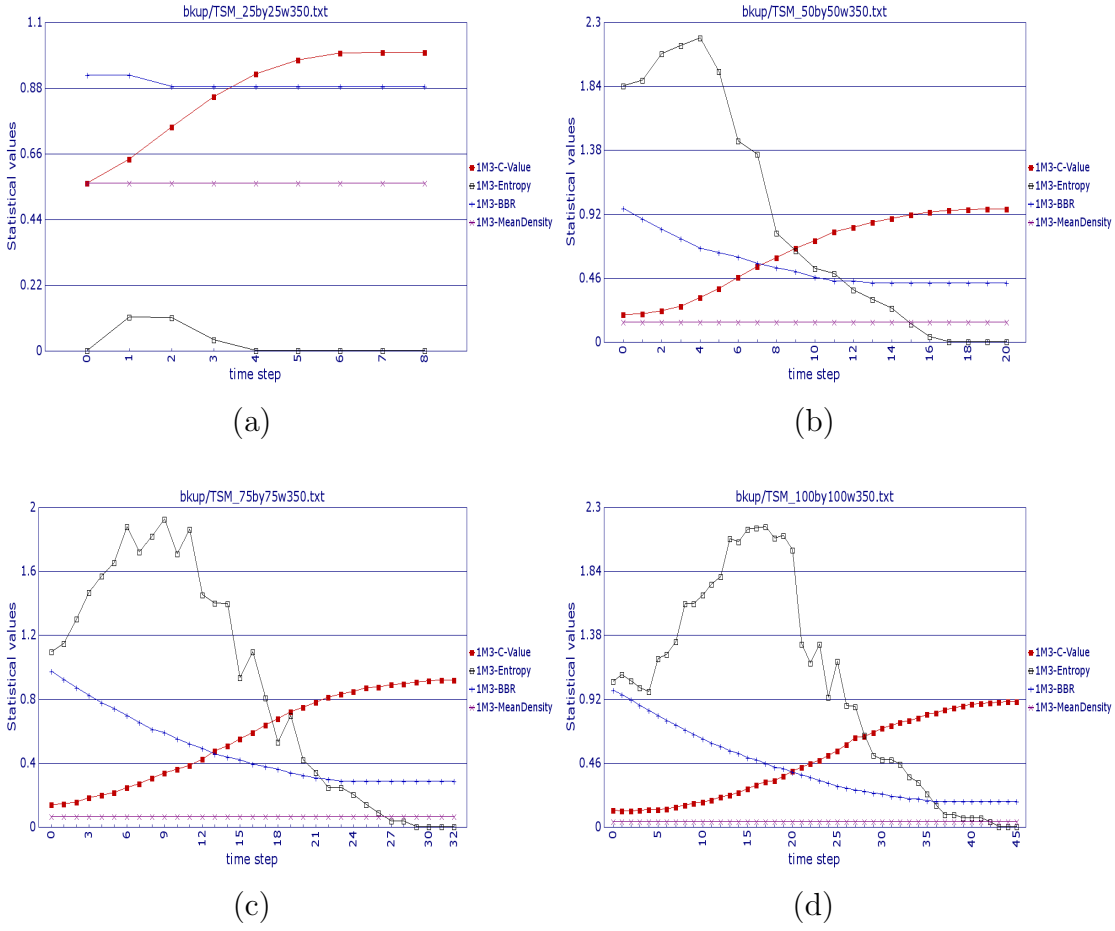


Figure 5.4: Examples of the gathering of 350 agents into the centre of a square grid of (a) 25 by 25, (b) 50 by 50, (c) 75 by 75 and (d) 100 by 100. The C-Value is consistent on all four graphs and is the only metric that provides provides information on the state of the grid across the time steps in (a). In (b)-(d) the other metrics also begin to register the gathering process, with the obvious exception of the mean density, as the number of active agents is consistent across the relevant time steps. The move of the C-Value from the mean density reflects the move from a random placement of the active cells on the initial grid to an ordered, centralised grouping. A Moore neighbourhood was used with a maximum range of 3; this is reflected in the legends which start with 1M3- followed by the relevant metric.

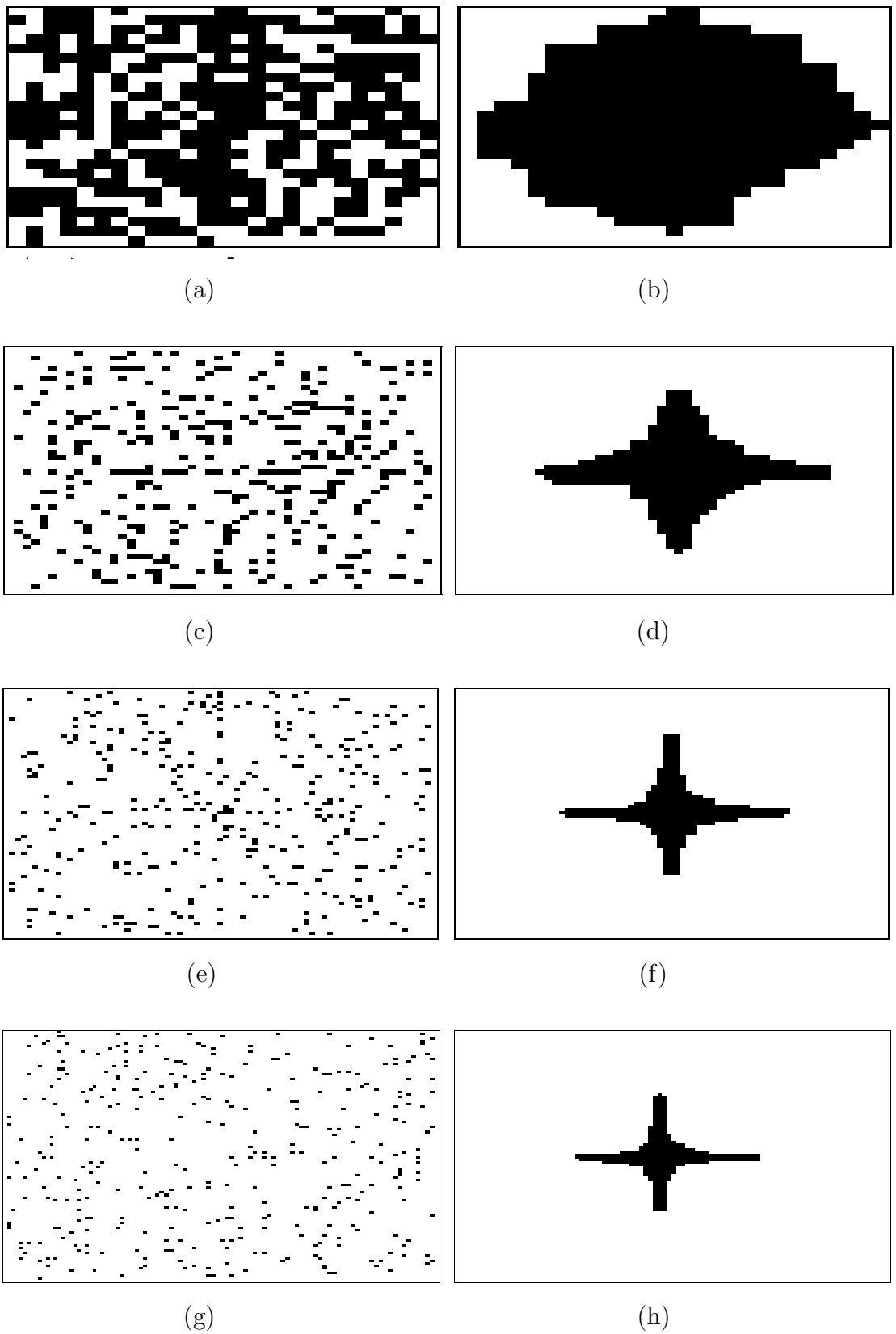


Figure 5.5: The first and last grids of the gathering of 350 agents into the centre of a square grid of (a)-(b) 25 by 25 taking eight time steps, (c)-(d) 50 by 50 taking 20 time steps, (e)-(f) 75 by 75 taking 32 time steps and (g)-(h) 100 by 100 taking 45 time steps. A Moore neighbourhood with a maximum range of 3 was used.

The other area of interest from these simulations was how the metrics preform when measuring just a few agents. Figure 5.6 shows the results of the simulations run with eight agents ((a)-(d)) and twelve agents ((e)-(h)), using the four grid sizes of 25 by 25, 50 by 50, 75 by 75 and 100 by 100. All the simulation use a Moor neighbourhood and a range of three. The scarcity of active agents meant that there was relatively few chances of clusters forming until the agents converged at the centre of the grid. This can be seen in the shape of the entropy line on the graphs. The metric is based on the probability that an active agent is part of a cluster of size 's' and that there is cluster size diversity, (*see subsection 3.6.3*). The latter means that there has to be more than one cluster and at least two different size of cluster, otherwise the entropy is zero, such as when all the active agents are gathered together in one single cluster or when the clusters all have the same number of active agents. Therefore, the potential for the entropy metric to track changes in this gathering process simulation, when there is a very small number of active agents, is low. For example, a grid with eight active cells has a limited number of cluster combinations involving different cluster sizes before all the cells are in one cluster of eight. Table 5.3 shows the breakdown of the clusters identified in the four simulations with eight active cells. The time steps where there is cluster size diversity is where the corresponding graphs in Figure 5.6(a)-(d) show the entropy rising above zero. In Figure 5.6(a) the entropy level is above zero at the initial time step through to the time step 8. This is reflected in Table 5.3(1a-b) where there are two clusters of different sizes. However, in Table 5.3(4d-e, and g) there are more than one cluster, but they are the same size, The resulting zero entropy can be seen for the corresponding time steps of 14-29 and 31 in Figure 5.6(d)

The mean density remains the same across all the time steps and the value is only just above zero as the active agents represent such a minuscule percentage of the total number of cells on a grid. The BBR metric shows a smooth curve across all the grids, but it tails off before the gathering process is completed. However, the C-Value continues to have an upwards trend after both the entropy and BBR lines have ceased to register anything. The shape of the C-Value metric curve is observed at a relatively low agent population level. Figure 5.6(a)-(d) shows the shape of the C-Value beginning to be evident in the tests run with 8 and (e)-(h) with 12 agents, although a smoother curve does not become evident until more agents are on the grid (*see appendix B.1.1 for graphs of all the permutations run for this test*). The tests on the reactive diffusion chemotaxis model will look at whether a non centralised gathering of agents affects the curve of the BBR and C-Value graph. But the analysis of grids with a low population of agents does

present a challenge, although the C-Value appears to provide more information across all the time steps than the other metrics.

Table 5.3: Clusters and singletons during the simulation of eight active agents on four grid sizes; (1) 25 by 25, (2) 50 by 50, (3) 75 by 75 and (4) 100 by 100. The entropy of the active agents and grid is zero unless there is at least two clusters of different sizes. Thus the 25 by 25 grid will start with an entropy value, see (1a), whereas multiple clusters of the same size will have a zero entropy, see (2b and c), (3b and d) and (4d, e and g). This is shown by the entropy values displayed in the corresponding graphs, (*see Figure 5.6*).

Grid size	Time steps: number and size of clusters, with number of singletons
(1) 25 by 25	(a) 0-5: a 2 and 3 cell cluster, with 3 singletons, (b) 6-8: a 2 and 4 cell cluster, with 2 singletons, (c) 9: a 6 cell cluster with 2 singletons (d) 10: an 8 cell cluster with no singletons
(2) 50 by 50	(a) 0-9: a 2 cluster, with 6 singletons, (b) 10-12: 2 x 2 cell cluster, with 4 singletons, (c) 13: 2 x 3 cell cluster with 2 singletons (d) 14: a 2 and 3 cell cluster with 3 singletons (e) 15-17: a 2 and 4 cell cluster with 2 singletons (f) 18: a 3 and 5 cell cluster no singletons (g) 14: a 2 and 6 cell cluster no singletons (h) 14: an 8 cell cluster no singletons
(3) 75 by 75	(a) 0-11: a 2 cell cluster, with 6 singletons, (b) 12: 2 x 2 cell cluster, with 4 singletons, (c) 13-17: a 2 cell cluster, with 6 singletons, (d) 18-23: 2 x 2 cell cluster, with 4 singletons, (e) 24-25: a 4 cell cluster, with 4 singletons, (f) 26: a 5 cell cluster with 3 singletons (g) 27-28: a 6 cell cluster with 2 singletons (h) 29-31: a 7 cell cluster with 1 singleton (i) 32: an 8 cell cluster with no singletons
(4) 100 by 100	(a) 0-3: a 2 cell cluster, with 6 singletons, (b) 4: a 3 cell cluster, with 5 singletons, (c) 5-13: a 2 cell cluster with 6 singletons (d) 14-16: 2 x 2 cell cluster with 4 singletons (e) 17-29: 3 x 2 cell cluster with 2 singletons (f) 30: 2 x 2 and 3 cell cluster with 1 singleton (g) 31: 3 x 2 cell cluster with 2 singletons (h) 32-33: a 2 and 4 cell cluster with 2 singletons (i) 34-39: a 2 and 5 cell cluster with 1 singleton (j) 40-42: a 2 and 6 cell cluster with 0 singletons (k) 43-44: a 7 cell cluster with 1 singleton (l) 45: an 8 cell cluster with 4 singletons

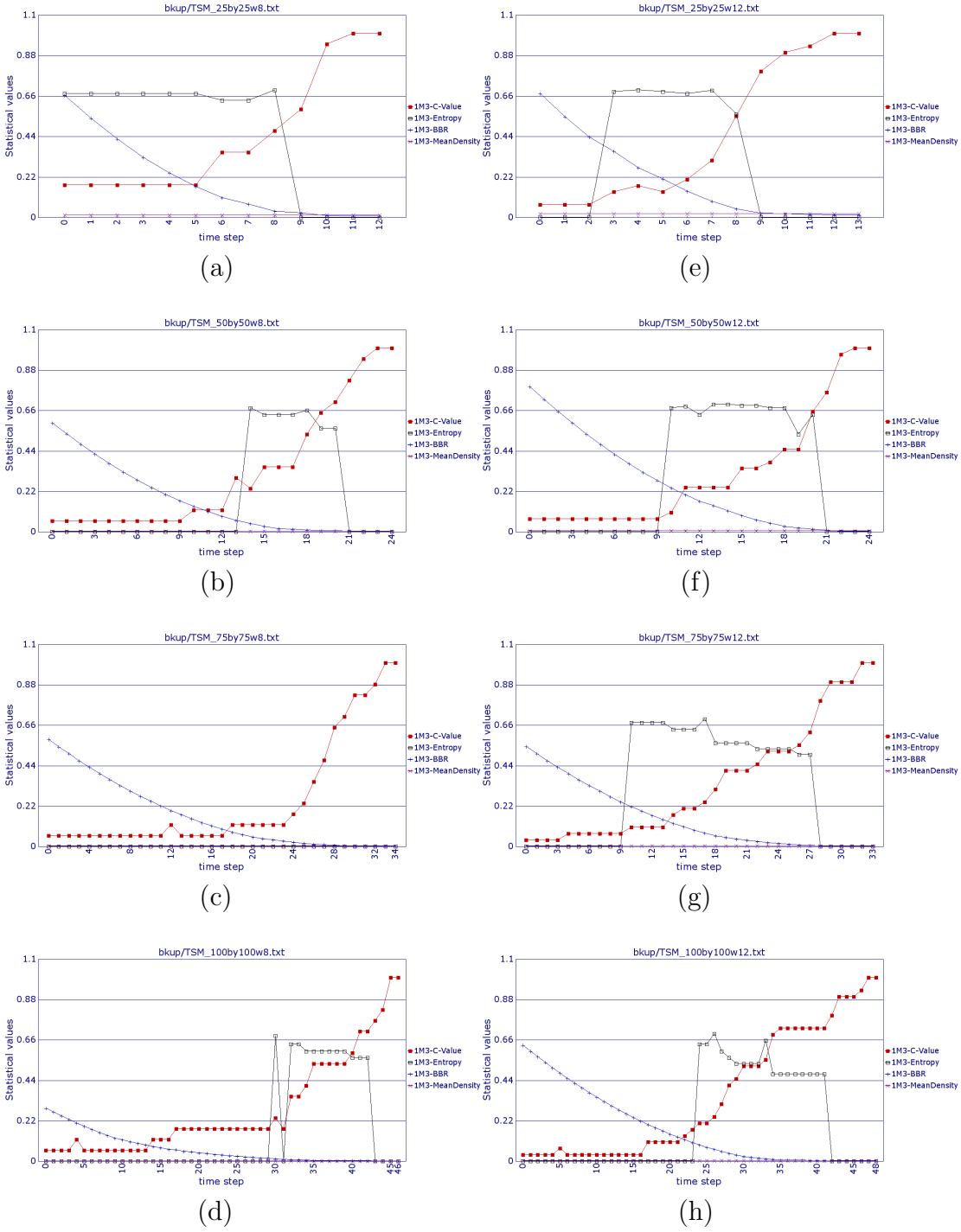


Figure 5.6: The measurement of simulations of the gathering of 8 (a-d) and 12 (e-h) active agents into the centre of a square grid. Four grid sizes were used, 25 by 25 (a & e), 50 by 50 (b & f), 75 by 75 (c & g) and 100 by 100 (d & h). The mean density remains the same across all time steps, while the BBR and entropy metrics do not register anything in the final time steps. The C-Value indicates the gathering process even for the smallest number of agents on the largest grid, see (d). The legend show the metric used with a prefix of 1M3, signifying the value of an occupied cell (1) and the use of a Moore neighbourhood (M) with a maximum range of 3 (3).

The next series of tests were run to see if changing the neighbourhood and range had any real affect on the performance of the metrics. The simulations compare the performance of the two types of neighbourhoods, Moore (M) and von Neumann (V), and the three ranges of 1, 2 and 3. They were carried out on a 25 by 25 grids with three low agent populations of 4, 8 and 12, and one 100 by 100 grid with 350 agents (*see Figure 5.8, Figure 5.9, Figure 5.10 and Figure 5.11 respectively*). This focuses mainly on the low end where there is obviously more difficulty in gaining any distinctive measure. The middle and higher populations of agents exhibit a smooth curve for both the C-Value and BBR. Consequently, only one grid with a large population of 350 agents is shown; this is on a larger grid of 100 by 100, where the the 350 agents only represent 3.5% of the total number of cells and, consequently, a longer sequence of time steps is needed to complete the gathering process, providing more information to analyse.

The three simulations using grids with a just a few active agents show that there is a very slight difference between the two neighbourhoods, and that the changing of the range to 3 has a minor effect. The nature of the simple algorithm used to gravitate the agents towards the approximate centre of the grid should not be forgotten, but the results suggest that the selection of a Moore neighbourhood and a maximum range of 3 has a very slight impact when there is a low number of active agents on the grid. When there are only four active agents there is no distinction between the different ranges used (Figure 5.8); whereas with eight (Figure 5.9) and twelve (Figure 5.10) active agents there is some differential, but it is very little and does not change the basic curve of the lines on the graphs. The difference is between the C-Value in Figure 5.9(c) and in Figure 5.9(a) & (b). In (c) the C-Value retains the level of connectedness for time steps 3 and 4, and then starts to rise for 5; whereas for (a) & (b) the C-Value drops for time steps 3 to 5, before starting to rise at time step 6. This indicates that the use of a range of 3 in (c) has picked up and grouped some active cells identified as singletons with range 1 and 2 in (a) and (b) respectively. The use of a range of three also highlights a difference in a Moor neighbourhood with twelve active agents in the grid, as can be seen in the graph in Figure 5.10(c) when compared to Figure 5.10(a) & (b). While the difference in both examples is minimal, it does suggest that the use of a range of 3 with a Moore neighbourhood provides a better option.

The simulations with eight and twelve active agents support the observations made in the previous test about measuring a small number of active agents. Consequently, the mean density is a constant figure running just above zero; the BBR registers more than the C-Value in the early time steps, but this reverses before

half the time steps are completed with the C-Value providing much clearer information about the changing state on the grid. The entropy value only rises above zero on Figure 5.9(b)-(c) and Figure 5.10(b)-(c) and (e)-(f). The simulation with four active agents also illustrates the potential for artefacts being produced by computational simulations. The four active agents gather together in the middle of the grid after about twelve time steps, however instead of stopping, the algorithm enters into a cycle where the shape of the cluster alternates through four stages, as seen in Figure 5.7. The steady state of the C-Value at a high value of connectedness indicates the artefactual behaviour.

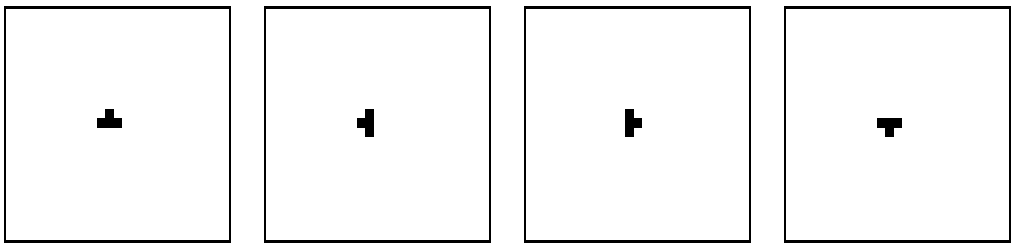


Figure 5.7: The last four grids from the simulation of 4 active agents on a 25 by 25 grid with a Moore or a von Neumann neighbourhood with a range of 1. This cycle of the four agents clustered together in the centre of the grid is an artefact of the simple gathering algorithm used in this test. This results in the C-Value in the graphs displayed in Figure 5.8(a)-(f) ending in a long horizontal line, rather than as soon as the active agents are first grouped together.

However, any thoughts that a larger group of agents on a larger grid would reveal significant difference between, for example, von Neumann with range 1 and Moore with range 3, would seem to be incorrect. There is little evidence of any real difference. If the agents are close enough, then sweeps with a range depth of 1 and 2 will pick them up, leaving little or nothing for a search at range 3 to find, often resulting in no difference in the figures between range 2 and range 3. While Figure 5.11(b) and (c) are almost identical in shape, (a) has a few differences including the entropy line starting at a much lower value. However, the starting entropy value is dependent on the number of different sized clusters, if any, found in the random composition of the initial grid. So while it could be construed as an indication of how random the spread of active agents might be on the initial grid, the interest in the entropy metric is in the shape of the graph and the at what point the apex and nadir of the graph occurs.

Indeed, the main difference between the two neighbourhoods is that the C-Value increases to 0.8843 for the Moore neighbourhood in Figure 5.11(a)-(c), and 0.9169 for the von Neumann neighbourhood in Figure 5.11(d)-(f). This is understandable as the maximum connectivity of 350 active agents is greater for the Moore's eight neighbourhood, resulting in a lower C-Value. Also, the shape of the entropy apex and descent are distinctive to each neighbourhood type, with a wider top section followed by a steep descent for the von Neumann neighbourhood; whereas the Moore neighbourhood has a staggered descent. But this does not appear of any great significance. The tables from which the graphs in Figure 5.11 were formed are shown in appendix B.1.2. These show that the C-Value and BBR cross over at time step 19 for both neighbourhoods, while the apex of the entropic measurement is at time step 14 for the Moore neighbourhood and time step 18 for the von Neumann neighbourhood. However, although the overall conclusion is that there appears to be only a very minimal benefit in using the Moore neighbourhood, as the larger Moore neighbourhood involves more connections, and thus has the theoretical potential of revealing a greater level of differentiation between the level of connectedness or compactness of a cluster, it will remain the principle setting used in the tests, with a catch all range of 3.

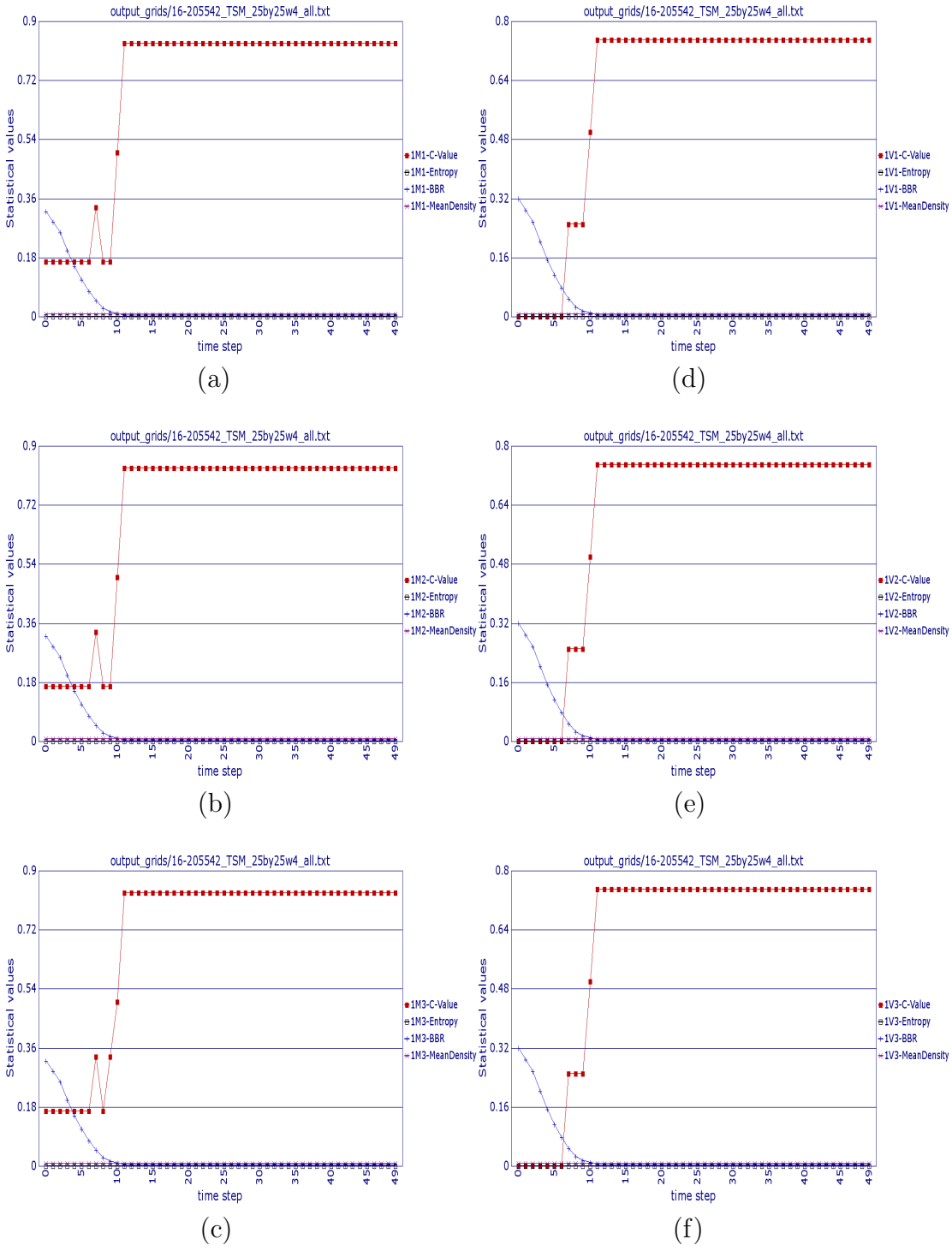


Figure 5.8: The combinations of neighbourhood search type and search range on a 25 by 25 grid with **4** agents. (a)-(c) show the Moore (M) neighbourhood, (d)-(e) the von Neumann (V). (a) and (d) have a range of 1, (b) and (e) a range of 2, and (d) and (f) a range of 3. The prefixes in the legends indicate the value of an active agent, the neighbourhood type and the range, such as 1M2 for (b) or 1V1 for (d). The gathering process ends relatively quickly, although an artefactual behaviour of the simple gathering algorithm results in the shape of the final four agent cluster cycling through 4 stages, (*see Figure 5.7*).

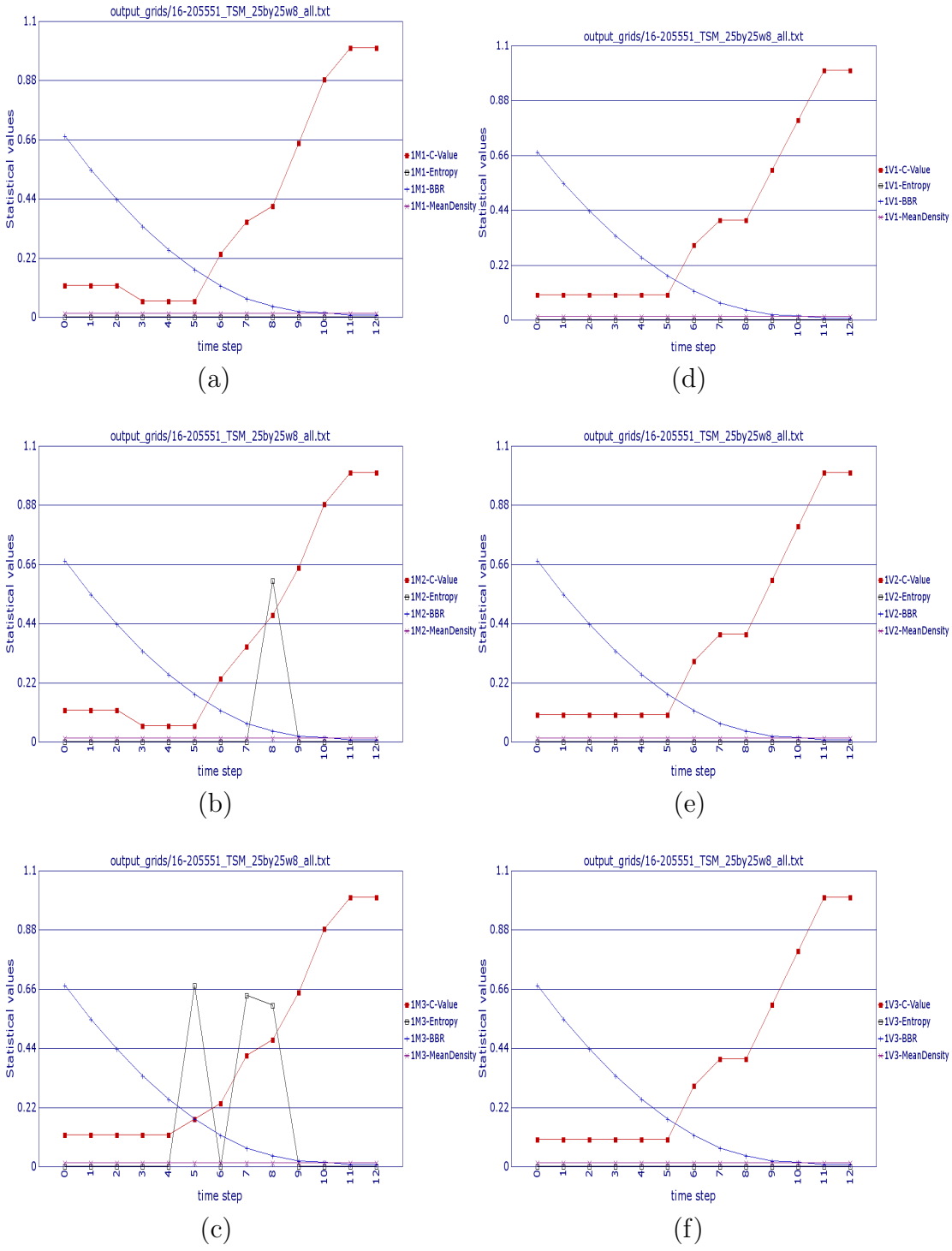


Figure 5.9: The combinations of neighbourhood search type and search range on a 25 by 25 grid with **8** agents. (a)-(c) show the Moore (M) neighbourhood, (d)-(e) the von Neumann (V). (a) and (d) have a range of 1, (b) and (e) a range of 2, and (d) and (f) a range of 3. The prefixes in the legends indicate the value of an active agent, the neighbourhood type and the range, such as 1M2 for (b) or 1V1 for (d). The shape of the C-Value is more distinguishable than with just 4 agents. In (c) a range of 3 finds more clusters in time steps 3 to 5, than range 1 in (a) & range 2 in (c).

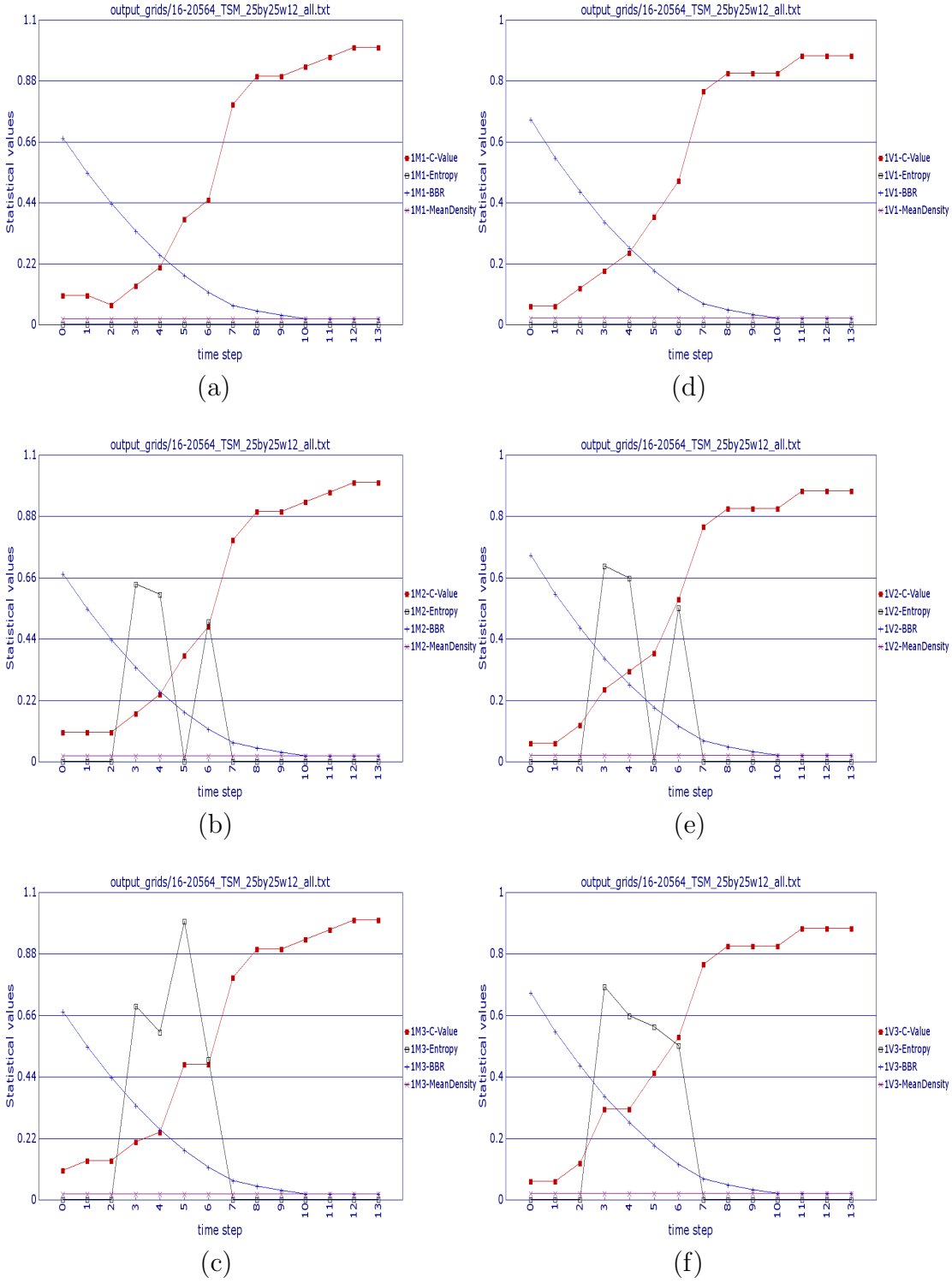


Figure 5.10: The combinations of neighbourhood search type and search range on a 25 by 25 grid with **12** agents. (a)-(c) show the Moore (M) neighbourhood, (d)-(e) the von Neumann (V). (a) and (d) have a range of 1, (b) and (e) a range of 2, and (d) and (f) a range of 3. The prefixes in the legends indicate the value of an active agent, the neighbourhood type and the range, such as 1M2 for (b) or 1V1 for (d). The C-Value provides a clearer indicator of the changing grid across all the time steps.

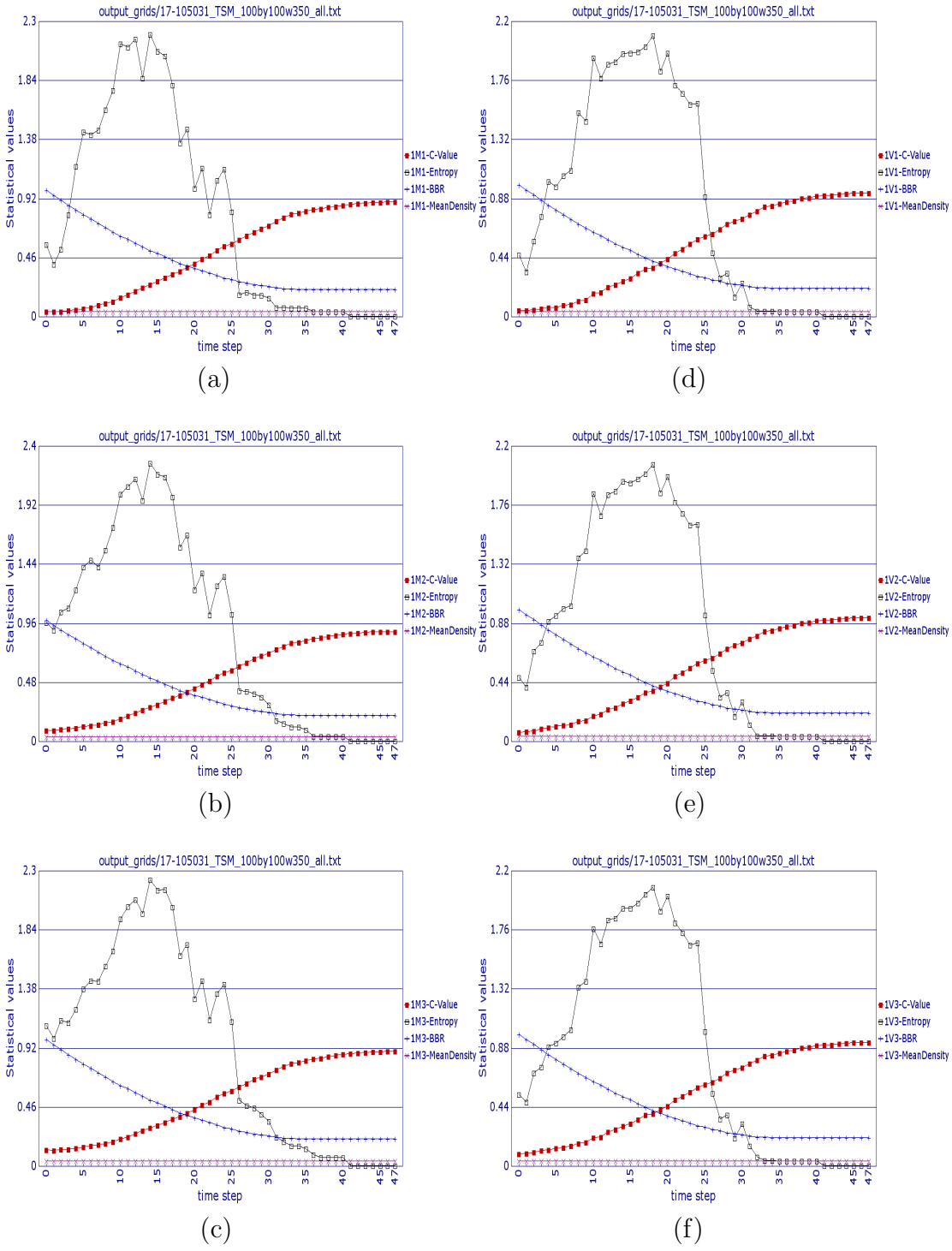


Figure 5.11: The combinations of neighbourhood search type and search range on a 100 by 100 grid with **350** agents. (a)-(c) show the Moore (M) neighbourhood, (d)-(e) the von Neumann (V). (a) and (d) have a range of 1, (b) and (e) a range of 2, and (d) and (f) a range of 3. The prefixes in the legends indicate the value of an active agent, the neighbourhood type and the range, such as 1M3 for (b) or 1V1 for (d). The C-Value and BBR appear almost as a reflection of each other, although the C-Value continues to rises over the later time steps, while the BBR remains the same.

5.2.3 Probability and dynamic scenarios combined

A final scenario test was run that combined both the probability and dynamic scenarios. Probability grids of 40 by 40 were created for nine probability settings, one for each of p_value={0.1, 0.2, ..., 0.9}, and then used as the initial grid for the dynamic scenario. This enabled the performance of the metrics to be observed across a full range of populated grids, especially the more densely packed ones. The dynamic scenario tests in the previous section were mainly focused on sparsely occupied grids.

In the probability scenario just the initial, randomly generated grids were measured and the C-Value ran alongside the mean density (*see Figure 5.2*). In that test the highest values were recorded, as would be expected, for the most densely populated grids with a probability setting of 0.9, where the C-Value peaked at an average of 0.8982 and the mean density at 0.8970 (*see Table 5.1*). Table 5.4 and Figure 5.12 show the C-Value for each starting and end grid of these combination simulations, where it rises to 0.9245 and above. The mean density remained the same across the time steps of each simulation as there was no fluctuation in the number of active agents. The tables showing the time steps and metrics for all four metrics are in appendix B.1.4.

This combination test shows how the four metrics capture the change of the grids from a randomly generated state to one where some form of order has been imposed by gathering the active agents towards the centre of grid (*see Figure 5.13*). The results correspond with those of the dynamic scenario tests, where overall the C-Value provided the most consistent and distinctive measure of the gathering process on grids with a low number of agents. What is of interest is how the performance of the C-Value persists through medium to densely populated grids. The BBR metric had provided a comparable performance in the dynamic scenario tests, but in this test it falls away as the number of active agents increases. As the other metrics register less and less, the C-Value continues to record the gathering process. Indeed, in the two most densely packed grids, Figure 5.13(h) & (i), the C-Value is the only metric that registers any relevant information concerning the changing state of the grid.

Table 5.4: The C-Value of the start and end grids of a range of probability grids, *p_value*={0.1, 0.2, ..., 0.9}, used as initial input for the dynamic scenario, using a 40 by 40 grid. The C-Value registers an increase across all the simulations. It rises from 0.1660 to 0.9245 for the grid with the least number of active agents on; and on the most populated grid it increases from 0.9034 to 0.9995. The latter is a grid where around 90% of the cells are occupied. A Moore neighbourhood with a range of 3 was used (M3).

probability	**Start-CV**	**End-CV**
0.1 1M3	0.1660	0.9245
0.2 1M3	0.2188	0.9544
0.3 1M3	0.3192	0.9760
0.4 1M3	0.4120	0.9901
0.5 1M3	0.5203	0.9940
0.6 1M3	0.6177	0.9965
0.7 1M3	0.7003	0.9986
0.8 1M3	0.7989	0.9988
0.9 1M3	0.9034	0.9995

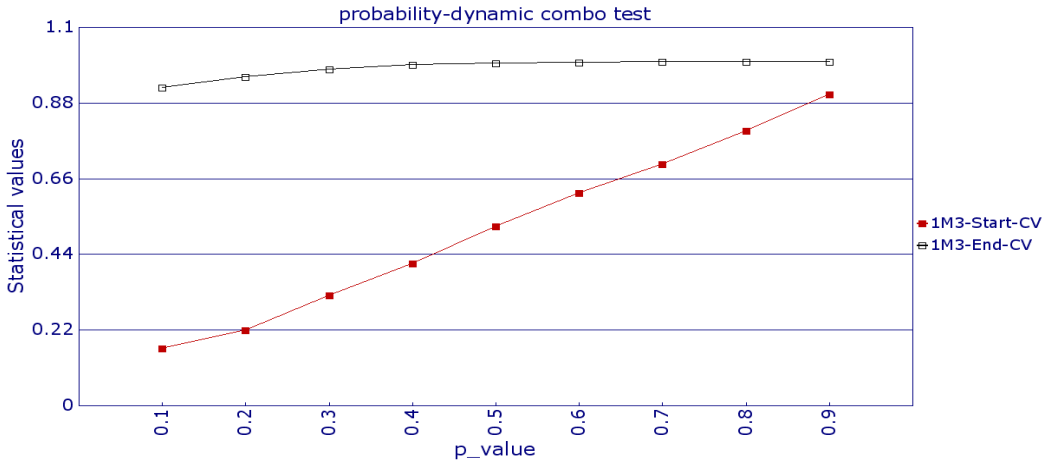


Figure 5.12: The initial and final grids of a range of probability grids, *p_value*={0.1, 0.2, ..., 0.9} used as initial input for the dynamic scenario, using a 40 by 40 grid. As can be seen, the C-Value registers an increase for all the simulations run. The size of the increase reflects the density of active agents on a grid. The probability value of 0.9 represents an occupation rate of around 90%, so the scope for the an increase in the C-Value is limited.

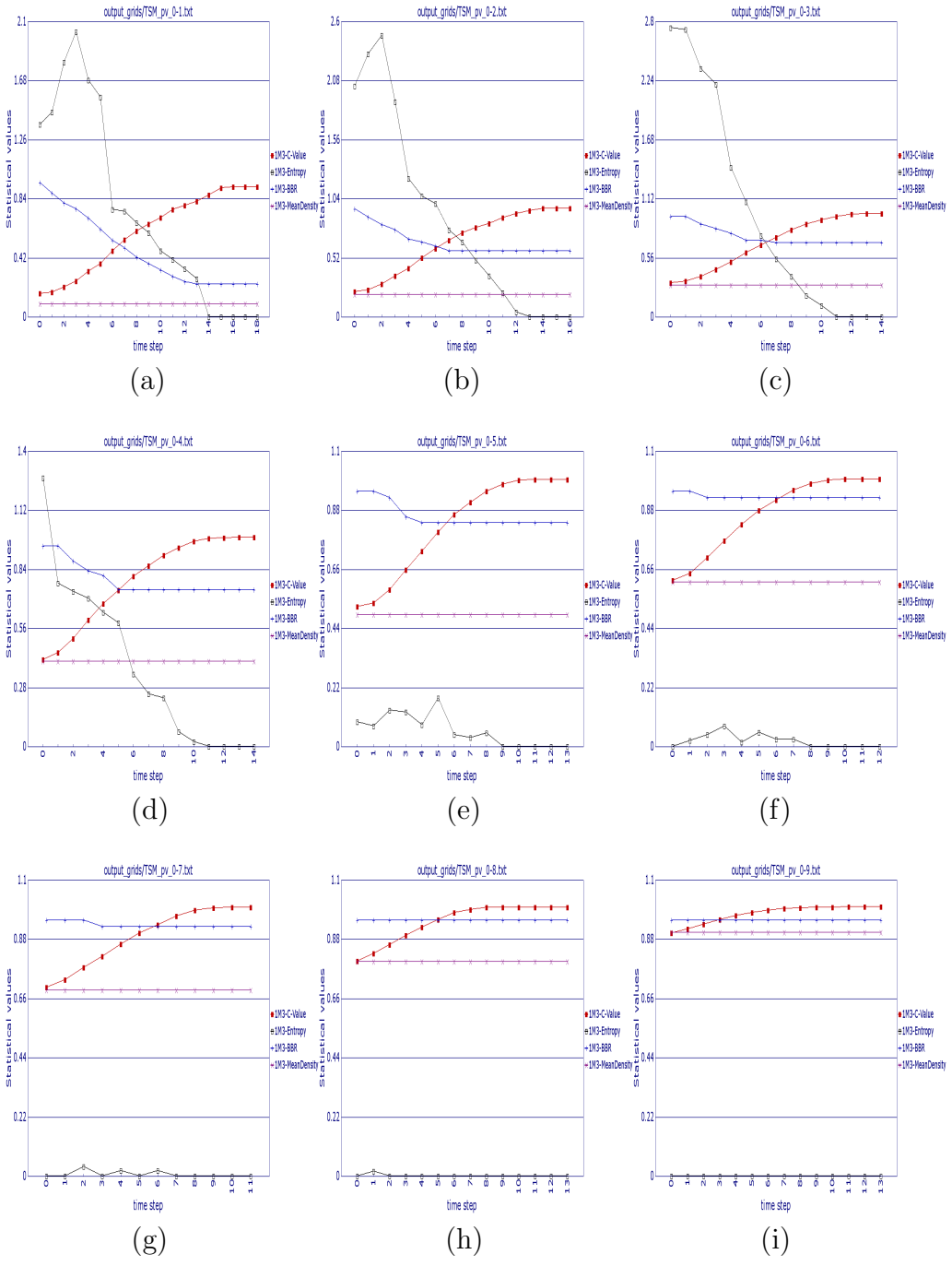


Figure 5.13: Measurement of a range of probability grids, p_value={0.1, 0.2, ..., 0.9} used as initial input for a dynamic scenario, using a 40 by 40 grid. (a) 0.1, (b) 0.2, (c) 0.3, (d) 0.4, (e) 0.5, (f) 0.6, (g) 0.7, (h) 0.8 and (i) 0.9. The C-Value provides a consistent representation of the changing state of the grids as the active agents on them gather towards the centre of the grid. The mean density provides no additional information and as the number of occupied cells increases the other metrics, apart from the C-Value, provide less and less useful information. On the two most densely populated grids (h) and (i) the C-Value is the only metric that provides any information concerning changes in the state of the grid.

5.3 Particle model

Two facets of the reaction-diffusion and chemotaxis model were used to analyse the performance of the metrics; the state of the output grid during the extinction regime and during the gathering regime (*see subsection 4.6.5*). The former is populated with cells in an excited state that do not gather together, but instead represent a random dispersion of excited cells across the grid. The latter models the decentralised gathering of amoeba within the grid space.

In the first test using the extinction regime, the initial grid had 10% of its cells set to an excited state. Table 5.5 shows the averaged state of the final output grid from a 40 by 40 grid run over 400 time steps 5 times each for a probability setting of $(p_E, p_A) = (0, 0)$ and $p_T = \{0.1, 0.2, ..., 0.9, 1\}$. The graphical representation of each metric can be seen in Figure 5.14.

Table 5.5: Average of 5 simulations of a 40 by 40 grid over 400 time steps for a $p_T = \{0.1, 0.2, ..., 0.9, 1.0\}$ and $(p_E, p_A) = (0, 0)$. The initial grids had 10% of their cells set to an excited state and randomly placed on the grid. The last grid in each simulation was measured. The ID shows the *p_value* used to create the initial grid, followed by 2M3, which represents the value of an excited cell (2) and a Moore neighbourhood (M) with a range of 3 (3).

ID	C-Value	Entropy	BBR	MeanDensity
0.1 2M3	0.0000	0.0000	0.0000	0.0000
0.2 2M3	0.0000	0.0000	0.0000	0.0000
0.3 2M3	0.1999	1.7456	0.9506	0.1293
0.4 2M3	0.2261	2.1890	0.9506	0.1923
0.5 2M3	0.2444	2.3601	0.9506	0.2279
0.6 2M3	0.2678	2.5131	0.9506	0.2632
0.7 2M3	0.3026	2.6423	0.9506	0.2896
0.8 2M3	0.3064	2.6136	0.9506	0.2954
0.9 2M3	0.3335	2.7269	0.9506	0.3206
1.0 2M3	0.0000	0.0000	0.0000	0.0000

In Figure 5.14 all four metrics show the sharp rise once p_T passes 0.2 and an equally rapid drop back down to zero excited cells after 0.9, as reported in the original work [Fatès et al., 2008] and replicated in subsection 4.6.5. As the p_T value increases between 0.3 and 0.9 the number of excited cells increase, after which it plummets back down to zero excited cells for $p_T = 1$. The BBR flat lines between 0.3 and

0.9 with a value of 0.9506, whereas the C-Value, entropy and mean density values show similar increases. The BBR values reflect the observations of the probability scenarios, (*see subsection 5.2.1*), where the random dispersal included 'active' cells on all four edges of the grid, giving a static BBR, (*see Figure 5.15*).

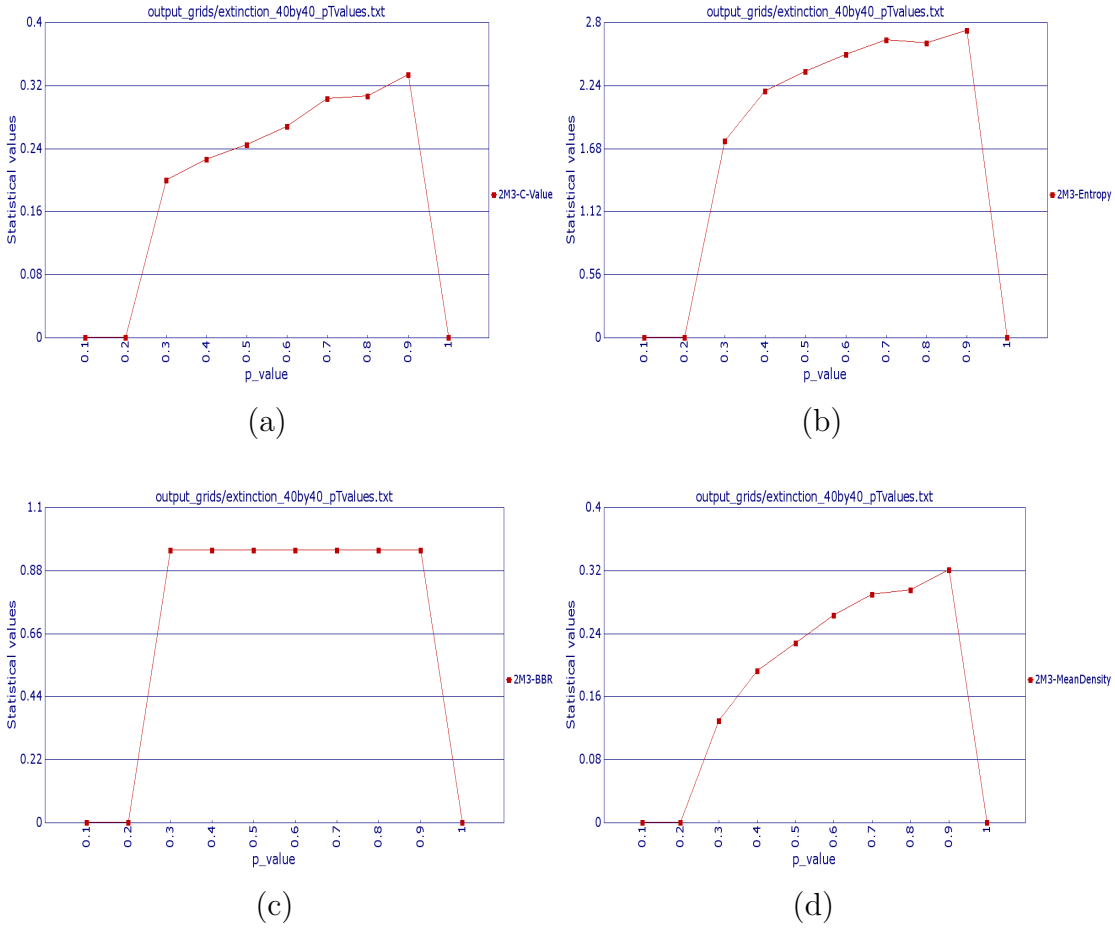


Figure 5.14: Average of 5 simulations of a 40 by 40 grid over 400 time steps for $(p_E, p_A) = (0, 0)$, and each of $p_T = \{0.1, 0.2, ..., 0.9, 1.0\}$, indicated by the *p_value* on the graphs. The initial grids had 10% of their cells set to an excited state and randomly located on the grid. The graphs show the average of the last grid of each of the five simulations, as measured by the four metrics: (a) C-Value, (b) Entropy, (c) BBR and (d) Mean Density. On all the graphs there were no excited cells left on the grid by time step 400 for $p_T = \{0.1, 0.2, 1\}$. The excited cells have a value of 2 on the grid and a Moore neighbourhood with a range of 3 was used, hence the prefix on the legends of 2M3.

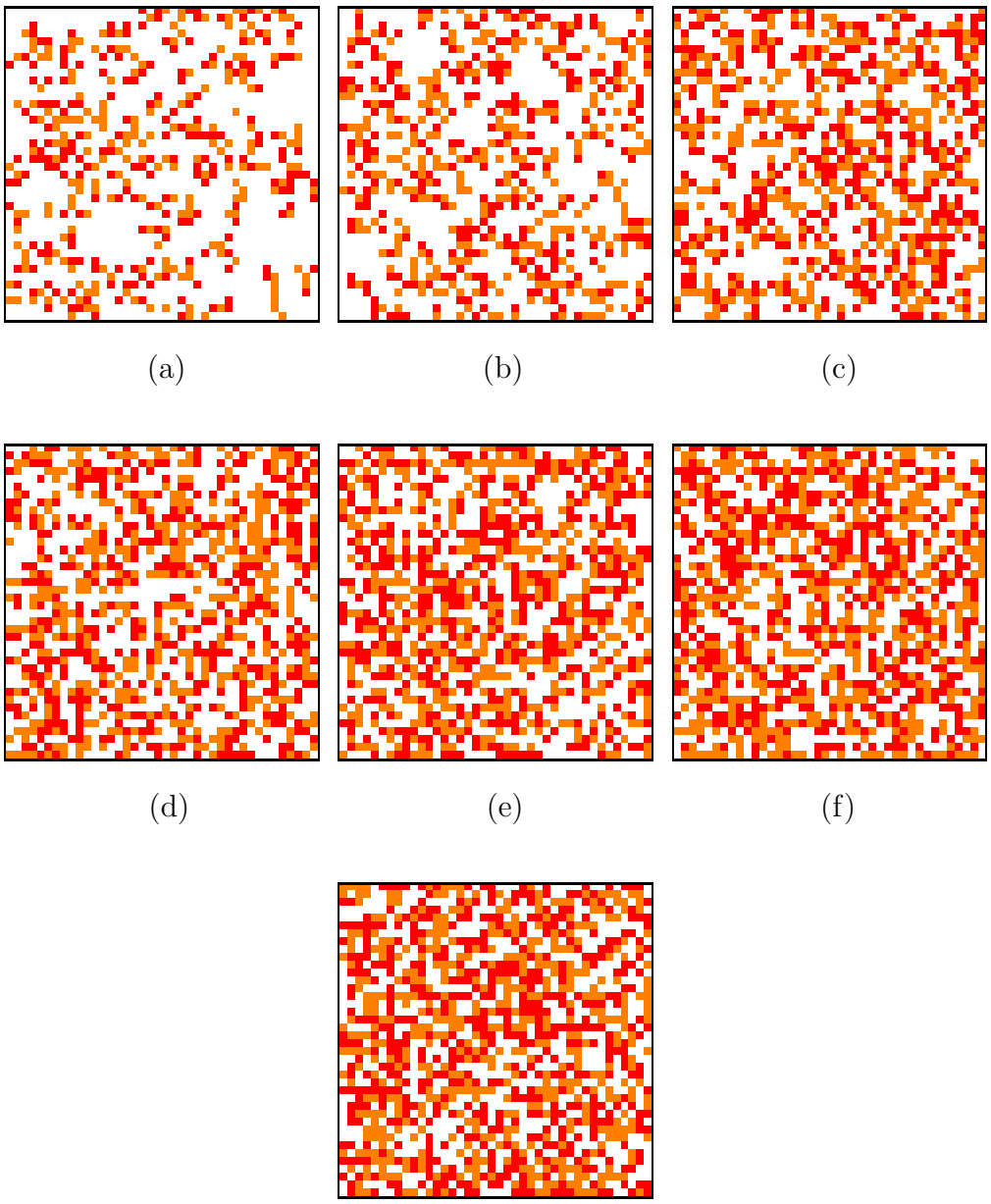


Figure 5.15: The last grid out of a run of 400 in an extinction regime simulation with a p_T set at (a) 0.3, (b) 0.4, (c) 0.5, (d) 0.6, (e) 0.7, (f) 0.8 and (g) 0.9; the graphs for p_T=$\{0.1, 0.2. 1.0\}$ are not shown as they do not contain any excited cells. The excited cells are red and are dispersed across the grids; the refractory cells are orange and the white cells are neutral. In (g) the randomness of the spread of excited cells can be seen, as well as a number of small groupings of cells with a diversity of sizes.

What is more unexpected is that the entropy metric has the same graphical shape as the C-Value and mean density metrics, indicating that the scattered population of excited cells has not reached percolation criticality where a large cluster has formed and the diversity of cluster sizes has started to decline [Tsang and Tsang, 1999]. This would seem to be confirmed by the fact that the C-Value peaks at 0.3335 which is near to the mean density high level of 0.3206, which roughly correlates to the start values on Figure 5.13(c); where the entropy value starts at just below 2.800, which is where it peaks in Figure 5.14(b). The start values are a measurement of the initial grid, which was randomly generated. It was noted earlier how the combined probability and dynamic test, (*see subsection 5.2.3*), showed the C-Value increasing as a cluster is formed up to a value above 0.9000, while it had run alongside the mean density in the probability test in subsection 5.2.1. The probability grids were randomly populated with active cells, suggesting a possible indication of the randomness of the active cells on a grid when the C-Value is aligned with the mean density and below 0.9000. Therefore, a C-Value of 0.3335 and a mean density of 0.3206 can be seen as symptomatic of the excited cells being spread randomly across the grid in this test. This, in turn, would increase the possibility of there being a number of clusters of diverse size on the grid, confirming the entropy results. Figure 5.15 shows the state of the final grids from one of the five simulations, which was used in the calculation of the metrics. As can be seen, the excited (red) cells in Figure 5.15(g) are spread across the grid without any discernible order and a number of small groups of cells of varying sizes can be seen.

The second series of tests of the particle automata looks at the decentralised gathering of the amoebae. A collection of 40 by 40 grids was created with 600 amoebae and 222 blocked cells, all randomly placed on the initial grids. The blocking cells were grouped as horizontal and vertical barriers across the grid. The probability settings were $(p_T, p_E, p_A) = (1, 0.01, 0)$ and the simulation was run for 2000 time steps, although the amoebae had gathered into one cluster by time step 650 (*see Figure 5.16*). The table with the values used can be seen in appendix B.28.

The analysis of the first 650 time steps can be seen in Figure 5.17. The decrease in the value of the mean density reflects the fact that a cell can accommodate more than one amoeba. The mean density reacts as if the number of active agents has reduced, or is fluctuating. The other three metrics also base their calculation on the number of occupied cells, rather than the number of active agents. Therefore, any apparent reduction of active agents will lead to the occupied area becoming

more compact, thus having a slight affect on the BBR. Likewise the C-Value calculates the connectedness of the occupied cells and the maximum connectedness is based on the number of occupied cells. But as suggested, any variance should be slight and, in some measure, experienced by all the metrics.

Indeed, the results shown in Figure 5.17 are comparable to those from the dynamic scenario tests, (*see subsection 5.2.2*). The mean density and C-Value start from around the same point and then the C-Value increases as the amoebae gather together. The entropy in Figure 5.17(b) also reflects the peak and descent of latter part of the graphs seen in the probability scenario test (*see Figure 5.2*). This is in line with the fact that the 40 by 40 grid has 1600 cells, which becomes 1378 available cells when the 222 blocked cells are taken into account. This means that the 600 amoebae randomly placed on the cell represents roughly 43.5%, or a probability value of 0.435; the graph for the combined probability and dynamic scenario using a probability value of 0.4 on the initial grid shows the same shape for the entropy values, (*see Figure 5.13(d)*). The BBR and C-Values, in Figure 5.17(a) & (c) respectively, both show the same graph shape and orientation as was seen in the dynamic scenario, (*see Figure 5.11*).

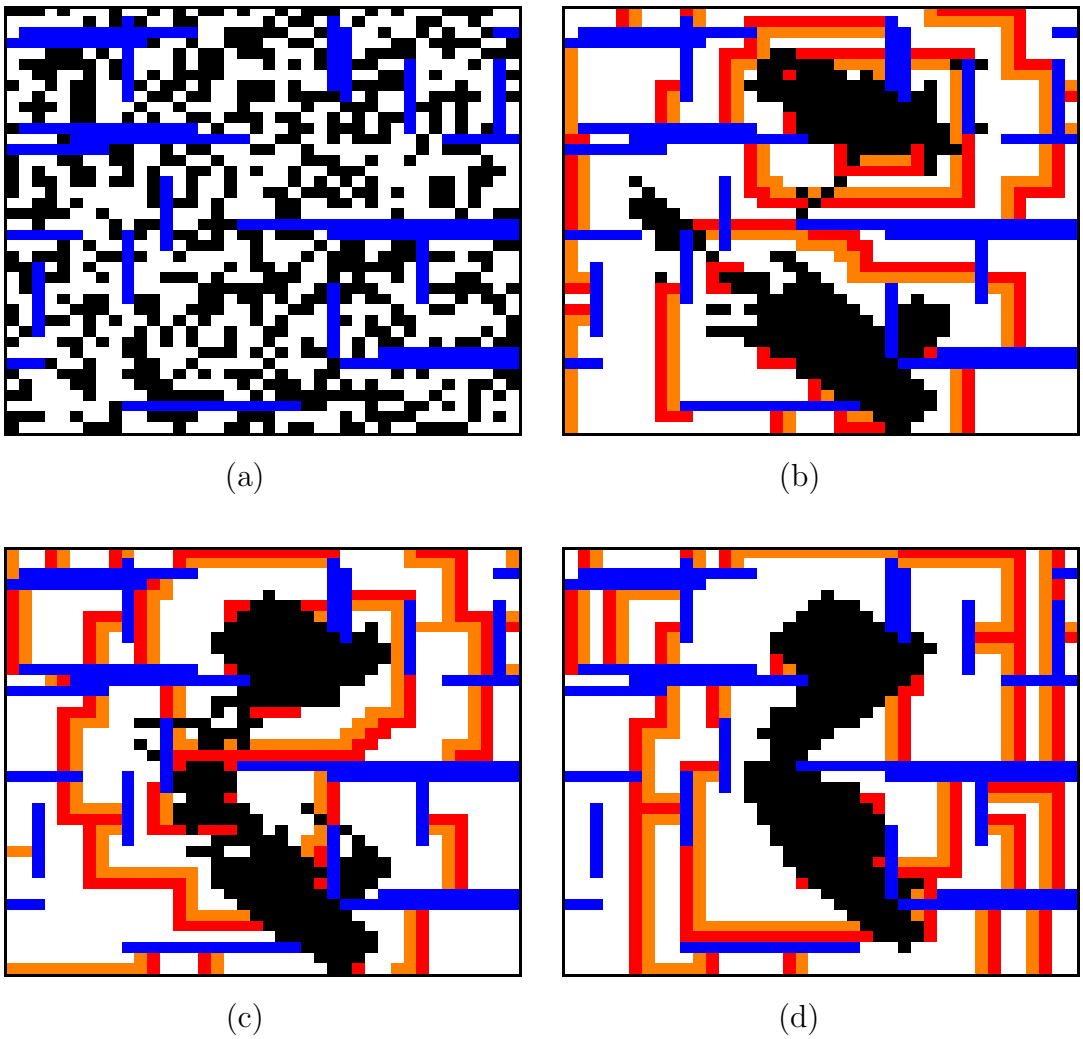


Figure 5.16: Amoebae and decentralised gathering with barriers. This shows (a) the initial grid and time steps (b) 250, (c) 450 and (d) 650 of one of the simulations using a 40 by 40 grid with 600 amoebae (black cells) and a total of 222 blocked cells (blue) and probability settings of $(p_T, p_E, p_A) = (1, 0.01, 0)$. The resulting reaction-diffusion waves are shown as red for excited cells, orange for refractory cells and the white cells are neutral. The run was for 2000 time steps, but gathering had been achieved by 650 time steps.

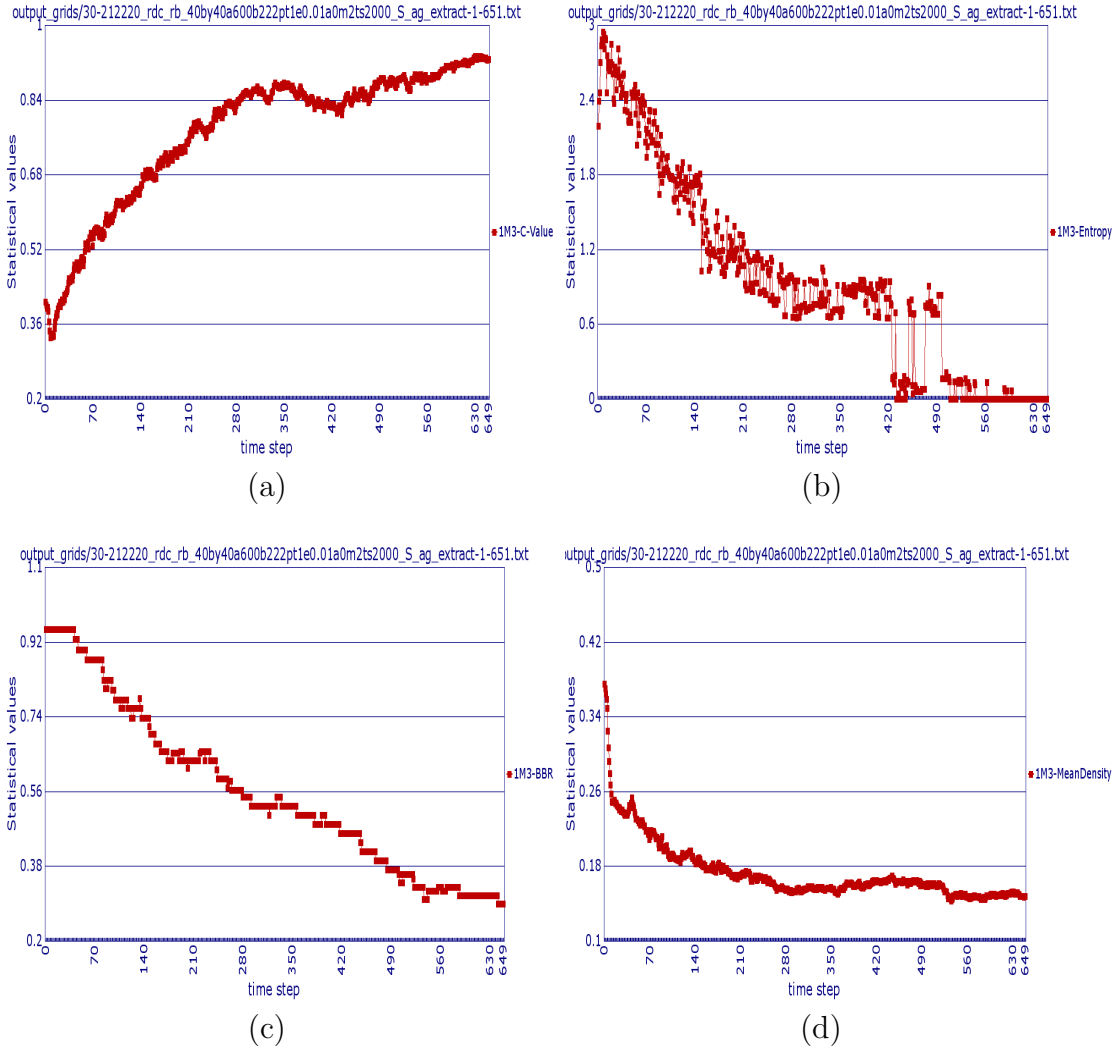
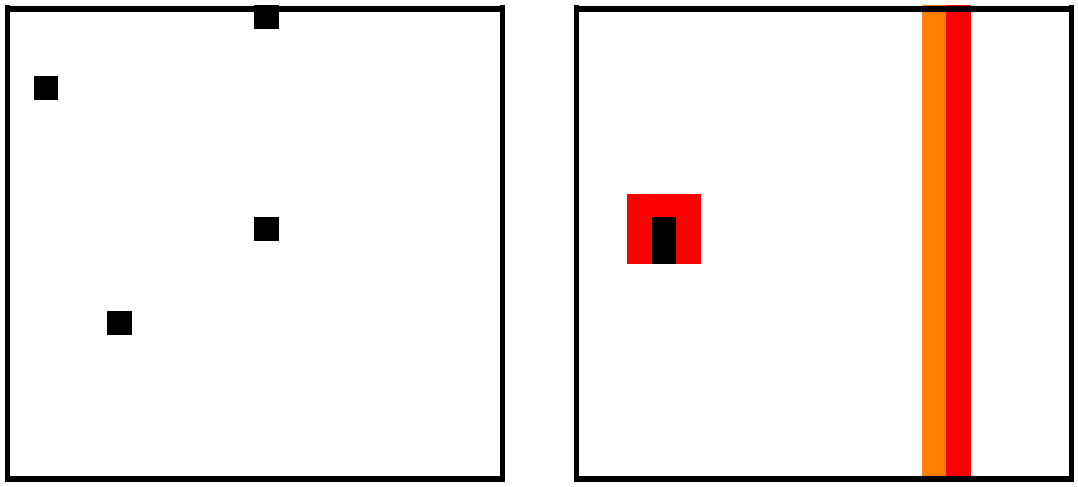


Figure 5.17: Results from the first 650 time steps of the simulation of amoebae and decentralised gathering with barriers. The simulation used a 40 by 40 grid with 600 amoebae and a total of 222 blocked cells and probability settings of $(p_T, p_E, p_A) = (1, 0.01, 0)$. The run was for 2000 time steps, but gathering had been achieved by 650 time steps. The graphs show the results for (a) C-Value, (b) Entropy, (c) BBR and (d) Mean Density. The variance of the mean density graphs is a result of more than a single amoeba being able to occupy a cell. The prefix to the legends of 1M3 signify the value of the cells being measured (1) and a Moore neighbourhood (M) with a range of 3 (3).

A final test was run to see if the reactive-diffusion model collaborated the findings of the scenarios when tracking a low and a high population of active agents (*see subsection 5.2.2 and subsection 5.2.3*). The relevant tables can be found in appendix B.2.

The low population test was run with four agents on a 20 by 20 grid with no obstructions, representing a probability scenario of 0.01. The probability settings were conducive to a 'quick' gathering, $(p_T, p_E, p_A) = (1, 0.01, 0)$, (*see Table 4.2*). The test was set for 1000 time steps, but the 4 amoebae had gathered after 290. The initial and final grid can be seen in Figure 5.18. The results can be found in Figure 5.19. They are similar to those seen in the 4 active agents dynamic scenario test in Figure 5.8(c). The C-Value and BBR values both register changes in the state of the grid. Although the C-Value registers that the four agents gather together, the process is not seen as a smooth transition, but one of sharp rises, followed by a period of inactivity. The dynamic scenario used a relatively simple algorithm to gather the active agents into the centre of the grid, leading to a smoother curve for the BBR than seen in Figure 5.19(c).



(a) (b)

Figure 5.18: Time steps (a) 0 and (b) 290 from a run of the 4 agents gathering together over 1000 time steps. The simulation used a 20 by 20 grid and probability settings of $(p_T, p_E, p_A) = (1, 0.01, 0)$. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as black cells and the white cells are neutral.

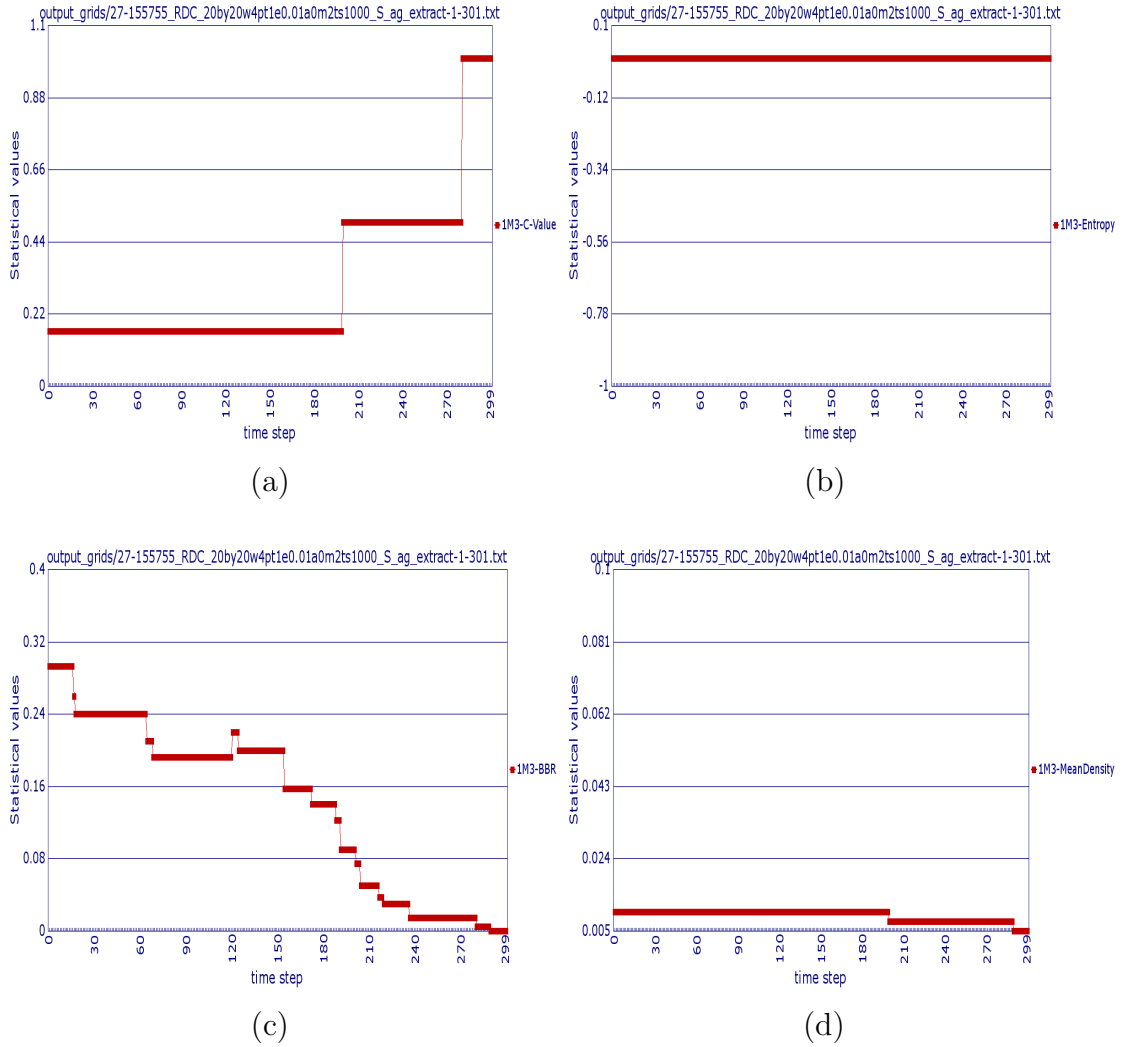
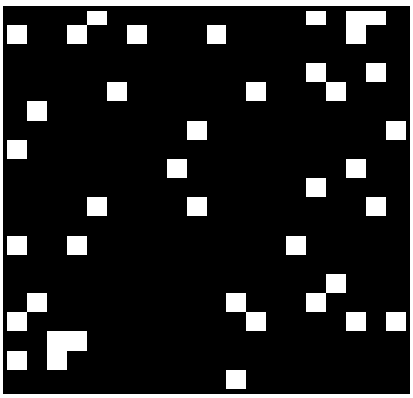
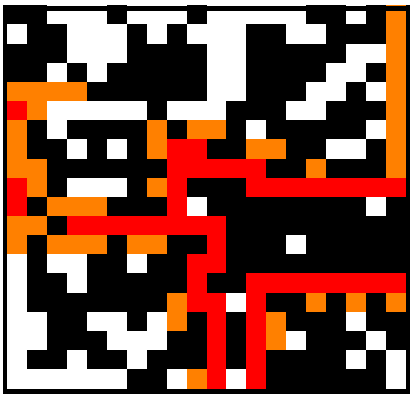


Figure 5.19: Results from the simulation of 4 agents gathering together over 300 time steps. The simulation used a 20 by 20 grid and probability settings of $(p_T, p_E, p_A) = (1, 0.01, 0)$. The graphs show the results for (a) C-Value, (b) Entropy, (c) BBR and (d) Mean Density. The prefix to the legends of 1M3 signify the value of the cells being measured (1) and a Moore neighbourhood (M) with a range of 3 (3).

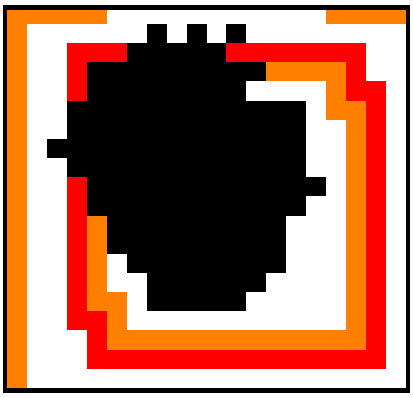
The high population test used the same probability settings, but with 360 amoebae and equivalent to a probability scenario with a *p_value* of 0.9. The amoebae had gathered together after 250 time steps. The initial and final grids can be seen in Figure 5.20. The results are displayed in Figure 5.21. Once again the C-Value and BBR provide the best information on the state of the grid over the 250 time steps, although the C-Value gives a clearer distinction in the gathering induced by the reactive-diffusion and chematoxis process. The initial decrease in the C-Value and mean density graphs is due to the number of occupied cells decreasing as amoebae start to share cells on the grid. Figure 5.20 shows how the number of unoccupied cells has increased from the initial grid.



(a)



(b)



(c)

Figure 5.20: Time steps (a) 0 and (b) 6 and (c) 250 from a run of the 360 agents gathering together over 1000 time steps. The simulation used a 20 by 20 grid and probability settings of $(p_T, p_E, p_A) = (1, 0.01, 0)$. The resulting reaction-diffusion waves are shown as red for excited cells, and orange for refractory cells. The amoebae are shown as black cells and the white cells are unoccupied. The effect of more than one amoeba being able to occupy a cell is shown by the increase in unoccupied cells in (b) and (c), compared to (a). The change between (a) and (b) can be seen in the initial decrease in the C-Value and mean density values seen in Figure 5.21(a) and (d) respectively.

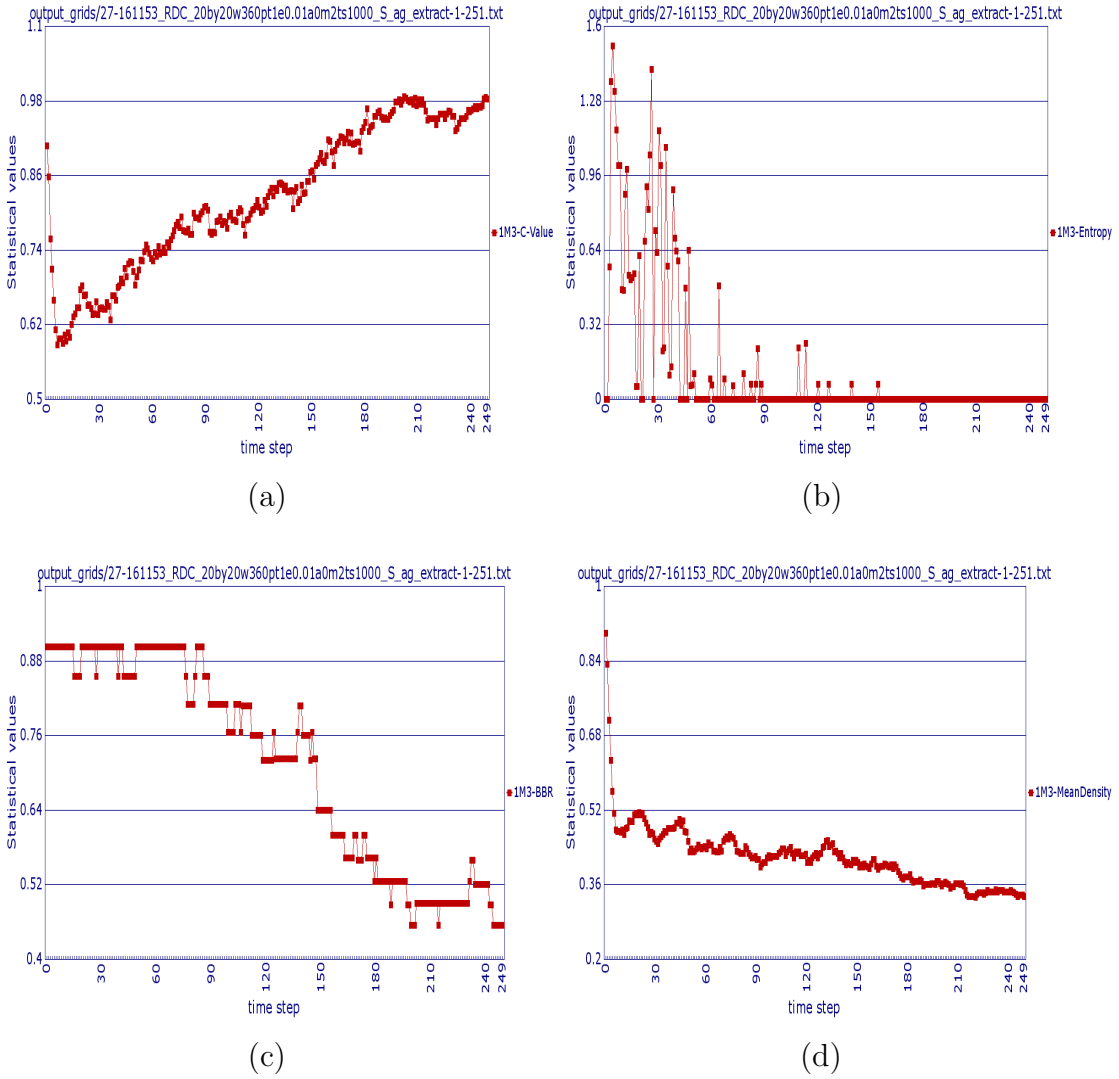


Figure 5.21: Results from a run of 360 agents gathering together over 250 time steps. The simulation used a 20 by 20 grid and probability settings of $(p_T, p_E, p_A) = (1, 0.01, 0)$. The graphs show the results for (a) C-Value, (b) Entropy, (c) BBR and (d) Mean Density. The initial time steps for the C-Value in (a) and the mean density in (d) show a sharp decrease before increasing in the expected manner. This reflects the impact of more than one amoebae being able to occupy a cell. The fluctuation continues at a much more reduced level after time step seven. It affects the smoothness of the C-value graph, but it does not alter the overall steady incline. The prefix to the legends of 1M3 signify the value of the cells being measured (1) and a Moore neighbourhood (M) with a range of 3 (3).

5.4 Randomised model

Two tests were run with the randomised model using two different asynchronous updating methods (*see subsection 2.6.3 and subsection 4.7.1*). The first looked at the grids produced when the delta-notch model was run with no noise and used a random order asynchronous (ROA) updating method. The output was measured with all three range settings, R={1, 2, 3}, allowing a comparison to be made. The second test looked at the results of noise being introduced into the delta-notch model and employed a random selection asynchronous (RSA) updating method.

The static array of signalling cells produced by the delta-notch model with no noise was shown in subsection 4.7.5. This static state is reached from an initial grid with no signalling cells on it and after only a couple of time steps. Table 5.6 shows the connectedness values when twenty of the static grids were measured three times each, one for each of the possible search range settings, R={1, 2, 3}.

Table 5.6: The measurement of 20 samples of 8 by 8 hexagonal grids with no noise for a range of 1, 2 and 3. The ID is made up of the sample reference followed by a block of, for example, 1HA1, which illustrate the value indicating a signalling cell (1), an hexagonal grid (H) using a ROA (A) updating method and a range of 1. The last grid of each sample in a run of 20 time steps was measured. The active cells are produced from an initial grid of inactive cells after 1 or 2 time steps and then the disorderly pattern becomes static. As was found seen in subsection 4.7.5, the absence of any noise means that the order of the signalling cells is generally not as compact or pervasive as it could be. Although sample 16 has a C-Value of one for ranges 2 and 3, implying that the 41 connections shared by the seventeen signalling cells is the maximum achievable for that number of cells. A threshold of one was used.

ID	C-Value	Entropy	BBR	MeanDensity
1 1HA1	0.0000	0.0000	0.7656	0.2344
1 1HA2	0.7381	0.0000	0.7656	0.2344
1 1HA3	0.7381	0.0000	0.7656	0.2344
2 1HA1	0.0000	0.0000	0.6562	0.2188
2 1HA2	0.7027	0.0000	0.6562	0.2188
2 1HA3	0.7027	0.0000	0.6562	0.2188
3 1HA1	0.0000	0.0000	0.7656	0.2344
3 1HA2	0.6429	0.0000	0.7656	0.2344
3 1HA3	0.6429	0.0000	0.7656	0.2344
4 1HA1	0.0000	0.0000	0.7656	0.2344

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
4 1HA2	0.7143	0.0000	0.7656	0.2344
4 1HA3	0.7143	0.0000	0.7656	0.2344
5 1HA1	0.0000	0.0000	0.7656	0.2344
5 1HA2	0.6905	0.0000	0.7656	0.2344
5 1HA3	0.6905	0.0000	0.7656	0.2344
6 1HA1	0.0000	0.0000	0.7656	0.2031
6 1HA2	0.6250	0.0000	0.7656	0.2031
6 1HA3	0.6250	0.0000	0.7656	0.2031
7 1HA1	0.0000	0.0000	0.7656	0.2188
7 1HA2	0.6486	0.0000	0.7656	0.2188
7 1HA3	0.6486	0.0000	0.7656	0.2188
8 1HA1	0.0000	0.0000	0.7656	0.2344
8 1HA2	0.6667	0.0000	0.7656	0.2344
8 1HA3	0.6667	0.0000	0.7656	0.2344
9 1HA1	0.0000	0.0000	0.7656	0.2500
9 1HA2	0.6875	0.0000	0.7656	0.2500
9 1HA3	0.6875	0.0000	0.7656	0.2500
10 1HA1	0.0000	0.0000	0.7656	0.2344
10 1HA2	0.6667	0.0000	0.7656	0.2344
10 1HA3	0.6667	0.0000	0.7656	0.2344
11 1HA1	0.0000	0.0000	0.7656	0.2344
11 1HA2	0.6667	0.0000	0.7656	0.2344
11 1HA3	0.6667	0.0000	0.7656	0.2344
12 1HA1	0.0000	0.0000	0.7656	0.2188
12 1HA2	0.6486	0.0000	0.7656	0.2188
12 1HA3	0.6486	0.0000	0.7656	0.2188
13 1HA1	0.0000	0.0000	0.7656	0.2188
13 1HA2	0.5946	0.0000	0.7656	0.2188
13 1HA3	0.5946	0.0000	0.7656	0.2188
14 1HA1	0.0000	0.0000	0.6562	0.2500
14 1HA2	0.7083	0.0000	0.6562	0.2500
14 1HA3	0.7083	0.0000	0.6562	0.2500
15 1HA1	0.0000	0.0000	0.7656	0.2188
15 1HA2	0.6486	0.0000	0.7656	0.2188
15 1HA3	0.6486	0.0000	0.7656	0.2188
16 1HA1	0.0000	0.0000	0.7656	0.2656
16 1HA2	1.0000	0.0000	0.7656	0.2656

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
16 1HA3	1.0000	0.0000	0.7656	0.2656
17 1HA1	0.0000	0.0000	0.6562	0.2188
17 1HA2	0.6757	0.0000	0.6562	0.2188
17 1HA3	0.6757	0.0000	0.6562	0.2188
18 1HA1	0.0000	0.0000	0.7656	0.2344
18 1HA2	0.6667	0.0000	0.7656	0.2344
18 1HA3	0.6667	0.0000	0.7656	0.2344
19 1HA1	0.0000	0.0000	0.6562	0.2500
19 1HA2	0.6875	0.0000	0.6562	0.2500
19 1HA3	0.6875	0.0000	0.6562	0.2500
20 1HA1	0.0000	0.0000	0.6562	0.2031
20 1HA2	0.5625	0.0000	0.6562	0.2031
20 1HA3	0.5625	0.0000	0.6562	0.2031

The BBR and mean density measures do not rely on the range, so they do not vary over the three range settings. The fact that no signalling (black) cells are next to each other is a result of the threshold being set to 1, leading to a signalling cell being surrounded by an immediate neighbourhood of inhibited (white) cells, (*see Figure 5.25*). Therefore, a check for connectedness at range 1 yields nothing; whereas all the clusters of active cells are found with range 2, thus rendering the range 3 search redundant. It must be remembered that a setting of 2 does a depth 1 search first, and a range 3 search does a search at depth 1 and then 2 before doing a final sweep to find any unconnected nodes at depth 3. The results of all three range settings on the C-Value is shown in graphical form in Figure 5.22, where the values from a range of 1 are all zero, and range 2 and 3 have the same values. Figure 5.23 shows just the values returned for the metrics with the range set to 3. The entropy metric finds nothing to register. Inspection of the grids in Figure 5.25 indicate that the signalling cells are all grouped in a single cluster within a range of 2, with the occasional singletons on some of the grids, such as on Figure 5.25(6).

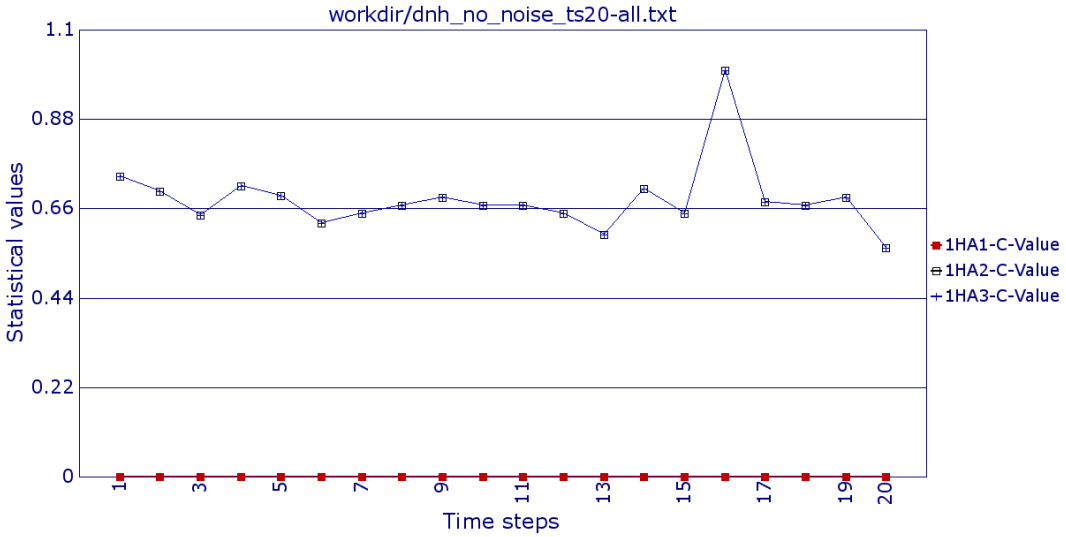


Figure 5.22: The C-Value connectedness measurement of 20 static grids on an 8 by 8 hexagonal grid with no noise using a search depth range of R={1, 2, 3}. In this example all of the clusters of signalling cells are grouped during the search at a range of 2. Consequently, 1HA1 (active value, Hexagonal grid using ROA updating, range 1) is at zero along the x axis as no connections were found, and ranges 2 (1HA2) and 3 (1HA3) are mapped on top of each other as all the connections are found within the search with range 2. The level of the C-Value for range 2 & 3 indicates that the signalling nodes on each grid are fairly well connected, with sample 16 peaking with a full C-Value of 1.

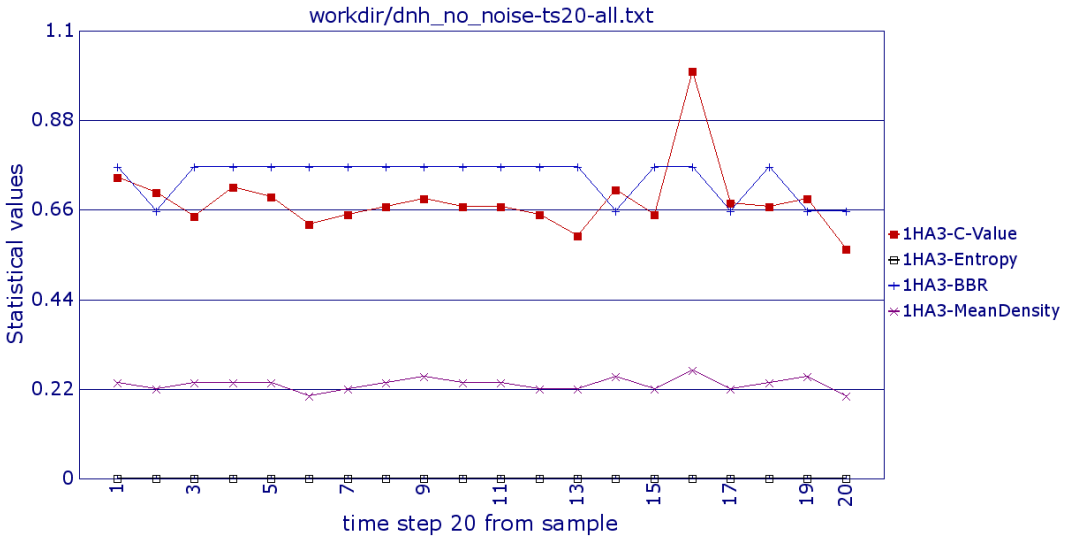


Figure 5.23: The four metrics of 20 static grids on an 8 by 8 hexagonal grid with no noise using a search range of 3 and a ROA updating method. The C-Value runs across the graph at a higher level than the mean density, indicating that the distribution of the signalling cells on each of the grids is not random, but are at some level of connectedness. The prefix to the legends of 1M3 signify the value of the cells being measured (1) and a Moore neighbourhood (M) with a range of 3 (3).

The twenty grids measured can be seen in Figure 5.25. Cohen et al. [2011, p.789] refer to "notably disordered packing of active cells" in the signalling cells produced by the delta-notch with no noise. The results visually do not look entirely random, and some order can be noted in some of the samples. The C-Value appears to bear this out if the correlation of C-Value with the mean density is seen as indicating a random state, as was suggested in the dynamic scenario and particle model tests. The mean density of the range 2 and 3 tests is between 0.2031 and 0.2656, while C-Value for nineteen of the samples is spread between 0.5625 and 0.7381, with one sample having a value of 1 (*see Figure 5.24*). This implies that nineteen of the samples have a medium level of connectedness, putting them above a random state, but below a highly ordered or grouped state.

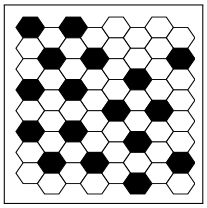


Figure 5.24: Sample 16, see text.

The 'perfect' C-Value of 1 indicates that the signalling cells have the maximum number of edges connected for that number of cells, range and neighbourhood. The C-Value of 1 is achieved by the last grid, time step 20, in sample 16, (*see Figure 5.25(16)*). The grid has 17 signalling cells and the connected sub grid in Figure 5.24 shows more clearly how the cells gather together in a cluster with the range set to 2; visually the connections are between each signalling cell (black) separated by a single non-signalling cell (white). Full toroidal wrap around is used yielding 41 connections. It is not entirely evident if wrap around toroidal boundary conditions are in full use in some of the diagrams in the original work [Cohen et al., 2011, pp.789, 791 and 794], which might account for any difference in their view of how packed the signalling cells are.

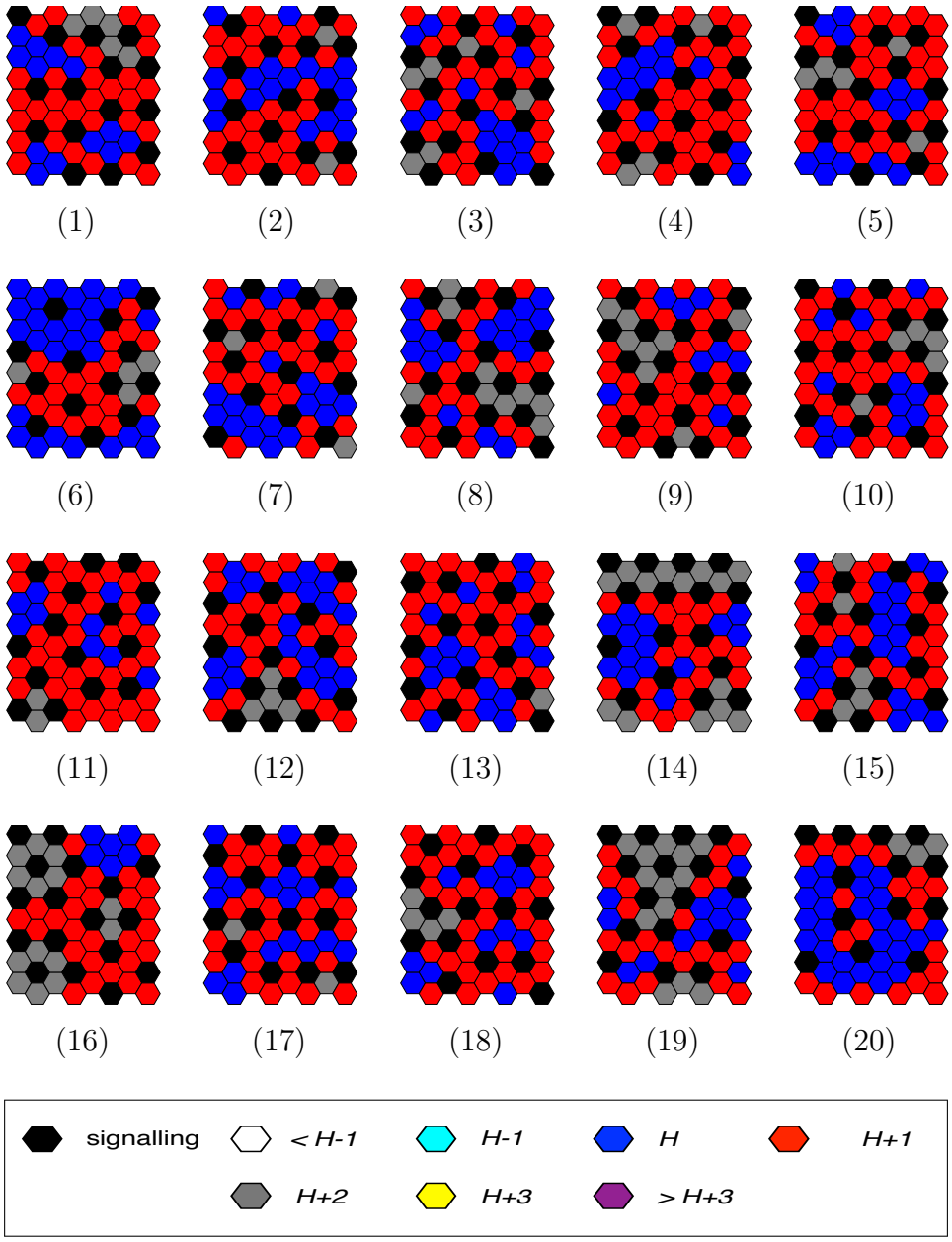


Figure 5.25: 20 hexagonal 8 by 8 grids produced with no noise and using a ROA updating system. The initial grid has only inactive cells on it; signalling cells become established within a couple of time steps. The last grid of a simulation of twenty time steps is shown. The absence of noise means that the grid is not completely populated with tightly packed signalling cells, but the use of a threshold of 1 means that each signalling cells is surrounded by an immediate neighbourhood of inactive cells; meaning that a range of 1 finds no connections. However, (16) has all 17 of its signalling cells with 41 connections in a single cluster and a C-Value of 1, indicating that they have the maximum number of connects possible in a wrap around toroidal environment. The signalling cells on the other grids are also found in clusters within a range of 2, with a few exceptions, such as the singleton in the top left quarter of (6).

The previous test looked at the identification of small changes in the state of the cells on a grid. This can be as challenging and problematic as trying to measure the movement of a very small number of agents. In Figure 5.23 the C-Value gives good information on the state of an individual grid, showing that there was a level of connectedness. The second test looks at two simulations that produce hexagonal grids that have different patterns of signalling cells on them. Both simulations were run for 200 time steps on a 20 by 20 hexagonal grid with a threshold of 4, a range of 2 and $N_t = 0$. The first has a spatial noise setting of $N_s = 0.1$, and the second $N_s = 0.9$, (*see Table 5.7 and Table 5.8 respectively*).

Table 5.7: Selected values from measurement of delta-notch model with noise. The simulation was run on a 20 by 20 hexagonal grid with a settings of threshold = 4, range = 3, $N_t = 0$ and $N_s = 0.1$ and using a RSA updating method. The ID is made of up the time step followed by a block of 1HR3, which illustrate the value indicating a signalling cell (1), an hexagonal grid (H) using a RSA updating method (R), and the range of 3 (3). The C-Value increases a little, but it does not rise much above the mean density, despite a very ordered pattern emerging, (*see Figure 5.26(a)*).

ID	C-Value	Entropy	BBR	MeanDensity
1 1HR3	0.2109	1.2821	0.9025	0.2475
2 1HR3	0.2007	0.8379	0.9025	0.2475
3 1HR3	0.2365	0.8548	0.9025	0.2600
4 1HR3	0.2359	0.5623	0.9025	0.2625
5 1HR3	0.2509	0.7658	0.9025	0.2575
6 1HR3	0.2591	0.8481	0.9025	0.2625
7 1HR3	0.2647	0.9704	0.9025	0.2650
8 1HR3	0.2612	0.8153	0.9025	0.2575
9 1HR3	0.2605	1.0794	0.9025	0.2675
10 1HR3	0.2549	0.7438	0.9025	0.2650
20 1HR3	0.2617	0.9672	0.9025	0.2725
30 1HR3	0.2785	0.5456	0.9025	0.2700
40 1HR3	0.2710	0.6274	0.9025	0.2725
50 1HR3	0.2975	0.5469	0.9025	0.2800
60 1HR3	0.3115	0.6712	0.9025	0.2825
70 1HR3	0.3263	0.3365	0.9025	0.2875
80 1HR3	0.3323	0.6135	0.9025	0.2875
90 1HR3	0.3259	0.3812	0.9025	0.2800
100 1HR3	0.3333	0.4405	0.9025	0.2900
110 1HR3	0.3323	0.2954	0.9025	0.2875

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
120 1HR3	0.3353	0.1480	0.9025	0.2950
130 1HR3	0.3353	0.1480	0.9025	0.2950
140 1HR3	0.3371	0.0000	0.8550	0.3000
150 1HR3	0.3371	0.0000	0.8550	0.3000
160 1HR3	0.3371	0.0000	0.8550	0.3000
170 1HR3	0.3371	0.0000	0.8550	0.3000
180 1HR3	0.3371	0.0000	0.8550	0.3000
190 1HR3	0.3371	0.0000	0.8550	0.3000
191 1HR3	0.3371	0.0000	0.8550	0.3000
192 1HR3	0.3371	0.0000	0.8550	0.3000
193 1HR3	0.3371	0.0000	0.8550	0.3000
194 1HR3	0.3371	0.0000	0.8550	0.3000
195 1HR3	0.3371	0.2303	0.9025	0.3000
196 1HR3	0.3371	0.2303	0.9025	0.3000
197 1HR3	0.3343	0.2023	0.9025	0.2925
198 1HR3	0.3353	0.1480	0.9025	0.2950
199 1HR3	0.3353	0.1480	0.9025	0.2950
200 1HR3	0.3353	0.1480	0.9025	0.2950

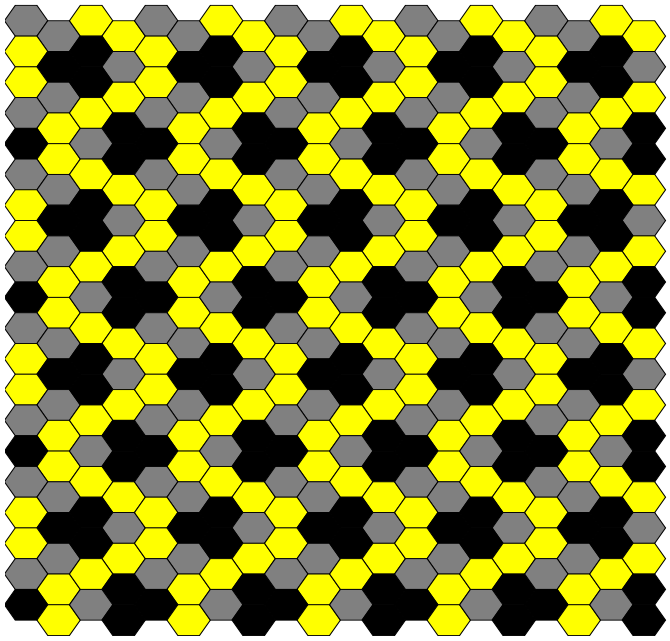
Table 5.8: Selected values from measurement of delta-notch model with noise. The simulation was run on a 20 by 20 hexagonal grid with a settings of threshold = 4, range = 3, $N_t = 0$ and $N_s = 0.9$. The ID is made of up the time step followed by a block of 1HR3, which illustrate the value indicating a signalling cell (1), an hexagonal grid (H) using a RSA updating method (R), and the range of 3 (3). The C-Value runs along side the mean density.

ID	C-Value	Entropy	BBR	MeanDensity
1 1HR3	0.4668	0.4495	0.9025	0.4675
2 1HR3	0.5546	0.2758	0.9025	0.5325
3 1HR3	0.4992	0.0000	0.9025	0.5125
4 1HR3	0.5156	0.1972	0.9025	0.5000
5 1HR3	0.5238	0.1461	0.9025	0.5250
6 1HR3	0.5333	0.1302	0.9025	0.5250
7 1HR3	0.5466	0.5817	0.9025	0.5425
8 1HR3	0.5237	0.1757	0.9025	0.5325
9 1HR3	0.5079	0.0000	0.9025	0.5250

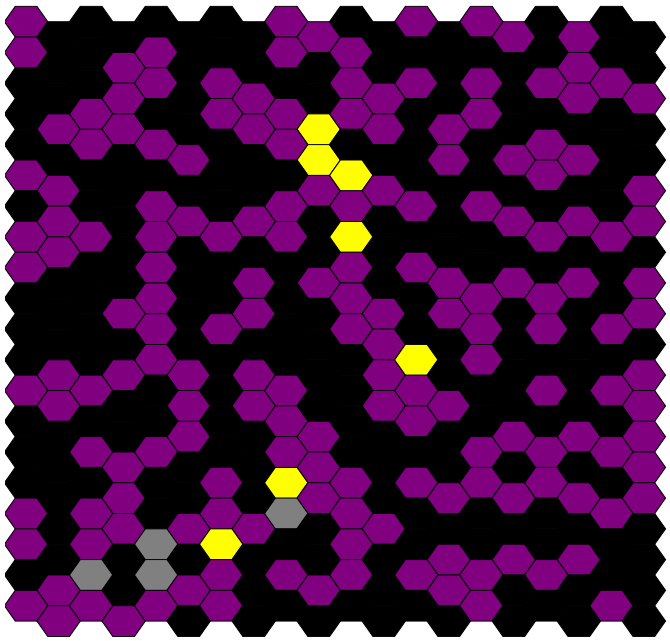
continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
10 1HR3	0.5441	0.0000	0.9025	0.5375
20 1HR3	0.5631	0.0000	0.9025	0.5350
30 1HR3	0.4939	0.4165	0.9025	0.4975
40 1HR3	0.5175	0.0000	0.9025	0.5250
50 1HR3	0.5026	1.2958	0.9025	0.5025
60 1HR3	0.4910	0.5766	0.9025	0.5175
70 1HR3	0.4629	0.5225	0.9025	0.4950
80 1HR3	0.5411	0.0538	0.9025	0.5300
90 1HR3	0.5191	0.0000	0.9025	0.5275
100 1HR3	0.4948	0.3228	0.9025	0.5000
110 1HR3	0.4939	0.5004	0.9025	0.4975
120 1HR3	0.5385	0.1156	0.9025	0.5175
130 1HR3	0.5237	0.2752	0.9025	0.5175
140 1HR3	0.5241	0.0915	0.9025	0.5475
150 1HR3	0.5457	0.0000	0.9025	0.5375
160 1HR3	0.5077	0.4598	0.9025	0.5025
170 1HR3	0.4908	0.0973	0.9025	0.5125
180 1HR3	0.4983	1.1506	0.9025	0.5050
190 1HR3	0.5482	0.0929	0.9025	0.5425
191 1HR3	0.5166	0.0000	0.9025	0.5425
192 1HR3	0.5137	0.2239	0.9025	0.5225
193 1HR3	0.5324	0.0000	0.9025	0.5350
194 1HR3	0.4923	0.7249	0.9025	0.5025
195 1HR3	0.5410	0.0000	0.9025	0.5550
196 1HR3	0.5324	0.1256	0.9025	0.5425
197 1HR3	0.5291	0.0936	0.9025	0.5350
198 1HR3	0.5267	1.1642	0.9025	0.5025
199 1HR3	0.5450	0.1624	0.9025	0.5425
200 1HR3	0.5345	0.0936	0.9025	0.5375

The spatial setting of $N_s = 0.1$ produces a symmetrical pattern of clusters consisting of three cells (time step 194 is shown on Figure 5.26(a)). Whereas the simulation with a setting of $N_s = 0.9$ evolves into a linked chain of clumped cells, spread across the grid (time step 200 is shown on Figure 5.26(b)). In Figure 5.26(a) all the black signalling cells are grouped in a neat pattern of forty clusters of three cell each. The clusters are spread in eight lines of five clusters. All the clusters are picked out with a range setting of 1.



(a)



(b)

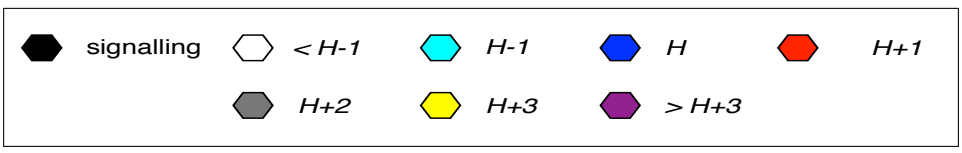


Figure 5.26: Time steps from two simulations run over 200 time steps with $H = 4$, range $= 2$ and $N_t = 0$. (a) shows the highly ordered pattern produced at time step 194 from the simulation with $N_s = 0.1$, and (b) is time step 200 from the run with $N_s = 0.9$. Despite the visual appearances, (b) has a higher C-Value than (a).

However, Figure 5.26(a) has a low C-Value, despite visually having a distinctive and highly ordered pattern of three cell clusters across the grid. Indeed, Figure 5.26(b), although being visually inelegant compared to the pattern of (a), scores a higher C-Value because the cells are more connected as a group. This is exasperated by the toroidal boundary condition which increases the potential maximum connections. A three cell cluster on an eight by eight grid will never wrap around and will only have a maximum of three connected edges, whereas the 120 signalling cells that make up the forty clusters of three cells have a much higher maximum number of connected edges, including via the toroidal boundary, leading to the low C-Value of the $N_s = 0.1$ simulation. The values from a selection of time steps for just the C-Values from both simulations can be seen in Table 5.9.

Table 5.9: A sample of the C-Value connectedness values for two simulations run over 200 time steps with $H = 4$, range $= 3$ and $N_t = 0$. The ID in the left column is made of up the time step followed by a block of 1HR3, which illustrate the value indicating a signalling cell (1), an hexagonal grid (H) using a RSA updating method (R), and the range of 3 (3). The middle column shows a selection of the simulated grids run with $N_s = 0.1$; the right column was run with $N_s = 0.9$. Although the simulation in the left column produced grids with patterns of small clusters the higher noise perturbation used by the simulation in the right column produced more active cells that, although a disorganised sprawl, generated a higher C-Value, (*see Figure 5.26*).

ID	$N_s = 0.1$	$N_s = 0.9$
1 1HR3	0.2109	0.4668
2 1HR3	0.2007	0.5546
3 1HR3	0.2365	0.4992
4 1HR3	0.2359	0.5156
5 1HR3	0.2509	0.5238
6 1HR3	0.2591	0.5333
7 1HR3	0.2647	0.5466
8 1HR3	0.2612	0.5237
9 1HR3	0.2605	0.5079
10 1HR3	0.2549	0.5441
20 1HR3	0.2617	0.5631
30 1HR3	0.2785	0.4939
40 1HR3	0.2710	0.5175
50 1HR3	0.2975	0.5026
60 1HR3	0.3115	0.4910
70 1HR3	0.3263	0.4629

continued on next page

ID	$N_s = 0.1$	$N_s = 0.9$
80 1HR3	0.3323	0.5411
90 1HR3	0.3259	0.5191
100 1HR3	0.3333	0.4948
110 1HR3	0.3323	0.4939
120 1HR3	0.3353	0.5385
130 1HR3	0.3353	0.5237
140 1HR3	0.3371	0.5241
150 1HR3	0.3371	0.5457
160 1HR3	0.3371	0.5077
170 1HR3	0.3371	0.4908
180 1HR3	0.3371	0.4983
190 1HR3	0.3371	0.5482
191 1HR3	0.3371	0.5166
192 1HR3	0.3371	0.5137
193 1HR3	0.3371	0.5324
194 1HR3	0.3371	0.4923
195 1HR3	0.3371	0.5410
196 1HR3	0.3371	0.5324
197 1HR3	0.3343	0.5291
198 1HR3	0.3353	0.5267
199 1HR3	0.3353	0.5450
200 1HR3	0.3353	0.5345

The metrics from both the $N_s = 0.1$ and the $N_s = 0.9$ simulations can be seen in the graphs in Figure 5.27. In Figure 5.27(a) the C-Value of the $N_s = 0.1$ simulation does have the curve that indicates a gathering or ordering of the signalling cells, but the values registered do not indicate a highly connected cluster and they only rise marginally above the mean density shown in Figure 5.27(d). Certainly it does not support the move from randomness to a state of higher order that is portrayed in Figure 5.26(a). Thus by itself the C-Value does not identify the regular pattern displayed on the grid in Figure 5.26(a) because it has already identified the lower level connectivity of each individual three cell cluster.

What is also of great interest is the shape of the C-Value graph of the $N_s = 0.9$ simulation displayed in Figure 5.27(e). While the values are higher than in (a), indicating a higher level of connectedness, the shape of the graph, with its alternation between just above 0.4500 and below 0.6000, lacks any evidence of a

gathering or compression process. Indeed, the values of the C-Value and mean density in Table 5.8 have similar fluctuations, indicating that although there is a large, sprawling cluster, it is random in its composition and not highly structured.

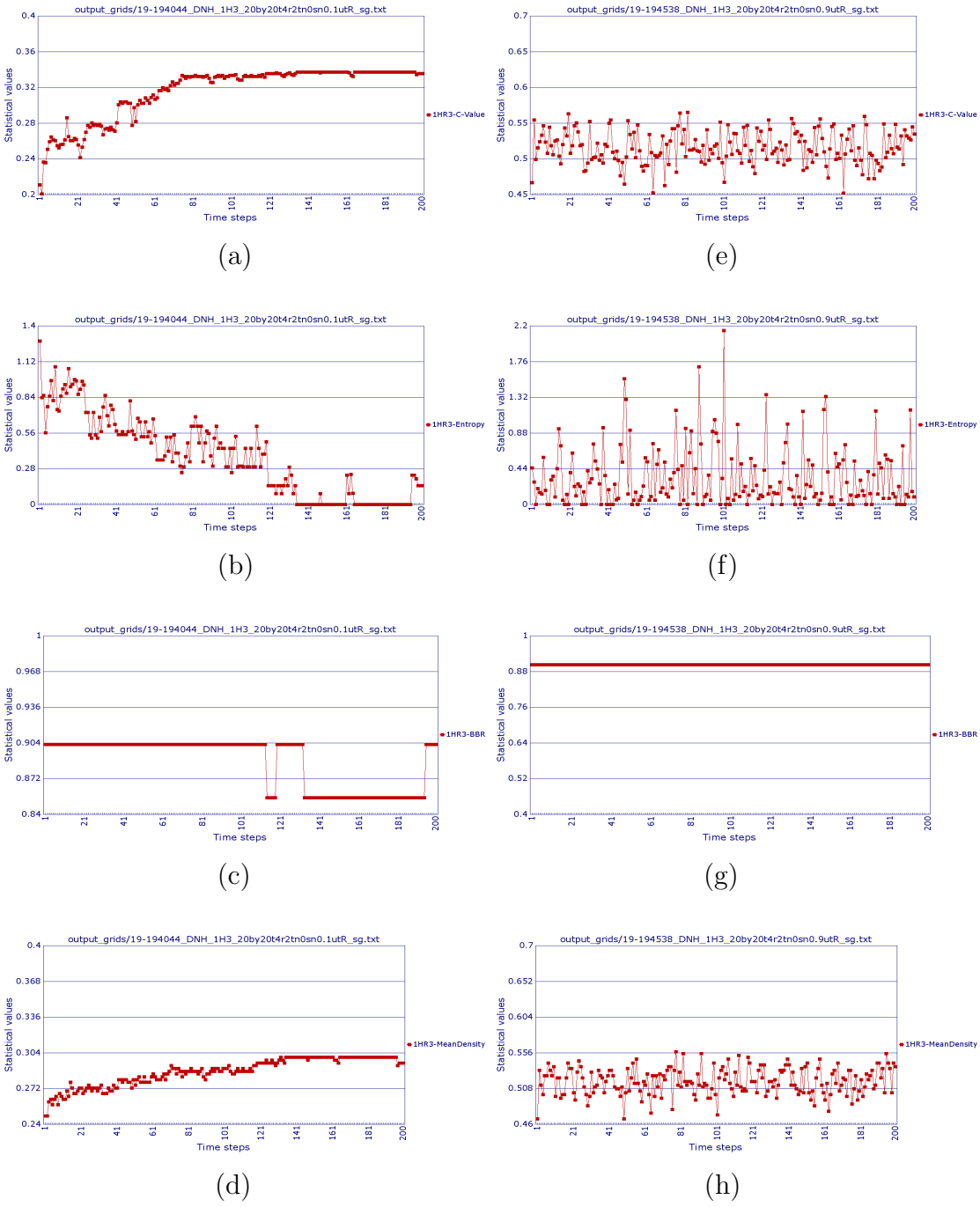


Figure 5.27: Analysis of two simulations run over 200 time steps with $H = 4$, range $= 3$, $N_t = 0$, but different spatial noise settings. (a)-(d) were run with $N_s = 0.1$; (e)-(h) were run with $N_s = 0.9$. (a) & (e) show the C-Value, (b) & (f) the BBR, (c) & (g) the entropy and (d) & (h) the mean density. The prefix to the legends of 1M3 signify the value of the cells being measured (1) and a Moore neighbourhood (M) with a range of 3 (3). See text for further information.

5.4.1 Identifying patterns of clusters in the randomised CA model

The problem the C-Value has in recognising the distinctive pattern of clusters in Figure 5.26(a) indicates how the level at which the metric is calculated is geared more towards the identification of the extent to which the active or occupied cells on a grid are gathered together in a compact, single cluster. A possible solution is to set the process at a lower level in an attempt to identified patterns of clusters. This involves calculating the C-Value for each cluster identified. The C-Value for the grid is then the sum of the localised C-Values divided by the number of clusters. A singleton is treated as a cluster with a zero C-Value.

The results of calculating the localised C-Value (LC-Value) can be seen alongside the C-Values evaluated for the two randomised simulations with noise from the previous section:

1. the simulation with noise set at $N_s = 0.1$, $N_t = 0$, threshold = 4, range = 3 and a RSA updating method; and
2. the simulation with noise set at $N_s = 0.9$, $N_t = 0$, threshold = 4, range = 3 and a RSA updating method.

The table listing the C-Value and LC-Value for selected time steps from both simulations with noise can be seen in Table 5.10. The LC-Value for the simulation using $N_s = 0.1$ captures the highly ordered pattern that is spread across the grid, as seen in Figure 5.26(a), with a LC-Value of one being achieved by most grids after time step eighty. This is in contrast to the C-Value which only picked up a slight rise in the connectedness of the signalling cells and did not recognise the move from an empty initial grid to a highly ordered one because the pattern is made up of small, evenly dispersed clusters of three cells.

Table 5.10: Comparison of the C-Value and the Localised C-Value from selected time steps of the simulations using a spatial noise setting of $N_s = 0.1$ and $N_s = 0.9$, (*see Table 5.7 and Table 5.8 respectively for tables of the original simulations*). The LC-Value of the $N_s = 0.1$ section reflects the highly ordered, dispersed pattern that can be seen in Figure 5.26(b)). The ID starts with the time step, then 1HR3 which represents the value identifying a signalling cell (1), a hexagonal grid (H) using a RSA (R) updating method and a range of 3. A threshold of 4 was used with both simulations with a full toroidal wrap around boundary and no temporal noise, $N_t = 0$.

ID	0.1 C-Value	0.1 LC-Value	0.9 C-Value	0.9 LC-Value
1 1HR3	0.2109	0.6826	0.4668	0.4441
2 1HR3	0.2007	0.7667	0.5546	0.6168
3 1HR3	0.2365	0.8000	0.4992	0.2132
4 1HR3	0.2359	0.7037	0.5156	0.5771
5 1HR3	0.2509	0.7500	0.5238	0.7984
6 1HR3	0.2591	0.7969	0.5333	0.5657
7 1HR3	0.2647	0.8206	0.5466	0.8373
8 1HR3	0.2612	0.7935	0.5237	0.4294
9 1HR3	0.2605	0.7683	0.5079	0.6598
10 1HR3	0.2549	0.8124	0.5441	0.3424
20 1HR3	0.2617	0.7725	0.5631	0.2367
30 1HR3	0.2785	0.8485	0.4939	0.6652
40 1HR3	0.2710	0.9167	0.5175	0.1705
50 1HR3	0.2975	0.9302	0.5026	0.7943
60 1HR3	0.3115	0.8682	0.4910	0.5273
70 1HR3	0.3263	0.9756	0.4629	0.5562
80 1HR3	0.3323	1.0000	0.5411	0.4205
90 1HR3	0.3259	0.9512	0.5191	0.3227
100 1HR3	0.3333	1.0000	0.4948	0.5661
110 1HR3	0.3323	1.0000	0.4939	0.6078
120 1HR3	0.3353	1.0000	0.5385	0.2607
130 1HR3	0.3353	1.0000	0.5237	0.7367
140 1HR3	0.3371	1.0000	0.5241	0.5569
150 1HR3	0.3371	1.0000	0.5457	0.1735
160 1HR3	0.3371	1.0000	0.5077	0.3553
170 1HR3	0.3371	1.0000	0.4908	0.2478
180 1HR3	0.3371	1.0000	0.4983	0.7152
190 1HR3	0.3371	1.0000	0.5482	0.3014

continued on next page

ID	0.1 C-Value	0.1 LC-Value	0.9 C-Value	0.9 LC-Value
191 1HR3	0.3371	1.0000	0.5166	0.1329
192 1HR3	0.3371	1.0000	0.5137	0.3494
193 1HR3	0.3371	1.0000	0.5324	0.3343
194 1HR3	0.3371	1.0000	0.4923	0.4998
195 1HR3	0.3371	1.0000	0.5410	0.2331
196 1HR3	0.3371	1.0000	0.5324	0.6706
197 1HR3	0.3343	1.0000	0.5291	0.3679
198 1HR3	0.3353	1.0000	0.5267	0.5625
199 1HR3	0.3353	1.0000	0.5450	0.6392
200 1HR3	0.3353	1.0000	0.5345	0.4480

The graphs of the C-Value and LC-Value for both simulations can be seen in Figure 5.28. The shape of the graphs are maintained for both $N_s = 0.1$ and $N_s = 0.9$. Although in the former the values for LC-value are much higher, indicating the ordered structure of the pattern. The graph of the LC-Value for $N_s = 0.9$ in Figure 5.28(d) has the same fluctuating pattern of the C-Value one in (c).

However, the range of the LC-Value results for $N_s = 0.9$ is much greater, spreading from 0.1329 (time step 191) to 0.8826 (time step 109), whereas the C-Value range is 0.4520 (time step 163) to 0.5653 (time step 82) (*see appendix B.3.1 for the full tables*). Each grid in the $N_s = 0.9$ simulation has one main connected sub grid that groups almost all of the signalling cells. The wide fluctuation of the LC-Value is where the there are some signalling cells that are not in the main group. These either form very small clusters with a high or maximum C-Value, or are singletons with a C-Value of zero. The former leads to a higher overall LC-Value, such as with time step 109, which has a main connected sub grid with a C-Value of 0.6478 and two other sub grids both with a two cell cluster on and each having a C-Value of 1; this leads to the LC-Value of 0.8826. In contrast, time step 191 has a main connected sub grid with a C-Value of 0.6646 and four singletons, giving a LC-Value of 0.1329 (*see Figure 5.29*).

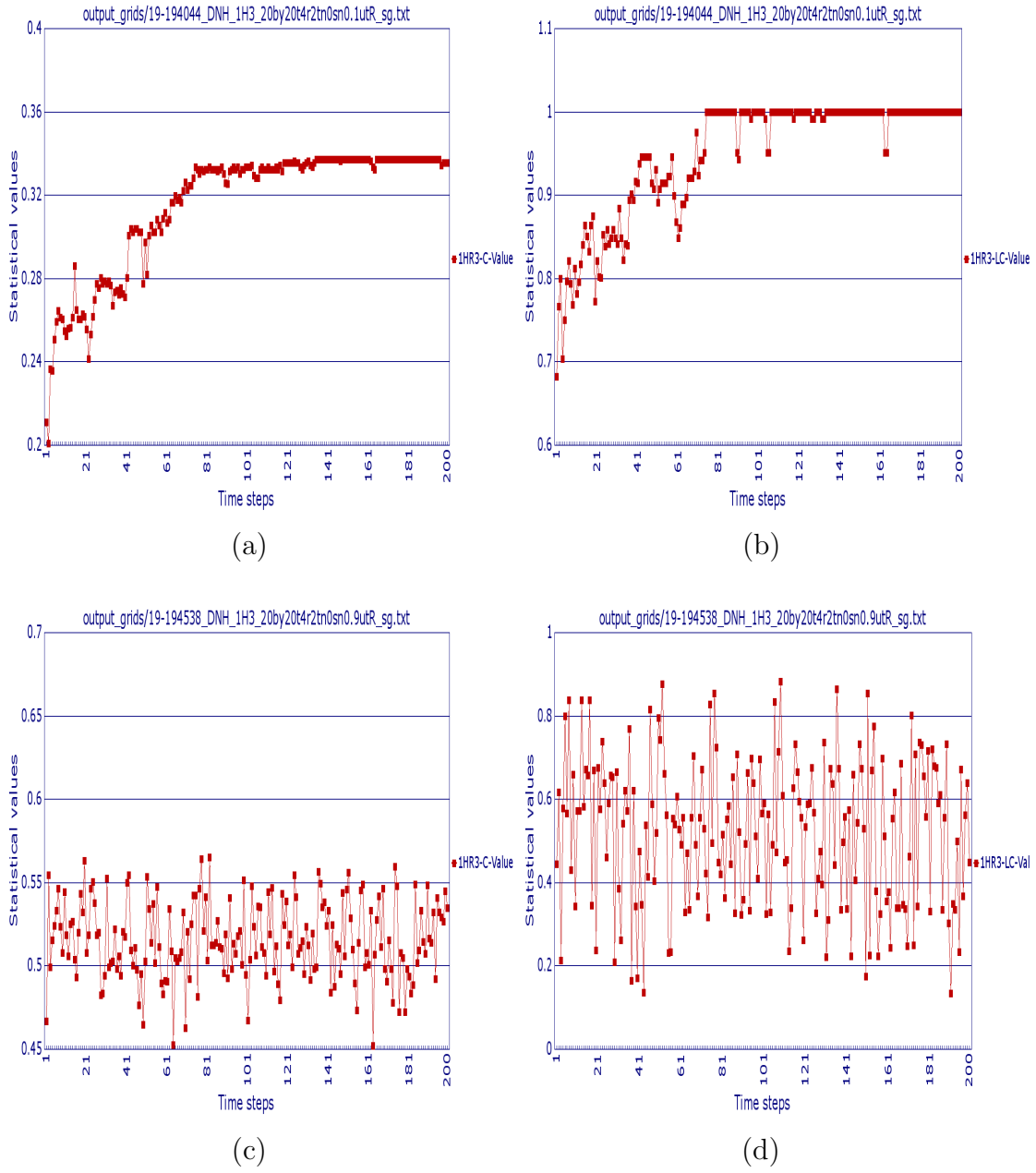


Figure 5.28: Comparison of C-Value and LC-Value for delta-notch simulations with noise. (a) C-Value for $N_s = 0.1$, (b) LC-Value for $N_s = 0.1$, (c) C-Value for $N_s = 0.9$ and (d) LC-Value for for $N_s = 0.9$. Both simulations used additional settings of threshold = 4, range = 3, toroidal boundary and a RSA updating scheme. The shape of the graphs are maintained, with (a) and (b) showing the increase in the connectedness of the signalling cells on the grid, although only (b) indicates how ordered the cells become on the grid. Both (c) and (d) indicate how the connectedness fluctuates with a setting of $N_s = 0.9$, implying that the cluster has a random distribution across the grid. The LC-Value shown in (d) also has a much wider range of values, which is brought on by too much weight being given to the few signalling cells that do not form part of the main grouping of cells. The prefix to the legends of 1M3 signify the value of the cells being measured (1) and a Moore neighbourhood (M) with a range of 3 (3).

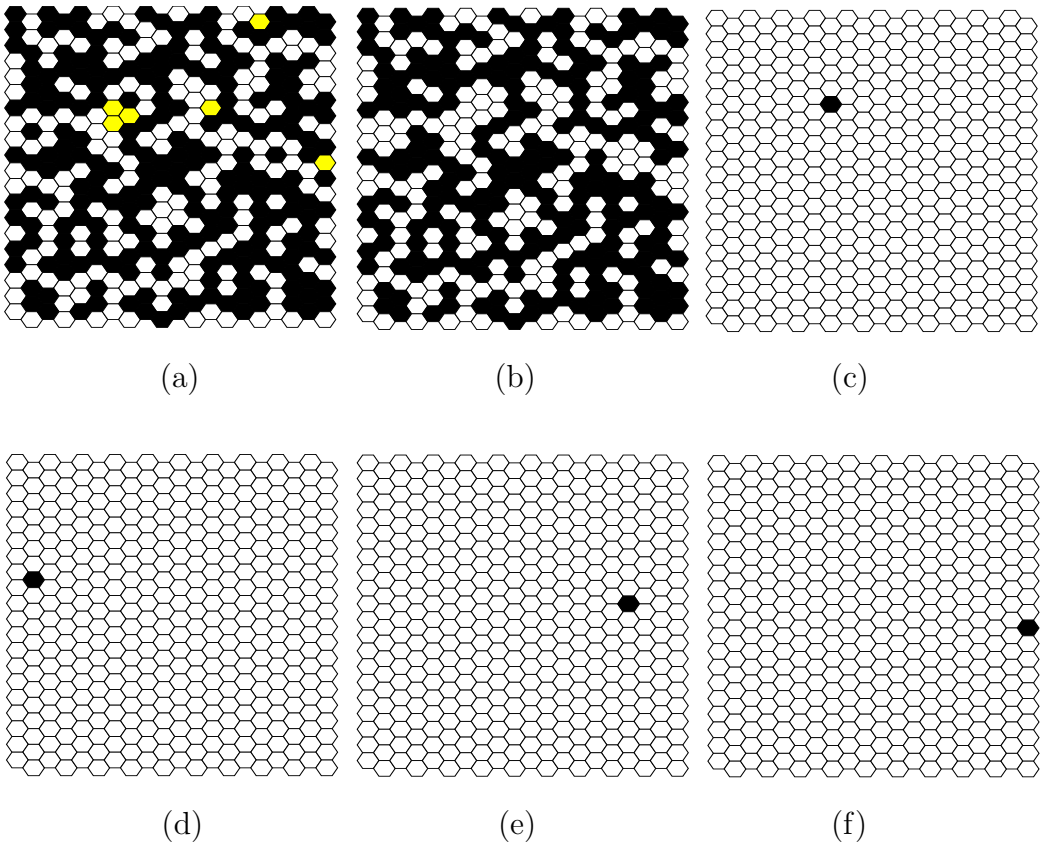


Figure 5.29: The grid for time step 191 and its connected sub grids. The averaged C-Value of the sub grids give a LC-Value of 0.1329. (a) the state of the grid at time step 191 of the simulation of the delta-notch randomised model. The settings used were $N_s = 0.9$, $N_t = 0$, threshold = 4, range = 3 and a toroidal wrap around boundary with a RSA updating scheme. Signalling cells are black, inactive cells are white and the yellow cells indicate an inactive cell with a threshold + 3 state; (b) the main connected sub grid of signalling cells extracted at range 1 and with a C-Value of 0.6478; (c) - (f) the four singletons extracted from the grid, each with a C-Value of zero.

In the $N_s = 0.1$ example the latter time steps have forty connected sub grids each with a three cell cluster and a C-Value of 1. Consequently the LC-Value reflects the regular pattern of three cells that is created across the grid. But the example of the $N_s = 0.9$ shows how the calculation of the LC-Value needs adjustment to ensure that more consideration is given to the distribution of the signalling cells and the prevention of a very small percentage of the signalling cells from disproportionally affecting the LC-Value.

5.5 Deterministic model

One set up was tested with the deterministic CA model. This simulated one package set to replicate moving across a 10 by 10 grid in order to see the changing state of the resource usage of the cells on the grid. A replicating package moving vertically or horizontally across the middle of the grid takes 9 time steps to move across the grid. In this time, providing resources are available, the resource usage across nodes is greatly increased as replicated packages are spun off to both sides of the replicating package. The life time of the replicated packages was set to one time step, in which time they could themselves replicate if there were resources available. This short life span was used so as to get as uncluttered a view of the replicating process and resource usage as possible.

Any AP that had a resource reservation requirement of the maximum of 50 cycles could result in the resource impact being increased by 50 cycles, providing the relevant node had available resources. This would suggest that the longest any footprint of an AP would be seen on the grid is with a merge AP traversing diagonally across the grid. If the AP is processed at every AN it visits, then it takes 45 cycles / time steps. This is extended to 91 cycles if it has a reserve memory requirement lasting for the maximum of 50 cycles. Consequently, after 91 cycles, or even less, it would be expected that no resource usage would be registered on the grid and all the cells would be inactive. However, a replicating package started off from a corner of the grid to move diagonally across the grid results in the replication folding back on itself and locking into a continual oscillation between two patterns. Figure 5.30 shows an example of a 10 by 10 grid initiated with a replicating agent injected at the bottom left corner. The grids shown in Figure 5.30(h) & (i) are the start of a two grid oscillation that continued for the rest of the simulation of 511 time steps. The measurement of this oscillating state can be observed in Figure 5.31.

The BBR and the mean density metrics reflect the basic difference in the number of active cells and the area covered by them between the two oscillating grids. The entropy and connectedness measure flat line as there is no difference in their perception of the active cells on the grid. But all indicate that the grid is still occupied, and the high C-Value points to the connectedness of the structure. While this is an interesting manifestation, any claims of emergence should be restrained. The model does not represent any factual representation of a live system, nor is it a model that can be compared with the output of a real time system. As such the oscillation has to be considered as an artefact, although a useful test case for this

research. The wrongful interpretation of artefactual events is an habitual problem with computational system that is aligned with the wrongful assumption that a model can be seen as more than it actually is. However realistic its construction is, it is only a model of the system.

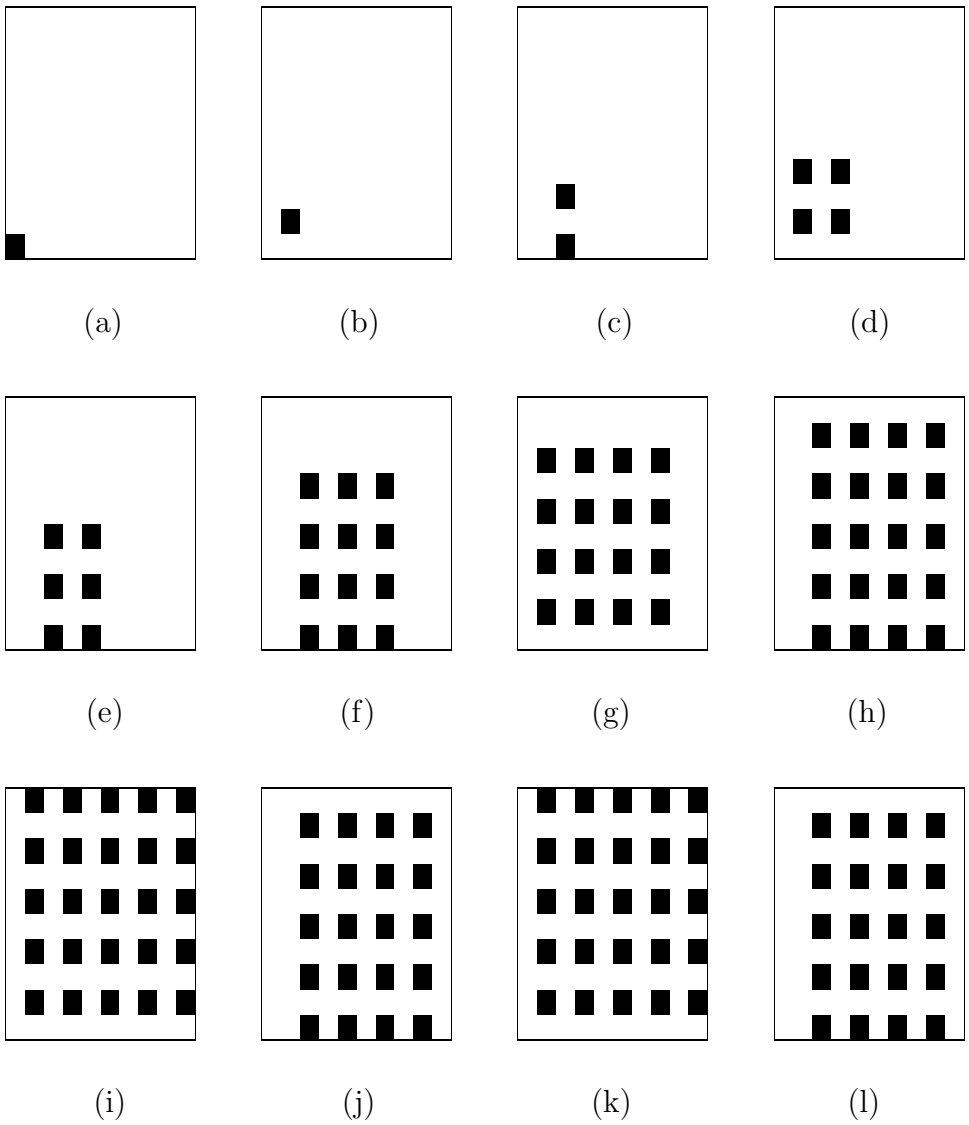


Figure 5.30: The results of a replication package moving diagonally across the grid from a corner. The AP was set a lifespan of 1 cycle in order to leave the grid as uncluttered as possible. The first 12 grids are shown with the black cells indicating the nodes where resources are being consumed. The oscillation cycle of two grid states starts with (h) & (i) and was still evident in steps 510 and 511 at the end of the simulation.

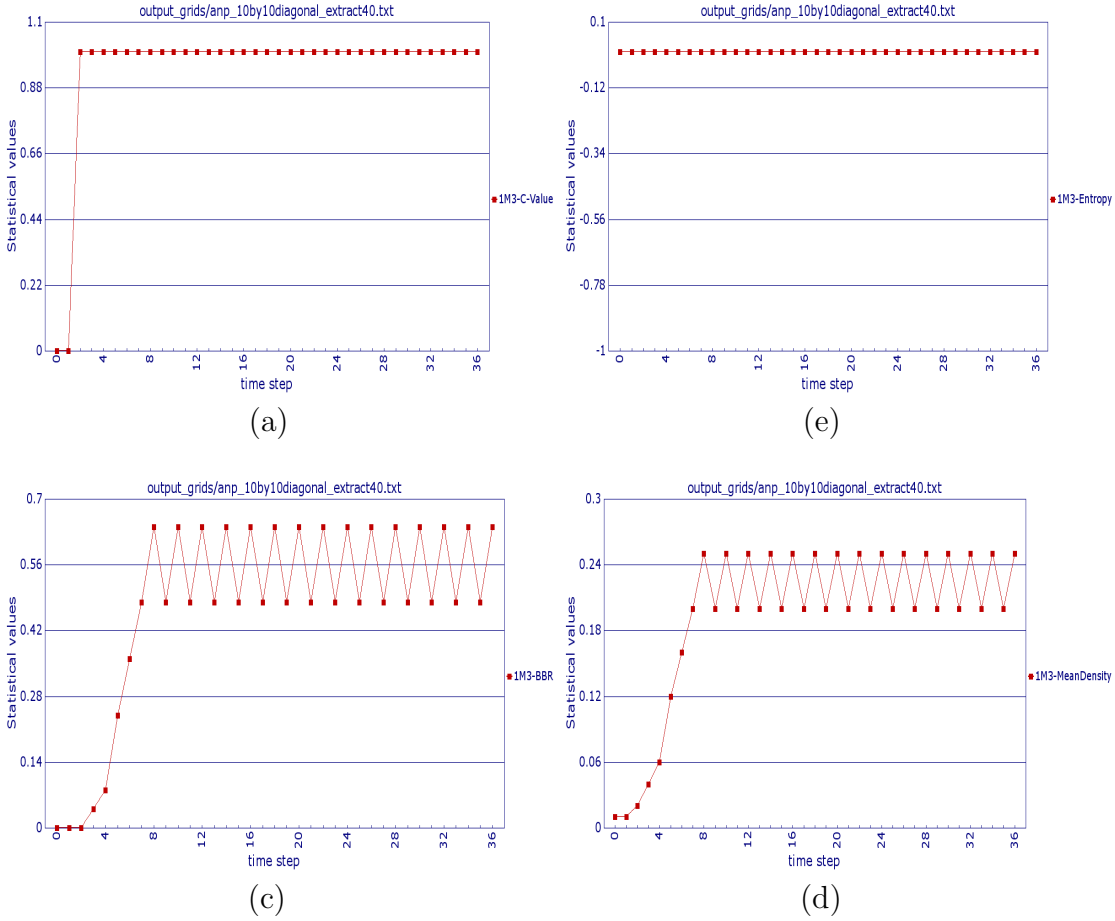


Figure 5.31: The metric analysis of the AN deterministic test. This was run on the first 36 grids. (a) C-Value, (b) entropy, (c) BBR and (d) mean density. The fluctuation in (c) and (d) reflects the different number of active cells in the two oscillating grids first seen in Figure 5.30(h) & (i). The two grids in the cycle both consist of one cluster, leading to the same C-Value for both grids and, consequently, flat lining on (a) for a value of one; the same occurs for the entropy value on (b) which settles at zero. The prefix to the legends of 1M3 signify the value of the cells being measured (1) and a Moore neighbourhood (M) with a range of 3 (3).

5.6 Summary

This chapter presented the results of the tests and simulations of probability and dynamic scenarios, and the models representing three CA categories. The probability scenarios gave the first indication that there was a correlation between the C-Value, the mean density and the random distribution of occupied or active cells on the CA grid. The probability scenario created grids with the percentage of randomly located occupied cells determined by a probability value. The BBR metric was relatively static, thus showing how even in the more sparsely occupied grids the randomness of the placement of the cells led to them being spread out to the edges of the grid. The entropy showed how the diversity of the cluster sizes on the grid increased until a single spanning cluster became prevalent across the grid. The mean density, as expected, increased as the number of occupied cells increased. The C-Value ran alongside the mean density, indicating that the increase in connectedness of the occupied cells was determined by the increase in their number on the fixed dimensions of the grid, rather than any rule or planned clustering of the cells.

The dynamic scenario used a simple gathering algorithm to move the active agents into the centre of the grid. The C-Value reflected this move from a random initial grid by increasing above the mean density. In this way, the randomness of the initial grid was shown by the mean density and C-Value having a similar starting value, and the loss of randomness was then illustrated by the increase in the C-Value. The gathering algorithm did not seek to pack the active agents tightly, but a C-Value above 0.900 indicated when the active agents had collected into a single centralised cluster. The dynamic scenario was used to see how the metrics performed when there were different extremes of cell occupation. This mainly concentrated on very low numbers of active agents, specifically four, eight and twelve. Of the four metrics, the BBR and C-Value provided information on the state of the grid, with the C-Value providing more information on the final time steps, or stages of the gathering process, than the BBR.

Both the Moore and the van Neumann neighbourhood were used in the probability scenario. The dynamic scenario was used to conducted a more detailed comparison of the two neighbourhoods and the use of different search ranges. The values generated by the two neighbourhoods will be different as, for example, a Moore neighbourhood will potentially find more clusters, affecting the entropy, and more connections, influencing the C-Value. Also with a Moore neighbourhood the C-Value will have a higher maximum number of connections for a specified number of

active cells. While the tests using 350 active agents revealed no obvious, observable difference in the graphical representation of the metrics, apart from the expected difference in values, a small difference was seen in the graphs of eight and twelve agents using a Moore neighbourhood and a range of 3. A range of three gathers any singletons identified with a range of 1 and of 2 into a cluster that is then reflected in the C-Value. Although this was of little overall significance, a Moore neighbourhood with a range of 3 was adopted for the subsequent simulations.

The two scenarios were then combined to evaluate how the metrics performed from a low population of active agents of occupying 10% of a grid up to a grid with 90% occupation. The graphs of the results showed clearly the effect on the entropy and the BBR as the graph became more densely populated. They both registered less and less information on the changing state of the grid. In contrast to this, the C-Value, starting alongside the mean density when measuring the initial grid, rose in value up to above 0.9000 in all the simulations, even in the final one where the 90% occupation of the grid resulted in a static value for the BBR and a zero value for the entropy. Overall the scenarios showed that the C-Value was consistent in identifying the gathering together of different density of active agents. The BBR proved to be a very simple but effective measure, but it lacked a certain finesse in the analysis of some of the simulations with a high number of occupied cells. The entropy measure proved to be effective for the job it was designed for, but its overall ability to provide a good level of information across all the time steps was limited. A value of zero could mean that there were one or no clusters, or that there were 2 or more clusters but all of the same size. Credence was also given to the concept that a C-Value aligned to the mean density implied a degree of randomness in the distribution of the cells being measured.

The simulation of the extinction regime of the reactive-diffusion and chemotaxis model confirmed the C-Value was approximately the same value as the mean density when the distribution of cells being measured was random. The subsequent simulation of the decentralised gathering of the amoebae showed the effect of more than one amoeba being able to occupy a cell, with all the metrics exhibiting some fluctuating in their values, but maintaining the expected shape of the graphical representation of measuring the decentralised gathering. The simulations using a low population of 4 amoebae and then a much higher one of 360 verified the findings of the dynamic scenario. Firstly, that it was difficult to capture information when only a few amoebae were present, although the C-Value and BBR gave some useful, albeit limited, information; secondly, that of the four metrics the C-Value provided the most distinct information on the gathering of the 360 amoebae.

The delta-notch model was simulated first without any noise perturbation, and then with two diverse spatial noise settings of $N_s = 0.1$ and $N_s = 0.9$. In the no noise simulation all three ranges were tested, confirming that the catch all range of 3 was preferable. The wrap around nature of the toroidal boundary means that the maximum connections achievable by a group of occupied cells is higher than the null boundary used by the previous model. This in turn can affect the general level that the C-Value reaches for a relatively compact cluster, as seen in the results of the no noise simulation. Although it can also give rise to a perfect C-Value score of 1. The simulations using noise highlighted the way the C-Value performed when the focus was not on the gathering of active agents or cells into a single cluster on the grid. The simulation using $N_s = 0.1$ produced a highly ordered pattern of 3 cell clusters in eight rows of five clusters on each row. This forty cluster pattern was hardly registered by the C-Value, which only just rose above the mean density. Whereas, in the simulation with $N_s = 0.9$ the occupied cells were visually sprawled across the grid. The cluster was connected but lacking any real order, but it had a higher C-Value than $N_s = 0.1$. The C-Value for $N_s = 0.9$ oscillated in a similar trajectory to that of the mean density, implying that while the signalling cells were connected, there was a certain randomness to their distribution. This drew attention to how the C-Value was set up to evaluate the connectivity of the signalling cells on the grid and hence the degree to which they were gathered together, rather than any pattern that might evolve.

The issue of how to identify the recurring three cell pattern from the $N_s = 0.1$ simulation was examined by averaging the sum of the C-Value of each cluster found on the grid, with each singleton having a zero C-Value; this produced a localised C-Value (LC-Value). This worked well, although the LC-Value results from the $N_s = 0.9$ simulation threw up the need to consider how to cater for the imbalance that a few singletons or small clusters could create when the main bulk of the signalling cells were in one cluster.

The final simulation of a theoretical active network failed to provide any further information on the performance of the metrics. However, it did demonstrate the potential for artefactual behaviour to be created by computation modelling. A similar incident had been observed in one of the tests run in the dynamic scenario (*see Figure 5.8*).

The results, including an assessment of the performance of the new connectedness metric, are discussed further in the next chapter.

Chapter 6

Discussion

6.1 Introduction

This chapter considers the main areas of interest raised by the work carried out in the thesis. It first revisits the CA types and the extension proposed in subsection 2.6.5, before looking at the performance of the scenarios and models outlined in chapter 4. This leads into a discussion of the modelling process and verification and validation of the models used in the thesis The performance of the metrics outlined in chapter 3 and analysed in chapter 5 is then reviewed, with special attention given to the new connectedness metric. The main discussion ends with a reconsideration of commonality and the identification of common metrics.

6.2 Classifying CA types

Ermentrout and Edelstein-Keshet [1993] categorised CA models within biology into three broad types (*see Table 2.2*). The versatility and relative easiness of implementing CA models has seen their widespread use within other areas of research (*see Table 2.1*). The original three categories were based on three different views of the lattice or CA grid, which was usually two-dimensional, and the process that evolved on it (*see subsection 2.6.5*). The first, deterministic automata, represented models that focused on the cells or nodes of the grid and their state. The second, particle automata, involved grids where particles or agents moved randomly or deterministically across the grid. The last, growth automata, dealt with irreversible models where the lattice grew as new 'sites' were absorbed as new, occupied cells. While this still forms the basis of any categorisation of CA models, there is one recommended definition change and one gap in the area of cell state based CA models.

Although the particle automaton was defined in terms of particles colliding and changing, it also includes non colliding particle models and the definition needs to reflect this. The change could be considered pedantic, as non collision agent CA models, including those modelling ant pheromone trails or traffic, often involve rules governing situations where a cell is occupied by more than one agent. This could be construed as a 'collision' situation, or at least an impending one. However, in such models it is not always the case that the particles undergo state change, as proposed in the original definition; although it might be argued that direction could be seen as a state of the particle, and a collision avoidance CA model involves agents avoiding colliding by changing direction, it would be more appropriate to talk in terms of additional rules governing collisions or multi occupancy of cells.

An obvious limitation to the three categories defined in [Ermentrout and Edelstein-Keshet, 1993] is the exclusion of the use of probability in any updating method or rules. The deterministic category was defined as using a synchronous updating scheme and deterministic rules that excluded the use if any randomisation either through an asynchronous updating method or rules that involved probability. However, as was outlined in subsection 2.6.3 a number of asynchronous updating schemes have been developed and are believed by many to better reflect real life when compared to the synchronous update of the whole grid. In the same way the use of probability with the rules set to update the state of the cells or the movement of agents across the grid introduces an element of chance and randomness that more closely resembles our perception of real life and, especially, complex systems. There has been an increasing number of CA models incorporating asynchronous updating or probability in their rules, or both. Thus the three categories outlined in [Ermentrout and Edelstein-Keshet, 1993] do not cover the range of CA models now being constructed and simulated. A new CA category was proposed titled randomised automata in subsection 2.6.5, that takes a cell state perspective and incorporates asynchronous updating and probability. Examples of this new category were listed in Table 2.2 and the delta-notch signalling CA of [Cohen et al., 2011] provided the basis of the randomised CA model used in the thesis. This model used asynchronous updating schemes and probability to mirror noise in the update rules, and illustrates the richness of the CA models excluded from the original categorisation.

6.3 Scenarios and models

Three sets of scenarios and three models were created, as well as an analysis suite of programs. The results of the analysis of the metrics used is discussed in section 6.5 and section 6.6. The purpose of this section is to review and appraise the programs and their performance.

The scenarios were constructed to verify the key functionality of the programs and to provide benchmark testing of the analysis part of the program suite. The hand crafted scenarios were used to test the ability of the programs to extract connected clusters of active cells from an output grid. The probability scenarios evaluated the ability of the metrics outlined in chapter 3 to distinguish between grids with varying populations of active cells randomly located on the grid. The dynamic scenarios benchmarked how the metrics tracked different numbers of active cells as they gathered together in a cluster. A final test used the output of the probability scenarios as initial grid input into the dynamic scenarios. The purpose of this was to cross reference the metrics using the breadth of the probability test with the results of running them under the dynamic scenario, providing an idea of how the metrics performed when tracking the gathering of agents on a more and more cluttered grid space. Both chapter 4 and chapter 5 showed that the scenarios successfully performed their tasks, both as test programs and as bench markers.

The three lattice based models outlined in chapter 4 were built to create output from three different types of CA models. Two of the CA model types, deterministic and particle automata, were from the original proposal by Ermentrout and Edelstein-Keshet [1993]; the other was the new randomised automaton proposed in this thesis.

The particle automaton model was based on a study of the decentralised gathering problem using reactive-diffusion and chemotaxis as illustrated in [Fatès et al., 2008]. The Perl program constructed was successful in confirming the findings of the static, non-coherent and extinction regimes tested in the original work, (*see subsection 4.6.5*). It demonstrated that a minimum introduction of noise was required to initiate a gathering of agents on a two dimensional grid, with and without obstacles on the grid. The ability of the amoebae to navigate around the obstacles and gather was also observed. The metrics conformed to the benchmark settings found by the dynamic scenarios and provided a good set of output grids tracking the gathering process over a series of time steps. It also corroborated the problem area of registering enough differentiation in the action of a very small

number of agents on a grid highlighted by dynamic scenario tests.

The randomised automaton followed the model outlined by Cohen et al. [2011] and their study of delta-notch signalling and the use of structured noise to generate self-organising tissue patterns. The original work was itself a progression from work using differential equations [Cohen et al., 2010] and, like the particle automaton model, it illustrates the power of a simple CA computational model to recreate and facilitate the investigation of highly complicated and interactively complex systems. The Perl program recreated the hexagonal grid and toroidal boundary conditions of the original work, and confirmed the pattern generating ability of different combinations of temporal and spatial noise using the specified three neighbourhood depth ranges, (*see subsection 4.7.5*). The challenge presented by this model is that the number of active cells is often fairly constant after a few time steps for a lot of the noise combinations. Thus it becomes problematic in identifying any significant change as the active / signalling cells become more or less compressed.

The deterministic automaton model was built using the Complex Cellular Automata (CxA) framework and the MUSCLE Java code base, (*see section 4.8 and appendix A*). This had the advantage of both providing a formal framework for defining and documenting how the grids link together and providing the Java code for the the interaction between the respective spatial and temporal grids. However, while the code framework is recommended for defining hybrid and CxA applications, the code base would benefit from being updated. Perhaps the most significant change that could be made is the method of connectivity between the respective grids. The existing method is a bound pipe. This means that once data are sent down the pipe all other processing is stopped on the grid at the sending end until the grid at the receiving end of the pipe has finished its processing and returned a response. As the very successful simulation of models using this set up have shown in [Kroc et al., 2010], this is not a problem unless a better form of asynchronous updating between the grids is desired. In many ways it is identical to the discussion of the merits of CA synchronous and various asynchronous updating of cells on a CA Grid (*see subsection 2.6.3*). However, regardless of thoughts over the improvement of the code base, the real issue is the suitability and performance of the deterministic model.

The problems over verifying the Java code developed to run the AN deterministic automaton model were highlighted in section 4.8. The crux of this problem was that the original model [de Silva, 2004], and thus the new Java one, was built on

an abstraction of the functions of a theoretical AN. There was nothing to validate the performance of the model. In this way the model was nearer a scenario than a model. While the deterministic model produced a very interesting result, as there is no way of checking the outcome against a real situation, it has to be classified as an artefact.

The scenarios verified that the key aspects of the program suite were working, while the particle and randomised models, once validated against the original works, provided information on the performance of the metrics. But the deterministic model does not help in any assessment of the metrics, rather any interpretation of the changing state of its output space has to be based on how the metrics executed with the other two models. Any observations made about the deterministic model has to be seen as hypothetical and lacking any validation. Consequently, the artefact of an oscillating pattern on the grid, (*see section 5.5*), could be hypothetical compared to a replicating virus flooding a network and promoted as a valid high level structure to analyse. But, its remoteness from a real life situation means it fulfils the role more of a scenario, rather than a model that can be verified and validated.

The particle and randomised automaton models both produced output that raised questions of not just how it would be measured in a quantitative way, but also how qualitative differences between output grids could be identified and potential key moments flagged. A significant factor in the real and perceived contribution of the particle and randomised automata models is also their association to a real life process, the activities of amoebae and delta-notch signalling respectively. As has been stated above, the deterministic model does not share the same connection. The status of the models within any modelling framework with reference to levels of abstraction and quality of data used is discussed in the next section.

This section cannot close without commenting on the decision to code the models, the analysis and the display routines. The use of modelling environments, such as Matlab, was an alternative to writing thousands of lines of code. However, the desire to model diverse CA types led to a decision to control the code directly, especially once Perl was adopted first to write the analysis of the Java AN model and then the particle and randomised CA models. Coding presented the only real option once it was decided to recreate existing models. This is not to gloss over the wide range of problems that can and inevitably do occur with developing, testing and running three models and an analysis suite of programs. But the greater understanding of the data format that coding the models brought, and

the flexibility and control it provided was felt necessary because of the range and diversity of the systems being tested. The benefits of this approach was mainly seen, for example, in the ability to capture input and output grids and re-run analysis without having to repeat a simulation; as well as the ability to select specific time steps or grids for rendering into *html* or *xml*.

6.4 Modelling

Modelling is an essential and fundamental part of the scientific process. But what constitutes a suitable or good model varies between disciplines. A scientific model can be seen as one reflecting reality. This ties the application of a model to the real world, where the validity of the simulation is associated with the qualitative assessment of both its input and output. In this way such *in simulacra* models are akin to a mechanistic view of the world, where systems can be reduced to component parts and their functions revealed. This is not only a valid approach, it is the heart of the scientific method that has allowed scientific investigations and discoveries to advance at such a pace over the recent centuries:

- a question is formulated through an observation of some process or event;
- a hypothesis is constructed to provide a possible explanation for the observation;
- predictions are made about the logical outcomes resulting from the hypothesis; and
- tests, often in the form of models, are then constructed and used to investigate if the hypothesis actually reflects the original observation of the real world.

But if you were to consider that observation is itself an abstraction or modelling process, then *in simulacra* models are part of a cyclic, reciprocal process of models modelling models, confined by the bounds of our knowledge. Scientific advancement is based on the recording, reproducing and validating of data and processes. This has to be done through acceptable and universally agreed and understood means. But, as was shown by Gödel's incompleteness theorem, our understanding has limits; limits that we can expand, but there will always be something outside our range of understanding, at least in terms of being able to explain and validate it within acceptable scientific rules.

Computer simulation, its epistemic worth and its role as an opaque though experiment were discussed in section 2.5. The three models used in the thesis not only represent three different categories of CA, but also three different aspects of computer simulations.

The particle model used a simulation of the observed behaviour of amoebae when faced with a shortage of food to examine the problem of decentralised gathering of agents connected only at a very localised level. Thus an abstraction of a real life biological system was modelled to provide insight into how, for example, robots could self organise into a group. The abstraction, modelling and simulation of the amoebae within a computer environment has the potential to provide guidance on the rules and attributes that could be used in the coding of the robots. So the analogy is drawn between the behaviour of the amoebae and the decentralised gathering of independent agents on a CA grid; and then between the decentralised gathering and the way of getting robots to gather in a group without a centralised control system.

The randomised model also abstracted and modelled a biological system, but its purpose was to understand how the system worked. How delta-notch signalling works is not fully agreed on, but the model showed how the process could, when a certain amount of spatial noise was introduced, create a highly organised pattern. This replication of how, for example, the patterns of skin pigmentation or hair growth could be stimulated, was an example of how a CA could be used as an experiment. This is further strengthened by the work in [Cohen et al., 2011] being a replication of modelling work carried out with mathematical models using partial differential equations in [Cohen et al., 2010]. While it is generally accepted that experiments are of more epistemic value than simulations, and mathematical models of more scientific validity than computational models, there is a role for computational models as experiments in areas where the system is complex or its operation still open to debate.

The deterministic model recreated the model of the abstraction of a theoretical active network. The simulation provided the least useful information, both on the system being modelled and for analysis by the metrics being used in the thesis. While the two previous models had some reference and connection to a real system, this model had none. In some ways it could be viewed as an opaque thought experiment that attempts to look at how an active network might behave under certain conditions and with a selection of attributes. However, it suffers from being overly complicated, which directly affects any worth it might have as

an opaque thought experiment as it assumes too much without having any ability to verify, let alone validate any output from the simulation. Although this model failed as an effective simulation, it did provide support for what did and did not make a good model.

The simulation methodology adopted in chapter 3 took account of the fact that the three models were all based on previous work (*see Figure 3.1*). While the original works, with the exception of the deterministic model, were validated against real systems, the particle and deterministic models were validated against the documentation of the original work. Both the particle and deterministic models were successfully validated against the original work in subsection 4.6.5 and subsection 4.7.5 respectively. But this validation also works the other way; the successful recreation of the original work using the same rules, but using a newly built computational model, validates the findings of the original, in the same way that an experiment gains credence when successfully replicated.

The deterministic model was based on a previous model that was not itself validated against a real system. This meant that there was no actual behaviour that could be observed and any recreation was based on hypothetical behaviour. In this way the deterministic model failed to comply with the simulation methodology. Consequently, its limitations as a model in turn validates the simulation methodology.

6.5 The metrics

Four metrics were introduced in chapter 3. Three had been used in previous work measuring 2D CA. The bounding box ratio (BBR) was a relatively simple metric that was based on the distance between the outer active cells on an x, y axis. An active cell might represent the state of the cell itself, or the presence of an agent that is capable of moving around the grid. The mean density was a straightforward measure of the number of active cells in respect to the number of cells on the grid. The third known metric was a more involved entropic one looking at the cluster size diversity. The fourth metric was newly proposed in this thesis, named the C-Value, and evaluated how connected the active cells on a grid were. The C-Value will be discussed in more detail in the next section. In this section the overall performance of the metrics will be considered.

The scenarios and models confirmed the fundamental states of a 2D CA output grid. It has two main characteristics that each have two extremes. Firstly there is

the number of active cells, which can alternate from none to a grid full of active cells. Secondly there is the formation or order of the cells on the grid, ranging from random to tightly compressed or clustered. The order of the active cells on the grid has the additional feature that they may form a pattern across the grid that is not represented by a single cluster. Pattern recognition is a topic in its own right, but it has to be considered because what may be a visually dispersed pattern of small clusters of cells, may not register as clusters with an automated metric. The other main problem with any measuring of 2D CA grid is differentiating when there is a restriction on the amount of change that can be registered. This can be seen both when there is a very small number of active cells and when there is a densely packed grid. But the main point of a lot of measures is to identify a change, expected or unexpected, in the current state of the grid; or even better to predict such a change.

The performance of the metrics are considered in six areas that cover the key aspects discussed in section 2.7 and in subsection 3.3.1.

(1) *density* - the detection of changes in active population density on a randomly generated grid:

Two tests were run to assess how the density of active cells was registered by the four metrics; the probability scenario test in subsection 5.2.1 and the combined probability and dynamic scenario test in subsection 5.2.3. The mean density obviously identified the density of active cells on the grid. The BBR was ineffective due to the random spread of the cells across the grid, while the entropy metric was only marginally better. The C-Value kept pace with the mean density, so initially it could be seen as a good indicator of the density. However, as will be seen in item (3), the C-Value is designed to measure the the connectedness of the active cells and moves away from the mean density as the active cells gather together. So the C-Value does not signify the density of active cells on a grid, rather the connectedness of the active cells.

(2) *gathering* - the tracking of the centralised or decentralised gathering of agents on a grid:

This was evaluated with the two tests using the dynamic scenario in subsection 5.2.2 and subsection 5.2.3, and in the particle automata simulation in section 5.3. The mean density does not indicate any change on the state of the grid unless there is a fluctuation in the number of active cells, (such as when in the particle automaton model where more than one amoeba can occupy a single cell). Obviously as the percentage of active cells increases on

the grid, the probability that the cells are connected increases, but it does not indicate any movement of the agents into a less or more clustered state on the grid. The entropy metric indicates the point at which the active cells approach and then reach the stage where they span across the width or height of a grid; for this to occur there have to be enough active cells on the grid as well as a degree of gathering of the active cells. But once this spanning cluster is formed, the entropy metric does not provide any additional level of information, only that there is a pathway of active cells from one edge of the grid across to the opposite edge. The difficulty, as has already been stated, is that the entropy metric will show a value of zero for a number of different reasons; such as when there are either no clusters, a single cluster, or multiple clusters all of the same size. The BBR does indicate when dimensions of the virtual 2D box encompassing all the active cells has shrunk. Thus a gathering of active cells is shown in the BBR values where the simulation involved a gathering of the all the active cells into one cluster. The C-Value likewise illustrates when a gathering of active cells is occurring. But while the BBR becomes stable in the closing stages of the gathering process, the C-Value continues to register any changes. This can be seen on the tables in Appendix B.1.2 relating to the test of 350 agents gathering on different sizes of grid. This test terminates when there is no additional gathering process recorded between two steps; thus the last two time steps will have the same values. On the 25 by 25 grids the BBR stabilises for the last seven of the nine time steps required to gather the agents together, whereas the C-Value continues to rise until the concluding two time steps. On the 50 by 50 grid the BBR settles at 0.4224 at time step thirteen, whereas the the C-Value is at 0.8551 at the same time step, but continues to increase until it settles down to the same value, 0.9507, for the last two time steps, nineteen and twenty. This trait is repeated for the 75 by 75 and 100 by 100 grids. In addition to this, the BBR value is dependent on how many cells are gathered together as this will affect the dimensions of the resulting cluster. Whereas the C-value, when used to analyse a grid with a null boundary condition, moves to 0.9000 and above when a single cluster is formed, regardless of the number of cells in the cluster.

(3) *low end* - the performance in monitoring the movement of a population of four, eight and twelve agents:

This was evaluated using the results from the scenario tests in subsection 5.2.2 and subsection 5.2.3, and in the particle automata simulation in section 5.3. As would be expected, the mean density provided no useful information regarding

any changes in the state of a grid with a very small number of agents on it. The entropy also registers nothing of any use as there are too few agents to form significant occurrences of different sized clusters needed to register a non zero entropy value over a series of time steps, if at all. Both the BBR and C-Value register the gathering of a small number of agents. Four agents are initially more easily picked up by the BBR as they move together across the grid as it, in effect, measures the reduction of the distance between them; while the C-Value depends on an increase in the connectedness that occurs once they have begun to get in closer proximity to each over. This can be seen in Figure 5.8 and Figure 5.19. But as the number of agents increases to eight or twelve the C-Value provides more information over the full range of time steps with greater variance over the closing time steps, (*see Figure 5.6, Figure 5.9, Figure 5.10 and Appendix B.1.1*).

(4) *high end* - how well the metrics tracked changes in state when the grid was densely populated:

This was evaluated using the results from the scenario tests in subsection 5.2.2 and subsection 5.2.3, and in the particle automata simulation in section 5.3. The mean density and entropy metrics both show no useful information on a grid of 25 by 25 with 350 agents, (*see Figure 5.4(a) and Appendix B.1.1*). The BBR also fails to register much information on any changes to the grid over the eight time steps it takes to gather the agents together. Figure 5.5(a)shows how the outer active cells are either on or next to the boundary of the grid even when the gathering process has stopped. The C-Value succeeds in identifying the changes in the layout of the agents on the grid as they gather together, starting with a value of 0.5574 and rising to 0.9913 when the gathering process is completed.

(5) *randomness* - the identification of randomness in the location of active cells on the grid:

This was evaluated using the results from the scenario tests in subsection 5.2.1, subsection 5.2.2 and subsection 5.2.3, and in the particle and randomised model simulations in section 5.3 and section 5.4 respectively. Neither the mean density nor the BBR are designed to capture anything specific about the randomness of the active cells on the grid and the results reflect this. The entropy and C-Value are based on identifying clusters of active cells and the move towards a spanning or single cluster. As this is the opposite to a random distribution of the active cells, it can be seen as a move away from randomness. However, the entropy metric is based on cluster size diversity

and a value of zero could indicate no clusters and, thus, a random state, or only clusters of one size, or a single spanning cluster. The C-Value gives a clear view of the gathering in the particle automata simulations using a null boundary condition, moving up to above 0.9000 and even to 1.000. Thus lower C-Values imply that the distribution of the active cells on the grid is more random. In various of the scenario tests and in the simulations used to consider how to identify randomness, active or occupied cells were set up randomly on the initial grid. These initial grids with their random spread of active cells had the distinction of having a mean density and C-Value with very similar values. This is likewise the case with the extinction regime simulation of the particle model in section 5.3, where the last grid for *p_value*={0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9} have excited cells on them that have not been created with rules dictating any order to their location, (*see Table 5.5*). This indicates a link between randomness, the C-Value and the mean density. This is discussed further below and in the next section.

(6) *patterns*

This aspect of measuring the 2D output space of a CA was considered with reference to the simulations of delta-notch signalling model used as the randomised CA. This model used a toroidal boundary and an asynchronous updating scheme, which presented a different context to the rectangular grid and synchronous updating scheme of the particle automaton model. The intention of the original model in [Cohen et al., 2011] was to show how variances in the spatial and temporal noise could be used to create patterns that reflected such things as skin pigmentations or the distribution of hair follicles.

In the first group of simulations using no noise, (*see Table 5.6, Figure 5.23 and Figure 5.25*), the entropy metric provides no information on the state of the grid. The mean density adds nothing beyond its basic purpose. The BBR indicates that the occupied cells on each of the samples cover essentially the same 2D dimensions. Although this does not encompass the full dimensions of the grid as the selection of a threshold of one ensured that each occupied cell was surrounded by unoccupied cells. The C-Value for each simulation is at least 2.7 times the mean density, and excluding the simulation that has a C-value of 1, the range is between 0.5625 and 0.7381. This indicates, with reference to the C-Values generated in the scenarios tests and the simulations of the particle model, that the distribution of the occupied cells is not random, but has a certain degree of order.

The second set of simulations produces two different patterns. One with a highly ordered pattern of forty clusters, each with three occupied cells and spread five to a row over eight rows; and the other with most of the occupied cells linked in a sprawled looping pattern across the grid (*see Figure 5.26(a) and (b) respectively*). The BBR and entropy do not provide any significant information about the two grids. The C-Value in both examples keeps pace with the mean density. This has been associated with a random distribution of active cells on the grid, but in Figure 5.26(a) the pattern is far from random. The contour of the C-Value graph in Figure 5.27(a) is similar to that of a graph showing a gathering of active cells into a single cluster, but the C-Value does not rise above 0.3371. The C-Value was set up to identify a gathering of active cells into a single cluster, not a highly ordered pattern of small clusters. In subsection 5.4.1 the C-Value was successfully set to calculate the localised C-Value (LC-Value) of the clusters on the grid. In this specific example the LC-Value successfully identified the highly ordered pattern. The LC-Value will be discussed further in the next section. In contrast to Figure 5.26(a), the graphs produced for Figure 5.26(b) in Figure 5.27(e), (f) and (h) show the values fluctuate across the time steps. The mean density ranges from 0.4675 to 0.5550 and the C-Value from 0.4668 and 5546. The active cells are largely contained in a single linked group that meanders across the grid, rather than a pattern of small clusters. This indicates that the distribution of the active cells is random, without any noticeable pattern or order.

Overall the tests and simulations have shown that the mean density, BBR and entropy metrics generated the results that were expected of them. The mean density does the job it is designed for, but on its own it does not provide any real information aside from the density of the active cells on the grid. The BBR's strength is its relatively easy way of assessing the gathering of agents into a single cluster. But its final value when measuring the active agents in the grid gathered into a single, ordered cluster is dependent on their number. The entropy measure selected was fairly specialised, but its purpose was to provide a closer comparison to the new connectedness metric. However, it did not give a useful measure across the full range of time steps in a simulation. It does have some use in indicating a certain level of randomness as a high level of cluster size diversity indicates a state removed from a compact, single cluster. The C-Value proved adept at identifying the gathering of active agents into a single cluster and when aligned with the mean density it indicated the randomness of the distribution of the active agents on the grid. Its ability to identify patterns of clusters is affected by the level at which the C-Value is calculated. The C-Value is discussed further in the next session.

This section closes with comments on the choice of neighbourhood and range. There was only a marginal difference in the tests comparing the von Neumann and Moore neighbourhood, (*see subsection 5.2.2*). The expectation was that the greater number of connection options of the Moore neighbourhood would allow a finer, more distinctive measure, as well as finding larger clusters. But the metrics, including the C-Value, performed in a similar way for both neighbourhoods. The Moore neighbourhood was selected because it did show a very slight improvement on the von Neumann. The Moore neighbourhood also includes all eight cells surrounding the cell in focus, which is more closely aligned with the hexagonal grid used in the delta-notch signalling model where all six surrounding cells are part of the neighbourhood. The choice of range was three, although many of the tests, especially the randomised CA model, gained no additional value than when using a setting of two. However, the choice of three includes, rather than precludes, searches for nodes two cells apart. Therefore the selection of a maximum range of three acts as a catch all.

6.6 Measuring connectedness

This section discusses the performance of the C-Value connectedness metric proposed in subsection 3.6.4. The judgement of the usefulness of the metric is first based on whether it successfully measures anything of use in at least one CA model type. Following it meeting that initial, but key criterion it is then a question of whether it has any uses in other areas and CA model types. Finally, if it performs across model types it is then a question of its level of commonality. The latter is important, as something might have uses across domains, but not qualify as a common metric. The question of commonality is discussed in the next section. This section will focus on the first two questions, based on the tests and analysis carried out in chapter 5.

The C-Value was initially set up to measure the level of connectedness between the active agents or occupied cells on grid. This grid level view meant it was geared towards registering the gathering of the active agents into a single cluster. On the tests and simulations using grids with a null boundary condition and modelling the gathering of active agents, as in subsection 5.2.2, subsection 5.2.3 and section 5.3, the C-Value performed extremely well. The dynamic scenarios and the particle CA using the model of amoebae grouping together showed the C-Value rising to just below 1.000, regardless of the number of active agents on the grid.

A key observation, starting with the grids randomly populated with different densities of active cells for the probability scenario in subsection 5.2.1, was that the C-Value aligned with the mean density when the location of the active cells on the grid had been randomly generated. This was confirmed in the tests combining the probability and dynamic scenarios in subsection 5.2.3, where the C-Value started at approximately the same value as the mean density and then increased as the active agents gathered together. This has two linked significances. First, if the C-Value is approximately the same value as the mean density then the spread of active agents on a grid can be termed random. Secondly, the movement away from this random state on the grid is indicated by the C-Value increasing towards 1.0000 and moving away from its correlation with the mean density. Obviously, if the grid is densely packed both the mean density and C-Value will be high and there is very little change between an initial random placement of the active cells on the grid and the formation of a single highly connected cluster. In this way randomness here refers to the state furthest away from the order of a single cluster for the number of active cells on the grid. Thus it can be stated that:

1. the C-Value approaches 1.0000 as the active agents on a grid moves to a single, ordered cluster

2. the active agents on a grid are farthest away from an ordered single cluster when the C-Value is approximately the value of the mean density

This can also be seen in the extinction regime simulation of the reactive-diffusion model (*see Table 5.5 and Figure 5.15*). The metrics measured the excited cells on the grids. It was observed that the entropy measure reflected that it had not peaked and, thus, a single spanning had not been formed; rather there was still a high diversity of cluster sizes, which puts the state of the grid nearer to a random one than a single, ordered cluster. The seven final grids that still have excited cells on can be seen in Figure 5.15 and their C-Values and mean density are approximately the same, although Figure 5.15(a) and (b) have higher differences of 0.0706 and 0.0338 respectively, while the differences between the two values for (c) to (g) ranges from 0.0046 and 0.0165. The actual C-Value range is 0.1999 to 0.3335, and 0.1293 to 0.3206 for the mean density. The excited cells are created without any planned or coded order in mind, and can be seen as random, which is reflected in the correlation of the C-Value and mean density.

This can be applied to the simulations of the delta-notch signalling model used to demonstrate a randomised CA, where a toroidal boundary condition was used. The first configuration of the model was run using no noise and a range of two or

three and produced a simulation that has the optimum configuration for achieving the maximum number of connections possible for the seventeen signalling cells, giving it a C-Value of 1.0000, (*see Figure 5.24*). The other nineteen simulations have a C-Value between 0.5625 and 0.7143, (*see Table 5.6*). As the mean density ranges between 0.2031 and 0.2500, then the nineteen grids can be seen as moving towards an ordered, single cluster. Visually the grids displayed in Figure 5.25 confirm that the grouping of the occupied cells to be generally ordered, with the odd singleton cell unattached to the main cluster.

The simulation of the delta-notch signalling model using a spatial noise setting of $N_s = 0.9$ also illustrate this correlation between the C-Value, mean density and randomness or a lack of order, (*see Figure 5.26(b)*). The occupied cells are all linked together in a single group, with the exception of three singletons and two two-cell clusters. The C-Value approximates with the mean density, (*see Table 5.8*) and the way the occupied cells are sprawled over the grid in Figure 5.26(b) affirms that while the cells are linked, they are far from being grouped into a single compact cluster.

The simulation of the delta-notch signalling model using a spatial noise setting of $N_s = 0.1$ produced a highly ordered pattern of forty 3-cell clusters (*see Figure 5.26(a)*). This visually identifiable pattern is not picked up by the C-Value that is geared towards identifying the level to which the active cells are gathered into an single ordered or compact cluster. Instead, the C-Value runs alongside the mean density. The pattern can be viewed as a cluster of 3-cell clusters, but as the C-value is set to evaluate the connectedness of each active cell it can also be seen as quite removed from a single, ordered cluster; so in the context of the latter, the C-Value is performing as expected as it is not set to identify patterns of clusters.

In order to see if the pattern of clusters could be identified, the focus of the C-Value was moved from the grid to the individual clusters found on the grid, (*see subsection 5.4.1*). The series of C-Values gained from this localised evaluation of clusters and singletons on the grid is then averaged to give a localised C-Value (LC-Value). In the reanalysis of Figure 5.26(a) the LC-Value was 1.0000. This first run of the LC-Value was successful, and it looks very promising when measuring a highly ordered pattern. When the LC-Value was calculated for Figure 5.26(b) it was 0.4880, as opposed to 0.5345 for the C-Value, (*see Table 5.10*). At some of the time steps the difference was much more, such as time step 191 where the LC-Value was 0.1329 and the C-Value 0.5166. This is fine in one way as the LC-Value has no correlation with the mean density. But it does highlight the need

for more work on the LC-Value in order to counter the effect of singletons and small clusters on a grid that is mainly made up of a linked group of active cells. This was demonstrated in subsection 5.4.1, where two 2-cell clusters raised the LC-Value of time step 109 from 0.6478 to 0.8826, and four singletons in time step 191 dropped the LC-Value from 0.6646 to 0.1329. Some form of weighting needs to be introduced in order to realise the potential that the LC-Value demonstrates

The C-Value showed that it was successful as an indicator of when the active cells on a grid were gathering together into a single, ordered cluster. When the C-Value is correlated with the mean density then an evaluation can be made concerning the state of the grid with reference to an unordered state when the C-Value approximated with the mean density and to a highly ordered, single cluster when the C-Value approached 1.000.

6.7 In search of commonality

The pervasiveness of the modelling process has been mentioned on a number of occasions. The importance of context and modelling was touched on in subsection 2.4.3, where it was noted that how a system is observed depends on the context and how a complex system can be modelled in part by imposing a localised context. The context of the models used in this thesis include the CA environment used to model the different systems, the perturbations selected to use in the simulation of the model, the neighbourhood selected, the type of grids and boundary conditions, and the update schemes.

One of the features that makes the modelling of everyday life and events by humans so powerful is how commonality between different models and different contexts is identified and used. New models are increasingly created from the experience and feedback from existing models of diverse systems or with a different context. This may be seen as using experience, lateral thinking, anticipation or some other term expressing how to react to a developing situation. A key feature is either the identification of a known model to cater for the situation, or the changing of the context of a model, or the selection of parts of one or more models that can be used quickly to form a new model. The new model is honed and tempered by feedback, as all models should be.

This process of defining things within the bounds of our experience and knowledge is part of how scientific investigation is approached. Existing metrics, experiments, abstractions, models or other methods are applied to new areas of research, or used

to develop or justify new avenues and methods of research. Work is recorded in a way that it is possible to reproduce and verify it. This then facilitates the cross pollination of ideas and methods to other research and even other disciplines. This utilisation of commonality should not be surprising when the superb and invaluable pattern recognition process that humans possess is considered. Elements of fuzzy processing and the detecting of common features have to form part of this process, otherwise it would paralyse a person into inaction while they handled all the sensory data being fed in.

Skills learnt in one context are transferable by recognition of some degree of commonality to another context. Otherwise, no skill or experience would be transferable when faced with a new situation, but instead things would have to be learnt again from basic concepts. A key part of problem solving in a new context is through finding aspects of the new system that have some commonality with experiences of other systems or contexts. But as with most things there is a conceptual and a physical side. The discussion above on modelling touched on the conceptual or theoretical side of modelling as compared to the physicality of a mechanistic 'real' model. The validating of a traditional model can be seen as giving physicality to the conceptual. In such a way the establishing of standard measurements give physical reality to what started out as a conceptual idea of commonality. Thus any search for a physical expression of commonality has to start with the conceptual identification of commonality between models. There is no guarantee that such an identification will lead to the realisation of a physical commonality, the link may remain conceptual. Indeed where the model is one of a hypothesis rather than something observed, then the process is conceptual or theoretical unless the hypothesis and model are subsequently applied to some observed event.

The approach of this thesis was to go one step further and to seek commonality between models through the commonality of the technique used to model them. This is an attempt to use the context of the modelling environment to provide the conceptual commonality. The selected 2D CA provides a platform for measuring different types of systems. This presented the opportunity to not only look for commonality between the same types of CA models, but also between different types using the same modelling environment in the shape of the two dimensional grid used to visually show the simulation of the model. The measurement of the number of active agents in relationship to the size of the grid is a common metric; but the mean density has a very limited view. However, a strong argument could be put that the mean density tells you the occupation state of a grid, whether

in terms of active agents or active cells. But this is more a measurement of the grid, rather than the activity on the grid. The object of the search was to find a common metric that said more about the differing state of the output space.

A new metric, the C-Value, was proposed as a possible candidate for acting as a metric for measuring the common output grid used by the various CA model types. The models, while sharing the context of the CA grid space, had other differences. In the case of the particle CA a rectangular grid was used with a null boundary condition and a synchronous updating scheme; perturbation was also used in the simulation. The synchronous updating scheme was required to produce the required reactive-diffusion waves that were necessary, with the chemotaxis, to gather the amoebae together. In contrast the randomised CA used a hexagonal grid with a toroidal boundary condition and an asynchronous updating scheme for the simulation of the delta-notch signalling model. This simulation introduced noise to create patterns of signalling cells. Thus the contexts of the two models were used to observe two different phenomena, namely the gathering of active agents into a single cluster and the formation of patterns. The C-Value, as was discussed above, proved very adept at identifying the gathering of active agents into an ordered, single cluster. When used with the mean density the random non ordered state could be identified, and thus an appreciation of the degree to which the grid had moved to or from a single, ordered cluster. The identification of patterns of clusters of active cells involved changing the context of the C-Value and moving it from a grid based view to a localised cluster one. This latter use of the C-Value, in the form of the LC-Value, has only been proved in a selective simulation and requires further work.

It is very difficult to separate a model from the specific purpose it was selected for and from the type of behaviour that was being investigated. A modeller of moving agents has different motives and objectives to a person who is modelling, for example, the spread of a virus. It is likely that the level of abstraction that would be required to achieve some real commonality would render the metric as unusable in a realistic, mechanistic context. It would remain conceptual. A conceptual commonality can exist across diverse systems. It may be in the form of the concept of measuring or a specific type of measurement, but the application is specific to the context and to the type of system. In this way the C-Value itself required a change in context to offer a way of identifying two different outputs.

Chapter 7

Conclusions

This chapter compares the work documented in this thesis with the seven objectives outlined in chapter 1, highlighting achievements and limitations. This is followed by a statement on the contribution made by the thesis. The chapter concludes with some suggestions for future work.

7.1 Achievements and limitations

This section reviews the seven objectives set out in chapter 1 with reference to the work completed in the thesis.

Objective 1 - *to classify systems using 2D CA modelling by means of the type of model used and the characteristics of the observed output*
The diversity of systems modelled with 2D CA are shown in Table 2.1. [Ermentrout and Edelstein-Keshet, 1993] proposed three broad categories of CA models within biology, deterministic, particle and growth automaton. Their deterministic automaton excluded any use of probability in either the grid's updating scheme or the rules used to update a cell. A new category of randomised automata was proposed in subsection 2.6.5 as an enhancement to their work to cover CA using probability in its rules. This new category included CA models using asynchronous updating and / or probability in their updating rules. Objective 1 is fulfilled by the examples of the range of systems, including non-biological ones, in the four categories laid out in Table 2.2.

Objective 2 - *to establish any analogy between different systems using 2D CA models*
The enhancement of the work in [Ermentrout and Edelstein-Keshet, 1993] showed the range of biological and non-biological systems modelled with 2D CA. The di-

versity of systems is reflected within and across the four categories. The building of three models representing two of the original CA classification and the newly proposed randomised automaton showed the different uses of the same basic 2D output space, meeting the second objective. This proved a useful analogy between systems within the same type and different types in the context of a shared output space, whilst illustrating that the analogy also needs to take into some consideration the use the grid is being put to. Otherwise any commonality is restricted to the basic measurement of the grid itself, rather than the activity on it.

Objectives 3 and 6 are evaluated together as they were both achieved by the developing and testing of a new metric.

Objective 3 - *to develop the basis of a measuring technique that can be used with 2D CA output from different domains and with different expected output*

Objective 6 - *to evaluate the effectiveness of the measure and metric*

The idea and usage of the similar concepts of connectedness, connectivity and compactness were outlined in subsection 3.3.5. The new metric proposed, the connectedness C-Value, although developed separately and applied in a different context, draws on these examples of assessing and using the connectivity between elements. The C-Value is calculated by measuring the number of connections between active cells or agents and dividing it by the optimised number of possible connections for that number of cells. This was applied successfully against different types of output. It provided a very good measure of the gathering of agents on a grid into a single, compact cluster; as shown by the particle automaton in section 5.3 and the randomised automaton in section 5.4. The start of the gathering process from a random distribution of the active agents to a single, compact cluster was marked by the correlation of the C-Value with the mean density. The C-Value increased and moved away from the value of the mean density as the agents gathered together. The correlation with the mean density gave a clear indication of when the agents on a grid were moving from a random distribution into an ordered state of a single cluster, or oscillating in an unstable configuration. The C-Value was set up to evaluate the overall state of the active cells or agents on the grid. This prevented it from identifying a pattern of clustered cells, as seen in Figure 5.26(a). The shifting of the context of the C-Value from an overall grid view to a localised one based on the groups of connected cells identified on the grid, allowed the pattern to be identified by the localised C-Value (LC-Value). The LC-Value was explored in subsection 5.4.1, where its success was balanced against the need to apply further tests and tuning.

Objectives 4 and 5 are discussed together as they involved the use of the metrics selected to measure the output from programs coded for the thesis; the former involved scenarios and the latter models.

Objective 4 - *to evaluate the suitability of the measurement technique against various 2D CA output scenarios*

Objective 5 - *to construct models to simulate and measure the output of three types of CA models, deterministic, particle and randomised*

Three models from different CA categories were built and provided the variety of outputs to test the new metric against, thus satisfying objectives four and five. In chapter 4, two of the three models built to provide varied output for the C-Value to test were successfully validated against the findings in the work they originated from. They also included different boundary conditions, the use of probability in the updating rules and a variety of updating methods. The particle automaton model employed reactive-diffusion and chemotaxis on a grid using a null boundary condition to stimulate the decentralised gathering of agents on a grid. The randomised automaton model used asynchronous updating, probability and toroidal boundary conditions to generate patterns in a delta-notch signalling simulation. The deterministic automaton model simulated the theoretical abstraction of an active network. This simulation had the least to contribute and with its complicated structure was illustrative of how simpler models could provide better output. The deterministic model, because of the theoretical nature of the model and the original work, could not be successfully validated, whereas it was possible to validate the other two simulations in the manner outlined in the simulation methodology in chapter 3.

Objective 7 - *to establish if the new metric provides the basis of tracking changes in the overall state of the CA output from different domain models*

The results from the tests enabled the final objective to be met. The C-Value effectively measured the gathering of active agents on the output grids produced by different domain models. The correlation of the C-Value with the mean density to identify the random distribution of the active cells on a grid provided an extremely useful indicator of the state of the output space. The initial example of using the C-Value at a localised level showed that the C-Value has the potential to track more than the movement between a random distribution and a single, compact cluster.

7.2 Contribution

The models built and run in the thesis replicated previous work. The thesis adopted a simulation methodology in chapter 3 where the new versions of the models were validated against the relevant original work. In this way, any validation against a real life system was by proxy through the original work. In section 4.6 the particle automaton model confirmed the results of the coupling of reaction-diffusion and chemotaxis to model amoebae gathering together in a mound at times of food shortage found in [Fatès et al., 2008]. While in section 4.7 the randomised automaton model reproduced the behaviour observed in the delta-notch signalling simulation in [Cohen et al., 2011]. In this way the new models were not only themselves validated but also validated the simulation and findings of the original work. This meant that a new set of code had been implemented using the same rules and conditions to produce similar results. Thus any negative view of the epistemic opaqueness of the computer models or the potential for the computer simulations to produce artefacts was reduced and the epistemic worth of the original work increased.

The three models represented three facets of the modelling process. The particle automaton model uses a real life system as an analogy for studying the decentralised gathering problem, which in turn was of interest in robotics and the self-organising of autonomous robots. The randomised automaton model simulated the real life system of delta-notch signalling to study how skin pigmentation and the placement of hair follicles were generated. The deterministic automaton model could be seen as an opaque thought experiment based on a theoretical active network. Although the deterministic automaton model produced artefactual behaviour and failed to generate any significant output, it was important in the affirmation of the simulation methodology; its failure can be seen as a consequence of its inability to fulfil the validation process proposed by the simulation methodology. It was built on the basis of the active network modelled in [de Silva, 2004], which was itself a theoretical model. Consequently, there was no real life system used to validate either the new or the original model. Indeed, the deterministic automaton model highlights the pitfalls of a computer simulations as a thought experiment mentioned in subsection 2.5.2; the epistemic opaqueness of the computer simulation leads to artefacts and there is no means of validating the results against a real life system.

The thesis selected 2D CA as its modelling platform, as explained in subsection 2.6.1. The diverse range of systems that could be modelled using 2D CA were listed, showing its versatility as a tool to explore how systems conceptually interact and work. This work has shown that the classification of 2D CA model types in biology proposed by Ermentrout and Edelstein-Keshet [1993] can be enhanced to include other disciplines. This enhancement included a recommendation for a slight amendment to the definition of the classification of a particle automaton. This was the removal of the implication that a particle automaton included merely collision models where the collision invoke a state change. But it was a clarification rather than a change to the definition, as the original work contained examples that were not particle collision models. More importantly, a new classification, *randomised*, was proposed to cover a growing trend for models of cell state change that used asynchronous updating methods or probability in their updating rules, or a combination of both. This enhancement was required as the original deterministic classification was restricted to synchronous updating and allowed no use of probability.

2D CA is a very visual modelling tool, with the simulation played out on a 2D grid. The diversity of systems that use CA as a modelling environment means that they share the commonality of that 2D grid. The three models selected were built to provide different 2D grid based output that could then be measured. The models were all from different CA classifications, two represented original classifications and the third fell within the newly proposed randomised category. The models, supported by the environment built to analyse the models, successfully produced and tested the variety of CA output produced.

A new C-Value metric was proposed as one that might prove applicable across the different type of output generated by the model types used. The metric reflected some of the ideas behind the concept of connectivity, connectedness and compactness explored in subsection 3.3.5, especially the local density discussed in [Barnes, 1969]; although the C-Value was developed separately and was applied in a different context and environment and with none of the connotations of paths or passage of information that were part of Barnes's local density. The 2D CA grid can be used to model agent movement and clustering, as well as the changing state of 'static' cells on the grid. Each modeller will have their own reason for using the environment and a fairly good expectation of what they will see, and thus how it needs to be measured. The finding of a common metric that could successfully cover the whole range needs to add more value than a simple measure of the occupancy rate, such as provided by a mean density measure of the grid.

The new metric was based on measuring the number of connections or edges that existed between active or occupied cells on the grid. The distance, or range was a minimum of one and a maximum of three cells, using a Manhattan distance. A cell could only be allocated to one cluster and clusters formed under one range could not then be joined together by the processing of the next, higher range. The resulting number of edges was then divided by the ideal number of edges formed by a compressed cluster of the number of active cells on the grid. The results from the simulations were analysed in chapter 5 and discussed in section 6.6. Three key conclusions were reached:

(1) the C-Value approaches 1.0000 as the active agents on a grid moves to a single, ordered cluster,

(2) the active agents on a grid are farthest away from an ordered single cluster when the C-Value is approximately the value of the mean density, and

(3) the changing of the context of the C-Value from the grid to individual clusters and singletons on the gird produces a localised C-Value (LC-Value) that shows potential for identifying patterns of clusters.

The combination of (2) and (1) marks the range between a random distribution of active agents and their gathering into a single, compact cluster on the 2D CA grid.

7.3 Future work

The LC-Value was proposed and tested in subsection 5.4.1 and while the results were promising, there were a number limitations highlighted that merit further research. The analysis code written for the thesis incorporated the identification of the number of clusters and singletons on a grid, as well as the number of agents in each cluster that was then used to establish the cluster size diversity on a grid. The correlation of the number and sizes of clusters on the grid, and the number of singletons, along with an appropriate weighting system, could be used to enhance the LC-Value to give an indicator of cluster pattern formation on the grid and both its configuration and connectedness. Tsang [2000, p.130] defined the complexity of the system he was examining in terms of the cluster size diversity, but he saw the "diversity of patterns as a more general way to indicate the complexity of a system". In terms of a 2D CA grid space, this complexity can be better seen as the level of randomness of the distribution of the active agents or clusters of agents on the grid. In this way an improved LC-Value would provide a more comprehensive indicator of randomness and the loss of it than the C-Value. It would be useful then to take the LC-Value and adapt it and test it within a 3D CA environment. This would require much more computational resources than available for the research in this thesis, but in effect the underlying principles of C-Value connectedness would remain the same, but the neighbourhood would increase to incorporate all the 3D neighbours of an active cell.

References

Abbott, R. (2006). Emergence explained: Abstractions - Getting epiphenomena to do real work. *Complexity*, 12(1):13–26. Understanding Complex Systems Conference, Urbana Champaign, IL, May, 2005.

Abbott, R. (2009). The Reductionist Blind Spot. *Complexity*, 14(5):10–22.

Abdulla, T., Schleich, J., and Summers, R. (2012). Multiscale modelling of notch-mediated lateral induction. In *Biomedical and Health Informatics (BHI), 2012 IEEE-EMBS International Conference on*, pages 257–260. IEEE.

Adachi, S., Peper, F., and Lee, J. (2004). Computation by asynchronously updating cellular automata. *Journal of Statistical Physics*, 114(1):261–289.

Adamatzky, A. (1996). Voronoi-like partition of lattice in cellular automata. *Mathematical and Computer Modelling*, 23(4):51–66.

Adamatzky, A. and Holland, O. (1998). Phenomenology of excitation in 2-d cellular automata and swarm systems. *Chaos, Solitons & Fractals*, 9(7):1233–1265.

Adams, K., Hester, P. T., Bradley, J. M., Meyers, T. J., and Keating, C. B. (2013). Systems theory as the foundation for understanding systems. *Systems Engineering*, 17(1):112–123.

Al-Ahmadi, K., See, L., Heppenstall, A., and Hogg, J. (2009). Calibration of a fuzzy cellular automata model of urban dynamics in Saudi Arabia. *Ecological Complexity*, 6(2):80–101.

Aljoufie, M., Zuidgeest, M., Brussel, M., van Vliet, J., and van Maarseveen, M. (2013). A cellular automata-based land use and transport interaction model applied to Jeddah, Saudi Arabia. *Landscape and Urban Planning*, 112:89–99.

Alonso-Sanz, R. and Bull, L. (2008). Elementary coupled cellular automata with memory. In Adamatzky, A., Alonso-Sanz, R., Lawniczak, A., Martinez, G., Morita, M., and Worsch, T., editors, *Automata-2008: Theory and Applications of Cellular Automata*, pages 72–98. Luniver Press, Frome UK.

Åm, O. (1994). Back to basics: introduction to systems theory and complexity. url: http://www.calresco.org/texts/backto.htm.

Anderson, P. W. et al. (1972). More is different. *Science*, 177(4047):393–396.

Avolio, M., Errera, A., Lupiano, V., Mazzanti, P., and Di Gregorio, S. (2010). Development and calibration of a preliminary cellular automata model for snow avalanches. In Bandini, S., Manzoni, S., Umeo, H., and Vizzari, G., editors, *Cellular Automata*, pages 83–94, Ascoli Piceno, Italy. 9th International Conference opn Cellular Automata for Research and Industry, ACRI 2010, Springer, Berlin.

Baas, N. (2009a). Extended Memory Evolutive Systems in a Hyperstructure Context. *Axiomathes*, 19(2):215–221.

Baas, N. (2009b). Hyperstructures, Topology and Datasets. *Axiomathes*, 19(3):281–295.

Baas, N., Ehresmann, A., and Vanbremeersch, J. (2004). Hyperstructures and memory evolutive systems. *International Journal of General Systems*, 33(5):553–568.

Bak, P. (1999). *How nature works: The Science of Self- Organized Criticality*. Copernicus, New York.

Baldwin, J. (2002). Computation versus simulation. url: http://www.math.uic.edu/jbaldwin/pub/cafom.ps.

Bandini, S., Bonomi, A., and Vizzari, G. (2010). What do we mean by asynchronous CA? A reflection on types and effects of asynchronicity. In *Cellular Automata*, pages 385–394. Springer, Berlin.

Barberousse, A., Franceschelli, S., and Imbert, C. (2009). Computer simulations as experiments. *Synthese*, 169(3):557–574.

Barnes, J. A. (1969). Graph theory and social networks: A technical comment on connectedness and connectivity. *Sociology*, 3(2):215–232.

Barnes, J. A. and Harary, F. (1983). Graph theory in network analysis. *Social Networks*, 5(2):235–244.

Barredo, J., Kasanko, M., McCormick, N., and Lavalle, C. (2003). Modelling dynamic spatial processes: simulation of urban future scenarios through cellular automata. *Landscape and urban planning*, 64(3):145–160.

Barrow, J. D. (2008). *New Theories of Everything: The Quest for Ultimate Explanation*. Oxford University Press, Oxford.

Barton, J. and Haslett, T. (2007). Analysis, synthesis, systems thinking and the scientific method: rediscovering the importance of open systems. *Systems Research & Behavioral Science*, 24(2):143 – 155.

Beautement, P. (2001). Exploiting the phenomena of emergence. Technical report, Distributed Technology Group, DERA, Saint Andrews Road, Great Malvern, Worcestershire, WR14 3PS.

Bedau, M. (2008). Downward causation and the autonomy of weak emergence. In Bedau, M. and Humphreys, P., editors, *Emergence: Contemporary readings in philosophy and science*, chapter 8, pages 155–188. MIT Press, Cambridge MA.

Behring, C., Bracho, M., Castro, M., and Moreno, J. (2000). An algorithm for robot path planning with cellular automata. In *ACRI*, pages 11–19. Citeseer.

Beisbart, C. (2012). How can computer simulations produce new knowledge? *European Journal for Philosophy of Science*, 2(3):395–434.

Benjaafar, S., Dooley, K., and Setyawan, W. (1997). Cellular Automata for Traffic Flow Modeling. Technical report, Center for Transportation Studies, University of Minnesota, Minneapolis.

Bernaschi, M., Succi, S., and Castiglione, F. (2000). Large-scale cellular automata simulations of the immune system response. *Physical Review E*, 61(2):1851–1854.

Berto, F. and Tagliabue, J. (2012). Cellular automata. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. http://plato.stanford.edu/archives/sum2012/entries/cellular-automata/, summer 2012 edition.

Boerlijst, M. C. and Hogeweg, P. (1991). Spiral wave structure in pre-biotic evolution: hypercycles stable against parasites. *Physica D: Nonlinear Phenomena*, 48(1):17–28.

Bonabeau, E. and Dessalles, J. L. (1997). Detection and emergence. *Intellectica*, 25(2):85–94.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*. Oxford University Press, New York.

Bortolussi, L. (2011). A short overview of kolmogorov complexity and its application for proving lower bounds of algorithms. url: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.113.7154&rep=rep1&type=pdf.

Boschetti, F., Fulton, E., Bradbury, R., and Symons, J. (2012). What is a model, why people don't trust them and why they should. In Raupach, M., McMichael, A., Finnigan, J., Manderson, L., and Walker, B., editors, *Negotiating our future: living scenarios for Australia to 2050*, volume 2, pages 107–119. Australian Academy of Science, Canberra.

Boschetti, F. and Gray, R. (2008). A Turing test for Emergence. In Prokopenko, M., editor, *Advances in applied self-organizing systems*, Advanced Information and Knowledge Processing, pages 349–364. Springer, London.

Boschetti, F., Prokopenko, M., Macreadie, I., and Grisogono, A. (2005). Defining and detecting emergence in complex networks. In Khosla, R., Howlett, R., and Jain, L., editors, *Knowledge-Based Intelligent Information and Engineering Systems, 9th International Conference, KES 2005, Melbourne, Australia*, volume 3684: Lecture Notes in Computer Science, pages 573–580. Springer, Berlin.

Bretschneider, T., Vasiev, B., and Weijer, C. J. (1997). A model for cell movement during dictyosteliummound formation. *Journal of Theoretical Biology*, 189(1):41–51.

Brodu, N. (2007). *Practical investigations of complex systems*. PhD thesis, The department of computer science and software engineering, Concordia University, Montréal, Québec, Canada.

Bryden, J. and Noble, J. (2006). Computational modelling, explicit mathematical treatments, and scientific explanation. *Artificial Life*, X:520–526.

Bull, L. and Alonso-Sanz, R. (2008). On coupling random Boolean networks. In Adamatzky, A., Alonso-Sanz, R., Lawniczak, A., Martinez, G., Morita, M., and Worsch, T., editors, *Automata-2008: Theory and Applications of Cellular Automata*, pages 292–304. Luniver Press, Frome UK.

Burstedde, C., Klauck, K., Schadschneider, A., and Zittartz, J. (2001). Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications*, 295(3-4):507 – 525.

Burzyński, M., Cudny, W., and Kosiński, W. (2004). Cellular automata: structures and some applications. *Journal of Theoretical and Applied Mechanics*, 42(3):461–482.

Bush, S. (2002). Active virtual network management prediction: complexity as a framework for prediction, optimization, and assurance. In *DARPA Active Networks Conference and Exposition, 2002. Proceedings*, pages 534–553. IEEE.

Caiazzo, A., Evans, D., Falcone, J., Hegewald, J., Lorenz, E., Stahl, B., Wang, D., Bernsdorf, J., Chopard, B., Gunn, J., et al. (2009). Towards a complex automata multiscale model of in-stent restenosis. *Computational Science–ICCS 2009*, pages 705–714.

Cakar, E., Mnif, M., Müller-Schloer, C., Richter, U., and Schmeck, H. (2007). Towards a quantitative notion of self-organisation. In *IEEE Congress on Evolutionary Computation*, pages 4222–4229. IEEE.

Camazine, S. (1991). Self-organizing pattern formation on the combs of honey bee colonies. *Behavioral ecology and sociobiology*, 28(1):61–76.

Çambel, A. (1993). *Applied chaos theory-A paradigm for complexity*. Academic Press, London.

CERN (2013). Physics. url: http://home.web.cern.ch/about/physics.

Chen, J. S., Gumbayan, A. M., Zeller, R. W., and Mahaffy, J. M. (2014). An expanded notch-delta model exhibiting long-range patterning and incorporating microrna regulation. *PLoS computational biology*, 10(6):e1003655.

Chen, L., Xu, X., Chen, Y., and He, P. (2004). A novel ant clustering algorithm based on cellular automata. In *Proceedings. IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2004.(IAT 2004).*, pages 148–154. IEEE.

Chen, Y. and Sundaram, H. (2005). Estimating complexity of 2d shapes. In *Multimedia Signal Processing, 2005 IEEE 7th Workshop on*, pages 1–4. IEEE.

Chen, Y. and Wang, J. (2007). Agent-based simulation of agricultural prices volatility using cellular automata. In *Second International Conference on Innovative Computing, Information and Control, 2007. ICICIC '07.*, pages 857–860, Kumamoto, Japan.

Cheng, J. and Masser, I. (2004). Understanding spatial and temporal processes of urban growth: cellular automata modelling. *Environment and Planning B*, 31(2):167–194.

Chitnis, A. B. (1995). The role of notch in lateral inhibition and cell fate specification. *Molecular and Cellular Neuroscience*, 6(4):311–321.

Chomsky, N. (1986). *Knowledge of language: Its nature, origin, and use*. Praeger Publishers, New York.

Chopard, B. and Droz, M. (1998). *Cellular automata modeling of physical systems*. Cambridge University Press, Cambridge.

Chopard, B., Dupuis, A., Masselot, A., and Luthi, P. (2002). Cellular automata and lattice Boltzmann techniques: An approach to model and simulate complex systems. *Advances in complex systems*, 5(02n03):103–246.

Chowdhury, D., Schadschneider, A., and Nishinari, K. (2005). Physics of transport and traffic phenomena in biology: from molecular motors and cells to organisms. *Physics of Life reviews*, 2(4):318–352.

Chu, D., Strand, R., and Fjelland, R. (2003). Theories of complexity. *Complexity*, 8(3):19–30.

Clarke, K. C. and Gaydos, L. J. (1998). Loose-coupling a cellular automaton model and gis: long-term urban growth prediction for san francisco and washington/baltimore. *International Journal of Geographical Information Science*, 12(7):699–714.

Clayton, P. and Davies, P. (2006). *The re-emergence of emergence: The emergentist hypothesis from science to religion*. Oxford University Press, New York.

Coe, J. B., Ahnert, S. E., and Fink, T. M. A. (2008). When are cellular automata random? *EPL (Europhysics Letters)*, 84:50005.

Cohen, M., Baum, B., and Miodownik, M. (2011). The importance of structured noise in the generation of self-organizing tissue patterns through contact-mediated cell–cell signalling. *Journal of The Royal Society Interface*, 8(59):787–798.

Cohen, M., Georgiou, M., Stevenson, N. L., Miodownik, M., and Baum, B. (2010). Dynamic filopodia transmit intermittent delta-notch signaling to drive pattern refinement during lateral inhibition. *Developmental Cell*, 19(1):78 – 89.

Collier, J., Monk, N., Maini, P., and Lewis, J. (1996). Pattern formation by lateral inhibition with feedback: a mathematical model of delta-notch intercellular signalling. *Journal of Theoretical Biology*, 183(4):429–446.

Cook, M. (2003). Still life theory. In Griffearth, D. and Moore, C., editors, *New constructions in cellular automata*, volume 226 of *Studies in the sciences of complexity*, pages 93–118. Oxford University Press, New York.

Corning, P. (2002). The re-emergence of "emergence": A venerable concept in search of a theory. *Complexity*, 7(6):18–30.

Couture, M. (2007a). Complexity and Chaos-State-of-the-Art; Formulations and Measures of Complexity. Technical Note 2006-451, Defence Research and Development Canada Valcartier (Quebec).

Couture, M. (2007b). Complexity and chaos-State-of-the-art; Overview of theoretical concepts. Technical Note 2006-450, Defence Research and Development Canada Valcartier (Quebec).

Crozatier, M., Glise, B., and Vincent, A. (2004). Patterns in evolution: veins of the drosophila wing. *TRENDS in Genetics*, 20(10):498–505.

Crutchfield, J. (1994). Is Anything Ever New? Considering Emergence. In Cowan, G., Pines, D., and Melzner, D., editors, *Complexity: Metaphors, Models, And Reality*, volume XIX of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 479–497. Addison-Wesley, Reading, MA.

Crutchfield, J., Farmer, J., Packard, N., and Shaw, R. (2008). Chaos. In Bedau, M. and Humphreys, P., editors, *Emergence: contemporary readings in philosophy and science*, chapter 21, pages 375–386. The MIT Press, Cambridge MA.

Dardashti, R., Thebault, K. P., and Winsberg, E. (2014). Confirmation via analogue simulation: What dumb holes could tell us about gravity. (unpublished). http://philsci-archive.pitt.edu/10955/1/Dumbholes_Archive.pdf.

Davies, P. (2006). The physics of downward causation. In Clayton, P. and Davies, P., editors, *The re-emergence of emergence: The emergentist hypothesis from science to religion*, chapter 2, pages 35–52. Oxford University Press, New York.

de Lope, J. and Maravall, D. (2013). Data clustering using a linear cellular automata-based algorithm. *Neurocomputing*, 114(0):86 – 91.

De Sales, J., Martins, M., and Stariolo, D. (1997). Cellular automata model for gene networks. *Physical Review E*, 55(3):3262–3270.

de Silva, M. S. (2004). *Emergence in Active Networks*. PhD thesis, Loughborough University.

De Wolf, T. and Holvoet, T. (2005). Emergence versus self-organisation: Different concepts but promising when combined. In Brueckner, S., Serugendo, G., Karageorgos, A., and Nagpal, R., editors, *Engineering self-organising systems: methodologies and applications*, volume 3464, pages 1–15. Springer-Verlag, Berlin.

Deacon, T. (2006). Emergence: The Hole at the Wheel's Hub. In Clayton, P. and Davies, P., editor, *The re-emergence of emergence: The emergentist hypothesis from science to religion*, chapter 5, pages 111–150. Oxford University Press, New York.

Decaestecker, C., Debeir, O., Van Ham, P., and Kiss, R. (2007). Can anti-migratory drugs be screened in vitro? a review of 2d and 3d assays for the quantitative analysis of cell migration. *Medicinal research reviews*, 27(2):149–176.

Di Paolo, E. A., Noble, J., and Bullock, S. (2000). Simulation models as opaque thought experiments. In Beadau, M., McCaskill, J., Packhard, N., and Rasmussen, S., editors, *Artificial life VII: Proceedings of the Seventh International Conference on the Simulation and Synthesis of Living Systems*, pages 497–506, Reed College, Portland, Oregon, USA. MIT Press, Cambridge, MA.

Dijkstra, J., Timmermans, H., and Jessurun, A. (2000). A multi-agent cellular automata system for visualising simulated pedestrian activity. *Theoretical and Practical Issues on Cellular Automata*, 10(4):6.

Ding, C. and He, X. (2004). K-nearest-neighbor consistency in data clustering: incorporating local information into global optimization. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 584–589. ACM.

Dubois, D. and Holmberg, S. (2010). Anticipatory artificial autopoiesis. In Trappl, R., editor, *Cybernetics and Systems*. Austrian Society for Cybernetic Studies, Vienna.

Dunne, J. A., Williams, R. J., and Martinez, N. D. (2002). Food-web structure and network theory: the role of connectance and size. *Proceedings of the National Academy of Sciences*, 99(20):12917–12922.

Dupuis, A. and Chopard, B. (2001). Parallel simulation of traffic in geneva using cellular automata. In Kuhn, E., editor, *Virtual shared memory for distributed architectures*, pages 89–107. Nova Science Publishers, Inc., New York.

Durán, J. M. (2014). *Explaining simulated phenomena: a defense of the epistemic power of computer simulations*. PhD thesis, Institute of Philosophy, Stuggart University.

Edelstein, L. (1982). The propagation of fungal colonies: a model for tissue growth. *Journal of Theoretical Biology*, 98(4):679–701.

Edelstein-Keshet, L. and Ermentrout, B. (1989). Models for branching networks in two dimensions. *SIAM Journal on Applied Mathematics*, 49(4):1136–1157.

Edmonds, B. (1999a). Pragmatic holism (or pragmatic reductionism). *Foundations of Science*, 4(1):57–82.

Edmonds, B. (1999b). *Syntactic measures of complexity*. PhD thesis, Department of Philosophy, University of Manchester.

Edmonds, B. (2007). The practical modelling of context-dependent causal processes–a recasting of Robert Rosen's thought. *Chemistry & Biodiversity*, 4(10):2386–2395.

Edmonds, B. (2010). Complexity and context-dependency. *Centre for Policy Modelling, Manchester Metropolitan University*.

Ellis, G. (2006). On the nature of emergent reality. In Clayton, P. and Davies, P., editors, *The re-emergence of emergence: The emergentist hypothesis from science to religion*, chapter 4, pages 77–107. Oxford University Press, New York.

Ermentrout, G. and Edelstein-Keshet, L. (1993). Cellular automata approaches to biological modeling. *Journal of Theoretical Biology*, 160:97–97.

Essam, J. W. (1980). Percolation theory. *Reports on Progress in Physics*, 43(7):833.

Evans, D., Lawford, P., Gunn, J., Walker, D., Hose, D., Smallwood, R., Chopard, B., Krafczyk, M., Bernsdorf, J., and Hoekstra, A. (2008). The application of multiscale modelling to the process of development and prevention of stenosis in a stented coronary artery. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1879):3343.

Evans, T. M. and Marcus, J. M. (2006). A simulation study of the genetic regulatory hierarchy for butterfly eyespot focus determination. *Evolution & Development*, 8(3):273–283.

Fatès, N. (2013). A guided tour of asynchronous cellular automata. In Kari, J., Kutrib, M., and Malcher, A., editors, *Cellular Automata and Discrete Complex Systems*, volume 8155 of *Lecture Notes in Computer Science*, pages 15–30. Springer Berlin Heidelberg.

Fatès, N. A. et al. (2008). Gathering agents on a lattice by coupling reaction-diffusion and chemotaxis. Technical report, Technical report, INRIA Nancy Grand-Est, 2008. http://hal.inria.fr/inria-00132266/.

Flake, G. (1999). *The Computational Beauty of Nature*. The MIT Press, Cambridge MA.

Forrester, J. W. (1971). Counterintuitive behavior of social systems. *Theory and Decision*, 2(2):109–140.

Franca, R., Marietto, M., de Santana, W., and Kobayashi, G. (2009). An agent-based simulation model for pedestrian unidirectional movement. In *Second International Conference on the Applications of Digital Information and Web Technologies, 2009. ICADIWT'09.*, pages 375–379. IEEE.

Fredkin, E. (1990). An informational process based on reversible universal cellular automata. *Physica D: Nonlinear Phenomena*, 45(1-3):254 – 270.

Fromm, J. (2005a). Ten questions about emergence. *Arxiv preprint nlin/0509049*.

Fromm, J. (2005b). Types and forms of emergence. *Arxiv preprint nlin/0506028*.

Fromm, J. (2006). On engineering and emergence. *Arxiv preprint nlin/0601002*.

Gammerman, A. and Vovk, V. (1999). Kolmogorov complexity: Sources, theory and applications. *The Computer Journal*, 42(4):252–255.

Garnier, S., Jost, C., Gautrais, J., Asadpour, M., Caprari, G., Jeanson, R., Grimal, A., and Theraulaz, G. (2008). The embodiment of cockroach aggregation behavior in a group of micro-robots. *Artificial Life*, 14(4):387–408.

Georgoudas, I., Sirakoulis, G., and Andreadis, I. (2007). Modelling earthquake activity features using cellular automata. *Mathematical and Computer Modelling*, 46(1-2):124–137.

Gerhardt, M., Schuster, H., and Tyson, J. J. (1990). A cellular automaton model of excitable media: Iii. fitting the belousov-zhabotinskii reaction. *Physica D: Nonlinear Phenomena*, 46(3):416 – 426.

Gershenson, C. (2002). Classification of random boolean networks. In Standish, R., Bedau, M., and Abbass, H., editors, *Artificial life VIII: Proceedings of the Eighth International Conference of Artificial Life.*, chapter 2, pages 1–8. MIT Press, Cambridge MA.

Gershenson, C. (2004). Introduction to random boolean networks. *Arxiv preprint nlin/0408006*.

Gershenson, C. (2005). A General Methodology for Designing Self-Organizing Systems. *eprint arXiv:nlin/0505009*.

Gershenson, C. (2007). *Design and control of self-organizing systems*. PhD thesis, Faculteit Wetenschappen and Center Leo Apostel for Interdisciplinary Studies, University of Brussels, Belgium.

Gershenson, C., Broekaert, J., and Aerts, D. (2003). Contextual random Boolean networks. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J., and Ziegler, J., editors, *Advances in Artificial Life, 7th European Conference, ECAL 2003, LNA1 2801*, pages 615–624. Springer, Berlin.

Gershenson, C. and Heylighen, F. (2003). When can we call a system self-organizing? In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J., and Ziegler, J., editors, *Advances in Artificial Life, 7th European Conference, ECAL 2003, LNA1 2801*, pages 606–614. Springer, Berlin.

Gershenson, C., LaViolette, R., Tolle, C., McJunkin, T., Stoner, D., Sergeev, S., Sreenivasan, K., Bershadskii, A., Niemela, J., Yulmetyev, R., et al. (2004). Updating schemes in Random Boolean Networks: do they really matter? In Pollack, J., Bedau, M., Husbands, P., Ikegami, T., , and Watson, R., editors, *Artificial life IX: proceedings of the Ninth International Conference on the Simulation and Synthesis of Artificial Life*, pages 238–243. MIT Press, Cambridge MA.

Gershenson, C., Leydesdorff, L., Altland, A., De Martino, A., Egger, R., Narozhny, B., Fontana, A., Medvedev, G., Pereira, T., and Brede, M. (2010). Guiding the Self-organization of Random Boolean Networks. *Arxiv preprint arXiv:1005.5733*.

Gershenson, C. and Rosenblueth, D. (2009). Modeling self-organizing traffic lights with elementary cellular automata. Technical Report C3 report 2009.06, Universidad Nacional Autonoma de Mexico.

Ghosh, R. and Tomlin, C. J. (2001). Lateral inhibition through delta-notch signaling: A piecewise affine hybrid model. In Di Benedetto, M. D., , and Sangiovanni-Vincentelli, A., editors, *Hybrid Systems: Computation and Control*, pages 232–246. Springer, Berlin.

Giere, R. N. (2009). Is computer simulation changing the face of experimentation? *Philosophical Studies*, 143(1):59–62.

Girau, B., Torres-Huitzil, C., Vlassopoulos, N., and Barrón-Zambrano, J. H. (2009). Reaction diffusion and chemotaxis for decentralized gathering on fpgas. *International Journal of Reconfigurable Computing*, 2009:12.

Glanville, R. (2002). Second order cybernetics. In *Encyclopaedia of Life Support Systems*. EoLSS Publishers, Oxford.

Gleick, J. (1987). *Chaos: making a new science*. Abacus, London.

Godfrey-Smith, P. (2006). The strategy of model-based science. *Biology and Philosophy*, 21(5):725–740.

Grassberger, P. (1999). Synchronization of coupled systems with spatiotemporal chaos. *Phys. Rev. E*, 59(3):R2520–R2522.

Gray, L. (2003). A mathematician looks at wolfram's new kind of science. *Notices-American Mathematical Society*, 50(2):200–211.

Greenwald, I. and Rubin, G. M. (1992). Making a difference: the role of cell-cell interactions in establishing separate identities for equivalent cells. *Cell*, 68(2):271–281.

Grilo, C. and Correia, L. (2008). The Influence of Asynchronous Dynamics in the Spatial Prisoner's Dilemma Game. *Lecture Notes in Computer Science*, 5040:362–371.

Gross, J. L. and Yellen, J. (2006). *Graph theory and its applications*. CRC press, Boca Raton, 2nd edition.

Grüne-Yanoff, T. and Weirich, P. (2010). The philosophy and epistemology of simulation: a review. *Simulation & Gaming*, 41(1):20–50.

Gruner, S. (2010). Mobile agent systems and cellular automata. *Autonomous Agents and Multi-Agent Systems*, 20(2):198–233.

Guala, F. (2002). Models, simulations, and experiments. In Magnani, L. and Nersessian, N., editors, *Model-Based Reasoning: Science, Technology, Values*, pages 59–74. Springer, New York.

Guisado, J., Jiménez-Morales, F., and Guerra, J. (2005). Application of shannon's entropy to classify emergent behaviors in a simulation of laser dynamics*. *Mathematical and Computer Modelling*, 42(7-8):847–854.

Hagiwara, T., Tode, H., and Ikeda, H. (2000). High-speed calculation method of the hurst parameter based on real traffic. In *Local Computer Networks, 2000. LCN 2000. Proceedings. 25th Annual IEEE Conference on*, pages 662–669, New York. IEEE.

Handl, J. and Knowles, J. (2007). An evolutionary approach to multiobjective clustering. *Evolutionary Computation, IEEE Transactions on*, 11(1):56–76.

Hart, D. and Craig-Hart, P. (2004). Reducing swarming theory to practice for UAV control. In *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, volume 5, pages 3050–3063 Vol.5.

Hatzikirou, H. and Deutsch, A. (2010). Lattice-Gas Cellular Automaton Modeling of Emergent Behavior in Interacting Cell Populations. In Hoekstra, A., Kroc, J., and Sloot, P., editors, *Simulating Complex Systems by Cellular Automata*, chapter 13, pages 301–331. Springer, Berlin.

Hegewald, J. (2009). Developers Guide to the MUSCLE. url: https://www.irmb.bau.tu-bs.de/muscle/.

Hegselmann, R. and Flache, A. (1998). Understanding complex social dynamics: A plea for cellular automata based modelling. *Journal of Artificial Societies and Social Simulation*, 1(3):1.

Helvik, T. (2006). *Dynamical systems of interacting units: Information transport and higher order structures*. PhD thesis, Norwegian University of Science and Technology, Trondheim.

Heylighen, F. and Joslyn, C. (2001). Cybernetics and second order cybernetics. *Encyclopedia of physical science & technology*, 4:155–170.

Hitchins, D. (1996). Getting to grips with complexity. url: http://www.hitchins.net/profs-stuff/profs-books/getting-to-grip-with-comple/.

Hitchins, D. (2000). Getting to grips with complexity or... a theory of everything else... url: http://www.hitchins.net/profs-stuff/profs-books/getting-to-grip-with-comple/.

Hitchins, D. (2003a). System philosophy. url: http://www.hitchins.net/systems/systems-philosophy/.

Hitchins, D. K. (2003b). Systems philosophy. *Systems Engineering: A 21st Century Systems Methodology*, pages 1–29.

Hoekstra, A., Caiazzo, A., Lorenz, E., Falcone, J., and Chopard, B. (2010a). Complex Automata: Multi-scale Modeling with Coupled Cellular Automata. In Kroc, J., Sloot, P., and Hoekstra, A., editors, *Simulating Complex Systems by Cellular Automata*, chapter 3, pages 29–57. Springer, Berlin.

Hoekstra, A., Chopard, B., Lawford, P., Hose, R., Krafczyk, M., and Bernsdorf, J. (2006). Introducing complex automata for modelling multi-scale complex systems. In *Proceedings of European Conference on Complex Systems ECCS 2006.* European Complex Systems Society, Oxford, UK.

Hoekstra, A., Kroc, J., and Sloot, P. (2010b). Introduction to Modeling of Complex Systems Using Cellular Automata. In Hoekstra, A., Kroc, J., and Sloot, P., editors, *Simulating Complex Systems by Cellular Automata*, chapter 1, pages 1–16. Springer, Berlin.

Hoekstra, A., Lorenz, E., Falcone, J., and Chopard, B. (2007). Towards a Complex Automata formalism for multi-scale modeling. *Int. J. Mult. Comp. Eng*, 5:491–502.

Holland, J. (1998). *Emergence: from chaos to order.* Oxford University Press, Oxford.

Holland, J. H. (1995). *Hidden Order: How Adaptation Builds Complexity.* (Perseus, New York, 1996).

Hoshen, J., Berry, M., and Minser, K. (1997). Percolation and cluster structure parameters: The enhanced hoshen-kopelman algorithm. *Physical Review E*, 56(2):1455.

Hosseinie, R. and Mahzoon, M. (2011). Irreducibility and emergence in complex systems and the quest for alternative insights. *Complexity*, 17(2):10–18.

Huberman, B. A. and Glance, N. S. (1993). Evolutionary games and computer simulations. *Proceedings of the National Academy of Sciences*, 90(16):7716–7718.

Humphreys, P. (2011). Computational science and its effects. In Carrier, M. and Nordmann, A., editors, *Science in the Context of Application*, pages 131–142. Springer, Berlin.

Ioannidis, K., Sirakoulis, G., and Andreadis, I. (2008). A Cellular Automaton Collision-Free Path Planner Suitable for Cooperative Robots. In *Informatics, 2008. PCI'08. Panhellenic Conference on*, pages 256–260. IEEE.

Iordache, S. (2011). *Emergent Phenomena in Agent-Based Systems.* PhD thesis, Faculty of Automation and Computers, Politehnica University Bucharest, Romania.

Janssens, K. (2010). An introductory review of cellular automata modeling of moving grain boundaries in polycrystalline materials. *Mathematics and Computers in Simulation*, 80(7):1361–1381.

Juwono, T. (2012). *Computational Studies Of Lattice Gas Models*. PhD thesis, College of Arts and Sciences, The Florida State University, USA.

Kauffman, S. A. (1991). Antichaos and adaptation. *Scientific American*, 265(2):78.

Keane, T. (2011a). Combat modelling with partial differential equations. *Applied Mathematical Modelling*, 35(6):2723–2735.

Keane, T. (2011b). Partial differential equations versus cellular automata for modeling combat. *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, XX(X):1–14.

Kellert, S. (1993). *In the wake of chaos: Unpredictable order in dynamical systems*. new. University of Chicago press.

Kicheva, A., Cohen, M., and Briscoe, J. (2012). Developmental pattern formation: Insights from physics and biology. *Science*, 338(6104):210–212.

Kier, L. and Cheng, C. (2000). A cellular automata model of an anticipatory system. *Journal of Molecular Graphics and Modelling*, 18(1):29–32.

Kier, L. and Witten, T. (2005). Cellular automata models of complex biochemical systems. In Bonchev, D. and Rouvray, D., editors, *Complexity in chemistry, biology, and ecology*, chapter 6, pages 237–301. Springer-Verlag, New York.

Kier, L. B., Cheng, C.-K., and Testa, B. (1999). A cellular automata model of the percolation process. *Journal of Chemical Information and Computer Sciences*, 39(2):326–332.

Kim, J. (1999). Making sense of emergence. *Philosophical studies*, 95(1):3–36.

Kim, J. (2006). Emergence: Core ideas and issues. *Synthese*, 151:547–559.

Kineman, J. (2007). *Relational Complexity in Natural Science and the Design of Ecological Informatics*. PhD thesis, Graduate School of the University of Colorado.

Kineman, J. (2011a). R-theory: A further commentary on the synthesis of relational science. In *Proceedings of the 55th Annual Meeting of the ISSS*, volume 55.

Kineman, J. (2011b). Relational science: A synthesis. *Axiomathes*, pages 1–45.

Kineman, J. J. (2012). R-theory: A synthesis of robert rosen's relational complexity. *Systems Research & Behavioral Science*, 29(5):527 – 538.

Kitto, K. (2006). *Modelling and generating complex emergent behaviour*. PhD thesis, Flinders University, School of Chemistry, Physics and Earth Sciences, Adelaide, South Australia.

Kitto, K. (2008). High end complexity. *International Journal of General Systems*, 37(6):689–714.

Kroc, J., Sloot, P., and Hoekstra, A. (2010). *Simulating Complex Systems by Cellular Automata*. Springer, Berlin.

Kusumaatmaja, H. and Yeomans, J. (2010). Lattice Boltzmann Simulations of Wetting and Drop Dynamics. *Simulating Complex Systems by Cellular Automata*, pages 241–274.

Ladyman, J., Lambert, J., and Wiesner, K. (2013). What is a complex system? *European Journal for Philosophy of Science*, 3(1):33–67.

Lam, L. (1998). *Nonlinear physics for beginners: fractals, chaos, solitons. pattern formation, cellular automata and complex systems*. World Scientific Publishing, Singapore.

Lambert, F. L. (2010). The Second Law of Thermodynamics. http://secondlaw.oxy.edu/.

Langton, C. G. (1990). Computation at the edge of chaos: phase transitions and emergent computation. *Physica D: Nonlinear Phenomena*, 42(1):12–37.

Lau, M., Saul, G., Mitani, J., and Igarashi, T. (2010). Modeling-in-context: user design of complementary objects with a single photo. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*, pages 17–24. Eurographics Association.

Laughlin, R. and Pines, D. (2000). From the cover: The theory of everything. *Proceedings of the National Academy of Sciences of the United States of America*, 97(1):28.

Lawniczak, A., Wu, H., and Di Stefano, B. (2008). DDoS attack detection using entropy of packet traffic in CA like data communication network model. In Adamatzky, A., Alonso-Sanz, R., Lawniczak, A., Martinez, G., Morita, M., and Worsch, T., editors, *Automata-2008 Theory and Applications of Cellular Automata*, pages 573–584. Luniver Press, Frome UK.

Lee, J. and Peper, F. (2008). On Brownian cellular automata. In Adamatzky, A., Alonso-Sanz, R., Lawniczak, A., Martinez, G., Morita, M., and Worsch, T., editors, *Automata-2008: Theory and Applications of Cellular Automata*, pages 278–290. Luniver Press, Frome UK.

Lee, S., Feng, D., and Gooch, B. (2008). Automatic construction of 3d models from architectural line drawings. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, pages 123–130. ACM.

Li, M. and Vitányi, P. (2008). *An introduction to Kolmogorov complexity and its applications*. Springer-Verlag, New York Inc.

Lipson, H. and Shpitalni, M. (2007). Optimization-based reconstruction of a 3d object from a single freehand line drawing. In *ACM SIGGRAPH 2007 courses*, page 45. ACM.

Louie, A. (2007). A rosen etymology. *Chemistry & Biodiversity*, 4(10):2296–2314.

Louie, A. (2008). Functional entailment and immanent causation in relational biology. *Axiomathes*, 18:289–302. 10.1007/s10516-008-9047-y.

Louie, A. (2009). *More than life itself: a synthetic continuation in relational biology*. Ontos-Verlag, Frankfurt.

Lowe, D. G. (1987). Three-dimensional object recognition from single two-dimensional images. *Artificial intelligence*, 31(3):355–395.

Luthi, P. O., Chopard, B., Preiss, A., and Ramsden, J. J. (1998). A cellular automaton model for neurogenesis in drosophila. *Physica D: Nonlinear Phenomena*, 118(1Äì2):151 – 160.

Magg, S. and te Boekhorst, R. (2006). Interaction and emergent behaviour in heterogeneous groups of artificial agents. *Explorations in the Complexity of Possible Life–GWAL 2007*, pages 95–103.

Maignan, L. and Gruau, F. (2008a). A 1D cellular automaton that moves particles until regular spatial placement. In Adamatzky, A., Alonso-Sanz, R., Lawniczak, A., Martinez, G., Morita, M., and Worsch, T., editors, *Automata-2008: Theory and Applications of Cellular Automata*, pages 323–337. Luniver Press, Frome UK.

Maignan, L. and Gruau, F. (2008b). Integer Gradient for Cellular Automata: Principle and Examples. In *Proceedings of the 2008 Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops-Volume 00*, pages 321–325. IEEE Computer Society.

Makarenko, A., Krushinsky, D., and Goldengorin, B. (2008). Anticipation and delocalization in cellular models of pedestrian traffic. *Proc. INDS*, pages 61–64.

Mäki, U. (2005). Models are experiments, experiments are models. *Journal of Economic Methodology*, 12(2):303–315.

Makowiec, D. (2008). Cellular Automata Model of Cardiac Pacemaker. *Acta Physica Polonica B*, 39(5):1067.

Malamud, B. and Turcotte, D. (2000). Cellular-automata models applied to natural hazards. *Computing in Science & Engineering*, 2(3):42–51.

Mamei, M., Roli, A., and Zambonelli, F. (2005). Emergence and control of macro-spatial structures in perturbed cellular automata, and implications for pervasive computing systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 35(3):337–348.

Markus, M., Böhm, D., and Schmick, M. (1999). Simulation of vessel morphogenesis using cellular automata. *Mathematical Biosciences*, 156(1):191–206.

Marro, J., Torres, J., and Cortés, J. (2007). Chaotic hopping between attractors in neural networks. *Neural Networks*, 20(2):230–235.

Maturana, H. and Varela, F. (1980). *Autopoiesis and cognition: The realization of the living*, volume 42. D. Reidel Publishing Company, Dordrecht.

Meinhardt, H. (1976). Morphogenesis of lines and nets. *Differentiation*, 6(2):117–123.

Mendelson, B. (1990). *Introduction to topology*. Dover Publications, INC, New York, 3rd edition.

Mikulecky, D. C. (2001). The emergence of complexity: science coming of age or science growing old? *Computers & Chemistry*, 25(4):341 – 348.

Mitchell, M. (1996). Computation in cellular automata: A selected review. *Non-standard Computation*, pages 95–140.

Mitchell, M. (2009). *Complexity: a guided tour*. Oxford University Press, Oxford.

Mitchell, M., Crutchfield, J., and Hraber, P. (1994). Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D*, 75(1-3):361–391.

Moere, A. and Clayden, J. (2005). Cellular ants: combining ant-based clustering with cellular automata. In *17th IEEE International Conference on Tools with Artificial Intelligence, 2005. ICTAI 05.*, pages 8–184. IEEE.

Moere, A., Clayden, J., and Dong, A. (2006). Data Clustering and Visualization using Cellular Automata Ants. *AI 2006: Advances in Artificial Intelligence*, pages 826–836.

Morgan, M. S. (2005). Experiments versus models: New phenomena, inference and surprise. *Journal of Economic Methodology*, 12(2):317–329.

Mori, T., Kudo, K., Namagawa, Y., Nakamura, R., Yamakawa, O., Suzuki, H., and Uesugi, T. (1998). Edge of chaos in rule-changing cellular automata. *Physica D: Nonlinear Phenomena*, 116(3-4):275–282.

Morrison, M. (2009). Models, measurement and computer simulation: the changing face of experimentation. *Philosophical Studies*, 143(1):33–57.

Murray, P. J., Maini, P. K., and Baker, R. E. (2013). Modelling oscillator synchronisation during vertebrate axis segmentation. In Capasso, V., Gromov, M., Harel-Bellan, A., Morozova, N., and Pritchard, L., editors, *Pattern Formation in Morphogenesis*, pages 95–105. Springer, Berlin.

Nadin, M. (2010). Anticipation and dynamics: Rosen's anticipation in the perspective of time. *International Journal of General Systems*, 39(1):3–33.

Nadin, M. (2012). The anticipatory profile. an attempt to describe anticipation as process. *International Journal of General Systems*, 41(1):43–75.

Nagel, K. and Rickert, M. (2001). Parallel implementation of the transims microsimulation. *Parallel Computing*, 27(12):1611–1639.

Nagle, F. S. (2011). Speed-dependent cellular decision making during laterally-inhibited morphogenesis. Technical report, University College London.

Nannen, V. (2010). A short introduction to Kolmogorov complexity. *Arxiv preprint arXiv:1005.2400*.

Naumov, L., Hoekstra, A., and Sloot, P. (2011). Cellular automata models of tumour natural shrinkage. *Physica A: Statistical Mechanics and its Applications*, 390(12):2283–2290.

Newman, D. (1996). Emergence and strange attractors. *Philosophy of Science*, pages 245–261.

Newth, D. and Cornforth, D. (2009). Asynchronous spatial evolutionary games. *Biosystems*, 95(2):120–129.

Nicholls, A., MacCuish, N., and MacCuish, J. (2004). Variable selection and model validation of 2d and 3d molecular descriptors¿. *Journal of Computer-Aided Molecular Design*, 18(7-9):451–474.

Nomura, T. (2007). Category theoretical distinction between autopoiesis and (m, r) systems. *Advances in Artificial Life*, pages 465–474.

Oliveira, G., Macêdo, H., Branquinho, A., and Lima, M. (2008). A cryptographic model based on the pre-image calculus of cellular automata. *Automata-2008: Theory and Applications of Cellular Automata*, pages 139–155.

Oreskes, N., Shrader-Frechette, K., and Belitz, K. (1994). Verification, validation, and confirmation of numerical models in the earth sciences. *Science*, 263(5147):641.

Ottino, J. M. (2003). Complex systems. *AIChE Journal*, 49(2):292–299.

Ottino, J. M. (2004). Engineering complex systems. *Nature*, 427(6973):399–399.

Packard, N. and Wolfram, S. (1985). Two-dimensional cellular automata. *Journal of Statistical Physics*, 38(5):901–946.

Pan, Z. (2010). Emergence from Symmetry: A New Type of Cellular Automata. *Arxiv preprint arXiv:1003.3394*.

Parke, E. C. (2014). Experiments, simulations, and epistemic privilege (in press). *Philosophy of Science*.

Peay, E. R. (1980). Connectedness in a general model for valued networks. *Social Networks*, 2(4):385 – 410.

Pepikj, B., Stark, M., Gehler, P., and Schiele, B. (2015). Multi-view and 3d deformable part models. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, pages 1–14.

Peschard, I. (2012). Is simulation a substitute for experimentation? In Vaienti, S., editor, *Simulations and Networks*. Hermann, Paris (forthcoming). Draft under review, from http://philpapers.org/rec/PESISA.Lastaccessed08.01.2015.

Picioreanu, C., van Loosdrecht, M., and Heijnen, J. (1999). *Multidimensional modeling of biofilm structure*. Delft University of Technology.

Pintus, A. M., Pazzona, F. G., Demontis, P., and Suffritti, G. B. (2011). A parallelizable block cellular automaton for the study of diffusion of binary mixtures containing CO_2 in microporous materials. *Journal of Chemical Physics*, 135(12).

Pomeau, B. H. Y. and Frisch, U. (1986). Lattice-gas automata for the navier-stokes equation. *Phys. Rev. Lett*, 56(14):1505.

Poudel, M., Kaffa-Jackou, R., França, F., and Mora-Camino, F. (2009). Modelling of aircraft emergency evacuation: a multiagent approach. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, pages 1–5. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Prigogine, I., Stengers, I., and Pagels, H. R. (1985). Order out of chaos. *Physics Today*, 38:97.

Prokopenko, M., Boschetti, F., and Ryan, A. (2007). An information-theoretic primer on complexity, self-organisation and emergence. *Advances in Complex Systems*.

Radicchi, F., Vilone, D., and Meyer-Ortmanns, H. (2007). Phase transition between synchronous and asynchronous updating algorithms. *Journal of Statistical Physics*, 129(3):593–603.

Radtke, F., Wilson, A., and MacDonald, H. R. (2005). Notch signaling in hematopoiesis and lymphopoiesis: lessons from drosophila. *Bioessays*, 27(11):1117–1128.

Ramage, M. and Shipp, K. (2009). *Systems thinkers*. Springer, London.

Rasmussen, S., Baas, N., Mayer, B., Nilsson, M., and Olesen, M. (2001). Ansatz for dynamical hierarchies. *Artificial Life*, 7(4):329–353.

Reed, R. D. and Serfas, M. S. (2004). Butterfly wing pattern evolution is associated with changes in a notch/distal-less temporal pattern formation process. *Current Biology*, 14(13):1159–1166.

Reggia, J., Chou, H., and Lohn, J. (1998). Cellular automata models of self-replicating systems. *Advances in computers*, 47:141–184.

Reiter, C. (2005). A local cellular model for snow crystal growth. *Chaos, Solitons & Fractals*, 23(4):1111–1119.

Rickles, D., Hawe, P., and Shiell, A. (2007). A simple guide to chaos and complexity. *Journal of Epidemiology and Community Health*, 61(11):933–937.

Ronald, E., Sipper, M., and Capcarrère, M. (1999). Design, observation, surprise! a test of emergence. *Artificial Life*, 5(3):225–239.

Rosen, J. and Kineman, J. (2005). Anticipatory systems and time: a new look at rosennean complexity. *Systems Research and Behavioral Science*, 22(5):399–412.

Rosen, R. (1987). On complex systems. *European Journal of Operational Research*, 30(2):129 – 134.

Rosen, R. (1991). *Life itself: a comprehensive inquiry into the nature, origin, and fabrication of life*. Columbia University Press, New York.

Rosen, R. (2012). *Anticipatory systems: philosophical, mathematical, and methodological foundations*. Springer, New York, 2nd edition.

Ruhl, J. (2006a). Attributes of complex adaptive systems. url: http://www.jurisdynamics.blogspot.co.uk/2006/07/attributes-of-complex-adaptive-systems.html.

Ruhl, J. (2006b). What is a complex adaptive system. url: http://www.jurisdynamics.blogspot.com/2006/07/what-is-complex-adaptive-system.html.

Ryan, A. (2007). Emergence is coupled to scope, not level. *Complexity*, 13(2):67–77.

Saif, M. A. and Gade, P. M. (2009). The prisoner's dilemma with semi-synchronous updates: evidence for a first-order phase transition. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(07):P07023.

Sarkar, P. (2000). A brief history of cellular automata. *ACM Computing Surveys (CSUR)*, 32(1):107.

Savill, N. J. and Sherratt, J. A. (2003). Control of epidermal stem cell clusters by notch-mediated lateral induction. *Developmental Biology*, 258(1):141–153.

Schadschneider, A. (2001). Cellular automaton approach to pedestrian dynamics-theory. *Arxiv preprint cond-mat/0112117*.

Schaefer, L., Mackulak, G., Cochran, J., and Cherilla, J. (1998). Application of a general particle system model to movement of pedestrians and vehicles. In Medeiros, D., Watson, E., Carson, J., and Manivannan, M., editors, *Proceedings of the 1998 Winter Simulation Conference*, pages 1155–1160. IEEE.

Scheidler, A., Merkle, D., and Middendorf, M. (2006). Emergent sorting patterns and individual differences of randomly moving ant like agents. In *Explorations in the complexity of possible life: abstracting and synthesizing the principles of living systems: proceedings of the 7th German Workshop on Artificial Life, July 26-28, 2006, Jena, Germany*, pages 105–115. IOS press.

Schiff, J. (2007). *Cellular automata: a discrete view of the world*. John Wiley & Sons, Inc.,, Hoboken, New Jersey.

Schönfisch, B. and de Roos, A. (1999). Synchronous and asynchronous updating in cellular automata. *Biosystems*, 51(3):123 – 143.

Schrödinger, E. (1992). *What is life? With mind and matter and autobiographical sketches*. Cambridge University Press, Cambridge.

Schultz, M. and Fricke, H. (2010). Stochastic Transition Model for Discrete Agent Movements. *Cellular Automata*, pages 506–512.

Schultz, M., Kretz, T., and Fricke, H. (2010). Solving the Direction Field for Discrete Agent Motion. *Cellular Automata*, pages 489–495.

Schützhold, R. and Unruh, W. G. (2002). Gravity wave analogues of black holes. *Physical Review D*, 66(4):044019.

Senge, P. M. (2006). *The fifth discipline: The art & practice of the learning organization*. Random House Digital, Inc., 2nd edition.

Shaya, O. and Sprinzak, D. (2011). From notch signaling to fine-grained patterning: Modeling meets experiments. *Current Opinion in Genetics & Development*, 21(6):732–739.

Sigmund, K. (1993). *Games of life: explorations in ecology, evolution and behaviour*. Oxford University Press, Inc.

Simpson, J. A., Weiner, E. S., et al. (1989). *The Oxford english dictionary*, volume 2. Clarendon Press Oxford.

Sloot, P., Chen, F., and Boucher, C. (2002). Cellular automata model of drug therapy for HIV infection. *Cellular Automata*, pages 282–293.

Sloot, P. and Hoekstra, A. (2010). Multi-scale modelling in computational biomedicine. *Briefings in Bioinformatics*, 11:142–52.

Sloot, P., Kaandorpa, J., Hoekstra, A., and Overeinder, B. (1999). Distributed simulation with cellular automata: Architecture and applications. In *SOFSEM'99: Theory and Practice of Informatics*, volume 1725 of *Lecture Notes in Computer Science*, pages 203–248. Springer, Berlin.

Smal, E. (2008). *Automated brick sculpture construction*. PhD thesis, Stellenbosch: Stellenbosch University.

Spracklin, L. and Saxton, L. (2007). Filtering spam using kolmogorov complexity estimates. In *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, volume 1, pages 321 –328.

Standish, R. (2001). On complexity and emergence. *Arxiv preprint nlin/0101006*.

Stevens, A. (2000). A stochastic cellular automaton modeling gliding and aggregation of myxobacteria. *SIAM Journal on Applied Mathematics*, 61(1):172–182.

Strogatz, S. (1994). *Nonlinear dynamics and chaos: with applications to Physics, Biology, Chemistry and Engineering (studies on nonlinearity)*. Addison-Wesley, Reading, MA.

Swindale, N. (1980). A model for the formation of ocular dominance stripes. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 208(1171):243–264.

Tamulonis, C. V. T. (2013). *Cell-based models*. PhD thesis, Faculty of Science, University of Amsterdam.

Tian, C., Masry, M., and Lipson, H. (2009). Physical sketching: Reconstruction and analysis of 3d objects from freehand sketches. *Computer-Aided Design*, 41(3):147–158.

Tissera, P., Printista, M., and Errecalde, M. (2007). Evacuation simulations using cellular automata. *Journal of Computer Science and Technology*, 7(1):14–20.

Toffoli, T. (1984). Cellular automata as an alternative to (rather than an approximation of) differential equations in modeling physics. *Physica D: Nonlinear Phenomena*, 10(1-2):117 – 127.

Topa, P., Dzwinel, W., and Yuen, D. (2006). A multiscale cellular automata model for simulating complex transportation systems. *International Journal of Modern Physics C*, 17(10):1437–1460.

Torrens, P. (2000). How cellular models of urban systems work (1. theory). Technical report, Centre for Advanced Spatial Analysis, University College London. url: http://eprints.ucl.ac.uk/1371/1/paper28.pdf.

Tsang, I. (2000). *Pattern recognition and complex systems*. PhD thesis, Faculty of Science, University of Antwerp.

Tsang, I. and Tsang, I. (1999). Cluster size diversity, percolation, and complex systems. *Physical Review E*, 60(3):2684.

Tsompanas, M. and Sirakoulis, G. (2012). Modeling and hardware implementation of an amoeba-like cellular automaton. *Bioinspiration & Biomimetics*, 7(3):036013.

Ulanowicz, R. (2007). Emergence, naturally! *Zygon®*, 42(4):945–960.

Ulgen, O. M., Black, J. J., Johnsonbaugh, B., and Klungle, R. (1994). Simulation methodology: a practitioner's perspective. *International Journal of Industrial Engineering, Applications and Practice*, 1(2).

Umpleby, S. (2001). What comes after second order cybernetics? *Cybernetics & Human Knowing*, 8(3):87–89.

van der Grient, L. (2011). On the epistemology of computer simulations. Master's thesis, Utrecht University.

van Dyke Parunak, H. and Brueckner, S. (2001). Entropy and self-organization in multi-agent systems. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 124–130, New York, NY, USA. ACM.

Van Dyke Parunak, H. and Brueckner, S. (2001). Entropy and self-organization in multi-agent systems. In *Proceedings of the fifth international conference on Autonomous agents*, page 130. ACM.

Vichniac, G. Y. (1984). Simulating physics with cellular automata. *Physica D: Nonlinear Phenomena*, 10(1-2):96 – 116.

Vickers, G. (1983). *Human systems are different*. Harper and Row, London.

Walliser, B. (2009). Emergent phenomena. In Gérin, M. and Maurel, M., editors, *Origins of Life: Self-Organization and/or Biological Evolution?*, pages 95–104. EDP Sciences (origins-and-evolution.org). http://dx.doi.org/10.1051/orvie/2009008.

Wcisło, R., Dzwinel, W., and Yuen, D. (2009). A 3-D model of tumor progression based on complex automata driven by particle dynamics. *Journal of Molecular Modeling*, 15(12):1517–1539.

Wcisło, R., Dzwinel, W., Yuen, D., and Dudek, A. (2008). A new model of tumor progression based on the concept of complex automata driven by particle dynamics. Submitted to: IEEE-ACM Transactions on Computational Biology and Bioinformatics, November 2008.

Weimar, J. R., Tyson, J. J., and Watson, L. T. (1992). Third generation cellular automaton for modeling excitable media. *Physica D: Nonlinear Phenomena*, 55(3):328–339.

Weinberg, S. (2008). Newtonianism, Reductionism and the Art of Congressional Testimony. In Bedau, M. and Humphreys, P., editors, *Emergence: contemporary readings in philosophy and science*, chapter 18, page 345. The MIT Press, Cambridge MA.

Wilkinson, D. and Willemsen, J. F. (1983). Invasion percolation: a new form of percolation theory. *Journal of Physics A: Mathematical and General*, 16(14):3365.

Winsberg, E. (2009). A tale of two methods. *Synthese*, 169(3):575–592.

Winsberg, E. (2014). Computer simulations in science. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. http://plato.stanford.edu/archives/fall2014/entries/simulations-science/, fall 2014 edition.

Wolfram, S. (1983). Statistical mechanics of cellular automata. *Reviews of Modern Physics*, 55(3):601.

Wolfram, S. (1984a). Cellular automata as models of complexity. *Nature*, 311(5985):419–424.

Wolfram, S. (1984b). Computation theory of cellular automata. *Communications in Mathematical Physics*, 96(1):15–57.

Wolfram, S. (1994). *Cellular automata and complexity: collected papers*. Addison-Wesley, Reading, Mass.

Wolfram, S. (2002). *A new kind of science*. Wolfram Media Inc, Champaign, IL.

Wolfram, S. (2003). The Past and Future of Scientific Computing 2003. http://www.stephenwolfram.com/publications/recent/atanasoff/.

Wright, W., Smith, R., Danek, M., and Greenway, P. (2000). A measure of emergence in an adapting, multi-agent context. In Meyer, J., Berthoz, A., Floreano, D., Roitblat, H., and Wilson, S., editors, *Proceedings of the Sixth International Conference on the Simulation of Adaptive Behaviour, SAB 2000*, pages 20–27.

Wuensche, A. (1998). Discrete dynamical networks and their attractor basins. Online journal *Complexity International*, Vol 6. http://www.complexity.org.au/ci/vol06/wuensche/wuensche.html. Last checked 08.01.2015.

Wuensche, A. (2008). Encryption using cellular automata chain-rules. *Automata-2008: Theory and Applications of Cellular Automata*, pages 126–137.

Xiao, X., Shao, S., Ding, Y., Huang, Z., and Chou, K. (2006). Using cellular automata images and pseudo amino acid composition to predict protein subcellular location. *Amino Acids*, 30(1):49–54.

Yan, G. and Yuan, L. (2000). Lattice boltzmann simulation for the spiral waves in the excitable medium. *Communications in Nonlinear Science and Numerical Simulation*, 5(4):147–150.

Yang, J., Tang, G., Cao, M., and Zhu, R. (2013). An intelligent method to discover transition rules for cellular automata using bee colony optimisation. *International Journal of Geographical Information Science*, pages 1–16.

Young, D. A. (1984). A local activator-inhibitor model of vertebrate skin patterns. *Mathematical Biosciences*, 72(1):51–58.

Zambonelli, F., Roli, A., and Mamei, M. (2003). Dissipative cellular automata as minimalist distributed systems: A study on emergent behaviors. In *11th Euromicro Conference on Parallel Distributed and Network based Processing, Genoa (I)*.

Zavsek, S., Pezdich, J., Shilko, E., Dmitriev, A., and Dimaki, A. (2005). Application of hybrid cellular automaton approach for computer-aided examination and forecast of strength properties of heterogeneous coal-beds. In *Proc. 11th Int. Conf. on Fracture, Turin, Italy*, page 4502.

Zorzenon dos Santos, R. M. and Bernardes, A. T. (1995). The stable-chaotic transition on cellular automata used to model the immune repertoire. *Physica A: Statistical Mechanics and its Applications*, 219(1):1–12.

Zuse, K. (1970). Calculating space. *Massachusetts Institute of Technology Technical Translation AZT-70-164-GEMIT (Project MAC). Cambridge.*

Appendix A

Modelling and Simulation Framework

The range and combination of CAs that have been used to model complex systems is extensive. An obvious issue when attempting to model and compare different systems is to establish some measure of commonality in the abstraction and modelling process. The abstraction of a system into a combination of coupled or linked models reinforces the argument for some form of formal or semi formal framework; especially when the decomposition of a single model reveals differences in spatial and, or temporal scale between the sub-models.

In the case of the Active Network (AN) domain, (*see section 4.8*), there is a spatial difference between the Active Data Packages (ADP) and the Active Nodes (ANode); the ANodes themselves have a temporal difference in the processing of ADPs being passed through and ones being processed; there is also a potential spatial difference between a fine and a coarse grain view of the processing status of the ANodes. The UAV domain has spatial differences between the clustering of the agents in a swarm and the individual actions of each agent; in the temporal scale there is the different velocities of the agents and, most importantly, the underlying difference between the overview of the location of the agents and the radar-like processing that needs to take place at a much quicker temporal pace to maintain that overview.

This appendix looks at the EU funded COAST project[1], and its associate software toolkit in Java (MUSCLE[2]); this project has promoted Complex Cellular Automata (CxA) as a framework for abstracting and modelling systems along a spatial and temporal axis and with a mixture of coupled CA and other modelling systems

[1]Complex Automata Simulation Technique, see http://www.complex-automata.org/
[2]Multi-scale Coupling Library and Environment, see http://muscle.berlios.de

[Caiazzo et al., 2009; Evans et al., 2008; Hoekstra et al., 2010a, 2006, 2010b, 2007; Kroc et al., 2010; Sloot and Hoekstra, 2010; Sloot et al., 1999; Wcisło et al., 2008].

A.1 CxA

Modelling a complex system as a single CA model can be very restrictive when there is a large difference in the micro and macro scales. The main premise for CxA is that a single, or 'flat' model of a system, whether CA or agent-based, often masks key differences in spatial and, or temporal scales between the main processes of the system being modelled. A multi-scale system in CxA can be split into a series of single-scale CA that are connected across the spatial and temporal scales. CxA provides a formal definition, a scale separation map that identifies 5 classes of scale separation and a message passing paradigm[3].

A.1.1 CxA Definition

A tuple defines a CA [Kroc et al., 2010]

$$C = \{A(\triangle x, L, \triangle t, T), \mathcal{F}, \Phi, f_{init} \in \mathcal{F}, \mathbf{u}, \mathbf{O}\} \tag{A.1}$$

where A is the spatio-temporal domain made up of the spatial units $\triangle x$ in the region L and the time-step units of $\triangle t$ over the time period T, giving $T/\triangle t$ iterations. $\mathcal{F}$ represents all the possible state of each cell and is updated by rule $\Phi : \mathcal{F} \to \mathcal{F}$, which can be defined in terms of the lattice gas automata framework [Chopard and Droz, 1998] as *collision + propagation*,

$$\Phi = \mathbf{PCB}, \tag{A.2}$$

where $\mathbf{B}$ is the *boundary condition*. The initial conditions are provided by f_{init}; any additional information concerning the boundaries of A is provided by the boundary conditions of $\mathbf{PCB}$; $\mathbf{u}$ acts as the conduit on each iteration for information between the CA and its environment. $\mathbf{O}$ is the *observable* that specifies the quantity of interest to the computation of the CA. Equation A.1 means that a CxA can be defined as a graph $\mathcal{X} = (V, E)$ where V is the set of vertices and E the set of edges with the following properties:

- Each vertex is a CA, $C_i = \{A_i(\triangle x_i, L, \triangle t_i, T_i), \mathcal{F}_i, \Phi_i, f_{init,i} \in \mathcal{F}_i, \mathbf{u}_i, \mathbf{O}_i\}$; and

[3]this section is based on Hoekstra et al in [Hoekstra et al., 2010b, 2007], which should be referred to for full details of CxA)

- Each edge E_{ij} describes how information is exchanged between the coupled CA_i and CA_j.

A.1.2 The Scale Separation Map

The Scale Separation Map (SSM) is used to illustrate the decomposed subsystems into areas representing the relevant spatial and temporal scales. The greater the scale difference, the easier it is to identify the process' position on the SSM. Figure A.1 shows how the SSM is laid out. The hypothetical example on the right shows how the system can be decomposed across the scales such that the processes modelled could be operating across the micro-, meso- and micro scales. The interaction regions on the SSM can be seen in figure A.2. A process in the bottom

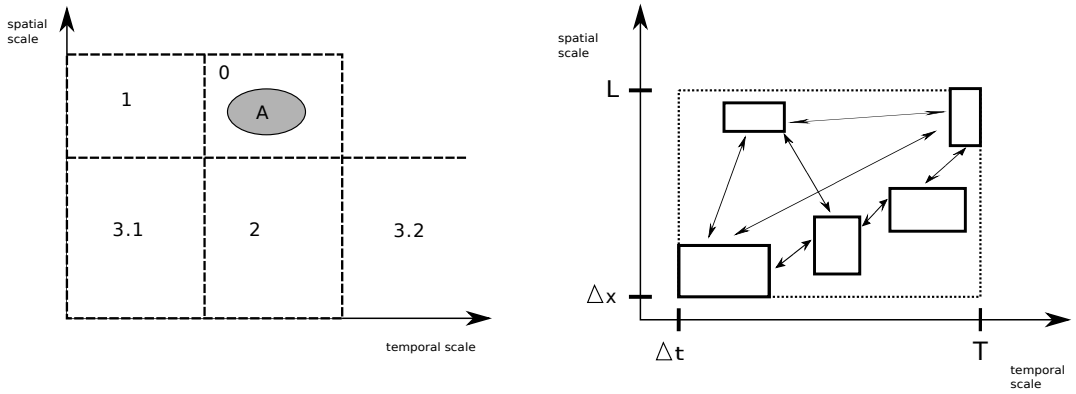


Figure A.1: Interaction regions on the scale map (left) and a hypothetical CxA with 5 single scale CA modelling the same process (right) *(adapted from [Kroc et al., 2010])*

left (region 3.1) runs at smaller spatial and temporal scale than the process in the top right (region 0). In figure A.1 process A is in region 0; the scale relationship between A and a second process depends on where it is placed on the SSM:

Region 0: Here there is no scale separation - a single scale model is required;

Region 1: Here the spatial scale is the same, while there is a difference in the temporal scale;

Region 2: Here the temporal scale is the same, while the spatial scale differs;

Region 3.1: Both 3.1 and 3.2 have a different spatial and temporal scale. With 3.1 process B operates at a smaller temporal and spatial scale; this is a common micro $\Longleftrightarrow$ macro coupling; and

Region 3.2: With 3.2 B operates at a greater temporal scale and a smaller spatial scale.

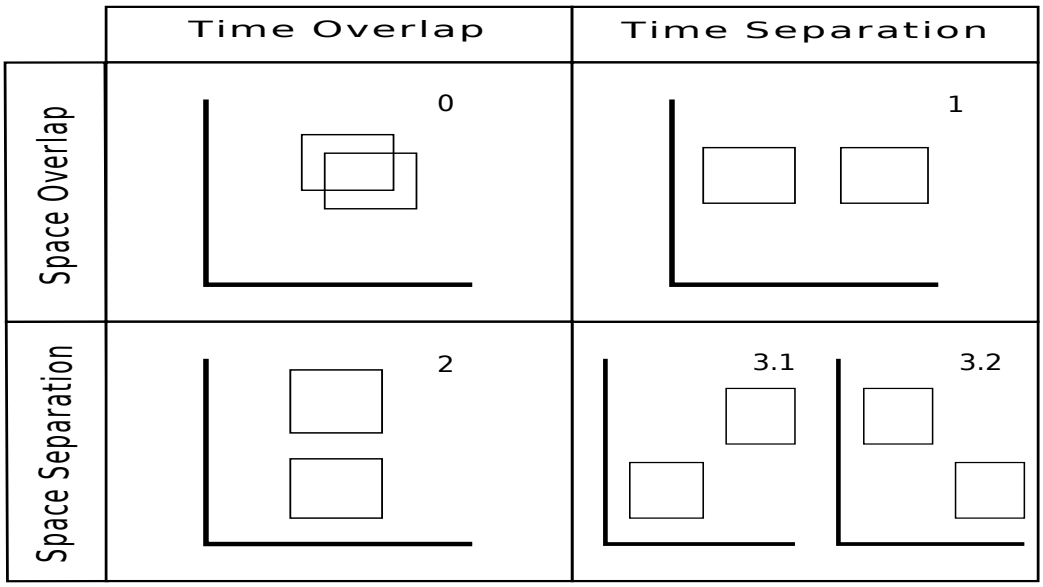


Figure A.2: Interaction regions on the SSM [Kroc et al., 2010]

A.1.3 Sub-model Execution Loop

It has been observed [Hoekstra et al., 2010b; Kroc et al., 2010] that the vertex of the CxA can be represented by a common instruction flow that uses the PCB for of the update rule (see equation A.2). The pseudo-code below, referred to as the Sub-model Execution Loop (SEL), has the operators written in bold and the state variables as plain text.

```
D := D_init          /* initialisation of the domain */
f  := f_init         /* initialisation of state variables */
t  := 0              /* initialisation of time */
    While Not EC     /* run until end condition reached */
        t+ = △t      /* increase time with one time step t */
        D := U(D)    /* update the domain */
        f  := B(f)   /* apply the boundary conditions */
        f  := C(f)   /* collision, update state of cells */
        f  := P(f)   /* propagation, send information to neighbours */
        O_i(f)       /* compute observable from new state */
    End              /* end of iteration loop*/
O_f(f)               /* compute observable from final state */
```

The operator **U** is used in instances where the domain is dynamic to, for example, add or delete cells. Operator **B** constructs any missing data that is required to complete the update of the boundary cells when operator **C** updates all the

cells. The propagation operator **P** sends information to neighbouring cells or agents. The observables are computed by **O** at the end of each iteration and at the termination of the main loop.

A further distinction raised by Hoekstra et al is whether the domain is a single (sD) or a multi-domain (mD):

> In case of sD processes A and B can access the whole simulated domain and communication can occur everywhere, whereas in case of mD each process is restricted to a different physical region and communication can only occur across an interface or small overlap region.
>
> [Hoekstra et al., 2010b]

The coupling between the sub-models in the CxA can be outlined in terms of linked SELs once the SSM has been drawn up. Some observations can be made in relationship of where sub-models are coupled [Kroc et al., 2010]:

- Time scale overlap coupling occurs inside the inner iteration loop;
- Time scale separation is coupled via the initialisation and final observation operators;
- Single-domain models are coupled through the collision operator; and
- Multi-domain models are coupled via the domain operator or the boundary operators.

Three modelling strategies have been observed: (1) time splitting - here process B in region 3.1 will cycle through more than one cycle of its internal loop per cycle of process A;(2) coarse graining - here A represents a coarse, less accurate representation of the system modelled by process B in region 2; and(3) amplification - here if B is a very slow process in 3.2, A is used to compress the view in terms of the temporal scale.

A.1.4 MUSCLE and the Execution Model

The Multi-scale Coupling Library and Environment (MUSCLE) is a Java based toolkit that provides the framework to build a CxA (see [Hegewald, 2009]). The key components of a MUSCLE built distributed CxA application are kernels, portals, conduits and a communication protocol. Each of the single scale sub-models identified in the SSM and represented by a SEL is built as a kernel; each kernel has an inlet and outlet portal; each portal is linked, (output $\Rightarrow$ input), via

a conduit. The conduit allows two types of communications, (1) a non-blocking, push mechanism, `send(data)`, and (2) a blocking, pull mechanism, `receive()`.

There can be only two conduits between two coupled kernels, but a kernel can be coupled to multiple kernels. Each conduit should be set up with a large buffer that acts on the FIFO basis. A conduit can have a filter on it that modifies the data to suit the requirements of the receiving kernel. The initialisation of a MUSCLE built CxA starts with the creation of the kernels and conduits; a special process termed *plumber* then connects all the conduits to the appropriate portals. The *plumber* then terminates and each kernel starts its computation; any `send(data)` calls will be buffered until the receiving kernel is connected while the sending kernel continues with its computations; a `receive()` in a kernel will cause the computation to be suspended until some data has been received. The conduits are robust, even if not connected at both ends, providing they have enough buffer storage.

Figure A.3 shows a hypothetical example where sub-model A is a process in region 0 of the SSM and sub-model B one in region 3.1, making it a time splitting micro (B) $\Longleftrightarrow$ macro (A) coupling where the time scale separation is coupled via the initialisation and final observation operators. After the *plumber* has connected the

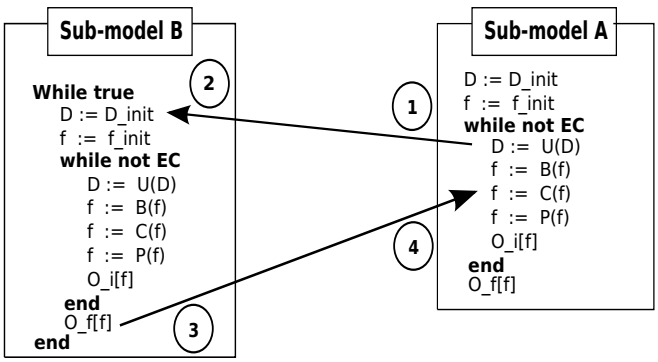


Figure A.3: Hypothetical coupling template for a time splitting modelling strategy *(adapted from [Kroc et al., 2010])*

CxA application up, both kernels (sub-models) A and B will start. Kernel B will pause at ② until kernel A sends some data from ①. Kernel A will then pause at ④ until B has completed its cycle of internal loops and sends the required information for A's update from ③; the interactive cycle between A and B is then repeated until A's end condition is met. It is important to note that a deadlock problem would occur if both A and B were set to receive data before sending any.

Appendix B

Test results

This appendix documents an expanded selection of grid displays, tables, and graphs relevant to those outlined or mentioned in chapter 5. The relevant source files, plus more data not illustrated below are contained on the CD that accompanies this thesis. The captions provide information on the parameters used in the run of the simulation.

B.1 Dynamic scenario

This section shows a range of results from tests run relating to the dynamic scenario model.

B.1.1 Dynamic scenario tables

The following are the tables used in the dynamic scenario tests using grids of four sizes, (25x25, 50x50, 75x75 and 100x100), each occupied with 350 agents.

Table B.1: Dynamic scenario with 25 by 25 grid and 350 agents.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.5574	0.0000	0.9216	0.5600
1 1M3	0.6369	0.1125	0.9216	0.5600
2 1M3	0.7456	0.1105	0.8832	0.5600
3 1M3	0.8458	0.0353	0.8832	0.5600
4 1M3	0.9221	0.0000	0.8832	0.5600
5 1M3	0.9676	0.0000	0.8832	0.5600
6 1M3	0.9915	0.0000	0.8832	0.5600
7 1M3	0.9931	0.0000	0.8832	0.5600
8 1M3	0.9931	0.0000	0.8832	0.5600

Table B.2: Dynamic scenario with 50 by 50 grid and 350 agents.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.1962	1.8404	0.9604	0.1400
1 1M3	0.1997	1.8817	0.8836	0.1400
2 1M3	0.2221	2.0705	0.8100	0.1400
3 1M3	0.2537	2.1331	0.7396	0.1400
4 1M3	0.3153	2.1877	0.6724	0.1400
5 1M3	0.3801	1.9450	0.6400	0.1400
6 1M3	0.4595	1.4453	0.6084	0.1400
7 1M3	0.5374	1.3517	0.5624	0.1400
8 1M3	0.5998	0.7826	0.5328	0.1400
9 1M3	0.6685	0.6552	0.5040	0.1400
10 1M3	0.7247	0.5253	0.4624	0.1400
11 1M3	0.7872	0.4917	0.4356	0.1400
12 1M3	0.8204	0.3718	0.4356	0.1400
13 1M3	0.8551	0.3061	0.4224	0.1400
14 1M3	0.8805	0.2419	0.4224	0.1400
15 1M3	0.9075	0.1254	0.4224	0.1400
16 1M3	0.9268	0.0355	0.4224	0.1400
17 1M3	0.9399	0.0000	0.4224	0.1400
18 1M3	0.9468	0.0000	0.4224	0.1400
19 1M3	0.9507	0.0000	0.4224	0.1400
20 1M3	0.9507	0.0000	0.4224	0.1400

Table B.3: Dynamic scenario with 75 by 75 grid and 350 agents.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.1384	1.0953	0.9735	0.0622
1 1M3	0.1430	1.1472	0.9216	0.0622
2 1M3	0.1527	1.3003	0.8711	0.0622
3 1M3	0.1812	1.4661	0.8220	0.0622
4 1M3	0.1966	1.5706	0.7744	0.0622
5 1M3	0.2113	1.6507	0.7396	0.0622
6 1M3	0.2429	1.8767	0.6944	0.0622

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
7 1M3	0.2683	1.7203	0.6507	0.0622
8 1M3	0.3045	1.8166	0.6084	0.0622
9 1M3	0.3323	1.9251	0.5877	0.0622
10 1M3	0.3562	1.7079	0.5476	0.0622
11 1M3	0.3824	1.8597	0.5182	0.0622
12 1M3	0.4217	1.4517	0.4896	0.0622
13 1M3	0.4703	1.3982	0.4530	0.0622
14 1M3	0.5027	1.3945	0.4352	0.0622
15 1M3	0.5482	0.9332	0.4178	0.0622
16 1M3	0.5875	1.0953	0.3920	0.0622
17 1M3	0.6345	0.8047	0.3746	0.0622
18 1M3	0.6731	0.5316	0.3593	0.0622
19 1M3	0.7140	0.6947	0.3353	0.0622
20 1M3	0.7440	0.4198	0.3200	0.0622
21 1M3	0.7726	0.3380	0.3058	0.0622
22 1M3	0.8049	0.2453	0.2981	0.0622
23 1M3	0.8250	0.2447	0.2843	0.0622
24 1M3	0.8412	0.2004	0.2843	0.0622
25 1M3	0.8643	0.1375	0.2843	0.0622
26 1M3	0.8689	0.0877	0.2843	0.0622
27 1M3	0.8843	0.0355	0.2843	0.0622
28 1M3	0.8890	0.0355	0.2843	0.0622
29 1M3	0.8990	0.0000	0.2843	0.0622
30 1M3	0.9082	0.0000	0.2843	0.0622
31 1M3	0.9113	0.0000	0.2843	0.0622
32 1M3	0.9113	0.0000	0.2843	0.0622

Table B.4: Dynamic scenario with 100 by 100 grid and 350 agents.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.1172	1.0443	0.9801	0.0350
1 1M3	0.1149	1.0947	0.9506	0.0350
2 1M3	0.1149	1.0510	0.9120	0.0350
3 1M3	0.1187	0.9993	0.8742	0.0350
4 1M3	0.1210	0.9707	0.8372	0.0350
5 1M3	0.1226	1.2094	0.8010	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
6 1M3	0.1280	1.2401	0.7656	0.0350
7 1M3	0.1396	1.3332	0.7310	0.0350
8 1M3	0.1542	1.6045	0.6972	0.0350
9 1M3	0.1658	1.6031	0.6642	0.0350
10 1M3	0.1781	1.6671	0.6320	0.0350
11 1M3	0.1904	1.7446	0.6006	0.0350
12 1M3	0.2136	1.7998	0.5775	0.0350
13 1M3	0.2282	2.0707	0.5475	0.0350
14 1M3	0.2452	2.0522	0.5256	0.0350
15 1M3	0.2706	2.1426	0.4970	0.0350
16 1M3	0.2992	2.1518	0.4830	0.0350
17 1M3	0.3231	2.1602	0.4556	0.0350
18 1M3	0.3315	2.0767	0.4290	0.0350
19 1M3	0.3631	2.0935	0.4160	0.0350
20 1M3	0.3978	1.9930	0.3906	0.0350
21 1M3	0.4248	1.3156	0.3720	0.0350
22 1M3	0.4541	1.1790	0.3540	0.0350
23 1M3	0.4734	1.3114	0.3306	0.0350
24 1M3	0.5127	0.9296	0.3136	0.0350
25 1M3	0.5443	1.1896	0.2916	0.0350
26 1M3	0.5875	0.8745	0.2756	0.0350
27 1M3	0.6361	0.8636	0.2652	0.0350
28 1M3	0.6469	0.6613	0.2550	0.0350
29 1M3	0.6739	0.5149	0.2400	0.0350
30 1M3	0.7032	0.4827	0.2352	0.0350
31 1M3	0.7232	0.4816	0.2162	0.0350
32 1M3	0.7440	0.4501	0.2115	0.0350
33 1M3	0.7579	0.3587	0.1978	0.0350
34 1M3	0.7787	0.3189	0.1978	0.0350
35 1M3	0.8057	0.2383	0.1886	0.0350
36 1M3	0.8157	0.1523	0.1840	0.0350
37 1M3	0.8358	0.0879	0.1840	0.0350
38 1M3	0.8520	0.0875	0.1840	0.0350
39 1M3	0.8635	0.0628	0.1840	0.0350
40 1M3	0.8774	0.0626	0.1840	0.0350
41 1M3	0.8820	0.0625	0.1840	0.0350
42 1M3	0.8867	0.0352	0.1840	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
43 1M3	0.8890	0.0000	0.1840	0.0350
44 1M3	0.8959	0.0000	0.1840	0.0350
45 1M3	0.8959	0.0000	0.1840	0.0350

The following are tables used in the dynamic tests using square grids 25, 50, 75 and 100 with 4, 8, 12 agents on.

Table B.5: Dynamic scenario with 25 by 25 grid and 8 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.1765	0.6730	0.6624	0.0128
1 1M3	0.1765	0.6730	0.5376	0.0128
2 1M3	0.1765	0.6730	0.4256	0.0128
3 1M3	0.1765	0.6730	0.3264	0.0128
4 1M3	0.1765	0.6730	0.2400	0.0128
5 1M3	0.1765	0.6730	0.1664	0.0128
6 1M3	0.3529	0.6365	0.1056	0.0128
7 1M3	0.3529	0.6365	0.0720	0.0128
8 1M3	0.4706	0.6931	0.0336	0.0128
9 1M3	0.5882	0.0000	0.0240	0.0128
10 1M3	0.9412	0.0000	0.0096	0.0128
11 1M3	1.0000	0.0000	0.0064	0.0128
12 1M3	1.0000	0.0000	0.0064	0.0128

Table B.6: Dynamic scenario with 50 by 50 grid and 8 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.0588	0.0000	0.5920	0.0032
1 1M3	0.0588	0.0000	0.5320	0.0032
2 1M3	0.0588	0.0000	0.4752	0.0032
3 1M3	0.0588	0.0000	0.4216	0.0032
4 1M3	0.0588	0.0000	0.3712	0.0032
5 1M3	0.0588	0.0000	0.3240	0.0032

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
6 1M3	0.0588	0.0000	0.2800	0.0032
7 1M3	0.0588	0.0000	0.2392	0.0032
8 1M3	0.0588	0.0000	0.2016	0.0032
9 1M3	0.0588	0.0000	0.1672	0.0032
10 1M3	0.1176	0.0000	0.1360	0.0032
11 1M3	0.1176	0.0000	0.1080	0.0032
12 1M3	0.1176	0.0000	0.0832	0.0032
13 1M3	0.2941	0.0000	0.0616	0.0032
14 1M3	0.2353	0.6730	0.0432	0.0032
15 1M3	0.3529	0.6365	0.0280	0.0032
16 1M3	0.3529	0.6365	0.0160	0.0032
17 1M3	0.3529	0.6365	0.0128	0.0032
18 1M3	0.5294	0.6616	0.0084	0.0032
19 1M3	0.6471	0.5623	0.0048	0.0032
20 1M3	0.7059	0.5623	0.0040	0.0032
21 1M3	0.8235	0.0000	0.0032	0.0032
22 1M3	0.9412	0.0000	0.0024	0.0032
23 1M3	1.0000	0.0000	0.0024	0.0032
24 1M3	1.0000	0.0000	0.0024	0.0032

Table B.7: Dynamic scenario with 75 by 75 grid and 8 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.0588	0.0000	0.5824	0.0014
1 1M3	0.0588	0.0000	0.5422	0.0014
2 1M3	0.0588	0.0000	0.5035	0.0014
3 1M3	0.0588	0.0000	0.4661	0.0014
4 1M3	0.0588	0.0000	0.4302	0.0014
5 1M3	0.0588	0.0000	0.3957	0.0014
6 1M3	0.0588	0.0000	0.3627	0.0014
7 1M3	0.0588	0.0000	0.3310	0.0014
8 1M3	0.0588	0.0000	0.3008	0.0014
9 1M3	0.0588	0.0000	0.2720	0.0014
10 1M3	0.0588	0.0000	0.2446	0.0014
11 1M3	0.0588	0.0000	0.2187	0.0014
12 1M3	0.1176	0.0000	0.1941	0.0014

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
13 1M3	0.0588	0.0000	0.1710	0.0014
14 1M3	0.0588	0.0000	0.1493	0.0014
15 1M3	0.0588	0.0000	0.1291	0.0014
16 1M3	0.0588	0.0000	0.1102	0.0014
17 1M3	0.0588	0.0000	0.0928	0.0014
18 1M3	0.1176	0.0000	0.0768	0.0014
19 1M3	0.1176	0.0000	0.0622	0.0014
20 1M3	0.1176	0.0000	0.0491	0.0014
21 1M3	0.1176	0.0000	0.0411	0.0014
22 1M3	0.1176	0.0000	0.0338	0.0014
23 1M3	0.1176	0.0000	0.0272	0.0014
24 1M3	0.1765	0.0000	0.0213	0.0014
25 1M3	0.2353	0.0000	0.0162	0.0014
26 1M3	0.3529	0.0000	0.0098	0.0014
27 1M3	0.4706	0.0000	0.0080	0.0014
28 1M3	0.6471	0.0000	0.0050	0.0014
29 1M3	0.7059	0.0000	0.0032	0.0014
30 1M3	0.8235	0.0000	0.0018	0.0014
31 1M3	0.8235	0.0000	0.0014	0.0014
32 1M3	0.8824	0.0000	0.0011	0.0014
33 1M3	1.0000	0.0000	0.0007	0.0014
34 1M3	1.0000	0.0000	0.0007	0.0014

Table B.8: Dynamic scenario with 100 by 100 grid and 8 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.0588	0.0000	0.2880	0.0008
1 1M3	0.0588	0.0000	0.2666	0.0008
2 1M3	0.0588	0.0000	0.2460	0.0008
3 1M3	0.0588	0.0000	0.2262	0.0008
4 1M3	0.1176	0.0000	0.2072	0.0008
5 1M3	0.0588	0.0000	0.1890	0.0008
6 1M3	0.0588	0.0000	0.1716	0.0008
7 1M3	0.0588	0.0000	0.1550	0.0008
8 1M3	0.0588	0.0000	0.1392	0.0008
9 1M3	0.0588	0.0000	0.1242	0.0008

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
10 1M3	0.0588	0.0000	0.1144	0.0008
11 1M3	0.0588	0.0000	0.1050	0.0008
12 1M3	0.0588	0.0000	0.0960	0.0008
13 1M3	0.0588	0.0000	0.0874	0.0008
14 1M3	0.1176	0.0000	0.0792	0.0008
15 1M3	0.1176	0.0000	0.0714	0.0008
16 1M3	0.1176	0.0000	0.0640	0.0008
17 1M3	0.1765	0.0000	0.0600	0.0008
18 1M3	0.1765	0.0000	0.0522	0.0008
19 1M3	0.1765	0.0000	0.0504	0.0008
20 1M3	0.1765	0.0000	0.0432	0.0008
21 1M3	0.1765	0.0000	0.0416	0.0008
22 1M3	0.1765	0.0000	0.0350	0.0008
23 1M3	0.1765	0.0000	0.0336	0.0008
24 1M3	0.1765	0.0000	0.0276	0.0008
25 1M3	0.1765	0.0000	0.0264	0.0008
26 1M3	0.1765	0.0000	0.0210	0.0008
27 1M3	0.1765	0.0000	0.0200	0.0008
28 1M3	0.1765	0.0000	0.0152	0.0008
29 1M3	0.1765	0.0000	0.0144	0.0008
30 1M3	0.2353	0.6829	0.0102	0.0008
31 1M3	0.1765	0.0000	0.0096	0.0008
32 1M3	0.3529	0.6365	0.0075	0.0008
33 1M3	0.3529	0.6365	0.0060	0.0008
34 1M3	0.4118	0.5983	0.0039	0.0008
35 1M3	0.5294	0.5983	0.0026	0.0008
36 1M3	0.5294	0.5983	0.0024	0.0008
37 1M3	0.5294	0.5983	0.0022	0.0008
38 1M3	0.5294	0.5983	0.0020	0.0008
39 1M3	0.5294	0.5983	0.0018	0.0008
40 1M3	0.5882	0.5623	0.0016	0.0008
41 1M3	0.7059	0.5623	0.0014	0.0008
42 1M3	0.7059	0.5623	0.0012	0.0008
43 1M3	0.7647	0.0000	0.0010	0.0008
44 1M3	0.8235	0.0000	0.0008	0.0008
45 1M3	1.0000	0.0000	0.0006	0.0008
46 1M3	1.0000	0.0000	0.0006	0.0008

Table B.9: Dynamic scenario with 25 by 25 grid and 12 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.0690	0.0000	0.6720	0.0192
1 1M3	0.0690	0.0000	0.5472	0.0192
2 1M3	0.0690	0.0000	0.4352	0.0192
3 1M3	0.1379	0.6829	0.3584	0.0192
4 1M3	0.1724	0.6931	0.2688	0.0192
5 1M3	0.1379	0.6829	0.2080	0.0192
6 1M3	0.2069	0.6730	0.1408	0.0192
7 1M3	0.3103	0.6890	0.0864	0.0192
8 1M3	0.5517	0.5623	0.0448	0.0192
9 1M3	0.7931	0.0000	0.0224	0.0192
10 1M3	0.8966	0.0000	0.0192	0.0192
11 1M3	0.9310	0.0000	0.0160	0.0192
12 1M3	1.0000	0.0000	0.0128	0.0192
13 1M3	1.0000	0.0000	0.0128	0.0192

Table B.10: Dynamic scenario with 50 by 50 grid and 12 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.0690	0.0000	0.7896	0.0048
1 1M3	0.0690	0.0000	0.7200	0.0048
2 1M3	0.0690	0.0000	0.6536	0.0048
3 1M3	0.0690	0.0000	0.5904	0.0048
4 1M3	0.0690	0.0000	0.5304	0.0048
5 1M3	0.0690	0.0000	0.4736	0.0048
6 1M3	0.0690	0.0000	0.4200	0.0048
7 1M3	0.0690	0.0000	0.3696	0.0048
8 1M3	0.0690	0.0000	0.3224	0.0048
9 1M3	0.0690	0.0000	0.2784	0.0048
10 1M3	0.1034	0.6730	0.2376	0.0048
11 1M3	0.2414	0.6829	0.2000	0.0048
12 1M3	0.2414	0.6365	0.1656	0.0048
13 1M3	0.2414	0.6931	0.1408	0.0048

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
14 1M3	0.2414	0.6931	0.1120	0.0048
15 1M3	0.3448	0.6870	0.0864	0.0048
16 1M3	0.3448	0.6870	0.0640	0.0048
17 1M3	0.3793	0.6730	0.0448	0.0048
18 1M3	0.4483	0.6730	0.0288	0.0048
19 1M3	0.4483	0.5297	0.0200	0.0048
20 1M3	0.6552	0.6365	0.0128	0.0048
21 1M3	0.7586	0.0000	0.0072	0.0048
22 1M3	0.9655	0.0000	0.0032	0.0048
23 1M3	1.0000	0.0000	0.0024	0.0048
24 1M3	1.0000	0.0000	0.0024	0.0048

Table B.11: Dynamic scenario with 75 by 75 grid and 12 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.0345	0.0000	0.5431	0.0021
1 1M3	0.0345	0.0000	0.5040	0.0021
2 1M3	0.0345	0.0000	0.4663	0.0021
3 1M3	0.0345	0.0000	0.4300	0.0021
4 1M3	0.0690	0.0000	0.3952	0.0021
5 1M3	0.0690	0.0000	0.3618	0.0021
6 1M3	0.0690	0.0000	0.3298	0.0021
7 1M3	0.0690	0.0000	0.2992	0.0021
8 1M3	0.0690	0.0000	0.2700	0.0021
9 1M3	0.0690	0.0000	0.2423	0.0021
10 1M3	0.1034	0.6730	0.2160	0.0021
11 1M3	0.1034	0.6730	0.1911	0.0021
12 1M3	0.1034	0.6730	0.1676	0.0021
13 1M3	0.1034	0.6730	0.1456	0.0021
14 1M3	0.1724	0.6365	0.1250	0.0021
15 1M3	0.2069	0.6365	0.1058	0.0021
16 1M3	0.2069	0.6365	0.0880	0.0021
17 1M3	0.2414	0.6931	0.0716	0.0021
18 1M3	0.3103	0.5623	0.0567	0.0021
19 1M3	0.4138	0.5623	0.0480	0.0021
20 1M3	0.4138	0.5623	0.0400	0.0021

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
21 1M3	0.4138	0.5623	0.0327	0.0021
22 1M3	0.4483	0.5297	0.0261	0.0021
23 1M3	0.5172	0.5297	0.0203	0.0021
24 1M3	0.5172	0.5297	0.0151	0.0021
25 1M3	0.5172	0.5297	0.0107	0.0021
26 1M3	0.5517	0.5004	0.0069	0.0021
27 1M3	0.6207	0.5004	0.0039	0.0021
28 1M3	0.7931	0.0000	0.0032	0.0021
29 1M3	0.8966	0.0000	0.0025	0.0021
30 1M3	0.8966	0.0000	0.0021	0.0021
31 1M3	0.8966	0.0000	0.0018	0.0021
32 1M3	1.0000	0.0000	0.0014	0.0021
33 1M3	1.0000	0.0000	0.0014	0.0021

Table B.12: Dynamic scenario with 100 by 100 grid and 12 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.0345	0.0000	0.6298	0.0012
1 1M3	0.0345	0.0000	0.5980	0.0012
2 1M3	0.0345	0.0000	0.5670	0.0012
3 1M3	0.0345	0.0000	0.5368	0.0012
4 1M3	0.0345	0.0000	0.5074	0.0012
5 1M3	0.0690	0.0000	0.4788	0.0012
6 1M3	0.0345	0.0000	0.4510	0.0012
7 1M3	0.0345	0.0000	0.4240	0.0012
8 1M3	0.0345	0.0000	0.3978	0.0012
9 1M3	0.0345	0.0000	0.3724	0.0012
10 1M3	0.0345	0.0000	0.3478	0.0012
11 1M3	0.0345	0.0000	0.3240	0.0012
12 1M3	0.0345	0.0000	0.3010	0.0012
13 1M3	0.0345	0.0000	0.2788	0.0012
14 1M3	0.0345	0.0000	0.2574	0.0012
15 1M3	0.0345	0.0000	0.2368	0.0012
16 1M3	0.0345	0.0000	0.2170	0.0012
17 1M3	0.1034	0.0000	0.1980	0.0012
18 1M3	0.1034	0.0000	0.1798	0.0012

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
19 1M3	0.1034	0.0000	0.1624	0.0012
20 1M3	0.1034	0.0000	0.1458	0.0012
21 1M3	0.1034	0.0000	0.1300	0.0012
22 1M3	0.1379	0.0000	0.1150	0.0012
23 1M3	0.1724	0.0000	0.1008	0.0012
24 1M3	0.2069	0.6365	0.0874	0.0012
25 1M3	0.2069	0.6365	0.0748	0.0012
26 1M3	0.2414	0.6931	0.0630	0.0012
27 1M3	0.3103	0.5983	0.0520	0.0012
28 1M3	0.4138	0.5623	0.0418	0.0012
29 1M3	0.4483	0.5297	0.0324	0.0012
30 1M3	0.5172	0.5297	0.0238	0.0012
31 1M3	0.5172	0.5297	0.0192	0.0012
32 1M3	0.5172	0.5297	0.0150	0.0012
33 1M3	0.5517	0.6555	0.0112	0.0012
34 1M3	0.6897	0.4741	0.0078	0.0012
35 1M3	0.7241	0.4741	0.0048	0.0012
36 1M3	0.7241	0.4741	0.0044	0.0012
37 1M3	0.7241	0.4741	0.0040	0.0012
38 1M3	0.7241	0.4741	0.0036	0.0012
39 1M3	0.7241	0.4741	0.0032	0.0012
40 1M3	0.7241	0.4741	0.0028	0.0012
41 1M3	0.7241	0.4741	0.0024	0.0012
42 1M3	0.7931	0.0000	0.0020	0.0012
43 1M3	0.8966	0.0000	0.0016	0.0012
44 1M3	0.8966	0.0000	0.0014	0.0012
45 1M3	0.8966	0.0000	0.0012	0.0012
46 1M3	0.9310	0.0000	0.0010	0.0012
47 1M3	1.0000	0.0000	0.0008	0.0012
48 1M3	1.0000	0.0000	0.0008	0.0012

B.1.2 Dynamic scenario neighbourhood simulations

The following are the tables for the test and graphs described in a subsection 5.2.2 and relate to the graphs shown in Figure 5.11.

Table B.13: Metrics values for 350 agents randomly placed on a 100 by 100 grid and moving towards the centre of the grid over a maximum of 50 time steps. A Moore neighbourhood was used with a search range depth of 1.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M1	0.0370	0.5591	0.9801	0.0350
1 1M1	0.0355	0.4041	0.9409	0.0350
2 1M1	0.0355	0.5222	0.9025	0.0350
3 1M1	0.0439	0.7921	0.8649	0.0350
4 1M1	0.0493	1.1676	0.8281	0.0350
5 1M1	0.0609	1.4366	0.7921	0.0350
6 1M1	0.0694	1.4133	0.7569	0.0350
7 1M1	0.0848	1.4483	0.7225	0.0350
8 1M1	0.0979	1.6111	0.6889	0.0350
9 1M1	0.1110	1.7586	0.6561	0.0350
10 1M1	0.1457	2.1241	0.6241	0.0350
11 1M1	0.1689	2.0943	0.6006	0.0350
12 1M1	0.1958	2.1600	0.5700	0.0350
13 1M1	0.2213	1.8547	0.5402	0.0350
14 1M1	0.2444	2.1969	0.5112	0.0350
15 1M1	0.2722	2.0639	0.4899	0.0350
16 1M1	0.2992	2.0261	0.4623	0.0350
17 1M1	0.3277	1.8005	0.4355	0.0350
18 1M1	0.3485	1.3504	0.4095	0.0350
19 1M1	0.3778	1.4585	0.3904	0.0350
20 1M1	0.4079	0.9938	0.3717	0.0350
21 1M1	0.4441	1.1558	0.3538	0.0350
22 1M1	0.4719	0.7892	0.3360	0.0350
23 1M1	0.5127	1.0608	0.3190	0.0350
24 1M1	0.5405	1.1439	0.2968	0.0350
25 1M1	0.5613	0.8118	0.2862	0.0350
26 1M1	0.5921	0.1701	0.2703	0.0350
27 1M1	0.6230	0.1876	0.2600	0.0350
28 1M1	0.6530	0.1659	0.2450	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
29 1M1	0.6793	0.1638	0.2401	0.0350
30 1M1	0.6993	0.1411	0.2304	0.0350
31 1M1	0.7371	0.0670	0.2209	0.0350
32 1M1	0.7633	0.0660	0.2116	0.0350
33 1M1	0.7895	0.0657	0.2116	0.0350
34 1M1	0.7980	0.0652	0.2070	0.0350
35 1M1	0.8126	0.0646	0.2070	0.0350
36 1M1	0.8258	0.0363	0.2070	0.0350
37 1M1	0.8327	0.0361	0.2070	0.0350
38 1M1	0.8466	0.0358	0.2070	0.0350
39 1M1	0.8520	0.0356	0.2070	0.0350
40 1M1	0.8612	0.0356	0.2070	0.0350
41 1M1	0.8689	0.0000	0.2070	0.0350
42 1M1	0.8705	0.0000	0.2070	0.0350
43 1M1	0.8766	0.0000	0.2070	0.0350
44 1M1	0.8820	0.0000	0.2070	0.0350
45 1M1	0.8836	0.0000	0.2070	0.0350
46 1M1	0.8843	0.0000	0.2070	0.0350
47 1M1	0.8843	0.0000	0.2070	0.0350

Table B.14: Metrics values for 350 agents randomly placed on a 100 by 100 grid and moving towards the centre of the grid over a maximum of 50 time steps. A von Neumann neighbourhood was used with a search range depth of 1.

ID	C-Value	Entropy	BBR	MeanDensity
0 1V1	0.0423	0.4556	0.9801	0.0350
1 1V1	0.0453	0.3326	0.9409	0.0350
2 1V1	0.0483	0.5627	0.9025	0.0350
3 1V1	0.0559	0.7418	0.8649	0.0350
4 1V1	0.0665	1.0043	0.8281	0.0350
5 1V1	0.0650	0.9649	0.7921	0.0350
6 1V1	0.0816	1.0499	0.7569	0.0350
7 1V1	0.0861	1.0861	0.7225	0.0350
8 1V1	0.1118	1.5158	0.6889	0.0350
9 1V1	0.1239	1.4537	0.6561	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
10 1V1	0.1677	1.9252	0.6241	0.0350
11 1V1	0.1767	1.7738	0.6006	0.0350
12 1V1	0.2175	1.8795	0.5700	0.0350
13 1V1	0.2341	1.8966	0.5402	0.0350
14 1V1	0.2613	1.9571	0.5112	0.0350
15 1V1	0.2825	1.9615	0.4899	0.0350
16 1V1	0.3202	1.9679	0.4623	0.0350
17 1V1	0.3505	2.0099	0.4355	0.0350
18 1V1	0.3595	2.0927	0.4095	0.0350
19 1V1	0.3943	1.8275	0.3904	0.0350
20 1V1	0.4245	1.9598	0.3717	0.0350
21 1V1	0.4713	1.7233	0.3538	0.0350
22 1V1	0.4985	1.6602	0.3360	0.0350
23 1V1	0.5378	1.5788	0.3190	0.0350
24 1V1	0.5755	1.5870	0.2968	0.0350
25 1V1	0.5937	0.8929	0.2862	0.0350
26 1V1	0.6148	0.4747	0.2703	0.0350
27 1V1	0.6495	0.2857	0.2600	0.0350
28 1V1	0.6858	0.3201	0.2450	0.0350
29 1V1	0.7100	0.1439	0.2401	0.0350
30 1V1	0.7266	0.2494	0.2304	0.0350
31 1V1	0.7568	0.0682	0.2209	0.0350
32 1V1	0.7840	0.0379	0.2116	0.0350
33 1V1	0.8157	0.0373	0.2116	0.0350
34 1V1	0.8233	0.0372	0.2070	0.0350
35 1V1	0.8399	0.0367	0.2070	0.0350
36 1V1	0.8535	0.0364	0.2070	0.0350
37 1V1	0.8625	0.0361	0.2070	0.0350
38 1V1	0.8761	0.0359	0.2070	0.0350
39 1V1	0.8837	0.0356	0.2070	0.0350
40 1V1	0.8943	0.0356	0.2070	0.0350
41 1V1	0.9018	0.0000	0.2070	0.0350
42 1V1	0.9018	0.0000	0.2070	0.0350
43 1V1	0.9079	0.0000	0.2070	0.0350
44 1V1	0.9139	0.0000	0.2070	0.0350
45 1V1	0.9154	0.0000	0.2070	0.0350
46 1V1	0.9169	0.0000	0.2070	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
47 1V1	0.9169	0.0000	0.2070	0.0350

Table B.15: Metrics values for 350 agents randomly placed on a 100 by 100 grid and moving towards the centre of the grid over a maximum of 50 time steps. A Moore neighbourhood was used with a search range depth of 2.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M2	0.0848	0.9626	0.9801	0.0350
1 1M2	0.0848	0.8996	0.9409	0.0350
2 1M2	0.0948	1.0464	0.9025	0.0350
3 1M2	0.0995	1.0811	0.8649	0.0350
4 1M2	0.1049	1.2291	0.8281	0.0350
5 1M2	0.1172	1.4139	0.7921	0.0350
6 1M2	0.1234	1.4720	0.7569	0.0350
7 1M2	0.1334	1.4136	0.7225	0.0350
8 1M2	0.1465	1.5497	0.6889	0.0350
9 1M2	0.1565	1.7364	0.6561	0.0350
10 1M2	0.1796	2.0078	0.6241	0.0350
11 1M2	0.2020	2.0696	0.6006	0.0350
12 1M2	0.2274	2.1315	0.5700	0.0350
13 1M2	0.2513	1.9543	0.5402	0.0350
14 1M2	0.2753	2.2555	0.5112	0.0350
15 1M2	0.2938	2.1681	0.4899	0.0350
16 1M2	0.3192	2.1421	0.4623	0.0350
17 1M2	0.3477	1.9843	0.4355	0.0350
18 1M2	0.3662	1.5763	0.4095	0.0350
19 1M2	0.3948	1.6741	0.3904	0.0350
20 1M2	0.4233	1.2305	0.3717	0.0350
21 1M2	0.4580	1.3677	0.3538	0.0350
22 1M2	0.4842	1.0246	0.3360	0.0350
23 1M2	0.5243	1.2624	0.3190	0.0350
24 1M2	0.5520	1.3388	0.2968	0.0350
25 1M2	0.5729	1.0290	0.2862	0.0350
26 1M2	0.6029	0.4070	0.2703	0.0350
27 1M2	0.6330	0.4022	0.2600	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
28 1M2	0.6631	0.3825	0.2450	0.0350
29 1M2	0.6870	0.3536	0.2401	0.0350
30 1M2	0.7055	0.2953	0.2304	0.0350
31 1M2	0.7402	0.1665	0.2209	0.0350
32 1M2	0.7656	0.1418	0.2116	0.0350
33 1M2	0.7911	0.1133	0.2116	0.0350
34 1M2	0.7995	0.1125	0.2070	0.0350
35 1M2	0.8134	0.0892	0.2070	0.0350
36 1M2	0.8258	0.0363	0.2070	0.0350
37 1M2	0.8327	0.0361	0.2070	0.0350
38 1M2	0.8466	0.0358	0.2070	0.0350
39 1M2	0.8520	0.0356	0.2070	0.0350
40 1M2	0.8612	0.0356	0.2070	0.0350
41 1M2	0.8689	0.0000	0.2070	0.0350
42 1M2	0.8705	0.0000	0.2070	0.0350
43 1M2	0.8766	0.0000	0.2070	0.0350
44 1M2	0.8820	0.0000	0.2070	0.0350
45 1M2	0.8836	0.0000	0.2070	0.0350
46 1M2	0.8843	0.0000	0.2070	0.0350
47 1M2	0.8843	0.0000	0.2070	0.0350

Table B.16: Metrics values for 350 agents randomly placed on a 100 by 100 grid and moving towards the centre of the grid over a maximum of 50 time steps. A von Neumann neighbourhood was used with a search range depth of 2.

ID	C-Value	Entropy	BBR	MeanDensity
0 1V2	0.0665	0.4725	0.9801	0.0350
1 1V2	0.0695	0.3983	0.9409	0.0350
2 1V2	0.0755	0.6699	0.9025	0.0350
3 1V2	0.0906	0.7360	0.8649	0.0350
4 1V2	0.0997	0.8932	0.8281	0.0350
5 1V2	0.1073	0.9366	0.7921	0.0350
6 1V2	0.1163	0.9854	0.7569	0.0350
7 1V2	0.1224	1.0104	0.7225	0.0350
8 1V2	0.1465	1.3655	0.6889	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
9 1V2	0.1541	1.4172	0.6561	0.0350
10 1V2	0.1858	1.8448	0.6241	0.0350
11 1V2	0.1994	1.6787	0.6006	0.0350
12 1V2	0.2356	1.8357	0.5700	0.0350
13 1V2	0.2477	1.8618	0.5402	0.0350
14 1V2	0.2779	1.9343	0.5112	0.0350
15 1V2	0.2991	1.9205	0.4899	0.0350
16 1V2	0.3278	1.9534	0.4623	0.0350
17 1V2	0.3625	1.9934	0.4355	0.0350
18 1V2	0.3746	2.0607	0.4095	0.0350
19 1V2	0.4033	1.8469	0.3904	0.0350
20 1V2	0.4320	1.9697	0.3717	0.0350
21 1V2	0.4834	1.7797	0.3538	0.0350
22 1V2	0.5076	1.6974	0.3360	0.0350
23 1V2	0.5423	1.6076	0.3190	0.0350
24 1V2	0.5801	1.6128	0.2968	0.0350
25 1V2	0.5982	0.9386	0.2862	0.0350
26 1V2	0.6193	0.5258	0.2703	0.0350
27 1V2	0.6526	0.3277	0.2600	0.0350
28 1V2	0.6888	0.3608	0.2450	0.0350
29 1V2	0.7130	0.1840	0.2401	0.0350
30 1V2	0.7296	0.2932	0.2304	0.0350
31 1V2	0.7598	0.1211	0.2209	0.0350
32 1V2	0.7840	0.0379	0.2116	0.0350
33 1V2	0.8157	0.0373	0.2116	0.0350
34 1V2	0.8233	0.0372	0.2070	0.0350
35 1V2	0.8399	0.0367	0.2070	0.0350
36 1V2	0.8535	0.0364	0.2070	0.0350
37 1V2	0.8625	0.0361	0.2070	0.0350
38 1V2	0.8761	0.0359	0.2070	0.0350
39 1V2	0.8837	0.0356	0.2070	0.0350
40 1V2	0.8943	0.0356	0.2070	0.0350
41 1V2	0.9018	0.0000	0.2070	0.0350
42 1V2	0.9018	0.0000	0.2070	0.0350
43 1V2	0.9079	0.0000	0.2070	0.0350
44 1V2	0.9139	0.0000	0.2070	0.0350
45 1V2	0.9154	0.0000	0.2070	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
46 1V2	0.9169	0.0000	0.2070	0.0350
47 1V2	0.9169	0.0000	0.2070	0.0350

Table B.17: Metrics values for 350 agents randomly placed on a 100 by 100 grid and moving towards the centre of the grid over a maximum of 50 time steps. A Moore neighbourhood was used with a search range depth of 3.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.1203	1.0918	0.9801	0.0350
1 1M3	0.1195	0.9928	0.9409	0.0350
2 1M3	0.1264	1.1306	0.9025	0.0350
3 1M3	0.1288	1.1135	0.8649	0.0350
4 1M3	0.1357	1.2178	0.8281	0.0350
5 1M3	0.1465	1.3778	0.7921	0.0350
6 1M3	0.1519	1.4425	0.7569	0.0350
7 1M3	0.1611	1.4358	0.7225	0.0350
8 1M3	0.1727	1.5525	0.6889	0.0350
9 1M3	0.1843	1.6722	0.6561	0.0350
10 1M3	0.2066	1.9245	0.6241	0.0350
11 1M3	0.2228	2.0195	0.6006	0.0350
12 1M3	0.2483	2.0725	0.5700	0.0350
13 1M3	0.2706	1.9655	0.5402	0.0350
14 1M3	0.2922	2.2292	0.5112	0.0350
15 1M3	0.3107	2.1461	0.4899	0.0350
16 1M3	0.3338	2.1480	0.4623	0.0350
17 1M3	0.3616	2.0147	0.4355	0.0350
18 1M3	0.3793	1.6386	0.4095	0.0350
19 1M3	0.4056	1.7238	0.3904	0.0350
20 1M3	0.4318	1.2985	0.3717	0.0350
21 1M3	0.4688	1.4425	0.3538	0.0350
22 1M3	0.4950	1.1367	0.3360	0.0350
23 1M3	0.5320	1.3427	0.3190	0.0350
24 1M3	0.5598	1.4151	0.2968	0.0350
25 1M3	0.5806	1.1226	0.2862	0.0350
26 1M3	0.6083	0.5084	0.2703	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
27 1M3	0.6369	0.4681	0.2600	0.0350
28 1M3	0.6669	0.4511	0.2450	0.0350
29 1M3	0.6893	0.3990	0.2401	0.0350
30 1M3	0.7078	0.3437	0.2304	0.0350
31 1M3	0.7425	0.2249	0.2209	0.0350
32 1M3	0.7672	0.1847	0.2116	0.0350
33 1M3	0.7926	0.1537	0.2116	0.0350
34 1M3	0.8011	0.1527	0.2070	0.0350
35 1M3	0.8150	0.1321	0.2070	0.0350
36 1M3	0.8273	0.0883	0.2070	0.0350
37 1M3	0.8335	0.0637	0.2070	0.0350
38 1M3	0.8473	0.0632	0.2070	0.0350
39 1M3	0.8527	0.0629	0.2070	0.0350
40 1M3	0.8620	0.0628	0.2070	0.0350
41 1M3	0.8689	0.0000	0.2070	0.0350
42 1M3	0.8705	0.0000	0.2070	0.0350
43 1M3	0.8766	0.0000	0.2070	0.0350
44 1M3	0.8820	0.0000	0.2070	0.0350
45 1M3	0.8836	0.0000	0.2070	0.0350
46 1M3	0.8843	0.0000	0.2070	0.0350
47 1M3	0.8843	0.0000	0.2070	0.0350

Table B.18: Metrics values for 350 agents randomly placed on a 100 by 100 grid and moving towards the centre of the grid over a maximum of 50 time steps. A von Neumann neighbourhood was used with a search range depth of 3.

ID	C-Value	Entropy	BBR	MeanDensity
0 1V3	0.0876	0.5297	0.9801	0.0350
1 1V3	0.0921	0.4753	0.9409	0.0350
2 1V3	0.0997	0.6919	0.9025	0.0350
3 1V3	0.1133	0.7362	0.8649	0.0350
4 1V3	0.1193	0.8874	0.8281	0.0350
5 1V3	0.1299	0.9150	0.7921	0.0350
6 1V3	0.1360	0.9616	0.7569	0.0350
7 1V3	0.1465	1.0114	0.7225	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
8 1V3	0.1631	1.3326	0.6889	0.0350
9 1V3	0.1737	1.3738	0.6561	0.0350
10 1V3	0.2100	1.7653	0.6241	0.0350
11 1V3	0.2130	1.6511	0.6006	0.0350
12 1V3	0.2523	1.8310	0.5700	0.0350
13 1V3	0.2674	1.8444	0.5402	0.0350
14 1V3	0.2946	1.9183	0.5112	0.0350
15 1V3	0.3127	1.9154	0.4899	0.0350
16 1V3	0.3429	1.9578	0.4623	0.0350
17 1V3	0.3776	2.0218	0.4355	0.0350
18 1V3	0.3867	2.0755	0.4095	0.0350
19 1V3	0.4169	1.8971	0.3904	0.0350
20 1V3	0.4426	2.0082	0.3717	0.0350
21 1V3	0.4909	1.8066	0.3538	0.0350
22 1V3	0.5166	1.7365	0.3360	0.0350
23 1V3	0.5483	1.6435	0.3190	0.0350
24 1V3	0.5846	1.6604	0.2968	0.0350
25 1V3	0.6027	0.9987	0.2862	0.0350
26 1V3	0.6208	0.5409	0.2703	0.0350
27 1V3	0.6541	0.3466	0.2600	0.0350
28 1V3	0.6903	0.3791	0.2450	0.0350
29 1V3	0.7145	0.2023	0.2401	0.0350
30 1V3	0.7311	0.3127	0.2304	0.0350
31 1V3	0.7613	0.1464	0.2209	0.0350
32 1V3	0.7855	0.0667	0.2116	0.0350
33 1V3	0.8157	0.0373	0.2116	0.0350
34 1V3	0.8233	0.0372	0.2070	0.0350
35 1V3	0.8399	0.0367	0.2070	0.0350
36 1V3	0.8535	0.0364	0.2070	0.0350
37 1V3	0.8625	0.0361	0.2070	0.0350
38 1V3	0.8761	0.0359	0.2070	0.0350
39 1V3	0.8837	0.0356	0.2070	0.0350
40 1V3	0.8943	0.0356	0.2070	0.0350
41 1V3	0.9018	0.0000	0.2070	0.0350
42 1V3	0.9018	0.0000	0.2070	0.0350
43 1V3	0.9079	0.0000	0.2070	0.0350
44 1V3	0.9139	0.0000	0.2070	0.0350

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
45 1V3	0.9154	0.0000	0.2070	0.0350
46 1V3	0.9169	0.0000	0.2070	0.0350
47 1V3	0.9169	0.0000	0.2070	0.0350

B.1.3 Dynamic scenario graphs

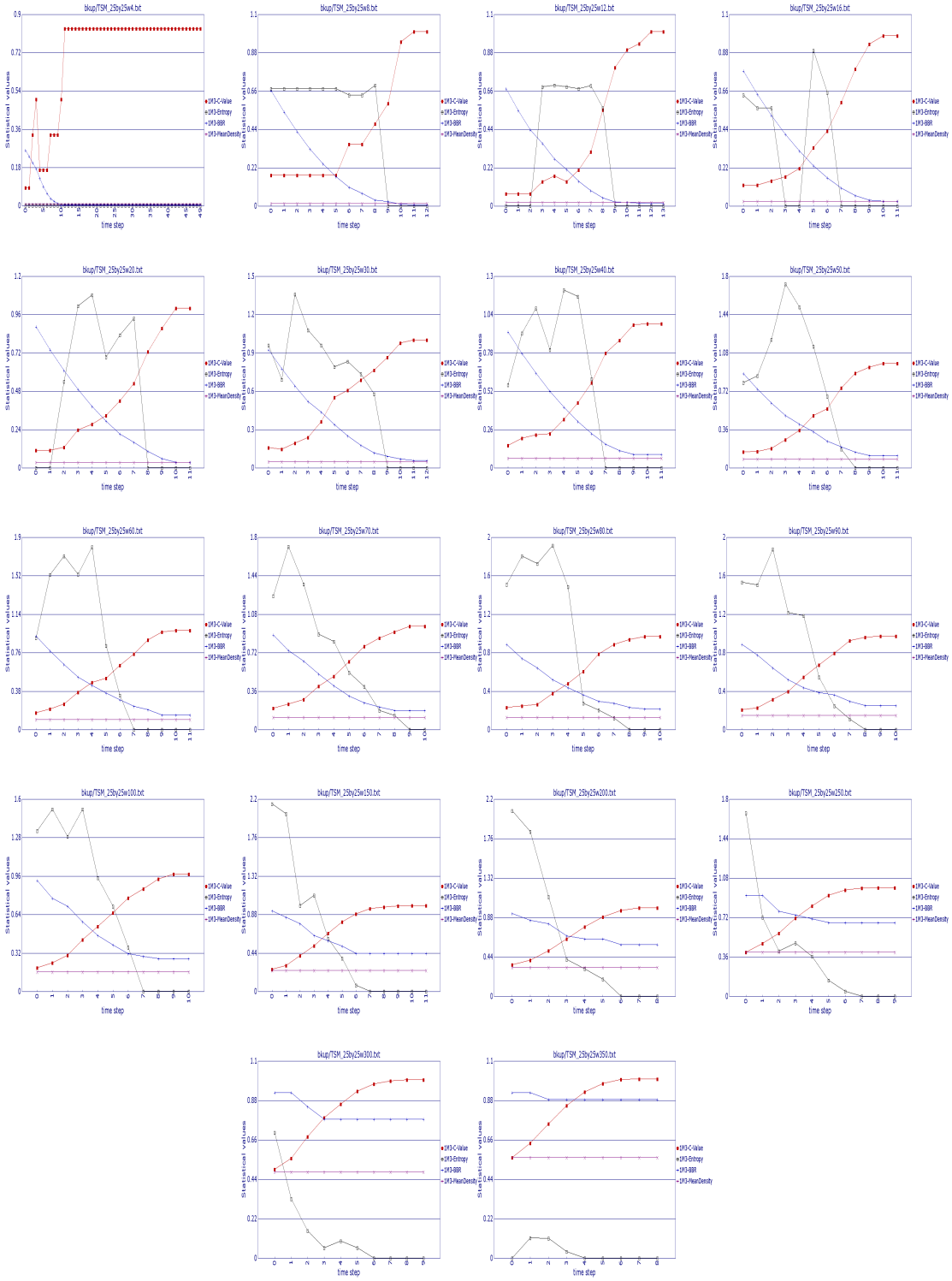


Figure B.1: Graphs showing the metric analysis of a 25 by 25 dynamic scenario run up to 50 time steps for agents of R={4, 8, 12, 16, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200, 250, 300, 350}

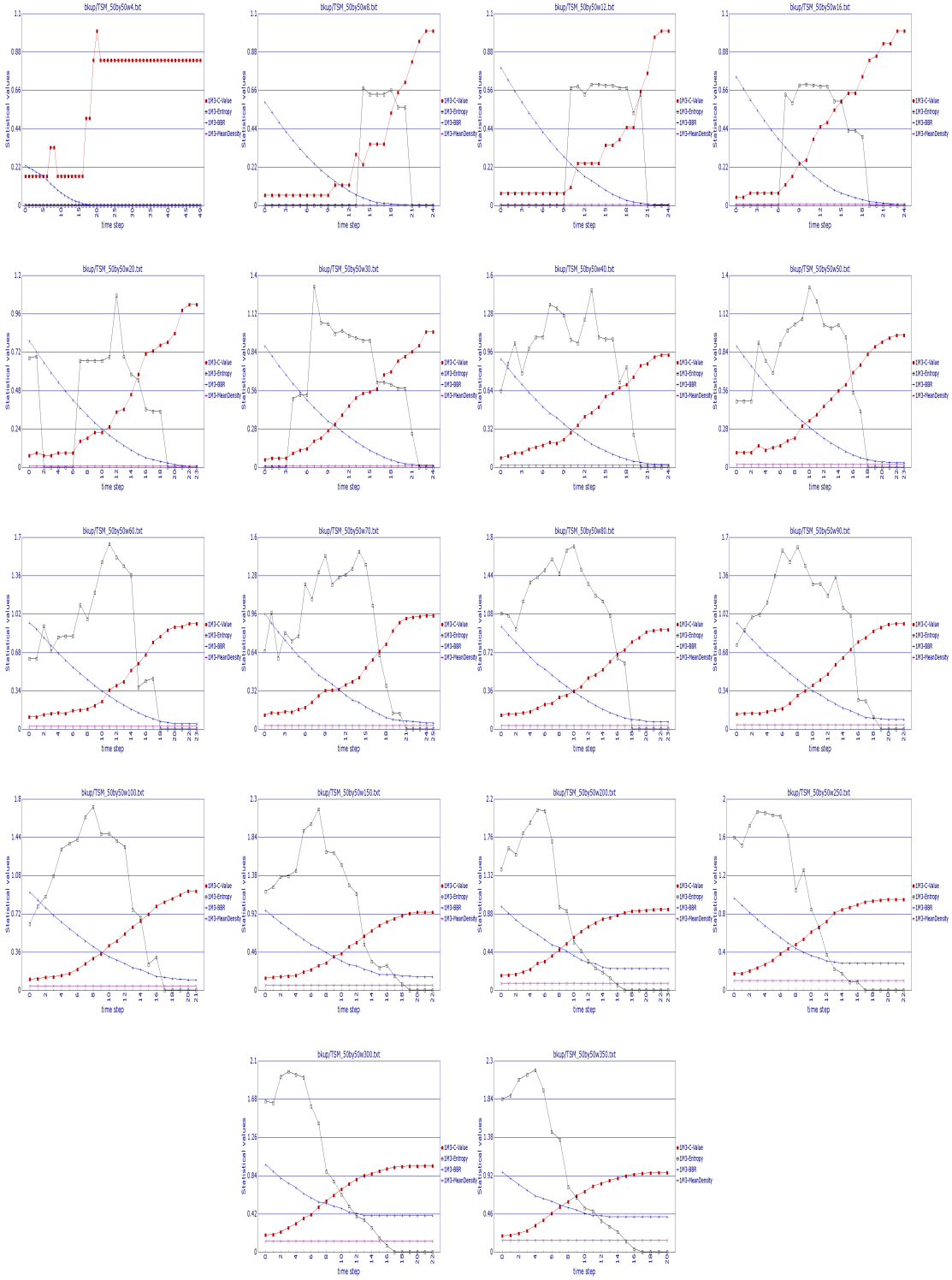


Figure B.2: Graphs showing the metric analysis of a 50 by 50 dynamic scenario run up to 50 time steps for agents of R={4, 8, 12, 16, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200, 250, 300, 350}

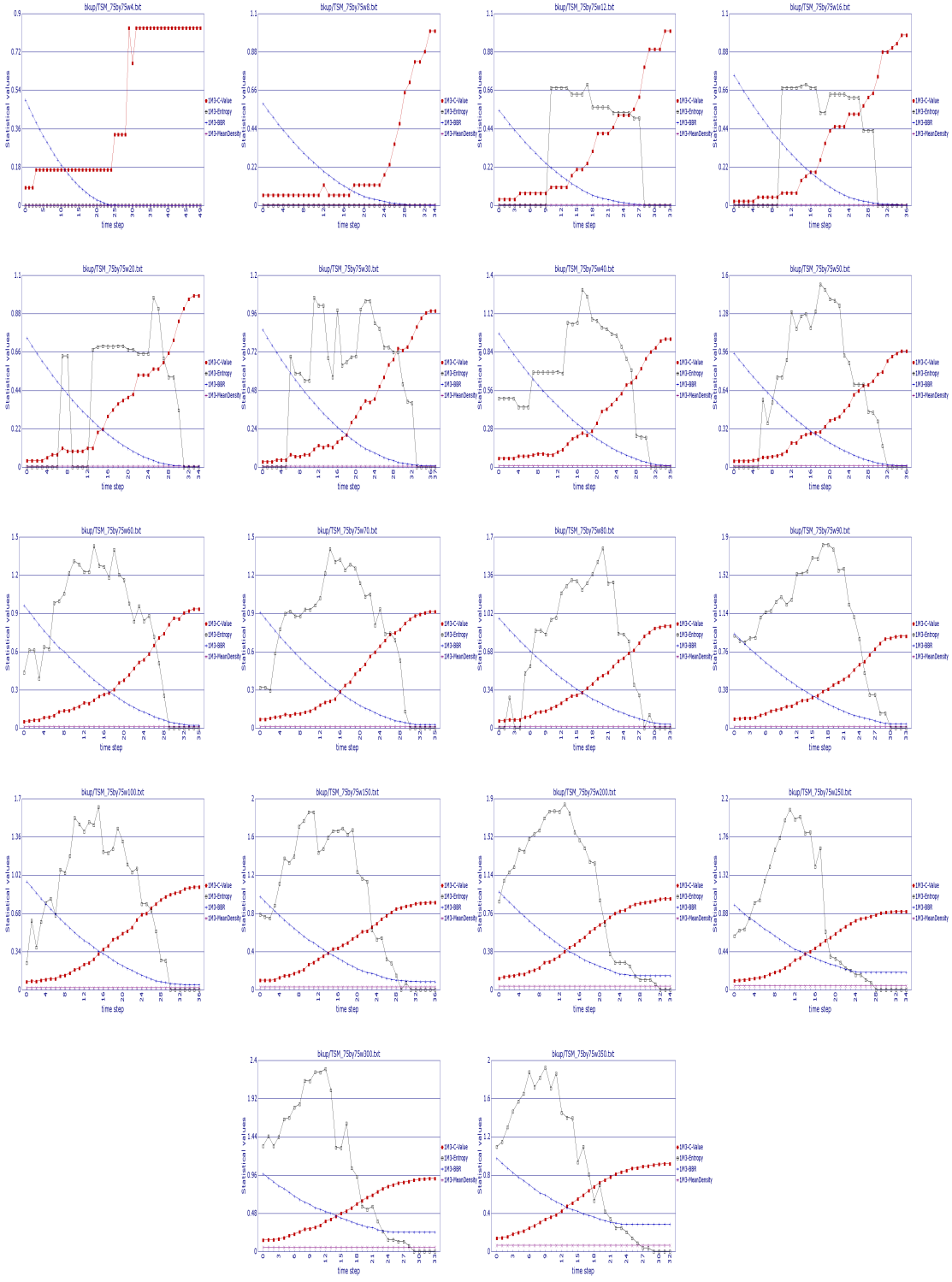


Figure B.3: Graphs showing the metric analysis of a 75 by 75 dynamic scenario run up to 50 time steps for agents of R={4, 8, 12, 16, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200, 250, 300, 350}

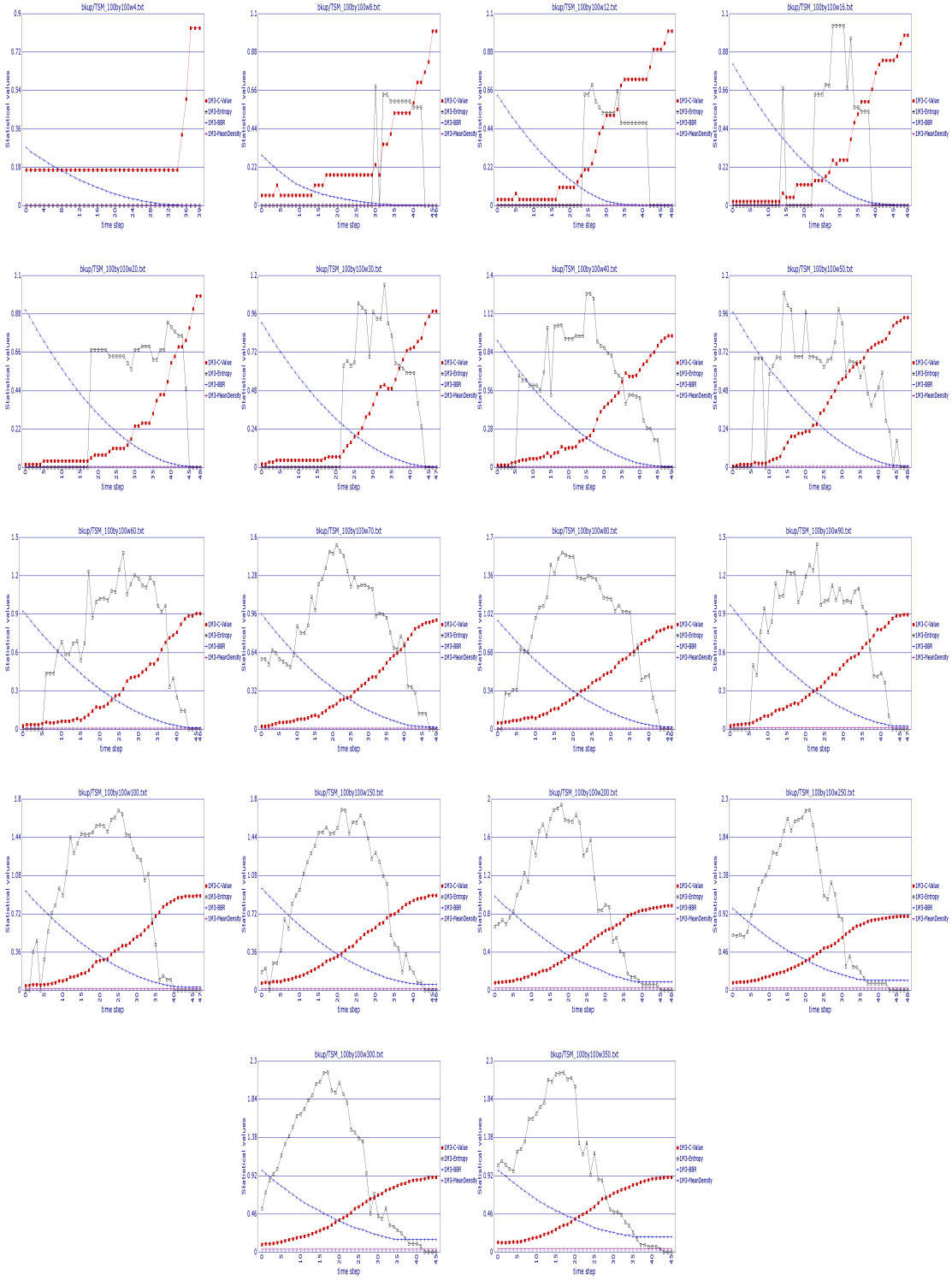


Figure B.4: Graphs showing the metric analysis of a 100 by 100 dynamic scenario run up to 50 time steps for agents of R={4, 8, 12, 16, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200, 250, 300, 350}

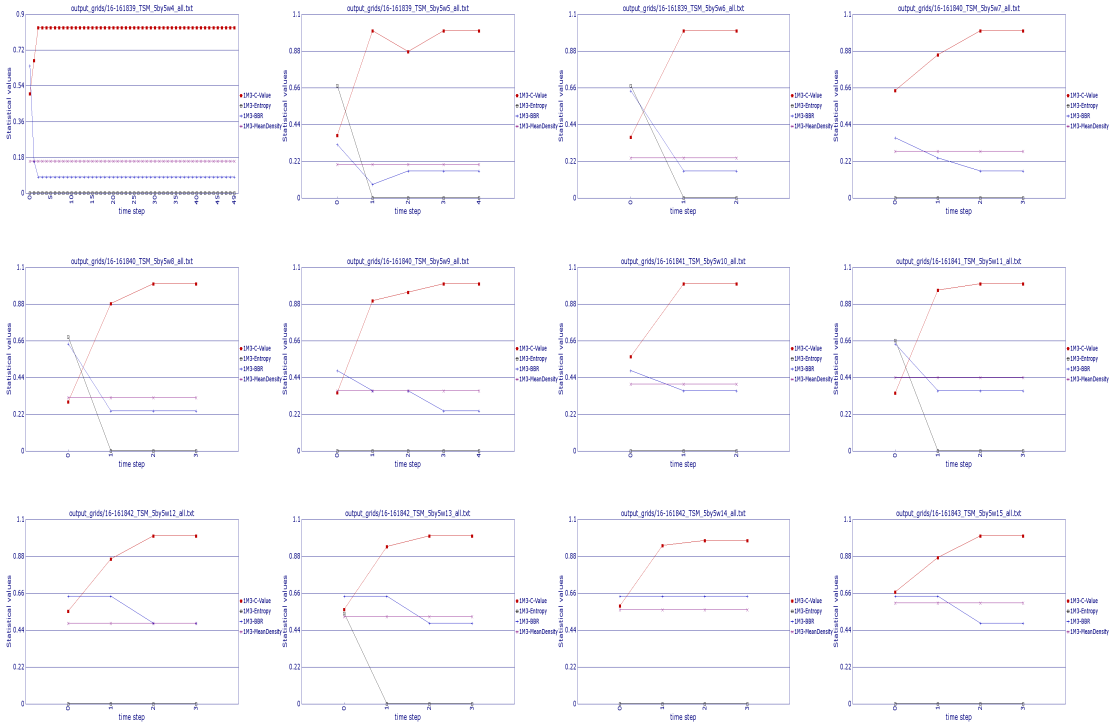


Figure B.5: Low agents - test 1 of 8 : Graphs showing the metric analysis of a 5 by 5 dynamic scenario run up to 50 time steps for agents of R={4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}

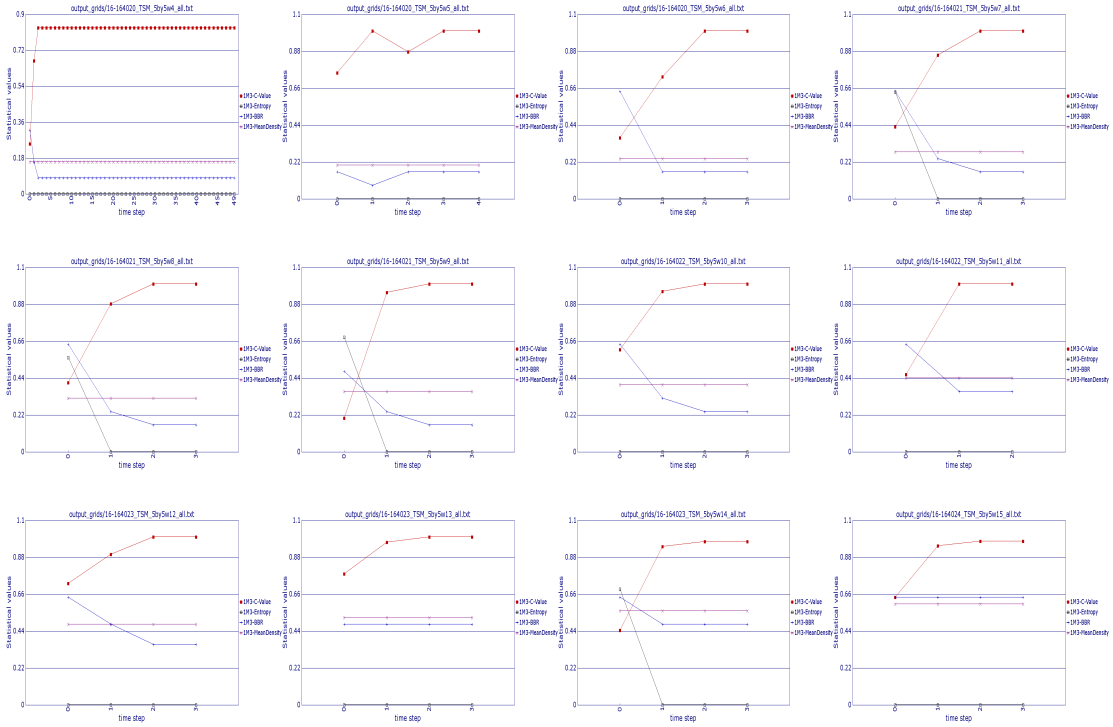


Figure B.6: Low agents - test 2 of 8: Graphs showing the metric analysis of a 5 by 5 dynamic scenario run up to 50 time steps for agents of R={4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}

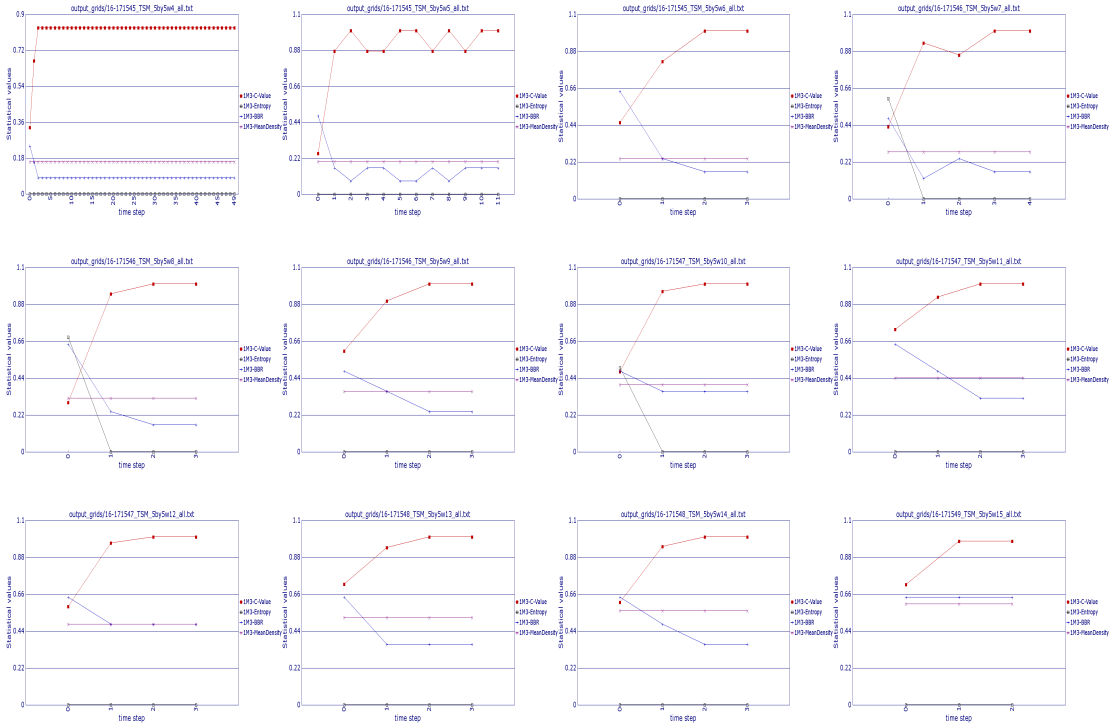


Figure B.7: Low agents - test 3 of 8: Graphs showing the metric analysis of a 5 by 5 dynamic scenario run up to 50 time steps for agents of R={4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}

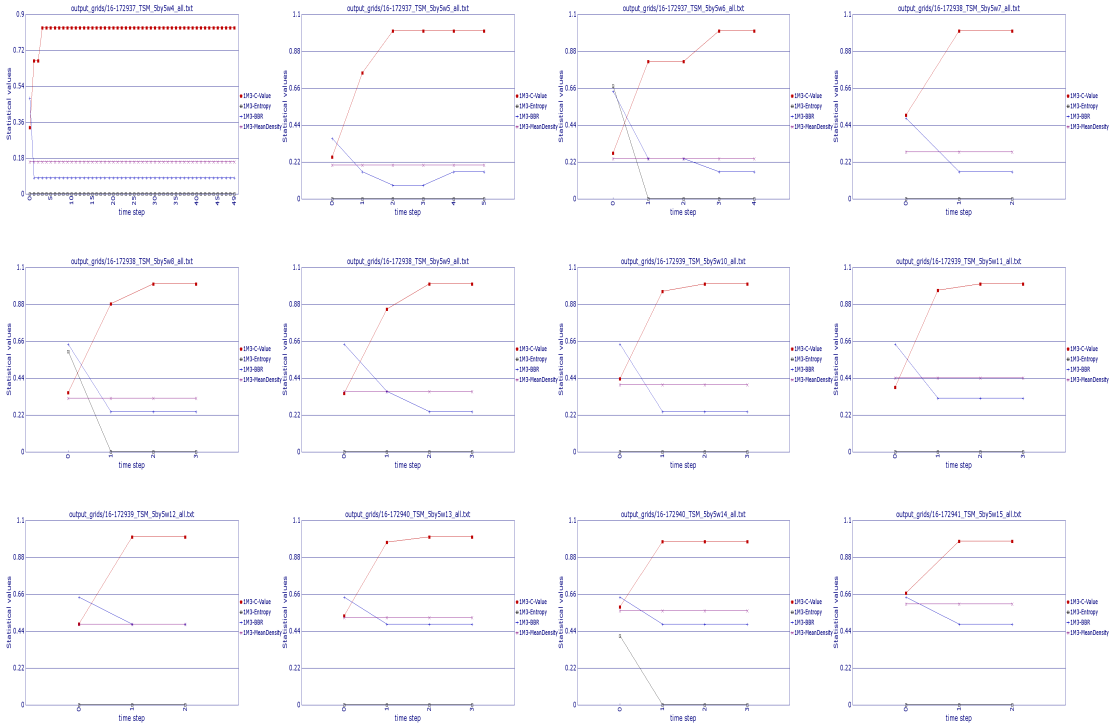


Figure B.8: Low agents - test 4 of 8: Graphs showing the metric analysis of a 5 by 5 dynamic scenario run up to 50 time steps for agents of R={4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}

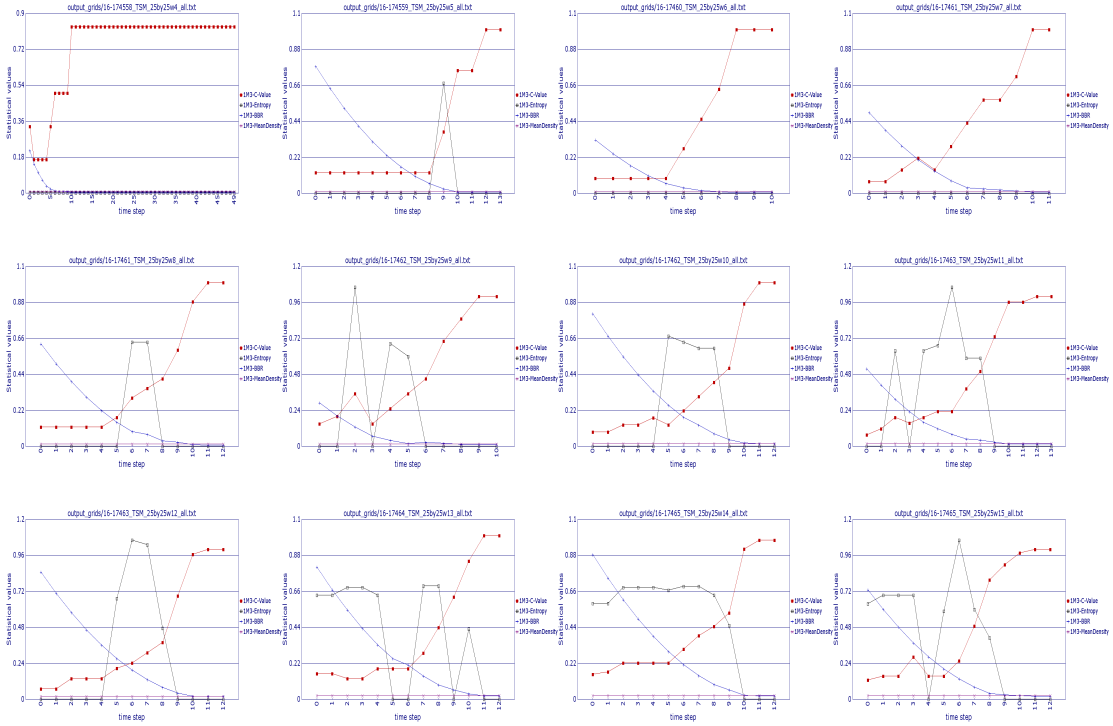


Figure B.9: Low agents - test 5 of 8: Graphs showing the metric analysis of a 25 by 25 dynamic scenario run up to 50 time steps for agents of R={4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}

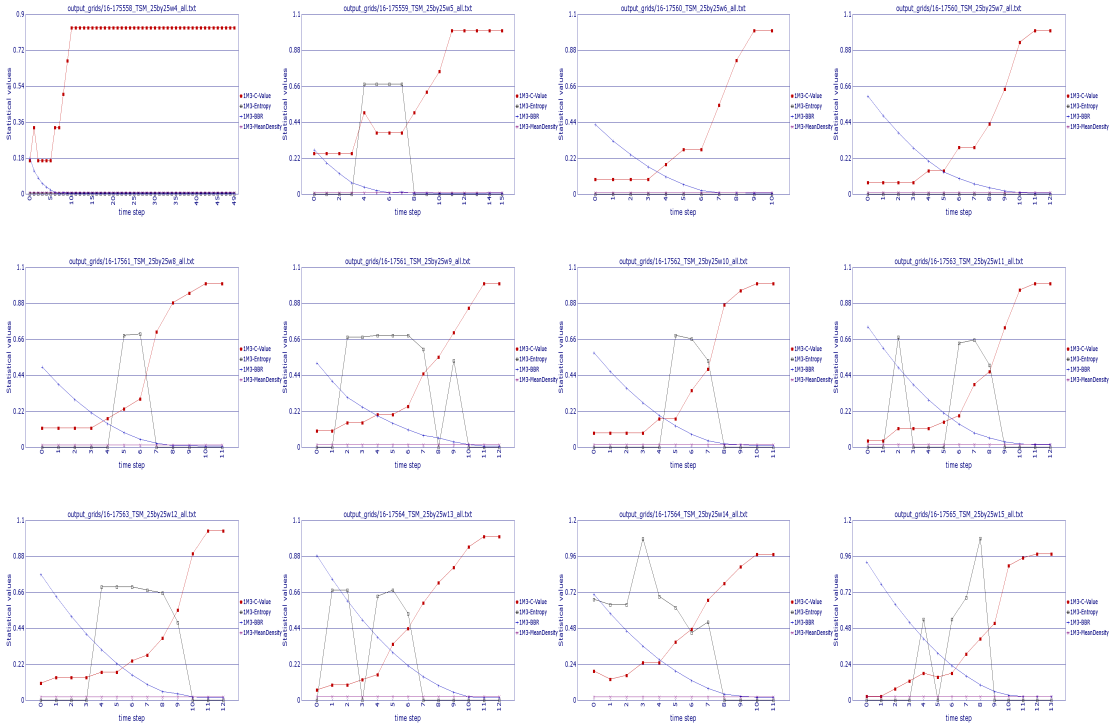


Figure B.10: Low agents - test 6 of 8: Graphs showing the metric analysis of a 25 by 25 dynamic scenario run up to 50 time steps for agents of R={4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}

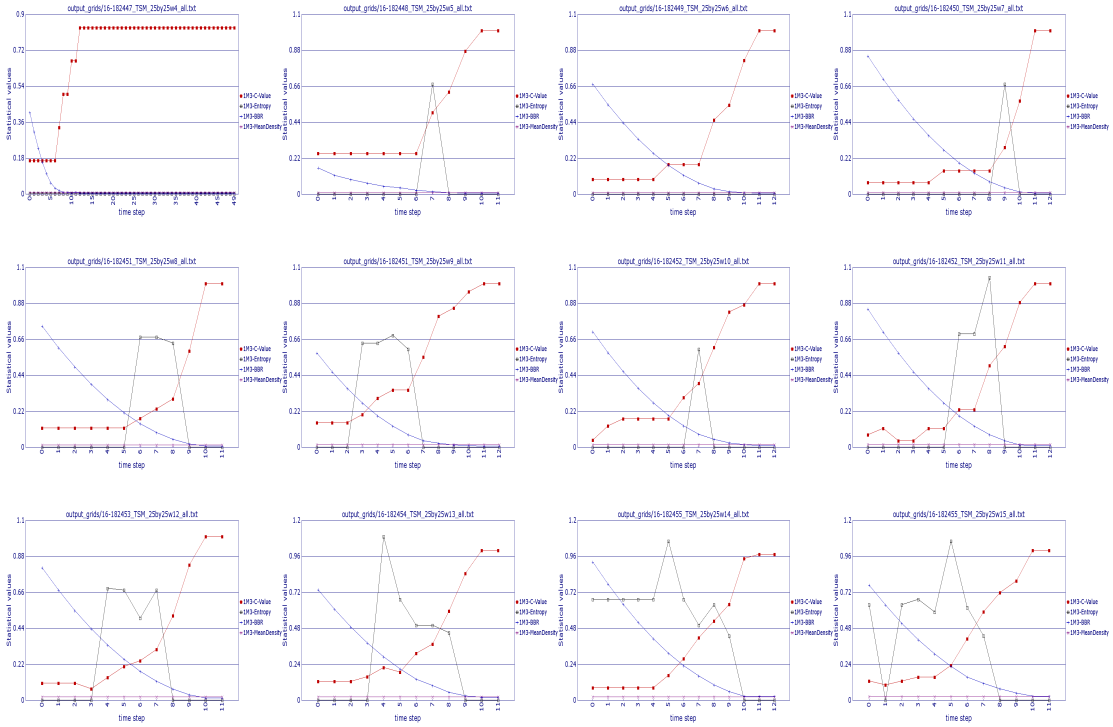


Figure B.11: Low agents - test 7 of 8: Graphs showing the metric analysis of a 25 by 25 dynamic scenario run up to 50 time steps for agents of R={4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}

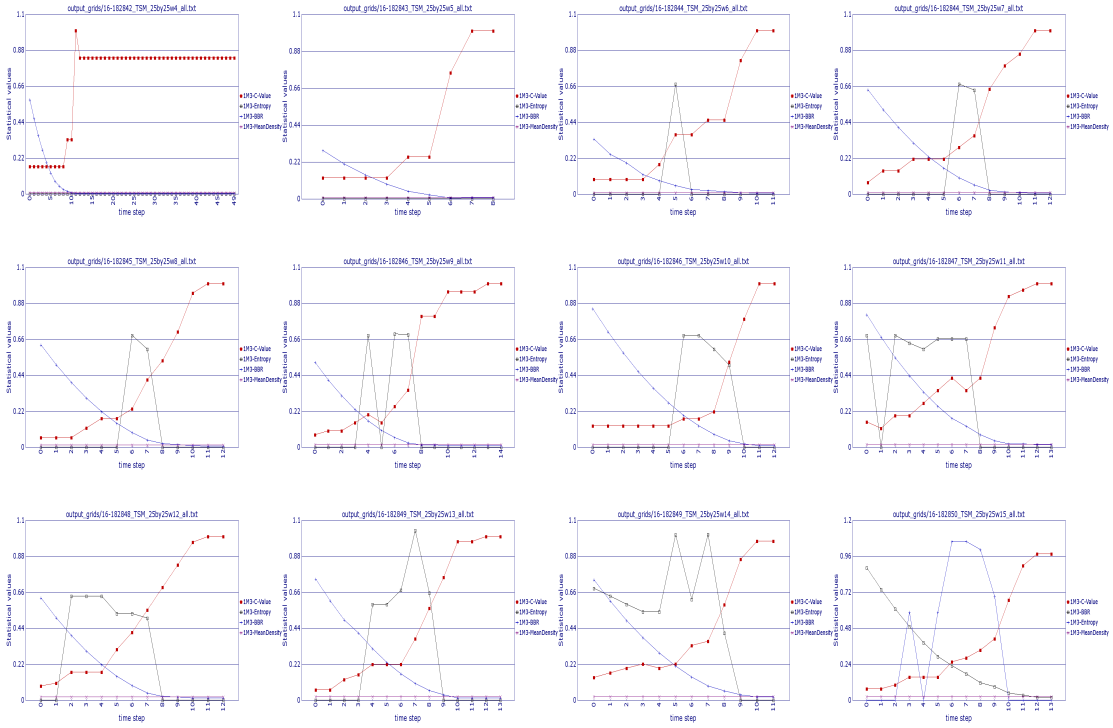


Figure B.12: Low agents - test 8 of 8: Graphs showing the metric analysis of a 25 by 25 dynamic scenario run up to 50 time steps for agents of R={4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}

B.1.4 Dynamic and probability scenario combination

This section contains the tables relevant to the tests shown in subsection 5.2.3. In general the C-Value of the initial grid starts aligned with the mean density, indicating that the random displacement of the active agents on the initial grid is indicated when the C-Value ≈ the mean density.

Table B.19: Dynamic scenario using an initial probability grid with *p_value*=0.1. The ID starts with the time step and the 1M3, signifying the value of an active cell (1), a Moore neighbourhood (M) and a range of 3. The C-Value of the initial grid starts just above the mean density value.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.1660	1.3653	0.9506	0.0931
1 1M3	0.1736	1.4533	0.8788	0.0931
2 1M3	0.2094	1.8114	0.8094	0.0931
3 1M3	0.2528	2.0248	0.7656	0.0931
4 1M3	0.3245	1.6831	0.7013	0.0931
5 1M3	0.3755	1.5609	0.6200	0.0931
6 1M3	0.4660	0.7631	0.5437	0.0931
7 1M3	0.5453	0.7506	0.4900	0.0931
8 1M3	0.6075	0.6661	0.4225	0.0931
9 1M3	0.6604	0.5949	0.3750	0.0931
10 1M3	0.7057	0.4680	0.3300	0.0931
11 1M3	0.7623	0.4073	0.2875	0.0931
12 1M3	0.7906	0.3381	0.2475	0.0931
13 1M3	0.8189	0.2693	0.2338	0.0931
14 1M3	0.8642	0.0000	0.2338	0.0931
15 1M3	0.9151	0.0000	0.2338	0.0931
16 1M3	0.9226	0.0000	0.2338	0.0931
17 1M3	0.9245	0.0000	0.2338	0.0931
18 1M3	0.9245	0.0000	0.2338	0.0931

Table B.20: Dynamic scenario using an initial probability grid with *p_value*=0.2. The ID starts with the time step and the 1M3, signifying the value of an active cell (1), a Moore neighbourhood (M) and a range of 3. The C-Value of the initial grid approximates with the mean density value.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.2188	2.0308	0.9506	0.1969
1 1M3	0.2347	2.3148	0.8788	0.1969
2 1M3	0.2889	2.4749	0.8100	0.1969
3 1M3	0.3603	1.8940	0.7650	0.1969
4 1M3	0.4282	1.2160	0.6800	0.1969
5 1M3	0.5159	1.0630	0.6587	0.1969
6 1M3	0.5993	0.9951	0.6188	0.1969
7 1M3	0.6724	0.7621	0.5800	0.1969
8 1M3	0.7360	0.6554	0.5800	0.1969
9 1M3	0.7859	0.4966	0.5800	0.1969
10 1M3	0.8220	0.3571	0.5800	0.1969
11 1M3	0.8719	0.2112	0.5800	0.1969
12 1M3	0.9089	0.0387	0.5800	0.1969
13 1M3	0.9338	0.0000	0.5800	0.1969
14 1M3	0.9536	0.0000	0.5800	0.1969
15 1M3	0.9544	0.0000	0.5800	0.1969
16 1M3	0.9544	0.0000	0.5800	0.1969

Table B.21: Dynamic scenario using an initial probability grid with *p_value*=0.3. The ID starts with the time step and the 1M3, signifying the value of an active cell (1), a Moore neighbourhood (M) and a range of 3. The C-Value of the initial grid approximates with the mean density value.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.3192	2.7373	0.9506	0.2988
1 1M3	0.3382	2.7220	0.9506	0.2988
2 1M3	0.3800	2.3553	0.8788	0.2988
3 1M3	0.4459	2.2061	0.8325	0.2988
4 1M3	0.5179	1.4151	0.7875	0.2988
5 1M3	0.6077	1.0900	0.7219	0.2988
6 1M3	0.6769	0.7695	0.7219	0.2988

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
7 1M3	0.7522	0.5483	0.7013	0.2988
8 1M3	0.8214	0.3800	0.7013	0.2988
9 1M3	0.8783	0.1984	0.7013	0.2988
10 1M3	0.9141	0.1060	0.7013	0.2988
11 1M3	0.9509	0.0000	0.7013	0.2988
12 1M3	0.9732	0.0000	0.7013	0.2988
13 1M3	0.9760	0.0000	0.7013	0.2988
14 1M3	0.9760	0.0000	0.7013	0.2988

Table B.22: Dynamic scenario using an initial probability grid with *p_value*==0.4. The ID starts with the time step and the 1M3, signifying the value of an active cell (1), a Moore neighbourhood (M) and a range of 3. The C-Value of the initial grid approximates with the mean density value.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.4120	1.2739	0.9506	0.4019
1 1M3	0.4445	0.7745	0.9506	0.4019
2 1M3	0.5095	0.7348	0.8788	0.4019
3 1M3	0.5991	0.7026	0.8325	0.4019
4 1M3	0.6756	0.6351	0.8094	0.4019
5 1M3	0.7405	0.5858	0.7438	0.4019
6 1M3	0.8055	0.3414	0.7438	0.4019
7 1M3	0.8573	0.2474	0.7438	0.4019
8 1M3	0.9071	0.2284	0.7438	0.4019
9 1M3	0.9428	0.0677	0.7438	0.4019
10 1M3	0.9712	0.0211	0.7438	0.4019
11 1M3	0.9856	0.0000	0.7438	0.4019
12 1M3	0.9889	0.0000	0.7438	0.4019
13 1M3	0.9901	0.0000	0.7438	0.4019
14 1M3	0.9901	0.0000	0.7438	0.4019

Table B.23: Dynamic scenario using an initial probability grid with *p_value*=0.5. The ID starts with the time step and the 1M3, signifying the value of an active cell (1), a Moore neighbourhood (M) and a range of 3. The C-Value of the initial grid starts just above the mean density value.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.5203	0.0902	0.9506	0.4894
1 1M3	0.5341	0.0752	0.9506	0.4894
2 1M3	0.5841	0.1356	0.9263	0.4894
3 1M3	0.6564	0.1286	0.8556	0.4894
4 1M3	0.7279	0.0810	0.8325	0.4894
5 1M3	0.7991	0.1796	0.8325	0.4894
6 1M3	0.8643	0.0431	0.8325	0.4894
7 1M3	0.9096	0.0322	0.8325	0.4894
8 1M3	0.9513	0.0499	0.8325	0.4894
9 1M3	0.9768	0.0000	0.8325	0.4894
10 1M3	0.9919	0.0000	0.8325	0.4894
11 1M3	0.9936	0.0000	0.8325	0.4894
12 1M3	0.9940	0.0000	0.8325	0.4894
13 1M3	0.9940	0.0000	0.8325	0.4894

Table B.24: Dynamic scenario using an initial probability grid with *p_value*=0.6. The ID starts with the time step and the 1M3, signifying the value of an active cell (1), a Moore neighbourhood (M) and a range of 3. The C-Value of the initial grid approximates with the mean density value.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.6177	0.0000	0.9506	0.6112
1 1M3	0.6434	0.0208	0.9506	0.6112
2 1M3	0.7030	0.0426	0.9263	0.6112
3 1M3	0.7651	0.0752	0.9263	0.6112
4 1M3	0.8256	0.0148	0.9263	0.6112
5 1M3	0.8796	0.0522	0.9263	0.6112
6 1M3	0.9181	0.0266	0.9263	0.6112
7 1M3	0.9551	0.0266	0.9263	0.6112
8 1M3	0.9778	0.0000	0.9263	0.6112
9 1M3	0.9920	0.0000	0.9263	0.6112

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
10 1M3	0.9957	0.0000	0.9263	0.6112
11 1M3	0.9965	0.0000	0.9263	0.6112
12 1M3	0.9965	0.0000	0.9263	0.6112

Table B.25: Dynamic scenario using an initial probability grid with *p_value*=0.7. The ID starts with the time step and the 1M3, signifying the value of an active cell (1), a Moore neighbourhood (M) and a range of 3. The C-Value of the initial grid approximates with the mean density value.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.7003	0.0000	0.9506	0.6900
1 1M3	0.7296	0.0000	0.9506	0.6900
2 1M3	0.7745	0.0338	0.9506	0.6900
3 1M3	0.8152	0.0000	0.9263	0.6900
4 1M3	0.8624	0.0188	0.9263	0.6900
5 1M3	0.9029	0.0000	0.9263	0.6900
6 1M3	0.9333	0.0188	0.9263	0.6900
7 1M3	0.9648	0.0000	0.9263	0.6900
8 1M3	0.9868	0.0000	0.9263	0.6900
9 1M3	0.9969	0.0000	0.9263	0.6900
10 1M3	0.9986	0.0000	0.9263	0.6900
11 1M3	0.9986	0.0000	0.9263	0.6900

Table B.26: Dynamic scenario using an initial probability grid with *p_value*=0.8. The ID starts with the time step and the 1M3, signifying the value of an active cell (1), a Moore neighbourhood (M) and a range of 3. The C-Value of the initial grid approximates with the mean density value.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.7989	0.0000	0.9506	0.7956
1 1M3	0.8261	0.0166	0.9506	0.7956
2 1M3	0.8602	0.0000	0.9506	0.7956
3 1M3	0.8945	0.0000	0.9506	0.7956
4 1M3	0.9248	0.0000	0.9506	0.7956

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
5 1M3	0.9526	0.0000	0.9506	0.7956
6 1M3	0.9785	0.0000	0.9506	0.7956
7 1M3	0.9898	0.0000	0.9506	0.7956
8 1M3	0.9971	0.0000	0.9506	0.7956
9 1M3	0.9982	0.0000	0.9506	0.7956
10 1M3	0.9986	0.0000	0.9506	0.7956
11 1M3	0.9986	0.0000	0.9506	0.7956
12 1M3	0.9988	0.0000	0.9506	0.7956
13 1M3	0.9988	0.0000	0.9506	0.7956

Table B.27: Dynamic scenario using an initial probability grid with p_value=0.9. The ID starts with the time step and the 1M3, signifying the value of an active cell (1), a Moore neighbourhood (M) and a range of 3. The C-Value of the initial grid approximates with the mean density value.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.9034	0.0000	0.9506	0.9038
1 1M3	0.9173	0.0000	0.9506	0.9038
2 1M3	0.9356	0.0000	0.9506	0.9038
3 1M3	0.9515	0.0000	0.9506	0.9038
4 1M3	0.9686	0.0000	0.9506	0.9038
5 1M3	0.9783	0.0000	0.9506	0.9038
6 1M3	0.9869	0.0000	0.9506	0.9038
7 1M3	0.9939	0.0000	0.9506	0.9038
8 1M3	0.9962	0.0000	0.9506	0.9038
9 1M3	0.9982	0.0000	0.9506	0.9038
10 1M3	0.9987	0.0000	0.9506	0.9038
11 1M3	0.9993	0.0000	0.9506	0.9038
12 1M3	0.9995	0.0000	0.9506	0.9038
13 1M3	0.9995	0.0000	0.9506	0.9038

B.2 Particle model

This section contains tables relevant to the the reactive-diffusion and chemotaxis model and the C-Value for low and high population tests.

Table B.28: Selected results from the simulation of amoebae and decentralisation gathering with barriers. A 40 by 40 grid was used with 600 amoebae placed randomly on the initial grid. An additional 222 cells were randomly selected as blocked. A probability setting of $(p_T, p_E, p_A) = (1, 0.01, 0)$. The run was for 2000 time steps, but gathering had been achieved by 650 time steps. The time steps 0 to 649 are shown below.

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.4053	2.1912	0.9506	0.3750
1 1M3	0.4002	2.3942	0.9506	0.3700
2 1M3	0.3931	2.4573	0.9506	0.3644
3 1M3	0.3946	2.7035	0.9506	0.3588
4 1M3	0.3865	2.8353	0.9506	0.3488
5 1M3	0.3632	2.8833	0.9506	0.3250
6 1M3	0.3438	2.8411	0.9506	0.3069
7 1M3	0.3295	2.9469	0.9506	0.2919
8 1M3	0.3375	2.9298	0.9506	0.2794
9 1M3	0.3312	2.8113	0.9506	0.2669
10 1M3	0.3318	2.8992	0.9506	0.2575
20 1M3	0.3974	2.6834	0.9506	0.2437
30 1M3	0.4235	2.5886	0.9506	0.2362
40 1M3	0.4617	2.4002	0.9506	0.2456
50 1M3	0.4810	2.4272	0.9019	0.2306
60 1M3	0.5248	2.3956	0.8775	0.2244
70 1M3	0.5434	1.9393	0.8775	0.2119
80 1M3	0.5567	2.2162	0.8775	0.2112
90 1M3	0.5748	2.1151	0.8287	0.2000
100 1M3	0.5918	1.9450	0.7800	0.1894
110 1M3	0.6184	1.5896	0.7800	0.1875
120 1M3	0.6229	1.7376	0.7600	0.1938
130 1M3	0.6264	1.5902	0.7600	0.1950
140 1M3	0.6549	1.7401	0.7362	0.1831
150 1M3	0.6857	1.4580	0.6975	0.1812
160 1M3	0.6683	1.0359	0.6750	0.1781

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
170 1M3	0.7017	1.3804	0.6562	0.1750
180 1M3	0.7017	1.1956	0.6525	0.1750
190 1M3	0.7286	1.3014	0.6344	0.1731
200 1M3	0.7243	1.2050	0.6344	0.1675
210 1M3	0.7505	1.2535	0.6344	0.1694
220 1M3	0.7701	0.9222	0.6562	0.1719
230 1M3	0.7691	1.1660	0.6562	0.1675
240 1M3	0.7759	0.8630	0.6344	0.1681
250 1M3	0.7940	1.1291	0.5906	0.1656
260 1M3	0.8044	0.7972	0.5850	0.1625
270 1M3	0.8282	0.6693	0.5625	0.1562
280 1M3	0.8293	0.9709	0.5469	0.1544
290 1M3	0.8427	0.6561	0.5469	0.1525
300 1M3	0.8462	0.7035	0.5250	0.1556
310 1M3	0.8620	0.7374	0.5250	0.1550
320 1M3	0.8504	0.7910	0.5250	0.1588
330 1M3	0.8384	0.8002	0.5469	0.1575
340 1M3	0.8614	0.7014	0.5250	0.1544
350 1M3	0.8643	0.6648	0.5250	0.1562
360 1M3	0.8576	0.8496	0.5031	0.1506
370 1M3	0.8486	0.8279	0.5031	0.1581
380 1M3	0.8444	0.8889	0.5031	0.1625
390 1M3	0.8403	0.6974	0.4813	0.1606
400 1M3	0.8349	0.6623	0.4813	0.1594
410 1M3	0.8264	0.9355	0.4813	0.1606
420 1M3	0.8232	0.6511	0.4594	0.1650
430 1M3	0.8227	0.6908	0.4594	0.1638
440 1M3	0.8433	0.1342	0.4594	0.1675
450 1M3	0.8363	0.7251	0.4156	0.1638
460 1M3	0.8519	0.0628	0.4156	0.1638
470 1M3	0.8422	0.0788	0.3937	0.1644
480 1M3	0.8467	0.7788	0.3937	0.1638
490 1M3	0.8730	0.6837	0.3719	0.1600
500 1M3	0.8650	0.1654	0.3613	0.1594
510 1M3	0.8741	0.0000	0.3613	0.1600
520 1M3	0.8800	0.1262	0.3613	0.1594
530 1M3	0.8598	0.0000	0.3300	0.1456

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
540 1M3	0.8836	0.0000	0.3000	0.1487
550 1M3	0.8878	0.0000	0.3200	0.1512
560 1M3	0.8792	0.0000	0.3300	0.1475
570 1M3	0.8871	0.0000	0.3300	0.1487
580 1M3	0.8980	0.0000	0.3300	0.1462
590 1M3	0.9051	0.0000	0.3094	0.1481
600 1M3	0.9048	0.0000	0.3094	0.1494
610 1M3	0.9106	0.0000	0.3094	0.1494
620 1M3	0.9101	0.0000	0.3094	0.1487
630 1M3	0.9221	0.0000	0.3094	0.1519
640 1M3	0.9255	0.0000	0.3094	0.1494
649 1M3	0.9210	0.0000	0.2888	0.1475

Table B.29: Selected values for reactive-diffusion chemataxis test using a 20 by 20 grid and 4 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.1667	0.0000	0.2925	0.0100
1 1M3	0.1667	0.0000	0.2925	0.0100
2 1M3	0.1667	0.0000	0.2925	0.0100
3 1M3	0.1667	0.0000	0.2925	0.0100
4 1M3	0.1667	0.0000	0.2925	0.0100
5 1M3	0.1667	0.0000	0.2925	0.0100
6 1M3	0.1667	0.0000	0.2925	0.0100
7 1M3	0.1667	0.0000	0.2925	0.0100
8 1M3	0.1667	0.0000	0.2925	0.0100
9 1M3	0.1667	0.0000	0.2925	0.0100
10 1M3	0.1667	0.0000	0.2925	0.0100
20 1M3	0.1667	0.0000	0.2400	0.0100
30 1M3	0.1667	0.0000	0.2400	0.0100
40 1M3	0.1667	0.0000	0.2400	0.0100
50 1M3	0.1667	0.0000	0.2400	0.0100
60 1M3	0.1667	0.0000	0.2400	0.0100
70 1M3	0.1667	0.0000	0.1925	0.0100
80 1M3	0.1667	0.0000	0.1925	0.0100
90 1M3	0.1667	0.0000	0.1925	0.0100

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
100 1M3	0.1667	0.0000	0.1925	0.0100
110 1M3	0.1667	0.0000	0.1925	0.0100
120 1M3	0.1667	0.0000	0.2200	0.0100
130 1M3	0.1667	0.0000	0.2000	0.0100
140 1M3	0.1667	0.0000	0.2000	0.0100
150 1M3	0.1667	0.0000	0.2000	0.0100
160 1M3	0.1667	0.0000	0.1575	0.0100
170 1M3	0.1667	0.0000	0.1575	0.0100
180 1M3	0.1667	0.0000	0.1400	0.0100
190 1M3	0.1667	0.0000	0.1225	0.0100
200 1M3	0.5000	0.0000	0.0900	0.0075
210 1M3	0.5000	0.0000	0.0500	0.0075
220 1M3	0.5000	0.0000	0.0300	0.0075
230 1M3	0.5000	0.0000	0.0300	0.0075
240 1M3	0.5000	0.0000	0.0150	0.0075
250 1M3	0.5000	0.0000	0.0150	0.0075
260 1M3	0.5000	0.0000	0.0150	0.0075
270 1M3	0.5000	0.0000	0.0150	0.0075
280 1M3	1.0000	0.0000	0.0050	0.0075
290 1M3	1.0000	0.0000	0.0000	0.0050

Table B.30: Selected statistics for a reactive-diffusion test using a 20 by 20 grid with 360 agents

ID	C-Value	Entropy	BBR	MeanDensity
0 1M3	0.9049	0.0000	0.9025	0.9000
1 1M3	0.8539	0.0000	0.9025	0.8325
2 1M3	0.7550	0.5677	0.9025	0.7125
3 1M3	0.7059	1.3622	0.9025	0.6275
4 1M3	0.6564	1.5161	0.9025	0.5600
5 1M3	0.6083	1.3196	0.9025	0.5125
6 1M3	0.5841	1.1564	0.9025	0.4775
7 1M3	0.5933	1.0027	0.9025	0.4750
8 1M3	0.5933	1.0027	0.9025	0.4750
9 1M3	0.5865	0.4712	0.9025	0.4725
10 1M3	0.6000	0.4684	0.9025	0.4775

continued on next page

ID	C-Value	Entropy	BBR	MeanDensity
20 1M3	0.6787	0.0000	0.9025	0.5150
30 1M3	0.6436	1.1530	0.9025	0.4525
40 1M3	0.6758	0.6377	0.9025	0.4800
50 1M3	0.6802	0.1110	0.9025	0.4300
60 1M3	0.7197	0.0628	0.9025	0.4375
70 1M3	0.7527	0.0000	0.9025	0.4525
80 1M3	0.7696	0.0000	0.8100	0.4275
90 1M3	0.8064	0.0000	0.8100	0.4150
100 1M3	0.7824	0.0000	0.7650	0.4200
110 1M3	0.7991	0.0000	0.8075	0.4400
120 1M3	0.8045	0.0662	0.7200	0.4075
130 1M3	0.8318	0.0000	0.7225	0.4425
140 1M3	0.8301	0.0000	0.7600	0.4275
150 1M3	0.8628	0.0000	0.6400	0.4075
160 1M3	0.9108	0.0000	0.6000	0.4150
170 1M3	0.9252	0.0000	0.5600	0.4025
180 1M3	0.9405	0.0000	0.5625	0.3775
190 1M3	0.9453	0.0000	0.5250	0.3600
200 1M3	0.9685	0.0000	0.4550	0.3575
210 1M3	0.9767	0.0000	0.4900	0.3625
220 1M3	0.9360	0.0000	0.4900	0.3325
230 1M3	0.9489	0.0000	0.4900	0.3450
240 1M3	0.9591	0.0000	0.5200	0.3450
249 1M3	0.9767	0.0000	0.4550	0.3350

B.3 Randomised model

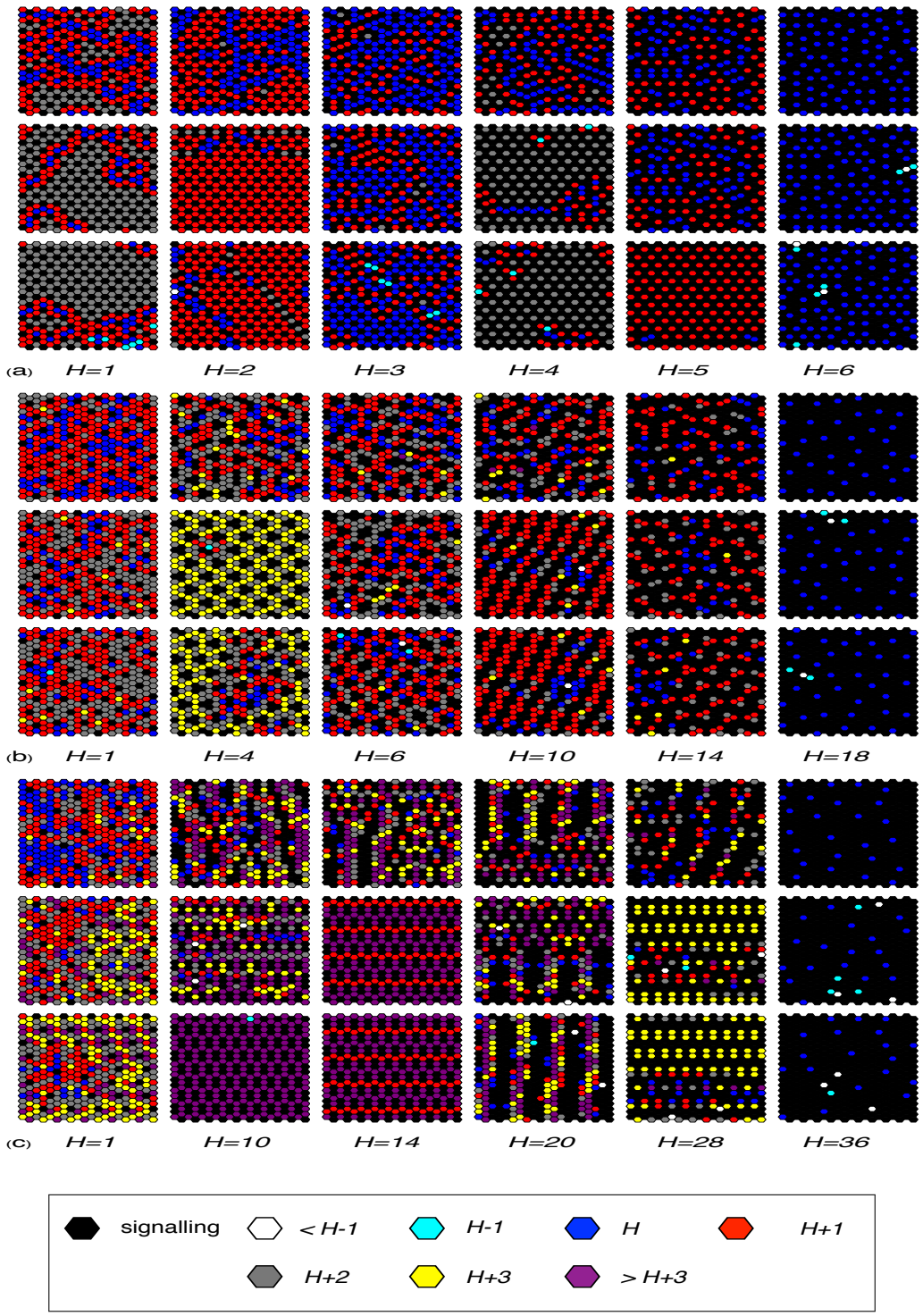


Figure B.13: Some more pattern formation using lateral inhibition with different thresholds, using a 20 by 20 grid over 1000 time steps (last time step shown). (a) range=1; (b) range =2; and (c) range=3. The top row in each grouping used a RSA update with no noise settings; the middle row used RSA with $N_t = 0.1, N_s = 0$; and the bottom row used ROA update with $N_t = 0.1, N_s = 0$.

B.3.1 Localised C-Value and randomised model

This section has the full tables showing the LC-Value and C-Value simulations using spatial noise settings of $p_S = 0.1$ and 0.9. The simulations are discussed in subsection 5.4.1.

Table B.31: Comparison of the C-Value and the Localised C-Value of simulation using a spatial noise setting of $p_S = 0.1$; range = 3, threshold = 4, a hexagonal grid was used with RSA updating with full toroidal wrap around boundary. See subsection 5.4.1 for details.

ID	C-Value	LC-Value
1 1HR3	0.2109	0.6826
2 1HR3	0.2007	0.7667
3 1HR3	0.2365	0.8000
4 1HR3	0.2359	0.7037
5 1HR3	0.2509	0.7500
6 1HR3	0.2591	0.7969
7 1HR3	0.2647	0.8206
8 1HR3	0.2612	0.7935
9 1HR3	0.2605	0.7683
10 1HR3	0.2549	0.8124
11 1HR3	0.2525	0.7814
12 1HR3	0.2558	0.7953
13 1HR3	0.2563	0.8171
14 1HR3	0.2614	0.8403
15 1HR3	0.2862	0.8636
16 1HR3	0.2648	0.8501
17 1HR3	0.2605	0.8326
18 1HR3	0.2605	0.8636
19 1HR3	0.2627	0.8746
20 1HR3	0.2617	0.7725
21 1HR3	0.2555	0.8207
22 1HR3	0.2412	0.8014
23 1HR3	0.2532	0.8008
24 1HR3	0.2617	0.8524
25 1HR3	0.2697	0.8387
26 1HR3	0.2773	0.8583
27 1HR3	0.2753	0.8409
28 1HR3	0.2804	0.8485

continued on next page

ID	C-Value	LC-Value
29 1HR3	0.2773	0.8583
30 1HR3	0.2785	0.8485
31 1HR3	0.2773	0.8409
32 1HR3	0.2788	0.8837
33 1HR3	0.2765	0.8485
34 1HR3	0.2669	0.8222
35 1HR3	0.2733	0.8409
36 1HR3	0.2741	0.8394
37 1HR3	0.2722	0.8939
38 1HR3	0.2753	0.9015
39 1HR3	0.2727	0.8939
40 1HR3	0.2710	0.9167
41 1HR3	0.2804	0.9147
42 1HR3	0.3006	0.9380
43 1HR3	0.3038	0.9457
44 1HR3	0.3023	0.9457
45 1HR3	0.3038	0.9457
46 1HR3	0.3038	0.9457
47 1HR3	0.3023	0.9457
48 1HR3	0.3023	0.9147
49 1HR3	0.2773	0.9070
50 1HR3	0.2975	0.9302
51 1HR3	0.2818	0.8915
52 1HR3	0.3006	0.9070
53 1HR3	0.3053	0.9147
54 1HR3	0.3023	0.9147
55 1HR3	0.3023	0.9147
56 1HR3	0.3084	0.9225
57 1HR3	0.3055	0.9225
58 1HR3	0.3023	0.9457
59 1HR3	0.3087	0.8992
60 1HR3	0.3115	0.8682
61 1HR3	0.3067	0.8485
62 1HR3	0.3084	0.8605
63 1HR3	0.3165	0.8889
64 1HR3	0.3165	0.8889
65 1HR3	0.3196	0.8968

continued on next page

ID	C-Value	LC-Value
66 1HR3	0.3178	0.9206
67 1HR3	0.3190	0.9206
68 1HR3	0.3165	0.9206
69 1HR3	0.3221	0.9286
70 1HR3	0.3263	0.9756
71 1HR3	0.3226	0.9238
72 1HR3	0.3244	0.9415
73 1HR3	0.3244	0.9415
74 1HR3	0.3282	0.9512
75 1HR3	0.3333	1.0000
76 1HR3	0.3323	1.0000
77 1HR3	0.3302	1.0000
78 1HR3	0.3323	1.0000
79 1HR3	0.3313	1.0000
80 1HR3	0.3323	1.0000
81 1HR3	0.3323	1.0000
82 1HR3	0.3333	1.0000
83 1HR3	0.3323	1.0000
84 1HR3	0.3323	1.0000
85 1HR3	0.3323	1.0000
86 1HR3	0.3313	1.0000
87 1HR3	0.3323	1.0000
88 1HR3	0.3333	1.0000
89 1HR3	0.3302	1.0000
90 1HR3	0.3259	0.9512
91 1HR3	0.3252	0.9431
92 1HR3	0.3313	1.0000
93 1HR3	0.3323	1.0000
94 1HR3	0.3333	1.0000
95 1HR3	0.3323	1.0000
96 1HR3	0.3333	1.0000
97 1HR3	0.3304	0.9917
98 1HR3	0.3323	1.0000
99 1HR3	0.3323	1.0000
100 1HR3	0.3333	1.0000
101 1HR3	0.3333	1.0000
102 1HR3	0.3333	1.0000

continued on next page

ID	C-Value	LC-Value
103 1HR3	0.3343	1.0000
104 1HR3	0.3293	0.9917
105 1HR3	0.3282	0.9512
106 1HR3	0.3282	0.9512
107 1HR3	0.3323	1.0000
108 1HR3	0.3333	1.0000
109 1HR3	0.3323	1.0000
110 1HR3	0.3323	1.0000
111 1HR3	0.3333	1.0000
112 1HR3	0.3323	1.0000
113 1HR3	0.3323	1.0000
114 1HR3	0.3323	1.0000
115 1HR3	0.3333	1.0000
116 1HR3	0.3323	1.0000
117 1HR3	0.3343	1.0000
118 1HR3	0.3314	0.9917
119 1HR3	0.3353	1.0000
120 1HR3	0.3353	1.0000
121 1HR3	0.3353	1.0000
122 1HR3	0.3353	1.0000
123 1HR3	0.3353	1.0000
124 1HR3	0.3362	1.0000
125 1HR3	0.3353	1.0000
126 1HR3	0.3353	1.0000
127 1HR3	0.3333	0.9917
128 1HR3	0.3324	0.9917
129 1HR3	0.3343	1.0000
130 1HR3	0.3353	1.0000
131 1HR3	0.3362	1.0000
132 1HR3	0.3343	0.9917
133 1HR3	0.3333	0.9917
134 1HR3	0.3353	1.0000
135 1HR3	0.3371	1.0000
136 1HR3	0.3371	1.0000
137 1HR3	0.3371	1.0000
138 1HR3	0.3371	1.0000
139 1HR3	0.3371	1.0000

continued on next page

ID	C-Value	LC-Value
140 1HR3	0.3371	1.0000
141 1HR3	0.3371	1.0000
142 1HR3	0.3371	1.0000
143 1HR3	0.3371	1.0000
144 1HR3	0.3371	1.0000
145 1HR3	0.3371	1.0000
146 1HR3	0.3371	1.0000
147 1HR3	0.3362	1.0000
148 1HR3	0.3371	1.0000
149 1HR3	0.3371	1.0000
150 1HR3	0.3371	1.0000
151 1HR3	0.3371	1.0000
152 1HR3	0.3371	1.0000
153 1HR3	0.3371	1.0000
154 1HR3	0.3371	1.0000
155 1HR3	0.3371	1.0000
156 1HR3	0.3371	1.0000
157 1HR3	0.3371	1.0000
158 1HR3	0.3371	1.0000
159 1HR3	0.3371	1.0000
160 1HR3	0.3371	1.0000
161 1HR3	0.3371	1.0000
162 1HR3	0.3362	1.0000
163 1HR3	0.3333	0.9512
164 1HR3	0.3324	0.9512
165 1HR3	0.3371	1.0000
166 1HR3	0.3371	1.0000
167 1HR3	0.3371	1.0000
168 1HR3	0.3371	1.0000
169 1HR3	0.3371	1.0000
170 1HR3	0.3371	1.0000
171 1HR3	0.3371	1.0000
172 1HR3	0.3371	1.0000
173 1HR3	0.3371	1.0000
174 1HR3	0.3371	1.0000
175 1HR3	0.3371	1.0000
176 1HR3	0.3371	1.0000

continued on next page

ID	C-Value	LC-Value
177 1HR3	0.3371	1.0000
178 1HR3	0.3371	1.0000
179 1HR3	0.3371	1.0000
180 1HR3	0.3371	1.0000
181 1HR3	0.3371	1.0000
182 1HR3	0.3371	1.0000
183 1HR3	0.3371	1.0000
184 1HR3	0.3371	1.0000
185 1HR3	0.3371	1.0000
186 1HR3	0.3371	1.0000
187 1HR3	0.3371	1.0000
188 1HR3	0.3371	1.0000
189 1HR3	0.3371	1.0000
190 1HR3	0.3371	1.0000
191 1HR3	0.3371	1.0000
192 1HR3	0.3371	1.0000
193 1HR3	0.3371	1.0000
194 1HR3	0.3371	1.0000
195 1HR3	0.3371	1.0000
196 1HR3	0.3371	1.0000
197 1HR3	0.3343	1.0000
198 1HR3	0.3353	1.0000
199 1HR3	0.3353	1.0000
200 1HR3	0.3353	1.0000

Table B.32: Comparison of the C-Value and the Localised C-Value of simulation using a spatial noise setting of $p_S = 0.9$; range = 3, threshold = 4, a hexagonal grid was used with RSA updating with full toroidal wrap around boundary. See subsection 5.4.1 for details.

ID	C-Value	LC-Value
1 1HR3	0.4668	0.4441
2 1HR3	0.5546	0.6168
3 1HR3	0.4992	0.2132
4 1HR3	0.5156	0.5771
5 1HR3	0.5238	0.7984

continued on next page

ID	C-Value	LC-Value
6 1HR3	0.5333	0.5657
7 1HR3	0.5466	0.8373
8 1HR3	0.5237	0.4294
9 1HR3	0.5079	0.6598
10 1HR3	0.5441	0.3424
11 1HR3	0.5184	0.5717
12 1HR3	0.5056	0.5725
13 1HR3	0.5250	0.8386
14 1HR3	0.5265	0.5814
15 1HR3	0.5035	0.6706
16 1HR3	0.4931	0.6572
17 1HR3	0.5201	0.8368
18 1HR3	0.5434	0.3447
19 1HR3	0.5324	0.6689
20 1HR3	0.5631	0.2367
21 1HR3	0.5077	0.6764
22 1HR3	0.5187	0.5754
23 1HR3	0.5461	0.7383
24 1HR3	0.5502	0.6384
25 1HR3	0.5377	0.4609
26 1HR3	0.5185	0.5907
27 1HR3	0.5201	0.6582
28 1HR3	0.4826	0.6525
29 1HR3	0.4835	0.2099
30 1HR3	0.4939	0.6652
31 1HR3	0.5526	0.3853
32 1HR3	0.4991	0.2624
33 1HR3	0.5017	0.5420
34 1HR3	0.5025	0.6214
35 1HR3	0.5220	0.5730
36 1HR3	0.4984	0.7697
37 1HR3	0.5056	0.1649
38 1HR3	0.4940	0.6197
39 1HR3	0.5205	0.3424
40 1HR3	0.5175	0.1705
41 1HR3	0.5498	0.4741
42 1HR3	0.5546	0.3469

continued on next page

ID	C-Value	LC-Value
43 1HR3	0.5095	0.1349
44 1HR3	0.5000	0.5372
45 1HR3	0.5110	0.4136
46 1HR3	0.4975	0.8149
47 1HR3	0.4764	0.5888
48 1HR3	0.4950	0.4037
49 1HR3	0.4649	0.5187
50 1HR3	0.5026	0.7943
51 1HR3	0.5534	0.7418
52 1HR3	0.5341	0.8761
53 1HR3	0.5137	0.6605
54 1HR3	0.5371	0.5616
55 1HR3	0.5017	0.2316
56 1HR3	0.5474	0.2337
57 1HR3	0.5116	0.5529
58 1HR3	0.4898	0.5399
59 1HR3	0.4830	0.6057
60 1HR3	0.4910	0.5273
61 1HR3	0.4908	0.4911
62 1HR3	0.5340	0.5558
63 1HR3	0.5087	0.3291
64 1HR3	0.4524	0.4706
65 1HR3	0.5048	0.3338
66 1HR3	0.5026	0.5550
67 1HR3	0.5048	0.7038
68 1HR3	0.5084	0.4916
69 1HR3	0.5323	0.3667
70 1HR3	0.4629	0.5562
71 1HR3	0.5206	0.6720
72 1HR3	0.4919	0.5289
73 1HR3	0.5253	0.4219
74 1HR3	0.5425	0.3174
75 1HR3	0.5425	0.8283
76 1HR3	0.4815	0.4951
77 1HR3	0.5465	0.8549
78 1HR3	0.5641	0.7248
79 1HR3	0.5208	0.4473

continued on next page

ID	C-Value	LC-Value
80 1HR3	0.5411	0.4205
81 1HR3	0.5034	0.5162
82 1HR3	0.5653	0.3640
83 1HR3	0.5123	0.5515
84 1HR3	0.5125	0.5851
85 1HR3	0.5139	0.4444
86 1HR3	0.5270	0.6522
87 1HR3	0.5116	0.3270
88 1HR3	0.5104	0.7070
89 1HR3	0.4955	0.5221
90 1HR3	0.5191	0.3227
91 1HR3	0.4925	0.3587
92 1HR3	0.5410	0.4933
93 1HR3	0.4983	0.6640
94 1HR3	0.5132	0.3336
95 1HR3	0.5075	0.6971
96 1HR3	0.5177	0.6399
97 1HR3	0.5212	0.5117
98 1HR3	0.5008	0.4090
99 1HR3	0.5513	0.6952
100 1HR3	0.4948	0.5661
101 1HR3	0.4674	0.5902
102 1HR3	0.5040	0.3254
103 1HR3	0.5481	0.5631
104 1HR3	0.5233	0.3293
105 1HR3	0.5065	0.4908
106 1HR3	0.5357	0.8333
107 1HR3	0.5350	0.4723
108 1HR3	0.5116	0.7148
109 1HR3	0.5076	0.8826
110 1HR3	0.4939	0.6078
111 1HR3	0.5444	0.4486
112 1HR3	0.5191	0.4546
113 1HR3	0.5467	0.2352
114 1HR3	0.4966	0.3385
115 1HR3	0.5119	0.6265
116 1HR3	0.4890	0.7328

continued on next page

ID	C-Value	LC-Value
117 1HR3	0.4795	0.6656
118 1HR3	0.5432	0.5947
119 1HR3	0.5245	0.5562
120 1HR3	0.5385	0.2607
121 1HR3	0.5123	0.5341
122 1HR3	0.5188	0.5876
123 1HR3	0.4991	0.5895
124 1HR3	0.5546	0.6757
125 1HR3	0.5414	0.5683
126 1HR3	0.5072	0.3274
127 1HR3	0.5111	0.4102
128 1HR3	0.5149	0.4752
129 1HR3	0.4949	0.3963
130 1HR3	0.5237	0.7367
131 1HR3	0.5123	0.2201
132 1HR3	0.4916	0.3111
133 1HR3	0.5195	0.6739
134 1HR3	0.4982	0.6364
135 1HR3	0.4992	0.4416
136 1HR3	0.5563	0.8631
137 1HR3	0.5493	0.6736
138 1HR3	0.5351	0.3347
139 1HR3	0.5382	0.4976
140 1HR3	0.5241	0.5569
141 1HR3	0.5334	0.3370
142 1HR3	0.4842	0.5733
143 1HR3	0.5247	0.2226
144 1HR3	0.4875	0.6597
145 1HR3	0.5127	0.4066
146 1HR3	0.5104	0.5428
147 1HR3	0.4949	0.7332
148 1HR3	0.5437	0.6729
149 1HR3	0.5060	0.5295
150 1HR3	0.5457	0.1735
151 1HR3	0.5561	0.8531
152 1HR3	0.5284	0.2249
153 1HR3	0.5098	0.6698

continued on next page

ID	C-Value	LC-Value
154 1HR3	0.4898	0.7756
155 1HR3	0.4731	0.3801
156 1HR3	0.5137	0.2225
157 1HR3	0.5453	0.3236
158 1HR3	0.5490	0.6976
159 1HR3	0.4992	0.5102
160 1HR3	0.5077	0.3553
161 1HR3	0.5009	0.3776
162 1HR3	0.5330	0.2429
163 1HR3	0.4520	0.5529
164 1HR3	0.5069	0.6162
165 1HR3	0.5274	0.3396
166 1HR3	0.5416	0.3388
167 1HR3	0.5116	0.6858
168 1HR3	0.5466	0.3467
169 1HR3	0.4983	0.3362
170 1HR3	0.4908	0.2478
171 1HR3	0.5153	0.4620
172 1HR3	0.4984	0.8011
173 1HR3	0.4781	0.2502
174 1HR3	0.5595	0.7085
175 1HR3	0.5481	0.3429
176 1HR3	0.4725	0.7355
177 1HR3	0.5076	0.7311
178 1HR3	0.5048	0.6557
179 1HR3	0.4724	0.5571
180 1HR3	0.4983	0.7152
181 1HR3	0.4935	0.3316
182 1HR3	0.4834	0.7208
183 1HR3	0.4883	0.6791
184 1HR3	0.5489	0.6753
185 1HR3	0.5016	0.5896
186 1HR3	0.5095	0.6113
187 1HR3	0.5334	0.3356
188 1HR3	0.5144	0.5551
189 1HR3	0.5075	0.7322
190 1HR3	0.5482	0.3014

continued on next page

ID	C-Value	LC-Value
191 1HR3	0.5166	0.1329
192 1HR3	0.5137	0.3494
193 1HR3	0.5324	0.3343
194 1HR3	0.4923	0.4998
195 1HR3	0.5410	0.2331
196 1HR3	0.5324	0.6706
197 1HR3	0.5291	0.3679
198 1HR3	0.5267	0.5625
199 1HR3	0.5450	0.6392
200 1HR3	0.5345	0.4480

Appendix C

Algorithms and code samples

This chapter contains some of the key algorithms and code segments used in the models. The full program suites are on the CD accompanying the thesis.

C.1 Calculating a toroidal neighbourhood

This appendix contains a selection of a few of the key routines created to run and test the models. The number of lines of codes prohibit the printing of anything more; but the full source code and the scripts are provided on the attached media.

This shows the searching of the toroidal boundary for the three ranges.

```
sub setup_h_grids_toroidal {
```

```
my $self = shift;
my ($state, $mdist) = @_;
my $loc = 0;
my $grid_size = $self->rows * $self->columns;

while ( $loc < $grid_size ) {
  # looking for a starting node - is the node unallocated, of the correct state and
  # with at least one unallocated neighbour within the specified manhattan distance
  if (!$self->get_allocated ($loc) && $state eq $self->get_state ($loc)
        && $self->unallocated_neighbour ($loc, $mdist, $state)) {
    # yes - we have the start of a connected grid
    my $grid = H_Grid->new( manhattan_distance => $mdist );
    my @to_check = ();
    my $next_node = $loc;
    # flag this startup one as connected (re first one)
    $self->nodes->[$loc]->set_connected;

    do {
      # unpack the location as we need to handle things as cartesian values:
      my ($row, $col) = (int $next_node / $self->columns,
                         $next_node % $self->columns);
      # if it is an odd value, then it is on the lower half row - so add one:
      my $lower = 0;
```

```
if ( $next_node % 2 == 1 ) {
  $lower = 1;
}

my $edges = 0;
if ( $mdist >= 1 )  {
  $edges += $self->check_node( $row - $mdist, $col, $state, \@to_check);
  $edges += $self->check_node( $row - ($mdist - $lower), $col + 1,
                               $state, \@to_check );
  $edges += $self->check_node( $row + ($mdist - 1 + $lower), $col + 1,
                               $state, \@to_check );
  $edges += $self->check_node( $row + $mdist, $col, $state, \@to_check );
  $edges += $self->check_node( $row + ($mdist - 1 + $lower), $col - 1,
                               $state, \@to_check );
  $edges += $self->check_node( $row - ($mdist - $lower), $col - 1,
                               $state, \@to_check );
}

if ( $mdist > 1 )  {
  $edges += $self->check_node( $row - 1, $col + $mdist,
                               $state, \@to_check);
  $edges += $self->check_node( $row, $col + $mdist, $state, \@to_check);
  $edges += $self->check_node( $row + 1, $col + $mdist,
```

```
                $state, \@to_check);
    $edges += $self->check_node( $row + 1, $col - $mdist,
                $state, \@to_check);
    $edges += $self->check_node( $row, $col - $mdist, $state, \@to_check);
    $edges += $self->check_node( $row - 1, $col - $mdist,
                $state, \@to_check);
}

if ( $mdist == 3 ) {
    $edges += $self->check_node( $row - 2, $col + 2, $state, \@to_check);
    $edges += $self->check_node( $row + 2, $col + 2, $state, \@to_check);
    $edges += $self->check_node( $row + 2, $col - 2 , $state, \@to_check);
    $edges += $self->check_node( $row - 2, $col - 2, $state, \@to_check);
    if ( $lower eq '1' ) {
        $edges += $self->check_node( $row + 2, $col - 3 ,
                $state, \@to_check);
        $edges += $self->check_node( $row + 2, $col + 3 ,
                $state, \@to_check);
    } else {
        $edges += $self->check_node( $row - 2, $col - 3,
                $state, \@to_check);
        $edges += $self->check_node( $row - 2, $col + 3,
                $state, \@to_check);
```

```
          }
        }

        # ok - set up the node in focus
        $grid->add_node($next_node, $edges);
      } while (defined ($next_node = shift(@to_check)));

      # set DNH_nodes in m_grid to allocated
      foreach ($grid->all_nodes) {
        $self->nodes->[$_->location]->set_allocated;
      }

      # add grid to connectedsubgrid array
      $self->_add_connectedsubgrid($grid);
    } # if potential starter node

    # increment the location:
    $loc++;
  }
  return;
}

sub check_node {
```

```
my $self = shift;
my ($x, $y, $state, $to_check_array) = @_;
my $edge = 0;

if (not defined ($x) or not defined ($y) or not defined ($state)
                    or not defined ($to_check_array)) {
  croak "You need to specify row, col, state and array for check_node";
}
# OK - get the location, adjusting for the toroidal boundary:
my $loc = $self->get_toroidal_location( $x, $y );

if ( $state eq $self->nodes->[$loc]->get_state
                && !$self->nodes->[$loc]->get_allocated) {
  # yes - check if already on the list, add if not
  if ($self->nodes->[$loc]->get_connected == 0) {
    # add to the to check list and mark as connected
    push( @$to_check_array, $loc );
    $self->nodes->[$loc]->set_connected;
  }
  # set edge to 1:
  $edge++;
}
return $edge;
```

```
}

sub get_toroidal_location {
  my $self = shift;
  my $i = shift;
  my $j = shift;

  # check if the values need adjusting for toroidal boundary:
  if ( $i < 0 ) {
    $i = $self->rows + $i;
  } elsif ( $i >= $self->rows ) {
    $i = $i - $self->rows;
  }

  if ( $j < 0 ) {
    $j = $self->columns + $j;
  } elsif ( $j >= $self->columns ) {
    $j = $j - $self->columns;
  }

  # convert to location:
  my $loc = ($i * $self->columns) + $j;
```

```
  return $loc;
}
```

C.2 Asynchronous updating

This section shows the code for asynchronous updating.

```
sub asynchronous_update_with_noise {
  my $self = shift;
  my $i = 0;
  my $j = 0;
  my $hash_list_ref;
  my ($nt_rate, $ns_rate) = (0, 0);

  $hash_list_ref = $self->get_async_update_list;

  # go through the list
  foreach my $key2 ( sort {$a <=> $b} keys %$hash_list_ref ) {
    my $loc = $hash_list_ref->{$key2};

    # unpack the location as we need to handle things as cartesian values:
    ($i, $j) = (int $loc / $self->columns, $loc % $self->columns);
```

```
# now get the various list we need of the state of the Moore neighbourhood
my $active_count = $self->get_neighbourhood_active_count( $i, $j, -1 );
if ( $active_count < $self->threshold ) {
  # check temporal noise:
  if ( rand(1) > $self->temporal_noise ) {
    # probability not met, so go ahead and activate, if not already active:
    if ( $self->timesteps->[-1]->get_state( $loc ) eq '0' ) {
      $self->timesteps->[-1]->activate_node( $loc );
      $self->timesteps->[-1]->tn_activated_add;
    }
  } else {
    # probability met, so ignore activation and de-activate if necessary:
    if ( $self->timesteps->[-1]->get_state( $loc ) eq '1' ) {
      $self->timesteps->[-1]->deactivate_node( $loc );
      $self->timesteps->[-1]->tn_deactivated_add;
    }
  }
} else {
  # check spatial noise - activate cell regardless if it meets the probability:
  if ( rand(1) <= $self->spatial_noise **
                  ( 1 + $active_count - $self->threshold ) ) {
    if ( $self->timesteps->[-1]->get_state( $loc ) eq '0' ) {
```

```
            $self->timesteps->[-1]->activate_node( $loc );
            $self->timesteps->[-1]->sn_activated_add;
        }
    } else {
        # no chance of redemption - inhibit the cell if it is active:
        if ( $self->timesteps->[-1]->get_state( $loc ) eq '1' ) {
            $self->timesteps->[-1]->deactivate_node( $loc );
            $self->timesteps->[-1]->sn_deactivated_add;
        }
    }
}
$self->set_up_nhood;
return;
}

sub get_async_update_list {
    my $self = shift;
    my $size = ($self->rows * $self->columns);
    my %hash_list;

    # is this a completely random / asynchronous list, with possible duplications:
    if ( $self->update_type eq 'r' or $self->update_type eq 'R'
```

```
            or $self->update_type eq 'a' or $self->update_type eq 'A' ) {
  # a randomly generated list of the same size as the grid will be used to update in
  # one time step - but there is no guarantee that all the cells will be updated -
  # theoretically one cell could be updated continuously.
  # Random Selection Asynchronous update:
  for ( my $index = 0; $index < $size; $index++ ) {
    $hash_list{$index} = int(rand($size));
  }
} else {
  # every cell will be updated in a synchronously from an asynchronous list.
  # Random Order Asynchronous update:
  my @array = ( 0 .. ($size-1) );
  for ( my $index = $size - 1; $index > 0; $index-- ) {
    my $loc = int(rand($index + 1));

    $hash_list{$index} = $array[$loc];
    splice @array, $loc, 1;
  }
  # should only be one left in the array - matches against 0;
  $hash_list{0} = $array[0];
}

return \%hash_list;
```

}

C.3 Calculating the ideal connectedness of a Moore neighbourhood

This shows the routine for calculating the ideal number of connections for a supplied number of active cells on a rectangular grid with null boundary conditions.

```
sub get_Moore_ideal_config {
  my $self = shift;
  my $nodes = shift;
  my $ideal_rows = 0;
  my $ideal_cols = 0;
  my $remainder = 0;
  my $ideal_edges =0;
  my $adjustment = 0;

  if ($nodes > 2) {
    # set up variables
    $ideal_rows = int (0.5 +sqrt($nodes)); # D8 rows
    $ideal_cols = int ($nodes / $ideal_rows); # E8 cols from quotient
    $remainder = int ($nodes % $ideal_rows); # F8 remainder from mod
```

```
my $starter_block = 0;
my $other_blocks = 0;
my $surplus_nodes = 0;

# 3 nodes do not have a starter block or other block
if ($nodes > 3) {
  $starter_block = 6 + (5*($ideal_rows-2));
  $other_blocks = ($ideal_cols -2) * (5 + (4 * ($ideal_rows-2)));
}

# is it the one node from a complete col (rows are longer)
if ($remainder eq $ideal_rows-1) {
  $surplus_nodes += 2;
  if ($remainder-1 > 0) {
    $surplus_nodes += (4 * ($remainder-1));
  }
  if( $ideal_rows > 3 ) {
    # OK - we can move two corners and gain an edge:
    if ( $ideal_rows < 6 ) {
      $surplus_nodes++;
    } else {
      # can move 3 corners:
      $surplus_nodes += 2;
```

```
    }
  }
} else {
  if ( $remainder > 0) {
    $surplus_nodes += 3 + (4 * ($remainder-1));
    # is it the one node from a complete col - NOTE: the way the ideal
    # rows and cols are calculated means that the rows will always be
    # equal to or more than the number of cols, so
    # adding a column gives the scope for more edges:
    if ( $remainder < $ideal_rows-2 ) {
      if ( ($ideal_rows - $remainder) eq '3' ) {
        # room to move one - so add one:
        $surplus_nodes++;
      } else {
        # room to move both:
        $surplus_nodes += 2;
      }
    } else {
      # OK - we cannot join the incomplete row, but we might still be
      # able to move the two opposite corners and gain another edge:
      if ( $ideal_rows > 3 ) {
        $surplus_nodes++;
      }
```

```
      }
    } elsif( $ideal_rows > 3 ) {
      # OK - we can move at least one corner and gain some more edges:
      if ( $ideal_rows < 6 ) {
        $surplus_nodes++;
      } elsif ( $ideal_rows == 6 ) {
        $surplus_nodes += 2;
      } else {
        $surplus_nodes += 3;
      }
    }
    # As the grid enlarges, more than the corners can become more
    # "rounded" and more edges created by stepping the edges and corners.
    # This adds edges. A rough measure is to add the quotient of the rows
    # divided by 3, but this only kicks in once the row is 5 or more, so reduce by.
    $adjustment = int (($ideal_rows -2) / 3);

    $ideal_edges = $starter_block + $other_blocks + $surplus_nodes + $adjustment;
# print "ir <$ideal_rows>, ic <$ideal_cols>, rem <$remainder>\n";
# print "ie <$ideal_edges>, sb <$starter_block>, ob <$other_blocks>, sn <$surplus_nodes>\n";
  } else {
    $ideal_rows = $nodes-1;
```

```
    $ideal_cols = $nodes-1;
    $ideal_edges = $nodes-1;
  }

  return ( $ideal_edges, $ideal_rows, $ideal_cols, $remainder );
}
```

This shows the routine for calculating the optimum number of connections for a supplied number of active cells on a hexagonal grid with toroidal boundary conditions

```
sub get_full_torridal_max_edges {
  my $self = shift;
  my $nodes = shift;
  my $rows = shift;
  my $cols = shift;
  my $edges = 0;

  # set up variables
  my $ideal_rows = int (0.5 +sqrt($nodes)); # D8 rows
  my $ideal_cols = int ($nodes / $ideal_rows); # E8 cols from quotient
  my $remainder = int ($nodes % $ideal_rows); # F8 remainder from mod
  my $num_of_edges = 6;
```

```
my $lowest = 0;
# set up lowest between rows and cols:
if ( $ideal_rows == $ideal_cols ) {
  $lowest = $ideal_rows;
} elsif ( $ideal_rows < $ideal_cols ) {
  $lowest = $ideal_rows;
} else {
  $lowest = $ideal_cols;
}

if ( $nodes == ($rows * $cols) ) {
  # we have a full grid with full wrap around:
  $edges = $nodes * $num_of_edges / 2;
} else {
  # first calculate for rectangle - treat as total wrap around;
  # every edge is connected, so total halved:
  $edges = $ideal_rows * $ideal_cols * $num_of_edges / 2;

  # now check if in fact it is not a perfect rectangle - if not do
  # specialised calculation:
  if ( $remainder > 0 ) {
    # OK - let's take away the wrap around of the lowest boundary:
    my $loss = 7 + ( 4 * int (($lowest - 2) / 2) ) + ( 1 * int ($lowest % 2) );
```

```
      my $gain = 4 + ( ($remainder-1) * 5 );

      $edges = $edges - $loss + $gain;
    }
  }

  return $edges;
}
```

C.4 Dynamic scenario gathering algorithm

This code from TSM_Kernel.pm shows the simple gathering algorithm used in the dynamic scenario.

```
sub get_new_location {
  my $self = shift;
  my $old = shift;

  # set up the new location as the old in case no movement is possible:
  my $new = $old;
  # calculate the cartesian coordinates:
  my ($x, $y) = (int $old / $self->columns, $old % $self->columns);
  # calculate the centre of the grid:
```

```
my $centre_x = int ( $self->rows / 2 );
my $centre_y = int ( $self->columns / 2 );

# OK: let's find our orientate.
# First were are we with relationship to the centre of the x-axis - to the left, on it,
# or to the right. Then within the relevant x-orientation, where are we in relationship
# to the centre of the y-axis - above, on, or below. The orientation then determines
# the order / location of the three cells checked; as soon as one is found empty, it is
# set as the new location, but if all are occupied the active agent does not move.

# is it left of the centre of the x-axis:
if ( $x < $centre_x ) {
  # Yes - find where it is re the centre of the y-axis & check potential new locations:
  if ( $y < $centre_y ) {
    # upper left
    if ( $self->timesteps->[-1]->get_node_state ( $x+1, $y+1 ) eq '0' ) {
      $new = (( $x+1 ) * $self->columns ) + $y + 1;
    } elsif ( $self->timesteps->[-1]->get_node_state ( $x, $y+1 ) eq '0' ) {
      $new = ( $x * $self->columns ) + $y + 1;
    } elsif ( $self->timesteps->[-1]->get_node_state ( $x+1, $y ) eq '0' ) {
      $new = (( $x+1 ) * $self->columns ) + $y;
    }
  } elsif ( $y == $centre_y ) {
```

```
# upper centre
if ( $self->timesteps->[-1]->get_node_state ( $x+1, $y ) eq '0' ) {
    $new = (( $x+1 ) * $self->columns ) + $y;
} elsif ( $self->timesteps->[-1]->get_node_state ( $x+1, $y+1 ) eq '0' ) {
    $new = ( ($x+1) * $self->columns ) + $y + 1;
} elsif ( $self->timesteps->[-1]->get_node_state ( $x+1, $y-1 ) eq '0' ) {
    $new = (( $x+1 ) * $self->columns ) + $y - 1;
  }
} elsif ( $y > $centre_y ) {
# upper right
  if ( $self->timesteps->[-1]->get_node_state ( $x+1, $y-1 ) eq '0' ) {
    $new = (( $x+1 ) * $self->columns) + $y-1;
  } elsif ( $self->timesteps->[-1]->get_node_state ( $x, $y-1 ) eq '0' ) {
    $new = ( $x * $self->columns ) + $y - 1;
  } elsif ( $self->timesteps->[-1]->get_node_state ( $x+1, $y ) eq '0' ) {
    $new = (( $x+1 ) * $self->columns ) + $y;
  }
}
# OK: is it on centre of the x-axis:
} elsif ( $x == $centre_x ) {
  # Yes - find where it is re the centre of the y-axis & check potential new locations:
  if ( $y < $centre_y ) {
    # centre left
```

```
    if ( $self->timesteps->[-1]->get_node_state ( $x, $y+1 ) eq '0' ) {
      $new = ( $x * $self->columns ) + $y + 1;
    } elsif ( $self->timesteps->[-1]->get_node_state ( $x-1, $y+1 ) eq '0' ) {
      $new = ( ($x-1) * $self->columns ) + $y + 1;
    } elsif ( $self->timesteps->[-1]->get_node_state ( $x+1, $y+1 ) eq '0' ) {
      $new = ((( $x+1 ) * $self->columns ) + $y + 1;
    }
  } elsif ( $y == $centre_y ) {
  # CENTRE - at target location - do nothing
  } elsif ( $y > $centre_y ) {
  # centre right
    if ( $self->timesteps->[-1]->get_node_state ( $x, $y-1 ) eq '0' ) {
      $new = ( $x * $self->columns ) + $y - 1;
    } elsif ( $self->timesteps->[-1]->get_node_state ( $x-1, $y-1 ) eq '0' ) {
      $new = ( ($x-1) * $self->columns ) + $y - 1;
    } elsif ( $self->timesteps->[-1]->get_node_state ( $x+1, $y-1 ) eq '0' ) {
      $new = ( ($x+1) * $self->columns ) + $y - 1;
    }
  }
# OK - it must be to the right of the centre of the x-axis - just confirm:
  } elsif ( $x > $centre_x ) {
  # Yes - find where it is re the centre of the y-axis & check potential new locations:
    if ( $y < $centre_y ) {
```

```
# lower left
if ( $self->timesteps->[-1]->get_node_state ( $x-1, $y+1 ) eq '0' ) {
    $new = ( ($x-1) * $self->columns ) + $y + 1;
} elsif ( $self->timesteps->[-1]->get_node_state ( $x-1, $y ) eq '0' ) {
    $new = ( ($x-1) * $self->columns ) + $y;
} elsif ( $self->timesteps->[-1]->get_node_state ( $x, $y+1 ) eq '0' ) {
    $new = ( $x * $self->columns ) + $y + 1;
}
} elsif ( $y == $centre_y ) {
# lower centre
if ( $self->timesteps->[-1]->get_node_state ( $x-1, $y ) eq '0' ) {
    $new = ( ($x-1) * $self->columns ) + $y;
} elsif ( $self->timesteps->[-1]->get_node_state ( $x-1, $y-1 ) eq '0' ) {
    $new = ( ($x-1) * $self->columns ) + $y - 1;
} elsif ( $self->timesteps->[-1]->get_node_state ( $x-1, $y+1 ) eq '0' ) {
    $new = ( ($x - 1) * $self->columns ) + $y + 1;
}
} elsif ( $y > $centre_y ) {
# lower right
if ( $self->timesteps->[-1]->get_node_state ( $x-1, $y-1 ) eq '0' ) {
    $new = ( ($x-1) * $self->columns ) + $y - 1;
} elsif ( $self->timesteps->[-1]->get_node_state ( $x-1, $y) eq '0' ) {
    $new = ( ($x-1) * $self->columns ) + $y;
```

```
    } elsif ( $self->timesteps->[-1]->get_node_state ( $x, $y-1 ) eq '0' ) {
        $new = ( $x * $self->columns ) + $y - 1;
    }
  }
 }
 # return the new location (which might be the old if no move was possible:
 return $new;
}
```

Appendix D

Outline of programs and utilities

This appendix outlines the main utilities and programs used in the thesis. It is is divided into three main sections. The first lists name and functions of the main shell scripts and Perl utilities written and used. The second deals with the Perl programs coded to create, run and analyse the scenarios and the particle and randomised automaton models. This section also outlines how the object orientated programs are constructed and illustrates how they work. The final section details the Java program built on the MUSCLE code base to provide the deterministic automaton model. The code and basic manuals can be found on the CD accompanying the thesis.

D.1 Shell and Perl utilities

This section lists examples of the main shell and Perl scripts written to facilitate the test, simulations and analysis carried out as part of this thesis. The shell scripts end with *.sh* and the Perl scripts with *.pl*. This list does not include the *.pl* files that are used to log and manage the calls to the Perl programs that create and simulate the dynamic scenario and the particle and randomised automaton models; these are outlined in section D.2

- **create_prob-array.pl** - this creates initial probability arrays. A *p_value* between 0.1 and 0.9, row and column size, and number of grids to create are passed in from the command line invocation. The grid is output in text format to the terminal and can be redirected into a file. This code also provided the basis for the internal creation of probability grids in the probability scenario program, (*TSM_Kernel*). A cell is set as active by the testing of a pseudo random number generated to fifteen decimal places between, but not including, 0 and 1 against the supplied *p_value*, if it is less or equal to the *p_value* the cell is flagged as active.

- **DNH_ALL_and_last_as_pdf.pl** - this Perl script invokes the delta notch signalling simulation. The following parameters are passed in: the row and column size of the grid, the threshold value, the range, the temporal noise probability value, the spatial noise value, the number of time steps, and the update type (ROA or RSA). The grids produced by the simulation are saved in a number of ways, including text and *xml*, (the last *xml* grid is extracted and saved as a *pdf* file). The statistics are saved for use in the analysis program.

- **DNH_get_xml_records.pl** - this Perl script takes in a file created from a previous run of the delta-notch signalling model that holds the grids created at each time step in text format. The script outputs a *xml* rendition of the grids.

- **dnh_noise.sh** - example of a script used to automate the running of multiple instances of one of the other programs or scripts. In this case the Perl script *DNH_ALL_and_last_as_pdf.pl* is called through a series of three loops and a number of times in each loop. The called script runs the randomised delta notch signalling model, (*DNH_Kernel*). The three loops cover the three range settings of 1, 2 and 3; the calls to *DNH_ALL_and_last_as_pdf.pl* are passed different noise, asynchronous updating schemes and threshold settings for the invocation of the randomised automaton model. The called program runs the simulation and also extracts a *xml* rendition of the last grid as a *pdf* files, (as used in Figure 4.31.

- **extract_grids.sh** - this script extracts a grid or sequence of grids into a separate file. The parameters are the input file that is having the grid(s) extracted, the first grid to be extracted and the number of grids to be extracted - so *fname 24 1*, extracts just the 24th grid, whereas *fname 24 4*, extracts the 24th, 25th, 26th and 27th grids. This allows further processing, such as rendering the grids into a different format, without having to reprocess the whole simulation again.

- **extract_record.sh** - this script takes in a *html* file and extracts a specified 'grid' display into a new *html* file, which is then rendered into a *pdf* file via the script *pdffile.sh*. The output can then be used in the LaTeX files used to write up this thesis.

- **g_csv.pl** - This program takes in a csv file from a spreadsheet and converts it into the grid text format used in the program suite. This is used to convert grids and patterns created in a spreadsheet for use with the hand crafted scenarios in subsection 4.4.1.

- **MemoryUse.pl** - this script extracts the memory use of each ANode on an AN grid, and for each time step.
- **p_value_text.sh** - this script creates a range of probability grids from 0.1, 0.125, 0.150, 0.175, 0.2, 0.225, ... 0.875, 0.9. These are used in the testing of the probability scenario.
- **pdffile.sh** - this script takes the input in a html file created from a simulation, amends the height and width so that it will display well within an A4 page and then renders it into a pdf file:
- **process_grids.sh** - this is an example of a script that extracts blocks of records from the output of a simulation. In this case the script is set up to extract and process blocks of records from the delta notch signalling simulation output. The parameters passed into the script are the number of records to be extracted in each block and the file from which they are to be extracted from. Each extraction is placed in a separate file and reprocessed by the *DNH_Kernel* suite.
- **rename.pl** - this replicates the 'rename' file utility.
- **s-data-it.sh** - this script takes in a stats file and then produces separate and combined graphs (*.png*) and LaTeX tables (*.tex*) for the four metrics.
- **tsm_combined2b.sh** - example of a script running a series of tests and then combining selected elements, (usually the last time step grid), to produce statistics and graphs. This script processes the output grids resulting from the dynamic scenario run with a 20 by 20 grid and for 300 to 390 agents. Each output is run with the *C_Kernel* program to produce statistics that are then combined into one file; this file is then processed by the *S_Date* program to produce a graph.

D.2 Perl programs

There are five Perl programs in the software suite created for this thesis:

dynamic scenarios	TSM_Kernel,
randomised model	RDC_Kernel,
statistics program	C_Kernel,
particle model	DNH_Kernel, and
analysis program	S_Data

The programs use the Perl Moose object orientated environment. Each program is made up of a series of *.pm* files, with the entry point through a *.pm* file with the program name, such as *TSM_Kernel.pm*. The invocation of each program is handled through a *.pl* Perl script with the equivalent name, i.e., *TSM_Kernel.pl*. The five programs and their scripts are outlined in the subsections below.

D.2.1 TSM_Kernel

This program represents a view of the basic structure used in the Perl programs. Its purpose is to create and run the dynamic scenario, as outlined and analysed in subsection 4.4.3 & subsection 5.2.2. The program is invoked through *TSM_Kernel.pm*, which has an array of grids, each representing a time step. The grids are coded in *TSMGrid.pm*, which has an array of nodes matching the number of cells on the grid. The nodes are coded in *TMS_Node.pm* A schematic of the program is shown in Figure D.1, and the options for invoking it through *TMS_Kernel.pl* in Figure D.2.

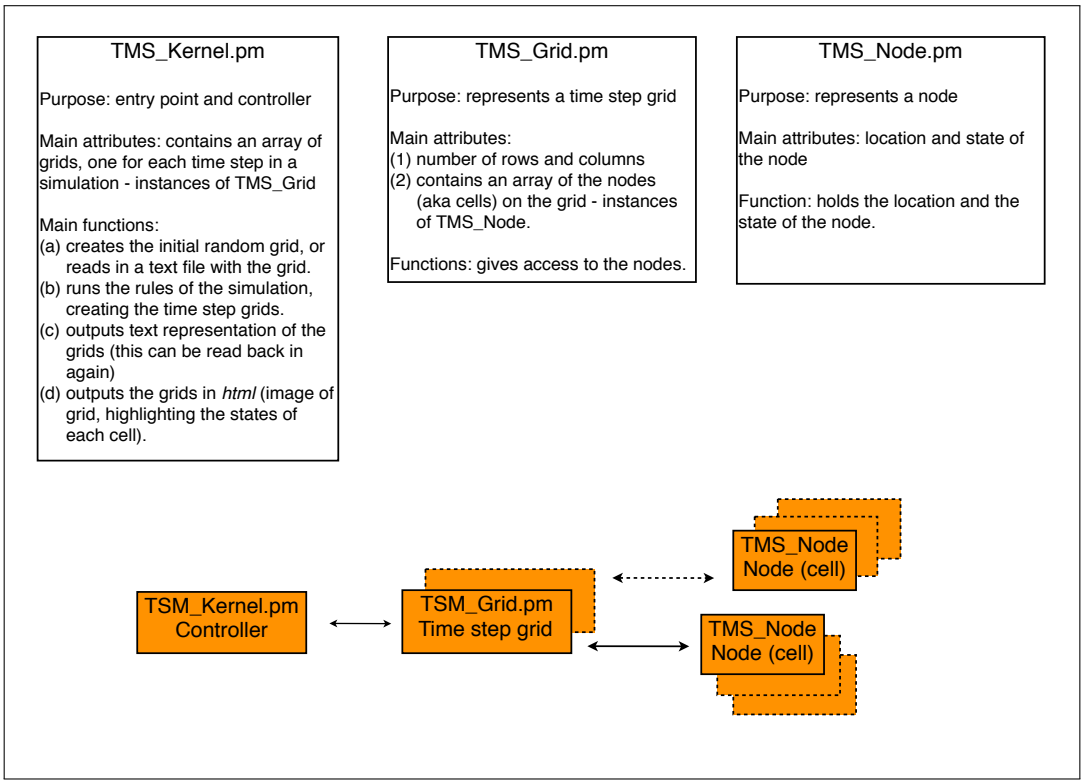


Figure D.1: Schematic of *TMS_Kernel*.

TMS_Kernel.pl (invokes the dynamic scenario)

RUN *perl TMS_Kernel.pl run-type*
*[*if 'r' then -> *rows cols a-count loop-count output-style]*
OR *[*if 'f' then -> *input_file loop-count output-style]*

run-type:		-	*r* - create the required random grid
			f - read grid in from file
input_file:		-	input file with starter grid in it
rows:		-	number rows
cols:		-	number columns
a-count:		-	number of agents
loop-count		-	number of iterations (aka time steps)
output-style	*sg*	-	save the grids as text --> output to txt file in *output_grids/*
	sh	-	save the grids as *html* --> output to html file in *displays/*
	st	-	save the state as text --> output to txt file in *output_texts/*
	all	-	perform the equivalent of *sg* and *sh*

Process:
(1) read in the parameters based on whether the first one is 'r' or 'f'.
(2) set up the log file.
(3) process the *output-style*.
(4) output the relevant file(s)

Files created:
Log files: everything is logged to a log file related to the run of the program

logs/ + month day + - + hour + minute + second + _RDC_ + rows + by + cols + w + a-count + pt + pT + e + pE + a + pA + m + m-range + ts + ts-count + _ + update-type + _log.txt

e.g. logs/19-141344_DNH_1H3_20by20t1r1tn0.1sn0utR_log.txt

The output files are also made up and put in relevant directories:
directory/ + month day + - + hour + minute + second + _CK_ + inputfile pre . name + active + nhood + range + *_option.file-type*

Figure D.2: *TMS_Kernel-pl* - used to invoke the *TMS_Kernel* program.

D.2.2 RDC_Kernel

This program creates and runs the reactive-diffusion chemotaxis amoebae model used as the particle automaton model. The structure is the same as *TSM_Kernel*, but with *RDC_Kernel.pm, RDC_Grid.pm* and *RDC_Node.pm* A schematic of the *RDC_Kernel* program is shown in Figure D.3, and the options for invoking it through *TMS_Kernel.pl* in Figure D.4.

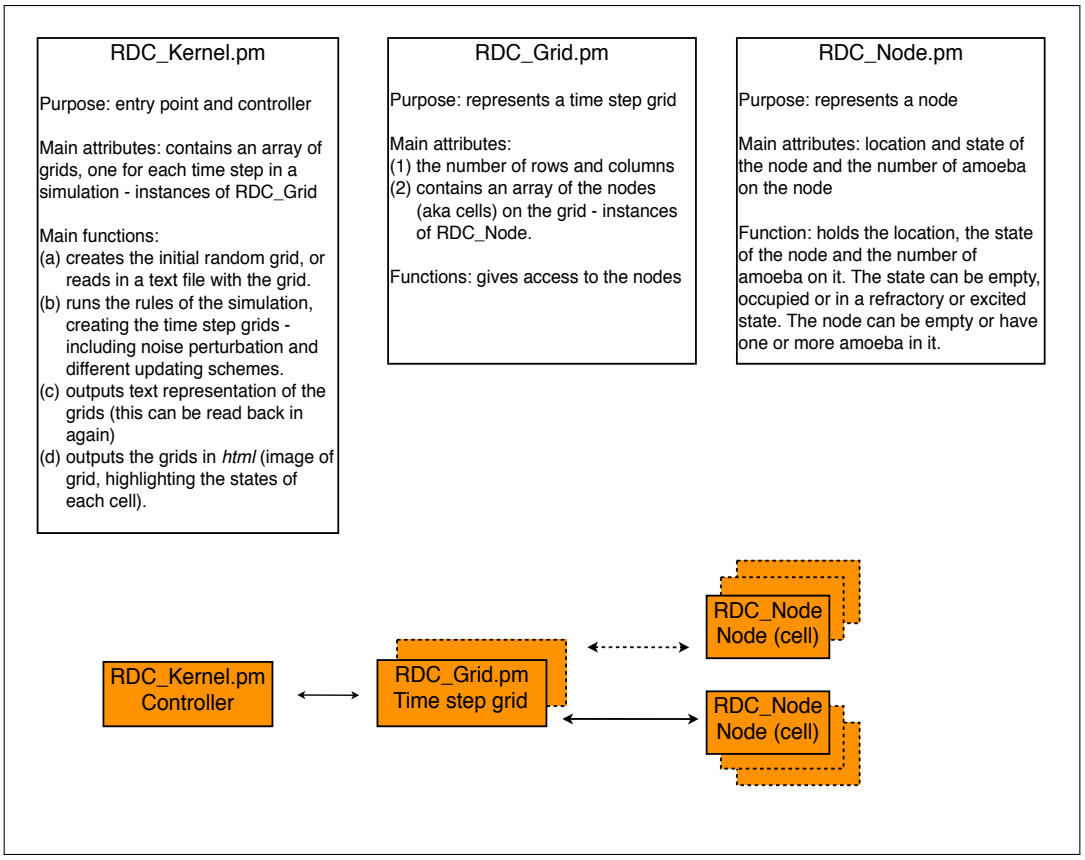


Figure D.3: Schematic of *RDC_Kernel*.

RDC_Kernel.pl (invokes the particle automaton model of amoebae and reaction-diffusion and chemotaxis)

RUN: *perl RDC_Kernel.pl run-type*
*[*if 'r' then -> *rows cols a-count pT pE pA m-range ts-count update-type ts-range]*
OR *[*if 'f' then -> *input_file pT pE pA m-range ts-count update-type ts-range]*

run-type:	-	*r* - create the required grid *f* - read grid in from file
input_file:	-	input file with starter grid in it
rows:	-	number rows
cols:	-	number columns
a-count:	-	amoeba count
pT:	-	0 =< transmission probability setting =< 1
pE:	-	0 =< environment probability setting =< 1
pA:	-	0 =< agitation probability setting =< 1
m-range:	-	1 =< Manhattan range =< 3
ts-count:	-	number of time steps
update-type:	-	*a/A,* or *s/A* [synchronous or asynchronous - defaults to *S* if not provided
ts-range:	-	time steps to be processed; if none specified then all time steps are processed

Process: (A) if *r* then (1) create a random grid with *rows cols a-count*, (2) run the simulation with *pT pE pA m-range ts-count update-type*, (3) save amoeba grid for analysis in *output_grids/* with *_ag.txt*, (4) save all the grids or the ones specified in *ts-range* as html in *displays/* with *_eh.html* and (5) save the last grid as text in *output_grids/* with *_eg.txt*.

(B) if *f* then (1) use input file name to set up log, (2) read in file and set up initial grid. (3) run the simulation with *pT pE pA m-range ts-count update-type*, (4) save amoeba grid for analysis in *output_grids/* with *_ag.txt*, (5) save all the grids or the ones specified in *ts-range* as html in *displays/* with *_eh.html* and (6) save the last grid as text in *output_grids/* with *_eg.txt*.

Files created:
Log files: everything is logged to a log file related to the run of the program
logs/ + month day + - + hour + minute + second + _RDC_ + rows + by + cols + w + a-count + pt + pT + e + pE + a + pA + m + m-range + ts + ts-count + _ + update-type + _log.txt
e.g. logs/19-141344_DNH_1H3_20by20t1r1tn0.1sn0utR_log.txt

The output files are also made up and put in relevant directories:
directory/ + month day + - + hour + minute + second + _CK_ + input-file pre . name + active + nhood + range + *_option.file-type*

Figure D.4: *RDC_Kernel.pl* - used to invoke the RDC_Kernel program.

D.2.3 C_Kernel

This program is used calculate the statistics from the output of the programs using a rectangular grid, (*TSM_Kernel* and *RDC_Kernel*. The basis program structure is the same as the previous programs, but with *C_Kernel.pm, C_Grid.pm* and *C_Node.pm*. In addition, each 'time step' or instance of *C_Grid.pm* has an array of sub-grids that individually holds each cluster and singleton on the grid; the array holds instances of *M_Grid.pm*, which holds an array of instances of *M_Node.pm*, (*M* signifies Manhattan distance). A schematic of the program is shown in Figure D.5, and the options for invoking it through *C_Kernel.pl* in Figure D.6.

D.2.4 DNH_Kernel

This program holds both the code for the delta notch signalling model used as the randomised automaton model, and the code to perform for the hexagonal grid the statistical analysis equivalent to that found in *C_Kernel.pm*. Consequently its structure follows that of *C_Kernel.pm*, with *DNH_Kernel.pm* providing the entry point and control code, *DNH_Grid.pm* and *DNH_Node.pm* providing the code representing the time step array of grids, and *H_Grid* and *H_Node* providing the code for the instances of sub-grids for each time step grid, (*H* signifies hexagonal grid). A schematic of the program is shown in Figure D.7, and the options for invoking it through *DNH_Kernel.pl* in Figure D.8 and Figure D.9.

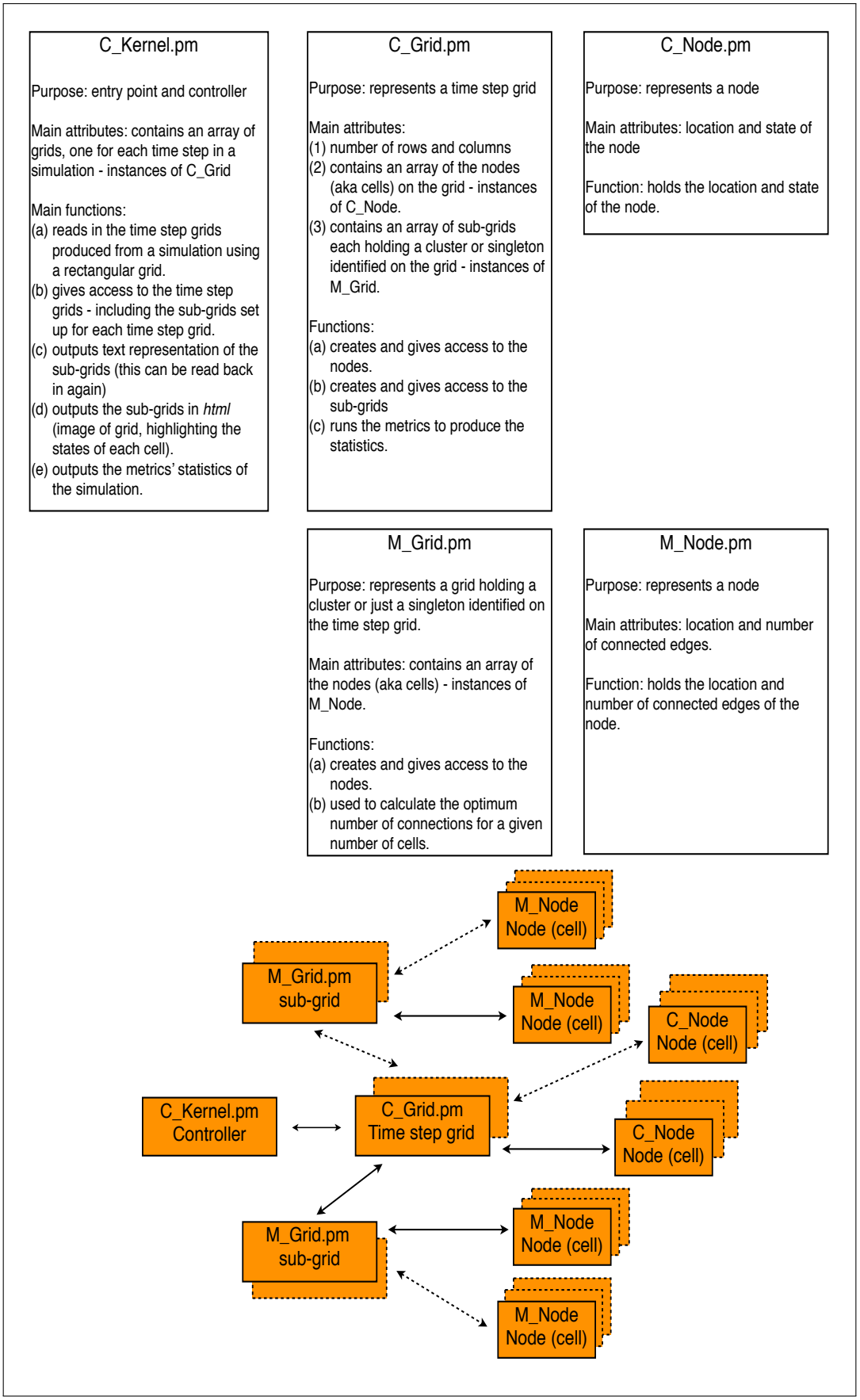


Figure D.5: Schematic of *C_Kernel*.

C_Kernel.pl (generates statistics from the measurement of rectangular grids)

RUN: *perl C_Kernel.pl input_file.txt active nhood range option [option2 [stat name(s)]]*

Parameters:

active:	designates the value that identifies a cell as active - usually 1
n[eighbour]hood	M or V
[Manhatten] range:	1, 2, or 3
option:	*sh* - state as html; output to html file in *displays/*
	st - state as text; output to txt file in *output_texts/*
	ch - connected grids as html; output to html file in *displays/*
	cst - this option has to have *option 2* supplied
option 2:	only used with option *cst,* so *cst 1,* or *cst 2.... cst 8*
	1 - output to screen or to file redirection the connected sub-grids as text
	2 - output to screen or to file redirection the stats as text
	3 - output to screen or to file redirection any experimental stats as text
(this combines 7 & 8)	*4* - get *specified* stats and output to *stats/* directory and create a table and output to *thesis_files/*
	5 - output to screen or to file redirection ALL experimental stats as text
	6 - output to screen or to file redirection a summary of the stats as text
	7 - get stats and output to *stats/* directory
	8 - get *specified* stats and create a table and output to *thesis_files/*

Process: (1) read in date from *input_file*, (2) setup connected sub-grids, then if option is one of *sh, st, ch,* or *cst with no further option* do as above, ELSE if *cst with a further option* do as above.

files created (using the input file TSM_pv_0-6.txt as an example):
The '.txt' is removed from the input file name, giving a *shortname*; e.g. TSM_pv_0-6.txt has a *shortname* = TSM_pv_0-6
The *shortname* is then supplimented with and '_' plus the *active*, *nhood* and *range* parameters, giving a *mid-filename* of TSM_pv_0-6_1M3
This is prefixed with a date / time stamp and a program ID and postfixed with the *option* (except for log files) and the relevant file type.

e.g. logs/24-153938_CK_TSM_pv_0-6_1M3_log.txt
displays/24-153938_CK_TSM_pv_0-6_1M3_sh.html
stats/24-153938_CK_TSM_pv_0-6_1M3_cst-gs.txt
thesis_files/24-153938_CK_TSM_pv_0-6_1M3_cst-sbt8.txt

Figure D.6: *C_Kernel.pl* - Perl script used to invoke the *C_Kernel* program.

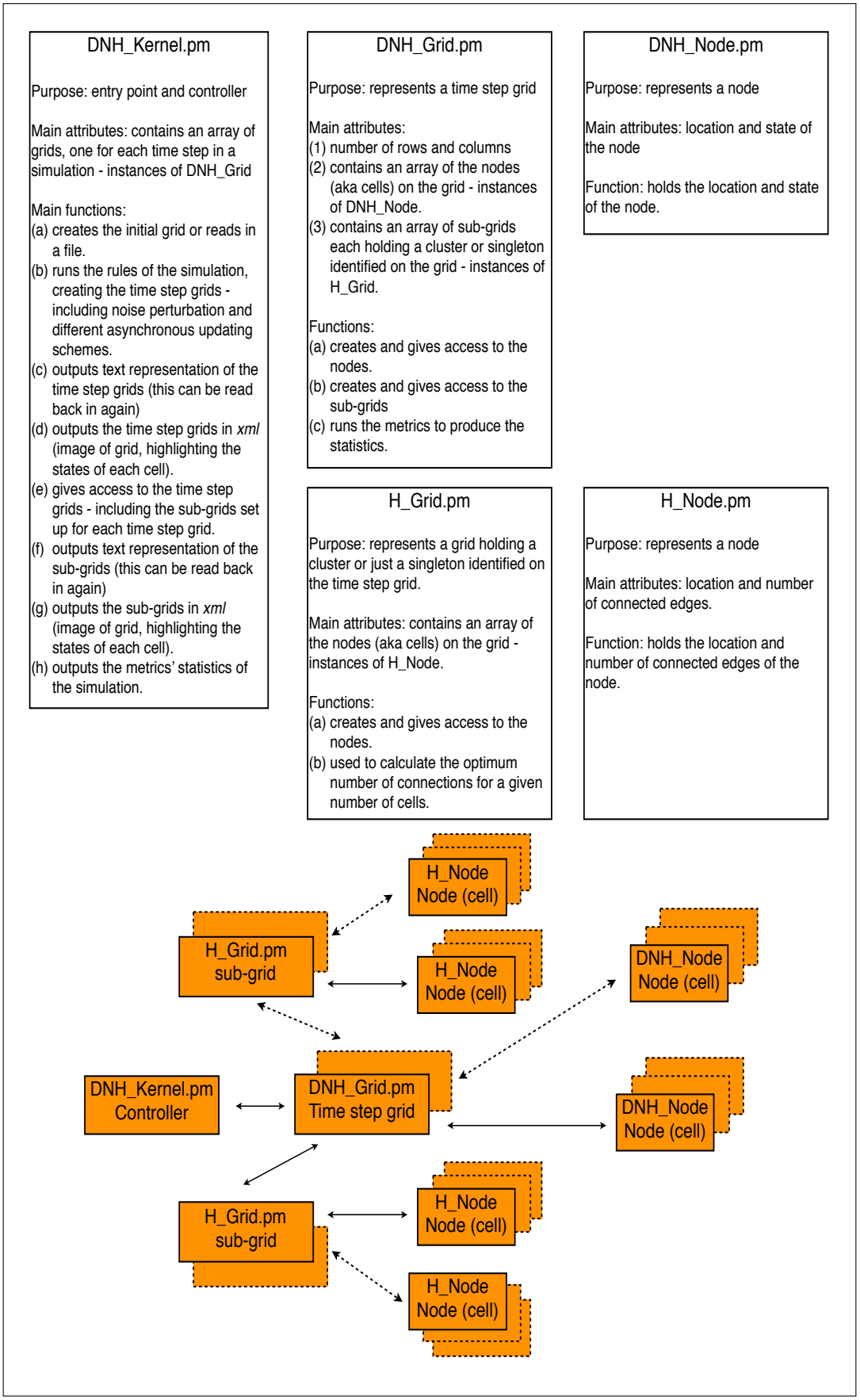


Figure D.7: Schematic of *DNH_Kernel*.

DNH_Kernel.pl (invokes the randomised automaton model of delta notch signalling)

RUN *perl DNH_Kernel.pl active range run-type*
*[*if 'f' or 'F' then -> *input-file output-option [ts-range]]*
OR *[*if 'c' or 'C' then -> *i j threshold range t-noise-prob s-noise-prob loop-count update-type output-option]*

active: - usually 1 (for active cell)
range: - Manhattan distance of 1, 2 or 3
run-type: - *c* or *C* - **C**reates the grids and performs the required action specified in other arguments below
f or *F* - uses an input **F**ile and then actions the output-option
i: - numeric number for rows
j: - numeric number for columns
threshold: - numeric number
range: - 1, 2 or 3 for neighbourhood count
t-noise-prob: - 0 =< temporal noise probability =< 1
s-noise-prob: - 0 =< spatial noise probability =< 1
loop-count: - numeric number
update-type: - *n* or *N* for ROA update WITH noise
- *r* or *R* forRSA update WITH noise
-*a* or *A* for RSA update WITHOUT noise
- [DEFAULT] ROA update WITHOUT noise

output-option:
- test - [c/C] only: return the activity as text
[f/F] only: run *setup_connectedsubgrids*
- all - [c/C] only: (1) save state as grid for future re-runs; saved in *output_grids/* with *_sg.txt*, (2) runs activity as text; saved in *stats/* with *_percentage_at.txt,* (3) gets the stats as text; saved in *stats/* with *_gs_stats.txt* and (4) gets time steps as xml; saved in *displays/* with *_tx.xml*
- tx - [c/C] only: gets time steps as xml; saved in *displays/* with *_tx.xml*
[f/F] only: gets time steps as specified by *ts-range* as xml; saved in *displays/* with *_tx.xml*
- txl - [c/C] only: (1) save state as grid for future re-runs; saved in *output_grids/* with *_sg.txt*, (2) runs activity as text; saved in *stats/* with *_percentage_at.txt* and (3) gets time steps as xml; saved in *displays/* with *_tx.xml*
- st - [c/C] only: save state as grid for future re-runs; saved in *output_grids/* with *_sg.txt*,
- sx - [c/C, f/F]]: save the state as xml; saved in *displays/* with *_sx.xml*

[continued in next figure]

Figure D.8: *DNH_Kernel* - invokes the *DNH_Kernel* program. [*continued in Figure D.9*].

*[**DNH_Kernel.pl - continued from previous Figure**]*

*output-option**: [**continued]*

- csgx - [c/C] only: save connectedsubgrids as xml; saved in *displays/* with *_csgx.xml*
  [f/F] only: (1) set up the connectedsubgrids, (2) save connectedsubgrids as xml; saved in *displays/* with *_csgx.xml*
- gs - [c/C] only: gets the stats as text; saved in *stats/* with *_gs_stats.txt*
  [f/F] only: (1) set up the connectedsubgrids, (2) gets the stats as text; saved in *stats/* with *_gs_stats.txt*
- sg - [c/C] only: get the state as grids for text output; save in *output_grids/* with *_sg.txt*
- sed - [f/F] only: (1) save timesteps as xml; saved in *displays/* with *_sx.xml,* (2) turn file into a pdf by calling *pdffile.sh*

ts-range: - time steps to be processed; if none specified then all time steps are processed

Process:
(A) if creating a file then (1) the hexagonal grid is created using *i* and *j*, (2) the simulation is run using *threshold range t-noise-prob s-noise-prob loop-count update-type*, (3) the connectedsubgrids are set up, and (4) the *output-option* is actioned.
(B) if reading in a file then (1) read in file and (2) action the *output-option*
(C) if no parameters are supplied then (1) a random grid of 10 by 10 is created, (2) it is run with [1, 1, 0.4, 0.6, 1, R] and (3) the time steps printed out as xml

files created:
Log files: everything is logged to a log file related to the run of the program
logs/ + month day + - + hour + minute + second + _DNH_ + active + H + range + i + by + j + t + threshold + r + range + tn + t-noise-prob + sn + s-noise-prob + ut + update-type + _log.txt
e.g. logs/19-141344_DNH_1H3_20by20t1r1tn0.1sn0utR_log.txt

The output files are also made up and put in relevant directories:
directory/ + month day + - + hour + minute + second + _CK_ + inputfile pre . name + active + nhood + range + *_option.file-type*

Figure D.9: *DNH_Kernel* - invokes the *DNH_Kernel* program. [*continued from Figure D.8*].

D.2.5 S_Data

This program performs the analysis of the statistics produced by *C_Kernel* and *DNH_Kernel*, producing graphs and tables suitably formatted for inclusion in LaTeX. Its format differs from the previous programs as it deals with statistical information, not grids. The entry and control program is *S_Data.pm*, which has a hash array of samples (*S_Sample.pm*), each of which holds a hash array of statistics (*S_Stats.pm*). A schematic of the program is shown in Figure D.10, and the options for invoking it through *S_Datal.pl* in Figure D.11.

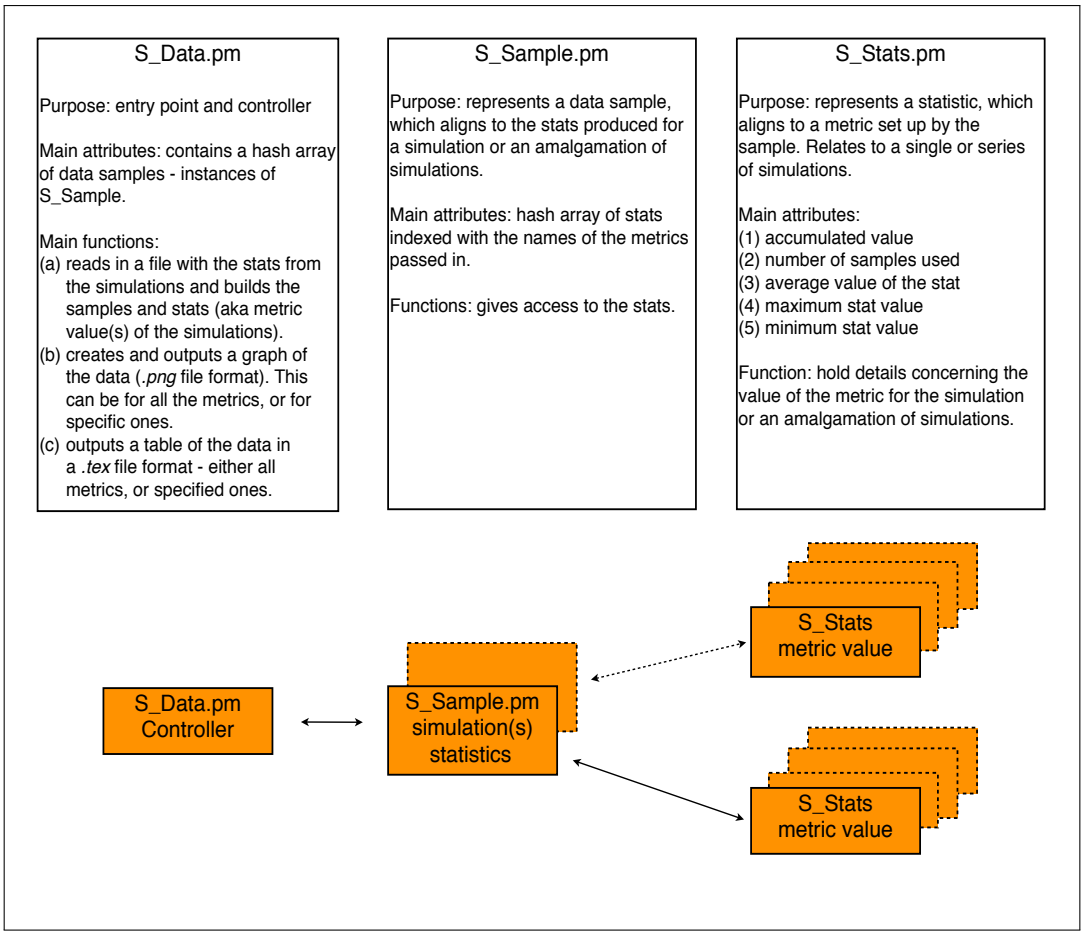


Figure D.10: Schematic of *S_Data*.

S_Data.pl (generates graphs and tables from the statistics produced from C_Kernel and DNH_Kernel)
This controls the processing of stat blocks passed in for analysis.

RUN *perl S_Data.pl input_file [option] [output_file] [stats-names]*

parameters:

option:	[default]	TEXT	- return stats specified in *stats-names* as a stats block as text (if no option, then defaulst to ALL stats)
		TABLE	- return stats specified in *stats-names* as a stats block as a table
		GRAPH	- return stats specified in *stats-names* as a stats block as a graph in *.png* file whose name has to be supplied before the *stats-names*
		STAT	- returns a stats block for a single *stat-name* (first of any supplied)
		DIFF	- returns difference between average, min and max of ALL stats
		DATA	- returns textual breakdown of ALL stats
stats-names:		*1 or more*	*- MeanDensity, Entropy, BBR, C-Value*

Process: read in input file into a S_Data object, then:
(a) if no second option or the second param is TEXT, print to screen the stats block as text (can be redirected there) - either all the stats or based on further parameters (they must all be valid ones)

OR

(b) if TABLE, there must be at least another parameter which is the output file; if no further params then all stats, else based on further parameters (they must all be valid ones)

OR

(c) if GRAPH, there must be at least another parameter which is the output *.png* file;if no further params then all stats, else based on further parameters (they must all be valid ones)

OR

(d) if STAT, DIFF or DATA the output is printed to the terminal (can be redirected from there into a file)

Figure D.11: *S_Data.pl* - used to invoke the *S_Data* program.

D.3 Java program

The design of the deterministic automaton model of a theoretical active network is outlined in section 4.8. The program is written in Java as an extension of the MUSCLE Java based toolkit that provides the framework to build a CxA, (*see appendix A*). The key components of a MUSCLE built distributed CxA application are kernels, portals, conduits and a communication protocol. The entry and control module for the deterministic automaton model is *ActiveNodeFlow_DS*; this holds the array of time step grids and links via a conduit to *ActiveNode_DS*, which itself has a conduit to *ActiverNodeProcessor_DS* module. The operation of the program is controlled by the *ActiveNodeModel_DS.ini* file, which is used to set variables; for example, the size of the active network grid, the resources allocated to each node, the process to carry out (such as Replicate), and a node's memory requirements. The messages between the modules are text based. The code, *ini* file and basic manuals can be found for the Java and Perl programs on the CD distributed with the thesis.