University of Leipzig
Institute for Theoretical Physics

Diploma thesis

Patterns in the bit string model of idiotypic networks

Sven Willner

Supervisor: Prof. Dr. Ulrich Behn

Leipzig, February 22nd, 2011

Contents

Introduction **3**

1 The model **7**
- 1.1 Design . . . 7
- 1.2 Analytical concepts . . . 9

2 Patterns and arrangements **13**
- 2.1 Determinant bits . . . 14
- 2.2 Arrangements . . . 17
- 2.3 Pattern examples . . . 22
- 2.4 Pattern stability . . . 38

3 Decompositions and tilings **45**
- 3.1 Cayley graphs . . . 46
- 3.2 Tilings . . . 47
- 3.3 The block-link matrix and pattern state vectors . . . 55

4 Symmetry **61**
- 4.1 Symmetries of the underlying network . . . 61
- 4.2 Pattern symmetries . . . 68

5 Extensions **73**
- 5.1 Link weightings . . . 73
- 5.2 Further occupation states . . . 76
- 5.3 Generalised antibody interactions . . . 78

6 Antigens and their influence on the network **81**
- 6.1 Subpattern transitions . . . 82
- 6.2 Outlook for introducing antigens . . . 98

Conclusion **101**

Bibliography **104**

Introduction

The immune system is a very complex system crucial for the survival of higher organisms. Its purpose is the protection of the organism against pathogens (such as fungi, viruses or bacteria) by identifying and destroying them. Also cancer cells and toxins are targets of it. Depending on the complexity of the organism the immune system has different possibilities to react.

Higher vertebrates such as fish, amphibians, reptiles and mammals developed mechanisms to respond unspecifically on pathogens, but also a specific immune response. The unspecific immune system yields immune responses such as inflammation that occur unspecifically due to intrusion of a pathogen. It is congenital and able to fight against a variety of pathogens, but is unable to establish a lasting immunity. The specific or adaptive immune system on the other hand reacts specifically on the pathogen and can in many cases build up a memory of that pathogen. This memory yields in a more direct and more efficient reaction the next time a pathogen of that type is encountered. Even a lasting immunity against it can be set up that way, which is the basic idea of vaccination. A very brief overview of the adaptive immune system is given here. Details can be found in [1], for instance.

Lymphocytes, a kind of white blood cells mainly produced in the bone marrow, carry out the main functions of the adaptive immune system. Two groups of them are known: T-lymphocytes (1) responsible for destruction of pathogens (cytotoxic T-cells) as well as (2) those for activation of other lymphocytes (T-helper cells) and (3) those for regulation of immune responses (regulatory T-cells). The other group, B-lymphocytes (B-cells), proliferate antibodies. Some lymphocytes can furthermore differentiate into so-called memory cells.

Antibodies are Y-shaped proteins used for marking and liquidating pathogens (see figure 0.1). They consist of constant binding regions and variable

one, which have different forms in a variety of amino acid sequences. The variable regions bind, depending on their structure to certain *antigens*, which can be any type of substance or molecule. They are located on each arm of the Y and of the same type for each arm. This type of the variable regions that is specific for the antibody and the antigens it can bind to, is called *idiotype*. The constant regions, on foot of the Y, determine how the antigen is treated after being bound by the antibody, e. g. it is marked for destruction by macrophages.

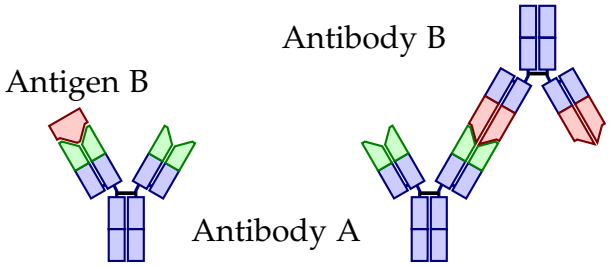


Figure 0.1: Schematic of antibodies and an antigen. The constant regions of the antibodies are coloured in blue, the variable regions are coloured in green and red. It is schematically shown how an antibody (type A) can bind to either an antigen or another antibody of complementary type (type B).

Each B-cell expresses antibodies of a certain idiotype on its surface, grouped together in surface immunoglobulin (sIg). If the variable regions of two of these antibodies are bound by another molecule and are thus cross-linked, the B-cell is stimulated. Upon stimulation B-cells differentiate into plasma cells, which produce and then release antibodies of the same idiotype (actually also some hypermutation can occur). In the bone marrow on the other hand the idiotype of antibodies on a B-cell is determined by gene segments. These encode different parts of the variable regions of these antibodies. They are shuffled during development of a lymphocyte and thus yield a supply of B-cells of fairly random idiotypes. These two mechanisms empower the immune system to produce antibodies of a huge variety of idiotypes which help to specifically fight pathogens. It is believed that all sorts of antigens in principal can be recognized by antibodies.

In fact antibodies are also capable of binding to other antibodies of complementary type (see figure 0.1). Hence they can also stimulate B-cells and the system of B-cells and their antigens becomes self regulating. This led Jerne to his theory of *idiotypic networks* published in 1974 [2]. The adaptive immune system is modelled in a network of idiotypes, concentrations of antibodies of

one idiotype influencing those of another. Whereas the network paradigm was very popular in the 1980s, it went out of fashion in the 1990s due to huge progress in molecular immunology. However, networks are better understood nowadays and are used in the uprising system biology to study phenomena from a systematic point of view. A renaissance of the network idea can be perceived, for instance, as given in a review by Behn [3]. Among other things, the network attempt is believed to improve the understanding of autoimmune diseases and the immunological memory.

This work is based on a network model developed by Brede and Behn in 2003 [4], the *bit string model*. In it the system is modelled by a probabilistic cellular automaton on a network. For a wide range of parameters stable states are reached, so-called *patterns*. They are of particular interest in the understanding of the system as their possibility and stability strongly contribute to its behaviour. Subject of this work is the recognition, classification and description of these patterns in terms of their structure and stability.

Chapter 1 presents the model and some basic methods for analysis. Apart from confirming previously discovered results, a visualisation of patterns is developed in chapter 2. A catalogue of many patterns found is given and discussed in matter of stability. As it turns out pattern classification methods so far are not sufficient to describe all patterns found. A method for more general patterns is introduced in chapter 3. Using an algebraic concept for network description it is proven to be complete in certain situations. System analysis done in chapter 4 supports the concept. It gives a hint for a general complete pattern classification which is a combination of the previously developed and the newly introduced concept here. For some patterns, namely the purely dynamic ones, symmetry is furthermore discussed to ascribe them to a root pattern. As the model is fairly abstract extensions are manifest to bring it closer to its real nature counterpart. Chapter 5 gives some of these extensions and their influence on the patterns emerging. Concepts and methods given here are then applied in chapter 6 to deepen the study of a scenario with an intruding antigen that was started in a previous work. The antigen induces patterns that can be interpreted as the memory of the antigen in the network. This inducing is explained and the internal images of the antigen are determined. Summing up, the main result of this work is classification of general patterns that helps understanding mechanisms such as emerging subpatterns, for instance, induced by an antigen.

1 The model

Subject of investigation in this work is the bit string model of idiotypic networks as developed in [4] in 2003. It is presented in this chapter together with some basic means of analysis. In later chapter it is simply referenced by "the model".

1.1 Design

The *bit string model of idiotypic networks* is a probabilistic cellular automaton. Its cells are discrete, they form indeed a network, and its states develop along a random walk in state space. Though its basic structure and dynamic rules are fairly simple, a vast dynamic behaviour emerges.

1.1.1 Underlying network

The basic structure of the model is an undirected graph $G_d^{(m)} := (\mathcal{V}, \mathcal{E})$ (the terms graph and network are used synonymously throughout this work). Every node is a bit string of length d, i. e. of the form $\mathbf{v} = v_1 \dots v_d$ with $v_i \in \{0, 1\}$. Thus, $\mathcal{V} := \{0, 1\}^d$, and the number of nodes is $|\mathcal{V}| = 2^d$. Every bit string is a model representation of an idiotype, more precisely a representation of the structure of the binding site (idiotopes) of an antibody of that idiotype. Bit strings have been chosen according to the key-lock principle: antibodies can bind to other antibodies, as long as their binding site structure is fairly complementary. In terms of bit strings, two antibodies are modelled to be able to bind, if the bit strings representing their kind are complementary in all bits except up to m bits, the so-called *mismatches*. In other words, they are able to bind, if the corresponding bit strings differ in at least $d - m$ bits. The number

of bits two bit strings differ in is called their *Hamming distance*. For example, the complement of 10110 is 01001 (in Hamming distance 5) and 11001 is its complement with one mismatch in the first bit (in Hamming distance 4).

In the graph, nodes are connected according to the ability of the corresponding antibody types to bind. Thus, the edge set of the graph (i. e. the set of all pairs of nodes that are connected to each other) is defined as:

$$\mathcal{E} := \{(\mathbf{u}, \mathbf{v}) \in \mathcal{V} \times \mathcal{V} : d_H(\mathbf{u}, \mathbf{v}) \geq d - m\} \tag{1.1}$$

($d_H(\mathbf{u}, \mathbf{v})$ is the Hamming distance: the number of bits differing between bit strings $\mathbf{u}$ and $\mathbf{v}$). Nodes that are directly linked to another one are called its *neighbours*. To avoid pathological situations, the number of mismatches m is chosen to be $0 < m < d - 1$ (For $m = 0$ nodes are only linked to their complements, the graph decomposes into pairs without connection in between. For $m \geq d - 1$ the graph is complete, i. e. every node is connected to every other).

Each node has two possible states: *occupied* or *unoccupied*. Occupation represents the presence of antibodies of the corresponding type within the body. The state of a node $\mathbf{v}$ at time $t \in \mathbb{N}$ is denoted by $n_t(\mathbf{v}) \in \{0, 1\}$ (0 for unoccupied, 1 for occupied). The totally unoccupied network is often referred to below as the empty network.

1.1.2 Dynamic rules

The occupation of nodes develops over time and is subject to simple dynamic rules that are applied during each discrete time step in following order:

1. Influx

The main source for antibodies in the body is the bone marrow, that produces lymphocytes of various idiotypes. This random supply of idiotypes is modelled by occupying nodes randomly: Each unoccupied node is occupied with probability p in the first stage of every iteration. p is a parameter of the model and as well independent of time and node.

2. Update

Since antibodies are carried by B-lymphocytes, which have limited life time, the presence of an idiotype depends on the stimulations of the corresponding lymphocytes, which triggers their reproduction. A B-lymphocyte is only stimulated by cross-linking two of its antibodies expressed

on the surface by one matching antibody (antibodies are Y-shaped proteins with binding sites on each arm). Experimental results and theoretical considerations (see [5], for instance) highly suggest a log-bell-shaped stimulation-response curve, that is simply modelled here by a rectangular window function: If there are less matching idiotypes than a lower threshold t_l present, stimulation is not high enough and the corresponding node is unoccupied. If there are more matching idiotypes than an upper threshold t_u present, possible binding partners constrict each other and cross-linking becomes unlikely, the corresponding node is also unoccupied.

The number of binding partners of an idiotype represented by the node $\mathbf{v}$ is the number of its occupied neighbours:

$$\partial n(\mathbf{v}) = \sum_{\mathbf{v}' \in \mathcal{V};\, d_H(\mathbf{v},\mathbf{v}') \geq d-m} n(\mathbf{v}'). \tag{1.2}$$

Is the node occupied and does the number of its occupied neighbours fulfil the window rule, $\partial n(\mathbf{v}) \in [t_l;\ t_u]$, the node stays occupied. Otherwise it is unoccupied. The update is applied in parallel on all nodes.

These two steps are iterated over time.

1.2 Analytical concepts

Up to now several analytical concepts have been used to study this model. Some of them are again used in this work.

1.2.1 Centre of mass

In [6, ch. 9] the analytical concept of the centre of mass vector is introduced.

Definition: For the time $t \in \mathbb{N}$ the *centre of mass vector* is defined as:

$$\mathbf{R} = \mathbf{R}_t := \frac{1}{\sum_{\mathbf{v} \in \mathcal{V}} n_t(\mathbf{v})} \sum_{\mathbf{v} \in \mathcal{V}} n_t(\mathbf{v}) \cdot (2 \cdot \mathbf{v} - \mathbf{1}^{(d)}), \tag{1.3}$$

with $\mathbf{1}^{(d)} = (1, \ldots, 1)^T \in \{0,1\}^d$.

In words, the centre of mass vector component is calculated by summing over the nodes. If an occupied bit vector has a bit set 1 in the corresponding com-

ponent it contributes to the sum with 1, if the bit is unset to 0 it contributes with -1, if the node is unoccupied it is not taken into account. The sum is then normalized by the number of occupied nodes.

Therefore a positive centre of mass vector component indicates that more bit vectors are occupied whose corresponding bit is set 1 than those with that bit unset to 0. A close-to-zero component on the other hand indicates no preference in the setting of the corresponding bit.

It is a great instrument for identifying groups of nodes such as in the determinant bit concept discussed in section 2.1.

1.2.2 Shannon entropy

Another concept proves useful: the Shannon entropy as defined in information theory.

Definition: The *Shannon entropy* shall here be used in the form ($p_{\mathbf{v}}$ is the mean occupation of node $\mathbf{v}$):

$$S = -\frac{1}{|\mathcal{V}|} \sum_{\mathbf{v} \in \mathcal{V}} \left(p_{\mathbf{v}} \cdot \ln(p_{\mathbf{v}}) + (1 - p_{\mathbf{v}}) \cdot \ln(1 - p_{\mathbf{v}}) \right), \tag{1.4}$$

Hence it is not a quantity of a microstate (being a snapshot at a given time $t \in \mathbb{N}$) but a quantity of a macrostate.

As seen in figure 1.1 and easily verifiable, the summand for every node $\mathbf{v} \in \mathcal{V}$, $S_{\mathbf{v}}(p_{\mathbf{v}}) = -p_{\mathbf{v}} \cdot \ln p_{\mathbf{v}} - (1 - p_{\mathbf{v}}) \cdot \ln(1 - p_{\mathbf{v}})$, is symmetrical around and maximal at $p_{\mathbf{v}} = \frac{1}{2}$. The Shannon entropy therefore gives a measure of "variability" of a macrostate of the network: the more a node is mean-wise occupied or unoccupied the smaller its summand in the entropy. In other words: the smaller the entropy of a macrostate the more its nodes show constant occupation; the smaller the entropy the more static the pattern, the larger the entropy the more dynamic it is. In this work the Shannon entropy was used to computationally distinguish macrostates reached by the network. For instance, in figure 2.1 the Shannon entropy was calculated to show patterns a network with given parameters reaches in mean thereby taking advantage of the Shannon entropy supplying information about the whole network in one measure.

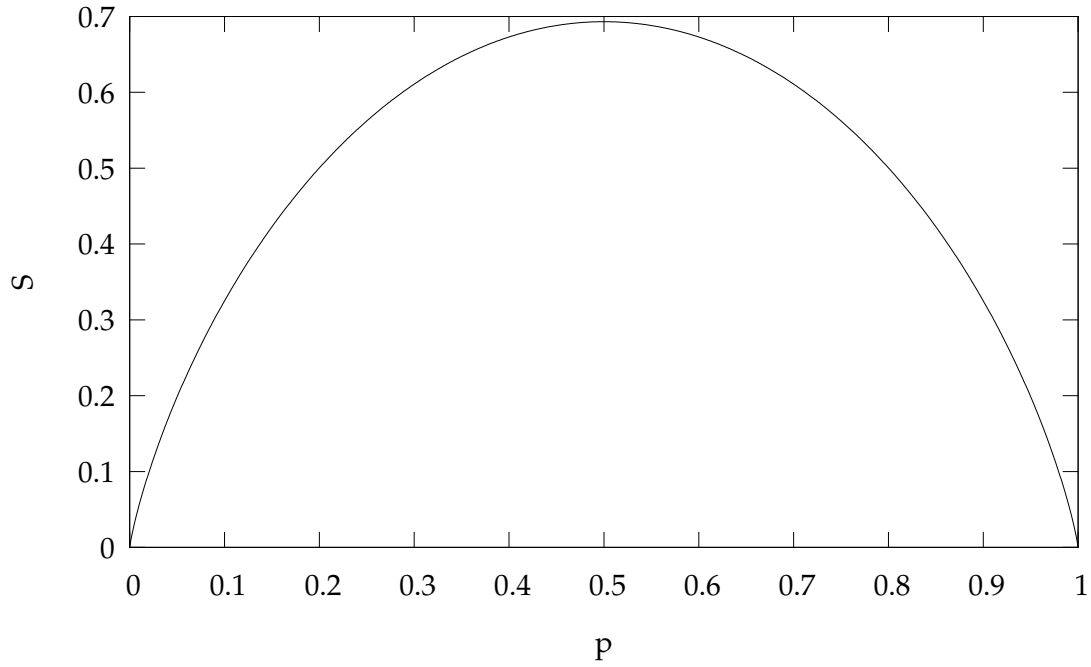


Figure 1.1: Plot of the function $S(p) = -p \cdot \ln p - (1-p) \cdot \ln(1-p)$.

With analysis methods like these at hand, computer simulations can be run. This is done by starting with a network in a certain state. In most cases in this work it is the totally unoccupied network (the "empty" network). Then, the two iteration steps are iterated over and over again. In parallel the analysis methods can be applied. In many cases the network has a transitional phase first, in which it is mostly filled by the influx and takes a few hundred iterations to develop a stable state. These will be studied further in the following chapters.

Patterns and arrangements

2

For a wide range of parameter choices the system reaches a steady (macro)state after some iterations during a transitional phase. Such macrostates are characterised by a constant mean occupation for each node over some time steps. In most cases nodes can also be grouped together according to their statistical properties in states (such as mean occupation, mean lifetime, mean neighbour occupation). These macrostates with a regular spacial structure shall be called *patterns*. A kind of pattern (called *architecture* in some previous works) can appear in different realisations or "rotations" of a pattern of that kind. Classifying and analysing these patterns has been done intensively in previous works and shall further be extended here.

For classification of patterns there are several ways of grouping together nodes. The allocation in the largest groups possible according to a grouping concept is to aim for and shall be used here when referring to a class of patterns. One way of establishing these groups, the concept of determinant bits, is described in this chapter and used to develop an arrangement of the network to allow quick recognition of a pattern and its substructure. Looking at the substructure of various examples the concept of determinant bits turns out not to be sufficient to describe all patterns occurring.

The Shannon entropy eq. (1.4) is a great tool for exploring patterns appearing in simulations since many patterns of different kinds have a different Shannon entropy. In figure 2.1 it was computed for 100 networks in a range of influx parameter p for the parameter setting that was mostly used in previous works, $d = 12,\ m = 2,\ [t_l; t_u] = [1; 10]$. Plotting the histogram of Shannon entropies (the blacker a point in the diagram the more often the corresponding Shannon entropy occurred) some patterns can be distinguished. The most important of those were marked with their kind as proposed by determinant bits (see

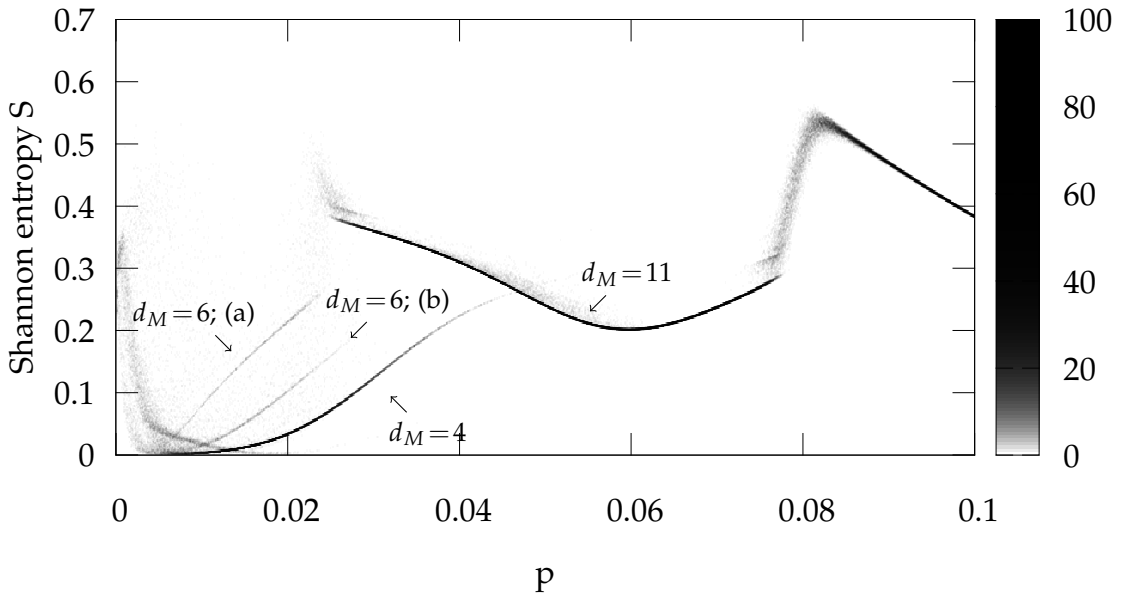


Figure 2.1: Histogram of the Shannon entropy. For each of 1000 p-values 100 simulations were run starting from an empty network. After 500 iterations the Shannon entropy was taken over 10^4 iterations. Some branches are marked according to the corresponding pattern as described in the following sections. There are two different kinds of $d_M = 6$ patterns. An example for kind "(a)" is shown in figure 2.8 (p. 27), for kind "(b)" see figure 2.9 (p. 27). [$d = 12$, $m = 2$, $[t_l; t_u] = [1; 10]$]

next section). For values of influx parameter p larger than $p_{max} \approx 0.077$ the network shows constantly rotating $d_M = 11$ patterns (i. e. ever changing to different realisations of the same kind of pattern). The larger p the higher the rotation rate, which eventually exceeds the possible resolution. Thus, no steady macrostate is reached and the impression of a fairly random distribution of mean occupation of the nodes is left.

2.1 Determinant bits

A possible concept of classifying patterns has been introduced in [7]: The concept of determinant bits and the consequent allocation of the nodes to groups S_i.

Definition: A choice of $d_M \in \{0, \ldots, d\}$ *determinant bits* is a choice of positions

$$(i_1, \ldots, i_{d_M}) \in \mathbb{N}^{d_M}, \quad 1 \leq i_1 < i_2 < \ldots < i_{d_M} \leq d, \tag{2.1}$$

of those determinant bits. They moreover shall have determinant values

$$(l_1, \ldots, l_{d_M}) \in \{0,1\}^{d_M}. \tag{2.2}$$

Given a choice of determinant bits the nodes are allocated into groups depending on their values in the determinant bits. Nodes of a group S_i have i variations from the determinant values:

$$S_k = \{\mathbf{v} \in \mathcal{V};\, k = \sum_{j=1}^{d_M} |l_j - v_{i_j}|\}. \tag{2.3}$$

Of course for d_M determinant bits there are groups $S_0, \ldots, S_{d_M}$ (from 0 variations till a maximum of d_M variations from the d_M determinant bits). For $k \in \{0, \ldots, d_M\}$ there are $\binom{d_M}{k}$ possible positions in which a node in group S_k can differ from the values of the determinant bits. For each of these variations there are $2^{(d-d_M)}$ possible bit values in non-determinant bit positions. Therefore the size of each group S_k is $|S_k| = 2^{(d-d_M)}\binom{d_M}{k}$.

Many patterns have been shown to fit into that group occupation, i. e. nodes in a group sharing the same statistical properties like mean occupation. Furthermore in most cases there is a direct relation between the determinant bits of a pattern and its centre of mass vector eq. (1.3):

- Components of the centre of mass vector with zero value indicate that the corresponding bit is non-determinant.
- Components with a positive value indicate a determinant bit with determinant value 1 at the corresponding position.
- Components with a negative value indicate a determinant bit with determinant value 0.

In many patterns the values of the components belonging to determinant bits have the same absolute value which itself depends on the number of indicated determinant bits. The exact value, $\frac{2}{d_M}$, can be determined theoretically as done in [6, ch. 9]. On the other hand, patterns emerge that have several different absolute values of non-zero components of the centre of mass vector (the according determinant bits were named ”primary”, ”secondary”, etc. in [6, sec. 4.4]). If a d_M is given here for a pattern, it is the number of determinant bits as indicated by the centre of mass vector. Though not always appropriate for the

consequent allocation of nodes into groups it turns out to be the best description of a pattern in terms of determinant bits.

Definition: If the components of the centre of mass vector corresponding to determinant bits are of same value, the determinant bits shall be named *simple*. If they are of different value, the determinant bits are *non-simple*.

In case of non-simple determinant bits, $d_M^{(1)}$ denotes the number of determinant bits with the largest absolute value in corresponding components of the centre of mass vector (*primary determinant bits*), $d_M^{(2)}$ is the number of determinant bits with the second largest absolute value in corresponding components (*secondary determinant bits*) and so on (for an example see figure 2.2).

In the following, for non-simple determinant bits the number of determinant bits of each priority is given by the notation $\mathbf{d}_M := (d_M^{(1)}, \ldots, d_M^{(k)})$. For simple determinant bits only d_M, the number of all determinant bits, is given.

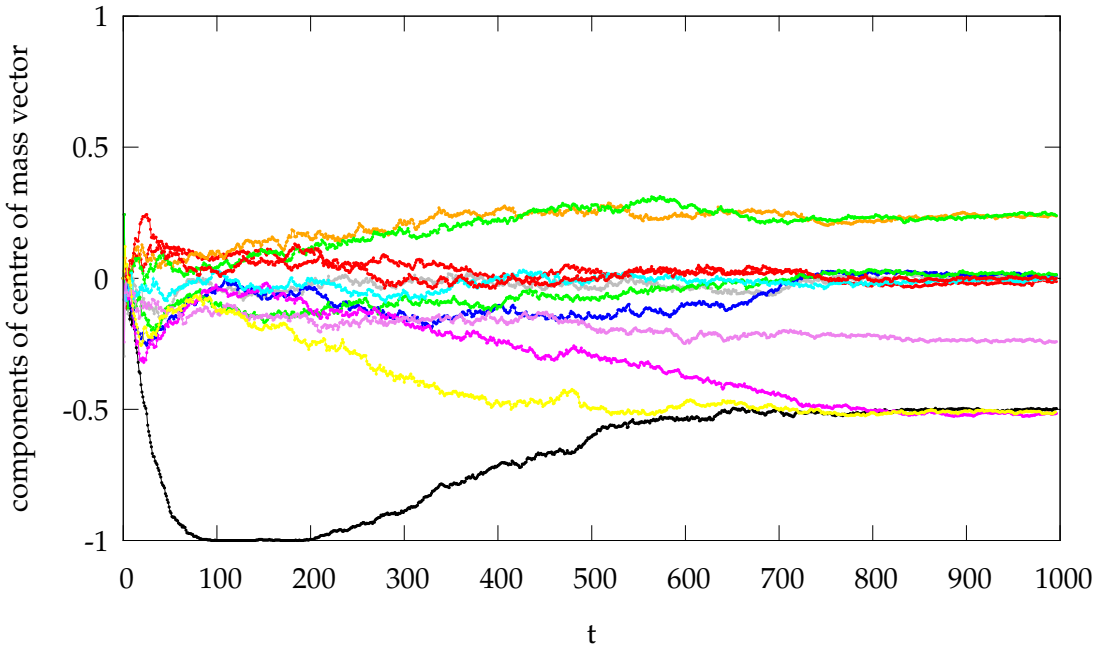


Figure 2.2: Components of centre of mass vectors; starting from a totally unoccupied network. The last 200 components correspond to non-simple determinant bits with $\mathbf{d}_M = (3,3)$: three components have value -0.5, the corresponding bits are the primary determinant bits with determinant value 0. Another component has value -0.25 and two further ones have value $+0.25$. Since they all have absolute value $0.25 < |-0.5|$, they correspond to the secondary determinant bits (one with determinant value 0 and two with 1). All the other components are almost 0 (corresponding to the non-determinant bits). $[d=12,\ m=2,\ p=0.015,\ [t_l;t_u]=[1;10]]$

2.2 Arrangements

In this chapter an arrangement of the nodes on a two-dimensional grid is developed. Its purpose is to visualise their occupation in a manner that enables quick recognition of appearing patterns and even substructures. Since the network consists of 2^d nodes an arrangement on a grid of the size $2^{\lfloor \frac{d}{2} \rfloor}$ x $2^{\lceil \frac{d}{2} \rceil}$ suggests itself. Further on the Gauß brackets will be used extensively:

$$\begin{aligned} \lceil x \rceil &:= \min\{y \in \mathbb{N} : y \geq x\} && \text{upper Gauß bracket} \\ \lfloor x \rfloor &:= \max\{y \in \mathbb{N} : y \leq x\} && \text{lower Gauß bracket,} \end{aligned}$$

they are simply the formal way of rounding "up" or "down".

For $d = 12$ the grid is a square with an edge length of $2^6 = 64$ grid points. Every grid point in such a grid represents a network node and can be coloured depending on its occupation and belonging to groups defined by determinant bits. For the pattern to be seen the nodes should be arranged in a way that nodes in the group S_n turn out to be close to the groups S_{n-1} and S_{n+1}.

It shows up that for certain patterns the allocation to groups S_i is broken. Therefore a more general allocation to equally-sized blocks is proposed. This new allocation furthermore allows calculation of possible static patterns. Understanding those is crucial for the understanding of dynamical patterns and their stability.

Let d_M be the number of chosen determinant bits. Without loss of generality those are said here to be the first d_M bits and the non-determinant bits are the $d - d_M$ bits at the right end of the bit chain:

$$a_1 \dots a_{d_M} b_1 \dots b_{d-d_M},$$

with $a_1, \dots, a_{d_m}$ determinant, $b_1, \dots, b_{d-d_m}$ non-determinant.

Moreover without loss of generality, let the values of the determinant bits be chosen to 0 (otherwise the bits used here describe a variation from the value of the determinant bits of 1 (respectively no variation for the value of 0)).

2.2.1 Arrangement of blocks

First of all the rectangular grid is divided into blocks, a block being the set of nodes sharing the same values in all determinant bits (i. e. all values of non-determinant bits vary among the nodes of one block and only these values do). Since there are 2^{d-d_M} possible values of non-determinant bits each block is of size 2^{d-d_M}.

The blocks of a column of blocks should have equal values in determinant bits, likewise the blocks in a row of blocks. This can be done by varying the first $\lceil \frac{d_M}{2} \rceil$ determinant bits along columns of blocks and the values of the last $\lfloor \frac{d_M}{2} \rfloor$ determinant bits along rows of blocks.

The first column has no variation in the first $\lceil \frac{d_M}{2} \rceil$ determinant bits, the next columns have one variation, the next columns have two and so on. Similarly for the rows in the last $\lfloor \frac{d_M}{2} \rfloor$ determinant bits.

In the case of $d = 12$ and $d_M = 6$ this method yields following arrangement ($\bullet$ stands for bits varying along a row or column respectively):

Determinant bits of the...

... 1st column:	0 0 0 • • •	... 1st row:	• • • 0 0 0
... 2nd column:	1 0 0 • • •	... 2nd row:	• • • 1 0 0
... 3rd column:	0 1 0 • • •	... 3rd row:	• • • 0 1 0
... 4th column:	0 0 1 • • •	... 4th row:	• • • 0 0 1
... 5th column:	1 1 0 • • •	... 5th row:	• • • 1 1 0
... 6th column:	1 0 1 • • •	... 6th row:	• • • 1 0 1
... 7th column:	0 1 1 • • •	... 7th row:	• • • 0 1 1
... 8th column:	1 1 1 • • •	... 8th row:	• • • 1 1 1

In total the determinant bits of the blocks turn out to be:

000000	100000	010000	100000	110000	101000	110000	111000
000100	100100	010100	100100	110100	101100	110100	111100
000010	100010	010010	100010	110010	101010	110010	111010
000001	100001	010001	100001	110001	101001	110001	111001
000110	100110	010110	100110	110110	101110	110110	111110
000101	100101	010101	100101	110101	101101	110101	111101
000011	100011	010011	100011	110011	101011	110011	111011
000111	100111	010111	100111	110111	101111	110111	111111

Colouring blocks depending on their belonging to groups yields:

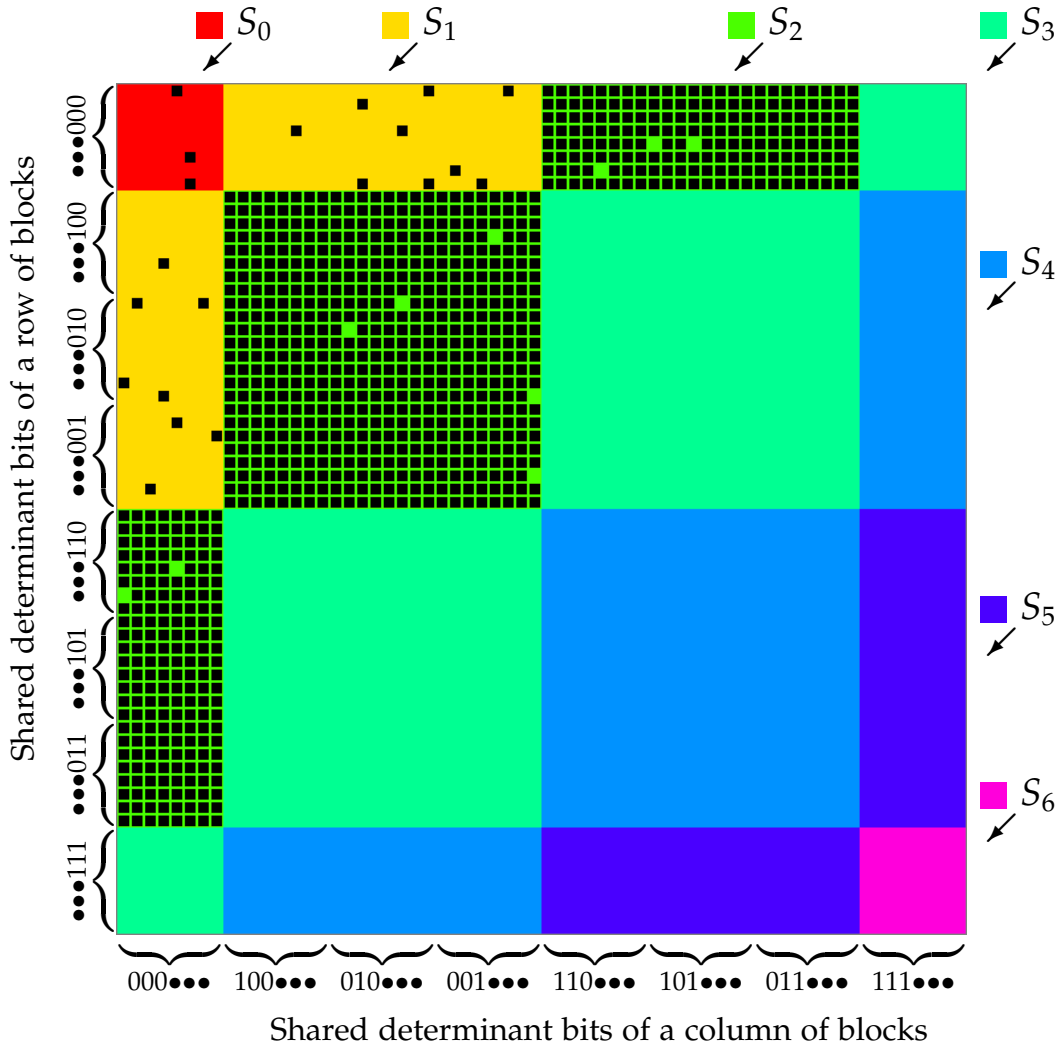


Figure 2.3: Snapshot of a network state with $d_M = 6$, a black coloured grid point represents an occupied node. Grid points have also been coloured according to the group the corresponding node belongs to (group names given above and at the right). [$d = 12$, $m = 2$, $p = 0.03$, $[t_l; t_u] = [1; 10]$]

2.2.2 Inner-block arrangement

The nodes in a block are arranged in the same way like the blocks in total: Values of the first $\lfloor \frac{d-d_M}{2} \rfloor$ non-determinant bits are varied along the columns and the values of the last $\lceil \frac{d-d_M}{2} \rceil$ non-determinant bits along the rows. The first column have no bit set 1 in the first $\lfloor \frac{d-d_M}{2} \rfloor$ non-determinant bits, the next columns have one, the very next have two and so on. Similarly for the rows in the last $\lceil \frac{d-d_M}{2} \rceil$ non-determinant bits.

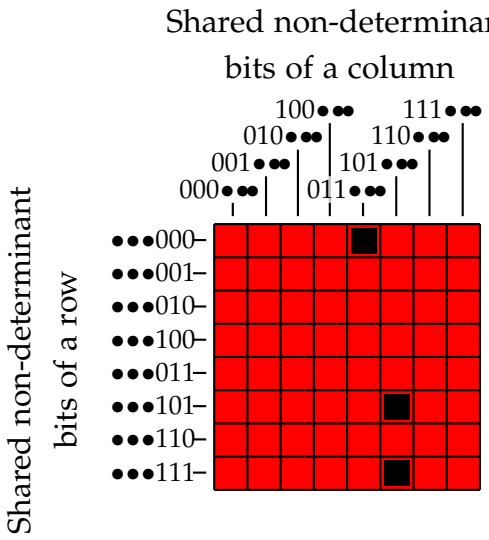


Figure 2.4: Block from figure 2.3, here, the group S_0 consisting of solely one block. A black coloured grid point again represents an occupied node.

2.2.3 Extension to non-simple determinant bits

In case of non-simple determinant bits the priority of the bits has to be taken into account. In the columns further to the left or rows closer to the top determinant bits of lesser priority vary. In columns further to the right or rows closer to the bottom determinant bits of higher priority vary. In this way nodes with the same variations in determinant bits with higher priority keep close inside their group and patterns and their substructure are easily recognisable.

2.3 Pattern examples

Patterns classifiable by determinant bits have been described in various previous works, such as in [7], [6, ch. 4, 6], [8, ch. 3] and, earlier, in [4]. Here, the general structure of those patterns is shortly described and shown in the example of $d_M = 4$. Moreover, examples for patterns are given using the arrangement introduced in the previous section.

2.3.1 Structure of patterns

Apart from their allocation to groups defined by determinant bits, nodes have been classified according to their role in the dynamics of a pattern. The following kinds of sets are further on referenced by *structure sets*.

- **Cluster**
  Set of nodes that is highly occupied by allowing nodes in it to exceed the lower threshold t_l by links into the same set. It does not appear in purely dynamic patterns (e. g. $d_M = d-1$) that only have influx-driven occupation.

- **Singletons**
  Set of nodes with neighbours in other less occupied sets (hubs, holes or periphery). They depend on the influx to exceed the lower threshold t_l and, in general, the larger the influx parameter p the higher their occupation.

- **Hubs**
  Set of nodes that connect clusters and holes. They have neighbours in both of them, but are less occupied due to their many neighbours in the highly occupied cluster and their consequent neighbour occupation slightly above the upper threshold t_u.

- **Holes**
  Set of nodes that are totally suppressed by their highly occupied neighbours in the clusters or core, i. e. their number of occupied neighbours highly exceeds the upper threshold t_u.

- **Periphery**
  Set of nodes that have neighbours in the holes and the higher occupied core, but not enough to maintain a high occupation.

- **Core**

  Set of nodes that are self-linked like the clusters, but so strongly (exceeding t_u) that they depend on the influx to become occupied. Such sets appear in purely dynamic patterns only.

In many cases of patterns the groups defined by determinant bits perfectly fit into those structure sets. An example for $d_M = 4$ is shown in figure 2.5. Some groups, like S_0, are structure sets themselves, other structure sets are made up of several adjacent groups, such as the holes in the example are found in group S_4 and S_5.

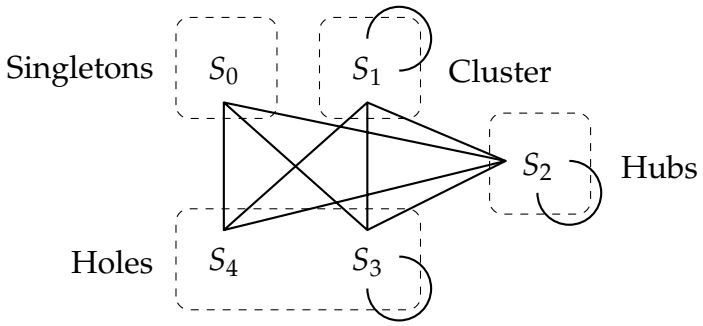


Figure 2.5: "Raupe" of the $d_M = 4$ pattern ("caterpillar", as in [6, ch. 6]). Solid lines mark linkage between groups, dashed lines indicate sets of groups.

Visualising the mean occupation of nodes in such a pattern as well reflects that structure. Figure 2.6 shows a snapshot of a $d_M = 4$ pattern on the left. The mean occupations of the nodes, arranged in the same way as on the left, are shown on right. A purely black grid point stands for a mean occupation of 1, a purely white one represents a mean occupation of 0. Intermediate mean occupations are visualized by grey grid points accordingly: the darker the grid point the higher the mean occupation. The red lines are borders between the groups. Easily the almost certain occupation of S_1 and the high occupation of S_0 can be recognised.

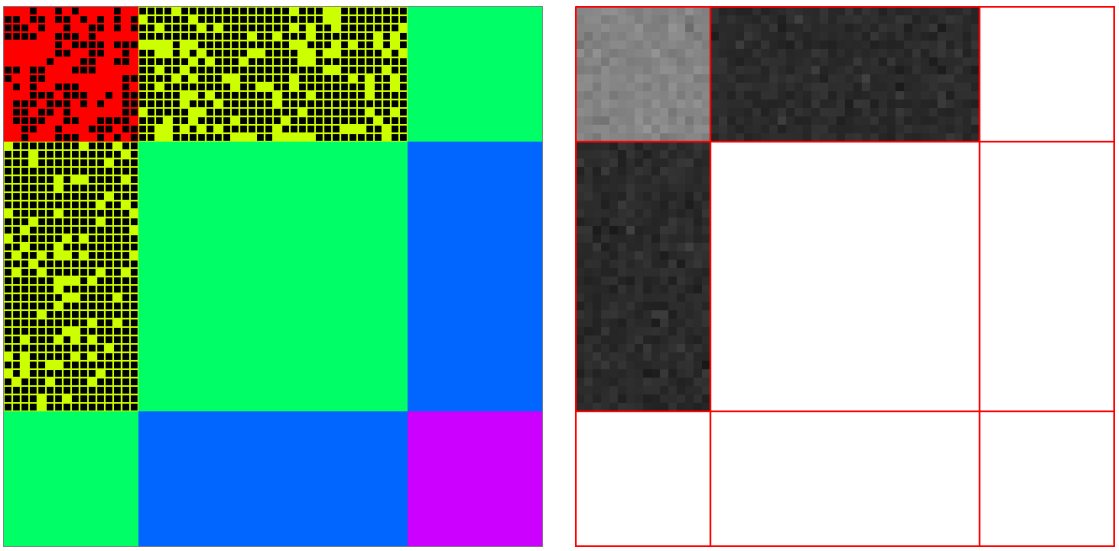
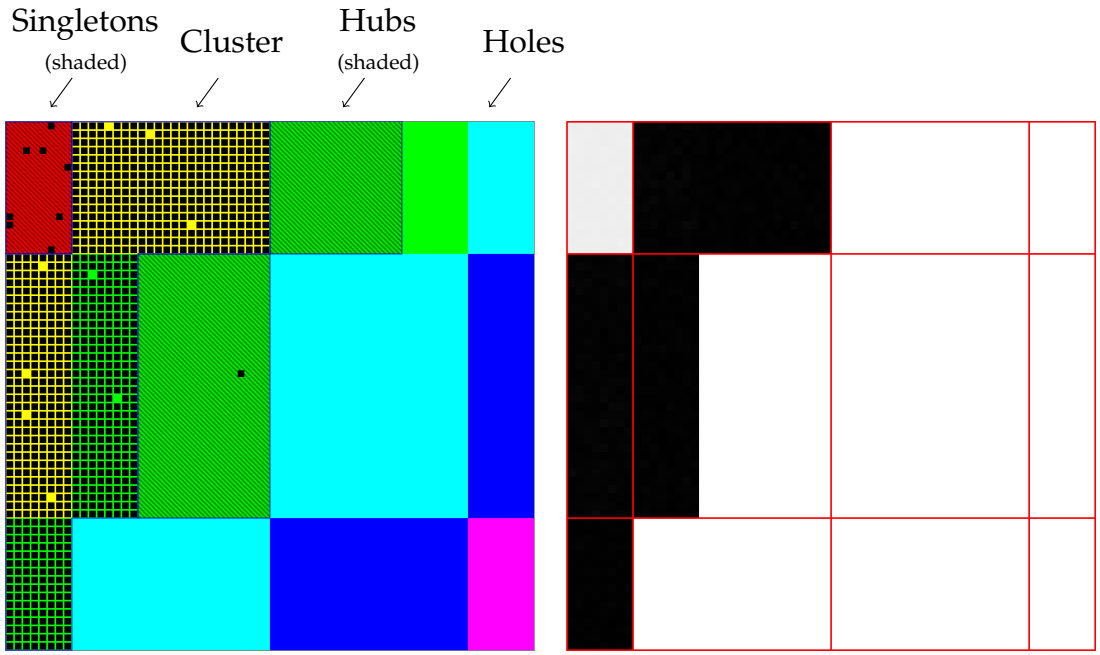
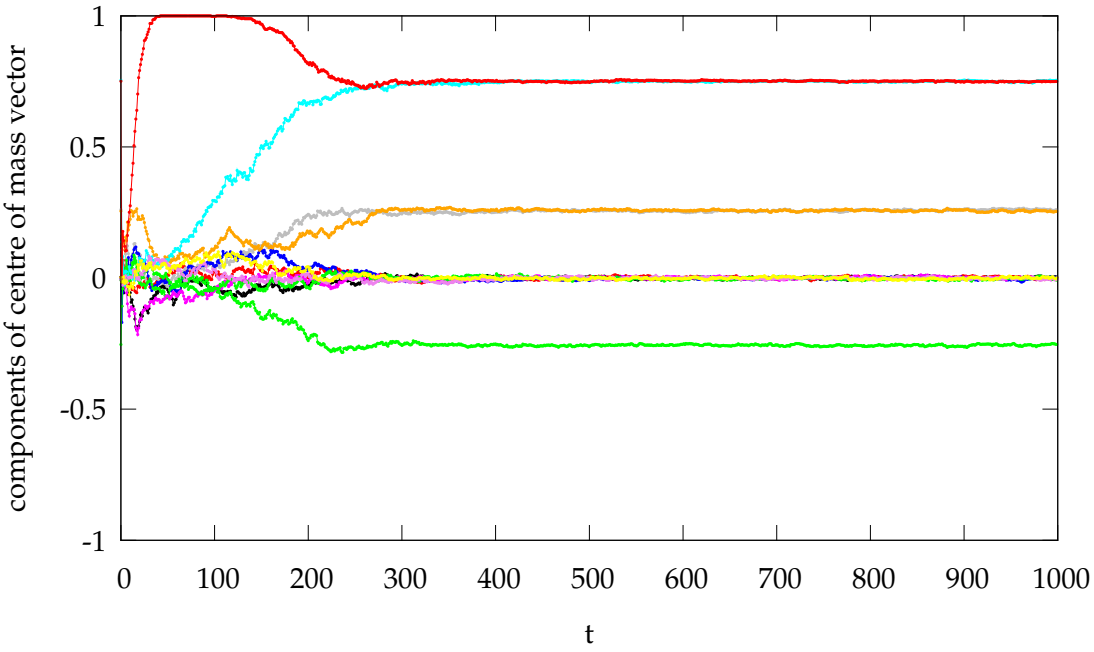


Figure 2.6: Dynamic $d_M = 4$ pattern with $p = 0.04$, $S \approx 0.226$.
[$d = 12$, $m = 2$, $[t_l; t_u] = [1; 10]$, started from empty network, after 10^3 iterations means taken over 10^4 iterations]

But then some patterns appear whose structure sets of nodes do not fit into groups according to determinant bits. For instance, consider the pattern shown in figure 2.7. Its centre of mass vector (see figure 2.7(b)) indicates 5 non-simple determinant bits, it is a $\mathbf{d}_M = (2, 3)$ pattern with two primary and three secondary determinant bits. Though some groups, such as S_0, S_3, S_4, S_5, only contain nodes of a single structure set, other groups seem to "fill up" with nodes of different structure sets. The structure sets however still align along the top-right to bottom-left diagonal. Since they furthermore consist of blocks of size of S_0, such a block grouping might be preferable, as will be discussed in the next chapter. Observations show that patterns with simple determinant bits fit into groups according to determinant bits, while patterns with non-simple determinant bits do not.



(a) Structure of the pattern. $S \approx 0.030$. Started from empty network, after 10^3 iterations means taken over $5 \cdot 10^4$ iterations.



(b) Components of centre of mass vectors. Starting from an empty network, the pattern is reached after about 300 iterations. The two primary determinant bits have a centre of mass component value of about 0.75, two secondary have 0.25 and another secondary determinant bit has -0.25. Thus, $\mathbf{d}_M = (2,3)$.

Figure 2.7: Structure and centre of mass diagram of a $d_M = 5$, $\mathbf{d}_M = (2,3)$ pattern. $[d = 12,\ m = 2,\ p = 0.02,\ [t_l; t_u] = [1; 10]]$

2.3.2 Simple dynamic patterns

Looking back at the entropy histogram (figure 2.1 on p. 14) for the parameter setting $d=12$, $m=2$, $[t_l;t_u]=[1;10]$ for various p, which was mostly used in the cited works, there are two main areas for different kinds of patterns emerging when starting from an empty network.

For $p < p_{min,11} \approx 0.026$ the very stable $d_M = 4$ (as in figure 2.6) and the less stable $d_M = 6$ patterns dominate. The $d_M = 6$ patterns appear in two forms, with simple (marked with "(a)") and with non-simple (marked with "(b)") determinant bits, shown in figures 2.8 and 2.9 respectively. Both appear very stable, e. g. for $p=0.02$, but have been observed to undergo transition to the more stable $d_M=4$ patterns after a long time.

For $p > p_{min,11} \approx 0.026$ patterns with $d_M = 11$ appear and $d_M = 4$ become unstable for larger p (they undergo transition to $d_M = 11$ patterns). Whereas the $d_M=4$ and $d_M=6$ patterns have a set of patterns that keeps highly occupied even for an influx $p=0$ (the cluster that keeps its high occupation by linkage into itself), the $d_M = 11$ patterns are purely dynamic and cannot withstand lesser than a least influx. For $p=0.06$ they are the most static (Shannon entropy has a minimum), as shown in figure 2.11. For larger and smaller values of p they are more dynamic; an example is shown for $p=0.03$ in figure 2.10.

Especially the $d_M = 11$ patterns have been subject of investigation in the previous works. The details are not restated here. Symmetry analysis in section 4.2.2 will give a hint to why those $d_M=d-1=11$ patterns arise, but $d_M=d=12$ patterns do not.

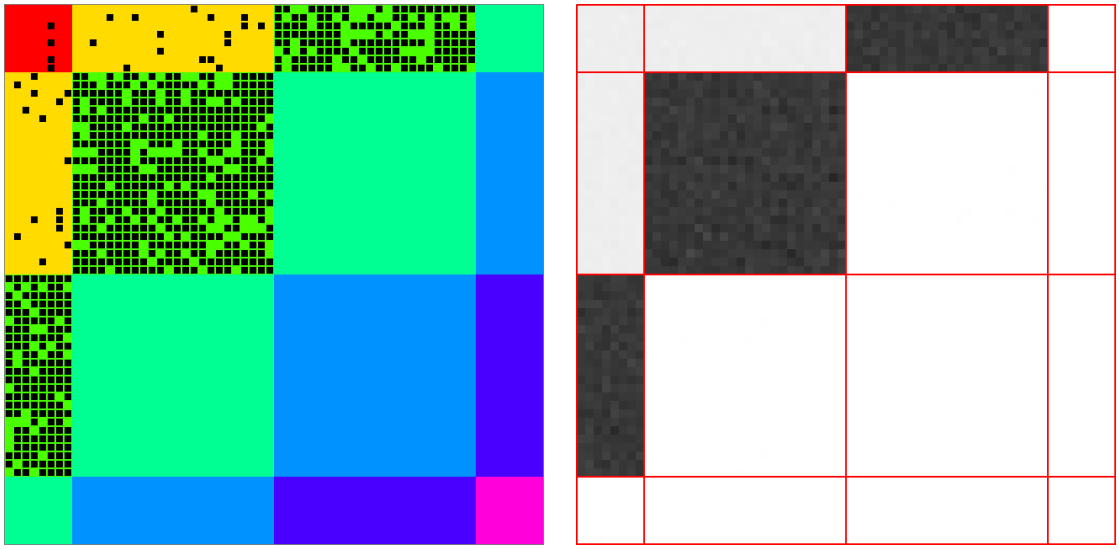


Figure 2.8: Dynamic $d_M = 6$ pattern with $p = 0.02$, $S \approx 0.217$. Patterns of this kind belong to the branch marked with "$d_M = 6$; (a)" in figure 2.1 (p. 14).
[$d = 12$, $m = 2$, $[t_l; t_u] = [1; 10]$, started from empty network, after 10^3 iterations means taken over $5 \cdot 10^4$ iterations]

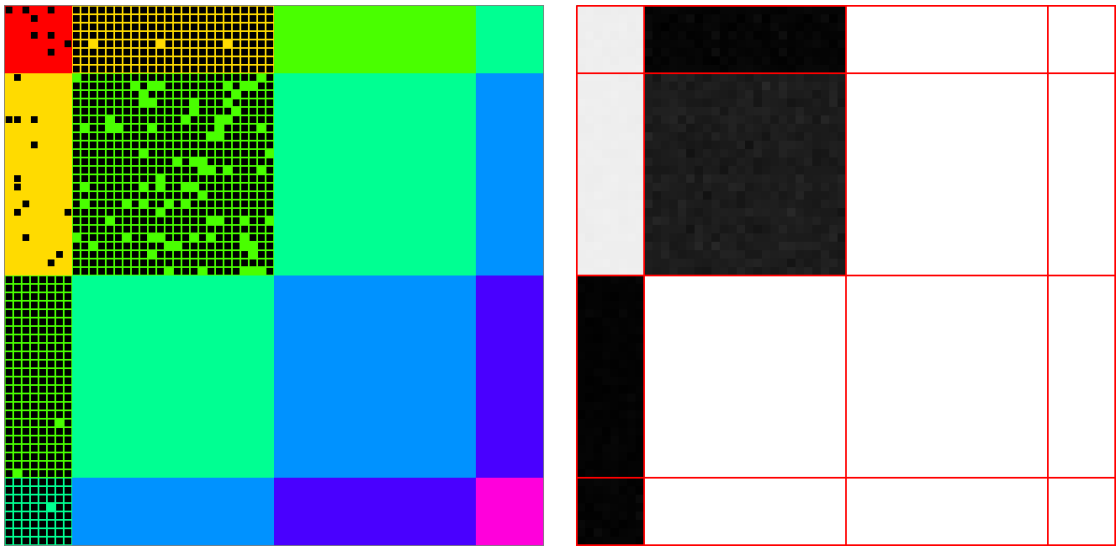


Figure 2.9: Dynamic $d_M = 6$, $\mathbf{d}_M = (3, 3)$ pattern with $p = 0.02$, $S \approx 0.107$. Patterns of this kind belong to the branch marked with "$d_M = 6$; (b)" in figure 2.1 (p. 14).
[$d = 12$, $m = 2$, $[t_l; t_u] = [1; 10]$, started from empty network, after 10^3 iterations means taken over $5 \cdot 10^4$ iterations]

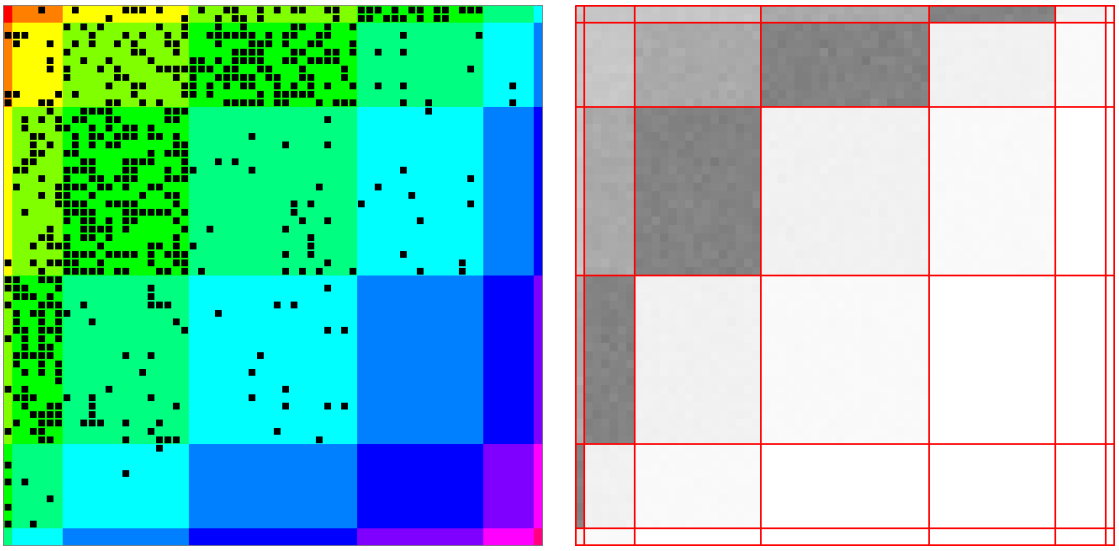


Figure 2.10: Dynamic $d_M = 11$ pattern with $p = 0.03$, $S \approx 0.363$.
$[d = 12, m = 2, [t_l; t_u] = [1; 10]$, started from empty network, after 10^3 iterations means taken over $5 \cdot 10^4$ iterations]

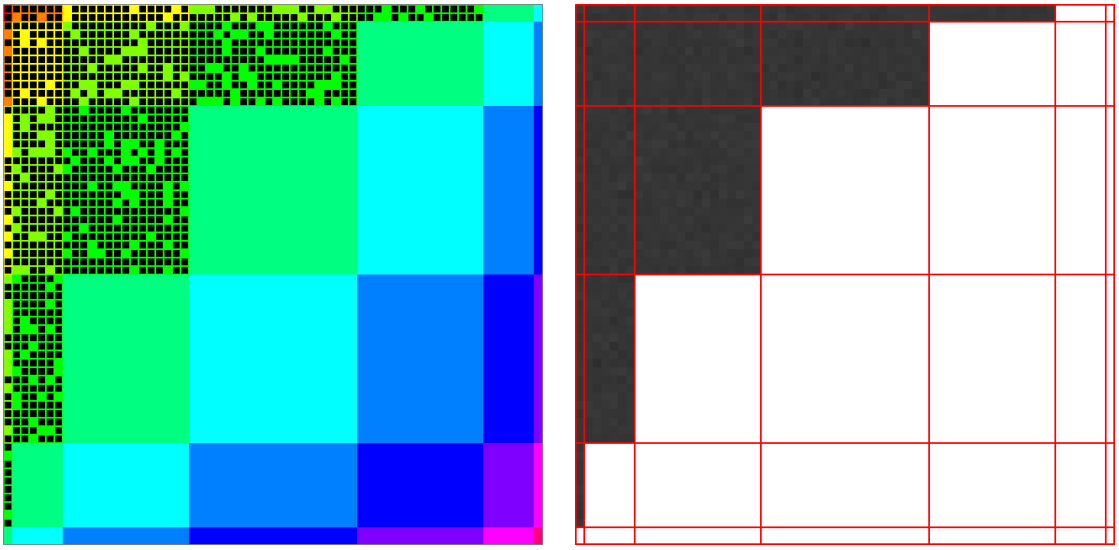
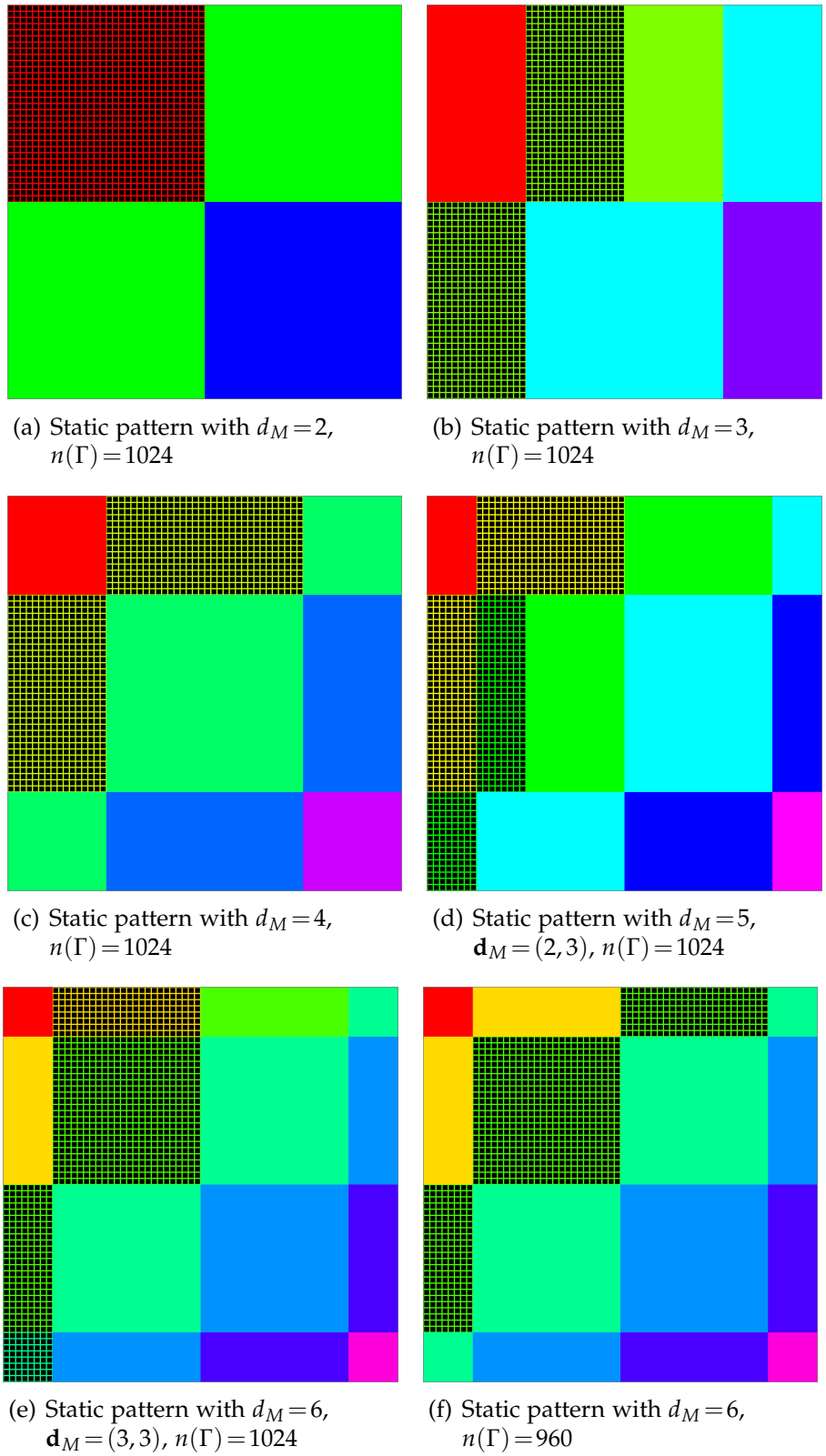


Figure 2.11: Dynamic $d_M = 11$ pattern with $p = 0.06$, $S \approx 0.204$.
$[d = 12, m = 2, [t_l; t_u] = [1; 10]$, started from empty network, after 10^3 iterations means taken over $5 \cdot 10^4$ iterations]

2.3.3 Simple static patterns

The appearance of certain patterns in simulations, of course, depends on the starting point of those simulations. To choose an unbiased starting point, simulations here were mostly run from a totally unoccupied ("empty") network. After the first influx it is occupied following a Bernoulli distribution with probability parameter p. In that way some patterns might never appear, though they are actually possible. A pattern is possible, if it survives the update and its window rule $[t_l;\ t_u]$ and does not have unoccupied nodes with an amount of occupied neighbours inside the window. Patterns that are not purely dynamic thus can be constructed and tested with omitted influx, $p = 0$. Obviously for purely static patterns with $p = 0$ the Shannon entropy is $S = 0$ (every node has mean occupation either 0 or 1). Since some patterns are stable only for $p = 0$, the Shannon entropy does not give a direct hint to pattern stability. Entropy furthermore is a subjective measure: one might think of an entropy that takes into account allocation into groups instead of consisting of a naive sum as the Shannon entropy does.

Following a list of patterns is presented (figures 2.12–2.16), which have been built using the recursion formula given in section 3.3.3. It definitely is an incomplete list (further examples are given in the next section), but shows that patterns with $d_M > 6$ are also possible. These are in fact too unstable to appear. The $d_M = 2$ pattern then again is very stable for an influx p not too large, but is not likely to be reached from an empty network. For each pattern the number of occupied nodes, $n(\Gamma)$ is given (there are $2^d = 4096$ nodes here in total). For every d_M there is a kind of such patterns with $n(\Gamma) = 1024$. For $d_M \in \{6, 7, 8, 9\}$ $n(\Gamma)$ can be less, even down to $n(\Gamma) = 952$ for $d_M = 9$. Different $n(\Gamma)$ also occurs with different types of non-simple determinant bits ($\mathbf{d}_M$ is given). Still the number of occupied nodes, $n(\Gamma)$, is an integer multiple of the size of S_0, $|S_0| = 2^{d-d_M}$.



(a) Static pattern with $d_M = 2$, $n(\Gamma) = 1024$

(b) Static pattern with $d_M = 3$, $n(\Gamma) = 1024$

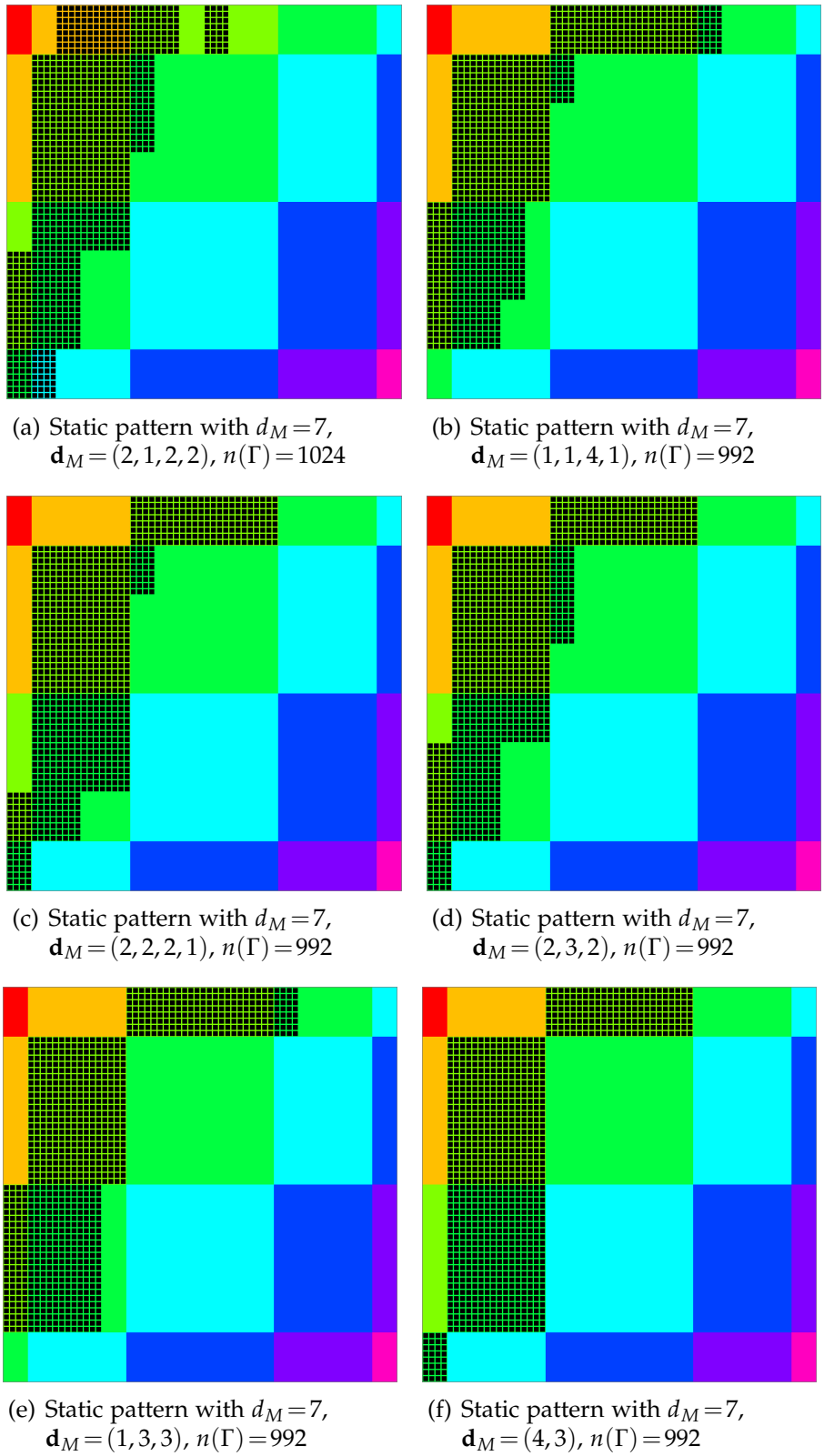
(c) Static pattern with $d_M = 4$, $n(\Gamma) = 1024$

(d) Static pattern with $d_M = 5$, $\mathbf{d}_M = (2, 3)$, $n(\Gamma) = 1024$

(e) Static pattern with $d_M = 6$, $\mathbf{d}_M = (3, 3)$, $n(\Gamma) = 1024$

(f) Static pattern with $d_M = 6$, $n(\Gamma) = 960$

Figure 2.12: Static $d_M \in \{2, \ldots, 6\}$ patterns, $p = 0$. $[d = 12, m = 2, [t_l; t_u] = [1; 10]]$



(a) Static pattern with $d_M = 7$, $\mathbf{d}_M = (2,1,2,2)$, $n(\Gamma) = 1024$

(b) Static pattern with $d_M = 7$, $\mathbf{d}_M = (1,1,4,1)$, $n(\Gamma) = 992$

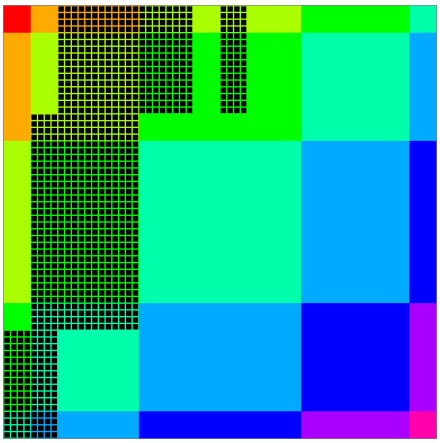
(c) Static pattern with $d_M = 7$, $\mathbf{d}_M = (2,2,2,1)$, $n(\Gamma) = 992$

(d) Static pattern with $d_M = 7$, $\mathbf{d}_M = (2,3,2)$, $n(\Gamma) = 992$

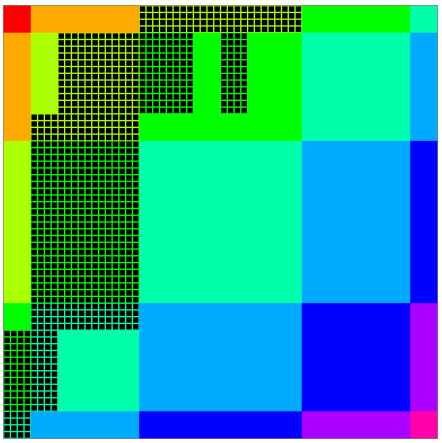
(e) Static pattern with $d_M = 7$, $\mathbf{d}_M = (1,3,3)$, $n(\Gamma) = 992$

(f) Static pattern with $d_M = 7$, $\mathbf{d}_M = (4,3)$, $n(\Gamma) = 992$

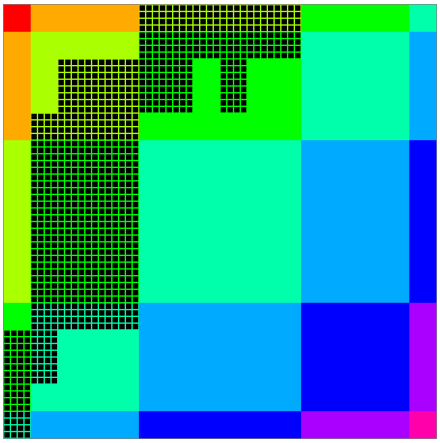
Figure 2.13: Static $d_M = 7$ patterns, $p = 0$.
$[d = 12, m = 2, [t_l; t_u] = [1; 10]$



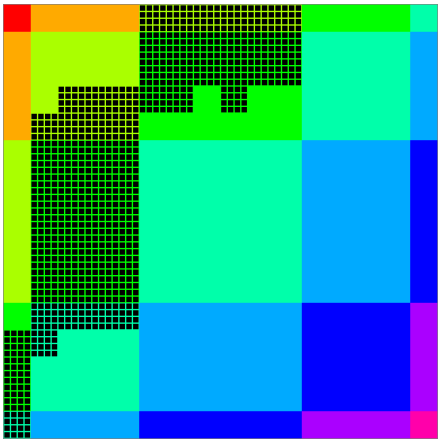
(a) Static pattern with $d_M = 8$, $\mathbf{d}_M = (3,2,3)$, $n(\Gamma) = 1024$



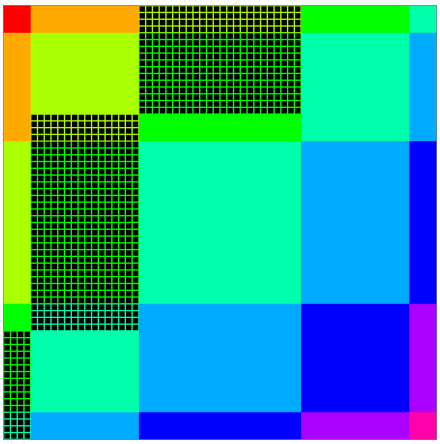
(b) Static pattern with $d_M = 8$, $\mathbf{d}_M = (3,2,3)$, $n(\Gamma) = 1008$



(c) Static pattern with $d_M = 8$, $\mathbf{d}_M = (3,2,2,1)$, $n(\Gamma) = 992$

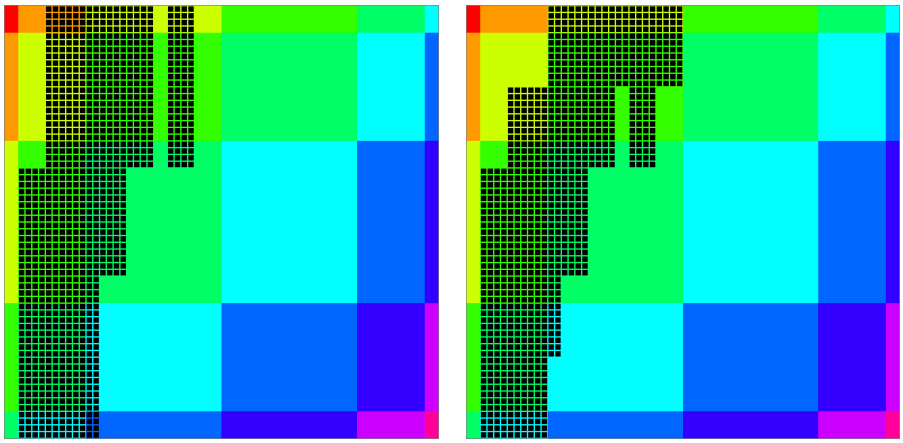


(d) Static pattern with $d_M = 8$, $\mathbf{d}_M = (3,2,1,2)$, $n(\Gamma) = 976$



(e) Static pattern with $d_M = 8$, $\mathbf{d}_M = (5,3)$, $n(\Gamma) = 960$

Figure 2.14: Static $d_M = 8$ patterns, $p = 0$. $[d = 12,\ m = 2,\ [t_l; t_u] = [1; 10]$



(a) Static pattern with $d_M = 9$, $\mathbf{d}_M = (2,1,2,2,2)$, $n(\Gamma) = 1024$

(b) Static pattern with $d_M = 9$, $\mathbf{d}_M = (2,1,2,2,2)$, $n(\Gamma) = 1024$

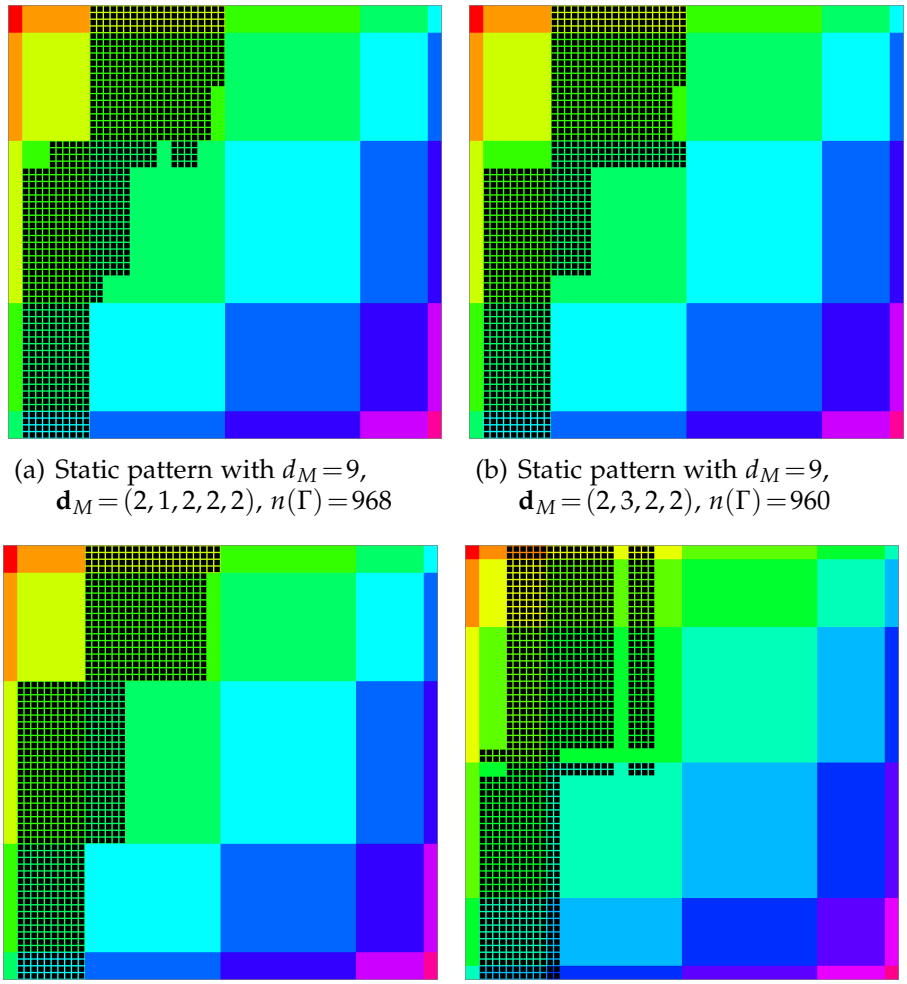
(c) Static pattern with $d_M = 9$, $\mathbf{d}_M = (2,1,2,2,2)$, $n(\Gamma) = 1016$

(d) Static pattern with $d_M = 9$, $\mathbf{d}_M = (2,1,2,2,2)$, $n(\Gamma) = 1008$

(e) Static pattern with $d_M = 9$, $\mathbf{d}_M = (2,1,2,2,2)$, $n(\Gamma) = 1000$

(f) Static pattern with $d_M = 9$, $\mathbf{d}_M = (2,1,2,2,2)$, $n(\Gamma) = 984$

Figure 2.15: Static $d_M = 9$ patterns, $p = 0$.
$[d = 12, m = 2, [t_l; t_u] = [1; 10]$



(a) Static pattern with $d_M = 9$, $\mathbf{d}_M = (2,1,2,2,2)$, $n(\Gamma) = 968$

(b) Static pattern with $d_M = 9$, $\mathbf{d}_M = (2,3,2,2)$, $n(\Gamma) = 960$

(c) Static pattern with $d_M = 9$, $\mathbf{d}_M = (2,3,4)$, $n(\Gamma) = 952$

(d) Static pattern with $d_M = 10$, $\mathbf{d}_M = (3,2,2,3)$, $n(\Gamma) = 1024$

Figure 2.16: Static $d_M = 9$ and $d_M = 10$ patterns, $p = 0$. $[d = 12,\ m = 2,\ [t_l; t_u] = [1; 10]]$

2.3.4 More general patterns

The insufficiency of the concept of simple determinant bits to classify all patterns has been made clear in the previous sections. Varying parameters or applying certain symmetry operations (see section 4.2.3) yields further patterns that cannot be classified satisfactorily by non-simple determinant bits either. Examples for these are shown in following figures 2.17–2.20. "Chess board" subpatterns emerge, whose classification is topic of the next chapter. For $d=12$ these had to be prepared, only once a chess board pattern has been observed to emerge after starting from an empty network. For $d=11$ and $d=13$, on the other hand, they appear very often (a hint for a reason is given in section 4.2 using symmetry analysis).

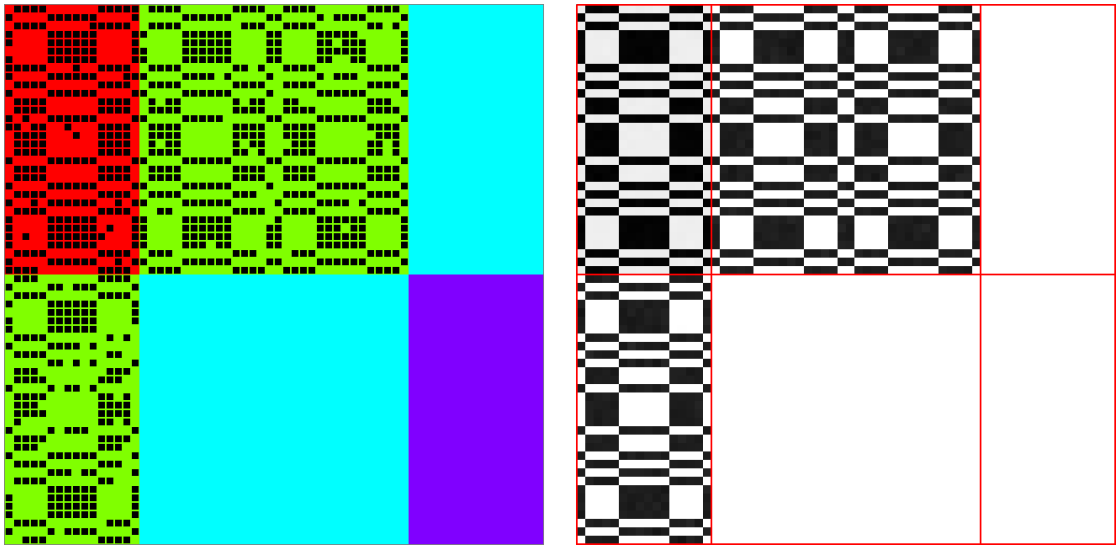


Figure 2.17: Dynamic $d_M=3$ pattern with $p=0.02$, $S\approx 0.123$.
$[d=12,\ m=2,\ [t_l;t_u]=[1;10]$, started from prepared pattern, means taken over $5\cdot 10^4$ iterations]

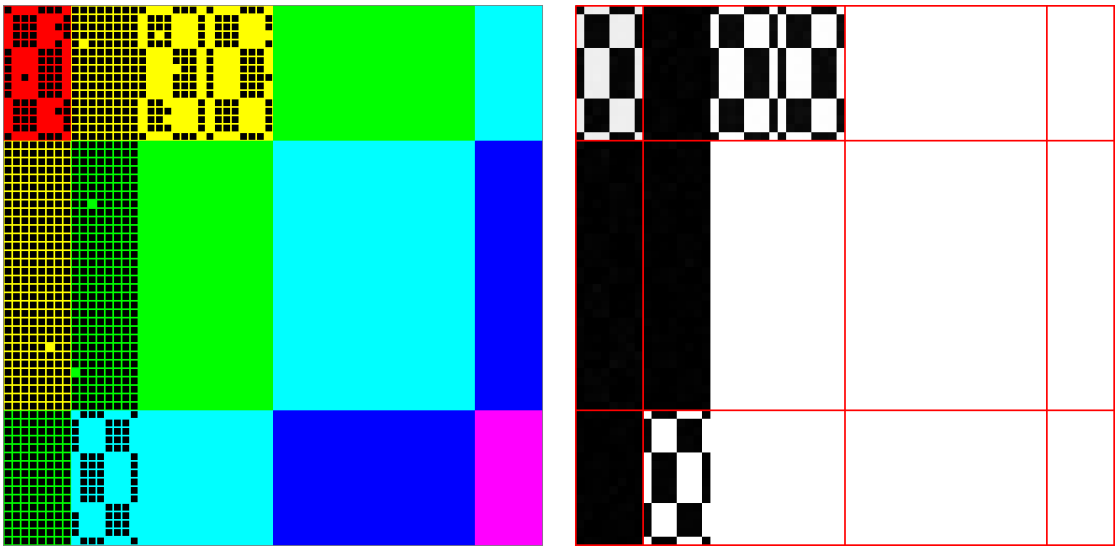


Figure 2.18: Dynamic $d_M = 5$ pattern with $p = 0.02$, $S \approx 0.027$. This pattern appears extremely rarely.
[$d = 12$, $m = 2$, $[t_l; t_u] = [1; 10]$, started from empty network, after 10^3 iterations means taken over $6 \cdot 10^4$ iterations]

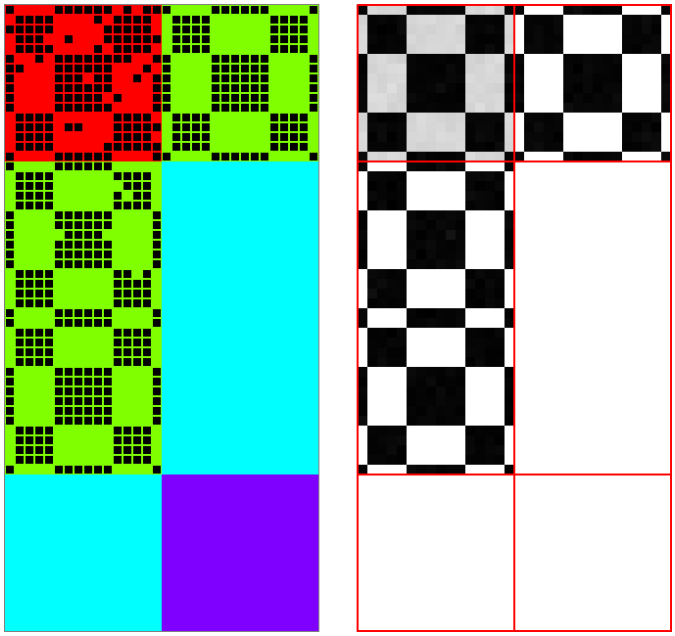


Figure 2.19: Dynamic $d_M = 3$ pattern with $p = 0.03$, $S \approx 0.075$.
[$d = 11$, $m = 2$, $[t_l; t_u] = [1; 10]$, started from empty network, after 2000 iterations means taken over 10^4 iterations]

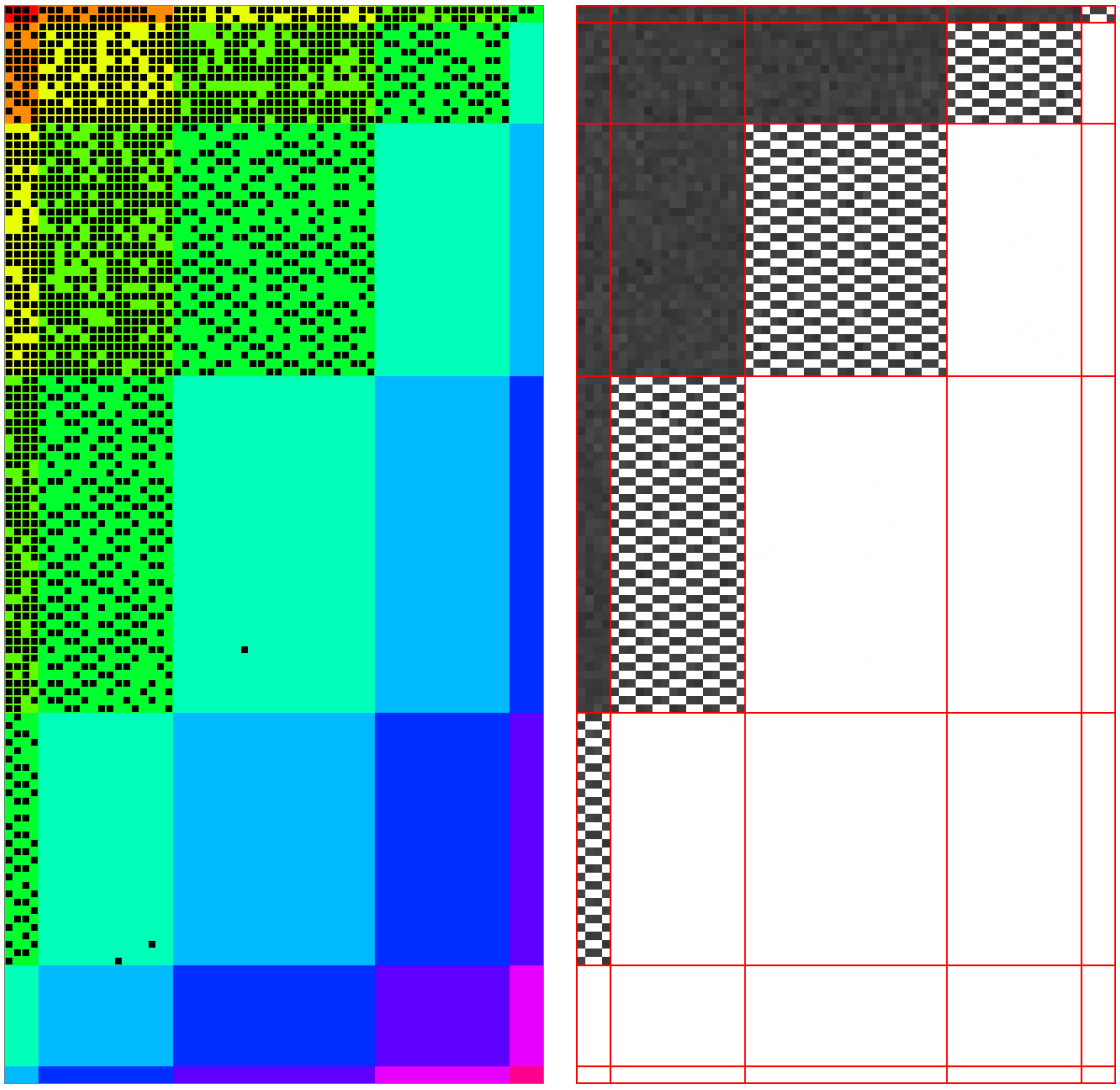


Figure 2.20: Dynamic $d_M = 10$ pattern with $p = 0.05$, $S \approx 0.221$. [$d = 13$, $m = 2$, $[t_l; t_u] = [1; 10]$, started from empty network, after 10^3 iterations means taken over 10^4 iterations]

2.4 Pattern stability

One of the central questions about the dynamics of the model is the possibility of patterns and especially their stability. There are different aspects regarding stability of patterns:

- **Stability under the influx**

  Once the network has developed a pattern it is liable to perturbations done by the influx. After experiencing "unfortunate" occupation events from the influx a pattern might dissolve and the network will change to a different pattern. The most perturbing effect of the influx happens on nodes with a mean neighbour occupation slightly below the lower or upper threshold. As the influx might occupy more neighbours of a node with mean neighbour occupation slightly below the lower threshold t_l, that neighbour occupation might exceed t_l and increase the occupation of the node itself. For nodes with mean neighbour occupation slightly below t_u, the influx might lead to a transgression of that upper threshold and yield in the node losing occupation. An example of how to analyse such a pattern transition is given in the next subsection.

  More important for the stability of a pattern than the mean occupation of single nodes is the mean occupation of groups of nodes according to the network decomposition the pattern fits into. Thus, the smaller the necessary size of the groups of such a decomposition the larger the perturbation of the influx on the mean occupation (over space and time) of that group. The instrument of the block-link matrix (see section 3.3.1) could prove useful for studying that sensitivity.

- **Degeneracy of a kind of pattern**

  In the state space of the system the stability of a pattern under the influx corresponds to the probability of staying in the macrostate that is the pattern. On the other hand the degeneracy of the macrostate of a pattern (i. e. the number of microstates which give that macrostate) and the amount of macrostates of patterns of that kind correspond to the probability of the network reaching such kind of a pattern when starting from a random point in state space.

  It is important to mention that the basin of a macrostate does not necessarily directly relate to its size. That means, though the size of an

macrostate in state space might be large, the probability to reach it from outside might be very small. For instance, as shown in section 4.2.1, for the setting $d = 12$, $m = 2$ the macrostate of the $d_M = 2$ pattern is much larger than that of the $d_M = 4$ pattern, but it appears much less often when starting simulations from an empty network.

- **Stability concerning topological variations**
  Of course, the possibility of the system to establish certain kinds of patterns not only depends on its parameters but also strongly on the topology of the underlying network. It is possible that some kinds of patterns simply are a relict of the simplicity of the model with its high symmetry. Fortunately the behaviour of the system proves comparably stable to several topological variations as done in chapter 5: many patterns still appear.

- **Stability under directed attacks**
  The influx is a special form of random attacks on the network: the state is perturbed by random unoccupied nodes becoming occupied by the influx. On the other hand, if nodes are selectively chosen and changed in their state, the attack is directed. The stability of a pattern under such attack depends on their nature: where the attacks takes place (i. e. which nodes are chosen), how they are changed and in which timely sequence. One might, for example, think that a clever choice of just a few nodes and a clever sequence of perturbing their occupation might lead to a sure transition to a different pattern.

  In this work this form of stability of patterns is studied for a very simple form of directed attack, namely an intruding antigen (see chapter 6). Such an antigen is modelled by a chosen node in the network, that is held occupied over a time interval. Depending on the position of the antigen it induces establishment of a subpattern.

2.4.1 Example with determinant bit pattern $d_M = 4$

In the parameter setting $d = 12$, $m = 2$, $p = 0.055$, $[t_l; t_u] = [1; 10]$ the $d_M = 4$ pattern (like in figure 2.6) becomes quite unstable and transitions in most cases after a few thousand time steps to the dynamic $d_M = 11$ pattern (like in figure 2.10). Preparing some of these $d_M = 4$ patterns and keeping track of the number left while iterating, as shown in figure 2.21, shows that the transition follows a

Bernoulli process. For every time step there is the probability ρ of the $d_M = 4$ pattern to dissolve into a different pattern and the probability of the pattern dissolving after exactly $t + 1$ time steps is geometrically distributed:

$$f(t) = \rho(1-\rho)^t \tag{2.4}$$

(for t time steps no transition with probability $1 - \rho$ and the transition with probability ρ in the $t+1$-th time step). It must be noted that such a pattern transition does not happen in a single time step. Actually there are transition states that can not be directly assigned to a particular pattern. But on the long scale the transition probability follows that distribution.

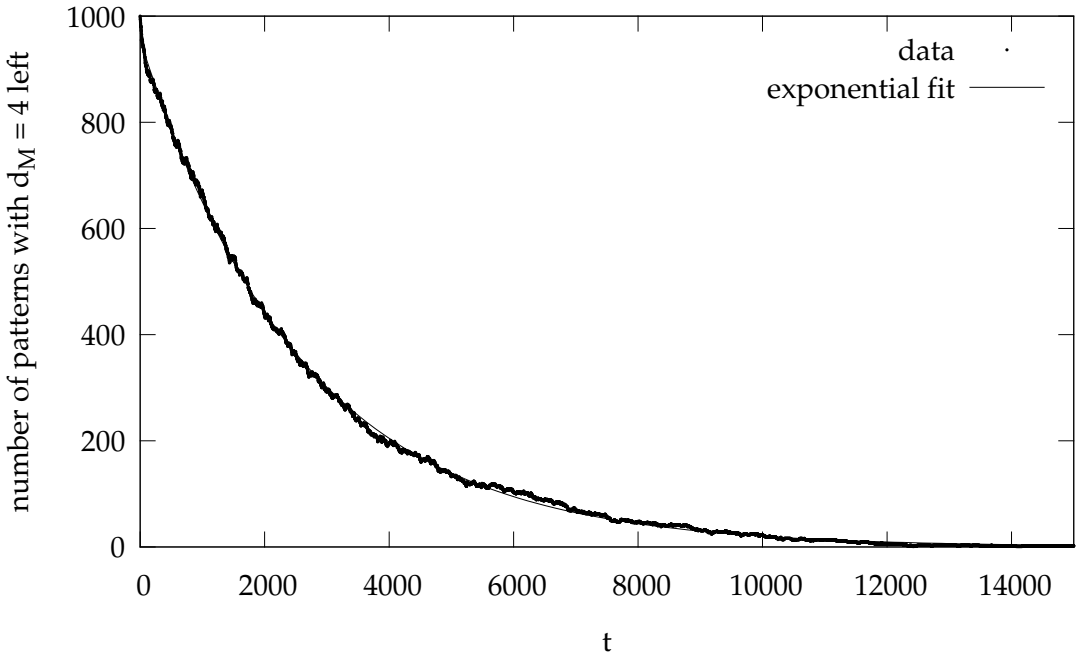


Figure 2.21: Number of $d_M = 4$ patterns left after preparing 1000 of those patterns and iterating.
$[d = 12,\ m = 2,\ p = 0.055,\ [t_l; t_u] = [1; 10]]$

Fitting the data in figure 2.21 by an exponential $A \cdot e^{\lambda t}$ yields $A = 954.8 \pm 0.3$ and the transition probability in one time step of

$$\rho = 1 - e^{\lambda} = 3.851 \cdot 10^{-4} \pm 2 \cdot 10^{-7}. \tag{2.5}$$

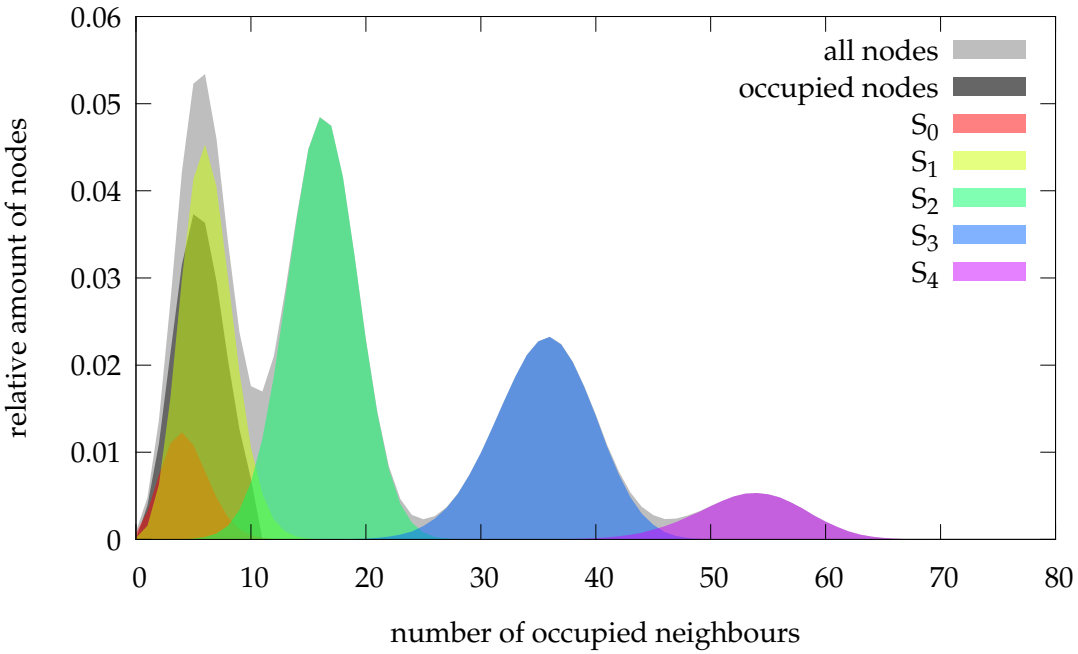
A good tool for observing a pattern transition is a network topology plot. The relative amount of nodes is plotted versus the number of occupied neigh-

bours which these nodes have after the influx and before applying the update rule. The plots show averages over some time steps (see captions). Apart from the plot for the whole network (grey areas) topology distributions of groups have been marked by colour according to the group. Figure 2.22 shows four of these, before, during and after a transition from a $d_M = 4$ pattern to a $d_M = 11$ pattern.

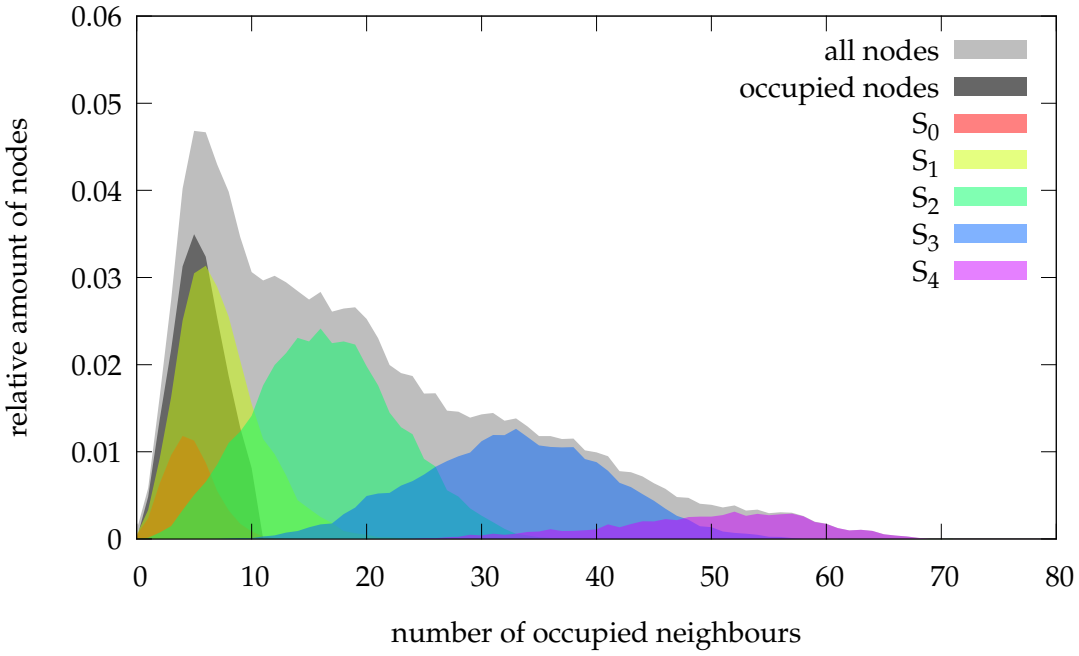
The sharp peaks that one might expect from the group allocation (nodes of a group all have equal number of occupied neighbours in a perfect static pattern) have been widened by the non-zero influx. As seen in figure 2.22(a), in the $d_M = 4$ pattern this widening leads the topology distribution of the group S_1 to "leak out" of the $[t_l; t_u] = [1; 10]$ window: some of the otherwise fully occupied nodes in group S_1 have neighbour occupation above t_u and become unoccupied. This in turn yields the topology distribution of the group S_2 (which is suppressed by its neighbours in S_1) to leak into the threshold window. Eventually some of the nodes in S_2 might become occupied and themselves suppress their neighbours in S_1; if there are enough of them over a sufficient time, the pattern changes.

During transition (figure 2.22(b)) the topology distributions of groups split up and parts move. For instance, parts of the topology distribution of S_1 move up and out of the threshold window (the nodes become unoccupied), parts of the topology distribution of S_2 move into the window.

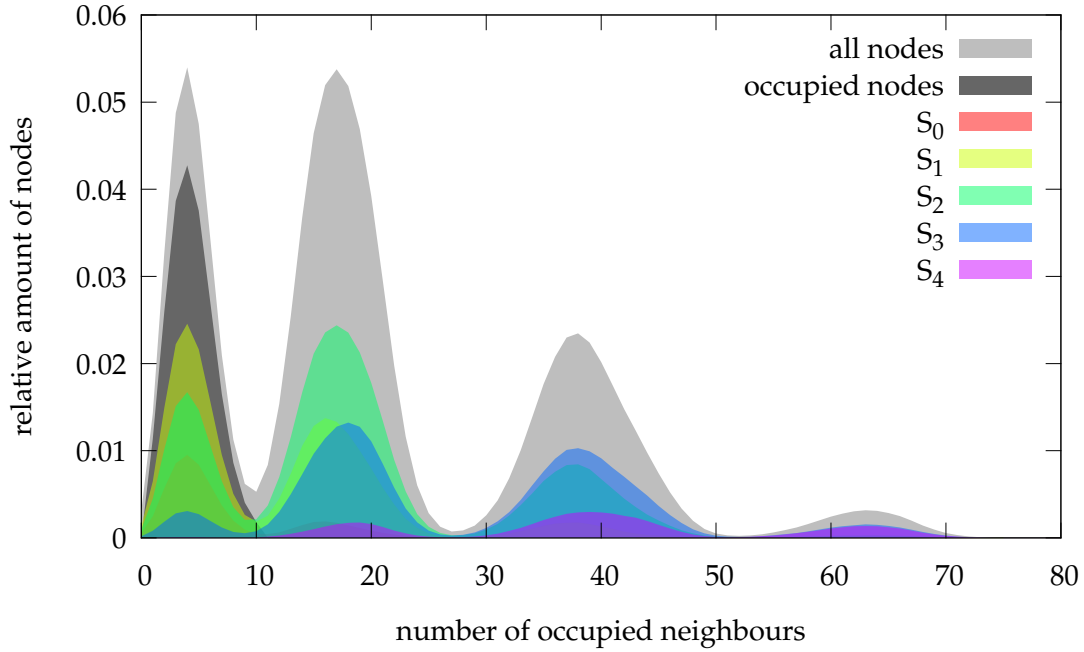
After transition to the $d_M = 11$ pattern topology distributions form new peaks (figure 2.22(c)) which can be coloured again according to their affiliation to $d_M = 11$ pattern groups (figure 2.22(d)).



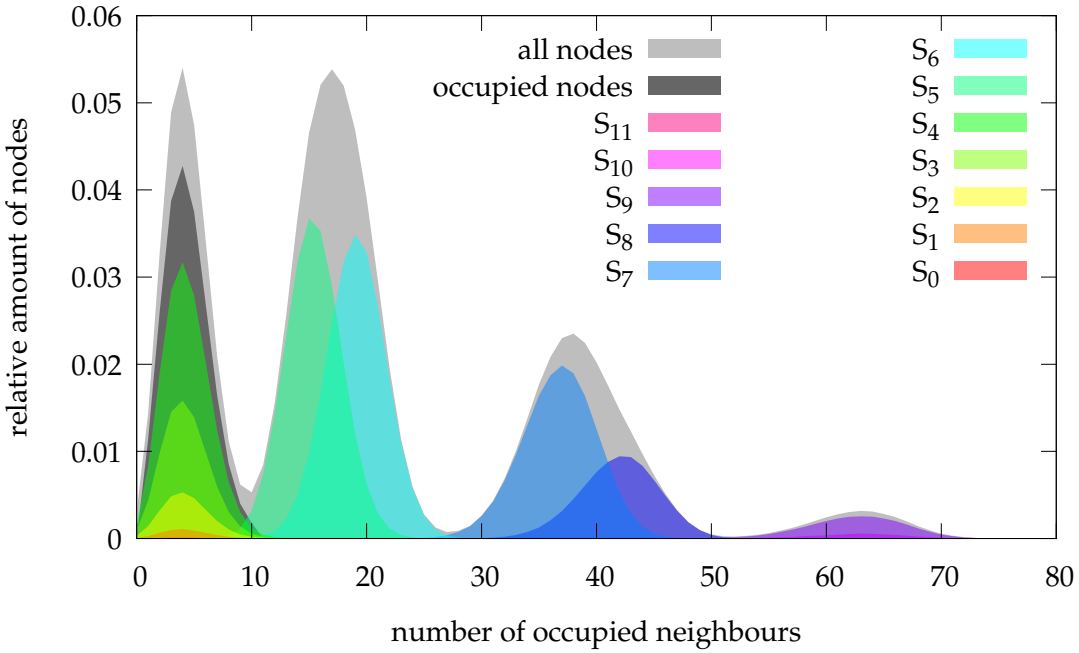
(a) $d_M = 4$ pattern before transition (taken over 1500 iterations).



(b) $d_M = 4$ pattern in the middle of transition (taken over 15 iterations).



(c) $d_M = 11$ pattern after transition (showing former groups of the $d_M = 4$ pattern, taken over 1500 iterations).



(d) $d_M = 11$ pattern after transition (taken over 1500 iterations).

Figure 2.22: Network topology diagrams before, during and after the transition from a $d_M = 4$ pattern to a $d_M = 11$ pattern. Grey areas represent the whole network, groups are coloured differently. [$d = 12$, $m = 2$, $p = 0.055$, $[t_l; t_u] = [1; 10]$]

Decompositions and tilings

3

Understanding the dynamics of the network requires a good understanding of its patterns. Thus, it is desirable to be able to classify patterns and to determine which are possible and at what stability and which are not. Since patterns have a regular structure on the underlying graph, a way to handle patterns is to group together nodes with equal statistical properties (e. g. mean occupation), i. e. to decompose the network into these groups as it has been done using determinant bits (see section 2.1). With general decompositions patterns could be classified based on the decompositions they fit in. Furthermore decompositions could prove useful for renormalisation approaches.

For a decomposition into groups of nodes that have the same statistical properties, the decomposition should be neighbourhood compatible. This means that two given groups of such a decomposition independent of the node chosen in the first group shall have the same number of neighbours in the other group. Thereby ensuring that all nodes of a group are treated equally by the update rule and thus have the same statistical properties (for a pattern that fits into that decomposition). In the following when speaking of *network decompositions* it is implied that they are neighbourhood compatible.

A more general decomposition than the one based on determinant bits is given in this chapter and is shown to help to construct patterns that rarely appear by just using simulations. Those decompositions, "tilings", are proven to have strong algebraic properties and to be easily computable.

3.1 Cayley graphs

The underlying graph of the model can be treated as a Cayley graph that allows an easy study of it by algebraic methods and thus allowing to prove some general statements about it.

Definition: As in [9], a *Cayley graph* $\Gamma(\mathcal{G}, \mathcal{S})$ is a graph whose nodes are elements of an (algebraic) group $\mathcal{G}$ (its group operation shall be "+"). The subset $\mathcal{S} \subseteq \mathcal{G}$ defines the edges of the graph: Linkage from a node $a \in \mathcal{G}$ to another node $b \in \mathcal{G}$ is defined to exist if and only if there exists $s \in \mathcal{S}$ such that $a + s = b$ (i. e. starting from node a its neighbours are reached by adding elements of $\mathcal{S}$).

Of course, if the group $\mathcal{G}$ is abelian (elements commute under "+") the graph is undirected (edges do not have a direction). If $\mathcal{S}$ generates $\mathcal{G}$ (i. e. every element in $\mathcal{G}$ can be expressed as a sum of elements in $\mathcal{S}$), the graph is complete, i. e. there is a path from any node to any other.

3.1.1 The underlying graph as a Cayley graph

The node set of the underlying graph of the model, $G_d^{(m)}$, $\mathcal{V} = \{0,1\}^d$, is indeed a group (actually it is even a vector space over the field $\mathbb{F}_2 = \{0,1\}$), the addition "+" is defined as the component-wise "exclusive or":

$$\begin{array}{rclcrcl} 0+0 & = & 0 & \qquad & 1+1 & = & 0 \\ 1+0 & = & 1 & & 0+1 & = & 1 \end{array}$$

The bit vectors in $\mathcal{V}$ are written $(a_1, \ldots, a_d)^T$ or $a_1 \ldots a_d$ with equal meaning. The edge-defining subset $\mathcal{S} \subseteq \mathcal{V}$ in this case is the set of nodes with at least $d - m$ bits set to 1:

$$\mathcal{S} = \{\mathbf{s} \in \mathcal{V};\, d_H(\mathbf{s}, \mathbf{0}) \geq d - m\}. \tag{3.1}$$

That is, in order to get a neighbour of a node $\mathbf{a} \in \mathcal{V}$ one has to add an element $\mathbf{s} \in \mathcal{S}$ thereby changing at least $d - m$ bits of $\mathbf{v}$. Since $\mathcal{V}$ is an abelian group the neighbourhood defining statement can be reformulated:

$$\begin{aligned} & \mathbf{a} \in \mathcal{V} \text{ and } \mathbf{b} \in \mathcal{V} \text{ are neighbours} \\ \Leftrightarrow\ & \exists\, \mathbf{s} \in \mathcal{S} :\ \mathbf{a} + \mathbf{s} = \mathbf{b} \\ \Leftrightarrow\ & \mathbf{a} + \mathbf{b} \in \mathcal{S}. \end{aligned} \tag{3.2}$$

In the following the *subgroup generated by* $\mathbf{a}_1, \ldots, \mathbf{a}_n \in \mathcal{V}$ is used, that is the subgroup of $\mathcal{V}$, whose elements can each be expressed as a sum of some of the generators $\mathbf{a}_1, \ldots, \mathbf{a}_n$. Here, every element $\mathbf{a} \in \mathcal{V}$ is self-inverse, that is $\mathbf{a} + \mathbf{a} = \mathbf{0}$. Furthermore, since all elements commutate, the subgroup generated by $\mathbf{a}_1, \ldots, \mathbf{a}_n \in \mathcal{V}$ is:

$$\langle \mathbf{a}_1, \ldots, \mathbf{a}_n \rangle := \{ \mathbf{a}_{i_1} + \ldots + \mathbf{a}_{i_j};\ 1 \leq i_1 < \ldots < i_j \leq n,\ 1 \leq j \leq n \}. \tag{3.3}$$

If $m > 0$, $\mathcal{S}$ generates the whole group $\mathcal{V}$, that is $\langle \mathcal{S} \rangle = \mathcal{V}$, thus the graph is complete. The reason is that, if $m > 0$, $11\ldots11 \in \mathcal{S}$ (corresponding to the perfect complement) and $011\ldots11,\ 101\ldots11,\ \ldots,\ 111\ldots10 \in \mathcal{S}$ (corresponding to the complements with one mismatch). Thus,

$$\begin{aligned} 011\ldots11 + 11\ldots11 &= 100\ldots00 \in \langle \mathcal{S} \rangle, \\ 101\ldots11 + 11\ldots11 &= 010\ldots00 \in \langle \mathcal{S} \rangle, \\ &\vdots \\ 111\ldots10 + 11\ldots11 &= 000\ldots01 \in \langle \mathcal{S} \rangle, \end{aligned} \tag{3.4}$$

and obviously every $\mathbf{a} \in \mathcal{V}$ can be expressed as sums of these.

3.2 Tilings

Let $\Gamma(\mathcal{V}, \mathcal{S})$ be the underlying graph of the model, $G_d^{(m)}$, with node set $\mathcal{V}$ regarded as a Cayley graph as described in the previous section 3.1. A possible decomposition is the decomposition into equally sized parts with neighbourhood distribution among them ensuring the decomposition to be neighbourhood compatible. Such a decomposition shall be called "network tiling" (in reference to tiling the network into tiles like a plain).

Definition: A *network tiling* $(\mathcal{A}_i)_{i\in I}$ of $\Gamma(\mathcal{V},\mathcal{S})$ is defined to be a disjunct decomposition of the network into blocks

$$\mathcal{A}_i \subseteq \mathcal{V}, \quad \mathcal{A}_i \cap \mathcal{A}_j = \varnothing \; \forall \, i \neq j, \; i,j \in I, \quad \mathcal{V} = \bigcup_{i\in I} \mathcal{A}_i, \tag{3.5}$$

satisfying:

1. The network is decomposed into equally sized parts, called "*tiles*" or "*blocks*"

$$|\mathcal{A}_i| = |\mathcal{A}_j| \quad \forall i,j \in I. \tag{3.6}$$

2. Neighbours of a particular type (i. e. for a particular $\mathbf{s} \in \mathcal{S}$) of a node in a block $\mathcal{A}_i$ belong to the same block as the neighbours of this type of any other node in $\mathcal{A}_i$. Put into a mathematical expression that is:

$$\begin{aligned} &\exists \, S_{ij} \subseteq \mathcal{S}, \; i,j \in I, \quad \text{with} \\ &\; S_i \cap S_j = \varnothing \; \forall \, i \neq j, \; i,j \in I, \quad \textstyle\bigcup_{j\in I} S_{ij} = \mathcal{S}, \quad \text{such that} \\ &\; \forall i,j \in I: \;\; \mathbf{a} + S_{ij} = \{\mathbf{a} + \mathbf{s};\; \mathbf{s} \in S_{ij}\} \subseteq \mathcal{A}_j \quad \forall \mathbf{a} \in \mathcal{A}_i. \end{aligned} \tag{3.7}$$

In a more general decomposition the parts do not have to be of equal size. For example, the group allocation according to determinant bits consists of groups of different size. Nevertheless for tilings the parts are claimed to be of equal size since this makes a very strong result as stated in the following theorem possible. Symmetry analysis however supports the two concepts (see section 4.1.3). Also, as shown in the previous chapter, the determinant bit groups are made up of groups of equally sized blocks of the size of the smallest group, S_0. Claim 1, equally sized parts of decomposition, implies $|I| = 2^k$, $k \in [1;d]$ (since $|\mathcal{V}| = 2^d$) and therefore $|\mathcal{A}_i| = 2^{d-k} \; \forall i \in I$.

A weaker version of claim 2 is to simply claim the same number of neighbours in a different block. Symmetry analysis, however, supports this version of claim 2 (see section 4.1). It furthermore makes sense for the later introduced weightings of links (see section 5.1).

The upside of using the Cayley graph concept on the network of the model is a result on the nature of such tilings allowing to classify them easily and calculating any possible tiling nearly without any cost. It is quite a strong

result that could be used to classify and describe every pattern observed up to now.

Theorem: A network tiling $(\mathcal{A}_i)_{i\in I}$ of $\Gamma(\mathcal{V},\mathcal{S})$ is a decomposition of the network into cosets. This means one of the $\mathcal{A}_i$ is a subgroup (without loss of generality $\mathcal{A}_0$. It shall also have $\mathbf{0}\in\mathcal{A}_0$),

$$\mathcal{A}_0 \text{ is a subgroup of } (\mathcal{V},+), \text{ written as } \mathcal{A}_0 \trianglelefteq \mathcal{V} \tag{3.8}$$

(i. e. $\mathbf{0}\in\mathcal{A}_0$ and $\mathbf{a},\mathbf{b}\in\mathcal{A}_0 \;\Rightarrow\; \mathbf{a}+\mathbf{b}\in\mathcal{A}_0$. Also the inverse of every element in $\mathcal{A}_0$ has to be in $\mathcal{A}_0$, which is trivial in this case since all elements are self-inverse).

Every other $\mathcal{A}_i$, $i\in I\setminus\{0\}$ is a coset of $\mathcal{A}_0$,

$$\mathcal{A}_i \text{ is a coset of } \mathcal{A}_0, \text{ that is } \mathcal{A}_i = \mathbf{a}+\mathcal{A}_0 \quad \text{for } \mathbf{a}\in\mathcal{A}_i, i\in I \tag{3.9}$$

(i. e. every $\mathcal{A}_i$ is a "translation" of $\mathcal{A}_0$, taking any $\mathbf{a}\in\mathcal{A}_i$ every element in $\mathcal{A}_i$ can be reached by adding an element of $\mathcal{A}_0$)

Proof: Let $S_{ij}\subseteq\mathcal{S}$, $i,j\in I$ as in eq. (3.7).

1. Since eq. (3.7):

$$\mathcal{A}_i + S_{ij} \subseteq \mathcal{A}_j \quad \forall i,j\in I, \tag{3.10}$$

and with eq. (3.6) (since therefore also $|\mathbf{x}+\mathcal{A}_i| = |\mathcal{A}_j|\ \forall\mathbf{x}\in\mathcal{V}, i,j\in I$):

$$\mathbf{s}+\mathcal{A}_i = \mathcal{A}_j \quad \forall\mathbf{s}\in S_{ij}. \tag{3.11}$$

2. Furthermore let $\mathbf{x}\in\mathcal{V}$ and $i\in I$, then since $\mathcal{V}=\langle\mathcal{S}\rangle$, $\mathbf{x}$ can be decomposed into

$$\mathbf{x} = \mathbf{s}_1+\ldots+\mathbf{s}_m, \quad \text{with } \mathbf{s}_1,\ldots,\mathbf{s}_m\in\mathcal{S}. \tag{3.12}$$

Because $\bigcup_{j\in I} S_{ij}=\mathcal{S}$, there is $j_1\in I$ such that $\mathbf{s}_1\in S_{ij_1}$. Likewise for every $k\in[2,m]$ can be found a $j_k\in I$, such that

$$\mathbf{s}_k \in S_{j_{k-1}j_k}. \tag{3.13}$$

Then with eq. (3.11):

$$\begin{aligned} \mathcal{A}_i + \mathbf{x} &= \mathcal{A}_i + \mathbf{s}_1 + \ldots + \mathbf{s}_m \\ &= \mathcal{A}_{j_1} + \mathbf{s}_2 + \ldots + \mathbf{s}_m \\ &= \ldots = \mathcal{A}_{j_m}. \end{aligned} \tag{3.14}$$

In total therefore

$$\forall i \in I, \mathbf{x} \in \mathcal{V}\ \exists j \in I: \quad \mathbf{x} + \mathcal{A}_i = \mathcal{A}_j. \tag{3.15}$$

3. Examine eq. (3.15) for $i = 0$:

$$\forall \mathbf{x} \in \mathcal{V}\ \exists j \in I: \quad \mathbf{x} + \mathcal{A}_0 = \mathcal{A}_j. \tag{3.16}$$

Since $\mathbf{x}$ takes all values in $\mathcal{V}$, j takes all values in I. But since $\mathbf{0} \in \mathcal{A}_0$, also $\mathbf{x} \in \mathcal{A}_j$ must hold. Therefore:

$$\mathbf{x} + \mathcal{A}_0 = \mathcal{A}_j \quad \forall \mathbf{x} \in \mathcal{A}_j\ \forall j \in I, \tag{3.17}$$

and in particular

$$\mathbf{x} + \mathcal{A}_0 = \mathcal{A}_0 \quad \forall \mathbf{x} \in \mathcal{A}_0 \quad \text{(i. e. } \mathcal{A}_0 \text{ is closed under ”+”).} \tag{3.18}$$

And since furthermore $\mathbf{0} \in \mathcal{A}_0$, $\mathcal{A}_0$ is a subgroup of $\mathcal{V}$ and $\mathcal{A}_i$, $i \in I$ is a coset of $\mathcal{A}_0$.

□

What is the use of this result? It has been shown that a decomposition that fulfils the two claims, such as consisting of equally sized parts, has to decompose the network in an algebraic way into cosets. Since these cosets are uniquely defined by the subgroup $\mathcal{A}_0$ of $\mathcal{V}$, all tilings of $\mathcal{V}$ can simply be given by calculating its subgroups. This is fairly easy in case of the underlying graph, only the generating elements have to be given.

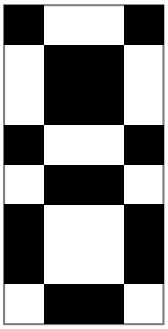
Though, unlike for determinant bit groups, all patterns found in the context of this work did fit into a tiling, these tilings might be so fine that they are of little use. For example, a tiling defined by a subgroup of only two elements only groups together two nodes, thus there are 2^{d-1} tiles. That is the case for

$d_M = d-1$ patterns. A concept combining tilings and determinant bit groups would be desirable.

The following subsections give examples of how to apply tilings on patterns and compare them to pattern describing concepts elaborated in previous works.

3.2.1 Examples of tilings and their arrangement in blocks

As described in section 2.3.2 some patterns do not fit into determinant bit indicated groups. The simplest example in the context of this work is the pattern shown in figure 3.2.1, a "chess board" pattern without determinant bits.



00000 00011
00101 00110
01001 01010
01100 01111
10001 10010
10100 10111
11000 11011
11101 11110

Figure 3.1: Left side: Pattern with $d = 5$, $p = 0.02$, started from empty network, got established after 10^3 iterations and stayed almost static afterwards. $[[t_l; t_u] = [1; 7]]$;
Right side: Bit vectors of occupied nodes.

Applying the tiling concept this pattern perfectly fits into the tiling

$$\mathcal{A}_0 = \langle 00011, 00101, 01001, 10001 \rangle \quad \text{and} \quad \mathcal{A}_1 = \mathcal{A}_0 + 11111. \tag{3.19}$$

In this case $\mathcal{A}_0$ is the block consisting of all occupied nodes, $\mathcal{A}_1$ the one with all unoccupied nodes.

Also more general patterns such as in section 2.3.4 can be described by tilings.

For example, the pattern of figure 2.17 (p. 35) does partly fit into groups defined by determinant bits. But these groups again split up into parts that have different statistical properties. Using a tiling defined by the subgroup

$$\begin{aligned}\mathcal{A}_0 \;= \langle\;& 100000001000,\;\; 100001000000,\\ & 100000000100,\;\; 000000100001,\\ & 100000000010,\;\; 000000010001,\\ & 100000000001,\;\; 010000000000 \;\rangle\end{aligned} \tag{3.20}$$

leads to tiles with equal statistical properties for nodes of each tile. Arranging nodes according to $\mathcal{A}_i$ leads to an arrangement as shown in figure 3.2(a).

Also the pattern of figure 2.18 (p. 36) fits into a tiling. This one is defined by

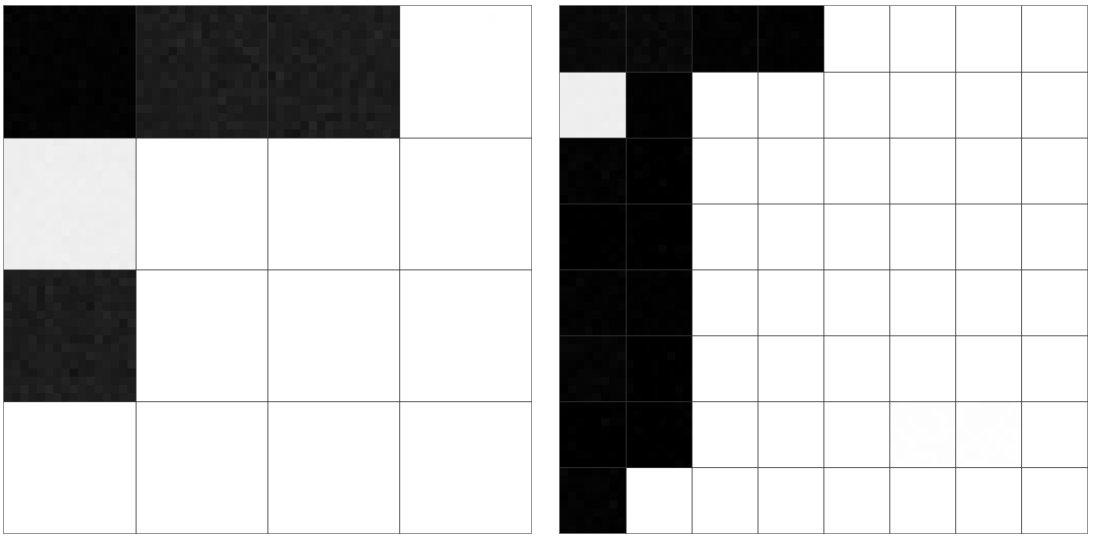
$$\begin{aligned}\mathcal{A}_0 \;= \langle\;& 010000000100,\;\; 000100000100,\\ & 010000000010,\;\; 000000100100,\\ & 010000000001,\;\; 000000001100 \;\rangle.\end{aligned} \tag{3.21}$$

The corresponding arrangement is shown in figure 3.2(b).

The fully dynamical pattern in figure 2.20 on p. 37 ($d=13$, $d_M=10$) only fits into a very fine tiling grouping four nodes together in tiles:

$$\mathcal{A}_0 = \langle\, 0000000000011, 0000000000110 \,\rangle. \tag{3.22}$$

This is an example of semi-determinant bits (see next subsection). Whereas the determinant bits are the ten leftmost bits, the three last bits, the non-determinant bits, only appear in some combinations inside a tile (always either an even or odd number of them set among nodes inside a single tile).



(a) Blocks of the $d_M = 3$ pattern shown in 2.17 (p. 35). [started from prepared pattern, means taken over $5 \cdot 10^4$ iterations]

(b) Blocks of the $d_M = 5$ pattern shown in 2.18 (p. 36). [started from empty network, after 10^3 iterations means taken over $6 \cdot 10^4$ iterations]

Figure 3.2: Blocks of two patterns shown in section 2.3.4. The nodes are arranged in tiles, that are separated by lines. The tiles again were arranged to be near the corresponding nodes in the arrangement of section 2.3.4.
[$p = 0.02$, $d = 12$, $m = 2$, $[t_l; t_u] = [1; 10]$]

3.2.2 Tilings and determinant bits

Tilings are a finer decomposition than that by determinant bit groups. For determinant bits as described in section 2.1 all group sizes are integer multiples of 2^{d-d_M} (all values of non-determinant bits vary in a group), which is the size of group S_0. Thus all groups themselves can be decomposed into blocks of that size, nodes of each block having the same values in determinant bits. The base tile $\mathcal{A}_0$ hence is generated by nodes which in turn generate all values in non-determinant bits. In case of the determinant bits being the last d_M of all d bits, those are:

$$\begin{array}{cc} \overbrace{1\,0\,\ldots\,0}^{d-d_M} & \overbrace{0\,\ldots\,0}^{d_M} \\ 0\,1\,\ldots\,0 & 0\,\ldots\,0 \\ \vdots & \vdots \\ 0\,0\,\ldots\,1 & 0\,\ldots\,0 \end{array} \tag{3.23}$$

If a pattern fits into determinant bit groups a finer decomposition should not be used. On the other hand, as seen in section 2.3.2, there are patterns that have a priority in determinant bits (primary, secondary, ...) and do not fit into groups induced by determinant bits. In those cases they still fit into the tiling as described above.

As mentioned in [6, p. 26] there is a further concept regarding determinant bits: semi-determinant bits. Semi-determinant bits are bits that are coupled together in their value within determinant bits, e. g. for most occupied nodes they have equal value. Tiling decompositions are consistent with them: coupled determinant bits correspond to a generating bit vector of the base tile $\mathcal{A}_0$ with these bits set to 1. For instance, if the last two bits are semi-determinant the corresponding generating bit vector is $0 \ldots 011$. Then all vectors in $\mathcal{A}_0$ have both bits set to 1 (unless of course other generating bit vectors have only one of the last bits set to 1). Other tiles have values in them dependent on their translation $\mathbf{v} + \mathcal{A}_0$.

3.2.3 Comparison to the module concept

In [7] and [10] patterns are described by *pattern modules*. A pattern module is a lower dimensional sub-cube (or "slice") of the hypercube of bit strings in a way that each block of a group according to determinant bits has one representative in it. The linkage between the representatives of blocks is exactly the neighbourhood distribution between the blocks. The whole graph thus decomposes in those slices. This concept is complementary to the determinant bit group concept in a way that it does not group nodes with equal statistical properties but groups nodes that represent the structure of the whole decomposition.

The tiling concept can be adapted to the module concept in following way. Given a tiling $(\mathcal{A}_i)_{i=0,\ldots,n}$ a possible module $\mathcal{M}$ is

$$\mathcal{M} = \{\mathbf{a}_0, \ldots, \mathbf{a}_n;\ \mathbf{a}_i \in \mathcal{A}_i,\ i = 0, \ldots, n\}. \tag{3.24}$$

All other modules are given by translations of $\mathcal{M}$ by elements $\mathbf{b}$ of $\mathcal{A}_0$, $\mathcal{M}_\mathbf{b} = \mathbf{b} + \mathcal{A}_0$ (in that way all translated representatives are still representatives of the same tile, $\mathbf{b} + \mathbf{a}_i \in \mathcal{A}_i$, because $\mathbf{A}_0$ is a subgroup). Unfortunately there is no natural way to choose the representatives $\mathbf{a}_0, \ldots, \mathbf{a}_n$ (unless they describe

determinant bits, in which case they can be chosen to span a sub-cube). The module concept however has not been developed further.

3.3 The block-link matrix and pattern state vectors

3.3.1 The block-link matrix

Definition: Given a tiling $(\mathcal{A}_i)_{i=0,\dots,n}$ of $\mathcal{V}$ the *block-link matrix* $K \in M_{n\times n}$ is defined as

$$(K)_{ij} := \text{Number of neighbours in } \mathcal{A}_j \text{ of a node in } \mathcal{A}_i. \tag{3.25}$$

In case of determinant bits the network is decomposed in a tiling with $n = 2^{d_M}$ blocks, each block consisting of nodes with equal values in determinant bits, but which vary only in non-determinant bits. The blocks shall be indexed by the decimal representation of their part bit string of determinant bits and the block-link matrix for d_M determinant bits shall be noted K_{d_M}.

Any node $\mathbf{v} \in \mathcal{A}_i$ has $\widetilde{l} = d - d_H(i,j)$ mismatches in determinant bits compared to any node in block $\mathcal{A}_j$ ($d_H(i,j)$ being the number of equal bits in the binary representation of i and j). If $l > m$ the node $\mathbf{v}$ has no neighbours in block $\mathcal{A}_j$, otherwise it has $\sum_{k=0}^{m-l} \binom{d-d_M}{k}$ neighbours in it ($m-l$ further possible mismatches in the $d - d_M$ non-determinant bits).

Thus, defining

$$N_{d_M}^{(\widetilde{m})} := \sum_{k=0}^{\widetilde{m}} \binom{d-d_M}{k}, \quad \text{for} \quad d_M \le d, \quad \widetilde{m} \le m, \tag{3.26}$$

the block-link matrices for determinant bits can be computed in a recursive manner:

$$K_{d_M} = \widetilde{K}_{d_M}^{(m,d_M)} \tag{3.27}$$

$$\widetilde{K}_{d_M}^{(\widetilde{m},l)} := \begin{bmatrix} \widetilde{K}_{d_M}^{(\widetilde{m}-1,l-1)} & \widetilde{K}_{d_M}^{(\widetilde{m},l-1)} \\ \widetilde{K}_{d_M}^{(\widetilde{m},l-1)} & \widetilde{K}_{d_M}^{(\widetilde{m}-1,l-1)} \end{bmatrix} \in M_{2^l\times 2^l} \tag{3.28}$$

$$\vdots$$

$$\widetilde{K}_{d_M}^{(\widetilde{m},0)} := \left[N_{d_M}^{(\widetilde{m})} \right]. \tag{3.29}$$

To get K_{d_M} apply eq. (3.27), then eq. (3.28) recursively replacing $\widetilde{K}_{d_M}^{(\widetilde{m},l)}$ till $l=0$. Finally eq. (3.29) gives the entries in the resulting matrix. There are at most $m+2$ different values of entries: $N_{d_M}^{(\widetilde{m})}$, $\widetilde{m}=0,\ldots,m$ and $N_{d_M}^{(-1)}=0$.

Explanation: Let $\widetilde{K}_{d_M}^{(\widetilde{m},l)}$ be the block-link matrix for a maximum of $\widetilde{m}$ possible mismatches in l of d_M determinant bits. The indices of the first 2^{l-1} rows of this matrix correspond to blocks with the same value in the l-th determinant bit (it is 0). Equally for those of its last 2^{l-1} rows (it is 1).

Thus, nodes of the blocks corresponding to the first half of the rows (second half respectively) are complements of other nodes of the same half with at least one mismatch in determinant bits. Hence, for nodes among one half to be neighbours they can have at most further $\widetilde{m}-1$ mismatches in the other $l-1$ determinant bits. For neighbours in the other half still all $\widetilde{m}$ mismatches in the other $l-1$ determinant bits are permitted.

On the other hand, the i-th row, $0 \leq i \leq 2^{d_M}-1$ (first half), and the $(i+2^{d_M})$-th row (in the second half) of K_{d_M+1} correspond to blocks whose values in determinant bits only differ in the first determinant bit (the reason is the ordering of rows and columns according to the decimal representation of the values of the determinant bits). The j-th entry of the sum of those two rows gives the number of neighbours of a node of block j in blocks i and $i+2^{d_M}$. These two blocks form the i-th block when considering only the last d_M of the d_M+1 determinant bits as determinant. Therefore K_{d_M} can be computed by adding the part matrix consisting of the first half of the rows and the first half of the columns of K_{d_M+1} to its part matrix consisting of the second half of the rows and the first half of the columns:

$$K_{d_M} = \frac{1}{2}\begin{bmatrix} I_{2^{d_M}} & I_{2^{d_M}} \end{bmatrix} K_{d_M+1} \begin{bmatrix} I_{2^{d_M}} \\ I_{2^{d_M}} \end{bmatrix}, \quad (I_k)_{ij} = \delta_{ij},\ i,j \in [0,\ldots,k-1]. \tag{3.30}$$

The recursive construction of block-link matrices reveals their structural self-similarity.

The first 5 block-link matrices for $d = 12$ and $m = 2$ for example are:

$$K_0 = \widetilde{K}_0^{(2,0)} = \begin{bmatrix} 79 \end{bmatrix}$$

$$K_1 = \begin{bmatrix} 12 & 67 \\ 67 & 12 \end{bmatrix}$$

$$K_2 = \begin{bmatrix} 1 & 11 & 11 & 56 \\ 11 & 1 & 56 & 11 \\ 11 & 56 & 1 & 11 \\ 56 & 11 & 11 & 1 \end{bmatrix}$$

$$K_3 = \begin{bmatrix} 0 & 1 & 1 & 10 & 1 & 10 & 10 & 46 \\ 1 & 0 & 10 & 1 & 10 & 1 & 46 & 10 \\ 1 & 10 & 0 & 1 & 10 & 46 & 1 & 10 \\ 10 & 1 & 1 & 0 & 46 & 10 & 10 & 1 \\ 1 & 10 & 10 & 46 & 0 & 1 & 1 & 10 \\ 10 & 1 & 46 & 10 & 1 & 0 & 10 & 1 \\ 10 & 46 & 1 & 10 & 1 & 10 & 0 & 1 \\ 46 & 10 & 10 & 1 & 10 & 1 & 1 & 0 \end{bmatrix}$$

$$K_4 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 9 & 0 & 1 & 1 & 9 & 1 & 9 & 9 & 37 \\ 0 & 0 & 1 & 0 & 1 & 0 & 9 & 1 & 1 & 0 & 9 & 1 & 9 & 1 & 37 & 9 \\ 0 & 1 & 0 & 0 & 1 & 9 & 0 & 1 & 1 & 9 & 0 & 1 & 9 & 37 & 1 & 9 \\ 1 & 0 & 0 & 0 & 9 & 1 & 1 & 0 & 9 & 1 & 1 & 0 & 37 & 9 & 9 & 1 \\ 0 & 1 & 1 & 9 & 0 & 0 & 0 & 1 & 1 & 9 & 9 & 37 & 0 & 1 & 1 & 9 \\ 1 & 0 & 9 & 1 & 0 & 0 & 1 & 0 & 9 & 1 & 37 & 9 & 1 & 0 & 9 & 1 \\ 1 & 9 & 0 & 1 & 0 & 1 & 0 & 0 & 9 & 37 & 1 & 9 & 1 & 9 & 0 & 1 \\ 9 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 37 & 9 & 9 & 1 & 9 & 1 & 1 & 0 \\ 0 & 1 & 1 & 9 & 1 & 9 & 9 & 37 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 9 \\ 1 & 0 & 9 & 1 & 9 & 1 & 37 & 9 & 0 & 0 & 1 & 0 & 1 & 0 & 9 & 1 \\ 1 & 9 & 0 & 1 & 9 & 37 & 1 & 9 & 0 & 1 & 0 & 0 & 1 & 9 & 0 & 1 \\ 9 & 1 & 1 & 0 & 37 & 9 & 9 & 1 & 1 & 0 & 0 & 0 & 9 & 1 & 1 & 0 \\ 1 & 9 & 9 & 37 & 0 & 1 & 1 & 9 & 0 & 1 & 1 & 9 & 0 & 0 & 0 & 1 \\ 9 & 1 & 37 & 9 & 1 & 0 & 9 & 1 & 1 & 0 & 9 & 1 & 0 & 0 & 1 & 0 \\ 9 & 37 & 1 & 9 & 1 & 9 & 0 & 1 & 1 & 9 & 0 & 1 & 0 & 1 & 0 & 0 \\ 37 & 9 & 9 & 1 & 9 & 1 & 1 & 0 & 9 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

As the block-link matrix describes neighbourhood distribution among blocks defined by a tiling, for groups determinant bits the *link matrix* is used. Developed in [11] it is defined as

$$L_{ij} = \text{number of neighbours of a node of group } S_i \text{ in group } S_j. \tag{3.31}$$

Because tilings are a more general decomposition, in case of determinant bits the block-link matrix can be used to calculate the link matrix. Without loss of generality let there be d_M determinant bits, each with determinant value 0 (otherwise consider a value 1 in one of these determinant bits as a deviation from the determinant value). Then all blocks $\mathcal{A}_k$ with j bits set to 1 in k form the group S_j, thus:

$$L_{ij} = \sum_{k \in [0;\, 2^{d_M}-1];\;\; d_H(k,0)=j} (K_{d_M})_{(2^i-1),k} \tag{3.32}$$

($2^i - 1$ is the decimal representation of a bit vector with exactly i bits set to 1. Since all nodes in a determinant bit group S_i have the same neighbour distribution among other groups, it does not matter which block in S_i is used in the sum. Here $2^i - 1$ was chosen.)

If the matrix rows and columns are sorted by the number of deviations from determinant values (first row corresponds to no deviation, the next $\binom{d_M}{1}$ rows correspond to one deviation and so on), the formula can be restated as

$$L_{ij} = \sum_{k=0}^{\binom{d_M}{j}} (K_{d_M})_{(2^i-1),k+\sum_{l=0}^{j}\binom{d_M}{l}}. \tag{3.33}$$

Summing up, the block-link matrix B_{ij} describes neighbourhood distribution among blocks (tiles) and, since determinant bit groups consist of equally sized blocks, it can be adapted to the determinant bit group concept and thereby be used to calculated the link matrix L_{ij}. One application is the theoretical construction of patterns as described in the next section.

3.3.2 The state vector of a pattern

Definition: The *state vector* $\mathbf{z}$ of a pattern with n Blocks is

$$z_i := \text{Mean occupation of nodes in block } \mathcal{A}_i, \quad i = 1, \ldots, n. \tag{3.34}$$

For static patterns (then: $\mathbf{z} \in \{0,1\}^n$) every occupied block should have a number of occupied neighbours in the interval $[t_l;\, t_u]$ and every unoccupied blocks outside of it. Since $(K\mathbf{z})_i$ (K being the block-link matrix of that network decomposition) is the number of occupied neighbours of a node in $\mathcal{A}_i$, the equation of state of the pattern can be given:

$$\chi^{(n)}_{[t_l;t_u]}\left(K_{d_M}\mathbf{z}\right) = \mathbf{z}, \tag{3.35}$$

where for $A \subseteq \mathbb{R}$:

$$\chi^{(n)}_A(\mathbf{x}) := (\chi_A(x_1), \ldots, \chi_A(x_n)) \quad \forall \mathbf{x} \in \mathbb{R}^n, \tag{3.36}$$

$$\text{with} \quad \chi_A(x) = \begin{cases} 0 & x \notin A \\ 1 & x \in A \end{cases} \quad \forall x \in \mathbb{R}.$$

3.3.3 Solutions for static patterns with determinant bits for $d = 12$, $[t_l;\, t_u] = [1;10]$

Taking into account the nodes

$$\begin{aligned} \mathbf{x}_0 &:= (0,0,0,0,0,0,0,0)^T \\ \mathbf{x}_1 &:= (1,0,0,0,0,0,0,0)^T \\ \mathbf{x}_2 &:= (1,0,0,0,1,0,0,0)^T \\ \mathbf{x}_3 &:= (0,1,1,0,1,0,0,0)^T, \end{aligned} \tag{3.37}$$

heuristically a recursion formula for static patterns for $d = 12$ and $[t_l;\, t_u] = [1;10]$ can be stated. "$(\bullet,\bullet)$" is the vector concatenation, for instance, $(\mathbf{x}_1, \mathbf{x}_3) = (1,0,0,0,0,0,0,0,\ 0,1,1,0,1,0,0,0)^T$. Starting with state vectors for $d_M = 2$ and $d_M = 3$ patterns,

$$\begin{aligned} \mathbf{z}_{d_M=3} &= \mathbf{x}_2 \\ \mathbf{z}_{d_M=4} &= (\mathbf{x}_3, \mathbf{x}_1), \end{aligned} \tag{3.38}$$

state vectors of higher d_M patterns can be computed by successively replacing pairs of part vectors $(\mathbf{x}_i, \mathbf{x}_j)$ according to the rules ("$\longrightarrow$" means the substitution of the left side by the right side for every recursion step)

$$\begin{array}{lcl} (\mathbf{x}_1, \mathbf{x}_1) & \longrightarrow & ((\mathbf{x}_1, \mathbf{x}_1), (\mathbf{x}_1, \mathbf{x}_1)) \\ (\mathbf{x}_3, \mathbf{x}_3) & \longrightarrow & ((\mathbf{x}_3, \mathbf{x}_3), (\mathbf{x}_3, \mathbf{x}_3)) \\ (\mathbf{x}_3, \mathbf{x}_1) & \longrightarrow & ((\mathbf{x}_3, \mathbf{x}_2), (\mathbf{x}_2, \mathbf{x}_1)) \\ (\mathbf{x}_3, \mathbf{x}_2) & \longrightarrow & ((\mathbf{x}_3, \mathbf{x}_3), (\mathbf{x}_3, \mathbf{x}_1)) \\ (\mathbf{x}_2, \mathbf{x}_1) & \longrightarrow & ((\mathbf{x}_3, \mathbf{x}_1), (\mathbf{x}_1, \mathbf{x}_1)). \end{array} \tag{3.39}$$

Thus, replacing all pairs in a given $\mathbf{z}_{d_M}$ of the above kind leads to $\mathbf{z}_{d_M+1}$ and so on.

This recursion formula yields state vectors for patterns with determinant bits $d_M \in \{3, \ldots, 10\}$, that for every particular d_M can not be ascribed to patterns with smaller d_M. For some d_M solutions of eq. (3.35) also exist which have the first i part vectors $\mathbf{x}_3$ of the state vector replaced by $\widetilde{\mathbf{x}}_3 := (0,0,0,1,0,1,1,0)^T$ and the last i part vectors $\mathbf{x}_1$ replaced by $\mathbf{x}_0$ (with $i = 1$ this is possible for $d_M \in \{6,7,8\}$, with $i = 2$ for $d_M = 8$, with $i = 3$ for $d_M \in \{8,9\}$).

This method has been used to obtain the static patterns given as examples in section 2.3.3. It is important to mention it does not give all possible static patterns (some of the patterns in section 2.3.4 can be static for $p = 0$, but are not reached by this recursion formula). For $d_M > 10$ no static patterns satisfying eq. (3.35) seem to exist. Unfortunately it is hard to check this issue, because of the immense calculational effort ($2^{2^{d_M}}$ calculation steps).

Some dynamical patterns arise from static patterns. For $p > 0$ some nodes in $\mathcal{A}_i$ with $(K\mathbf{z})_i < t_l$ have a non-negligible probability to become occupied by the influx and to survive the update afterwards. The number of such blocks and the sensitivity of the pattern towards them is crucial for its stability.

A more general approach to theoretically construct patterns is the *mean field approach*. The nodes are assumed to be independently occupied with a given probability which leads to an equation of state for these. Steady macrostates are fixed points of this equation. It has been started in [12] and further studied in [13] using determinant bit groups. The adaption of the tiling concept might be useful to get more general patterns. It might as well be easier for the approach to be performed due to its equally sized parts and a very regular neighbourhood distribution among tiles.

4 Symmetry

The underlying network is not only a regular graph (i. e. all nodes have the same number of neighbours) but also highly symmetric. This chapter gives an overview about the symmetry properties of the underlying network and the established network in case of patterns.

4.1 Symmetries of the underlying network

Studying the symmetry of the system means looking for its symmetry group, i. e. the algebraic group of operations the network is invariant under their action. In case of a network a symmetry group element, a symmetry operation, is a map that maps every node one-to-one to another while all connected nodes stay connected after mapping them (they are *neighbourhood compatible*). Since the network is mapped to itself its symmetry group is its automorphism group.

Three cases of parameter settings lead to pathological situations. If $m = 0$, every node is only connected to its perfect complement and the graph decomposes into these pairs of nodes. In this case the symmetry group is huge, for example, every permutation of a given node with its complement is a symmetry operation. If $m \geq d-1$, all nodes are directly linked to each other (for $m = d$ every node is even connected to itself). Also for this case the symmetry group is extremely large and in its structure different from those for smaller m. For these reasons, the symmetry analysis is limited here to networks with $0 < m < d-1$ (and thus $d > 2$).

For such a network there are two easy-to-find types of symmetry operations:

- *Translations* by $\mathbf{v} \in \mathcal{V}$, i. e. mapping each $\mathbf{a} \in \mathcal{V}$ to $\mathbf{a} + \mathbf{v}$ as described in subsection 3.1.

That is, the translation by $\mathbf{v}$ swaps bit values in $\mathbf{a}$ in every position $\mathbf{v}$ has bit value 1 in. Translating two nodes $\mathbf{a}, \mathbf{b} \in \mathcal{V}$ by $\mathbf{v}$ leaves their Hamming distance invariant (bit values are swapped in both $\mathbf{a}$ and $\mathbf{b}$ in same positions, thus they still differ in the same bit positions): $d_H(\mathbf{a} + \mathbf{v}, \mathbf{b} + \mathbf{v}) = d_H(\mathbf{a}, \mathbf{b})$. Therefore it does not change the links between nodes because those are defined by the Hamming distance between them. Translations are hence symmetry operations.

They form the group $(\mathcal{V}, +) \cong \mathbb{Z}_2^d$.

- *Permutations* of bits, $\pi \in \mathcal{S}_d$, i. e. mapping each $(a_1, \ldots, a_d)^T = \mathbf{a} \in \mathcal{V}$ to $(a_{\pi(1)}, \ldots, a_{\pi(d)})^T =: \pi(\mathbf{a})$.

  A permutation π when applied to two nodes $\mathbf{a}, \mathbf{b} \in \mathcal{V}$ permutes the bits in both nodes in the same way and hence leaves the Hamming distance between them invariant: $d_H(\pi(\mathbf{a}), \pi(\mathbf{b})) = d_H(\mathbf{a}, \mathbf{b})$. Again, it does thus not change links between the nodes. Permutations therefore are symmetry operations, too. Furthermore *transpositions* are used. These are permutations that only swap two bits at position i and j, written $\pi_{(ij)}$.

  The permutations form the group $\mathcal{S}_d$.

Therefore the symmetry group of the network is at least $\mathcal{V} \times S_d$ (every combination of those translations and permutations can be written as one translation followed by one permutation). It is inherited from the underlying hypercube structure.

4.1.1 Dependence of symmetry on d and m

As said above, depending on the number of allowed mismatches, m, there are pathological networks with totally different symmetry. For the interesting range of m, $0 < m < d-1$, however, are the translations and permutations the only symmetry operations? And how does the symmetry group for a given d depend on m? Are there differences between networks of different d?

Computing the size of the symmetry group for a given network is tedious and algorithmically very sophisticated. An algorithm for general networks has been developed in [14] and implemented in the `nauty` program [15]. In the context of this work it was applied on the underlying graph of the model only

for networks with d up to $d=16$ (computational effort is too large for larger d). The size of the symmetry group turns out to be:

$$| \operatorname{Aut}(\mathcal{G}) | = \begin{cases} 2^d \cdot d! & \text{if } (d-m) \text{ even,} \\ 2^d \cdot (d+1)! & \text{if } (d-m) \text{ odd.} \end{cases} \tag{4.1}$$

Interestingly the types of symmetries do not directly depend on m, but on the parity of $(d-m)$. For even $(d-m)$ the only symmetries are translations ($| \mathcal{V} | = 2^d$ of them) and permutations ($| \mathcal{S}_d | = d!$ in total). For odd $(d-m)$ however there are additional symmetries, which appear to be some kind of permutations, enlarging the permutation group to $\mathcal{S}_{d+1}$.

A proof showing the symmetry groups are of that kind for all $d > 2$ would exceed the frame of this work. However it is believed that the results also hold for larger d.

4.1.2 The σ symmetry operations

As seen in the previous section, there are additional kinds of symmetries for odd $(d-m)$. These are named σ operations. Certain types of them are σ_i, $i \in \{1, \ldots, d\}$. σ_i swaps the i-th bit for every node with an odd number of bits set among the other $d-1$ bits and leaves the other nodes as is. The reason why σ_i are symmetry operations only for odd $(d-m)$ is explained in the following figure 4.1.

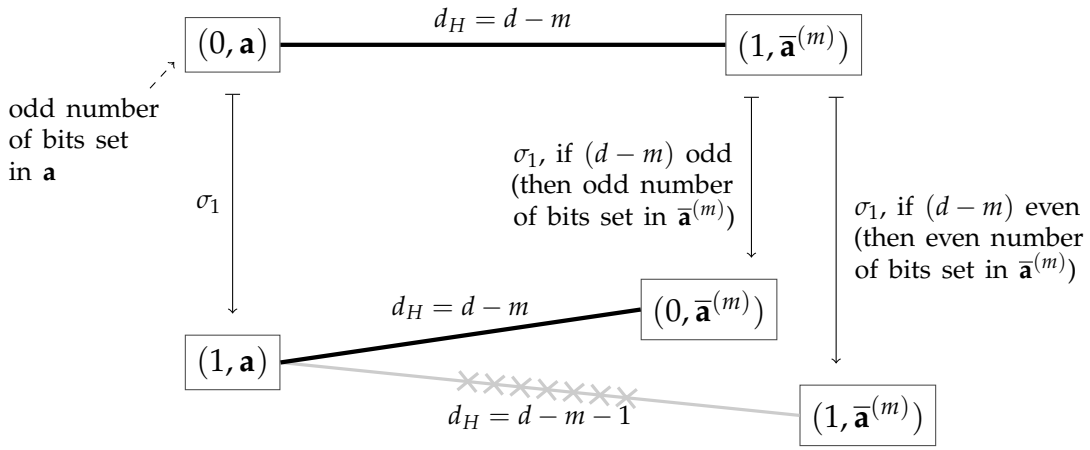


Figure 4.1: A node $\mathbf{v} = (0, \mathbf{a}) \in \mathcal{V}$ and one of its m-mismatch neighbours, $(1, \overline{\mathbf{a}}^{(m)})$, being mapped by σ_1.
The rectangles represent nodes (their bit vector given inside). A thick line between nodes represents their connection in the network with the Hamming distance between the nodes given at the line. The connecting, but crossed out grey line indicates that the two nodes are not directly linked in the network, again with the Hamming distance given. The arrows show the mapping of nodes done by σ_1, with cases given next to the arrow.
$\mathbf{a} \in \{0,1\}^{d-1}$ is the part vector with the bits in positions $2, \ldots, d$ of node $\mathbf{v}$ and $\overline{\mathbf{a}}^{(m)}$ is its complement with exactly m mismatches. It shall have an odd number $k \in \mathbb{N}$ of bits set to 1.
Therefore there are $d-1-k$ bits set to 1 in $\overline{\mathbf{a}}$, the perfect complement of node $\mathbf{a}$. And since m of the bits of $\overline{\mathbf{a}}^{(m)}$ differ from those in $\overline{\mathbf{a}}$, $\overline{\mathbf{a}}^{(m)}$ has an odd number of bits set if and only if $((d-1-k+m)$ odd$) \Leftrightarrow ((d-1+m)$ even$) \Leftrightarrow ((d+m)$ odd$) \Leftrightarrow ((d-m)$ odd$)$. For that reason $(1, \overline{\mathbf{a}}^{(m)})$ is mapped to $(0, \overline{\mathbf{a}}^{(m)})$ by σ_1 if and only if $(d-m)$ odd. Otherwise the link is broken, indicated by the grey crossed out line.

Similarly, in case of odd $(d-m)$, σ_i does not break linkage between nodes with Hamming distance $d_H > d-m$ and is thus a symmetry operation in that case.

Looking at its behaviour together with permutations, one finds, that σ_1 commutates with all permutations that leave the first bit untouched (those permu-

tations do not change the parity of number of bits set in position $2, \ldots, d$, which determines, whether σ_1 changes the first bit or not):

$$\sigma_1 \circ \pi_{(ij)} = \pi_{(ij)} \circ \sigma_1 \qquad \forall i, j \in \{2, \ldots, d\} \tag{4.2}$$

($\circ$ is the concatenation of operations, i. e. in $\tau_2 \circ \tau_1$, τ_1 is applied first, followed by τ_2).

For a transposition that affects the first bit holds:

$$\sigma_1 \circ \pi_{(1i)} = \sigma_i \circ \sigma_1. \tag{4.3}$$

Proof: Let $a_1 \ldots a_d = \mathbf{a} \in \mathcal{V}$. To determine $\sigma_1 \circ \pi_{(1i)} \circ \sigma_1(a_1 \ldots a_d)$, cases have to be discussed separately:

1. If $a_2 \ldots a_d$ has an **odd** number of bits set:

$$\begin{aligned} \sigma_1 \circ \pi_{(1i)} \circ \sigma_1(a_1 \ldots a_d) &= \sigma_1 \circ \pi_{(1i)}(\overline{a}_1 a_2 \ldots a_d) \\ &= \sigma_1(a_i a_2 \ldots a_{i-1} \overline{a}_1 a_{i+1} \ldots a_d). \end{aligned} \tag{4.4}$$

   a) If $a_2 \ldots a_{i-1} \overline{a}_1 a_{i+1} \ldots a_d$ has an **odd** number of bits set:
   Then (because $a_2 \ldots a_d$ has an odd number of bits set) $\overline{a}_1 = a_i$ ($a_1 = \overline{a}_i$ respectively) and $a_1 \ldots a_{i-1} a_{i+1} \ldots a_d$ has an even number of bits set, thus

$$\begin{aligned} \sigma_1(a_i a_2 \ldots a_{i-1} \overline{a}_1 a_{i+1} \ldots a_d) &= \overline{a}_i a_2 \ldots a_{i-1} \overline{a}_1 a_{i+1} \ldots a_d \\ &= a_1 a_2 \ldots a_{i-1} a_i a_{i+1} \ldots a_d \\ &= \sigma_i(a_1 \ldots a_d). \end{aligned} \tag{4.5}$$

   b) If $a_2 \ldots a_{i-1} \overline{a}_1 a_{i+1} \ldots a_d$ has an **even** number of bits set:
   Then (because $a_2 \ldots a_d$ has an odd number of bits set) $a_1 = a_i$ and $a_1 \ldots a_{i-1} a_{i+1} \ldots a_d$ has an odd number of bits set, thus

$$\begin{aligned} \sigma_1(a_i a_2 \ldots a_{i-1} \overline{a}_1 a_{i+1} \ldots a_d) &= a_i a_2 \ldots a_{i-1} \overline{a}_1 a_{i+1} \ldots a_d \\ &= a_1 a_2 \ldots a_{i-1} \overline{a}_i a_{i+1} \ldots a_d \\ &= \sigma_i(a_1 \ldots a_d). \end{aligned} \tag{4.6}$$

2. If $a_2 \ldots a_d$ has an **even** number of bits set:

$$\begin{aligned} \sigma_1 \circ \pi_{(1i)} \circ \sigma_1(a_1 \ldots a_d) &= \sigma_1 \circ \pi_{(1i)}(a_1 \ldots a_d) \\ &= \sigma_1(a_i a_2 \ldots a_{i-1} a_1 a_{i+1} \ldots a_d). \end{aligned} \tag{4.7}$$

a) If $a_2 \dots a_{i-1} a_1 a_{i+1} \dots a_d$ has an **odd** number of bits set:
Then (because $a_2 \dots a_d$ has an even number of bits set) $\overline{a}_1 = a_i$ ($a_1 = \overline{a}_i$ respectively) and

$$\begin{aligned} \sigma_1(a_i a_2 \dots a_{i-1} a_1 a_{i+1} \dots a_d) &= \overline{a}_i a_2 \dots a_{i-1} a_1 a_{i+1} \dots a_d \\ &= a_1 a_2 \dots a_{i-1} \overline{a}_i a_{i+1} \dots a_d \\ &= \sigma_i(a_1 \dots a_d). \end{aligned} \tag{4.8}$$

b) If $a_2 \dots a_{i-1} a_1 a_{i+1} \dots a_d$ has an **even** number of bits set:
Then (because $a_2 \dots a_d$ has an even number of bits set) $a_1 = a_i$ and

$$\begin{aligned} \sigma_1(a_i a_2 \dots a_{i-1} a_1 a_{i+1} \dots a_d) &= a_i a_2 \dots a_{i-1} a_1 a_{i+1} \dots a_d \\ &= a_1 a_2 \dots a_{i-1} a_i a_{i+1} \dots a_d \\ &= \sigma_i(a_1 \dots a_d). \end{aligned} \tag{4.9}$$

In total,

$$\sigma_1 \circ \pi_{(1i)} \circ \sigma_1(a_1 \dots a_d) = \sigma_i(a_1 \dots a_d), \tag{4.10}$$

and since $\sigma_1 \circ \sigma_1 = id$ (applying σ_1 twice does not change anything since if the first application changed one bit, it is changed back by the second application. id is the identity operation)

$$\sigma_1 \circ \pi_{(1i)}(a_1 \dots a_d) = \sigma_i \circ \sigma_1(a_1 \dots a_d). \tag{4.11}$$

□

In the combination with transpositions, as described in eq. (4.2) and eq. (4.3), σ_1 algebraically acts like a transposition of the first bit with another "$d+1$-th" bit, written symbolically as X:

$$\sigma_1 = \pi_{(X1)}. \tag{4.12}$$

An analogous result holds for all σ_i and general σ operations consist of transpositions, like all permutations, but including a σ_i. In case of odd $(d-m)$ they therefore extend the symmetry subgroup of all permutations, $\mathcal{S}_d$, to $\mathcal{S}_{d+1}$.

Summing up, the automorphism groups for $0 < m < d-1$ (hence $d > 2$) are:

$$\mathrm{Aut}(\mathcal{G}) \cong \begin{cases} \mathbb{Z}_2^d \times S_d & \text{if } (d-m) \text{ even,} \\ \mathbb{Z}_2^d \times S_{d+1} & \text{if } (d-m) \text{ odd.} \end{cases} \tag{4.13}$$

Unfortunately for $d = 3$ it is $m = 1$, thus the smallest d with odd $(d-1)$ is $d = 4$ and $m = 1$, which is hard to visualise.

All these considerations are given a graph without different link weightings (see section 5.1). In case of different link weightings $w_m \neq w_{m-1}$ ($m = \max\{i \in \mathbb{N}; w_i \neq 0\}$), the additional symmetries σ_i are not automorphisms in any case since they map a node pair of m-mismatch neighbours to a $m-1$-mismatch node pair. For instance, for odd $(d-m)$ and σ_1 reconsider $(0, \mathbf{a})$ as in figure 4.1. $(0, \mathbf{a})$ and $(0, \overline{\mathbf{a}}^{(m-1)})$ are m-mismatch neighbours, but are mapped to $(1, \mathbf{a})$ and $(0, \overline{\mathbf{a}}^{(m-1)})$, which are $m-1$-mismatch neighbours. Also other graph topological changes might break symmetry, for instance, shifted neighbours as described in section 5.3.

4.1.3 Symmetry and network decompositions

Looking at the two types of network decompositions done so far, one sees they follow certain symmetries of the underlying network.

A tiling $(\mathcal{A}_i)_{i \in I}$ (see section 3.2) is invariant under translations $\mathbf{v} \in \mathcal{A}_0$: adding $\mathbf{v}$ to every node leaves nodes in the same tile as before (since $\mathcal{A}_i$ are cosets of the subgroup $\mathcal{A}_0$). Other symmetry operations break allocation of nodes to tiles. Thus, its symmetry group is $\mathcal{A}_0$, which is a subgroup of the symmetry group of the underlying network,

$$\mathcal{A}_0 \trianglelefteq \mathcal{V} = \mathbb{Z}_2^d \trianglelefteq \mathrm{Aut}(\mathcal{G}). \tag{4.14}$$

A decomposition into groups according to d_M determinant bits (see section 2.1) is invariant under translations by $\mathbf{v} \in S_0 =: \mathcal{A}_0$ since S_0 actually is a base tile and thus a subgroup (see section 3.2.2). Furthermore any permutation of determinant bits of all nodes leaves them in the same group as before (the group allocation only depends on the number of deviations from the determinant values in the determinant bits, which is not changed by such permutations). This yields the symmetry group $\mathcal{S}_{d_M}$ of those permutations. The

grouping is also invariant under permutation of non-determinant bits corresponding to the symmetry group $\mathcal{S}_{d-d_M}$. The total symmetry group again is a subgroup of the symmetry group of the underlying network,

$$\mathcal{A}_0 \times \mathcal{S}_{d_M} \times \mathcal{S}_{d-d_M} \trianglelefteq \mathcal{V} \times \mathcal{S}_d = \mathrm{Aut}(\mathcal{G}). \tag{4.15}$$

Actually all network decompositions into disjunct sets of nodes should be invariant under a subgroup of the symmetry group of the underlying network, thereby ensuring the neighbourhood distribution among those sets is invariant. Thus, in general there might be a network decomposition for every subgroup of $\mathrm{Aut}(\mathcal{G})$, under which the corresponding decomposition is invariant. For subgroups of the $\mathcal{V} = \mathbb{Z}_2^d$ part of $\mathrm{Aut}(\mathcal{G})$ these are tilings. Determinant bit groups then are a mixture of simple tilings (according to determinant bits) and decompositions corresponding to subgroups of the $\mathcal{S}_d$ or S_{d+1} part of $\mathrm{Aut}(\mathcal{G})$.

Altogether any neighbourhood compatible decomposition consists of groups of tiles of a tiling, groups invariant under a subgroup of S_d (S_{d+1}), e. g. determinant bit groups. Thus, a generalisation of the determinant bit concept taking into account general tilings would be desirable.

4.2 Pattern symmetries

As the network establishes patterns some of the symmetries of the underlying graph are broken. A pattern is defined by the probability of occupation for each node, that is constant as long as the pattern is adhered to by the network. Hence a symmetry operation of a pattern must keep these probabilities invariant, i. e. has to map nodes to nodes with the same probability of occupation. Moreover it has to be neighbourhood compatible and as such has to be an element of the symmetry group of the underlying network. The symmetry group of a pattern thus is a subgroup of the symmetry group of the underlying network. If it fits into a network decomposition (as all patterns found so far do), it inherits its symmetry group and adds symmetries to it that interchange decomposition parts with same occupation probability. As the network evolves not all symmetries can be broken for patterns to emerge. From all possible patterns and their symmetry the update rule chooses those that have node occupations, according to the influx parameter p, fulfilling the window thresholds $[t_l; t_u]$.

Again one might assume there are patterns for every subgroup of $\mathrm{Aut}(\mathcal{G})$, which such a pattern is invariant under, as long as it is compatible with the

window threshold. The additional types of symmetries for odd $d-m$ might thus be an explanation for more general patterns appearing in simulations in that case than for even $d-m$. Examples for these have been given in section 2.3.4.

And though the large symmetries of the underlying network are due to the simplicity of the model, patterns are comparably stable against local symmetry defects and perturbations (as seen in chapter 5).

4.2.1 Symmetries for determinant bits

In case of patterns that fits into the group allocation according to determinant bits the symmetry group is easy to determine. The symmetries of the decomposition according to determinant bits was deduced in the previous section (see eq. (4.15)). Further symmetries would interchange nodes between groups, which is only valid for groups with same statistical properties. In determinant bit patterns found so far, however, such a symmetry operation can not be of translation or permutation type (they would, for instance, map the single group of highest mean occupation to a different group of lesser mean occupation). Altogether the size of the basic symmetry group Aut_{d_M} of such a d_M determinant bit group is (from eq. (4.15)):

$$|\,\mathrm{Aut}_{d_M}\,| = 2^{d-d_M} \cdot (d-d_M)! \cdot d_M!. \tag{4.16}$$

The dependence of this size on d_M for $d = 12$ is shown in figure 4.2. The stable patterns (see section 2.3), e. g. $d_M = 4$, seem to have higher symmetry than less stable patterns, e. g. $d_M = 7$. Of course, some patterns have very high symmetry but are not possible due to the window thresholds (e. g. $d_M = 3$ for $[t_l;\ t_u] = [1;10]$). Also this only holds for small influx parameter p. For larger p the purely dynamic patterns such as $d_M = d-1$ dominate.

Furthermore possible are σ symmetries as in section 4.1.2, which are symmetry operations for the underlying network for odd $d-m$ only. For patterns on the other hand the conservation of neighbourhood between blocks is important so σ-operations can rearrange nodes inside blocks while leaving the determinant bits untouched. For even $d-m$ they might break neighbourhood between nodes, but leave neighbourhood distribution among groups invariant, thereby being a quasi-symmetry. For the purely dynamic patterns, such as $d_M = d-1$

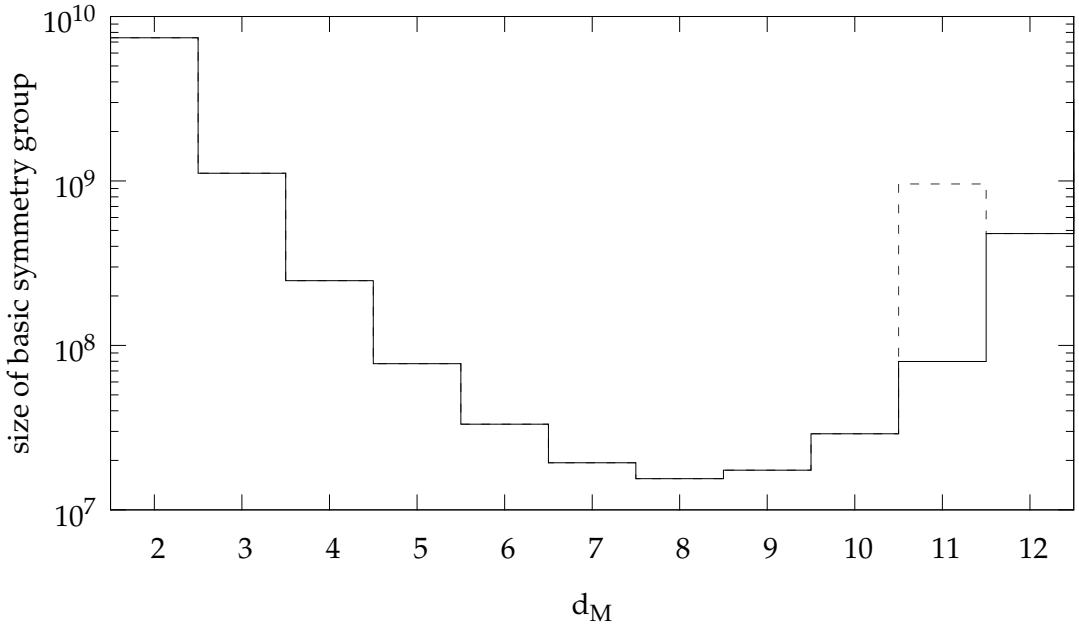


Figure 4.2: Size of the basic symmetry group eq. (4.16) for different d_M in case of $d = 12$. The dashed line indicates the size of the advanced symmetry group with the quasi-symmetry as described in the next section for $d_M = 11$.

patterns, the σ-symmetries seem to play an important role, see next section 4.2.2.

The number of symmetries of a pattern should not be confused with its degeneracy. Whereas the symmetry is a property of a particular pattern with a particular choice of determinant bits, the degeneracy of a pattern is the number of configurations of determinant bits that lead to the same kind of pattern, i. e. another pattern with the same d_M. For determinant bits the degeneracy of a pattern with d_M determinant bits is $\binom{d}{d_M} 2^{d_M}$, that is the number of possible choices of d_M determinant bits out of the d bits and their d_M values.

4.2.2 Quasi-symmetry for purely dynamic patterns

Purely dynamic patterns have groups of similar statistical properties depending on the influx parameter p (nodes with higher mean occupation "fill up" the first $\lfloor \frac{d-1}{2} \rfloor$ groups. Two examples are given in figures 2.10 and 2.11). Applying a σ-operation breaks direct symmetry but can rearrange blocks between adjacent groups. The link violation (as described in subsection 4.1.2) is com-

pensated by the grouping of nodes in blocks, the linkage between blocks stays as-is.

Details are given for the case $d_M = d-1$ with even $d-m$. In such a pattern nodes are grouped together in blocks varying only in one bit (same bit for all blocks). That is the non-determinant bit. Treating blocks as single nodes corresponds to a $d' = d-1$ system in which σ is a symmetry operation. Thus, σ_1, for instance, swaps the first determinant bit depending on the parity of number of bits set in the other $d-2$ determinant bits. In that way some blocks in each group according to determinant bits are exchanged with blocks in an adjacent group. In a $d_M = d-1$ pattern on the other hand most adjacent groups have similar mean occupation; the pattern is altered in a small way using a σ-operation and the S_{d_M} symmetry subgroup (see eq. (4.15)) is increased to S_{d_M+1} which increases the size of the symmetry group by the factor $d_M+1 = d$. Hence, in case of even $d-m$ the symmetry group of $d_M = d-1$ patterns is larger than that of $d_M = d$ patterns (for $d = 12$ the dashed line in figure 4.2).

This increased symmetry could be one explanation why $d_M = d-1$ patterns are favoured over $d_M = d$ patterns in case of even $d-m$ whereas it is the opposite case for odd $d-m$. It might also explain, why these patterns are most stable (least likely to rotate) in their most static forms (in which difference of mean occupation in adjacent groups is least). The behaviour of $d_M = d-1$ patterns is based on that of $d'_M = d'$ patterns in a smaller $d' = d-1$ network, they simply handle two nodes (differing only in the non-determinant bits) as one. Taking advantage of this fact might lead to a method for renormalisation, that ascribes behaviour of general purely dynamic patterns to purely dynamic $d_M = d$ patterns (by grouping nodes to yield a smaller d' network with a $d'_M = d'$ pattern).

The same thoughts hold for other purely dynamic patterns, such as for $d_M = d-3$ for $d = 10$ and $m = 2$ (thus, $d-m$ even), see figure 2.20.

4.2.3 Pattern rotations

For high influx parameters p the $d_M = d-1$ patterns undergo transitions to other $d_M = d-1$ patterns, that is one of the $d-1$ determinant bits becomes non-determinant while the former non-determinant bit becomes determinant (see figure 4.3). Such transitions can be described by rotations of the vector representing the state of the network (done in [8, sec. 7.2]).

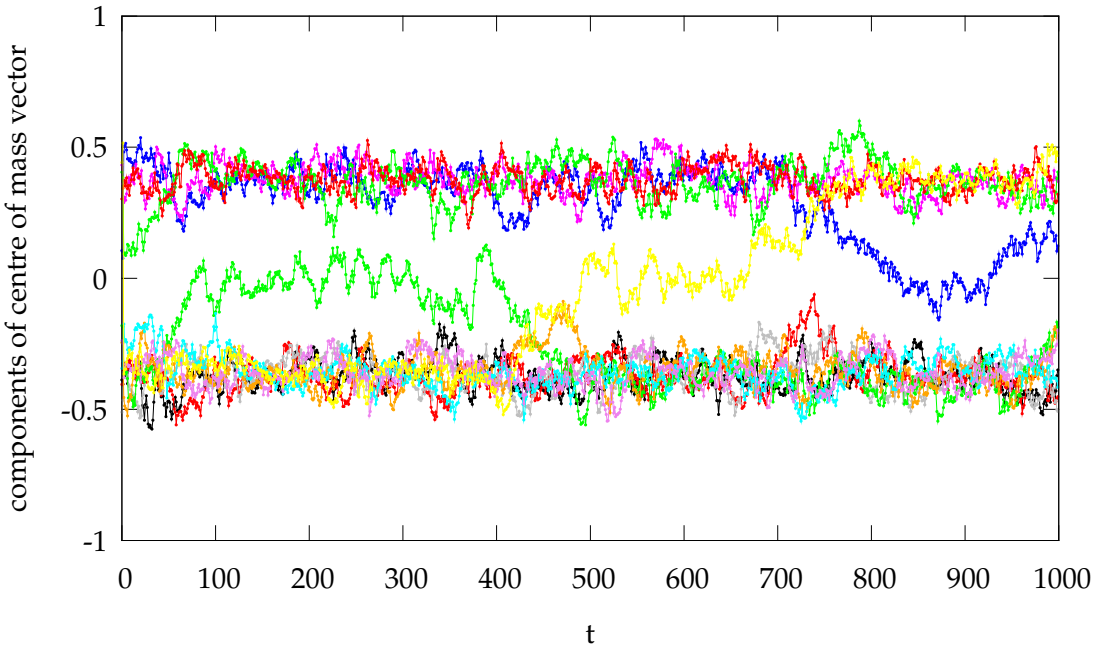


Figure 4.3: Components of the centre of mass vector of a prepared $d_M = 11$ pattern. Whereas it is stable for $t \in [100;400]$, it undergoes a transition in $t \in [400;500]$ to a different $d_M = 11$ pattern: one determinant bit (green) becomes non-determinant and a non-determinant bit (yellow) becomes determinant instead, the pattern "rotates". $[d=12,\ m=2,\ p=0.08,\ [t_l;t_u]=[1;10]]$

A different approach is using the Cayley graph concept and the symmetries of the underlying network. During a rotation of a $d_M = d-1$ pattern a formerly determinant bit becomes non-determinant while the non-determinant bit becomes determinant. Thus, the transition can be described by swapping those two bits (a permutation). If the value of the formerly determinant bit differs from the value of the newly determinant bit, it has to be complemented (a translation, adding a node with solely one bit set to 1 in position of the formerly non-determinant bit). In total, that is a symmetry operation of the underlying graph, but not of one of the patterns, thus changing them.

For other patterns the application of such operations yields other stable patterns that have not yet been discovered simply by observation of simulations. The $d_M = 3$ pattern with chess board subpattern in figure 2.17 (p. 35), for instance, was constructed by starting with an ordinary $d_M = 4$ pattern with the first four bits determinant. On the first $d-1 = 11$ bits σ_1 was applied (ignoring the last bit, as for a $d = 11$ network) thus breaking the symmetry yielding a different kind of pattern.

5 Extensions

5.1 Link weightings

So far the underlying network of the model was defined by the complements of the nodes considering mismatches up to a certain number of them. In that way every kind of neighbourhood is taken equally into account, i. e. independent of number of mismatches, when calculating number of occupied neighbours before doing the update. A more realistic approach would be to think of lesser binding affinity between an antibody of one idiotype and one of another idiotype the more it differs from perfect complement of the first one. Then the stimulation of a B-cell would be lesser the more the antibody binding partners differ from their perfect complement. In our model this can be realized by introducing *link weightings*: Depending on the number m of mismatches from the perfect complement, occupations of the neighbours of a node are weighted by a factor w_m when calculating the total neighbour occupation of that node for the update rule:

$$\partial_w n(\mathbf{v}) = \sum_{i=0}^{m} \sum_{\mathbf{v}' \in \mathcal{V};\, d_H(\mathbf{v},\mathbf{v}')=d-i} w_i \cdot n(\mathbf{v}'). \tag{5.1}$$

In that way for the modelled stimulation of the B-cells of type $\mathbf{v}$ the presence of antibodies of type $\mathbf{v}'$ is weighted depending on its number of mismatches from the perfect complement of $\mathbf{v}$. The underlying graph then becomes a *weighted graph* where links between nodes are weighted according to the w_i.

Looking back at the jigsaw picture underlying the bit string idea one might as well think of mismatches that are crucial for an antibody-antibody interaction to take place: There might be some kind of mismatches that are ”deadly”,

i. e. totally suppressing interaction, e. g. by having an unfortunate spacial structure. Realising that in the bit string model either some links could be removed depending on the probability of them relating to a deadly mismatch. Unfortunately that would make the model and its calculation much more complicated and harder to survey. Alternatively all links of a relating mismatch could be weighted depending on the probability of one of the mismatches being deadly. This idea was first elaborated in [8, ch. 4], given here are simply some corrections.

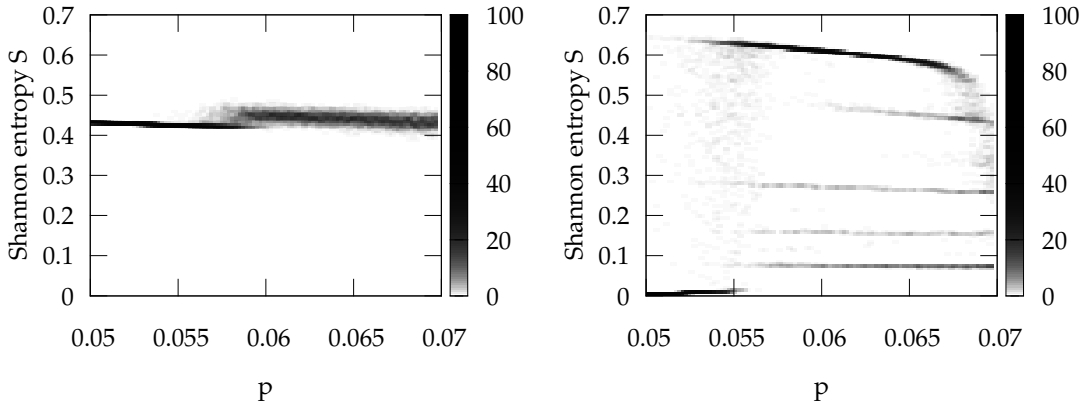
Let $\rho \in [0;1]$ be the probability of a mismatch being deadly, i. e. suppressing interaction between antibodies at all. For simplicity this probability is supposed to be independent of the position of the mismatch on the bit string. Looking at an i mismatch neighbour of a node the probability of none of these mismatches being deadly is simply $(1-\rho)^i$. Since this is the amount of i mismatch neighbours that are supposed to be able to bind the weighting according to i mismatches is:

$$w_i = (1-\rho)^i. \tag{5.2}$$

Since $0 < 1-\rho < 1$ this weighting decreases exponentially with i. Therefore even $w_d > 0$, i. e. even the interaction of an antibody with another of its same idiotype is taken into account, but with a very small weighting. Though this might appear strange at first sight, idiotypic self-binding has been shown to exist in nature (e. g. in [16]). The corresponding antibodies are termed autobodies. As shown later these autobodies do not have big influence on the results of the model.

On the other hand the number of i mismatch neighbours increases fast with i by $\binom{d}{i}$. Since summing neighbour occupations is the most computationally intensive part of the simulations, for sake of economic calculations summing the neighbour occupations should be done just up to those of mismatch i with a weighting w_i greater than a threshold $w_{threshold}$.

Even for a threshold that yields summing neighbour occupations up to 2 mismatches the variety of patterns appearing increases vastly compared to simple link weighting of 1. In figure 5.1 a configuration with link weightings is compared to one without by computing the Shannon entropy over a range of the influx parameter p. Whereas for the system with link weightings different patterns appear, the system without link weightings shows only one in the



(a) Simulations without link weighting, $m = 2$

(b) Simulations with link weighting: $w_0 = 1$, $w_1 = 0.5$, $w_2 = 0.25$

Figure 5.1: Comparison of simulations without and with link weightings. For each of 100 p-values in $[0.05; 0.07]$ 100 simulations were run and after 500 iterations the Shannon entropy was taken over another 10^3 iterations.
$[d = 12, [t_l; t_u] = [2.75; 10]]$

same range of p (a larger range for p for a system without link weighting is given in figure 2.1 on p. 14).

5.2 Further occupation states

In the basic model every node can be occupied, i. e. antibodies of the corresponding idiotype are present, or not. A more realistic approach could take into account antibody concentrations by using more than those two node occupation states. Whereas in the basic model $n(\mathbf{v}) \in \{0, 1\}$, using discrete $N \in \mathbb{N}$ occupation states means $n(\mathbf{v}') \in \{0, \ldots, N-1\}$.

For two occupation states as done so far the B-cell response is modelled by the window rule: to get stimulated the concentration of the binding partners of the antibodies on the B-cell has to be in the window interval $[t_l; t_u]$. For further occupation states the binding partner concentration has to yield a response depending on its size. Experimental results and theoretical considerations (e. g. given in [5]) highly suggest a log-bell-shaped response curve, see also figure 5.2. For determining binding partner concentrations nodes are

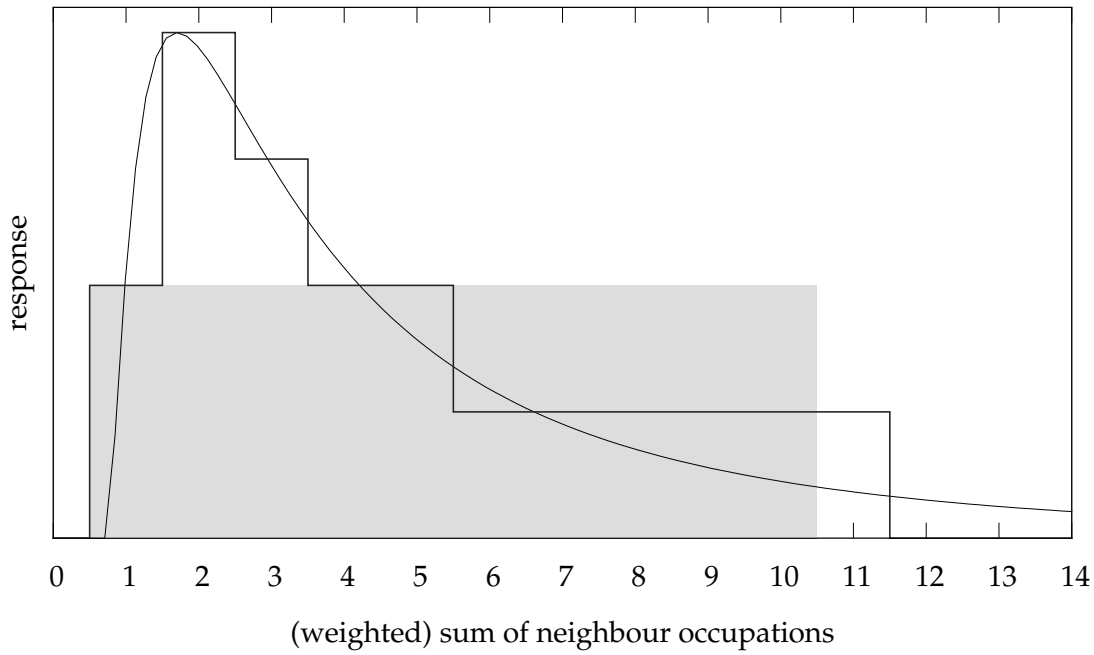


Figure 5.2: Schematic of modelled B-cell response according to concentration of binding partners. The smooth curve is a possible (realistic) log-bell-curve and the thick black piecewise constant function represents a possible modelled response for $N = 5$ occupation states. The grey area is the response modelled so far by the window rule, here $[t_l; t_u] = [1; 10]$.

taken into account with their occupation state number in eq. (1.2), respectively eq. (5.1) for weighted links.

Some simulations have been done, most of them using $N = 5$ occupation states. Using a response function as in figure 5.2 many patterns show up with quite similar structure as for the basic model with two occupation states. For instance, figure 5.3 shows a $d_M = 10$ pattern very similar to the well-known $d_M = d - 1$ pattern in the basic model. In this case highly occupied nodes in the group $S_0, \ldots, S_3$ show all occupation states, mostly the higher ones, whereas the group S_4 only reaches occupation states 0 and 1.

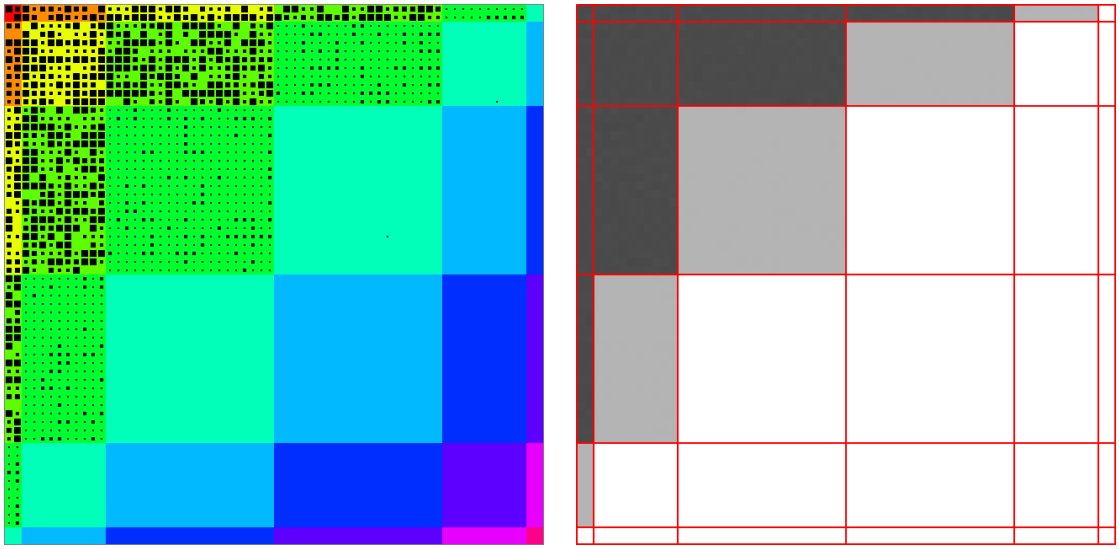


Figure 5.3: $d_M = 10$ pattern in case of 5 possible node occupation states. Black box sizes in left picture (snapshot) according to node occupation state: the bigger the box the higher the state. The structure is similar to the purely dynamic patterns such as the $d_M = d-1$ pattern in figure 2.11 (p. 28).
[$d = 12$, $w_0 = 1$, $w_1 = 0.5$, $w_2 = 0.25$, response function as in figure 5.2, $p = 0.08$, started from an empty network, after 10^3 iterations means taken over 10^5 iterations]

5.3 Generalised antibody interactions

In the basic model binding of antibodies is assumed to cover the whole binding site, i. e. partial binding is not allowed. One might also think of a possible real binding that takes place on part of the binding sites only. A way to extend the model in that way is to allow nodes to have additional neighbours that only have a shifted complement with up to some mismatches. Those kinds of neighbours shall be called "shift-neighbours".

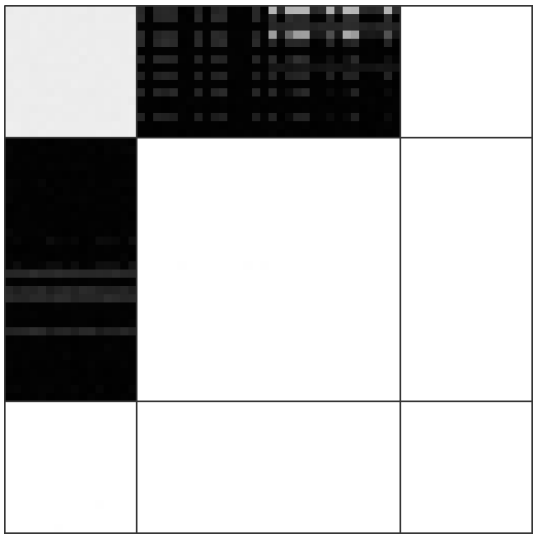
Let $(v_1, \ldots, v_d)^T = \mathbf{v} \in \mathcal{V}$ then its perfect complement shift-neighbours are of the form $(\bar{v}_2, \ldots, \bar{v}_d, v')^T$ and $(v', \bar{v}_1, \ldots, \bar{v}_{d-1})^T$ ($v' \in \{0, 1\}$ being an arbitrary bit value). Schematically shift-neighbours with a shift of 1 look like:

$$
\begin{array}{c|ccccc|c}
a_1 & a_2 & a_3 & \ldots & a_{d-1} & a_d & \\
 & & & \uparrow & & & \\
 & \multicolumn{5}{c}{\text{complementary vector parts}} & \\
 & \multicolumn{5}{c}{\text{up to } m_{shift} \text{ mismatches}} & \\
 & & & \downarrow & & & \\
 & b_1 & b_2 & \ldots & b_{d-2} & b_{d-1} & b_d,
\end{array}
$$

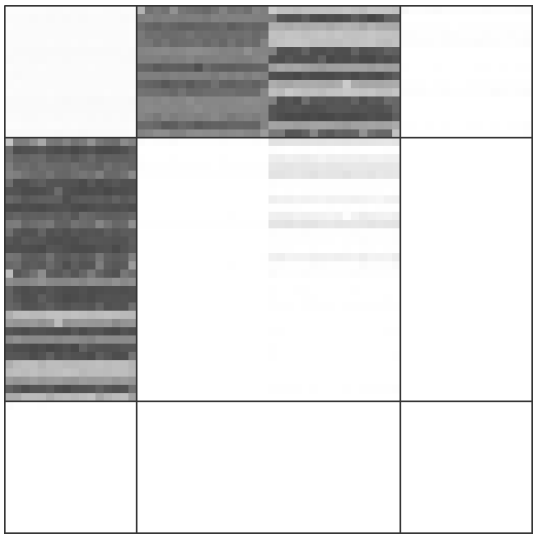
i. e. the antibodies of corresponding idiotypes bind on parts of their binding sites. Using those additional kinds of neighbours the highly symmetrical and regular (all nodes have the same number of neighbours) underlying graph gets local symmetry defects. An extreme example are some nodes becoming self-linked, e. g. the alternating bit vector $1010 \ldots 10$. Those bit vectors correspond to real antibodies being able to bind themselves, so called *autobodies* which have been shown to exist in nature.

Simulations have been done using 1-shift-neighbours without weighting, for no mismatches in vector parts and with up to one mismatch. Figure 5.4 shows examples of $d_M = 4$ and $d_M = 11$ patterns for $d = 12$ and different shift-neighbours. Almost the same patterns as without shift-neighbours appear, the defects only show local effects, e. g. suppressing mean occupation of autobodies.

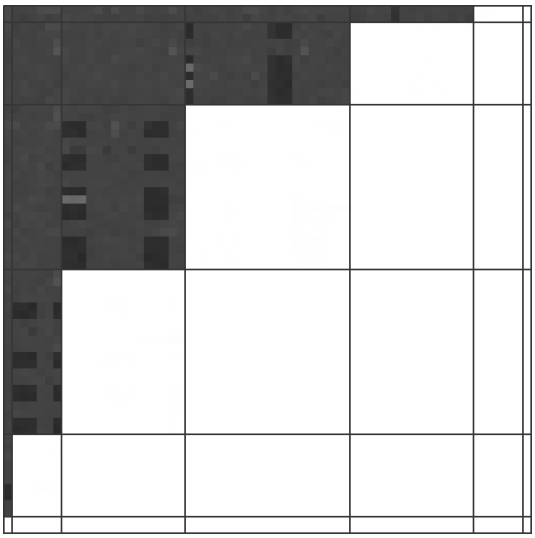
Though these changes are small (4 additional neighbours in case of no-mismatch shift-neighbours compared to 79 "ordinary" neighbours for $d = 12$) it is quite astonishing that the dynamic behaviour of the networks shows to be quite resistant to topological perturbations like these.



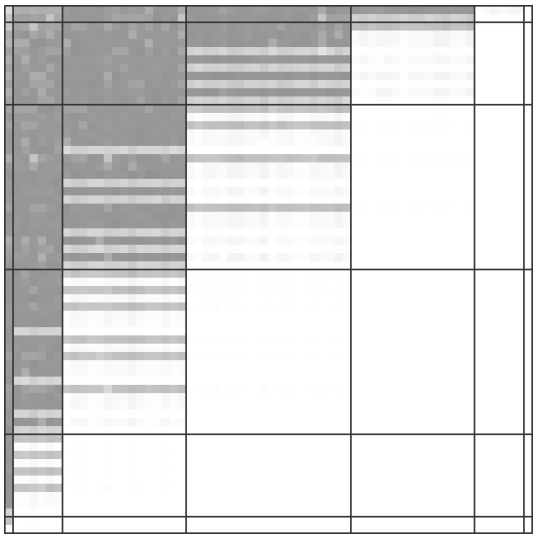
(a) $d_M = 4$ pattern. Underlying network with shift-neighbours with no mismatches, $p = 0.02$, $[t_l; t_u] = [1; 10]$.



(b) $d_M = 4$ pattern. Underlying network with shift-neighbours with up to one mismatch, $p = 0.015$, $[t_l; t_u] = [2; 10]$.



(c) $d_M = 11$ pattern. Underlying network with shift-neighbours with no mismatches, $p = 0.05$, $[t_l; t_u] = [1; 10]$.



(d) $d_M = 11$ pattern. Underlying network with shift-neighbours with up to one mismatch, $p = 0.04$, $[t_l; t_u] = [2; 10]$.

Figure 5.4: Comparison of $d_M = 4$ and $d_M = 11$ patterns with different no-mismatch shift-neighbours (left side) and up-to-one-mismatch shift-neighbours (right side). Because of the self-linkage for many nodes in case of one-mismatch shift-neighbours, the lower threshold t_l has been raised to 2 for these cases (otherwise the system shows the purely dynamic $d_M = 11$ pattern already for small p). Please note the greyscale is the same for both cases, but for more allowed mismatches in shift-eighbours the patterns get more dynamic.
[$d = 12$, $m = 2$, started from an empty network, after 10^3 iterations means taken over 10^5 iterations]

6 Antigens and their influence on the network

One of the great powers of the adaptive immune system is its capability of memorizing once arisen antigens. This memory enhances it to fight a previously encountered antigen more efficiently. Apart from being an essential advantage of species with an adaptive immune system the principle of vaccination is based on that memory.

In sense of idiotypic networks an antigen is believed to induce a stronger concentration of antibodies that can bind the antigen and which itself induces concentration of other bind-able antibodies. These again are believed to keep the concentration of the first antibodies high even after the antigen has vanished. They are thus called the *internal image* of the antigen.

Started in [8], introducing antigens into this model shall be studied further in this chapter. The parameter setting used so far is introduced as well as the procedure of modelling the intrusion of an antigen and its effects on the network. The antigen turns out to induce transitions of the network state. An attempt to understand that mechanism is made using the tiling concept. Furthermore possible internal images of the antigen are identified and an outlook is given to a simpler parameter setting and to the effect of several simultaneously intruding antigens.

In [8] a parameter setting has been given that yields a pattern that might as well establish subpatterns. That setting has been used for results given here. It is the parameter choice

$$d = 12,\ p = 0.06,\ [t_l; t_u] = [2.75; 10],$$
with weighted links $w_0 = 1,\ w_1 = 0.5,\ w_2 = 0.25,\ w_3 = 0.005.$

6.1 Subpattern transitions

The parameters have been chosen for the network to develop a $d_M = 1$ pattern, i. e. with two groups. The group S_0 with a mean neighbour occupation above t_l thus being highly occupied and the less occupied group S_1 due to suppression by a mean neighbour occupation of slightly above t_u. Details for the case of no subpattern are shown in table 6.1.

group	mean occupation	mean neighbour occupation after influx, before update
S_0	0.444 ± 0.003	4.083 ± 0.003
S_1	0.0460 ± 0.0005	10.741 ± 0.007

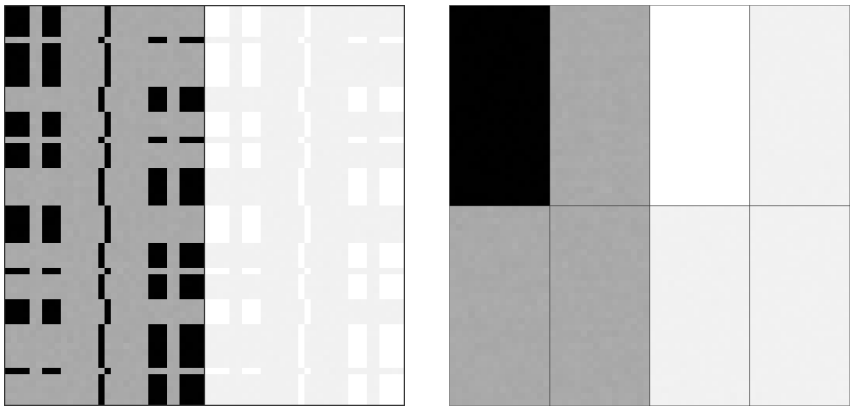
Table 6.1: Statistical properties of nodes in groups without subpattern. Shannon entropy $S \approx 0.630$. Means taken over 10^6 iterations, errors given are statistical errors of summing over nodes in a group.

The $d_M = 1$ pattern again appears in seven different forms. It either has no subpattern or one of six possible types of subpatterns. Nodes in these subpatterns can be arranged in four different sets with equal statistical properties among the nodes in a set:

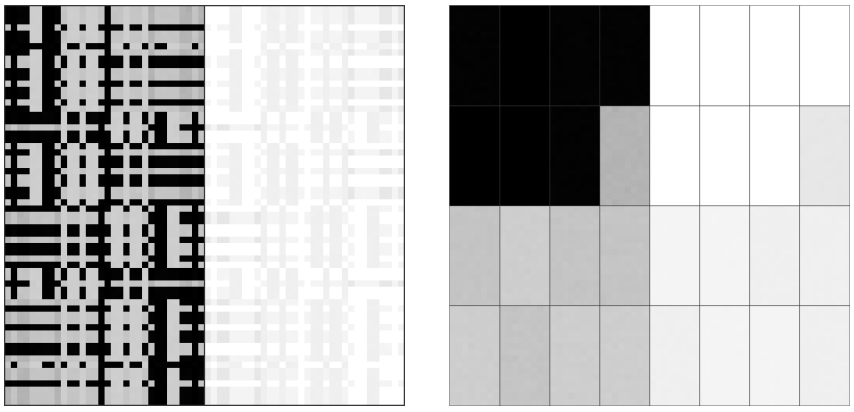
- $S_0^{(high)}$: very highly occupied nodes in S_0,
- $S_0^{(low)}$: less highly occupied nodes in S_0
- $S_1^{(low)}$: slightly occupied nodes in S_1, of same size as $S_0^{(low)}$
- $S_1^{(empty)}$: almost always unoccupied nodes in S_1, of same size as $S_0^{(high)}$

The different kinds of subpatterns can easily be distinguished by their characteristic Shannon entropy and differ in the sizes of these sets, that are in fact sets of tiles. An overview of them is given in figure 6.1. The larger the almost static sets $S_0^{(high)}$ and $S_1^{(empty)}$ are, the smaller the Shannon entropy of the pattern and the more stable it seems to be (less fluctuations due to the influx possible). Nevertheless the larger these sets the less often the pattern appears in simulations, the $S = 0$ pattern (see figure 6.1(f)) actually has never been observed when starting with an empty network.

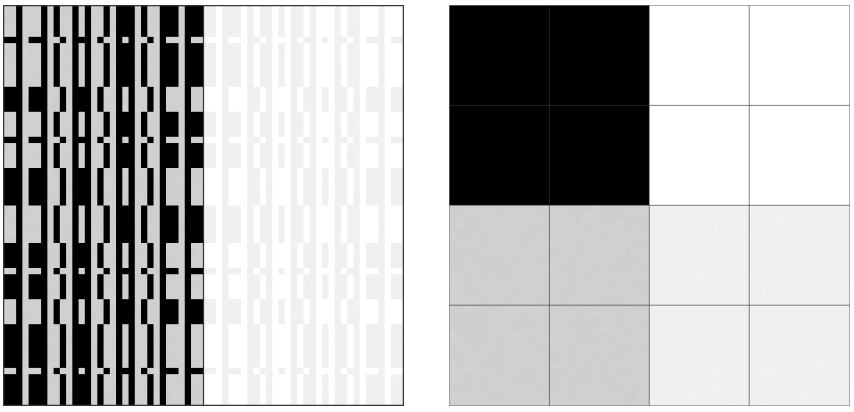
The possibility of a pattern to establish different forms of subpatterns appears to be a way of realising a memory effect in the network. Mechanisms as introducing an antigen turn out to trigger switches to a different subpattern.



(a) Subpattern with $S \approx 0.462$, $|S_0^{(high)}| = 512$



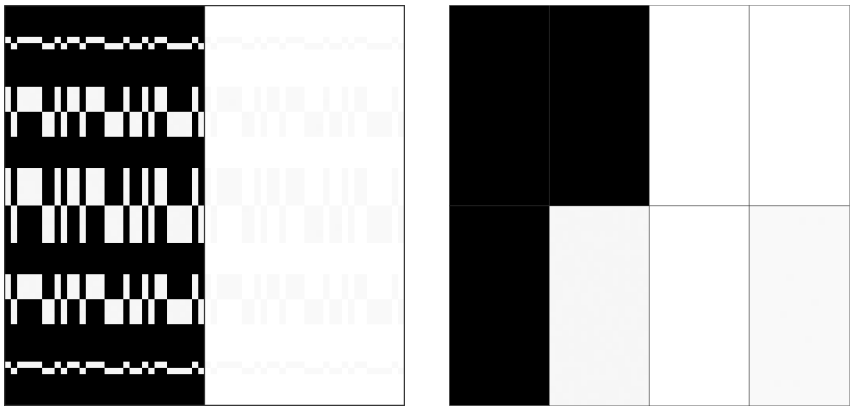
(b) Subpattern with $S \approx 0.312$, $|S_0^{(high)}| = 896$



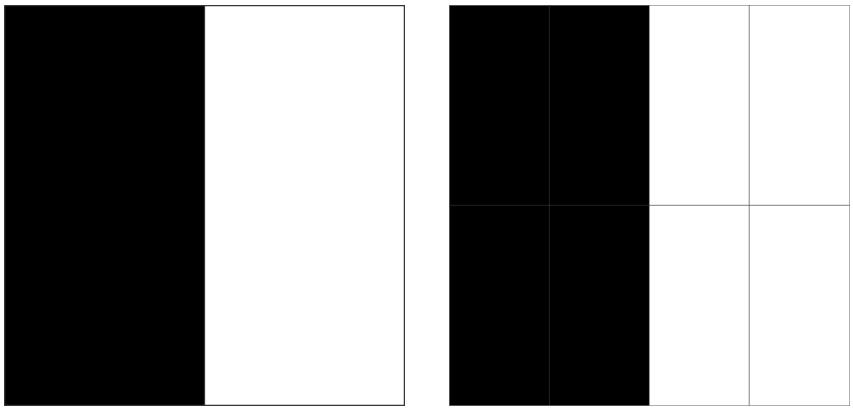
(c) Subpattern with $S \approx 0.254$, $|S_0^{(high)}| = 1024$

Figure 6.1: Possible subpatterns of the $d_M = 1$ pattern. Each one is arranged as in section 2.2 in the left part. The right parts show the block arrangements as in section 3.2.1 and figure 3.2 (p. 53). The darker a grid point the higher the mean occupation of the corresponding node. Means were taken over $1.2 \cdot 10^5$ iterations.

(d) Subpattern with $S \approx 0.128$, $|S_0^{(high)}| = 1280$



(e) Subpattern with $S \approx 0.047$, $|S_0^{(high)}| = 1536$



(f) Subpattern with $S = 0$, $|S_0^{(high)}| = 2048$

Figure 6.1: *(continued)*

6.1.1 Introducing an antigen

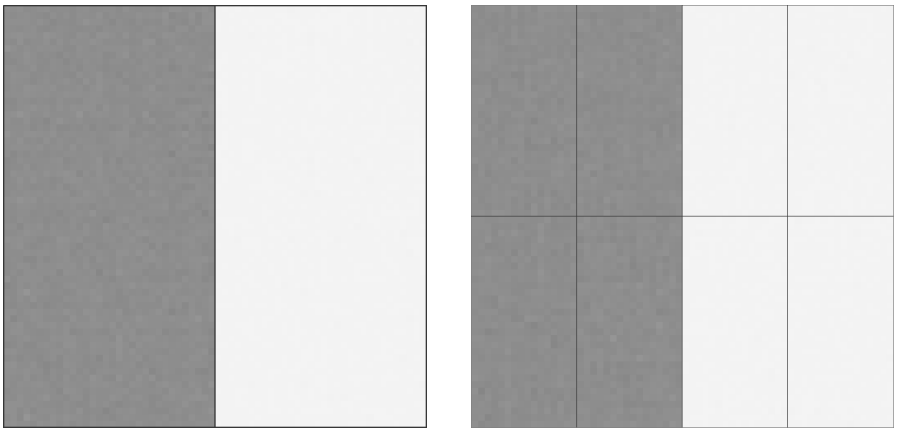
In this model every node of the graph represents an idiotype, i. e. a certain binding type of antibody. Though in biology the kinds of binding an antigen is dependent on the paratope of the antibody, in the simple model an antigen can as well be represented by a node in the network (actually it may be represented by several nodes according to how many different binding sites it has). An intruding antigen is modelled by a node whose occupation is held high independent of influx and update, that is antigens of that type are present. When switching the antigen off that node is simply taken back to follow the ordinary rules like all the others.

Depending on the pattern there are several possible choices of nodes to represent the antigen each one with different influences on the rest of the network. Putting it into a highly occupied area of the network does make little sense as it would hardly be possible for the network to distinguish that always occupied node from other highly occupied ones. Putting it in a group of nodes that are lowly occupied because they have neighbour occupation below t_l does not make much sense either: such an antigen would likely not be fought by the network at all. Though of course in reality there might be such unfortunate antigens, to model the memory effect it is best to choose a node representing the antigen that is lowly due to suppression by highly occupied neighbours. For the parameter setting used here the antigen turns out to induce a subpattern that decreases the mean occupation of the node formerly representing the antigen.

In the $d_M = 1$ pattern without subpattern there are two possibilities to place the antigen: the group S_0 or S_1. As stated above, the less occupied group S_1 (since suppressed by highly occupied neighbours) is the better choice. Figure 6.2 on the next double page step-by-step explains the observed stages. Starting with a $d_M = 1$ pattern without subpattern, (a), an antigen is inserted into group S_1. This node influences its neighbours, their neighbours, and so on, in a way that some nodes in S_0 are more likely to have a neighbour occupation above the lower threshold t_l and thus have a higher mean occupation. Some nodes in S_1 then again become less occupied in means because they become more likely to have a neighbour occupation above the upper threshold t_u. This state stays for some time, (b), till it undergoes a very rapid transition (in the order of 100 iterations) to a stable state with subpattern, (c). That stable state even stays after the antigen is switched off, (d).

preparation of $d_M = 1$ pattern without subpattern

↓

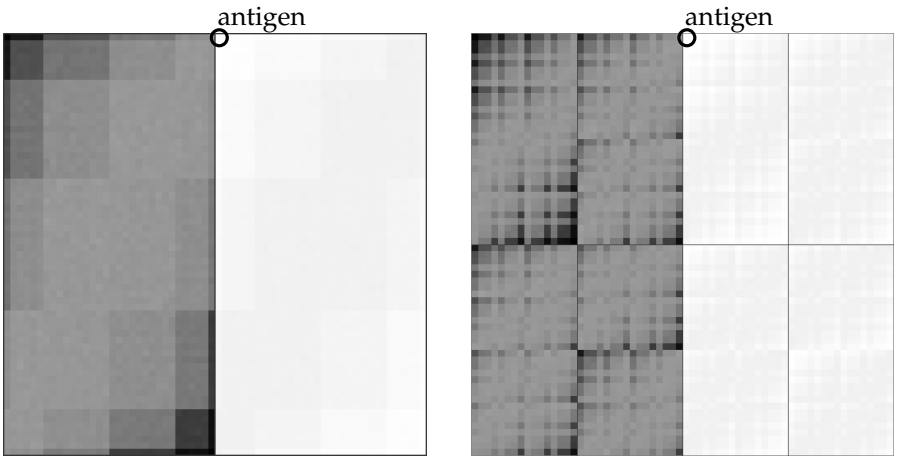


(a) Simple $d_M = 1$ pattern without subpattern.

↓

insertion of antigen into S_1

↓



(b) After the antigen has been inserted (position marked), some of the nodes are encouraged, some are suppressed in their occupation due to the antigen. If the antigen is switched off before the pattern undergoes transition, it falls back to the pattern as in subfigure (a). Though a transition state, it has been taken means over to smoothen the mean occupations.

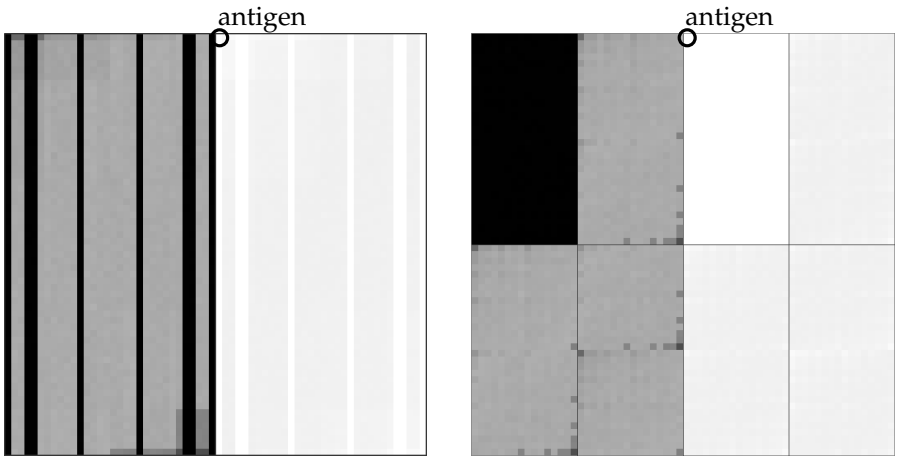
↓

transition to pattern with subpattern occurs

↓

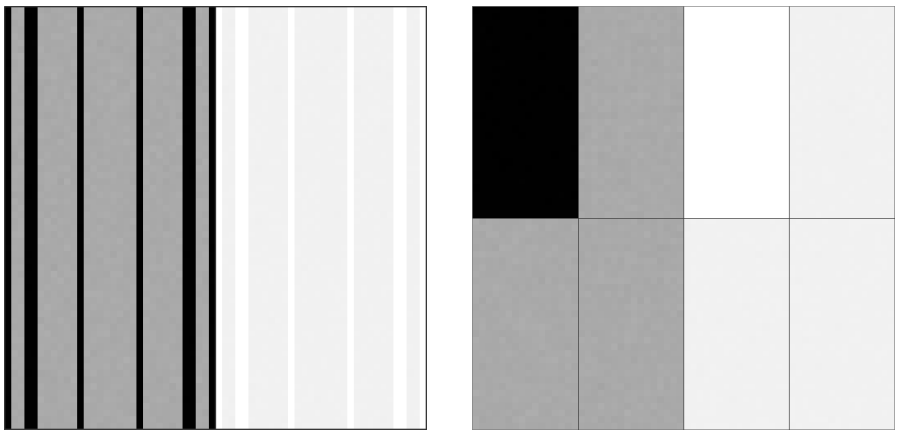
(continued on the next page)

(continued)



(c) The pattern has established a subpattern. Still some of the nodes are exceptionally highly or lowly occupied because of the antigen.

deactivating the antigen (it becomes an ordinary node again)



(d) Pattern with subpattern after the antigen has been switched off. The subpattern stays stable. $S \approx 0.462$, $|S_0^{(high)}| = 512$.

Figure 6.2: Different stages of the network for antigen induced subpattern transition. For each stage the pattern is shown arranged as in section 2.2 in the left part. The right parts show the block arrangements as in section 3.2.1 (see also figure 3.2 on p. 53) according to the blocks the finally reached subpattern fits into. The darker a grid point the higher the mean occupation of the corresponding node. In every case means were taken over $1.2 \cdot 10^5$ iterations.

6.1.2 Subpatterns induced by the antigen

When inserting an antigen like described above the pattern always seems to make transitions to the same kind of subpattern. Even without an antigen such subpatterns are established but with a much smaller probability. This can be quantified by preparing many $d_M = 1$ patterns without subpatterns and observing each of them over time, with an inserted antigen and without. This results in a "pattern decay" as shown in figure 6.3. The subpattern moreover seems more stable than the pattern without subpattern (less transitions back to a pattern without subpattern).

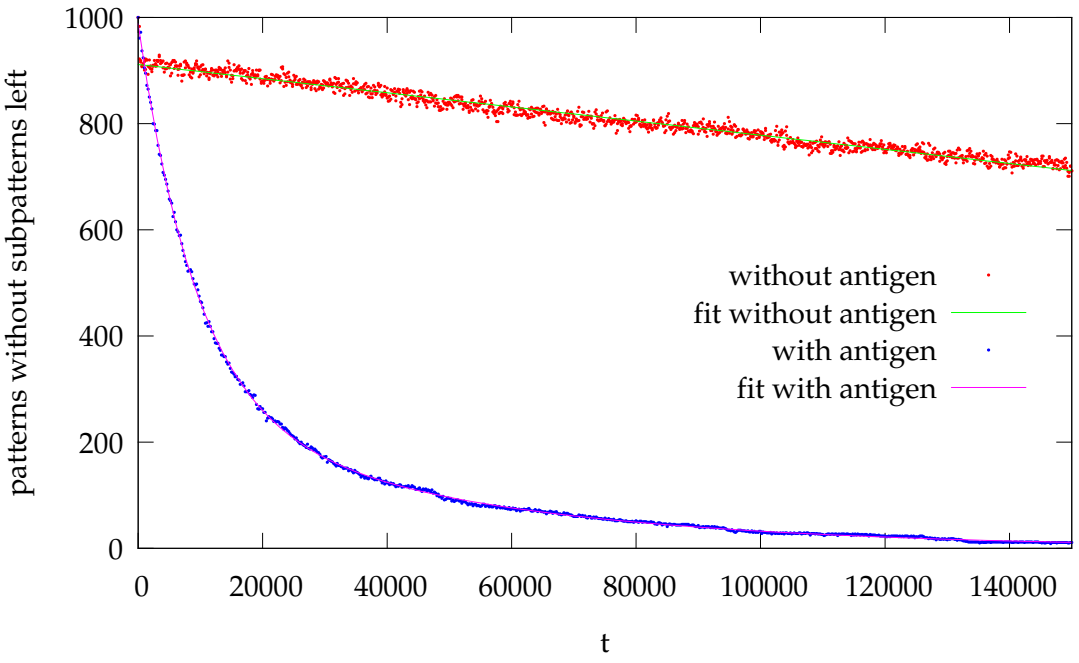


Figure 6.3: Number of networks without subpattern left after preparing 2000 $d_M = 1$ patterns without subpattern and iterating. 1000 of these patterns were inserted an antigen into group S_1. Over each 200 iterations the Shannon entropy was taken to distinguish between patterns with and those without subpatterns.
$[d = 12,\ w_0 = 1,\ w_1 = 0.5,\ w_2 = 0.25,\ w_3 = 0.005,\ p = 0.06,\ [t_l; t_u] = [2.75; 10]]$

Fitting data in figure 6.3 yields the transition probabilities: The data with network preparations without an antigen were fitted by a linear function $A(1 + \lambda t)$ (not enough data to properly fit an exponential function, but an exponential

function $Ae^{\lambda t}$ can be approximated by such a linear function for small $|\lambda t|$). This yields a transition probability for one time step of

$$\rho = 1 - e^{\lambda} = 1.462 \cdot 10^{-6} \pm 6 \cdot 10^{-9}. \tag{6.1}$$

The data corresponding to network preparations with an antigen inserted into group S_1 had to be fitted by a sum of exponentials $A_{Ag,1} \cdot e^{\lambda_{Ag,1} t} + A_{Ag,2} \cdot e^{\lambda_{Ag,2} t}$ which fits much more nicely than a single exponential. The fit yields transition probabilities for one time step and the corresponding amount of transition modes:

$$\begin{aligned} A_{Ag,1} &= 266 \pm 2 \\ \rho_{Ag,1} &= 1 - e^{\lambda_{Ag,1}} = 2.11 \cdot 10^{-5} \pm 10^{-7} \end{aligned} \tag{6.2}$$

and

$$\begin{aligned} A_{Ag,2} &= 722 \pm 2 \\ \rho_{Ag,2} &= 1 - e^{\lambda_{Ag,2}} = 1.068 \cdot 10^{-4} \pm 5 \cdot 10^{-7}. \end{aligned} \tag{6.3}$$

Thus, an antigen increases the probability of establishing a subpattern by two orders of magnitude and one might say the antigen "induces" a subpattern.

Moreover there are at least two modes of transition for the transitions induced by the antigen. For instance, there could be different types of subpatterns the pattern establishes. They have not been distinguished when doing the simulations of figure 6.3. The modes have not been studied further here, but seem to play a larger role for transitions due to two antigens as in section 6.2.2.

In direct observation the antigen induced a subpattern of type identified by $S \approx 0.462$ and $|S_0^{(high)}| = 512$ (see figure 6.2(d)). Its statistical properties after 10^6 iterations after switching off the antigen is shown in table 6.2.

subgroup	mean occupation	mean neighbour occupation after influx, before update
$S_0^{(high)}$	0.9976 ± 0.0003	4.777 ± 0.001
$S_0^{(low)}$	0.336 ± 0.002	3.922 ± 0.002
$S_1^{(low)}$	0.0537 ± 0.0005	10.438 ± 0.006
$S_1^{(empty)}$	0	15.966 ± 0.004

Table 6.2: Statistical properties of nodes in subgroups after an antigen induced the subpattern. Means were taken over 10^6 iterations after the antigen was switched off. Errors given are statistical errors of summing over nodes in a subgroup.

The subgroup corresponding to the tiling (see section 3.2) such a subpattern fits in nicely is of the form (permutations of the bits are possible. Here, the determinant bit shall be the first bit):

$$\begin{aligned} \mathcal{A}_0 = \langle\ & 010000000000,\ 000001000000, \\ & 001000000000,\ 000000100000, \\ & 000100000000,\ 000000010000, \\ & 000010000000,\ 000000001000, \\ & 000000000111\ \rangle. \end{aligned} \tag{6.4}$$

This kind of tiling has been used to group together nodes in figure 6.2 and will be referenced further on when using blocks of the established subpattern. Depending on the determinant value l_0 of the determinant bit, the tilings belong to sets as following ($\overline{l_0}$ is the complement of l_0):

$$\begin{cases} S_0^{(high)} &= \mathcal{A}_0 + l_0 00000000000, \\ S_1^{(empty)} &= \mathcal{A}_0 + \overline{l_0} 00000000000, \\ S_0^{(low)} &= (\mathcal{A}_0 + l_0 00000000001) \\ &\quad \cup (\mathcal{A}_0 + l_0 00000000010) \\ &\quad \cup (\mathcal{A}_0 + l_0 00000000100), \\ S_1^{(low)} &= (\mathcal{A}_0 + \overline{l_0} 00000000001) \\ &\quad \cup (\mathcal{A}_0 + \overline{l_0} 00000000010) \\ &\quad \cup (\mathcal{A}_0 + \overline{l_0} 00000000100). \end{cases} \tag{6.5}$$

The node formerly representing the antigen, $l_0$00000000000, gets located in $S_1^{(empty)}$ and is thus suppressed by the highly occupied group $S_0^{(high)}$.

So why does the antigen increase the probability of establishing a subpattern? As proven in section 3.2 every pattern has to fit into some tiling (that might as well be trivial). In this parameter setting only six different stable subpattern types appear possible. Therefore if the antigen is to induce a subpattern that is stable after switching off the antigen, it has to be one of these. This switch of pattern structure realises a memory, the memory of the antigen is "stored" in the subpattern.

Looking at the direct effect the antigen has as an always occupied node in an otherwise less occupied group, leads to a way to explain the induced transition. As shown in figure 6.2(b) the antigen enforces occupation of some nodes in the group S_0 (in particular its neighbours and their neighbours in that group) by lifting their mean neighbour occupation above the lower threshold t_l. In an analogous manner it suppresses its neighbours and their neighbours in group S_1 (by lifting their mean neighbour occupation above the upper threshold t_u). Most of the enforced nodes turn out to be in the highly occupied set $S_0^{(high)}$ and most of the suppressed are situated in the unoccupied $S_1^{(empty)}$ after transition (compare the right sides of figures 6.2(b) and 6.2(c)). In a way the direct influence of the antigen "pushes" the pattern towards a specific kind of subpattern. Because of fluctuations due to the influx the subpattern is eventually established and stays since it is more stable than the pattern without subpattern.

This idea shall be quantified in the following. Since a pattern is defined by the mean occupation of its corresponding tiling of nodes, a difference between patterns (or states of constant mean occupation in a window of time) can be defined in the following way:

Definition: The *sum of squared differences* (*SSD*) between state 1 (with mean occupations $p_{\mathbf{v}}^{(1)}$) and state 2 (with mean occupations $p_{\mathbf{v}}^{(2)}$) is defined as

$$\mathrm{SSD}(1,2) = \sum_{\mathbf{v}\in\mathcal{V}} \left(p_{\mathbf{v}}^{(1)} - p_{f(\mathbf{v})}^{(2)} \right)^2, \tag{6.6}$$

with $f \in \mathrm{Aut}(\mathcal{G})$ being a network automorphism (as in section 4.1) that yields the smallest SSD.

In this case of subpatterns, f has to map the $d_M = 1$ groups of pattern 2 to the corresponding group of pattern 1.

Computing the SSD between a $d_M = 1$ pattern without subpattern and without antigen and the subpattern as induced by an antigen (the $S \approx 0.462$ subpattern) yields SSD ≈ 176. The SSD between the same pattern without subpattern but with one antigen and the subpattern induced by that antigen afterwards, on the other hand, is SSD ≈ 163. Hence, the antigen as an always occupied node changed the mean occupation of its neighbours and their neighbours (and so on) in a way that got it closer in sense of mean occupation of nodes to the $S \approx 0.462$ subpattern. In figure 6.2(b) the increased and the decreased mean occupation of some nodes (especially neighbours of the antigen and their neighbours) can be seen clearly. Arranging the nodes in the same blocks as those of the induced subpattern, elucidates the small SSD: the block with most increased mean occupation becomes $S_0^{(high)}$, whereas the block with most decreased mean occupation becomes $S_1^{(empty)}$, see figure 6.2(d).

Other subpatterns on the other hand are more distant. An overview of the sum of square differences between the possible subpatterns and a $d_M = 1$ pattern without subpattern but with one antigen in S_1 is given in table 6.3.

entropy $S(\pm 0.001)$	$\lvert S_0^{(high)} \rvert$	SSD (± 1)
0.630	–	19
0.462	512	163
0.312	896	305
0.254	1028	373
0.128	1280	472
0.047	1536	546
0	2048	613

Table 6.3: Properties of possible subpatterns and their sum of square differences to the pattern without subpattern but with one antigen. Means were taken over 10^5 iterations.

Which kind of subpattern and which of its realisation (defined by the base tile $\mathcal{A}_0$ and the mean occupation of tiles) is chosen depends on the influx and is thus subject to coincidence. It has been shown here that an $S \approx 0.462$ subpattern is established most likely. Nonetheless there are several realisations of that pattern with the same SSD and thus the same probability to be established.

6.1.3 Internal images of the antigen

After the antigen is switched off, i. e. the node formerly held occupied is treated by the influx and the update rule again, the pattern remains stable (as seen in

figure 6.3 the pattern with subpattern actually is more stable than the one without subpattern). It can therefore be said that the network memorizes the antigen by changing to a different substructure. Following the basic idea of the memory effect as introduced in the beginning of this chapter, the antigen might be memorized by its internal images in the network.

As shown in the previous section the antigen is suppressed by its highly occupied neighbours, the antibodies fighting it. Their higher occupied neighbours again enforce occupation of these antibodies and keep them occupied even after the antigen has vanished. They are in this case the internal images of the antigen.

In this parameter setting neighbours of nodes are weighted differently. Many slightly occupied nodes weakly linked to the antigen might suppress it as long as they are enough. In the same way a huge set of nodes might be the internal images of the antigen. Here, the most important candidates for internal images shall be determined, i. e. highly occupied internal images that have largest possible and altogether sufficient influence on the antibodies fighting the antigen. The influence a node has on its neighbours is given by the summand it has in the sum of neighbour occupations when applying the update rule (eq. (5.1)). For N nodes each of mean occupation $\langle n \rangle$ their mean contribution to a common m-mismatch neighbour in its sum of neighbour occupation is given by:

$$N \cdot w_m \cdot \langle n \rangle. \tag{6.7}$$

First, the most important antibodies fighting the antigen have to be identified. This is done examining the block-link matrix for the subpattern for neighbours of different numbers of mismatches (see table 6.4). Since these are weighted according to the number of mismatches the block-link matrices should be weighted as well (see table 6.5), each entry thus corresponding to the factor $N \cdot w_M$ in eq. (6.7). Now, since the antigen is in set $S_1^{(emtpy)}$, the rows of $S_1^{(emtpy)}$ give the weighted number of neighbours of the antigens in the other blocks.

Taking into account the mean occupations of groups (see table 6.2, $\langle S_1^{(emtpy)} \rangle = 0, \langle S_1^{(low)} \rangle \approx 0.054 \langle S_0^{(high)} \rangle \approx 1, \langle S_0^{(low)} \rangle \approx 0.34$) the most important antibodies fighting the antigen can be determined, shown in table 6.6, as well as from that the most important internal images, given in table 6.7.

	$S_1^{(empty)}$	$S_1^{(low)}$			$S_0^{(high)}$	$S_0^{(low)}$		
$S_1^{(empty)}$					1			
$S_1^{(low)}$						1		
							1	
								1
$S_0^{(high)}$	1							
$S_0^{(low)}$		1						
			1					
				1				

(a) $m = 0$

	$S_1^{(empty)}$	$S_1^{(low)}$			$S_0^{(high)}$	$S_0^{(low)}$		
$S_1^{(empty)}$	1				8	1	1	1
$S_1^{(low)}$		1			1	8	1	1
			1		1	1	8	1
				1	1	1	1	8
$S_0^{(high)}$	8	1	1	1	1			
$S_0^{(low)}$	1	8	1	1		1		
	1	1	8	1			1	
	1	1	1	8				1

(b) $m = 1$

	$S_1^{(empty)}$	$S_1^{(low)}$			$S_0^{(high)}$	$S_0^{(low)}$		
$S_1^{(empty)}$	8	1	1	1	28	9	9	9
$S_1^{(low)}$	1	8	1	1	9	28	9	9
	1	1	8	1	9	9	28	9
	1	1	1	8	9	9	9	28
$S_0^{(high)}$	28	9	9	9	8	1	1	1
$S_0^{(low)}$	9	28	9	9	1	8	1	1
	9	9	28	9	1	1	8	1
	9	9	9	28	1	1	1	8

(c) $m = 2$

	$S_1^{(empty)}$	$S_1^{(low)}$			$S_0^{(high)}$	$S_0^{(low)}$		
$S_1^{(empty)}$	28	9	9	9	57	36	36	36
$S_1^{(low)}$	9	28	9	9	36	57	36	36
	9	9	28	9	36	36	57	36
	9	9	9	28	36	36	36	57
$S_0^{(high)}$	57	36	36	36	28	9	9	9
$S_0^{(low)}$	36	57	36	36	9	28	9	9
	36	36	57	36	9	9	28	9
	36	36	36	57	9	9	9	28

(d) $m = 3$

Table 6.4: Block-link matrices for different numbers m of mismatches (see section 3.3.1, each only for m-mismatch neighbours). Since the sets $S_0^{(low)}$ and $S_1^{(low)}$ each consist of three blocks, they each span over three columns and rows. Empty entries are of value 0.

	$S_1^{(empty)}$	$S_1^{(low)}$			$S_0^{(high)}$	$S_0^{(low)}$		
$S_1^{(empty)}$					1			
$S_1^{(low)}$						1		
							1	
								1
$S_0^{(high)}$	1							
$S_0^{(low)}$		1						
			1					
				1				

(a) $m = 0$

	$S_1^{(empty)}$	$S_1^{(low)}$			$S_0^{(high)}$	$S_0^{(low)}$		
$S_1^{(empty)}$	0.5				4.0	0.5	0.5	0.5
$S_1^{(low)}$		0.5			0.5	4.0	0.5	0.5
			0.5		0.5	0.5	4.0	0.5
				0.5	0.5	0.5	0.5	4.0
$S_0^{(high)}$	4.0	0.5	0.5	0.5	0.5			
$S_0^{(low)}$	0.5	4.0	0.5	0.5		0.5		
	0.5	0.5	4.0	0.5			0.5	
	0.5	0.5	0.5	4.0				0.5

(b) $m = 1$

	$S_1^{(empty)}$	$S_1^{(low)}$			$S_0^{(high)}$	$S_0^{(low)}$		
$S_1^{(empty)}$	2.0	0.25	0.25	0.25	7.0	2.25	2.25	2.25
$S_1^{(low)}$	0.25	2.0	0.25	0.25	2.25	7.0	2.25	2.25
	0.25	0.25	2.0	0.25	2.25	2.25	7.0	2.25
	0.25	0.25	0.25	2.0	2.25	2.25	2.25	7.0
$S_0^{(high)}$	7.0	2.25	2.25	2.25	2.0	0.25	0.25	0.25
$S_0^{(low)}$	2.25	7.0	2.25	2.25	0.25	2.0	0.25	0.25
	2.25	2.25	7.0	2.25	0.25	0.25	2.0	0.25
	2.25	2.25	2.25	7.0	0.25	0.25	0.25	2.0

(c) $m = 2$

	$S_1^{(empty)}$	$S_1^{(low)}$			$S_0^{(high)}$	$S_0^{(low)}$		
$S_1^{(empty)}$	0.14	0.045	0.045	0.045	0.285	0.18	0.18	0.18
$S_1^{(low)}$	0.045	0.14	0.045	0.045	0.18	0.285	0.18	0.18
	0.045	0.045	0.14	0.045	0.18	0.18	0.285	0.18
	0.045	0.045	0.045	0.14	0.18	0.18	0.18	0.285
$S_0^{(high)}$	0.285	0.18	0.18	0.18	0.14	0.045	0.045	0.045
$S_0^{(low)}$	0.18	0.285	0.18	0.18	0.045	0.14	0.045	0.045
	0.18	0.18	0.285	0.18	0.045	0.045	0.14	0.045
	0.18	0.18	0.18	0.285	0.045	0.045	0.045	0.14

(d) $m = 3$

Table 6.5: Weighted block-link matrices for different numbers m of mismatches (see section 3.3.1, each only for m-mismatch neighbours weighted by the weighting of such links, w_m). Since the sets $S_0^{(low)}$ and $S_1^{(low)}$ each consist of three blocks, they each span over three columns and rows. Empty entries are of value 0.

In set	m	N	Contribution (eq. (6.7))	Neighbours of antigen ($Ag = 000000000000$)
$S_0^{(high)}$	0	1	1	111111111111
$S_0^{(high)}$	1	8	4	$1a_2 \dots a_9 111$ (one 0 in $a_2 \dots a_9$)
$S_0^{(high)}$	2	28	7	$1a_2 \dots a_9 111$ (two 0 in $a_2 \dots a_9$)
	Total:	37	$12 > t_u$	

Table 6.6: Highest occupied nodes that are responsible for the non-occupation of nodes in ${S_1}^{(empty)}$ due to lifting their mean neighbour occupation above the upper threshold t_u, other contributing nodes have mean occupation of 0.3 and less. The set they belong to is given, as well as the number of their mismatches from their neighbours in ${S_1}^{(empty)}$, the number of such nodes (see also tables 6.4 and 6.5) and their mean contribution to the mean neighbour occupation of their neighbours (according to eq. (6.7) and table 6.2).
Since the antigen lies in that group, these are the most important antibodies fighting it. For an antigen $Ag = 000000000000$ (then, the determinant bit l_0 has value 1) their form is given in the last column.

In set	m	N	Contribution (eq. (6.7))	Neighbours of antibodies above
$S_0^{(high)}$	1	1	0.5	$1b_2 \dots b_9 000$ (no mismatch in $b_2 \dots b_9$)
$S_0^{(high)}$	2	8	2	$1b_2 \dots b_9 000$ (one mismatch in $b_2 \dots b_9$)
$S_0^{(low)}$	2	3	0.25	$1b_2 \dots b_9 100$ (no mismatch in $b_2 \dots b_9$)
				$1b_2 \dots b_9 010$ (no mismatch in $b_2 \dots b_9$)
				$1b_2 \dots b_9 001$ (no mismatch in $b_2 \dots b_9$)
	Total:	12	$2.75 = t_l$	

Table 6.7: Highest occupied nodes that are responsible for the almost-sure occupation of nodes in ${S_0}^{(high)}$ due to lifting their mean neighbour occupation above the lower threshold t_l, other contributing nodes have mean occupation of less than 0.3 and have a smaller contribution in sum. The set they belong to is given, as well as the number of mismatches from their neighbours in ${S_0}^{(high)}$, the number of such nodes (see tables 6.4 and 6.5) and their mean contribution to the mean neighbour occupation of their neighbours (according to eq. (6.7) and table 6.2).
Their form as neighbours of the antibodies above (in table 6.6) is given in the last column for sets as in eq. (6.5) (again the determinant bit l_0 has determinant value 1).

Putting the tables 6.6 and 6.7 together, the internal images are nodes of the form (in situation of eq. (6.5) and a determinant value of 1 of the determinant bit l_0, the antigen then is $Ag = 000000000000$):

$$\begin{aligned} \mathbf{v}_{int} = \quad & 1\, c_2\, \ldots\, c_9\, d_1\, d_2\, d_3, \\ \text{with} \quad & d_1 = d_2 = d_3 = 0 \\ & \text{and up to 3 bits of value 1 in } c_2 \ldots c_9 \text{ (others of value 0)}, \\ \text{or with} \quad & \text{one of } d_1, d_2, d_3 \text{ of value 1 (others of value 0)} \\ & \text{and up to 2 bits of value 1 in } c_2 \ldots c_9 \text{ (others of value 0)}. \end{aligned} \tag{6.8}$$

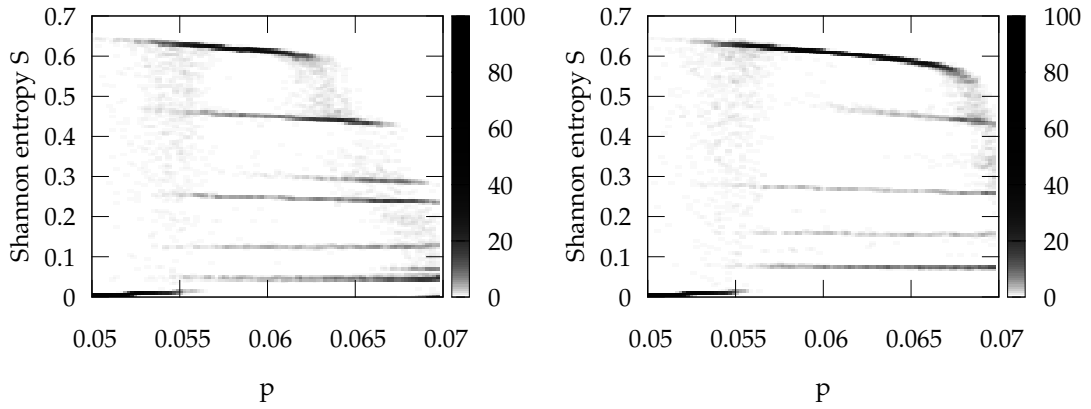
These are in fact $N = \binom{8}{0} + \binom{8}{1} + \binom{8}{2} + \binom{8}{3} + 3 \cdot (\binom{8}{0} + \binom{8}{1} + \binom{8}{2}) = 204$ different nodes (approximately $\frac{1}{20}$ of the whole network). An amount of $\binom{8}{0} + \binom{8}{1} + \binom{8}{2} + \binom{8}{3} = 93$ is situated in the set $S_0^{(high)}$ and $3 \cdot (\binom{8}{0} + \binom{8}{1} + \binom{8}{2}) = 111$ are in $S_0^{(low)}$. Some of them are quite similar to the antigen ($Ag = 000000000000$): 4 internal images is of Hamming distance 1 to the antigen, $4 \cdot \binom{8}{1} = 32$ are of distance 2, $4 \cdot \binom{8}{2} = 112$ are of distance 3 and $\binom{8}{3} = 56$ are of distance 3 to the antigen.

6.2 Outlook for introducing antigens

6.2.1 Simpler parameter values

The property of having patterns with several possible kinds of stable subpatterns is necessary for a setting to show a memory effect behaviour like described above. This property has yet only be discovered using weighted links. For sake of less intensive computability it is desirable to use as few neighbours as possible. In fact 3-mismatch neighbours seem to play a negligible role due to the small weighting $w_3 = 0.005$ (e. g. compare the weighted block-link matrix for 3-mismatch neighbours to the other block-link matrices in table 6.5).

As seen in figure 6.4 almost the same behaviour as above can be seen changing $w_3 = 0$ and $p = 0.065$. In that case four instead of five possible subpatterns emerge (the $S = 0$ subpattern does not emerge when starting an empty network). This is the simplest setting with possible memory effect found so far.



(a) $w_0 = 1$, $w_1 = 0.5$, $w_2 = 0.25$, $w_3 = 0.005$ (b) $w_0 = 1$, $w_1 = 0.5$, $w_2 = 0.25$

Figure 6.4: Shannon entropy histogram for different link weighting settings. For each of 100 p-values in $[0.05; 0.07]$ 100 simulations were run and after 500 iterations the Shannon entropy was taken over another 10^3 iterations. [$d = 12$, $[t_l; t_u] = [2.75; 10]$]

6.2.2 Introducing a second antigen

A realistic antigen has more than one type of epitope, i. e. binding sites antibodies can bind to. Such an antigen therefore could be represented by more than one node in the network. As well, the network should have the ability to

memorize more than one antigen. Keeping this in mind, simulations have been done treating two nodes instead of one like an antigen, i. e. holding it occupied regardless of the dynamic rules. Both nodes were chosen in group S_1 of the pattern without subpattern.

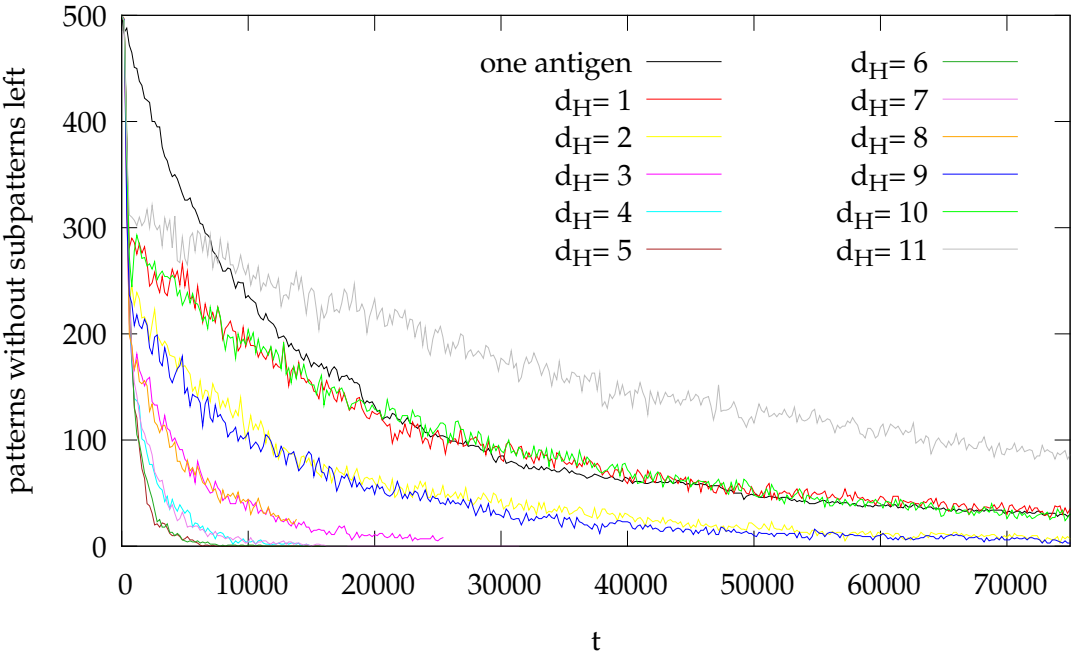


Figure 6.5: Number of networks without subpattern left after preparing 500 $d_M = 1$ patterns without subpattern for each possible Hamming distance d_H of two antigens in group S_1. [$d = 12$, $w_0 = 1$, $w_1 = 0.5$, $w_2 = 0.25$, $w_3 = 0.005$, $p = 0.06$, $[t_l; t_u] = [2.75; 10]$]

As seen in figure 6.5 their influence on the network vastly depends on the similarity of the two nodes representing antigens (similarity in sense of their Hamming distance). Whereas the presence of a second antigen in medium distance to the first one ($d_H = 5$ or $d_H = 6$) highly enforces the establishment of a subpattern, the transition probability for closer or more distant antigens is less. A most distant second antigen ($d_H = 11$) even suppresses it. Again there are two modes of decay with different transitions probabilities. One of these is highly enforced by a second antigen (many decays in the first 1000 iterations). The possibility of two antigens inducing transitions to patterns with different kinds of subpatterns depending on the similarity of the antigens should further be inspected as the ability of the network to memorize two antigens might be based on that.

In this chapter a setting was studied that yields several possible stable subpatterns with transitions between them. These transitions can be induced by an inserted antigen. Using methods developed in previous chapters an explanation for the transition probability thereby being increased has been discovered: the antigen ″pushes″ the pattern towards a certain subpattern in sense of a distance defined between patterns. Moreover the antibodies fighting the antigen and the internal images of the antigen could be identified. Further research is suggested to use a simpler parameter setting and to study the influence of more than one antigen.

Conclusion

Patterns in the bit string model of idiotypic networks are the central subject of this work. A visualisation of them was developed, that proved useful when trying to discover substructures in patterns described by determinant bits so far. Several pattern examples could be given using that visualisation and by finding patterns with the numerical tool of the Shannon entropy. The concept of topology diagrams furthermore helped to get an idea of what stability of a pattern depends on and how pattern transitions happen. Moreover, for the same example the transition probability was determined.

Looking at patterns emerging led to a more general classification of patterns: the decomposition into tilings. Using the Cayley graph concept it could be shown to be complete in for patterns with equally sized blocks. The block-link matrix, the link matrix conveyed to the tiling concept, and the pattern state vector have been defined. Among other issues, they are useful to construct more general patterns some of which do not emerge in simulations.

Even more general patterns could be given using results from symmetry analysis. This furthermore suggested a more general decomposition, in particular, a mixture of tilings and determinant bit groups. The symmetry operations were identified and shown to be only indirectly dependent on the number m of allowed mismatches. A class of symmetry operations was discovered that are only symmetry operations for odd $d-m$. They have been suggested as an explanation for certain purely dynamic patterns, such as the $d_M = d-1$ patterns and could be of help for renormalisation.

The effect of some extensions of the model on its emerging patterns have been discussed. They can be used to make the model more realistic. Especially weighting of links results in an increased variety of possible patterns. Even variations of the network topology such as shifted neighbours turn out to have

a small influence on the structure of patterns emerging though they vastly decrease symmetry.

Many of these newly developed concepts were applied to a scenario of an intruding antigen. The mechanism of modelling an antigen was discussed and an explanation was given for the transition to a different subpattern it induces. Those subpatterns were analysed in detail and the most important internal images of the antigen were identified. For classifying and studying the subpattern the tiling concept proved its great usefulness.

The model this work is based on is not only interesting for modelling its real nature counterpart. It is one of the first of its kind combining the concept of a cellular automaton on a network with probabilistic rules. Models of this kind might be studied for modelling many complex systems in nature. Though its rules are fairly simple a very complex behaviour emerges. Many of its aspects are not clearly understood yet and lots of approaches to it are thinkable.

For understanding patterns more deeply the tiling concept could be generalized to develop a classification concept for all patterns possible. Symmetry analysis is of use here. With its algebraic description it can easily be applied in calculations. Especially in the mean field approaches it might prove to be more useful. Patterns also play an important role in modelling mechanisms as observed in nature. One of these is the intrusion of an antigen. Here, analysing influence of an antigen on the network structure and its appearing subpatterns should be continued. Also the capability of several antigens, either appearing simultaneously or sequentially, to switch between subpatterns is of interest.

Concluding, there is still a long but exciting way to go to fully understand the model and to convey results on its designated subject, the adaptive immune system.

Bibliography

[1] Murphy, K. M., Travers, P., and Walport, M. Janeway's Immunobiology. 7th edition (Garland Science, New York, 2007).

[2] Jerne, N. K. Towards a network theory of the immune system. *Ann. Inst. pasteur Immunol.* **125C**, 373–389 (1974).

[3] Behn, U. Idiotypic networks: toward a renaissance? *Immunological Reviews* **216**(1), 142–152 (2007).

[4] Brede, M. and Behn, U. Patterns in randomly evolving networks: Idiotypic networks. *Physical Review E* **67**(3), 031920 (2003).

[5] Vogelstein, B., Dintzis, R. Z., and Dintzis, H. M. Specific cellular stimulation in the primary immune response: a quantized model. *Proceedings of the National Academy of Sciences of the United States of America* **79**(2), 395–399 (1982).

[6] Thüne, M. Zur Architektur idiotypischer Netzwerke: Ein minimales Modell und einfache Erweiterungen. Diploma thesis, University of Leipzig (2008).

[7] Schmidtchen, H. and Behn, U. Randomly Evolving Idiotypic Networks: Analysis of Building Principles. In *Artificial Immune Systems*, Bersini, H. and Carneiro, J., editors, volume 4163 of *Lecture Notes in Computer Science*, 81–94 (Springer, Berlin/Heidelberg, 2006).

[8] Werner, B. Idiotypische Netzwerke mit Antigenen: Gedächtnis und Selbsttoleranz. Diploma thesis, University of Leipzig (2010).

[9] Godsil, C. and Royle, G. Algebraic Graph Theory. 1st edition (Springer, New York, 2001).

[10] Schmidtchen, H., Thüne, M., and Behn, U. Randomly Evolving Idiotypic Networks I: Structural Properties and Architecture. In preparation.

[11] Schmidtchen, H. Architecture of randomly evolving networks. Diploma thesis, University of Leipzig (2006).

[12] Schmidtchen, H. and Behn, U. Randomly Evolving Idiotypic Networks II: A Mean Field Approach. In preparation.

[13] Kühn, A. Idiotypische Netzwerke: Mean-Field-Theorie mit Korrelationen. Diploma thesis, University of Leipzig (2010).

[14] McKay, B. D. Practical graph isomorphism. *Congressus Numerantium* **30**, 45–87 (1981).

[15] McKay, B. D., `http://cs.anu.edu.au/~bdm/nauty`. Accessed January 20th, 2011.

[16] Kang, C. and Kohler, H. A Novel Chimeric Antibody with Circular Network Characteristics: Autobody. *Annals of the New York Academy of Sciences* **475**(1 Autoimmunity), 114–122 (1986).

Acknowledgement

I would like to thank my supervisor, Prof. Ulrich Behn, for all the discussions about this work and for giving me the opportunity to follow my own ideas. Most of the time I worked on this thesis I spent at my working space in the Institute for Theoretical Physics. I am grateful to all the folks there for the great working atmosphere and for the great time I had working there. Lots of discussions helped me to stay passionate about the Idiotypic Networks topic. In this context I would like to thank Holger Schmidtchen and especially Andreas Kühn. I could not have completed this thesis without the moral support by my parents and my friends, especially Carolin Bode.