Chapter 1

Introduction

1.1 A Computable Universe

The study of cellular automata has attracted considerable interest in recent years, which stems from the fact that they are fundamentally simple in construction, and yet have the capacity for profoundly complex behaviour. Cellular automata may be described as *fully discrete* dynamical systems, the evolution of which is determined *locally* and *homogeneously*. Thus defined, they are inherently suitable for digital computer implementation. Indeed, the growing interest in cellular automata corresponds with the increasing sophistication and accessibility of computer technology. Equipped with a standard microcomputer and some basic programming skills, one has a 'laboratory' in which an extraordinary range of behaviour may be observed – including such things as self organisation and replication, soliton formation and interaction, and chaos.

It should be stressed that what makes cellular automata interesting is not just the complex behaviour which they generate, but the fact that their implementation on digital machines, by definition, is computationally both *efficient*[1] and *exact*. Normally, the computer implementation of mathematical models implies approximation. We have developed a language of continuum mathematics, and the job of a numerical analyst is to impose this language upon a discrete machine, which does an admirable

[1] Particularly when exploiting appropriate parallel architecture.

job, except that it loses the very small and balks at the very large. Therefore, having obtained a 'numerical solution' to a continuous model, the problem remains to determine whether the observed properties are indeed attributable to the mathematical model, or simply manifestations of discretisation and/or subsequent rounding errors.

When we enter the world of cellular automata, the computer is allowed to speak its own language. The word 'number' begins to lose its meaning: no longer is a bit of information aspiring to represent part of a quantity that may not even have a finite discrete representation – it simply represents itself. The model fully exploits the power of the machine, and the machine provides perfect implementation of the model.

At present, we have cellular automata machines which can iterate extremely complex, virtually random sequences of electronic 'events' at extraordinary speed, and we have absolute control over the machines. We may record or reproduce any sequence of events exactly, freeze any point in time, and make measurements without altering the state of the system - a *computable universe*. The analogy for the experimental scientist is a world where laboratory instruments have zero tolerance; where every phenomenon is visible, and every event is measurable without an uncertainty principle; where every experiment can be reproduced at the touch of a button, giving precisely the same results.

Such a universe permits a perfect marriage of experiment and theory. However, therein lies a major pitfall: it is so easy to observe, record, and hence precisely identify and characterise phenomena that there is a continual production of mathematical results from many and varied points of view - ranging from familiar methods in the analysis of dynamical systems to ones peculiar to cellular automata. Though it is hard to know which of these methods (if any) will prevail, it is not unreasonable to believe that fundamentally suitable approaches exist, and somehow elude us.

1.2 First Foot Forward

In the study of cellular automata, reference is made to two types of problems: *forward* and *inverse*. Forward problems are to determine the properties of a given cellular automaton (or class of cellular automata). Inverse problems are to find cellular automata which have a given set of properties. Supplementary to both, Gutowitz [6] proposes a third type: *lateral* problems, which consist of attempts to adapt existing methodologies to the study of cellular automata.

Accordingly, this thesis may be described as a *forward lateral* problem: to establish some framework in which to analyse and possibly *solve* nontrivial nonlinear cellular automata, and to use the traditional differential calculus as a rough 'procedural' guide.

What then is the 'procedure' of the differential calculus? To begin with, a student is introduced to the *elementary functions* as *solutions of linear differential equations*, and these form a basis for the varied analyses of more interesting nonlinear differential equations. The results of this thesis suggest that there is only one such elementary function necessary for the analysis of cellular automata: the *binary binomial coefficients*, which are precisely the *solution of the most elementary nontrivial linear cellular automaton*. From a thorough understanding of the properties of binary binomial coefficients, which may be achieved in a few theorems, we construct a calculus of cellular automata.

This thesis is motivated by the hypothesis that cellular automata should be easy to understand when viewed correctly, and it is up to the forward analyst to find the right point of view. Further, we are motivated to start looking at the simplest first, and to provide a formal framework whereby results are easily generalised to higher orders of complexity. The inverse problem is given little consideration, beyond that of the behavioural generality of the class of cellular automata which we propose to study. The belief is that the potentials and/or limitations for the practical implementation of cellular automata will be revealed incidentally; that is, only when we know what cellular automata *do (or don't do)* may we fully understand what they *can do for us*.

1.3 The Calculus of Binary Cellular Automata

The structure of this thesis may be summarised as follows:

Chapter 2: We refine the formulation and class (1-dimensional, binary, first order in time) of cellular automata which we propose to study in this thesis, and show that these cellular automata have behavioural generality up to all computable (1-dimensional) cellular automata

Chapter 3: We discuss, as a consequence of assuming the algebraic framework of the field F_2, an alternative polynomial rule representation which facilitates both analysis and algebraic classification of rules.

Chapter 4: We analyse the elementary (nontrivial) linear cellular automaton which generates the *binary binomial coefficients*, [2] comprising the *binomial sequences* which provide a basis for the analysis of binary cellular automata in general.

Chapter 5: We outline various methods for the analysis of (binary 1-dimensional) cellular automata based on the properties of the binomial sequences discussed in Chapter 4. These methods yield exact solutions of a narrow class of nonlinear cellular automata, and an explicit stability analysis relevant to cellular automata in general. Preliminary algebraic classification of rules is also achieved.

The methods discussed in Chapter 4 and implemented in Chapter 5 are analogous to the construction of the differential calculus: from the solution of a linear system, we achieve a basis for the analysis of nonlinear systems. The binomial sequences represent the 'elementary functions' in the context of cellular automata. Their behaviour and computation is straightforward they form natural basis for finite difference calculus, they are closed under element by element multiplication, and therefore provide a natural basis for the analysis of cellular automata. [3]

The methodology of this thesis is believed to be a substantial contribution to the study of cellular automata. Chapters 2 through 4 contain well known results in conjunction with some novel, however their application in Chapter 5 is, to my

[2] that is, Pascal's Triangle (mod 2).

[3] Interestingly enough, it appears that the understanding of a *single* linear cellular automaton is sufficient for the calculus of *all* binary 1-dimensional cellular automata which we propose to study, as the binomial sequences alone dominate their limiting behaviour in at least one frame of reference.

knowledge, entirely original, and forms the basis upon which this thesis should be judged.

1.4 Preliminaries

This thesis may be regarded as self-contained, insofar as very little prior knowledge is assumed. Though we operate within the algebraic framework of the finite field F_2, no field theory is required beyond the polynomial identity

$$(p(x))^2 = p(x^2).$$

Most combinatorial results are stated and proved as required, and number theoretical results are no more sophisticated than

$$n \mid 2^m \Leftrightarrow n = 2^{m'} \text{ for some } 0 \leq m' \leq m.$$

Indeed, the success of this thesis may well be measured by the simplicity and not the complexity of the mathematics which follows. We have attempted to adopt 'standard' notation wherever possible, but it should be pointed out that there are almost as many formalisms as there are cellular automata theorists. Much work needs to be done integrating the various results and achieving more uniform notation and conventions. It is hoped that the framework outlined in this thesis will, if only in a small part, further this cause.

1.5 Literature Review

This thesis does not represent a natural progression in an existing school of research – rather, the methodology presented here is constructed 'from the ground up' as it were – and for this reason it is difficult to provide a standard literature review. There will be occasional relevance to existing literature on the cellular automata forward problem [9, 12, 13, 15, 19, 23, 24]. However, these works may be viewed more as alternative approaches than as actual background to the material presented here.

Therefore, we will make brief reference to them only where such relevance occurs and not discuss them in any detail.

It would appear that this thesis actually bears stronger connection with the area of symbolic dynamics, central to which is the study of shift dynamical systems. To begin with, one-dimensional cellular automata have been precisely characterised as the endomorphisms of shift dynamical systems in the work of Curtis, Hedlund and Lydon [7]. From this research emerged the commuting block maps problem studied by Coven, Hedlund and Rhodes [3, 16, 17] which has been solved in special cases. Interestingly enough, the form of the block maps for which solution was obtained bears strong resemblance to the cellular automata for which results are achieved in Chapter 5 of this thesis.

Interest in symbolic dynamics continues to grow. A recent publication of Lind and Marcus [11] gives a comprehensive treatment to the subject with applications to coding theory. We will outline the important similarities and differences between the subject of this thesis and these works on symbolic dynamics in the conclusion to this thesis.

Chapter 2

1-Dimensional Cellular Automata

This thesis will be concerned primarily with the discussion of binary 1-dimensional cellular automata on infinite lattices. In this chapter, we consider the consequences of restricting the scope of our analysis, reaching the conclusion that the narrowly defined class of cellular automata which we propose to study are capable of precisely simulating a much larger class of cellular automata, including all those which have an explicit finite representation. We also find other classes with this property, indicating alternative but equivalent courses of study.

2.1 Rules and Alphabets

In general, a cellular automaton consists of a regular lattice of sites which take values from a finite *alphabet* of $k \geq 2$ symbols, usually represented by $\mathbb{Z}_k = \{0, 1, \ldots, k-1\}$. The discrete time evolution of each site is homogeneously defined by a *local rule*, which gives the value of each site at some point in time as a function of previous site values in some local neighbourhood of the site.

In the case of a 1-dimensional cellular automaton on an infinite lattice, we label each site by x_i, where the integer $i \in \mathbb{Z}$ is referred to as the *spatial index*. The local rule is written as a function $f : \mathbb{Z}_k^{2r+1} \to \mathbb{Z}_k$ such that, for all $i \in \mathbb{Z}$ and $t \in \mathbb{N}$ (the non-negative integers),

$$x_i^{t+1} = f(x_{i-r}^t, \ldots, x_i^t, \ldots, x_{i+r}^t), \tag{2.1}$$

where x_i^t denotes the value of site i at time t, and r is a non-negative integer referred to as the *range*[1] of the rule f, describing the size of the neighbourhood $(x_{i-r}, \ldots, x_i, \ldots, x_{i+r})$ of x_i upon which f operates.

We define the *state* of the cellular automaton lattice at time t to be the infinite[2] sequence $x^t = \{x_i^t\}_{i=-\infty}^{\infty} \in \mathbb{Z}_k^{\mathbb{Z}}$. Each local rule then defines a *global mapping* $F : \mathbb{Z}_k^{\mathbb{Z}} \to \mathbb{Z}_k^{\mathbb{Z}}$ by

$$(Fx^t)_i = x_i^{t+1} = f(x_{i-r}^t, \ldots, x_i^t, \ldots, x_{i+r}^t). \tag{2.2}$$

We may then describe the evolution of the cellular automaton globally by

$$x^t = F^t x^0,$$

where we refer to x^0 as the *initial state*, and F^t is the composition of t operations of F. Stated simply, the forward problem is to determine the behaviour of F^t from the properties of f.

2.1.1 Homogeneity and Spatial Relativity

To begin with, the homogeneity of the local rule has global consequences. Define the *spatial shift* operator $E : \mathbb{Z}_k^{\mathbb{Z}} \to \mathbb{Z}_k^{\mathbb{Z}}$ by $(Ex)_i = x_{i+1}$. It follows from equation (2.2) that F commutes with E, so F is invariant under spatial shifts; that is, for any integer n,

$$E^{-n} \circ F \circ E^n = F,$$

which we may interpret as *spatial relativity*. This allows a certain amount of flexibility in the formulation of the local rule. If we let $\hat{F} = E^n F$, then

$$\begin{aligned} (\hat{F}x^t)_i &= (E^n F x^t)_i \\ &= (Fx^t)_{i+n} \\ &= x_{i+n}^{t+1} \\ &= f(x_{i+n-r}^t \ldots x_{i+n+r}^t), \end{aligned}$$

[1] Sometimes *radius*.

[2] A finite lattice, otherwise referred to as periodic or cyclic lattice, is a special case of the infinite lattice where we have $x_i = x_{i+P}$ for all $i \in \mathbb{Z}$ and some positive integer P.

so we see that the shifted global mapping $\hat{F}$, which is equivalent to F, corresponds to the local rule

$$x_i^{t+1} = f(x_{i+n-r}^t \ldots x_{i+n+r}^t).$$

In particular, if we put $n = -r$ into the previous equation, and let $d = 2r$, which we refer to as the *degree*[3] of the local rule, we have

$$x_i^{t+1} = f(x_{i-d}^t, \ldots, x_i^t). \tag{2.3}$$

Throughout this thesis we will assume that *all local rules are written in the form* (2.3), and without loss of generality as discussed.

2.1.2 Causal Structure

One reason we adopt (2.3) over the more typical (2.1) is that the *causal structure* of (2.3) is simplified, which facilitates most of the formal analysis which follows. In general, we denote the set of site values upon which x_i^t depends (implicitly) by $D(x_i^t)$, the *domain of dependence* of x_i^t; similarly, the set of site values which depend (implicitly) upon x_i^t are denoted $R(x_i^t)$, the *range of influence* of x_i^t.

For a rule of the form (2.1)

$$\begin{aligned} D(x_i^t) &= \{x_{i-j}^{t-s} : s \geq 0, -rs \leq j \leq rs\}, \\ R(x_i^t) &= \{x_{i+j}^{t+s} : s \geq 0, -rs \leq j \leq rs\}; \end{aligned}$$

for a rule of the form (2.3),

$$\begin{aligned} D(x_i^t) &= \{x_{i-j}^{t-s} : 0 \leq j \leq ds\} \\ R(x_i^t) &= \{x_{i+j}^{t+s} : 0 \leq j \leq ds\}. \end{aligned}$$

The union of D and R is referred to as a *light cone*.

Referring to Figure 2.1, we see that for rules of the form (2.3), $D(x_i^t)$ is 'right justified', and $R(x_i^t)$ is 'left justified', so information (if it moves at all) flows exclusively

[3] Alternatively *diameter*.

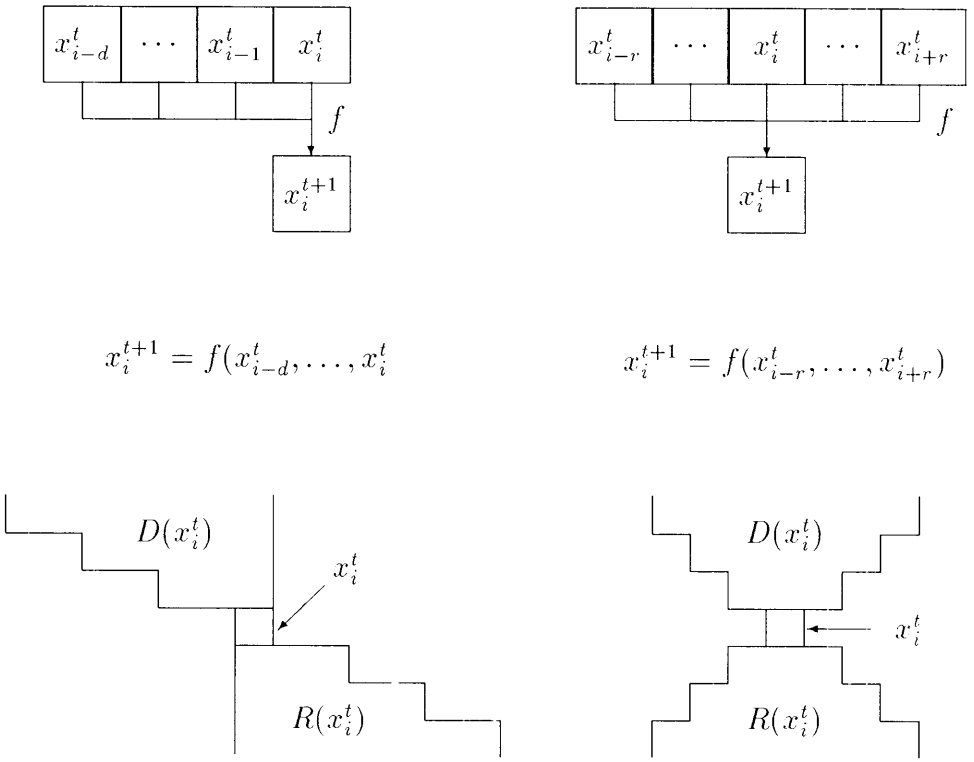


Figure 2.1: Causal structure of the alternative forms of the local rule. Here, and in all diagrams to follow, the spatial index i increases from left to right, and time t from top to bottom.

from left to right in the evolution of the cellular automaton. This has significant advantages in following discussions where we consider generally rules of arbitrary degree; by increasing d, we need only augment D and R to the left and right respectively.

One more important point to be made in view of spatial relativity is the question "What happens to site i as t evolves?" is too particular. It makes an arbitrary identification between x_i^{t+1} and only one of the preceding site values upon which it is dependent. Spatial relativity implies there is more than one frame of reference worth considering when analysing temporal behaviour, which motivates the following definition.

For an integer[4] v such that $0 \leq v \leq d$, we define[5] the *temporal sequence of velocity* v, denoted $T_i(v)$, to be the sequence of site values

$$T_i(v) = \{x_{i+vt}^t\}_{t=0}^{\infty} \in \mathbb{Z}_k^{\mathbb{N}}.$$

We may extend the temporal sequences in the $-t$ direction if the global mapping F is invertible, as will be discussed later.

The temporal sequences of a given velocity constitute a frame of reference in which to look for solutions, or more generally, to characterise the behaviour of a particular cellular automaton. In this thesis, we consider temporal sequences of velocity $v = 0$ (alternatively $v = d$) which lie on the boundaries of light cones, and therefore represent the simplest causal 'paths' between site values. For rules of the form (2.3), we have $T_i(0) = \{x_i^t\}_{t=0}^{\infty}$, which which we may write simply as $T_i(0) = x_i$ for the sake of formal simplicity; that is, we now regard x^t as a spatially infinite sequence, and x_i as a temporally infinite sequence.

2.2 Rule and Alphabet Reduction

Equation (2.3) is said to be a *first order* rule in time; that is, site values at time $t+1$ depend explicitly on site values at time t only. To be more general, we could define

[4]More generally, we could consider rational numbers: for $0 \leq v = \frac{a}{b} \leq d$, where a and b are relatively prime, we have $T_i(v) = \{x_{i+at}^{bt}\}_{t=0}^{\infty}$.

[5]Generalization of [9], which discusses aperiodicity of the sequences $T_i(1)$ for binary $d = 2$ cellular automata.

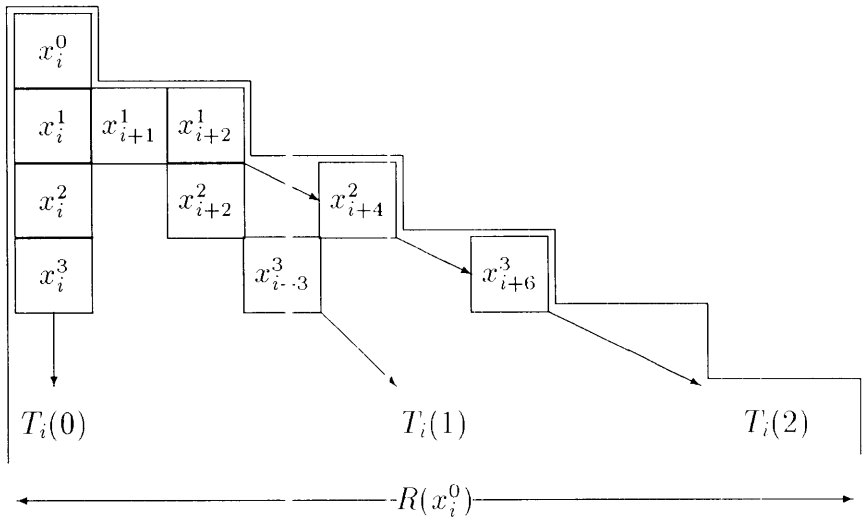


Figure 2.2: Temporal sequences of a 1-dimensional $d = 2$ cellular automaton, beginning at the site x_i at time $t = 0$. $T_i(v) \subseteq R(x_i^0)$ for all $0 \leq v \leq 2$; $T_i(0)$ and $T_i(2)$ define the boundaries of $R(x_i^0)$

a rule $f : (\mathbb{Z}_k)^{m(d+1)} \to \mathbb{Z}_k$ of (temporal) order m and (spatial) degree d by

$$x_i^{t+1} = f\begin{pmatrix} x_{i-d}^{t-m+1} & \cdots & x_i^{t-m+1} \\ \vdots & \ddots & \vdots \\ x_{i-d}^{t} & \cdots & x_i^{t} \end{pmatrix}, \tag{2.4}$$

where now our cellular automaton state consists of the m consecutive sequences

$$\bar{x}^t = (x^t, x^{t-1}, \ldots, x^{t-m+1}).$$

Hence, the global mapping F induced by f is now defined by

$$x^{t+1} = F\bar{x}^t.$$

Cellular automata of higher orders have been studied, however it is easy to show that this is a matter of formal convenience and not one of generality. Both the order m and degree d may be reduced to unity by extending the size of the alphabet as follows.

To begin with, let $\bar{x}_i^t = (x_i^t, x_i^{t-1}, \ldots, x_i^{t-m+1}) \in \mathbb{Z}_k^m$, be a block of m consecutive site values (in time), and then a rule f of the form (2.4) may be reduced to the first order (in time),

$$\bar{x}_i^{t+1} = \bar{f}(\bar{x}_{i-d}^t, \ldots, \bar{x}_i^t),$$

where $\bar{f} : (\mathbb{Z}_k^m)^{d+1} \to \mathbb{Z}_k^m$ is defined in terms of f by

$$\bar{f}(\bar{x}_{i-d}^t, \ldots, \bar{x}_i^t) = (f(\bar{x}_{i-d}^t, \ldots, \bar{x}_i^t), x_i^t, \ldots, x_i^{t-m+2}).$$

Now introduce $y_i^t = (\bar{x}_{id}^t, \bar{x}_{id+1}^t, \ldots, \bar{x}_{id+d-1}^t) \in \mathbb{Z}_k^{md}$, and by a similar argument, we may write

$$y_i^{t+1} = g(y_{i-1}^t, y_i^t), \tag{2.5}$$

for some function $g : (\mathbb{Z}_k^{md})^2 \to \mathbb{Z}_k^{md}$. Thus, we have reduced a cellular automaton of arbitrary order m and degree d on an alphabet $\mathbb{Z}_k$ to one of *first order* and *first degree*, by extending the alphabet [6] to $\mathbb{Z}_k^{md}$.

We will now turn this argument around and show that in general, a cellular automaton of the form (2.4) may be reduced to one of the first order on the alphabet $\mathbb{Z}_2$, this time by extending the degree d. To begin with, we may assume by the above discussion that the local rule has been reduced to the form (2.5),

$$y_i^{t+1} = g(y_{i-1}^t, y_i^t),$$

and we have some finite alphabet[7] $\mathbb{Z}_k \subseteq \mathbb{Z}_{2^n}$, where $n = \lceil \log k \rceil$ is the least integer exceeding $\log k$.

We may regard each site value as ordered set of independent binary variables,

$$y_i^t = (y_i^t(0), y_i^t(1), \ldots, y_i^t(n-1)),$$

where each $y_i^t(j) \in \mathbb{Z}_2$ (essentially the binary representation). Accordingly, we may separate the function g into n components $g_0, g_1, \ldots, g_{n-1} : (\mathbb{Z}_2)^{2n} \to \mathbb{Z}_2$ such that

$$\begin{aligned} y_i^{t+1}(0) &= g_0(y_{i-1}^t, y_i^t) \\ y_i^{t+1}(1) &= g_1(y_{i-1}^t, y_i^t) \end{aligned}$$

[6] This particular reduction may be regarded as well known [13].

[7] Here we assume no algebraic structure on either $\mathbb{Z}_k$ or $\mathbb{Z}_{2^n}$.

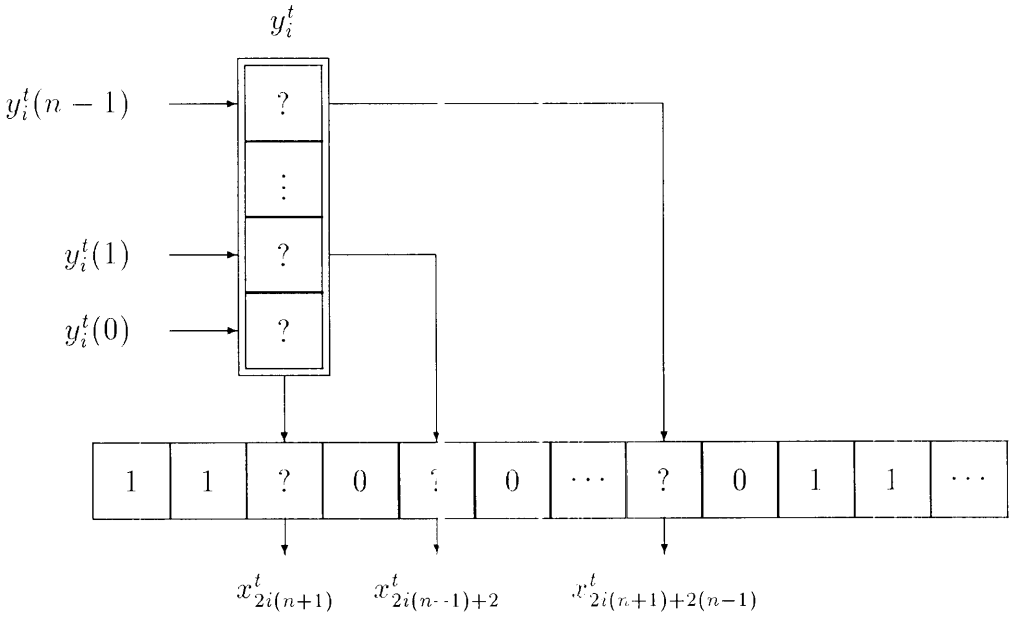


Figure 2.3: Reducing a first degree first order cellular automaton on $\mathbb{Z}_k \subseteq \mathbb{Z}_{2^n}$ to a first order cellular automaton on $\mathbb{Z}_2$ of degree $6n-4$.

$$\vdots$$

$$y_i^{t+}\ (n-1) \;=\; g_{n-1}(y_{i-1}^t, y_i^t).$$

Now introduce sites $x_i^t \in \mathbb{Z}_2$ defined as follows (refer to Figure 2.3):

$$x^t_{2i(n+1)+j} = \begin{cases} y_i^t(k) & \text{if } 0 \le j = 2k < 2n \\ 0 & \text{if } 0 \le j = 2k+1 < 2n \\ 1 & \text{if } j = 2n \text{ or } j = 2n+1. \end{cases}$$

The pairs of 1's act as a markers which allow us to locate the augmented representation of the original site values, and hence define a rule which performs the same

computation as the original with the following algorithm:

FOR ALL $-\infty$ <i< ∞
  FOR $j = 0$ TO $2n + 1$
    IF $(x^t_{i-j-3}, x^t_{i-j-2}, x_{-j-1}) = (0, 1, 1)$ THEN
      IF j is odd OR $j = 2n$ THEN LET $x^{t+1}_i = x^t_i$
      ELSE LET $y_0 = (x^t_{i-2(n+1)-j}, x^t_{i-2(n+1)-j+2}, \ldots, x^t_{i-j-4})$
        LET $y_1 = (x^t_{i-j}, x^t_{i-j+2}, \ldots, x^t_{i-j+2(n-1)})$
        LET $x^{t+1}_i = g_{j/2}(y_0, y_1)$

The last line invokes the components of the function g defined above. It should be noted that the function described by this algorithm is homogeneous; it does not depend on i to make any 'decisions', but upon j which is calculated locally by 'finding' the configuration 011, so the function takes only local site values as arguments. Evaluation of x^{t+1}_i requires knowledge of (at most) $x^t_{i-(4n-2)}$ on the left and $x^t_{i+(2n-2)}$ on the right, so we may write

$$x^{t+1}_i = f(x^t_{i-(4n-2)}, \ldots, x^t_{i+(2n-2)}),$$

hence a cellular automaton of first order and degree $6n - 4$ on $\mathbb{Z}_2$.

Finally, an analogous argument may be used to show that a cellular automaton of the form (2.4) may be reduced to the first degree on the alphabet $\mathbb{Z}_2$, by increasing the order m. Suffice it to say that we augment the site values in the t direction, separated by zeroes and marked by pairs of ones as above (rotate Figure 2.3 90° clockwise), and adjust the algorithm. Summarising these results, we have the following:

Proposition 2.1 Any 1-dimensional cellular automaton may be reduced to one of degree d and order m on an alphabet $\mathbb{Z}_k$ such that two of the constants $\{d, m, \log k\}$ are equal to 1.

An interpretation of this proposition is that somehow in the definition of cellular automata – being totally discrete, and locally determined on a finite alphabet – we

have composed space (degree), time (order) and information (alphabet) from the same 'fabric'. We may reduce any two with no loss of generality, so long as we allow the extension of the third. There need be no practical reason for reduction of a cellular automaton; each may tend to have a natural representation for the purpose of analysis. Rather, questions of generality are of great importance when faced with the open horizon of the forward problem. To this end, Proposition 2.1 strongly motivates various courses of study, one of which is pursued in this thesis.

All begin with the *elementary*[8] cellular automata, which are defined to by

$$\{d, m, \log k\} = \{1, 1, 1\}.$$

If we refer to $\{d, m, \log k\}$ as the *class* of a cellular automaton, there are in general $k^{k^{m(d+1)}}$ local rules for a given class, some of which would reduce to lower classes. Of the 16 elementary rules, most are *trivial*, by which we mean they reduce to a class $\{d, m, \log k\}$ where at least one of $d, m, \log k$ is equal to zero.[9] From the elementary we may proceed in three directions, which by Proposition 2.1 are equivalent. That is, increment d, m or k, and eventually you will end up with the same classes of behaviour.

Moore [13] pursues the direction k, considering rules of the form (2.5). This means that a local rule may be viewed as a binary operator, having various algebraic structures as k is increased, for which some lend themselves to explicit complexity analysis. The advantage of this course of study is that the causal structure remains constant and elementary ($d = n = 1$).

We pursue the direction d, which means that we may work within the algebraic framework of the field F_2. We begin with an analysis of the *unique elementary non-trivial linear cellular automaton* iterated from the *simplest initial state*, from which is built a difference calculus peculiar to the analysis of binary cellular automata. Working within a field structure would appear necessary to develop such a calculus, and furthermore, there are particular advantages working with F_2 which we will point out in the next chapter.

[8]Here departing from the more common definition of elementary as binary first order $d = 2$ cellular automata, due to Wolfram.

[9]The case $m = 0$ we may interpret as a cellular automaton which does not evolve with time.

The methods discussed in this thesis would in principle apply to the remaining direction m, or more generally all cellular automata of class $\{d, m, 1\}$, indicating a course for further study beyond the scope of this thesis, with a particular view to the inverse problem of modelling.[10]

2.2.1 Dimension and Infinity

The discussions in this section may be generalised to cellular automata of higher dimensions, as indeed most results presented in this thesis. However, it would appear that dimension itself cannot be reduced in the same fashion as the degree, order or alphabet size, so long as we are working with spatially infinite lattices with non-periodic states.

Restriction to finite lattices has considerable consequences: consider the state of a finite dimensional cellular automata on a finite lattice, which could be viewed as a single site taking values from some 'huge' but finite set $\mathbb{Z}_k^n$, where n is the total number sites in the lattice. From there our cellular automata collapses to a first order recurrence relation on a finite set – by our definition, trivial. Therefore, if we want to consider the behaviour which cellular automata in general are capable of generating, we don't want to throw away the last infinity just yet. Proposition 2.1 would suggest that one infinity is sufficient for arbitrary precision of modelling in a cellular automata framework, since the accuracy whereby we discretise space, time or alphabet are demonstrably interchangeable.

2.3 Classification of States

So far in this chapter, we have considered various restrictions that may be placed on either the form of the local rule or the size of the alphabet without loss of generality to the 1-dimensional forward problem. Here we consider the consequences of restricting the states of the cellular automata, which may be viewed as the imposition of *boundary conditions*. In particular, we propose to look at states which may be explicitly

[10] We stumble across, amongst other things, a host of discrete nonlinear 'wave equations' as soon as we step up to $\{d, m, k\} = \{1, 2, 1\}, \{2, 2, 1\}, \ldots,$.

represented on a finite machine, which allows for further refinement of the forward problem before we proceed to construct a difference calculus.

For this discussion, the classification of a cellular automaton requires not only specification of the degree and order of the local rule and the size of the alphabet, but also the *space of sequences* which we allow as states of the cellular automaton. This motivates the following formal definition:

Definition: In general, a 1-dimensional cellular automaton $(f, \mathcal{S})$ of class $\{d, m, \log k\}$[11] consists of a local rule $f : \mathbb{Z}_k^{m\ d+1)} \to \mathbb{Z}_k$ of degree d and order m, and a set of *allowable sequences* $\mathcal{S} \subseteq \mathbb{Z}_k^{\mathbb{Z}}$, such that, given any state $(x^t, \ldots, x^{t-m+1})$ where each $x^t, \ldots, x^{t-m+1} \in \mathcal{S}$, then x^{t+1} defined by (2.4)

$$x_i^{t+1} = f\begin{pmatrix} x_{i-d}^{t-m+1} & \cdots & x_i^{t-m+1} \\ \vdots & \ddots & \vdots \\ x_{i-d}^{t} & \cdots & x_i^{t} \end{pmatrix},$$

satisfies $x^{t+1} \in \mathcal{S}$. That is, so long as $\mathcal{S}$ is closed under the global mapping F defined by f, we've got ourselves a cellular automaton.

2.3.1 Ultimately Periodic States

We propose to restrict our cellular automata states to *ultimately periodic* sequences, denoted $\mathcal{U}_k \subset \mathbb{Z}_k^{\mathbb{Z}}$. The motivation for this restriction is that the ultimately periodic sequences represent the infinite sequences which may be given an explicit representation on a finite machine. [12] We define the set by $x \in \mathcal{U}_k$ if there exist (least) constants $L, R > 0$ and $Q \geq 0$ such that

$$\begin{aligned} x_{i-L} &= x_i, \qquad \text{for all } i < 0, \text{ and} \\ x_{i+R} &= x_i, \qquad \text{for all } i \geq Q, \end{aligned}$$

where we refer to L, R and Q as the *left period, right period* and *preperiod* of the sequence x respectively, sometimes written $L(x)$, $R(x)$ and $Q(x)$. Further, to simplify

[11] Where we require d and m are non-negative integers, and k a positive integer.

[12] Being the first and last instance where we lose generality amongst 1-dimensional cellular automata on infinite lattices. We may regard the alternative as computationally intractable, though nonetheless interesting.

discussions on the periodic components of more than one sequence, we define for $x, y, \ldots, z \in \mathcal{U}_k$,

$$\begin{aligned} L(x, y, \ldots, z) &= \operatorname{lcm}(L(x), L(y), \ldots, L(z)) \\ R(x, y, \ldots, z) &= \operatorname{lcm}(R(x), R(y), \ldots, R(z)), \end{aligned}$$

and for the preperiodic component,

$$Q(x, y, \ldots, z) = \max(Q(x), Q(y), \ldots, Q(z)).$$

The spatial relativity discussed in the beginning of this chapter allows us to be specific in the above definition about the placement of the preperiodic component of the states:

Proposition 2.2 Let f be a local rule of the form (2.4), then $(f, \mathcal{U}_k)$ is a cellular automaton.

In other words, if the preperiodic component of the states is placed to the right of the origin, it stays there. The proof is straightforward, and only outlined here. Let $L' = L(x^{t-m+1}, \ldots, x^t)$, $R' = R(x^{t-m+1}, \ldots, x^t)$, and $Q' = Q(x^{t-m+1}, \ldots, x^t) + d$. Then $x_i^{t+1} = x_{i-L'}^{t+1}$ for all $i < 0$, and $x_i^{t+1} = x_{i+R'}^{t+1}$ for all $i \geq Q'$, both following from (2.4).

We will also refer to the following subsets of $\mathcal{U}_k$. The *finite* sequences, denoted $\mathcal{F}_k \subset \mathcal{U}_k$ are defined by $x \in \mathcal{F}_k$ if there exists a (least) constant $Q \geq 0$ such that

$$x_i = 0, \qquad \text{for all } i < 0 \text{ and } i \geq Q.$$

The *semi-infinite ultimately periodic* sequences, denoted $\mathcal{F}_k^- \subset \mathcal{U}_k$ are defined by $x \in \mathcal{F}_k^-$ if $x \in \mathcal{U}_k$ and

$$x_i = 0, \qquad \text{for all } i < 0.$$

The *periodic* sequences, denoted $\mathcal{P}_k \subset \mathcal{U}_k$ are defined by $x \in \mathcal{P}_k$ if there exists a (least) constant $P > 0$ such that

$$x_{i+P} = x_i \qquad \text{for all } i \in \mathbb{Z},$$

where we refer to the constant P as the *period* of the sequence x. As for L and R, we define for $x, y, \ldots, z \in \mathcal{P}_k$,

$$P(x, y, \ldots, z) = \text{lcm}(P(x), P(y), \ldots, P(z)).$$

It will be useful to discuss various subsets of the periodic sequences. For a positive integer n, we define

$$\mathcal{T}_k^n = \{x \in \mathcal{P}_k : P(x) \mid n\}$$

to be the set of all periodic sequences with period that divides n. In particular, we discuss extensively $\mathcal{P}^{2^m}$ in the context of binary cellular automata. We also write

$$\mathcal{P}^{2^{\mathbb{N}}} = \bigcup_{m=0}^{\infty} \mathcal{P}^{2^m}.$$

We now have

Proposition 2.3 Let f be a local rule of the form (2.4), then $(f, \mathcal{P}_k)$ is a cellular automaton. Furthermore, for all integers $n > 0$, $(f, \mathcal{P}_k^n)$ are cellular automata.

The proof is straightforward along the lines of the previous proposition. We make the observation here that all of the above sets of sequences are vector spaces on a field alphabet, in which case $\mathcal{U}_k = \mathcal{P}_k \oplus \mathcal{F}_k^-$. We will allow the natural restriction of these definitions to apply to *semi-infinite* sequences $\mathbb{Z}_k^{\mathbb{N}}$. When discussing binary sequences, we will omit the subscript $k = 2$.

2.3.2 Simulation

Throughout this chapter we have been using the term 'reduction' without having defined exactly what is meant by the term, and relying on an analogy with 'reduction of order' of differential equations. We now propose a mathematical definition of the term 'simulation' in a cellular automata framework, which includes the various 'reductions' performed in Section 2.2, and also takes into account the allowable sequences of the cellular automaton being simulated.

The very purpose of constructing a calculus of cellular automata is to answer questions about *long term* behaviour. For example, we may look for closed form

solutions (of the nature "What are the values of site i for *all* t ?"), and failing this, to characterise limiting behaviour ("What happens to site i as $t \to \infty$?"). Questions about the short term (of the nature "What is the value of site i *at some particular point in time* t ?") are all computationally finite, and therefore may be answered by simply iterating the cellular automaton on some finite machine which is 'large enough'. In view of this, the definition we propose is to make precise the idea that one cellular automaton may precisely and *ultimately* simulate the behaviour of another.

We say that $(f_2, \mathcal{S}_2)$ of class $\{d_2, m_2, \log k_2\}$ *simulates* $(f_1, \mathcal{S}_1)$ of class $\{d_1, m_1, \log k_1\}$ if given any initial state $(x^0, \ldots, x^{m_1-1}) \in (\mathcal{S}_1)^{m_1}$, there exists a constant $b > 0$, an initial state $(y^0, \ldots, y^{m_2-1}) \in (\mathcal{S}_2)^{m_2}$, a function π^t, and sequences $s^t \in \mathbb{Z}^{\mathbb{Z}}$ and $j_i^t \in \mathbb{Z}^{\mathbb{Z}^2}$ such that for all $i \in \mathbb{Z}$ and $t \geq b$,

$$x_i^t = \pi^t(y_{j_i^t - b}^{s^t}, y_{j_i^t - b+1}^{s^t}, \ldots, y_{j_i^t}^{s^t}),$$

where s_t and j_i^t satisfy[13]

$$\begin{aligned} 0 \leq s^{t+1} - s^t &\leq b, \\ |j_{i+1}^t - j_i^t| &\leq b, \\ |j_i^{t+1} - j_i^t| &\leq b. \end{aligned}$$

This would seem a rather complex definition, but the idea is reasonably simple: the states of the simulated may be reconstructed from the states of the simulator by some process, the complexity of which is *bounded for all time*. In this definition, everything is 'bounded' by the constant b. The degree of π^t remains constant, (though the function itself may change with time). The conditions imposed on the sequences s^t and j_i^t simply require that the information needed to reconstruct adjacent sites (in space or time) in $(f_1, \mathcal{S}_1)$ remains 'cohesive' in the evolution of $(f_2, \mathcal{S}_2)$ (refer to Figure 2.4). More precisely, any neighbourhood $\{x_{i\pm n}^{t\pm n}\}$ may be reconstructed from the neighbourhood $\{y_{j_i^t \pm b(n+1)}^{s^t \pm bn}\}$, which may 'move around' but does not grow with time.

[13]It might be required that s^t, j_i^t and π^t are computable in some sense, which would indicate scope for refinement of this definition. For the purposes of proving Proposition 2.5, it suffices that π^t, $\Delta_t s^t$, $\Delta_t j_i^t$, and $\Delta_i j_i^t$ are ultimately periodic.

One consequence of this definition is transitivity: if $(f_3, \mathcal{S}_3)$ simulates $(f_2, \mathcal{S}_2)$ simulates $(f_1, \mathcal{S}_1)$, then $(f_3, \mathcal{S}_3)$ simulates $(f_1, \mathcal{S}_1)$. We say that two cellular automata are *equivalent* if they simulate each other.

2.3.3 The Refined Forward Problem

Restriction to ultimately periodic states allows for further refinement of the forward problem. We say a function f is *non-generative* if

$$f(0, 0, \ldots, 0) = 0,$$

and generative otherwise. Firstly, we have the following:

Proposition 2.4 Let f be a local rule of the form (2.4), then $(f, \mathcal{F}_k^-)$ and $(f, \mathcal{F}_k)$ are cellular automata if and only if f is non-generative.

We propose to study only binary non-generative $(f, \mathcal{F}^-)$ in this thesis. These cellular automata are capable of precisely simulating the behaviour of all computable 1-dimensional cellular automata in general, as stated in the following proposition, which is the most important result of this chapter:

Proposition 2.5 In general, a cellular automaton $(f_1, \mathcal{U}_{k_1})$ of class $\{d_1, m_1, \log k_1\}$ may be simulated by a cellular automaton $(f_2, \mathcal{F}_{k_2}^-)$ of class $\{d_2, m_2, \log k_2\}$ such that f_2 is non-generative, and two of the constants $\{d_2, m_2, \log k_2\}$ are equal to 1.

The proof by construction is quite involved, and relies on the methods of reduction discussed in the previous section, where we must give special attention to the consequences of reduction upon the states of the cellular automata. We give an outline here. To begin with, suppose that we have some general $(f_1, \mathcal{U}_{k_1})$ of class $\{d_1, m_1, \log k_1\}$. Reduce f_1 to a rule g of first order and first degree; that is, of the form (2.5). It may be verified that ultimately periodic states remain ultimately periodic under such reduction. Hence we have $(g, \mathcal{U}_k)$, of class $\{1, 1, \log k\}$ for some k, which simulates $(f_1, \mathcal{U}_{k_1})$.

Now, suppose that the states of $(g, \mathcal{U}_k)$ have left period L; the region to the left of the origin must be (temporally) ultimately periodic, with period T say. We may now

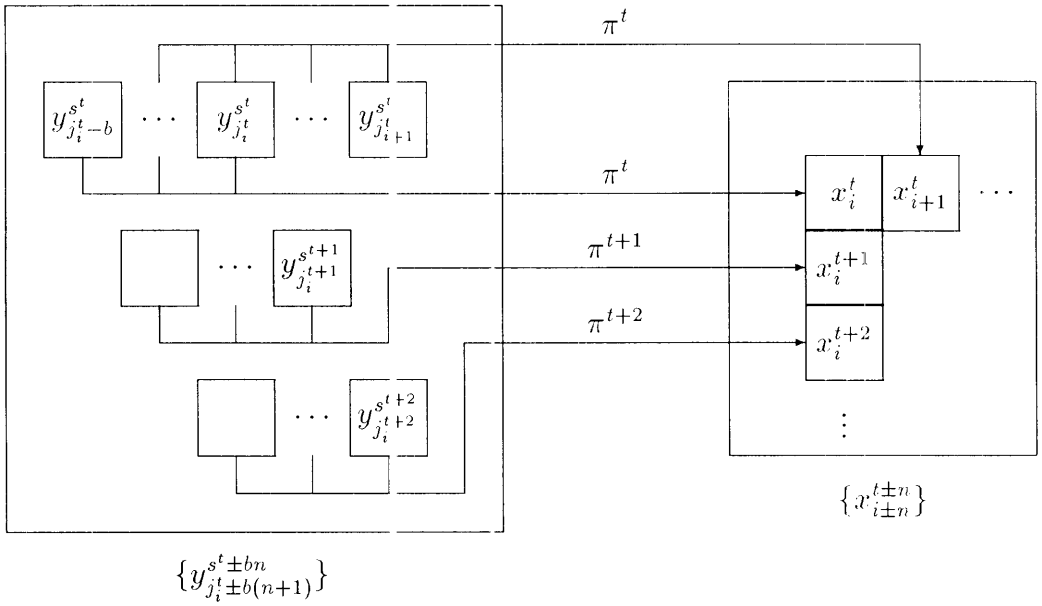


Figure 2.4: Simulation of one cellular automaton with another. We require that the adjacency in the simulated x_i^t is reflected by 'cohesion' in the evolution of the simulator y_j^s.

extend our site values to be (spatial) blocks of length L, and consider the T^{th} order action of g to be a rule $\hat{g} : (\mathbb{Z}_k)^n \to \mathbb{Z}_{k^L}$ where $n = \lceil \frac{\max(T,L)}{L} \rceil + 1$, which maps the (period 1) sequence of some 'large' site value to the left of the origin into itself. We may re-shuffle the site values[14] and call this one '0'; that is, we may simulate $(g, \mathcal{U}_k)$ with $(\hat{g}, \mathcal{F}^-_{k^{nL}})$, where $\hat{g}$ is non-generative.

Now it remains to be shown that any non-generative $(\hat{g}, \mathcal{F}^-_k)$ may be simulated with non-generative $(f_2, \mathcal{F}^-_{k_2})$, of class $\{d_2, 1, 1\}$, $\{1, m_2, 1\}$ or $\{1, 1, \log k_2\}$. The last case is straightforward – reduce $\hat{g}$ to the first order and first degree (the 0's to the left of the origin remain 0 under such reduction). We outline the case $\{d_2, 1, 1\}$ here, and the case $\{1, m_2, 1\}$ is similar. As in the previous section, consider the binary representation of $x \in \mathbb{Z}_k$, augmented horizontally with pairs of 1's acting as markers, except now omit the markers to the left of the origin. The function (described previously by an algorithm) now must be altered to give a value of 0 in the case where no markers are 'found'. Hence, a non-generative binary cellular automaton $(f_2, \mathcal{F}^-)$ of the first order and some finite degree.

2.4 Summary

The purpose of this chapter is to demonstrate that the cellular automata which we propose to study in this thesis namely non-generative $(f, \mathcal{F}^-)$ of class $\{d, 1, 1\}$, are capable of precisely simulating all (generative or non-generative) $(g, \mathcal{U}_k)$. The consideration here has been purely one of behavioural (alternatively computational) generality, and certainly not one of efficiency of methods of reduction or simulation.

Just briefly, there would seem to be some scope for further research into the relationship between d, m, and $\log k$. We might propose a measure of the complexity of a cellular automaton $(f, \mathcal{S})$ to be the minimum of the product $dm \log k$ taken over all cellular automata which simulate $(f, \mathcal{S})$, having seen that this product is 0 in the case of trivial cellular automata and 1 for the elementary cellular automata.

[14] Since we assume no particular algebraic structure on the alphabets at this stage.

Chapter 3

Representation of Binary Rules

As stated in the previous chapter, we have restricted the scope of this thesis to the discussion of non-generative ($f \ \mathcal{F}^-$) of class $\{d, 1, 1\}$, where we assume the local rule is written in the form (2.3),

$$x_i^{t+1} = f(x_{i-d}^t, \ldots, x_i^t).$$

In this chapter we show there are favourable consequences in assuming the algebra of the field F_2 for the purpose of rule representation. In particular, we may represent a rule as a polynomial in binary variables, which represents a natural form for analysis. We compare the polynomial representation with the standard rule table representation, and find a useful duality which, significantly, does not generalise to larger finite fields. This would suggest a natural framework in which to construct a difference calculus of cellular automata.[1]

The following is presented exclusively within the framework of the field F_2. We will give brief mention of the extent to which discussions generalise to F_p, $F_{q=p^m}$ ($p > 2$ prime) or $\mathbb{Z}_k$ in that order. [2]

[1] Particularly in view of the behavioural generality of binary cellular automata discussed in Chapter 2

[2] Hereafter assuming the algebra of a commutative ring for $\mathbb{Z}_k$.

3.1 Rule Representation and Enumeration

3.1.1 Rule Tables and Enumeration

(Here the discussion generalises to $\mathbb{Z}_k$.) A local rule f of degree d is typically specified by a *rule table* [3] which lists $f(\cdot)$ for each possible *configuration* of the variables $\{\bar{x} = (x_d, \ldots, x_0) : x_i \in F_2\}$. The configurations and hence the rules may be enumerated as follows [25].

Firstly, we consider each configuration $\bar{x} \in F_2^{d+1}$ as the binary representation of an integer, denoted $[\bar{x}]$ and defined by

$$\begin{aligned} [\bar{x}] &= [x_d \ldots x_0] \\ &= \sum_{i=0}^{d} x_i 2^i, \end{aligned}$$

taking values $0 \leq [\bar{x}] \leq 2^{d+1} - 1$. For simplicity, let $N = 2^{d+1}$ be the total number of configurations, and we may write $[\bar{x}] \in \mathbb{Z}_N$.

For a rule f of degree d, let $f_{[\bar{x}]} = f(\bar{x})$ for each configuration $\bar{x}$. We define the *(Wolfram) rule number*, denoted $[f]^W \in \mathbb{Z}_{2^N}$, by

$$\begin{aligned} [f]^W &= [f_{N-1} f_{N-2} \cdots f_0] \\ &= \sum_{n=0}^{N-1} f_n 2^n. \end{aligned} \tag{3.1}$$

3.1.2 Polynomial Form and Enumeration

(Generalises to F_q.) A local rule $f : F_2^{d+1} \to F_2$ of degree d may be represented as a polynomial in $d+1$ binary variables,

$$f(\bar{x}) = \sum_{n_d=0}^{1} \cdots \sum_{n_1=0}^{1} \sum_{n_0=0}^{1} a_{n_d \cdots n_1 n_0} (x_d)^{n_d} \cdots (x_1)^{n_1} (x_0)^{n_0},$$

where each of the coefficients $a_{n_d \cdots n_1 n_0} \in F_2$. As for the configuration $\bar{x}$, we may group the indices $\bar{n} = (n_d, \ldots, n_1, n_0)$ and consider $n = [\bar{n}] \in \mathbb{Z}_N$ as the binary

[3] Alternatively *lookup table*.

representation of an integer. We may simplify the previous equation to become

$$f(\bar{x}) = \sum_{n=0}^{N-1} a_n (x_d)^{n_d} \cdots (x_1)^{n_1} (x_0)^{n_0}. \tag{3.2}$$

We may enumerate the polynomial form using the coefficients a_n, in precisely the same manner as for the rule table, where we used the function values f_n. Define the *polynomial number* of f, denoted $[f]^P \in \mathbb{Z}_{2^N}$, by

$$\begin{aligned} [f]^P &= [a_{N-1} a_{N-2} \cdots a_0] \\ &= \sum_{n=0}^{N-1} a_n 2^n. \end{aligned} \tag{3.3}$$

Both representations have their place for the purpose of analysis. On the one hand, the rule table may become prohibitively large in situations where the polynomial form (3.2) is reasonably simple; on the other hand, a 'sparse' rule table consisting of only a few nonzero values may not have such a simple polynomial representation. However, wherever feasible, the polynomial form presents a useful basis for rule definition and analysis.

3.1.3 Polation

(Generalises to F_q.) The transformation from polynomial form to rule table is straightforward. The inverse transformation, that is construction of the polynomial form from the rule table, which we refer to as *polation*,[4] is more interesting. For f of degree $d+1$, we may write

$$f(\bar{x}) = \sum_{\bar{\alpha} \in F_2^{d+1}} f(\bar{\alpha})(x_d + \alpha_d + 1)(x_{d-1} + \alpha_{d-1} + 1) \cdots (x_0 + \alpha_0 + 1), \tag{3.4}$$

which is essentially the Lagrange interpolation formula. [5] This formula represents a reasonable amount of work for larger values of d, but will shortly be simplified by

[4] May be considered as alternatively interpolation or extrapolation in this context.

[5] For $f : F_q^{d+1} \to F_q$,

$$f(\bar{x}) = (-1)^{d+1} \sum_{\bar{\alpha} \in F_q^{d+1}} f(\bar{\alpha}) \prod_{\bar{\beta} \in F_q^{d+1} \setminus \bar{\alpha}} (x_d - \beta_d)(x_{d-1} - \beta_{d-1}) \cdots (x_0 - \beta_0).$$

viewing polation as a transformation between polynomial/Wolfram rule numbers.

Example 3.1 The (binary) $d = 1$ rule defined by the polynomial

$$g(yz) = yz$$

(we will generally omit commas in function declaration hereafter) has polynomial number $[g]^P = 2^{[11]} = 2^3 = 8$. We may construct the rule table:

$[yz]$	[11]	[10]	[01]	[00]
$g(yz)$	1	0	0	0

The second row may be read as the binary expansion of the rule number, $[g]^W = [1000] = 8$.

Now consider the $d = 2$ rule defined by the rule number $[f]^W = 136 = [10001000]$, with rule table:

$[xyz]$	[111]	[110]	[101]	[100]	[011]	[010]	[001]	[000]
$f(xyz)$	1	0	0	0	1	0	0	0

Using (3.4) we may calculate

$$\begin{aligned} f(xyz) &= f(011)(x+1)(y+0)(z+0) + f(111)(x+0)(y+0)(z+0) \\ &= yz, \end{aligned}$$

which is implicitly the same rule as g – that is, $f(xyz) = g(yz)$ – so the function $f(xyz)$ is independent of x. The equality of the polynomial numbers ($[f]^P = [g]^P = 8$) implies this independence.

This example points out one advantage of the polynomial enumeration scheme[6] over the more commonly used Wolfram scheme. When defining a rule by its polynomial number, which is useful for the sake of reference, we do not have to specify the degree d of the rule,[7] as two rules with the same polynomial number will have equivalent polynomial forms regardless of d (which is not true for the rule number). In both cases, enumeration is only practical for smaller values of d, as both $[f]^W$ and $[f]^P$ take values up to $2^{2^{d+1}}$.

[6] Which of course may only be exploited when we have a field alphabet.

[7] In fact, the degree satisfies $d = \lfloor \log \log [f]^P \rfloor$.

3.2 Binary Rule Table/Polynomial Duality

Given that local rules are typically specified by their Wolfram rule number, and we are interested here in their polynomial form, it is natural to look for ways of simplifying the process of polation. In viewing polation as an operation on rule numbers, we obtain the following result (which *does not generalise* to $F_{p>2}$).

Proposition 3.1 Given a binary rule f of degree d, let $N = 2^{d+1}$, and suppose

$$[f]^W = [f_{N-1}f_{N-2}\dots f_0]$$

$$[f]^P = [a_{N-1}a_{N-2}\dots a_0]$$

are the binary representations of the Wolfram rule number and polynomial number of f respectively, then for all $n \in \mathbb{Z}_{2^N}$

$$a_n = \sum_{m=0}^{N-1} \binom{n}{m}_2 f_m, \tag{3.5}$$

$$f_n = \sum_{m=0}^{N-1} \binom{n}{m}_2 a_m, \tag{3.6}$$

where $\binom{n}{m}_2 \in F_2$ are the binomial coefficients (mod 2).

The proof is straightforward, although it relies on well known results which will be presented in the next chapter, where we discuss the binomial coefficients in the context of linear cellular automata:
Given f written as a polynomial,

$$\begin{aligned} f(xy\dots z) &= \sum_{m=0}^{N-1} a_m x^{m_d} y^{m_{d-1}} \dots z^{m_0} \\ &= \sum_{m=0}^{N-1} a_m \binom{[x]}{m_d}_2 \binom{[y]}{m_{d-1}}_2 \dots \binom{[z]}{m_0}_2, \end{aligned}$$

where it is understood that $m = [m_d m_{d-1} \dots m_0]$. The second equality comes from observing that for $x \in F_p$, $x^0 = 1 = \binom{[x]}{0}_2$ and $x^1 = x = \binom{[x]}{1}_2$. Now if we let

$n = [n_d n_{d-1} \dots n_0] = [xy \dots z]$ we have

$$\begin{aligned} f_n &= \sum_{m=0}^{N-1}{}' a_m \binom{n_d}{m_d}_2 \binom{n_{d-1}}{m_{d-1}}_2 \cdots \binom{n_0}{m_0}_2 \\ &= \sum_{m=0}^{N-1}{}' a_m \binom{n}{m}_2 \end{aligned}$$

as required, the last equality by Theorem 4.2. The first equality in Proposition 3.1 follows directly from the second by applying the binomial inversion formula, [1] which will also be presented in the context of F_2 in the next chapter.

The reason that this proposition does not generalise to $F_{p>2}$, stated simply, is that the binomial coefficients do not agree with the powers of $x \in F_p$. That is, we have $x^0 = 1 = \binom{[x]}{0}_p$, and $x^1 = x = \binom{[x]}{1}_p$, but $x^2 \neq \binom{[x]}{2}_p$. If instead we use the binomial coefficients as a basis for rule representation, and enumerate accordingly, then Proposition 3.1 applies, but we lose the algebraic advantages of having a polynomial form. This points out one motivation for considering only binary cellular automata.

Hereafter, we will tend to omit the subscript $()_2$ and the square brackets $[x]$ when $x \in F_2$ to simplify notation. In most contexts, there is no confusion if we regard configurations or elements in F_2 as binary representations of integers, and vice versa, as we will see in the following example.

Example 3.2 To find the polynomial form of the $d = 2$ rule number $[f]^W = [10010110] = 150$, we simply calculate by Proposition 3.1

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Hence we have $[f]^P = [00010\ 10] = 22$; that is, $a_1(= a_{001}) = a_2(= a_{010}) = a_4(= a_{100}) = 1$, and therefore

$$\begin{aligned} f(xyz) &= x^0y^0z^1 + x^0y^1z^0 + x^1y^0z^0 \\ &= x + y + z. \end{aligned}$$

This is an example of a *linear* cellular automaton rule; we will discuss linear cellular automata the following chapter. The (8×8) matrix consists of the binomial coefficients in F_2 – a (mod 2) form of Pascal's Triangle – which is itself generated by the unique elementary non-trivial linear cellular automaton rule

$$g(yz) = y + z,$$

for which $[g]^P = [0110] = 6$. Hence by Proposition 3.1 again, where we now take $d = 1$, then $[g]^W = [0110] = 6$.

3.2.1 The Big Picture

This chapter has been primarily to introduce the conventions we will adopt for the definition of rules, and to discuss the reasons for working in the algebraic framework of the field F_2. In general, rules will be defined in polynomial form, and referred to by their polynomial number for the reasons discussed.

Proposition 3.1 not only provides a useful computational tool; it suggests the importance of linear cellular automata (in particular, the binomial coefficients) to the analysis of linear and nonlinear cellular automata in general. In the next chapter we discuss the binomial coefficients in detail.

Before we proceed, we suggest the following abstract view, which explains why $[f]^P$ is independent of d, whereas $[f]^W$ is apparently not. The 'correct' way to view $[f]^W$ is not as a 'number' at all, but as a (semi-infinite) periodic sequence, $[f]^W \in \mathcal{P}^{2^{d+1}}$. The periodicity of $[f]^W$ corresponds with the finite degree of the rule f. Increasing d implicitly takes a larger periodic component of the same sequence, hence the 'rule number' changes, but the sequence (i.e. the rule table) does not. The polynomial number, however, resides in the space of finite sequences, $[f]^P \in \mathcal{F}$. Hence the

enumeration $[f]^P$ is more useful for the definition of cellular automata, and will be employed throughout this thesis.

The *binomial transform* (equations (3.5) and (3.6)) which links the two spaces is the 'image' of polation under enumeration, which happens to be an involution in the case of the binary [8] binomial coefficients. Everything now may be viewed as binary sequences with various properties – rule table, polynomial, states and temporal sequences – and the binomial transform is the glue which holds them together.

[8] More generally, a p^{th} root of identity in the case of $()_p$

Chapter 4

Calculus of Cellular Automata

In this chapter we discuss the properties of the binomial coefficients in the field F_2 which are needed for the construction of a calculus of binary cellular automata. The binomial coefficients are generated by an elementary linear cellular automaton, so we begin with a discussion of linear binary cellular automata before we present results relevant to the calculus of binary cellular automata in general. We will give brief consideration to the extension of various results to prime fields F_p.

4.1 Linear Cellular Automata

We consider here first order linear [1] binary cellular automata. These have been variously studied [2] and may be considered 'solved' in the sense that "you seen one, you seen 'em all." In general, these are defined by a local rule of the form

$$x_i^{t+1} = \sum_{j=0}^{d} a_j x_{i-j}^t, \tag{4.1}$$

where $a_j \in F_2$ are constant, and $a_d = a_0 = 1$ assumed. We use this form to make natural the connection between linear cellular automata and convolution, for the

[1] There is some discrepancy as to the meaning of 'linear' in the context of cellular automata, so we will understand the term only to apply when the alphabet may be given a field structure; that is, F_{p^m}, p prime. Otherwise, for the ring $\mathbb{Z}_k$ we refer to the 'additive' cellular automata as those which obey the superposition principle.

[2] Notably [12] using generating functions, [19] fractal analysis, [5] circulant matrix methods, [15] combinatorial analysis similar to that presented here.

purpose of employing the techniques of generating functions.

4.1.1 Pascal's Triangle Revisited

We have already made reference to the elementary ($\{d, m, k\} = \{1, 1, 1\}$) non-trivial linear cellular automata, defined by

$$\begin{aligned} x_i^{t+1} &= x_i^t + x_{i-1}^t, \\ [\ldots x_{-1}^0 x_0^0 x_1^0 \ldots] &= [\ldots 010 \ldots], \end{aligned} \tag{4.2}$$

which is iterated from the simplest nonzero finite state known as a *seed*. We will refer to this particular rule as f_6, and in general the subscript of a rule is taken to be the polynomial number defined in Chapter 3. The resulting pattern is Pascal's Triangle (mod 2), one of the most fascinating mathematical objects, and ubiquitous in the study of cellular automata and fractals.

As we unravel the structure, a pretty picture becomes a highly structured tool for the study of cellular automata in general. Perhaps the most striking property of this pattern is its self-similarity (see Figure 4.1). From (4.2), observing that $x_i^0 = x_{2i}^0$ for all i, then for $t > 0$

$$\begin{aligned} x_{2i}^{2t} &= x_{2i-1}^{2t-1} + x_{2i}^{2t-1} \\ &= (x_{2i-2}^{2t-2} + x_{2i-1}^{2t-2}) + (x_{2i-1}^{2t-2} + x_{2i}^{2t-2}) \\ &= x_{2i-2}^{2t-2} + x_{2i}^{2t-2} \\ &= x_{i-1}^{t-1} + x_i^{t-1}, \end{aligned}$$

the last equality by induction on t. Hence for all $i \in \mathbb{Z}$ and $t \in \mathbb{N}$,

$$x_{2i}^{2t} = x_i^t.$$

More generally, looking at the values 'in between' as it were, it is similarly shown by induction that for all $i \in \mathbb{Z}$, $t \in \mathbb{N}$, and $s, j \in \{0, 1\}$ we have

$$x_{2i+j}^{2t+s} = x_i^t x_j^s. \tag{4.3}$$

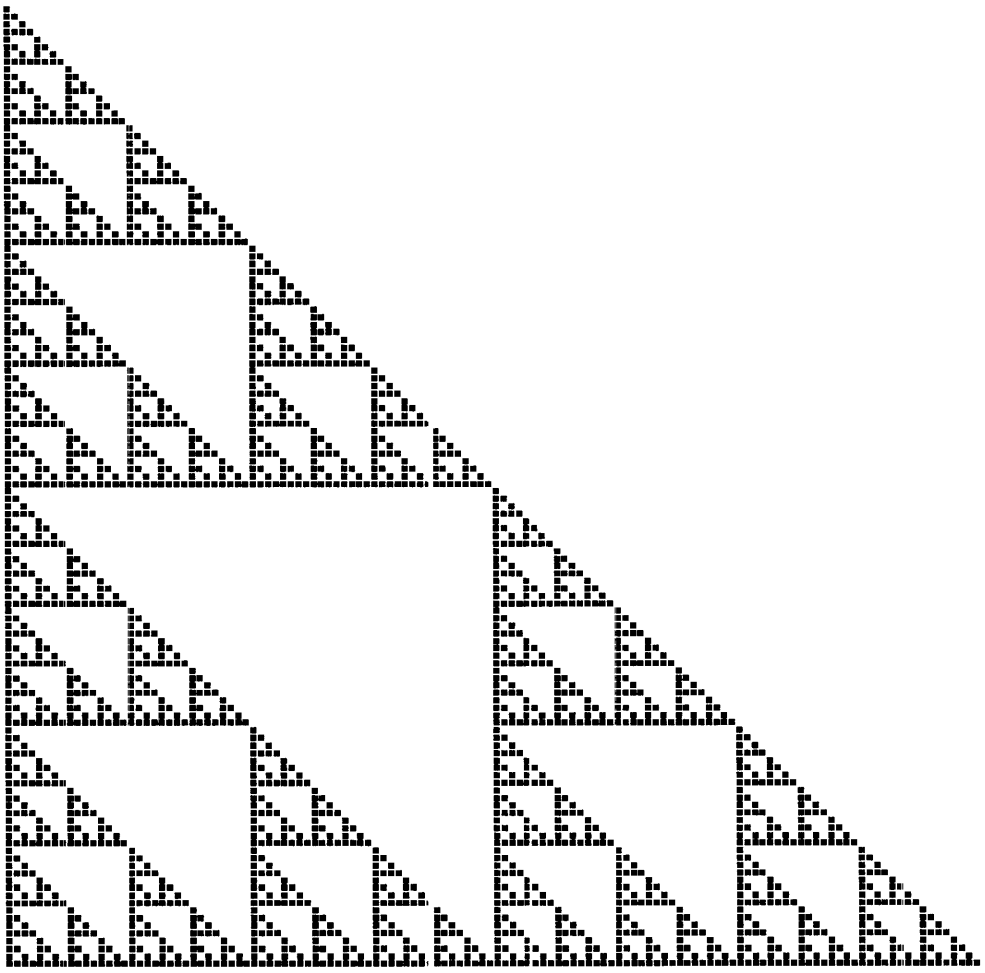


Figure 4.1: Pascal's Triangle (mod 2): rule f_6 iterated from a seed, this diagram showing the first 128 iterations. In this diagram and all that follow, unless otherwise stated: represent 1 with a black square, 0 with a blank square; x_0^0 is in the top left corner, i increases to the right and t increases down the page.

This recurrence relation is sufficient to generate the same pattern from the given initial state, and gives a better 'feel' for the growth of the structure. If we let $[t] = [t_m \dots t_0]$ and $[i] = [i_m \dots i_0]$ be the binary representations of $t, i \in \mathbb{N}$ (padded with zeros if necessary) then by (4.3)

$$\begin{aligned} x &= x_{[i_m \dots i_1]}^{[t_m \dots t_1]} x_{i_0}^{t_0} \\ &= x_{[i_m \dots i_2]}^{[t_m \dots t_2]} x_{i_1}^{t_1} x_{i_0}^{t_0} \\ &= \cdot \ \cdot \\ &= x_{i_m}^{t_m} x_{i_{m-1}}^{t_{m-1}} \cdots x_{i_1}^{t_1} x_{i_0}^{t_0}, \end{aligned} \tag{4.4}$$

which we may regard as a solution of the cellular automata, insofar as it reduces the evaluation of a site to a series of bitwise operations performed on the site coordinates, where we observe that for $s, j \in F_2$,

$$x_j^s = 1 + j(1 + s).$$

Equation (4.4) is in fact a special case of a theorem attributed [15] to Lucas (1877) which is fundamental to the analysis of cellular automata.

4.1.2 Generating Functions

The analysis of linear binary cellular automata on $\mathcal{F}^-$ is facilitated with the use of generating functions. Normally they are defined for semi-infinite sequences $\{x_i\}_{i=0}^{\infty}$, however we make a more general definition to liberate some analysis from the consideration of limits of summation. If we understand that

$$\sum_i \equiv \sum_{i=-\infty}^{\infty}, \tag{4.5}$$

then for a sequence $x \in F_2^{\mathbb{Z}}$, we define the *generating function of* x, denoted $x(z)$, to be the formal power series

$$x(z) = \sum_i x_i z^i. \tag{4.6}$$

Addition and scalar multiplication are defined by

$$x(z) + \alpha y(z) = \sum_i (x_i + \alpha y_i) z^i,$$

for all $\alpha \in F_2$, and the product of two generating functions is given by

$$x(z)y(z) = \sum_i \sum_j x_j y_{i-j} z^i,$$

so long as the R.H.S. is well defined.

The methods of generating functions provide a useful framework for the discussion of linear cellular automata on $\mathcal{F}^-$. What follows may be regarded as well known [3] in the context of cellular automata analysis, and is presented here for the discussion of the binomial coefficients in F_2.

4.1.3 Solution of Linear Cellular Automata

If we restrict our states to $\mathcal{F}^-$, then we may represent each state $x^t \in \mathcal{F}^-$ with a generating function, denoted $x^t(z) \in \mathcal{F}^-(z)$, where

$$\begin{aligned} x^t(z) &= \sum_i x_i^t z^i \\ &= \sum_{i=0}^{\infty} x_i^t z^i. \end{aligned}$$

This means that the product of $x^t(z)$ with any $a(z) \in \mathcal{F}^-(z)$ is well defined, as we have

$$\begin{aligned} a(z)x^t(z) &= \sum_i \sum_j a_j x_{i-j} z^i \\ &= \sum_{i=0}^{\infty} (\sum_{j=0}^{i} a_j x_{i-j}) z^i, \end{aligned}$$

so clearly $a(z)x^t(z) \in \mathcal{F}^-(z)$. Now let $a(z) = \sum_{i=0}^{d} a_i z^i \in \mathcal{F}(z) \subset \mathcal{F}^-(z)$ be some polynomial, and the previous equation becomes

$$a(z)x^t(z) = \sum_{i=0}^{\infty} (\sum_{j=0}^{d} a_j x_{i-j}) z^i.$$

[3] Notably [12] analyses 'additive' cellular automata on $\mathcal{P}_k$ using methods of generating functions; that is, exploiting the properties of cipolynomials mod $(z^n - 1)$, where n is the period of the allowable states (alternatively the size of the lattice). Here we we have a similar treatment.

Now put

$$x^{t+1}(z) = a(z)x^t(z), \tag{4.7}$$

and it follows that

$$x_i^{t+1} = \sum_{j=0}^{d} a_j x_{i-j}^t,$$

which is precisely our definition of a linear cellular automaton, equation (4.1).

We may regard (4.7) as an alternative definition of a linear cellular automaton rule, for which we write the solution

$$x^t(z) = (a(z))^t x^0(z), \tag{4.8}$$

for some initial state $x^0(z)$. The behaviour of the cellular automaton may be analysed by considering the properties of $(a(z))^t$. We will not go into any detail here, as the linear problem may be considered 'solved'.

4.1.4 Inversion of Linear Cellular Automata

Firstly we point out the importance of the restriction[4] on the polynomial $a(z)$ that $a_0 = 1$, which we write $a(z) \in \mathcal{F}_0^-(z)$, for then and only then [2] does there exist a unique $b(z) \in \mathcal{F}_0^-(z)$ such that $a(z)b(z) = 1$, which we may write $b(z) = a^{-1}(z)$. It follows that for all $x^t(z) \in \mathcal{F}^-(z)$ there exists a unique $x^{t-1}(z) = a^{-1}(z)x^t(z) \in \mathcal{F}^-(z)$ such that $x^t(z) = a(z)x^{t-1}(z)$. Hence we conclude:

Proposition 4.1 If $(f, \mathcal{F}^-)$ is a linear cellular automaton where $f(0\ldots01) \neq 0$, then the global mapping F defined by f is invertible on $\mathcal{F}^-$.

In terms of our more general definition of a generating function (4.6), a polynomial $a(z) \in \mathcal{F}_0(z)$ of degree a has 2^d inverses[5] in $\mathcal{U}(z)$, which may be constructed from $a^{-1}(z) \in \mathcal{F}_0^-(z)$ as follows. The homogeneous equation $a(z)p(z) = 0$ has 2^d solutions $p(z) \in \mathcal{P}(z)$, each corresponding to an arbitrary choice of $p_0, p_1, \ldots, p_{d-1} \in F_2$. Hence $a(z)\left(a^{-1}(z) + p(z)\right) = 1 + a(z)p(z) = 1$. The $p(z)$ are precisely analogous

[4] With no loss of generality by spatial relativity as discussed in Chapter 2.

[5] More generally p^d for a polynomial in $F_p[z]$

to the arbitrary functions of integration or summation of a d^{th} order differential or difference equation respectively. The restriction of $x^t(z)$ to $\mathcal{F}^-(z)$ may be viewed as the imposition of boundary conditions.

This is not so much a characteristic of linear cellular automata, but more a consequence of of restricting the allowable states of the cellular automaton. In general, it is not hard to find a set of states upon which a given rule has an invertible global mapping. For example, a cellular automaton $(f, \mathcal{P}^m)$ is (temporally) ultimately periodic, so if we let

$$\begin{aligned} \mathcal{S}^\infty &= \lim_{n\to\infty} F^n \mathcal{P}^n \\ &= \{x \in \mathcal{P}^m \quad F^n x = x, \text{for some } n > 0\} \end{aligned}$$

be the *basin* of $(f, \mathcal{P}^m)$, then $(f, \mathcal{S}^\infty)$ is an invertible cellular automaton.

4.2 The Binomial Coefficients

Equation (4.2) is of course one of various equivalent definitions of the binomial coefficients, each useful depending on the context in which they are used. For simplicity, given that we are operating exclusively within a prime field structure, we will define them recursively without reference to factorial functions. Further, they will be understood to be field elements rather than integers (basically to avoid writing '(mod p)' everywhere). Normally $()_p \quad (t,i) \in \mathbb{N} \times \mathbb{Z} \longrightarrow \binom{t}{i}_p \in F_p$ is defined by

$$\begin{aligned} \binom{t+1}{i}_p &= \binom{t}{i-1}_p + \binom{t}{i}_p, \\ \text{where} \quad \binom{0}{i}_p &= \begin{cases} 1 & \text{if } i = 0 \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{4.9}$$

We shall omit the subscript $p = 2$ when discussing binary binomial coefficients.

Comparing this with (4.2) we may write

$$a_i^t = \binom{t}{i},$$

which we may regard as a closed form solution of our cellular automaton, or alternatively consider the cellular automaton as a reasonably fast calculator of the binomial coefficients. This points out a conceptual difficulty with the notion of 'solution' of cellular automata, since we want our solution to be more easily computed than the definition of the cellular auton aton itself. To these ends we have the following theorem, which is fundamental to understanding the binomial coefficients on a prime field.

Theorem 4.2 (Lucas) Let p be a prime, and let

$$\begin{aligned} [t]_p &= [t_m t_{m-1} \dots t_0] \\ [i]_p &= [i_m i_{m-1} \dots i_0] \end{aligned}$$

be the base p representations of $t, i \in \mathbb{N}$. Then

$$\binom{t}{i}_p = \binom{t_m}{i_m}_p \binom{t_{m-1}}{i_{m-1}}_p \cdots \binom{t_0}{i_0}_p . \tag{4.10}$$

This theorem greatly simplifies the calculation of the binomial coefficients in a prime field. As discussed in [13], the calculation of the coefficients requires in the order of $\log \log \min(t, i)$ operations on average, when we consider that we stop when one of the $\binom{t_n}{i_n}$ is zero. As one consequence we have Proposition 3.1, but the significance of this theorem extends further, particularly in the field F_2.

By way of proof, we could proceed inductively as for equation (4.4), however there is a very short proof due to Fine [4] well worth seeing which exploits the properties of polynomials on finite fields.

4.2.1 Extended Binomial Coefficients

For our purposes it will be useful to extend the binomial coefficients for $t < 0$, which is equivalent to inverting a linear cellular automaton, and as discussed in the previous section, such extension of the binomial coefficients will not be unique. Unambiguous definition of the extended binomial coefficients is achieved with the following definition:

Definition : For all $t \in \mathbb{Z}$,

$$(1+z)^t = \sum_i \binom{t}{i} z^i, \tag{4.11}$$

where we take $(1+z)^t \in \mathcal{F}^-(z)$ for $t < 0$.

We now propose to extend Lucas' Theorem, but first we need to define $[t]$ for $t < 0$, for which we employ 2's complement. Suppose for some $t > 0$, we have

$$\begin{aligned} [t] &= [t_d t_{d-1} \ldots t_0] \\ &= [\ldots t_{d+1} t_d t_{d-1} \ldots t_c 0 \ldots 0] \end{aligned}$$

where $t_{d+1} = t_{d+2} = \cdots = 0$, and c is the least integer such that $t_c = 1$ – in other words, $2^c | t$ (read as '2^c divides t') and $2^{c+1} \not| t$. Then we define

$$[-t] = [\ldots \bar{t}_{c+2} \bar{t}_{c+1} t_c 0 \ldots 0],$$

where $\bar{t}_n$ is the complement of t_n, that is $\bar{t}_n = 1 - t_n$. Notice that $[t]$ and $[-t]$ agree for $t_c \ldots t_0$.

It is now easily shown that for all $t \in \mathbb{Z}$ and $s \in \{0, 1\}$, $[t]$ is obtained by performing a *right shift* on $[2t + s]$ (truncating the trailing bit s),

$$[2t + s] = [[t][s]]. \tag{4.12}$$

We may now prove:

Theorem 4.3 (Binary Lucas extended)[6] For all $t \in \mathbb{Z}$ and $i \in \mathbb{N}$,

$$\begin{aligned} \binom{t}{i} &= \prod_{m=0}^{\infty} \binom{t_m}{i_m} \\ &= \prod_{m=0}^{n} \binom{t_m}{i_m}, \end{aligned} \tag{4.13}$$

where $[t] = [\ldots t_1 t_0]$, $[i] = [\ldots i_1 i_0]$ and $n = \lceil \log \max(|t|, i) \rceil$.

[6] This theorem extends to F_p, which is not obvious given that we have an infinite product which converges if and only if the sequence takes values 0 or 1 in the limit. However, employing p's complement (that is, represent negative integers with leading $p-1$'s), and observing that $\binom{p}{0}_p = \binom{p-1}{p-1}_p = 1$, then a proof along the lines of the following is possible.

To prove this, we consider separately t even and t odd. We have for all $t \in \mathbb{Z}$,

$$\begin{aligned} 0 &= (1+z)^{2t} + (1+z^2)^t \\ &= \sum_i \binom{2t}{i} z^i + \sum_i \binom{t}{i} z^{2i} \\ &= \sum_i \left[\binom{2t}{2i} + \binom{t}{i} \right] z^{2i} + \sum_i \binom{2t}{2i+1} z^{2i+1}, \end{aligned}$$

by grouping odd and even powers of z in separate sums. Similarly,

$$\begin{aligned} 0 &= (1+z)^{2t+1} + (1+z)(1+z^2)^t \\ &= \sum_i \binom{2t+1}{i} z^i + (1+z) \sum_i \binom{t}{i} z^{2i} \\ &= \sum_i \left[\binom{2t+1}{2i} + \binom{t}{i} \right] z^{2i} + \sum_i \left[\binom{2t+1}{2i+1} + \binom{t}{i} \right] z^{2i+1}, \end{aligned}$$

again, by grouping odd and even powers. These two equalities imply that, for all $s, j \in \{0,1\}$, we have

$$\binom{2t+s}{2i+j} = \binom{t}{i}\binom{s}{j}, \tag{4.14}$$

where we observe that $\binom{0}{0} = \binom{1}{0} = \binom{1}{1} = 1$, and $\binom{0}{1} = 0$. The result follows inductively from (4.14) and (4.12), where we observe that the finite product is equal to the infinite product, since

$$\begin{aligned} \prod_{m=n+1}^{\infty} \binom{t_n}{i_n} &= \begin{cases} \prod_{m=n+1}^{\infty} \binom{1}{0}, & t < 0 \\ \prod_{m=n+1}^{\infty} \binom{0}{0}, & t \geq 0 \end{cases} \\ &= 1. \end{aligned}$$

We note that for $i < 0$, equation (4.13) defines an alternative set of binomial coefficients, which agree with our restricted binomial coefficients for $i \geq 0$. The relationship between the two is precisely the relationship between generating functions as they are usually defined (positive power series in say x) and "z-transforms" (negative power series in z) under the transformation $z = \frac{1}{x}$ [10].

4.2.2 Binomial Sequences

As a consequence of Theorem 4.3, we present a brief discussion of the temporal (i fixed) and spatial (t fixed) 'behaviour' of the binomial coefficients $\binom{t}{i}$; that is, the rows and columns of the extended Pascal's Triangle, which we refer to as the *spatial* and *temporal binomial sequences* respectively.

Proposition 4.4 (Temporal Periodicity) If $i \geq 0$ is fixed, then the temporal binomial sequence $\binom{t}{i} \in \mathcal{P}^{2^{\mathbb{N}}}$ with period $P\binom{t}{i} = 2^{\lceil \log(i+1) \rceil}$.

(Note that for $i < 0$ fixed, $\binom{t}{i} = 0$, so $P\binom{t}{i} = 1$ in this case.)

Proposition 4.5 (Spatial Semi-periodicity) If $t < 0$ is fixed, then the spatial binomial sequence $\binom{t}{i} \in \mathcal{F}^-$ with preperiod $Q\binom{t}{i} = 0$ (which we refer to as *semi-periodicity*) and right period $R\binom{t}{i} = 2^{\lceil \log(-t) \rceil}$.

Proposition 4.6 (Spatial Finiteness) If $t \geq 0$ is fixed, then the spatial binomial sequence $\binom{t}{i} \in \mathcal{F}$ with preperiod $Q\binom{t}{i} = t + 1$.

The last of these properties is well known. The others may be shown from Theorem 4.3. For example, to prove Proposition 4.4, first show for $i \geq 0$, $\binom{t}{i} = \binom{t + 2^{\lceil \log(i+1) \rceil}}{i}$ for all t, so $P\binom{t}{i}$ divides $2^{\lceil \log(i+1) \rceil}$. Then show $\binom{t}{i} = 0$ for $0 \leq t < i$, and $\binom{i}{i} = 1$, thus establishing a lower bound $P\binom{t}{i} \geq i + 1$, and the result follows.

4.2.3 Algebra of Binomial Sequences

Here we discuss algebraic properties of the binary [7] binomial sequences which are consequences of Theorem 4.3.

Theorem 4.3 reduces the computation of $\binom{t}{i}$ to a series of bitwise operations which may be expressed in terms of Boolean algebra; that is, for $s, j \in \{0, 1\}$,

$$\binom{s}{j} = j' \vee s, \tag{4.15}$$

where we use the symbols $\{\wedge\ \vee, '\}$ to denote the Boolean operations. Hence, for $t \in \mathbb{Z}$ and $i \in \mathbb{N}$ we may write

$$\binom{t}{i} = \bigwedge_{m=0}^{\infty} i'_m \vee t_m. \tag{4.16}$$

We may extend the Boolean operations bitwise to integers as follows:

$$\begin{aligned} [i] \vee [t] &= [\,..[i_d \vee t_d]\ldots[i_0 \vee t_0]] \\ [i] \wedge [t] &= [\,..[i_d \wedge t_d]\ldots[i_0 \wedge t_0]] \\ [t\,' &= [\ldots t'_d \ldots t'_0] \\ &= [-t - 1]. \end{aligned}$$

We then have as a consequence of Theorem 4.3 and De Morgan's Laws:

Theorem 4.7 (Algebra of Binary Binomial Sequences) For all $i, j, s, t \in \mathbb{Z}$,

$$\binom{t}{i}\binom{t}{j} = \binom{t}{[i] \vee [j]} \tag{4.17}$$

$$\binom{t}{i}\binom{s}{i} = \binom{[t] \wedge [s]}{i}. \tag{4.18}$$

[7] The extension of these results to F_p is unclear, and would appear difficult, as we now work with Boolean algebra. There is some irony in this statement, as I initially set out to avoid Boolean algebra, convinced that it did not have properties conducive to explicit analysis of homogeneous nonlinear systems.

The proof is easy, but we must be careful with $i < 0$ and $j < 0$, as Theorem 4.3 does not apply in these cases. Assume first that $i, j \geq 0$, and we have

$$\begin{aligned}
\binom{t}{i}\binom{t}{j} &= \left(\bigwedge_{m=0}^{\infty} i'_m \vee t_m\right) \wedge \left(\bigwedge_{m=0}^{\infty} j'_m \vee t_m\right) \\
&= \bigwedge_{m=0}^{\infty} (i'_m \vee t_m) \wedge (j'_m \vee t_m) \\
&= \bigwedge_{m=0}^{\infty} (i'_m \wedge j'_m) \vee t_m \\
&= \bigwedge_{m=0}^{\infty} (i_m \vee j_m)' \vee t_m \\
&= \binom{t}{[i] \vee [j]},
\end{aligned}$$

as required. Similarly,

$$\begin{aligned}
\binom{t}{i}\binom{s}{i} &= \bigwedge_{m=0}^{\infty} (i'_m \vee t_m) \wedge (i'_m \vee s_m) \\
&= \bigwedge_{m=0}^{\infty} i'_m \vee (t_m \wedge s_m) \\
&= \binom{[t] \wedge [s]}{i}.
\end{aligned}$$

If $i < 0$, then

$$\begin{aligned}
\binom{t}{i}\binom{s}{i} &= 0 \\
&= \binom{[t] \wedge [s]}{i},
\end{aligned}$$

since we have chosen to define $\binom{t}{i} = 0$ for all $t \in \mathbb{Z}$ and $i < 0$. Finally if $i < 0$ or $j < 0$, we observe that $[i] \vee [j] < 0$, so we have

$$\begin{aligned}
\binom{t}{i}\binom{t}{j} &= 0 \\
&= \binom{t}{[i] \vee [j]}.
\end{aligned}$$

as required, and the proof is complete.

This theorem is central to the calculus of binary cellular automata. It is well known that the binomial sequences provide a natural basis for difference calculus, however the algebra of the binomial coefficients in most environments is anything but simple. Theorem 4.7 demonstrates that the algebra of the *binary* binomial sequences is straightforward, and most importantly *closed under bitwise (element by element) multiplication*, which has significant consequences for the calculus of nonlinear binary cellular automata.

4.3 Difference Calculus of Cellular Automata

As we have seen in the previous section, the binomial sequences have unusual and useful algebraic properties. To exploit these properties, we describe a familiar difference calculus, where the binomial sequences are used as a natural basis for temporal and spatial difference operations.

First we define the spatial and temporal difference operators, $\Delta_{i-}, \Delta_{t+} : F_2^{\mathbb{Z}} \to F_2^{\mathbb{Z}}$ by

$$\begin{aligned} \Delta_{i-}x)_i &= x_{i-1} + x_i, \\ \Delta_{t+}x)^t &= x^{t+1} + x^t. \end{aligned}$$

No significance should be attributed to the fact that we have defined the *backward* spatial difference operator Δ_{i-} and the *forward* temporal difference operator Δ_{t+}, beyond our particular choice for the recurrence relation defining the binomial coefficients, equation (4.9), which we may now alternatively write

$$\begin{aligned} \binom{t-1}{i} &= \Delta_{i-}\binom{t}{i}, \qquad \text{or} \\ \binom{t-1}{i-1} &= \Delta_{t+}\binom{t}{i}. \end{aligned} \tag{4.19}$$

Corresponding to each difference operator is a summation operator. We say z_i is an *indefinite sum* of y_i, written $z_i = \sum_{i-} y_i$, if

$$\Delta_{i-}z_i = y_i,$$

and similarly for $\sum_{t+}$. It follows for binary sequences that every indefinite sum of y_i has the form $z_i + C$, where $C \in \{0, 1\}$ is an *arbitrary constant* of summation. We have already discussed these operations in the context of generating functions; differentiation is equivalent to multiplication by $(1 + z)$, and summation is equivalent to multiplication by $(1 + z)^{-1}$ which was not unique unless 'boundary conditions' were imposed. We may now write

$$\begin{aligned} \sum_{i-} \binom{t}{i} &= \binom{t-1}{i} + C_1, \qquad \text{and} \\ \sum_{t+} \binom{t}{i} &= \binom{t}{i+1} + C_2. \end{aligned}$$

Now if we restrict our sequences spatially to $\mathcal{F}^-$, then $C_1 = 0$. Also, to keep things tidy, we may write $C_2 = C \binom{t}{0}$ to get

$$\sum_{i-} \binom{t}{i} = \binom{t-1}{i}, \tag{4.20}$$

$$\sum_{t+} \binom{t}{i} = \binom{t}{i+1} + C \binom{t}{0}. \tag{4.21}$$

Hence the binomial coefficients are closed under both differentiation and summation in both variables, where both operations are easily implemented in the framework of binary fields. In this thesis, we will be primarily concerned with temporal differentiation/summation, and will more frequently use equation (4.21).

4.4 Binomial Transform and Inversion

Having demonstrated that the binomial sequences are natural bases for both algebra and difference operations, we are interested here with the transformation to and from these bases. We will show that the spatial binomial sequences provide a basis for $\mathcal{F}$, and the temporal binomial sequences a basis for $\mathcal{P}^{2^{\mathbb{N}}}$. Each may be used effectively for the analysis of cellular automata, and both are discussed here for sake of generality. However, we will primarily exploit the latter for the purpose of analysing temporal sequences of velocity 0.

4.4.1 Spatial Transform

We want to be precise here about the length of finite sequences, so we introduce the following notation: for $l, r \geq 0$, define $\mathcal{F}_{l,r} \subset \mathcal{F}$ by

$$\mathcal{F}_{l,r} = \{x \in \mathcal{F} : x_l = x_r = 1; x_i = 0, \text{ for } i < l \text{ and } i > r\}.$$

In some cases we may only want to be specific about either l or r, in which case we omit the other subscript. For example,

$$\mathcal{F}_{,r} = \{x \in \mathcal{F} : x_r = 1; x_i = 0, \text{ for } i > r\}.$$

Now suppose we have a (spatial) sequence $a \in F_2^{\mathbb{Z}}$, we define the *spatial transform* of a, denoted $\check{a} \in F_2^{\mathbb{Z}}$, by

$$\check{a}_i = \sum_j \binom{j}{j-i} a_j, \tag{4.22}$$

so long as this infinite sum is well defined. [8] This is clear in the case where a is finite, say $a \in \mathcal{F}_{l,r}$, for which equation (4.22) simplifies to

$$\begin{aligned} \check{a}_i &= \sum_j \binom{j}{j-i} a_j \\ &= \sum_{j=l}^{r} \binom{j}{j-i} a_j \\ &= \sum_{j=l}^{r} \binom{j}{i} a_j \\ &= \sum_j \binom{j}{i} a_j, \end{aligned} \tag{4.23}$$

the second last equality by a well known property of the binomial coefficients.

Discussion of the spatial transform is simplified by using generating functions:

Proposition 4.8 Let $a \in \mathcal{F}$ have transform $\check{a} \in F$, then $\check{a}(z) = a(z+1)$.

[8] This would appear the 'right' way to define the spatial binomial transform for the extended binomial coefficients, which is well defined on broader classes of sequences than $\mathcal{F}$. In this thesis, we are only concerned with $\mathcal{F}$, in which case the transform simplifies to the more familiar (4.23), which suffices for discussions in this thesis. The extended binomial transform and inversion is discussed briefly in Appendix A.

Proof:

$$\begin{aligned}
\check{a}(z) &= \sum_i \check{a}_i z^i \\
&= \sum_i \left(\sum_j \binom{j}{i} a_j \right) z^i \\
&= \sum_j \left(\sum_i \binom{j}{i} z^i \right) a_j \\
&= \sum_j (z+1)^j a_j \\
&= a(z+1).
\end{aligned}$$

It follows that the operations of addition and convolution (corresponding to polynomial addition and multiplication respectively) are invariant under the spatial transform:

Corollary 4.9 For $a(z), b(z) \in \mathcal{F}(z)$,

$$\begin{aligned}
\widehat{\check{a}(z) + b(z)} &= \check{a}(z) + \check{b}(z), \\
\widehat{\check{a}(z) b(z)} &= \check{a}(z)\check{b}(z).
\end{aligned} \tag{4.24}$$

The inverse transform is easily achieved with generating functions. To determine a from $\check{a}$, put $z' = z + 1$ and we have

Corollary 4.10 $a(z) = \check{a}(z+1)$.

Put otherwise, $a = \check{\check{a}}$, so the spatial transform is an involution on $\mathcal{F}$. [9]

Corollary 4.11 $a \in \mathcal{F}_r \Leftrightarrow \check{a} \in \mathcal{F}_r$.

The proof comes simply from observing $a(z+1)$ is a polynomial of the same degree as $a(z)$.

[9] More generally, a p^{th} root of identity on $\mathcal{F}_p$.

Example 4.1 A spatial transform analysis of the rule f_6 would yield, from equation (4.8) with $a(z) = (z+1)$ and $x^0(z) \in \mathcal{F}(z)$ arbitrary,

$$\begin{aligned} \check{x}^t(z) &= \overbrace{(z+1)^t x^0(z)}^{\check{}} \\ &= z^t \check{x}^0(z+1), \quad \text{by (4.24)}, \end{aligned}$$

which would appear that we are simulating f_6 with the trivial $f_4(xy) = x$. This is not the case, as the inverse transform represents precisely the same amount of work as the iteration of f_6.

4.4.2 Binomial Inversion

Corollary 4.10 simply states the binomial inversion formula [1] which simplifies for the binary binomial coefficients: for $t, i \in \mathbb{N}$,

$$\sum_m \binom{t}{m} \binom{m}{i} = \sum_{m=i}^{t} \binom{t}{m} \binom{m}{i} = \delta_i^t, \tag{4.25}$$

where $\delta_m^m = 1$, and $\delta_n^m = 0$ for $n \neq m$.

4.4.3 Temporal Transform

The sequences with which we are primarily concerned in the following chapter are $\mathcal{P}^{2^{\mathbb{N}}}$. Here we show that the columns of the extended Pascal's Triangle, that is the temporal sequences $\left\{ \binom{t}{i} \right\}_{i \in \mathbb{N}}$, form a basis for $\mathcal{P}^{2^{\mathbb{N}}}$.

Firstly, to establish that these sequences are linearly independent, we consider the *Casorati matrix* [10]: for $n \in \mathbb{N}$, let

$$W_n(t) = \begin{bmatrix} \binom{t}{0} & \binom{t}{1} & \cdots & \binom{t}{n-1} \\ \binom{t+1}{0} & \binom{t+1}{1} & \cdots & \binom{t+1}{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \binom{t+n-1}{0} & \binom{t+n-1}{1} & \cdots & \binom{t+n-1}{n-1} \end{bmatrix}.$$

The *Casoration*, which may be viewed as the discrete analogue of the Wronskian, is the determinant

$$w_n(t) = \det W_n(t).$$

Following [10], the sequences $\left\{ \binom{t}{i} \right\}_{i=0}^{n}$ are linearly independent if and only if $w_n(t) \neq 0$ for some (alternatively *all*) t. We observe that (Proposition 4.6) the matrix

$$W_n(0) = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \cdots & \cdots & \cdots & 1 \end{bmatrix}.$$

is lower diagonal, and hence $w_n(0) = w_n(t) = 1$ for all $n \in \mathbb{N}$. [10]

It follows from Proposition 4.4 that the linearly independent set of 2^m temporal binomial sequences $\left\{ \binom{t}{i} \right\}_{i=0}^{2^m-} \subset \mathcal{P}^{2^m}$, and hence spans $\mathcal{P}^{2^m}$, since there are clearly 2^{2^m} sequences in $\mathcal{P}^{2^m}$, and 2^{2^m} linear combinations of the $\left\{ \binom{t}{i} \right\}_{i=0}^{2^m-1}$ with coefficients in F_2.

Now, suppose we have a (temporal) sequence $a \in \mathcal{P}^{2^m}$, we define the *temporal transform* of a, denoted $\hat{a} \in \mathcal{F}$ such that

$$a^t = \sum_j \binom{t}{j} \hat{a}^j. \tag{4.26}$$

By the previous discussion, $\hat{a}$ is unique, and satisfies $\hat{a} \in \mathcal{F}_n$ where $n < 2^m$. We may write

$$a^t = \sum_{j=0}^{2^m-1} \binom{t}{j} \hat{a}^j,$$

and hence, by the binomial inversion formula (4.25),

$$\hat{a}^t = \sum_{j=0}^{2^m-1} \binom{t}{j} a^j. \tag{4.27}$$

[10] The fact that $w_n(t) = 1$ for all n is an interesting property of the binomial coefficients which is not pursued here.

We note that an infinite sum is not well defined here, however if we now interpret

$$\sum_j \equiv \lim_{n\to\infty} \sum_{j=-2^n}^{2^n-1},$$

which agrees with the previous convention, then we may write

$$\hat{a}^t = \sum_j \binom{t}{j} a^j$$

for sake of formal simplicity.

We are particularly interested in the sequences $\mathcal{P}^{2^N}$ as they tend to 'dominate' the behaviour of *all* temporal sequences of velocity 0 of *any* cellular automaton $(f, \mathcal{F}^-)$, which is the subject of the following chapter.

4.5 Summary

The binomial sequences are natural bases for the analysis of cellular automata, for the following reasons as discussed in this chapter.

- The computation of the binomial coefficients is well understood by Theorem 4.3. Consequently:
- the behaviour of the binomial sequences is known by Propositions 4.4 to 4.6;
- the bitwise algebra of the binomial sequences is understood by Theorem 4.7.
- The difference calculus of the binomial sequences is well known – equations (4.19) to (4.21) – which follow from the definition of the binomial coefficients.
- Finally, the binomial transforms to these bases are understood by the discussions in the previous section.

Given that we have natural bases for *both* algebra *and* difference calculus, [11] we are interested in the extent to which the techniques of calculus have application

[11] Which is by no means the case for finite difference calculus in general; for example, the (temporal) binomial sequences are a natural basis for the difference calculus of sequences of real numbers, whereas polynomials are a natural basis for algebra; the transformation between these two bases is by no means straightforward.

to nonlinear cellular automata which is the subject of the following chapter. The significance of these results would probably be unclear at this point, given the unusual combination of field and boolean algebras which has preceded. Before we give some examples in the next chapter, it is probably best described here by continuous analogy.

When integrating/differentiating the product of two analytic functions, one has a choice: either integrate by parts/invoke the product rule, or first express the product as a sum of analytic functions if possible, and then proceed to integrate/differentiate. Anyone familiar with the difference calculus will know that the latter approach is certainly preferable if feasible; discrete analogues of integration by parts and the product rule do exist, but they become frustratingly complex in most practical situations.

Theorem 4.7 indicates that if we use the binomial sequences as bases for our states or temporal sequences, then the latter approach becomes feasible. In fact, as we will demonstrate in the next chapter, it is possible to find *exact solutions* for a class of nonlinear cellular automata for restricted initial states. Failing this, it is straightforward to develop methods of stability analysis for much broader classes of cellular automata, and to provide a reasonably explicit discussion of sensitivity to initial conditions.

Chapter 5

Nonlinear Cellular Automata

In this chapter we will analyse nonlinear binary cellular automata, beginning with the elementary and proceeding in the direction $\{d, 1, 1\}$; that is, first order and incrementing degree. Analysis will be restricted either to non-generative $(f_n, \mathcal{F}^-)$ or $(f_n, \mathcal{F})$ as stated, where the subscript n is the polynomial number of f_n defined in Chapter 3.

5.1 A Cellular Automaton Experiment

In this section we look at an example of a 'simple' $d = 2$ rule

$$f_{66}(xyz) = xy + z,$$

for which the procedure of analysis is easily extended to similar rules of higher order. This is done to illustrate one advantage of writing a rule in the form (2.3): we may generate temporal sequences of velocity 0 of arbitrary length on a finite machine. [1] Furthermore, it demonstrates the ease with which results are extended *algebraically* to rules of higher degrees, given that rules are expressed in polynomial form. We also present something resembling a closed form solution, derived from the techniques outlined in the previous chapter.

[1] We will assume that all temporal sequences referred to in this chapter have velocity 0 unless otherwise stated.

The rule f_{66} may be written formally

$$\Delta_{t+} x_i^t = x_{i-2}^t \cdot x_{i-1}^t, \tag{5.1}$$

and without loss of generality, we may assume that the initial state $x^0 \in \mathcal{F}_0$ that is, satisfies $x_0^0 = 1$.

We begin with an empirical approach, by iterating this rule from some simple initial states. If we put $[x_0^0] = [1]$, then nothing very interesting happens. The solution by observation is $x_0^t = 1$, $x_i^t = 0$ otherwise. Now if we put $[x_0^0 x_1^0] = [11]$, a 'double seed', then things start warming up (refer to Figure 5.1).

Observing the self-similarity of the resulting pattern, one may well expect that some closed-form solution should be attainable. Starting with the temporal sequence x_0, we have

$$\begin{aligned} x_0^t &= \textstyle\sum_{t+} x_{-1}^t \cdot x_{-2}^t \\ &= \textstyle\sum_{t+} 0 \\ &= C \binom{t}{0}, \end{aligned}$$

where $C = 1$ is determined from the initial state. Next, we have

$$\begin{aligned} x_1^t &= \textstyle\sum_{t+} x_0^t \cdot x_{-1}^t \\ &= \textstyle\sum_{t+} 0 \\ &= C \binom{t}{0}, \end{aligned}$$

where again $C = 1$. Proceeding in this fashion:

$$\begin{aligned} x_2^t &= \textstyle\sum_{t+} x_1^t \cdot x_0^t \\ &= \textstyle\sum_{t+} \binom{t}{0} \binom{t}{0} \\ &= \textstyle\sum_{t+} \binom{t}{0}, \qquad \text{by Theorem 4.7} \\ &= \binom{t}{1} + C \binom{t}{0} \\ &= \binom{t}{1}, \end{aligned}$$

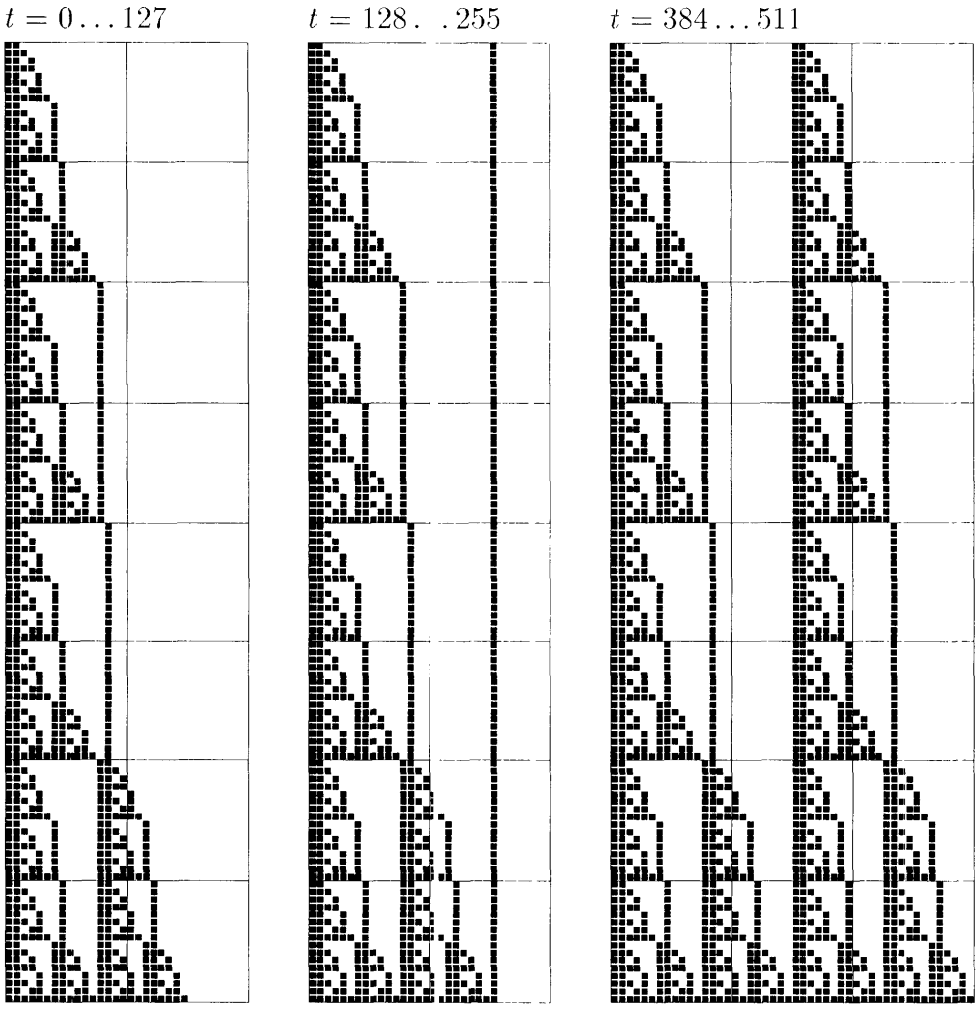


Figure 5.1: The $d = 2$ nonlinear rule $f_{63}(xyz) = xy + z$, iterated from a 'double seed'. A (16×16) grid is included for reference.

because now $x_2^0 = x_3^0 = \ldots = 1$. Hence, there exists a sequence of integers $y_i \in \mathbb{Z}$ such that

$$x_i^t = \binom{t}{y_i}$$

for all $i \in \mathbb{Z}$ (take $y_i = -1$ for all $i < 0$), and we have reduced the problem to solving for the sequence y_i. This is easy to show by induction; by Theorem 4.7, for $i > 2$,

$$\begin{aligned} x_i^t &= \sum_{t+} \binom{t}{y_{i-1}} \binom{t}{y_{i-2}} \\ &= \sum_{t+} \binom{t}{y_{i-1} \vee y_{i-2}} \\ &= \binom{t}{(y_{i-1} \vee y_{i-2}) + 1} + C, \end{aligned}$$

and since $\binom{0}{(y_{i-1} \vee y_{i-2}) + 1} = 0$ for $i > 2$, as $(y_{i-1} \vee y_{i-2}) + 1 \geq 1$, we must have $C = 0$. Now

$$y_i = (y_{i-1} \vee y_{i-2}) + 1 \tag{5.2}$$

is something of an unusual equation, as the bitwise boolean and arithmetic operations certainly do not associate or distribute with each other. Iterating the previous equation, we have

$$\begin{aligned} \{y_i\}_{i=0}^{\infty} = \{&0, 0, 1, 2, 4, 7, 8, 16, 25, 26, 28, 31, 32, 64, 97, 98, \\ &100, 103, 104, 112, 121, 122, 124, 127, 128, \ldots\}, \end{aligned}$$

which when 'differenced' reveals some structure. Let

$$\begin{aligned} z_i &:= \Delta_{i+} y_i \\ &:= y_{i+1} - y_i, \end{aligned}$$

then

$$\begin{aligned} \{z_i\}_{i=0}^{\infty} = \{&0, 1, 2, 3, 1, 8, 9, 1, 2, 3, 1, 32, 33, 1, 2, 3, 1, \\ &8, 9, 1, 2, 3, 1, 128, 129, 1, 2, 3, 1, 8, 9, 1 \ldots\}. \end{aligned}$$

If we write $n = mn'$ such that m is odd (so n' is the greatest power of 2 that divides $n > 0$, or $n' = 0$ if $n = 0$), then it appears that

$$\begin{aligned} z_{3n} &= 2(n')^2 \\ z_{3n+1} &= z_{3n} + 1 \\ z_{3n+2} &= 1 \end{aligned} \tag{5.3}$$

for all $n \geq 0$.

We note here that the sequence z_n is easily computed, given a binary representation of n. In this respect, we may regard the above as an exact, though not yet closed form solution of the cellular automata – computation of y_n still requires summation. For certain values of n, we may express the sum in a closed form. Firstly, consider

$$\{z_{3i}\}_{i=0}^{\infty} = \{0, 2, 8, 2, 32, 2, 8, 2, 128, 2, 8, 2, 32, 2, 8, 2, 512, \ldots\}.$$

If we take the first 2^n terms of this sequence, collect like terms (2's, 8's, 32's, ... up to 2^{2n-1}) and sum, we have

$$\begin{aligned} \sum_{i=0}^{2^n-1} z_{3i} &= 2^{n-1}(2) + 2^{n-2}(8) + 2^{n-3}(32) + \ldots + 2^0(2^{2n-1}) \\ &= 2^n + 2^{n+1} + 2^{n+2} + \ldots + 2^{2n-1} \\ &= 4^n - 2^n. \end{aligned} \tag{5.4}$$

Now consider

$$\begin{aligned} y_{3\cdot 2^n} &= \sum_{i=0}^{3\cdot 2^n-1} z_i \\ &= \sum_{i=0}^{2^n-1} [z_{3i} + z_{3i+1} + z_{3i+2}] \\ &= \sum_{i=0}^{2^n-1} [z_{3i} + (z_{3i} + 1) + (1)], \quad \text{by (5.3)}, \\ &= 2\sum_{i=0}^{2^n-1} z_{3i} + \sum_{i=0}^{2^n-1} 2 \\ &= 2(4^n - 2^n) + 2^n(2), \qquad \text{by (5.4)}, \end{aligned}$$

and so we have a 'partially' closed solution of the cellular automaton:

$$y_{3\cdot 2^n} = 2 \cdot 4^n. \tag{5.5}$$

This may be regarded as a significant achievement, as I am not aware that such closed form solutions (partial or otherwise) exist for non-trivial nonlinear cellular automata. We may interpret equation (5.5) as

$$y_n \in \Theta(n^2), \tag{5.6}$$

which has consequences for both the periodicity and the rate of expansion of this particular cellular automaton. We will consider this in greater detail in following discussions.

5.1.1 Strange Counters

Here we give a curious 'arithmetic' interpretation of the previous cellular automaton. We may apply the analysis to a class of rules of higher degrees, iterated from special initial conditions, which we would expect to have similar behaviour. In general these have the form

$$x_i^{t+1} = x_i^t + x_{i-1}^t x_{i-2}^t \cdots x_{i-d}^t$$

where we take

$$[x_{-d}^0 \ldots x_{-1}^0] = [1 \ldots 1],$$

that is, d 1's positioned to the left of the origin this time, departing from our usual convention. These may be viewed as as finite binary *counters* – take the nonzero component to the right of the origin as a binary number written backwards, and it is of course incremented by 1 each iteration up to a point determined by d, where the nonlinear component of the rule represents the carry operation.

We state here as a conjecture that this class of rules, as we increment d, represent in the limit the lower bound to the rate which non-trivial nonlinear cellular automata may expand. Following [9], define the length of a finite state $x^t \in \mathcal{F}$, written $Q(x^t)$, to be the difference between the spatial coordinates of the leftmost and rightmost ones (plus 1):

$$Q(x^t) = \max(r - l : x_r^t = x_l^t = 1) + 1.$$

Now let

$$Q^t = \max(Q(x^0), \ldots, Q(x^t)).$$

We may use the set $\Theta(Q^t)$ – the set of functions bounded above and below in the limit by positive constant multiples of Q^t – as a measure of the *rate of expansion* of the cellular automaton.

For f_{66} iterated from a double seed, we look for the convex hull containing the generated pattern. This would seem to be identifiable (refer to Figure 5.1) by the states directly following a state consisting of a finite sequence of 1's of length divisible by 3, which occur at $t = 2^{2n+1}$

t	2^1	2^3	2^5	2^7	2^9
$Q_2^{t+1} - 1$	$3 \cdot 2^0$	$3 \cdot 2^1$	$3 \cdot 2^2$	$3 \cdot 2^3$	$3 \cdot 2^4$

From this we conclude that

$$\frac{\log\left(\frac{Q_2^{t+1}-1}{3}\right)}{\log\left(\frac{t}{2}\right)} \leq \frac{1}{2}$$

for all $t > 0$, with the equality holding when $t = 2^{2n+1}$. Hence

$$\Theta(Q_2^t) = \Theta(\sqrt{t})$$

for this particular cellular automaton.

Now if we analyse the extension of f_{66} to the $d = 3$ counter f_{16386}, iterated from a triple seed, in a similar fashion (refer to Figure 5.2),

	2^1	2^4	2^7	2^{10}	2^{13}
$Q_3^{t+1} - 1$	2^2	2^3	2^4	2^5	2^6

we find that

$$\Theta(Q_3^t) = \Theta(\sqrt[3]{t}).$$

More generally, for a counter of degree d, we would expect

$$\Theta(Q_d^t) = \Theta(\sqrt[d]{t}).$$

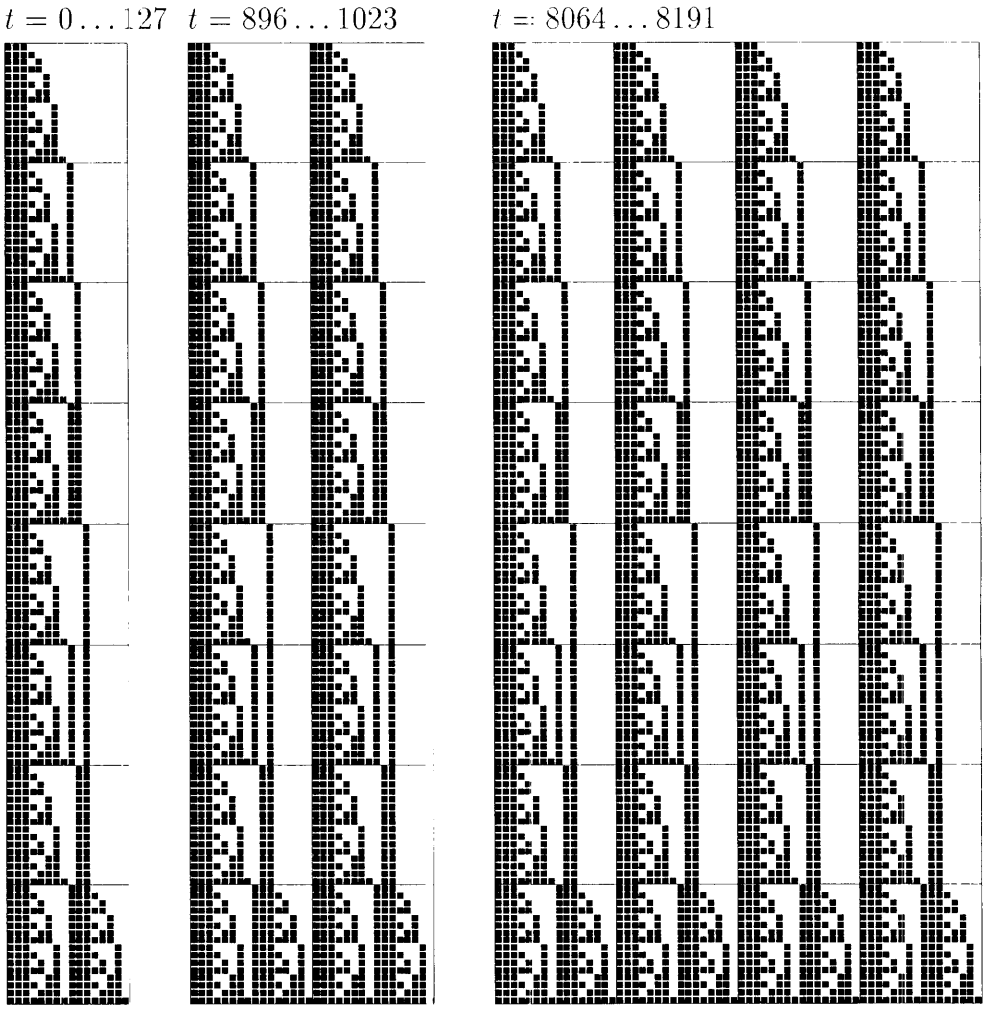


Figure 5.2: Extension of $d = 2$ rule f_{65} to $d = 3$ rule f_{16386}, this time iterated from a 'triple seed', which is behaviourally similar.

Observing that, for $t > 1$,

$$\frac{d}{dt}\sqrt[d]{t} = \frac{\sqrt[d]{t}}{t} > \frac{1}{t} = \frac{d}{dt}\log(t),$$

then we have an exclusive lower bound (where equality is approached in the limit as $d \to \infty$),

$$\Theta(\mathcal{Q}_d^t) = \Theta(\sqrt[d]{t}) \supset O(\log(t)).$$

It is worth noting here that the linear rule f_6 also fits the definition of a $d = 1$ counter, with rate of expansion $\Theta(\sqrt[1]{t}) = \Theta(t)$. (We might also view the $d = 0$ generative rule $f_3(x) = 1 + x$ as a pathological counter with an infinite rate of expansion.)

5.1.2 One Dimensional Cellular Automata Can't Count

We may make a few conjectures based on the previous discussion. We begin by noting that the counters all have one feature in common – they run out of fingers. In the limit as $d \to \infty$ we have an 'imaginary cellular automaton' which can perform an infinite carry, but which 'loses' the origin according to our definition. We could overcome this problem by redefining the initial state to be $x^0 = [1]$, and adjusting the infinite rule to take the first occurrence of a 1 to be a marker of the origin. The pattern generated would have rate of expansion $\Theta(\log(t))$, and would incrementally generate every finite sequence (up to spatial translation), therefore having a *minimal non-constant rate of expansion*. From this we conclude:

Proposition 5.1 Any cellular automaton with rate of expansion $\Theta(Q^t) \subset O(\log(t))$ is trivial; that is, satisfies $\Theta(Q^t) = \Theta(1)$.

The proof is straightforward. If such a cellular automata does not repeat any finite state (up to spatial translation) then it must satisfy $O(Q^t) \supseteq O(log(t))$, and hence a contradiction. If it does repeat a finite state, it will be ultimately temporally periodic (up to spatial translation), and therefore $\Theta(Q^t) = \Theta(1)$. Now just for the fun of it,

Conjecture 5.2 No cellular automaton exists with rate of expansion $\Theta(Q^t) = \Theta(log(t))$.

'Proof': All finite counters run out of fingers.

5.2 Quasi-Linear Cellular Automata

5.2.1 Algebraic Classification of Cellular Automata

One important consequence of being able to express any cellular automata rule as a polynomial is a natural algebraic classification of the rules, bearing analogy to methods of classification of partial differential equations. If we may associate certain behavioural characteristics with certain algebraic properties, then we have taken a considerable step towards the 'solution' of the forward problem. This will be achieved with particular assistance of the results of Chapter 4, which are proposed as the natural framework for the calculus of cellular automata.

The results here are strongly motivated by [9], which begins with analysis of a more 'difficult' region of behaviour – that is, temporal sequences which have velocities $0 < v < d$. The results are achieved with direct reference to the rule table. The approach here is to start from the 'outer' temporal sequences and 'work in', with a view to understanding some of the behaviour that [9] has identified on a specific class of rules, and to outline some techniques for the analysis of nonlinear cellular automata in general. Further, within the framework outlined previously, the intention here is to find results which are associated with algebraic properties of the rule, and therefore easily defined and generalised to higher degrees.

To begin with, every non generative rule f of degree d may be decomposed (uniquely)

$$f(wx \ldots yz) = g(wx \ldots y) + h(wx \ldots y)z$$

such that g is non-generative, and g and h are both functions of (implicitly) d variables. Here, to simplify notation, we have adopted the convention that z is the *rightmost* variable in the domain of f, and w the *leftmost*. Wherever possible, we will omit indices in this fashion. The classification and analysis of cellular automata begins with consideration of the properties of g and h. [2]

[2] We may view the function $h(wx \ldots y)$ as the formal partial derivative of the function $f(wx \ldots yz)$ w.r.t. z; that is,

$$\frac{\partial}{\partial z} f(wx \ldots yz) = \frac{\partial}{\partial z}(g(wx \ldots y) + h(wx \ldots y)z)$$

5.2.2 Quasi-Linear Cellular Automata

It would seem natural to begin with analysis of rules which have partial linear dependence upon one or more of their variables. We define a rule f to be *quasi-linear* if it depends linearly on either (or both) the *rightmost* or *leftmost* variables in its domain. If we need to be specific, we will use the terms *r-quasi-linear* or *l-quasi-linear* respectively. An r-quasi-linear rule may be written

$$f(wx \ldots yz) = g(x \ldots y) + z. \tag{5.7}$$

These have been studied in the case $d = 2$ by Jen [9], who refers to them as *injective* in the variable z, by deriving results directly from the rule table. We use the term "quasi-linear" in view of the relationship with quasi-linear differential equations. We present an analysis here which aims to make precise the sensitivity to initial conditions and strictly periodic behaviour of the temporal sequences of velocity 0 generated by these these rules, which for simplicity we will refer to as $x_i (= T_i(0))$. [9] looks at the aperiodicity of temporal sequences $T_i(1)$.

An r-quasi-linear rule may be written formally

$$\Delta_{t+} x_i^t = g(x_{i-d}^t \ldots x_{i-1}^t), \tag{5.8}$$

and therefore 'summated',

$$x_i^t = \Sigma_{t+} g(x_{i-d}^t \ldots x_{i-1}^t) + C \binom{t}{0}, \tag{5.9}$$

where C is determined from x^0. It follows that each of the x_i^t depend linearly – that is, unconditionally – upon x_i^0.

More needs to be known about the feasibility of actually 'integrating' or 'solving' (5.9) for the temporal sequences $x_0^t, x_1^t, \ldots$ for various initial conditions and functions g. This is straightforward for linear rules, but has only been achieved for nonlinear (non-trivial) cellular automata in the case of the counters in the previous section,

$= h(wx \ldots y).$

This suggests notation and treatment similar to [23], except that this text refers to Boolean rather than formal field derivatives.

which are defined by $g(wx\ldots y) = wx\cdots y$. These have the rather nice property that when iterated from the finite state consisting of d consecutive 1's, the temporal sequences may be written

$$x_i^t = \begin{pmatrix} t \\ y_i \end{pmatrix}$$

for some sequence y_i. We refer to this as a *mono-spectral* solution, which is not common amongst cellular automata in general, and tends to occur only for some rules iterated from very restricted classes of states.

The solution of 'multi-spectral' cellular automata – including the counter rules iterated from other states – would appear very difficult at this stage. This may be due either to inappropriate formalism, insufficient research towards this goal, or the inherent insolubility of nonlinear cellular automata. This returns us to the observation that cellular automata solve themselves, and any proposed solution should provide information which is not immediately apparent from the simple graphic iteration of the rule.

We now turn our attention to the periodicity of the temporal sequences generated by r-quasi-linear cellular automata, about which we may say a reasonable amount within this framework which is not too obvious; further, these discussions may be generalised for the consideration of the ultimate periodicity of temporal sequences of cellular automata in general.

5.2.3 Transform Analysis

Here we show that any r-quasi-linear cellular automata $(f, \mathcal{F}^-)$ is invertible, and that the temporal sequences of velocity 0 are periodic; more precisely, $x_i \in \mathcal{P}^{2^{\mathbb{N}}}$. This means we may view the temporal sequences under the temporal binomial transform defined in Chapter 4, which we will refer to as the *t-transform*. This is useful for the consideration of period doubling of temporal sequences, and the related matter of rate of expansion of quasi-linear $(f, \mathcal{F})$.

To begin with, we have the following (which is one reason why we refer to these cellular automata as "quasi-linear", in view of Proposition 4.1):

Proposition 5.3 All r-quasi-linear cellular automata $(f, \mathcal{F}^-)$ are invertible.

The proof comes simply from rewriting (5.8) in the form

$$x_i^t = x_i^{t+1} + g(x_{i-d}^t \dots x_{i-1}^t),$$

and then iterating for $i = 0, 1, 2, \dots$, where we have $x_{-1}^t = x_{-2}^t = \dots = 0$ by definition.[3]

Now that we have 'pinned down' the solution x_i^t for all $t, i \in \mathbb{Z}$, we may now show:

Proposition 5.4 Let $(f, \mathcal{F}^-)$ be an r-quasi-linear cellular automata of degree d; that is, f may be written in the form

$$f(x_{i-d} \dots x_i) = g(x_{i-d} \dots x_{i-1}) + x_i.$$

Given any initial state $x^0 \in \mathcal{F}^-$, then $x_i \in \mathcal{P}^{2^N}$ for all $i \in \mathbb{Z}$.

The proof is by induction on i: firstly, we have $x_i^t = 0$ for all $t \in \mathbb{Z}$ and $i < 0$, so clearly $x_i \in \mathcal{P}^{2^N}$ for $i < 0$, and hence $\hat{x}_i \in \mathcal{F}$ for $i < 0$. Now, suppose $x_{i-d}, \dots, x_{i-1} \in \mathcal{P}^{2^N}$, hence $\hat{x}_{i-d}, \dots, \hat{x}_{i-1} \in \mathcal{F}$, and let $y_i^t = g(x_{i-d}^t \dots x_{i-1}^t)$. By Theorem 4.7, [4] we must have $\hat{y}_i \in \mathcal{F}$. Now,

$$\begin{aligned} x_i^t &= \Sigma_{t+} y_i^t \\ &= \Sigma_{t+} \sum_j \binom{t}{j} \hat{y}_i^j \\ &= \sum_j \Sigma_{t+} \binom{t}{j} \hat{y}_i^j \\ &= \sum_j \left(\binom{t}{j+1} \hat{y}_i^j \right) + x_0^i \binom{t}{0} \\ &= \sum_{j'} \left(\binom{t}{j'} \hat{y}_i^{j'-1} \right) + x_0^i \binom{t}{0}, \end{aligned} \tag{5.10}$$

so clearly, $x_i \in \mathcal{P}^{2^N}$, and the result follows by induction.

[3] It is worth noting that such time inversion by iteration is an example of a *filter automata*, which have been studied.

[4] Which implies closure of temporal binomial sequences under bitwise multiplication in t.

This proposition is quite intuitive – it simply relies on the fact that $\mathcal{P}^{2^{\mathbb{N}}}$ is closed under bitwise products and sums, and also summation as discussed in the previous chapter. We note that equation (5.10) allows us to be reasonably precise about the t-transforms of the successive temporal sequences; that is, we can construct the t-transform $\hat{x}_i$ simply by performing a *right shift* on the t-transform of $g(x^t_{i-d} \dots x^t_{i-1})$ and adding x^i_0 to the zeroth position. The action of g under t-transformation is more difficult, but we can place bounds using Theorem 4.7 as follows.

Suppose two temporal sequences $x, y \in \mathcal{P}^{2^{\mathbb{N}}}$ have t-transforms $\hat{x} \in \mathcal{F}_{l_x,r_x}$ and $\hat{y} \in \mathcal{F}_{l_y,r_y}$. It follows from Theorem 4.7 that $\widehat{xy} \in \mathcal{F}_{l,r}$ where $l \geq l_x \vee l_y \geq \max(l_x, l_y)$ and also $\lceil \log(r+1) \rceil \leq \max(\lceil \log(r_x+1) \rceil, \lceil \log(r_y+1) \rceil)$ (we may interpret $l > r$ as a zero product). This is intuitive: l is equal to the number of leading zeros in the bitwise product $x^0 y^0 = x^1 y^1 = \cdots = x^{l-1} y^{l-1} = 0$ which is of course bounded below by $\max(l_x, l_y)$. Also, for $l \leq r$, $2^{\lceil \log(r+1) \rceil}$ is the least temporal period of $x^t y^t$, which is bounded above by $\mathrm{lcm}(2^{\lceil \log(r_x+1) \rceil}, 2^{\lceil \log(r_y+1) \rceil}) = 2^{\max(\lceil \log(r_x+1) \rceil, \lceil \log(r_y+1) \rceil)}$.

Similarly, the t-transform of the bitwise sum of two temporal sequences may be bounded: by $\widehat{x+y} \in \mathcal{F}_{l,r}$ where $\lceil \log(r+1) \rceil \leq \max(\lceil \log(r_x+1) \rceil, \lceil \log(r_y+1) \rceil)$ as before, but now $l \geq \min(l_x, l_y)$ Summarising, we have:

Proposition 5.5 Let $w, x, \dots, y \in \mathcal{P}^{2^{\mathbb{N}}}$ be temporal sequences with t-transforms $\hat{w} \in \mathcal{F}_{l_w,r_w}$ and similarly for $\hat{x}, \dots, \hat{y}$. If we let $z = g(wx \dots y)$ for some (bitwise polynomial) function g, then $\hat{z} \in \mathcal{F}_{l,r}$, where

$$\lceil \log(r+1) \rceil \leq \max(\lceil \log(r_w+1) \rceil, \lceil (r_x+1) \rceil, \dots, \lceil \log(r_y+1) \rceil),$$

and

$$l \geq \min(l_w, l_x, \dots, l_y).$$

5.2.4 Period Doubling

We want to say more about the conjectured minimal rate of expansion of the counters, for which Proposition 5.5 is useful. We first consider the question of period increase as we move from left to right through the temporal sequences. We say the temporal

sequence x_i has *doubled in period* (over $x_0, x_1, \ldots, x_{i-1}$) if

$$P(x_i) = 2P(x_0, x_1, \ldots, x_{i-1}).$$

Proposition 5.5 implies that the period of $g(wx\ldots y)$ is bounded above by the maximum (=lcm) of the periods of $w, x, \ldots, y$. This may be written

$$P(g(wx\ldots y)) \leq P(w, x, \ldots, y).$$

Period doubling may therefore only occur as a result of summation. Simplify (5.9) to

$$z = \sum_{t+} g(wx\ldots y),$$

that is, let $w = x_{i-d}$, $x = x_{i-d+1}, \ldots,$ $z = x_i$. Supposing $g(\widehat{wx\ldots y}) \in \mathcal{F}_{l,r}$, it follows from (5.10) that $\hat{z} \in \mathcal{F}_{l',r+1}$, where $l' = 0$ if $z^0 = 0$ and $l' = l + 1$ otherwise. Now $P(z) = 2^{\lceil \log(r+2) \rceil}$, and may double only if $\lceil \log(r+2) \rceil = \lceil \log(r+1) \rceil + 1$, which implies that we have

$$r = 2^{\lceil \log(r+1) \rceil} - 1, \tag{5.11}$$

and

$$P(z) = 2(r+1). \tag{5.12}$$

We then have following property of temporal sequences that double in period.

Proposition 5.6 Let $(f, \mathcal{F}^-)$ be an 1-quasi-linear cellular automata, with $x^0 \in \mathcal{F}^-$ given, and let $P_i = P(x_0, \ldots, x_i)$. If x_i doubles in period, that is $P_i = 2P_{i-1}$, then

$$x_i^t + \binom{t}{0} = x_i^{t+P_{i-1}}.$$

So if period doubling occurs, the two possible sequences generated (depending on the initial state) are the same up to the temporal translation $t \to t + P_{i-1}$. The proof comes from equations (5.11) and (5.12), where we consider

$$\begin{aligned} x_i^t &= \binom{t}{r+1} + \sum_{j=0}^{r} \binom{t}{j} \hat{x}_i^j \\ &= \binom{t}{P_{i-1}} + \sum_{j=0}^{P_{i-1}-1} \binom{t}{j} \hat{x}_i^j, \end{aligned}$$

The sum consists of periodic sequences which all have periods that divide P_{i-1}, and hence we need to show that

$$\binom{t}{P_{i-1}} + \binom{t}{0} = \binom{t + P_{i-1}}{P_{i-1}}.$$

This may be achieved by putting $P_{i-1} = 2^m$ for some $m > 0$, and then using Theorem 4.3.

5.2.5 Minimal Rate of Expansion Revisited

The reason which we have considered bounds on l in these discussions is the obvious relationship between the rate of expansion of a cellular automata on finite states and the number of leading zeros of the temporal sequences. We return now to the example of the counters, and attempt to shed some more light on the conjecture that they represent the slowest expanding non-trivial cellular automata.

Proposition 5.7 Let $(f, \{x^t\}_{t=0}^{\infty})$ for some $x^0 \in \mathcal{F}$ be a non-generative non-trivial quasi-linear cellular automaton, and suppose that $\hat{x}_i \in \mathcal{F}_{l_i,r_i}$, then

$$\in (Q^{l_n}) = \Theta(l_{Q^n}) = \Theta(n).$$

Proof: Assume without loss of generality that f is r-quasi-linear, and that $x^0 \in \mathcal{F}_{0,r}$ for some r, and then we have $x_0^t = \binom{t}{0}$. Now given any $i' > r$, there is a least $t > 0$ such that $x_{i'}^t = 1$. (If no such t exists, then the cellular automaton is trivial.) Now find the greatest $i \geq i'$ such that $x_i^t = 1$ (which must exist, otherwise the cellular automaton is generative). For these values of t and i we have $Q^t = i + 1$ and $l_i = t$, from which the result follows.

The idea here is straightforward: both functions 'find' the boundary of the cellular automaton pattern, Q^t from the 't-axis', and l_i from the 'i-axis'. Hence, one function is bounded by the 'inverse' of the other. For example, we have already seen for the $d = 2$ counter that $y_n (= l_n) \in \Theta(n^2)$ (Equation (5.6)), and that $Q_2^n \in \Theta(\sqrt{n})$ (Section 5.1.1).

More generally, all counters are mono-spectral as discussed (refer to Section 5.2.2), which means that $l_i = r_i$. Now, bearing in mind Theorem 4.7, any function

$$g(\binom{t}{y_{i-d}} \cdots \binom{t}{y_{i-1}})$$

has t-transform $\widehat{g(\cdots)} \in \mathcal{F}_{l,r}$ where $r \leq y_{i-d} \vee \cdots \vee y_{i-1}$. In particular, the counters defined by

$$\begin{aligned} g(\binom{t}{y_{i-d}} \cdots \binom{t}{y_{i-1}}) &= \binom{t}{y_{i-d}} \cdots \binom{t}{y_{i-1}} \\ &= \binom{t}{y_{i-d} \vee \cdots \vee y_{i-1}} \end{aligned}$$

achieve this limit, which represents the maximal rate of increase for $r_i = y_i = (y_{i-d} \vee \cdots \vee y_{i-1}) + 1$, and therefore the maximal rate of increase for $l_i \leq r_i$.

Proposition 5.7 states the inverse relationship between Q^t and l_i in the case of quasi-linear cellular automata: so Q^t is minimal when l_i is maximal. An argument along these lines would have to be generalised to consider non-quasi-linear rules, and also 'multi-spectral' cases, which is beyond the scope of this thesis. Nevertheless, these results suggest an interesting relationship between two different forms of stability.

We may view a minimal rate of expansion as *temporally* stable behaviour. By Proposition (5.7) and subsequent discussions, this is strongly (and inversely) linked to a maximal rate of period increase, which we may view as *spatially* unstable behaviour. To be clear what we mean by 'spatial' behaviour, we consider the solution of a cellular automaton as a static object which may be traversed in either space or time. Here we move from left to right through the temporal sequences of velocity 0, and consider their progression (under the t-transform) in much the same way that we consider the actual time evolution of a cellular automaton.

Later in this chapter we consider the opposite extreme: we find a class of cellular automata that have a maximal rate of expansion, and which appear to have a minimal (non-constant) rate of period increase as we move from left to right through the temporal sequences.

5.3 More Generally

Here we discuss how the types of analysis previously outlined may be applied to non-linear cellular automata in general. We again restrict our attention to the behaviour of the temporal sequences of velocity 0, which in the case of the r-quasi-linear cellular automata are characterised as *strictly* periodic, and unconditionally sensitive to the initial state; that is, x_i^t depends on x_i^0 regardless of $x_{i-1}, x_{i-2}, \ldots$, which we may view as unstable behaviour.

For finite cellular automata in general, these temporal sequences are *ultimately* periodic, where the dependence of x_i^t upon x_i^0 may be conditional upon $x_{i-1}, x_{i-2}, \ldots$. Here we outline a stability analysis of these temporal sequences, with the purpose of relating their limiting behaviour to algebraic properties of the local rule.

5.3.1 Limiting Behaviour and Stability

To begin with we have the following result which may be regarded as well known and is easily shown by induction on the spatial index i (similarly to Proposition 5.4).

Proposition 5.8 The temporal sequences of velocity 0 of any non-generative $(f, \mathcal{F}^-)$ are ultimately periodic, each with (right) period 2^m for some m.

Hence, each temporal sequence x_i converges to some unique periodic *limit sequence*, which we will denote $x_i' \in \mathcal{P}^{2^{\mathbb{N}}}$ that is, there exists some Q such that $x_i^t = x_i'^t$ for all $t \geq Q$.

In the following discussion we will again omit indices wherever possible to simplify notation. Given some rule written in the form

$$f(wx \ldots yz) = g(wx \ldots y) + h(wx \ldots y)z,$$

and suppose that we fix $w, x, \ldots, y$, then the site values z^t depend upon z^0 for all $t \geq 0$ if and only if $h(w^t x^t \ldots y^t) = 1$ for all $t \geq 0$, in which case we may write

$$z^t = \textstyle\sum_{t+} g(w^t x^t \ldots y^t) + C \binom{t}{0}.$$

where C is determined from z^0. We refer to temporal sequence z as being *unstable* in this case, and *stable* otherwise,[5] for if $h(w^s x^s \dots y^s) = 0$ for some $s \geq 0$, then it is easy to show that z^t is independent of z^0 for all $t > s$.

This definition of stability may be extended to the limit sequences; we refer to z' as unstable if $h(w'^t x'^t \dots y'^t) = 1$ for all $t \in \mathbb{Z}$,[6] and stable otherwise. It follows from these definitions that z unstable implies z' unstable, or equivalently z' stable implies z stable.

5.3.2 Period Doubling of Limit Sequences

If z' is unstable, then we may write

$$z'^t = \Sigma_{t+} g(w'^t x'^t \dots y'^t) + C' \begin{pmatrix} t \\ 0 \end{pmatrix},$$

where C' depends on z^0 if and only if z is unstable. As for the periodic temporal sequences, period doubling of limit sequences may only occur when z' is unstable; that is, when $h(w'^t x'^t \dots z'^t) = 1$ for all t. Analysis of this necessary condition is reasonably straightforward along the lines of previous discussions, however sufficient conditions given knowledge of both the rule and the initial state are somewhat harder to achieve, and will not be discussed here.

Using a similar argument as for Proposition 5.6, we do have the following consequence of period doubling:

Proposition 5.9 For any non-generative $(f, \mathcal{F}^-)$, with $x^0 \in \mathcal{F}^-$ given, let $P_i = P(x'_0, \dots, x'_i)$. If x'_i doubles in period, that is $P_i = 2P_{i-1}$, then

$$x_i'^t + \begin{pmatrix} t \\ 0 \end{pmatrix} = x_i^{t+P_{i-1}}.$$

[5] This definition considers only one of many types of stability to be considered, and perhaps should be called 'partial stability', since a stable temporal sequence x_i may depend on $x^0_{i-1}, x^0_{i-2}, \dots$ according to this definition; in this thesis we are only concerned with the stability property defined here, hence the narrow definition.

[6] Noting that the limit sequences constitute a solution of the cellular automata which may be extended in the $-t$ direction.

5.4 Weakly Injective Rules

The limiting behaviour of temporal sequences of velocity 0 of cellular automata $(f, \mathcal{F}^-)$ is quite often trivial,[7] that is $x'_i = 0$ for all i (and t). The exceptions turn out to be weakly injective rules, which are defined as follows. Suppose we have some rule f written in polynomial form

$$f(wx \ldots yz) = g(wx \ldots y) + h(wx \ldots y)z,$$

then f is said to be *weakly injective* in the variable z if the function h is generative.

Weakly injective rules are easily identified as having the term 'z' in polynomial form, or in terms of a rule table $f(00\ldots01) = 1$, thus having both rule numbers and polynomial numbers that are even (non-generative) but not divisible by 4. For example, linear and quasi-linear rules are weakly injective. Generally speaking, these rules constitute the candidates for 'interesting' behaviour of limit sequences in view of the following proposition.

Proposition 5.10 Let $(f, \mathcal{F})$ be some (finite non-generative) cellular automaton, where the initial state $x^0 \in \mathcal{F}$ is nonzero, then there exists some nonzero limit sequence x'_i if and only if f is weakly injective in its rightmost variable.

Proof: Suppose f is weakly injective in its rightmost variable, and that $x^0_0 = 1$ (without loss of generality), then $x^t_0 = x'^t_0 = \binom{t}{0} = 1$ for all $t \in \mathbb{N}$. Conversely, suppose x'_i is the leftmost nonzero limit sequence. Then there exists some $t \geq 0$ such that

$$\begin{aligned} &= x'^{t+1}_i \\ &= f(x'^t_{i-d} \ldots x'^t_{i-1} x'^t_i) \\ &= f(0 \ldots 0 x'^t_i), \end{aligned}$$

where non-generativity of f implies $x'^t_i = 1$, so we have $f(0\ldots01) = 1$ as required, and the proof is complete.

[7] Which is *not* to say that the cellular automaton is necessarily trivial, as there are other temporal sequences/frames of reference to consider.

A rule is referred to as *doubly* weakly injective if it is weakly injective in both its leftmost and rightmost variables. Doubly weakly injective finite cellular automata of degree d satisfy $Q^t = Q^0 + dt$, and therefore achieve the maximum rate of expansion of cellular automata of degree d, effectively 'filling' the combined domain of dependence of the initially nonzero sites

$$D(x^0) \equiv \bigcup_{x_i^0=1} D(x_i^0)$$

with often interesting behaviour.

We will illustrate this discussion with an analysis of stability and period doubling of the temporal sequences of a well known doubly weakly injective rule.

5.4.1 Wolfram's Rule 30

This rule has attracted as much interest as any other amongst the nonlinear one-dimensional binary $d = 2$ cellular automata, which are referred to as 'elementary' in the more widely accepted terminology. It is known mostly for its ability to generate high quality pseudo-random sequences from a seed, [26] and represents one of few 'interesting' $d = 2$ cellular automata on an infinite lattice.

The rule may be written as a polynomial,

$$f_{30}(xyz) = x + y + yz + z$$

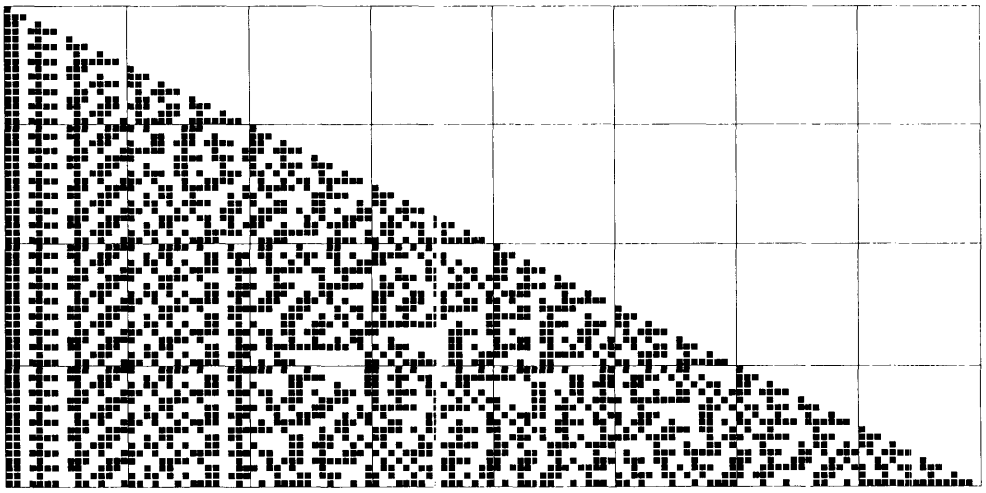
where we make the observation that the polynomial number is equal to the (Wolfram) rule number. Clearly, this rule is l-quasi-linear, and the temporal sequences of velocity 2 are therefore understood as previous. We look at the temporal sequences of velocity 0 which have interesting stability properties.

Referring to Figures 5.3 and 5.4, it would seem that the temporal sequences converge to the same set of x_i' up to temporal[8] translation from any (finite nonzero) initial state. This stable behaviour is quite interesting in view of the complexity of the limit sequences x_i' generated by f_{30}.

We define this property precisely as follows:

[8] and strictly speaking spatial, unless we stipulate (with no loss of generality) that $x_0^0 = 1$.

$t = 0 \ldots 63$



$t = 192 \ldots 255$

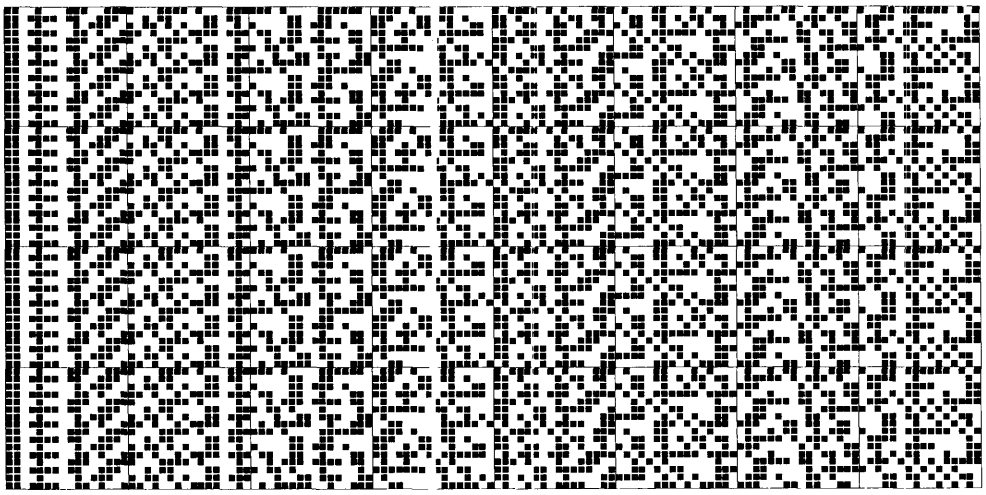


Figure 5.3: Rule f_{30} iterated from a seed.

Property P1: Given any two nonzero initial states $x^0, y^0 \in \mathcal{F}^-$, where $x_0^0 = y_0^0 = 1$ is assumed, then a rule f which is weakly injective in its rightmost variable is said to have property **P1** if the limit sequences x_i' of $(f, \mathcal{S}_1 = \{x^t\}_{t=0}^{\infty})$ and y_i' of $(f, \mathcal{S}_2 = \{y^t\}_{t=0}^{\infty})$ satisfy the following: given any $j \geq 0$, there exists $s_j \in \mathbb{Z}$ such that for all $i \leq j$ and $t \in \mathbb{Z}$,

$$x_i'^t = y_i'^{t+s_j}. \tag{5.13}$$

Now, it would seem that this property is related to another observed characteristic of f_{30}. The unstable limit sequences are easily identified for this rule: if we write

$$f_{30}(x'y'z') = g_{30}(x'y') + h_{30}(x'y')z' = x' + y' + (1 + y')z',$$

then we see z' is unstable if and only if $h_{30}(x'y') = 1 + y' = 1$; that is $y' = 0$ (or more precisely $y'^t = 0$ for all t). These unstable sequences occur less frequently as we move from left to right, and when they do, would seem to be characterised by *unconditional period doubling*, which we define as follows:

Property P2: Given any initial state $x^0 \in \mathcal{F}^-$, where $x_0^0 = 1$ is assumed, then a rule f which is weakly injective in its rightmost variable is said to have property **P2** if the limit sequences x_i' of $(f, \mathcal{S} = \{x^t\}_{t=0}^{\infty})$ satisfy the following: for $i > 0$, if x_i is unstable then

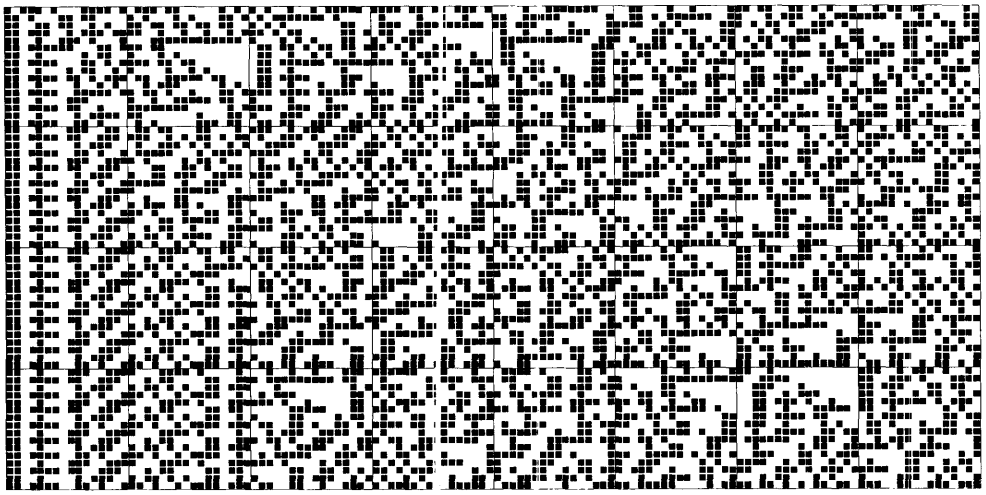
$$P(x_i') = 2P(x_0', x_1', \ldots, x_{i-1}'). \tag{5.14}$$

We then have the following result for all rules which are weakly injective in the rightmost variables (that is, all rules which have nonzero limit sequences by Proposition 5.10):

Proposition 5.11 Let f be a non-generative rule which is weakly injective in its rightmost variable, then f has property **P1** if and only if f has property **P2**.

Proof: (**P1** $\Leftarrow$ **P2**) Assume f has property **P2**, show **P1** by induction on j. Given $x^0, y^0 \in \mathcal{F}^-$ such that $x_0^0 = y_0^0 = 1$, then we have $x_0 = x_0' = y_0 = y_0' = 1$, so (5.13) is

$t = 0 \ldots 63$



$t = 192 \ldots 255$

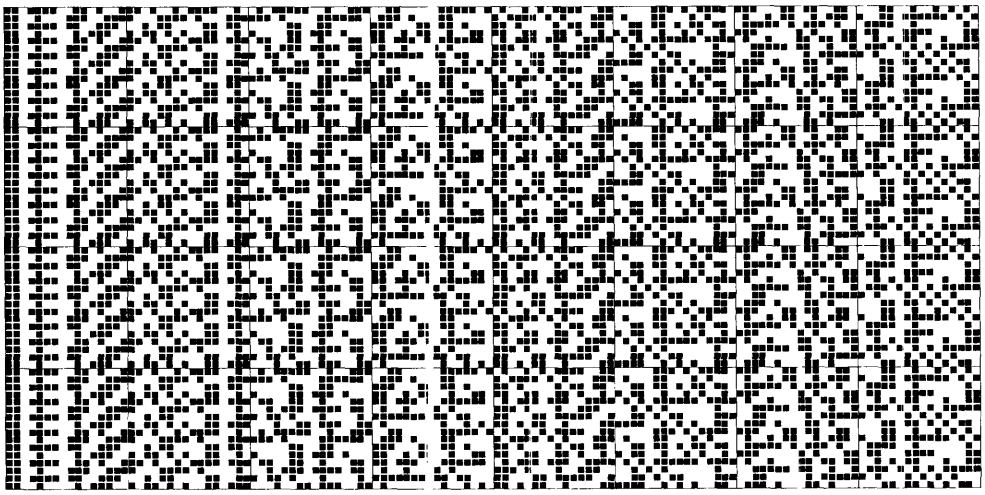


Figure 5.4: Rule f_{30} iterated from a random finite state. The limit sequences are the same up to temporal translation as the iteration from a seed (refer to Figure 5.3)

true for $j = 0$ by taking $s_0 = 0$. Now assume (5.13) for some $j \geq 0$. If x'_{j+1} stable, then y'_{j+1} stable, in which case (5.13) is true by taking $s_{j+1} = s_j$, as stability implies $x'^t_{j+1} = y'^{t+s_j}_{j+1}$. If x'_{j+1} is unstable, then y'_{j+1} unstable, in which case there are two possibilities, (by summation or $g(x'^t_{j-d+1} \dots x'^t_j) = g(y'^{t+s_j}_{j-d+1} \dots y'^{t+s_j}_j)$)

$$x'^t_{j+1} = y'^{t+s_j}_{j+1} + C' \begin{pmatrix} t \\ 0 \end{pmatrix},$$

depending on C'. If $C' = 0$ then we may take $s_{j+1} = s_j$. If $C' = 1$ then we take $s_{j+1} = s_j + P(x'_0 \dots x'_j)$ by Proposition 5.9.

(**P1** $\Rightarrow$ **P2**) Assume f has property **P1**. Now assume, for the sake of contradiction, that f does not have property **P2**. Then there exists some initial state x^0 and hence limit sequence x'_j for some $j \geq 0$ which is unstable but does not double in period. Then we have $P(x'_j) \leq P(x'_0 \dots x'_{j-1}) = P(x'_0 \dots x'_j) = P$, say. Now let $y'_i = x'_i$ for all $i < j$, and let $y'^0_i = y'^0_i$ for all $i < j$. Then the sequence y_j is unstable. Hence choose y^0_j such that $y^t_j = y'^t_j = x'^t_j + \begin{pmatrix} t \\ 0 \end{pmatrix}$, and it follows that $P(y'_j) = P(x'_j) \leq P = P(y'_0 \dots y'_j)$. By property **P1** there must exist s_j such that $x'^t_i = y'^{t+s_j}_i$ for all $t \in \mathbb{Z}$ and $i \leq j$. Further, this s_j may be chosen to satisfy $0 \leq s_j < P$. We can't have $s_j = 0$, as we have chosen $y'^0_j \neq x'^0_j$. Nor can we have $0 < s_j < P$, for then $P | s_j < P$. Hence, a contradiction.

Given the nature of the previous proof, we offer an heuristic one. If x'_j unstable, then for **P1** to be true, we must have period doubling so that the two possible limit sequences x'_j and $x'_j + 1$ may be temporally 'phased in' by a multiple of $P(x'_0 \dots x'_{j-1})$. Conversely, if we always have period doubling of unstable limit sequences x_j, then by Proposition 5.9 we may always 'phase' the two possible limit sequences without disturbing the 'phase' relationship of $x'_0, \dots, x'_{j-1}$.

Having proved Proposition 5.11, we still have not shown that f_{30} actually has the stated properties. This proves difficult, but the above discussion implies that we may verify the phase property (**P1**) by iteration of f_{30} *from any nonzero finite state.* That is, all we have to do is isolate the unstable sequences, which is easy in the case of f_{30} (look for $y' = 0$), establish that they undergo period doubling (**P2**), and then the

limiting behaviour (within a finite spatial region of interest) from any initial state is known.

5.4.2 Unbounded Periodicity for Rule 30

The properties discussed in the previous section, though verifiable for a finite set of limit sequences, appear hard to prove in the limit as $i \to \infty$. Here we present a result for f_{30} which applies to all l-quasi-linear cellular automata which are weakly injective in the rightmost variable, stating that the period of the limit sequences increases without bound as $i \to \infty$.

Firstly, any l-quasi-linear rule of degree d may be written formally

$$x_i^{t+1} = x_{i-d}^t + g(x_{i-d+1}^t \dots x_i^t).$$

Denote the forward temporal shift operator by E_{t+}, that is, $(E_{t+}z)^t = z^{t+1}$. We may now omit the temporal index, and rearrange to get

$$x_{i-d} = g(x_{i-d+1} \dots x_i) + E_{t+}x_i. \tag{5.15}$$

This equation, though easily achieved, has significant consequences for the limiting sequences:

Proposition 5.12 Let $(f, \mathcal{F}^-)$ be an l-quasi-linear cellular automaton of degree $d > 0$. If for some $j \in \mathbb{Z}$ and $R > 0$ we have

$$(x'_{j-d+1}, \dots, x'_j) = (x'_{j-d+1+R}, \dots, x'_{j+R}), \tag{5.16}$$

then $x'_i = x'_{i+R}$ for all $i \leq j$.

In other words, whenever there is duplication of d consecutive limit sequences, such duplication must occur for all limit sequences to the left. The proof comes simply by iterating (5.15) 'backwards', as it were: assume (5.16) for some $j \in \mathbb{Z}$, then consider

$$\begin{aligned} x'_{j-d} &= g(x'_{j-d+1} \dots x'_j) + E_{t+}x'_j \\ &= g(x'_{j-d+1+R} \dots x'_{j+R}) + E_{t+}x'_{j+R} \\ &= x'_{j-d+R}. \end{aligned}$$

the first and last equalities by 5.15). The result then follows inductively.

We may now show

Proposition 5.13 Let $(f, \mathcal{F}^-$ be an l-quasi-linear cellular automaton of degree $d > 0$ which is weakly injective in its rightmost variable. Given any nonzero initial state $x^0 \in \mathcal{F}^-$, then $P(x'_i)$ is unbounded as $i \to \infty$.

Proof: assume, without loss of generality, that $x'_0 = 1$ (refer to proof of Proposition 5.10). Now suppose, for the sake of contradiction, that $P(x'_i)$ is bounded as $i \to \infty$. Then there exist $M, m \in \mathbb{N}$ such that $P(x'_i) < M$ for all $i > m$; that is, $x'_i \in \mathcal{P}^{2^{\lceil \log M \rceil}}$ for all $i > m$. There are only finitely many such sequences, so we must have eventual duplication of d consecutive limit sequences. Hence, by Proposition 5.12, there will be some $R > 0$ such that $x'_i = x'_{i+R}$ for all $i \leq m$. In particular, $0 = x'_{-R} = x'_0 = 1$, and we have a contradiction.

The importance of this result is to show that we may achieve a strong statement of limiting behaviour for a very broad class of cellular automata of arbitrary degree from generic initial states, and such with relative ease, simply by rearranging the local rule. We now know that an l-quasi-linear cellular automaton will have highly complex limiting sequences if it is weakly injective in its rightmost variable, and trivial limiting sequences otherwise (Proposition 5.10).

Proposition (5.13) almost certainly has further consequences which are not considered here. It may bring us closer to proving Properties **P1** $\Leftrightarrow$ **P2** for f_{30}. The result is consistent with observations by [9] working in a different frame of reference, and makes precise and rigorous several of the results of [26] which were achieved numerically.

5.4.3 Algebraic Extrapolation

If we look to extend this behaviour to higher degrees, the polynomial form becomes particularly useful. We list some properties of

$$f_{30}(xyz) = x + y + (1 + y)z :$$

$t = 0 \ldots 127$

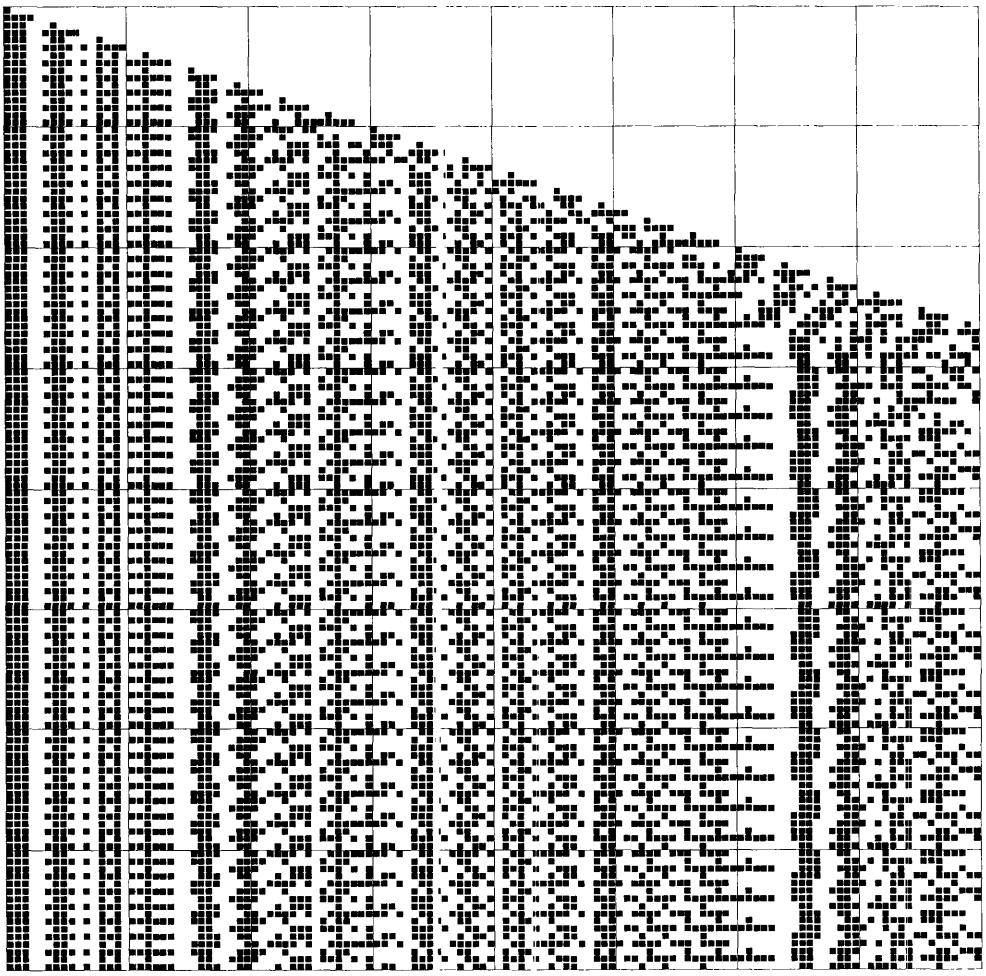


Figure 5.5: Rule f_{510} - an extrapolation of f_{30} - iterated from a seed. Observation of period doubling of unstable limit sequences, which are found to the right of pairs of columns of zeros, implies that limiting sequences always 'converge' to the lower pattern from any initial state (refer to Figure 5.6).

quasi-linearity in x; weak injectivity in z; z unstable if and only if $y = 0$ in which case $z = \sum_{t+}x$. If we try something of the form

$$f(wxyz) = w + e(xy) + (1 + e(xy))z,$$

where $e(xy) = 0$ if and only if $x = y = 0$, then we have

$$f_{510}(wxyz) = w + x + xy + y + (1 + x + xy + y)z.$$

Interestingly enough, this rule also satisfies $[f_{510}]^W = 510$. Referring to Figures 5.5 and 5.6, we find almost precisely the same behaviour for f_{510} as for f_{30}. Convergence to limiting sequences appears to be more rapid, whereas period doubling is slower.

5.5 Conclusions

There would seem to be some connection between these rules and the counters in the beginning of this chapter. If we observe that

$$f_{30}(xyz) = x + (y \vee z),$$

$$f_{510}(wxyz) = w + (x \vee y \vee z),$$

and so on, and we may write for the counters

$$f_{66}(xyz) = (x \wedge y) + z,$$

$$f_{16385}(wxyz) = (w \wedge x \wedge y) + z,$$

then we might offer the following conjectures on the basis of the discussions in this chapter:

Conjecture 5.14 The cellular automata $(f_n, \mathcal{F}^-)$ of degree $d > 0$, where f_n is of the form

$$f_n(x_d \ldots x_0) = x_d + (x_{d-1} \vee \cdots \vee x_0)$$

(that is $n = [f_n]^P = 2^{2^d+1} - 2$), and $x^0 \in \mathcal{F}^-$ an arbitrary (nonzero) initial state, each have the slowest non-constant rate of period increase of limit sequences of velocity 0 amongst all cellular automata of the same degree d.

Conjecture 5.15 The counters $(f_n, \{v^t\}_{i=0}^{\infty})$ of degree $d > 0$, where f_n is of the form

$$f_n(x_d \ldots x_0) = (x_d \wedge \cdots \wedge x_1) + x_0,$$

that is $n = [f_n]^P = 4^{2^d-1} + 2$, and $v^0 \in \mathcal{F}$ as defined in Section 5.1, each have the fastest rate of period increase of limiting sequences (= temporal sequences for r-quasi-linear cellular automata) amongst all cellular automata of the same degree d.

The cellular automaton which belongs to both of these groups is Pascal's Triangle; that is, f_6 iterated from a single seed. It would appear that f_6 is the key to understanding broad classes of nonlinear cellular automata, and it has often been observed that certain nonlinear rules tend to 'mimic' this rule in their behaviour (e.g. [9]). Here we have made this observation more precise, by employing the binomial sequences generated by f_6 as an actual basis for the solution or limiting analysis of nonlinear cellular automata.

$t = 0 \ldots 127$

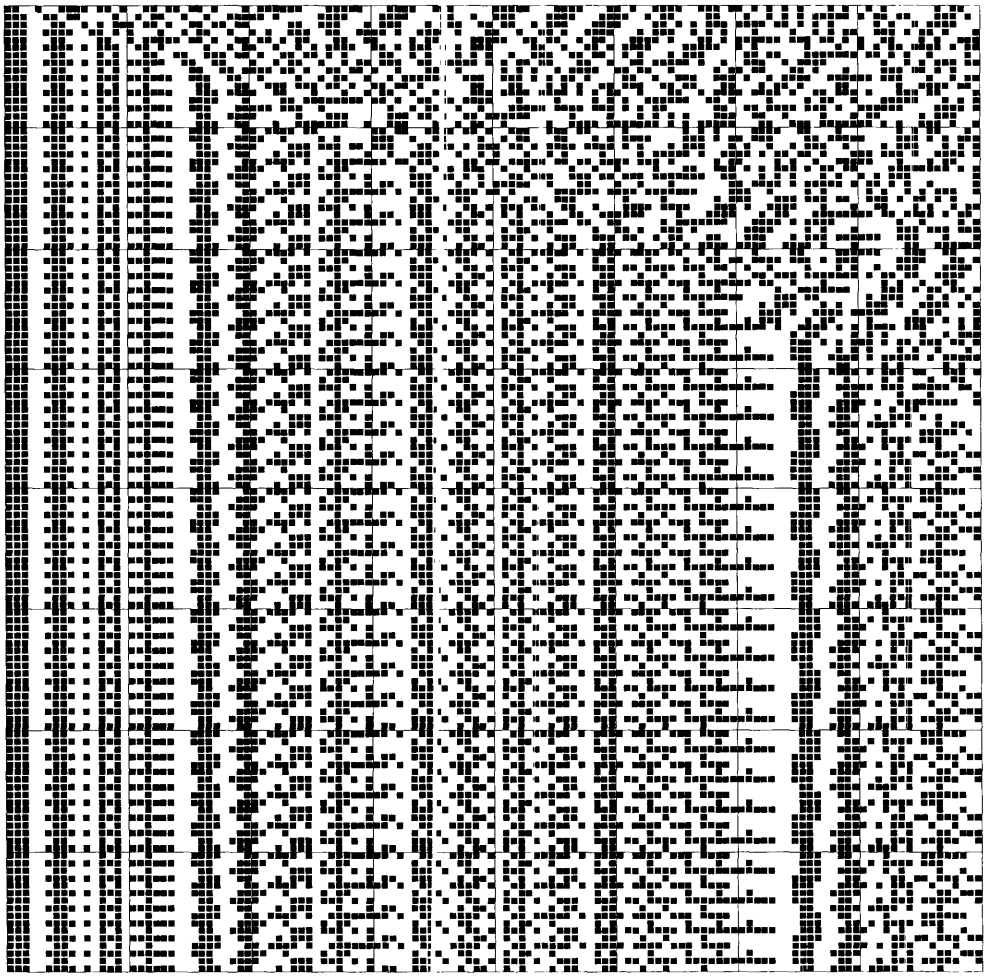


Figure 5.6: Rule f_{510} iterated from a random finite state. Convergence of limit sequences is predictable (Figure 5.5).

Chapter 6

Conclusions

6.1 Non-metric Calculus

The primary achievement of this thesis has been to provide a framework for the analysis of cellular automata with particular application to the forward problem. The methods outlined in this thesis are analogous with those used in the construction of the differential calculus and the algebraic classification of differential equations, with the important distinction that these methods may be developed without reference to a metric in the context of cellular automata.

In Chapter 2, we began by showing that the class of cellular automata to which this calculus applies – namely, binary 1-dimensional non-generative first order $(f, \mathcal{F}^{-})$ of arbitrary degree – are capable of precisely simulating the behaviour of all computable one dimensional cellular automata (Propositions 2.1 and 2.5). [1] As a consequence we may work entirely within the algebraic framework of the field F_2 without loss of behavioural generality. Stated simply, the cellular automata we study do everything that computable 1-dimensional[2] cellular automata can do.

Most of the results which follow hinge upon the fact that may we work with a field algebra, and significantly, there are results which only apply for the field F_2:

[1] The reduction to $d = m = 1$ is well known, e.g. [13], however the reduction of alphabet to $m = \log k = 1$ is believed to be an original contribution.

[2] The reduction of alphabet in higher dimensions is straightforward by similar arguments.

firstly, a field algebra is necessary for the polynomial representation of a local rule, as discussed in Chapter 3 and exploited in Chapter 5. Such representation allows for straightforward enumeration and algebraic classification of rules (mostly on the basis of looking for partial or conditional linear dependences) and extrapolation of behaviour to rules of higher degree. [3]

Secondly, a field algebra is necessary for the analysis of linear cellular automata from which we construct bases for the analysis of nonlinear cellular automata. In particular, we analyse and extend the binomial coefficients in F_2, which is equivalent to the iteration and inversion of the unique non-trivial elementary linear binary cellular automaton from a seed. This cellular automaton generates the binomial sequences which are natural bases for the discussion of nonlinear cellular automata: they are computable by 4.3, [4] and hence have behaviour understood by Propositions 4.4 to 4.6; they are by definition natural bases for finite difference calculus; they are closed under bitwise multiplication by Theorem 4.7; finally, the temporal binomial sequences are a basis for $\mathcal{P}^{2^{\mathbb{N}}}$, which significantly is known to dominate the limiting behaviour of temporal sequences of velocity 0 of all non-generative $(f, \mathcal{F}^-)$ (Proposition 5.8).

Finally, we have a result which only applies to binary cellular automata: the relationship between the the polynomial form and the rule table, Proposition 3.1. This is presented as a useful computational tool, and not investigated much further here. Nevertheless, it demonstrates that an understanding of the binomial coefficients and the binomial transform is central to the analysis of cellular automata, and gives further reason to restrict our attention to binary cellular automata.

6.2 Nonlinear Cellular Automata

We propose that the non-metric calculus developed in this thesis is a natural framework for the forward analysis of cellular automata. The extent to which this is true,

[3] This is an important achievement, as much of the related literature on cellular automata presents results which are relevant only to cellular automata of some fixed degree. For example, the use of non-additive basis operators [24] would be somewhat cumbersome for rules of degree $d \geq 3$.

[4] The extended version of Lucas' Theorem using 2's complement is believed to be an original contribution.

and indeed the success of this thesis, may only be gauged by the results of Chapter 5. Therefore, we will summarise the most important achievements in this chapter.

Of primary importance, we have achieved the exact (partial) solution of a non-trivial nonlinear cellular automaton: $f_{66}(xyz) = xy + z$ iterated from a double seed. The temporal binomial sequences provide the basis in which to express this solution:

$$x_t^i = \binom{t}{y_i},$$

where

$$y_{3\cdot 2^n} = 2 \cdot 4^n.$$

By simply observing the polynomial form of this rule, we found a class of cellular automata of *arbitrary degree* – the so-called *counters* – which may be solved in a similar fashion. Hence we determined the rates of growth of these cellular automata,

$$\Theta(\ell_d^t) = \Theta(\sqrt[d]{t}) \supset O(\log(t)),$$

indicating (Conjecture 5.15) that these cellular automata are significant as the *slowest expanding* among all cellular automata of the same degree.

We would expect that such explicit solutions of cellular automata are not always so easily achieved. For these cases, we have proceeded to outline various methods for the analysis of the limiting behaviour of cellular automata in general, and have achieved strong statements relevant to the stability and periodicity of the temporal sequences generated by very broad classes of cellular automata, as follows.

Proposition 5.8 states that the temporal sequences of velocity 0 of any non-generative $(f, \mathcal{F}^-)$ are ultimately periodic, each with right period 2^m for some $m \in \mathbb{N}$. Since any computable cellular automaton may be simulated by such an $(f, \mathcal{F}^-)$, this implies the ultimate periodicity of computable cellular automata in general in this frame of reference. Further, this proposition establishes the importance of the temporal binomial sequences (as a basis for $\mathcal{F}^{2^{\mathbb{N}}}$) to the analysis of binary cellular automata.

Proposition 5.4 states that the temporal sequences of velocity 0 of any r-quasi-linear $(f, \mathcal{F}^-)$ are strictly periodic, again with period 2^m for some $m \in \mathbb{N}$.

Proposition 5.10 states that any $(f, \mathcal{F}^-)$ with ultimately non-trivial behaviour in the velocity 0 frame of reference must be weakly injective in the rightmost variable.

Proposition 5.13 goes further to state that such an $(f, \mathcal{F}^-)$ which is also l-quasi-linear will generate limit sequences with unbounded periods; that is, $P(x_i') \to \infty$ as $i \to \infty$.

Such results are useful in narrowing the scope of the general forward problem to those cellular automata with ultimately 'interesting' behaviour. For example, we consider the 'famous' rule f_{30} which is both l-quasi-linear and weakly injective in the rightmost variable. This rule has an interesting stability property of convergence to the same set of limiting sequences from any nonzero initial state which we show is equivalent to the property of unconditional period doubling of unstable limit sequences (Proposition 5.11). As for the counters, we extrapolate from the polynomial form of this rule to find a class of cellular automata with the same behaviour (Conjectures 5.14 and 5.15). [5]

Therefore, the primary achievement of this thesis overall has been to present a combinatorial calculus which is natural for the analysis of nonlinear cellular automata. It is not intended that the results in Chapter 5 are in any way 'the last word' on the subject. Rather, it is hoped that the framework outlined should act as a starting point for further analysis.

6.3 Cellular Automata and Symbolic Dynamics

Here we discuss the relevance of this thesis to some of the existing literature on symbolic dynamics; in particular, the commuting block maps problem [3].

6.3.1 Shift Dynamical Systems

Over some *alphabet* (i.e. finite set of symbols) $\mathcal{A}$, we make the following definitions: The *full $\mathcal{A}$-shift*, denoted $X(\mathcal{A})$, [6] is the set of all bisequences

$$X(\mathcal{A}) = \mathcal{A}^{\mathbb{Z}} = \{x = (x_i) : x_i \in \mathcal{A}, \text{ for all } i \in \mathbb{Z}\}.$$

[5] Such extrapolation of behaviour to rules of higher degrees as achieved would appear intractable with only specific reference to a rule table.

[6] We assign the discrete topology to $\mathcal{A}$, and the product topology to $X(\mathcal{A})$; hence $X(\mathcal{A})$ is homeomorphic to the Cantor discontinuum [7].

A *block* (or more precisely *n-block*) is a finite sequence $u = u_1 \cdots u_n \in \mathcal{A}^n$. We say a sequence x *contains* an N-block u if there exists some $m \in \mathbb{Z}$ such that $x_{m+1} \cdots x_{m+n} = u_1 \cdots u_n$. Any collection F of *forbidden blocks* defines a *shift space*, denoted $X_F \subseteq X(\mathcal{A})$, by

$$X_F = \{x \in X(\mathcal{A}) : x \text{ does not contain any } u \in F\}.$$

Define the *shift* homeomorphism $\sigma : X(\mathcal{A}) \to X(\mathcal{A})$ by $[\sigma(x)]_i = x_{i+1}$. A *shift dynamical system* (X_F, σ_{X_F}) consists of a shift space X_F together with the restriction σ_{X_F} of σ to X_F.

The study of shift dynamical systems arose from the discovery that certain dynamical systems, notably geodesic flows on compact manifolds of negative curvature [14], have orbits which can be characterised by symbolic bisequences constituting a shift space. More recently, the methods of symbolic dynamics have been applied to data storage and transmission [11]. The relevance of symbolic dynamics to cellular automata began with the study of endomorphisms of the shift dynamical system $(X(\mathcal{A}), \sigma)$ and their relationship with block maps, as follows.

An *endomorphism* of $(X(\mathcal{A}), \sigma)$ is a continuous shift-commuting map from $X(\mathcal{A})$ to itself. An *n-block map* is a map $f : \mathcal{A}^n \to \mathcal{A}$. [7] Every n-block map f induces an endomorphism of $(X(\mathcal{A}), \sigma)$, denoted f_∞, by

$$[f_\infty(x)]_i = f(x_{i+1} \cdots x_{i+n}).$$

Clearly, $(f, X(\mathcal{A}))$ is a one-dimensional cellular automaton. Not so obvious is a theorem of Curtis, Hedlund and Lydon [7] which asserts that *every* endomorphism of $(X(\mathcal{A}), \sigma)$ is of the form $\sigma^m \circ f_\infty$ for some block map f and $m \in \mathbb{Z}$.

Much is known about the endomorphisms of $(X(\mathcal{A}), \sigma)$, [8] all of which, according to this theorem, is relevant to the cellular automata forward problem. Certainly, more needs to be done to integrate this existing literature with 'mainstream' cellular automata theory. However, it should be pointed out that the motivation for studying the endomorphisms of $(X(\mathcal{A}), \tau)$ in the context of symbolic dynamics is not exactly

[7] Essentially a one-dimensional first order cellular automaton *local rule*.

[8] See survey article [7].

the same as the motivation of the cellular automata forward problem, which to some extent limits their mutual relevance.

Firstly, the subsets of $X(\mathcal{A})$ which are typically of interest to cellular automata theorists are not usually shift spaces. For example, the space of finite sequences, of interest in this thesis for reasons of computability, cannot be defined by a collection of forbidden *finite* blocks.

Secondly, the cellular automata forward problem in general terms is to determine properties of the *orbit* $\{x^n\}_{n\in\mathbb{N}} = \{(f_\infty)^n(x^0)\}_{n\in\mathbb{N}}$ for some cellular automaton $(f, \mathcal{S})$ and initial state $x^0 \in \mathcal{S}$, which is of no particular interest in symbolic dynamics.

6.3.2 The Commuting Block Maps Problem

One problem emerging from symbolic dynamics which would seem to have particular relevance to the subject of this thesis is the commuting block maps problem [3, 16, 17]. For the same reason that we have restricted our attention to binary cellular automata, this problem has been studied exclusively for block maps over $\mathbb{Z}_2$, thereby allowing the representation of n-block maps as polynomials in n variables over $\mathbb{Z}_2$. In the discussion which follows we assume the alphabet $\mathbb{Z}_2$.

Firstly, equality and composition of block maps are defined in such a way as to agree with equality and composition of the induced endomorphisms as follows. Let $\mathcal{F}_n$ denote the set of all n-block maps, and $\mathcal{F} = \bigcup_{n=1}^{\infty} \mathcal{F}_n$ the set of all block maps. If $f \in \mathcal{F}_m$ and $g \in \mathcal{F}_n$, then we say $f = g$ provided $f(b_1 \cdots b_m) = g(b_1 \cdots b_n)$ for every N-block $b_1 \cdots b_N$ where $N = max\{m, n\}$. $g \circ f \in \mathcal{F}_{m+n-1}$ is defined by $(g \circ f)(x_1 \cdots x_{m+n-1}) = g(y_1 \cdots y_n)$ where $y_i = f(x_i \cdots x_{i+m-1})$ for $1 \leq i \leq n$.

The *commuting block maps problem* is to determine for a given $f \in \mathcal{F}$ the set

$$\mathcal{C}(f) = \{g \in \mathcal{F} : g \circ f = f \circ g\},$$

which has been solved for linear block maps and for block maps of the (polynomial) form

$$f(x_0 \cdots x_k) = x_0 + \delta_0 + \prod_{i=1}^{k}(x_i + \delta_i),$$

where $k \geq 2$ and $\delta_0 \cdots \delta_k$ is a constant $k+1$-block.

The form of these block maps has a striking resemblance to the quasi-linear cellular automata rules for which results were achieved in Chapter 5 of this thesis: put $\delta_i = 0$ for all i and you have the counters; put $\delta_i = 1$ for all i and you have the other class of cellular automata which drew our attention with interesting stability properties. It is unclear whether this resemblance is merely coincidental – perhaps this form of block map most easily lends itself to analysis – or whether it hints at some deeper connection. In any case, these observations suggest a worthwhile avenue for continued research.

6.4 Further Research

Certain areas in this thesis require more work, and others suggest considerable scope for further research:

In Chapter 2, there would appear to be some relationship between d, m and $\log k$ worth investigating as a possible measure of the complexity of a local rule, with the possible inclusion of the dimension D as a fourth quantity. Also, the definition of simulation proposed could be refined somewhat, though this particular definition is sufficient for the purposes of this thesis.

In Chapter 3, we have observed that, under enumeration, rule tables may be regarded as sequences in $\mathcal{P}^{2^{\mathbb{N}}}$, and polynomial forms as sequences in $\mathcal{F}$, where the relationship between them is the binomial transform. There would seem to be some scope for abstract analysis of cellular automata in this framework, allowing for the possible discussion of 'infinite' polynomial forms, which would correspond with aperiodic rule tables.

Chapter 4 is reasonably self-contained, however there are many properties of the binomial coefficients which may have some relevance to the calculus of cellular automata which are not considered.

The results in Chapter 5 present the widest scope for continued research, which is of course the purpose of developing a calculus of cellular automata: not just to answer questions, but to provide a general framework in which questions of interest may be formulated. Specifically, more needs to be known about solving equations similar

to (5.2) for the purpose of obtaining solutions for quasi-linear cellular automata. Proofs of stated conjectures should be possible within the framework outlined. More comprehensive algebraic classification of rules may also be achieved, but this is a long term goal, as it amounts to a general forward problem.

Some other areas worth consideration beyond the scope of this thesis are as follows. The question remains as to how well the type of analysis used here apply to other frames of reference. We have only analysed binomial sequences of velocity 0 for the rule f_6, and implemented these in the analysis of the velocity 0 frame of reference of nonlinear cellular automata. Fractional velocity binomial sequences may have some relevance to the analysis of 'harder' (and more interesting) regions, for example the aperiodicity of velocity 1 sequences of $d = 2$ rules quasi-linear[9] in the leftmost or rightmost variables as studied by [9]. Similarly, one may consider linear dependence of a local rule upon 'inner' variables of its domain, although such would appear more difficult and perhaps less relevant to the overall behaviour.

The techniques outlined here should in principle apply to binary cellular automata of arbitrary order in time. Behaviourally speaking, this class of cellular automata would offer nothing new, however they provide a useful environment for the construction of rules which generate solitons, and therefore would seem to have some significance to the inverse problem, with specific application to the modelling of physical systems.

Perhaps if we view homogeneous differential or partial differential equations on Cartesian spaces at one extreme, and cellular automata on infinite lattices at another, with various levels of discretisation (of space, time, or alphabet) in between, it would appear that the 'procedural' methodology of calculus applies best at the extremes – the former in a metric framework, the latter in a combinatorial framework. There are strong analogies to be drawn between the two, and at the same time an incompatibility, which may in part explain why various methods of analysis seem to become harder at intermediate levels of discretisation, with some exceptions. For example, there exist few techniques for the solution of nonlinear recurrence relations on real numbers. However, sophisticated techniques such as inverse scattering transform work after at

[9] 'Injective' in the terminology of [9].

least one level of discretisation in the case of the Toda lattice. The possibility of the inverse scattering transform applied to cellular automata is an interesting one, in view of the fact that the techniques in Chapter 5 amount to a discrete spectral analysis.

Finally, it should be said that much work needs to be done integrating the various methods and notations already existing in the analysis of cellular automata. This is already becoming a difficult task, but would probably assist the evolution of the 'right' formalisms, assuming such things exists.

Appendix A

Extended Binomial Inversion

We observe that the binomial inversion formula (4.25) does not apply for the extended binomial coefficients; that is, al $t, i \in \mathbb{Z}$. The following results outline what appears to be the correct way to view t ıe binomial transform and inversion for the extended binomial coefficients. This ex ension agrees with (4.25) for $t, i \in \mathbb{N}$, and is not required for the discussions in his thesis. However, I am not aware that this result is known, and is presented here for sake of generality.

First, we need the following

Lemma A.1 For all $t \in \mathbb{N}$ and $i \in \mathbb{Z}$,

$$\binom{t}{i} = \binom{t}{t-i},$$

and if $i < 0$ or $i > t \geq 0$, then

$$\binom{t}{i} = 0.$$

These results are well known, and will not be shown here. Also, we need:

Lemma A.2 For all $t < 0$ and $i \in \mathbb{N}$,

$$\binom{t}{i} = \binom{i-(t+1)}{-(t+1)}.$$

This deserves a proof, which we will do by induction on $\min(i, -(t+1))$. For $t = -1$, we have

$$\binom{-1}{i} = 1 = \binom{i}{0}$$

for all $i \geq 0$. Similarly for $i = 0$, we have

$$\binom{t}{0} = 1 = \binom{-(t+1)}{-(t+1)}$$

for all $t < 0$. Hence, for $\min(i, -(t+1)) > 0$ we have

$$\begin{aligned}
\binom{t}{i} &= \binom{t}{i-1} - \binom{t+1}{i}, \quad \text{by (4.9)} \\
&= \binom{i-1-(t+1)}{-(t+1)} + \binom{i-(t+2)}{-(t+2)}, \quad \text{by induction} \\
&= \binom{i-(t+2)}{-(t+1)} + \binom{i-(t+2)}{-(t+2)} \\
&= \binom{i-(t+1)}{-(t+1)}, \quad \text{by (4.9)}
\end{aligned}$$

as required. We may now prove:

Theorem A.3 (Extended Binomial Inversion) For all $t, i \in \mathbb{Z}$

$$\sum_{m=-\infty}^{\infty} \binom{t}{t-m}\binom{m}{m-i} = \delta_i^t, \tag{A.1}$$

To begin with, we should stress that this sum is well defined; it is expressed with infinite limits so that we may ignore limits of summation for analysis. The first term is 0 if $m > t$, and the second is 0 if $i > m$, so we may write interchangeably

$$\sum_{m=-\infty}^{\infty} \binom{t}{t-m}\binom{m}{m-i} = \sum_{m=i}^{t} \binom{t}{t-m}\binom{m}{m-i}$$

for all $t, i \in \mathbb{Z}$. Clearly, the sum will be 0 for $i > t$, so now we analyse the summand for three other cases. Firstly, if $0 \leq i \leq t$ then by Lemma A.1,

$$\begin{aligned}
\sum_{m=i}^{t} \binom{t}{t-m}\binom{m}{m-i} &= \sum_{m=i}^{t} \binom{t}{m}\binom{m}{i} \\
&= \delta_i^t,
\end{aligned}$$

by (4.25). Secondly, if $i < 0 \leq t$ then

$$\sum_{m=i}^{t} \binom{t}{t-m} \binom{m}{m-i} = \sum_{m=0}^{t} \binom{t}{t-m} \binom{m}{m-i},$$

since $\binom{t}{t-m} = 0$ for $m < 0$ by Lemma A.1. Hence the sum is 0 as required since $\binom{m}{m-i} = 0$ for $i < 0$, again by Lemma A.1. Finally, if $i \leq t < 0$ then

$$\begin{aligned} \sum_{m=i}^{t} \binom{t}{t-m} \binom{m}{m-i} &= \sum_{m=i}^{t} \binom{t-m-(t+1)}{-(t+1)} \binom{m-i-(m+1)}{-(m+1)} \\ &= \sum_{m=i}^{t} \binom{-(m+1)}{-(t+1)} \binom{-(i+1)}{-(m+1)}, \end{aligned}$$

the first equality by Lemma A.2. Now we put $m' = -(m+1)$ to obtain

$$\begin{aligned} \sum_{m=i}^{t} \binom{t}{t-m} \binom{m}{m-i} &= \sum_{m'=-(t+1)}^{-(i+1)} \binom{m'}{-(t+1)} \binom{-(i+1)}{m'} \\ &= \delta^{-(i+1)}_{-(t+1)}, \quad \text{by 4.25} \\ &= \delta^{t}_{i}, \end{aligned}$$

and the proof is complete.

Bibliography

[1] M. Aigner, *Combinatorial Theory*, Springer-Verlag, New York, 1979.

[2] E.R. Berlekamp, *Algebraic Coding Theory*, Aegean Park Press, 1984.

[3] E.M. Coven, G.A. Hedlund and F. Rhodes, *The Commuting Block Maps Problem*, Transactions of the American Mathematical Society **249** (1979) 113-138.

[4] N.J. Fine, *Binomial Coefficients Modulo a Prime*, American Mathematical Monthly **54** (1947) 589-592.

[5] P. Guan and Y. He, *Exact Results for Deterministic Cellular Automata with Additive Rules*, Journal of Statistical Physics **43** (1986) 463-478.

[6] H.A. Gutowitz, *Introduction*, Physica **45D** (1990) vii-xiv.

[7] G.A. Hedlund, *Endomorphisms and Automorphisms of the Shift Dynamical System*, Math. Systems Theory **3** (1969) 320-375.

[8] E. Jen, *Cylindrical Cellular Automata*, Commun. Math. Phys. **118** (1988) 569-590.

[9] E. Jen, *Aperiodicity in One-Dimensional Cellular Automata*, Physica **45D** (1990) 3-18.

[10] W.G. Kelley and A.C. Peterson, *Difference Equations: An Introduction with Applications*, Academic Press, London, 1991.

[11] D. Lind and B. Marcus, *An Introduction to Symbolic Dynamics and Coding*, Cambridge University Press, 1995.

[12] O. Martin, A.M. Odlyzko S. Wolfram, *Algebraic Properties of Cellular Automata*, Commun. Math. Phys. **93** (1984) 219-258.

[13] C. Moore, *Quasi-Linear Cellular Automata*, Santa Fe Institute preprint 1995.

[14] M. Morse and G.A. Hedlund, *Symbolic Dynamics*, Amer. J. Math. **60** (1938) 815-866.

[15] H. Peitgen, H. Jürgens and D. Saupe, *Chaos And Fractals: New Frontiers of Science*, Springer-Verlag, New York, 1992.

[16] F. Rhodes, *The Sums of Powers Theorem for Commuting Block Maps*, Transactions of the American Mathematical Society **271** (1982) 225-236.

[17] F. Rhodes, *The Principal Part of a Block Map*, Journal of Combinatorial Theory **A33** (1982) 48-64.

[18] F. Robert, *Discrete Iterations: A Metric Study*, Springer-Verlag, Berlin, 1986.

[19] S. Takahashi, *Linear Cellular Automata and Multifractals*, Physica **45D** (1990) 36-48.

[20] M. Toda, *Theory of Nonlinear Lattices* (2nd ed.), Springer-Verlag. Berlin, 1989.

[21] T. Toffoli and N.H. Margolus, *Invertible Cellular Automata: A Review*, Physica **45D** (1990) 229-253.

[22] G.Y. Vichniac, *Simulating Physics with Cellular Automata*, Physica **10D** (1984) 96-116.

[23] G.Y. Vichniac, *Boolean Derivatives on Cellular Automata*, Physica **45D** (1990) 63-74.

[24] B. Voorhees, *Nearest Neighbour Cellular Automata over $\mathbb{Z}_2$ with Periodic Boundary Conditions*, Physica **45D** (1990) 26-35.

[25] S. Wolfram, *Universality and Complexity in Cellular Automata*, Physica **10D** (1984) 1-35.

[26] S. Wolfram, *Random Sequence Generation by Cellular Automata*, Advances in Applied Mathematics **7**, (1986) 123-169.