Characteristic Polynomials of One-Dimensional Linear Hybrid Cellular Automata

by

Kevin Michael Cattell
B.Sc., University of Victoria, 1989

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

We accept this dissertation as conforming
to the required standard

Dr. J.C. Muzio, Supervisor (Department of Computer Science)

Dr. M. Serra, Departmental Member (Department of Computer Science)

Dr. D.M. Miller, Departmental Member (Department of Computer Science)

Dr. V.K. Bhargava, Outside Member (Department of Electrical Engineering)

Dr. W.E. Pfaffenberger, Outside Member (Department of Mathematics and Statistics)

Dr. P.H. Bardell, Jr., External Examiner (Carmel, N.Y., U.S.A.)

Supervisor: Dr. J.C. Muzio

ABSTRACT

A one-dimensional linear hybrid cellular automaton (CA) is a specialised form of linear finite state machine. These machines are of interest, both for their theoretical properties and for their applications in VLSI built-in-self-test, random number generation, cryptography, coding theory, and other areas. This work is a study of the algebraic properties of the characteristic polynomials of CA, primarily for machines defined over GF(2). Several problems, previously open, are solved: the efficient synthesis of a CA from an irreducible polynomial, the existence and uniqueness of CA for irreducible polynomials, the reducibility of the characteristic polynomial of a cyclic-boundary CA, and the form of a similarity transform between CA and linear feedback shift registers. A probabilistic algorithm for the synthesis of CA over finite fields other than GF(2) is presented. Various other results concerning the characteristic polynomial of CA are derived, and possible directions for future research are discussed.

Examiners:

Dr. J.C. Muzio, Supervisor (Department of Computer Science)

Dr. M. Serra, Departmental Member (Department of Computer Science)

Dr. D.M. Miller, Departmental Member (Department of Computer Science)

Dr. V.K. Bhargava, Outside Member (Department of Electrical Engineering)

Dr. W.E. Pfaffenberger, Outside Member (Department of Mathematics and Statistics)

Dr. P.H. Bardell, Jr., External Examiner (Carmel, N.Y., U.S.A.)

Contents

List of Tables

List of Figures

Acknowledgement

I would like to thank Dr. Jon Muzio for his invaluable guidance, support, patience and friendship throughout my time as a graduate student.

I would also like to thank my committee and my external examiner, Dr. Paul Bardell, for their comments and suggestions.

My fellow students in the VLSI group have endured many talks on the theoretical aspects of CA. My thanks especially to Shujian Zhang for his helpful discussions and dedicated proofreading.

Finally, I would like to acknowledge NSERC, the Advanced Systems Institute, and the University of Victoria, for providing me with scholarships during the course of my graduate studies.

Dedication

To my loving wife Susan.

Chapter 1

Introduction

The purpose of this dissertation is to increase our understanding of a specialised type of linear finite state machine, known as a one-dimensional linear hybrid cellular automaton (CA)[1]. The approach of this work is to study the properties of the characteristic polynomials of these machines. Several open problems are addressed and solved.

A linear finite state machine (LFSM) is an abstract device, consisting of a clock, a memory, and a next-state function. The memory holds the state of the machine, and the next-state function calculates a new state when the clock ticks. This new state is then stored in the memory.

LFSMs are appealing to VLSI (very large scale integration) designers for two reasons. First, they are easily implemented in hardware, using flip-flops as memory elements and XOR gates for arithmetic. Second, their linearity makes them amenable to theoretical analysis using tools of linear algebra. This makes properties such as their cycle structure and their aliasing probability computable without resorting to empirical evidence gathering.

The most common type of LFSM used by VLSI designers is the linear feedback shift register (LFSR). LFSRs perform polynomial division, where the LFSR structure determines the divisor, and the dividend is shifted into the LFSR's input. Many

[1]Both the singular and the plural are abbreviated as *CA*.

aspects of LFSRs are well understood (see [56] and [62] for material on the theory and applications of these machines).

Cellular automata in general (not the highly restricted variety studied here) have a long history, starting with the work of von Neumann in the early 1950's, work which was motivated by the study of self-replicating systems. In the 1980's, Wolfram published several landmark papers on CA ([69, 70, 71]).

More recently, the VLSI community has become interested in CA, primarily for test pattern generation and signature analysis. CA have been proposed as an alternative to LFSRs for both of these applications. There are arguments and evidence for CA performing both better and worse than LFSRs, in a number of different situations. Also, there is a lack of consensus about whether CA are more or less difficult to implement in hardware than are LFSRs. This dissertation does not address these questions, though some discussion is contained in section 2.4.

The CA in this dissertation are restricted as follows: they are *finite*, meaning that the number of cells is finite; they are *one-dimensional*, meaning that the cells are laid out in the form of a linear graph, rather than on a higher dimensional lattice; they are *nearest-neighbour*, so that each cell is connected only to its immediately adjacent cells; they are *linear*, which restricts the next-state function to a linear operator; they are either *null-boundary* or *cyclic-boundary*, which describes how the inputs to the end cells are handled; and they are *hybrid*, meaning that different cells can use different next-state functions. All of these properties, except for the last, are restrictions on general CA. The use of hybrid CA is a generalisation, in that most of the work on CA has been for *uniform CA*, where all cells must be identical. Most of the VLSI-related work has been for CA that have most, if not all, of the above restrictions.

This work is a study of properties of the *characteristic polynomial* of a CA. Properties of this polynomial determine properties of the CA, such as cycle structure. The research set out with the following goals:

1. to study the relationship between a CA and its characteristic polynomial,

2. to study the properties of the characteristic polynomial of a CA,

3. to study the relationship between a CA and its corresponding LFSR,

4. to study cyclic-boundary CA, and

5. to generalise results whenever possible to larger finite fields.

In reference to goal (1), conjectures that had been made in the literature lead to the following more specific goals:

- to show that there is a CA for each irreducible polynomial,
- to show that there are at most two CA for each irreducible polynomial, and
- to find an efficient method to synthesise a CA from an irreducible polynomial,

Each of these goals was attained.

Pursuing goal (2) resulted in the determination of relationships that turned out to be the underpinnings of most of the other work.

For goal (3), the existence of a similarity transform between CA and LFSRs was shown by [61]. However, the proof is non-constructive, and no general form for such a transform was known. This dissertation demonstrates such a transform for CA with irreducible characteristic polynomials.

Goal (4) involves the study of a slightly different LFSM known as a cyclic-boundary CA. During the course of the research, two authors ([1] and [51]) conjectured that no cyclic-boundary CA has an irreducible characteristic polynomial. This conjecture was refined by analysing a large number of these machines, and the conjecture was proved.

For goal (5), a number of the results derived for CA defined over GF(2) have been generalised to CA defined over larger finite fields. Unfortunately, the key results for proving the existence, uniqueness, and synthesis do not seem to hold for any field other than GF(2).

This dissertation is organised as follows. Chapter 2 provides the necessary algebraic background for the dissertation. LFSMs are formally defined, as are LFSRs and CA. Previous work relating to applications and theory of CA are discussed.

Chapter 3 contains basic results about the characteristic polynomials of CA. The previously known results about the recurrence used to calculate the characteristic polynomial of a CA are presented. Several other relationships satisfied by the characteristic polynomial are given. A similarity transform between CA and LFSRs is shown constructively. The chapter closes with an analysis of cyclic-boundary CA, and shows that these machines have reducible characteristic polynomials.

Chapter 4 is a GF(q) "mirror" of Chapter 3. Most of the basic results about the characteristic polynomials of CA are generalised to larger fields, the only exception being the reducibility of cyclic-boundary CA characteristic polynomials.

Chapter 5 contains various additional results about CA characteristic polynomials. Different forms of CA are classified according to their interconnection structure, and the characteristic polynomial for each class is analysed. Alternate formulations of the characteristic polynomial are discussed. The effect on the characteristic polynomial of a CA when simple modifications are made to the machine are derived. Specialised forms of CA, namely palindromic, self-concatenated and uniform, are defined and their characteristic polynomials are analysed.

Chapter 6 presents the solution to several of the open problems concerning CA. Specifically, a fast method for synthesising CA from irreducible polynomials is obtained. Further, CA are shown to exist and be unique for irreducible polynomials.

Chapter 7 explores the problem of synthesising CA from polynomials for general finite fields. No results analogous to the GF(2) results are found, but a probabilistic method that is tractable for limited fields and machine lengths is derived. This, in turn, leads to another probabilistic algorithm, that significantly extends the range of values for which the problem is solvable.

Chapter 8 summarises the dissertation, and discusses possible future work.

Chapter 2

Background

2.1 Finite fields

This section presents the necessary background material on finite fields. This material is based on [44], [45] and [62]. Also, [29] and [30] contain information on polynomials over finite fields.

A *field* is an algebraic structure $(F, +, \cdot)$, where the set F contains the *elements* of the field, and the binary operations $+$ and $\cdot$ on F, called *addition* and *multiplication*, satisfy certain properties:

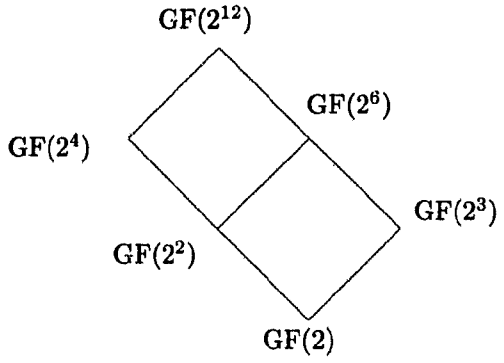
- $(F, +)$ is an abelian group (that is, $+$ is associative and commutative, F contains an identity (denoted 0), and every element a has an inverse (denoted $-a$)),
- $(F\backslash\{0\}, \cdot)$ is an abelian group,
- the distributive law holds: for $a, b, c \in F$, $a \cdot (b + c) = a \cdot b + a \cdot c$.

The multiplicative identity is denoted 1, and the multiplicative inverse of a is denoted a^{-1}. If F is finite, the field is called a *finite field.*

A finite field contains m^r elements, for some prime m and positive integer r. Further, for each m and r there exists exactly one finite field with m^r elements.

The finite field with q elements is denoted GF(q), where q is understood to be a power of a prime. If it is required that the prime and its power be known, then it is written explicitly as such in the text.

Figure 2.1: Lattice of subfields of GF(2^{12})



If the field has prime order m, then it is isomorphic to the integers modulo m ($\mathbb{Z}_m$). In general, the field with m^r elements is isomorphic to the set of all polynomials of degree less than r that have coefficients from $\mathbb{Z}_m$. Addition is as usual, with coefficients reduced modulo m. Multiplication is performed modulo some fixed degree r polynomial that is irreducible over the prime order field. The field has *characteristic* m, meaning that $m \cdot a = 0$ for all a.

The field GF(m^r), m prime, contains (an isomorphic copy of) the field GF(m^d) if and only if d divides r. For example, Figure 2.1 shows the subfields of GF(2^{12}).

Definition 2.1 *The set of all polynomials over GF(q) in indeterminate x is denoted GF(q)[x].*

Polynomial arithmetic is as usual, with coefficient arithmetic performed in the finite field.

Definition 2.2 *A non-zero polynomial p is said to be* irreducible *over GF(q) if for every factorisation $p = p_1 p_2$ with $p_1, p_2 \in$ GF(q)[x], either* $\deg(p_1) = 0$ *or* $\deg(p_2) = 0$.

For example, the polynomial $x^3 + 1 \in \mathrm{GF}(2)[x]$ is reducible, as it has the factorisation $(x+1)(x^2+x+1)$.

The following are examples of finite fields.

Example 2.3. The finite field GF(3) is isomorphic to $\mathbb{Z}_3$. The operation tables are

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

and

·	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

□

Example 2.4. The finite field GF(4) is represented using the irreducible degree 2 polynomial $y^2+y+1 \in \mathrm{GF}(2)[y]$. The field consists of $\{0, 1, y, y+1\}$. The operation tables are

+	0	1	y	$y+1$
0	0	1	y	$y+1$
1	1	0	$y+1$	y
y	y	$y+1$	0	1
$y+1$	$y+1$	y	1	0

and

·	0	1	y	$y+1$
0	0	0	0	0
1	0	1	y	$y+1$
y	0	y	$y+1$	1
$y+1$	0	$y+1$	1	y

□

There are q^n monic degree n polynomials in $\mathrm{GF}(q)[x]$. The number of these that are irreducible is given by the formula

$$\frac{1}{n}\sum_{d|n}\mu(d)q^{n/d},$$

where $\mu(i)$ is the Möbius function, defined as

$$\mu(i) = \begin{cases} 1, & \text{if } i = 1, \\ 0, & \text{if } i \text{ has a perfect square as a factor,} \\ (-1)^m, & \text{otherwise, where } m \text{ is the number of prime divisors of } i. \end{cases}$$

The summation is taken over all distinct divisors d of n.

Theorem 2.5 ([44]) *A degree n irreducible polynomial $p \in GF(q)[x]$ splits in the field $GF(q^n)$. That is, all n roots of p are in $GF(q^n)$. Furthermore, if α is a root of p, then the n roots of p are*

$$\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{n-1}}.$$

Definition 2.6 ([44]) *A degree n irreducible polynomial $p \in GF(q)[x]$ is* primitive *if it has a root α such that*

$$\{\alpha^i\}_{i=0}^{q^n-1} = GF(q^n) \setminus \{0\}.$$

(i.e. α generates the multiplicative subgroup of $GF(q^n)$).

If a degree n polynomial is primitive, then all n of its roots are generators of the multiplicative subgroup. The number of degree n primitive polynomials over GF(q) is simply the number of positive integers less than q that are relatively prime to q. This is Euler's ϕ function, for which one formula is

$$\phi(n) = n \prod_{d|n} (1 - \frac{1}{d}).$$

Tables of irreducible and primitive polynomials can be found in [3], [4] and [58].

A useful property that holds in a finite field is that $(a+b)^{m^i} = a^{m^i} + b^{m^i}$ for all a and b, where m is the characteristic of the field and i is an integer. This also holds for a and b polynomials. As a consequence, a polynomial in GF(2)[x] is a perfect square if and only if all of its non-zero terms have even degree.

The synthesis material in Chapter 6 requires the following definition.

Definition 2.7 *Let $p(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ be a polynomial in $GF(q)[x]$. The* formal derivative *$p'(x)$ of $p(x)$ is defined as*

$$p'(x) = n a_n x^{n-1} + (n-1) a_{n-1} x^{n-2} + \cdots + a_1,$$

where the coefficients are reduced modulo the order of the underlying prime field.

Even though there is no interpretation analogous to "slope" for real-valued functions, the derivative plays an important role in many aspects of finite field theory. For example, p has a multiple root if and only if p and p' are relatively not prime. Note that the derivative of a polynomial over GF(2) has only even degree terms, and so it is a perfect square.

The *trace* function is used in several places in this work. The trace of a matrix A, denoted $\mathrm{Tr}(A)$, is the sum of the elements on the main diagonal of A. The trace of an irreducible polynomial p, $\mathrm{Tr}(p)$, is the sum (in the splitting field of p) of the roots of p.

2.2 Linear finite states machines and cellular automata

This section introduces linear finite state machines. As each concept is defined, the corresponding properties for CA and linear feedback shift registers (LFSRs) are discussed. This section starts out with definitions applicable to general finite fields, but then concentrates on machines defined over GF(2). More detailed discussion of CA and LFSRs over larger fields is in Chapter 4.

A *linear finite state machine* (LFSM) is composed of n cells, labeled 1 through n. Each cell consists of a memory device capable of storing an element of GF(q), and a next-state function. In an LFSM, time evolves in discrete steps; that is, the LFSM is assumed to be operating in a synchronous mode under the control of an external clock. At each time step, each cell has a *state*, which is the value contained in its memory. The state of cell i at time t is an element of GF(q) and is denoted by s_i^t. For time step $t+1$, each cell i computes its new state s_i^{t+1}, using its next-state function f_i. Each f_i must be a linear function of $s_1^t, s_2^t, \ldots, s_n^t$.

A linear function from n-tuples in GF(q)^n to GF(q) can involve only multiplication by constants, and addition. In GF(2), the constants are 0 and 1, and so the multiplication can be represented as a connection being missing or present, respectively. Thus the *interconnection diagram* of an LFSM shows an arrow from cell j to cell i if f_i depends on the state of cell j. Over GF(q), an arrow is present from cell i to cell j if cell j *can* depend on cell i.

There are two forms of LFSR. In a type-1, or *internal XOR*, LFSR each cell receives input from its left neighbour and the rightmost cell. In a type-2, or *external*

Figure 2.2: Interconnection structure of a type-1 LFSR

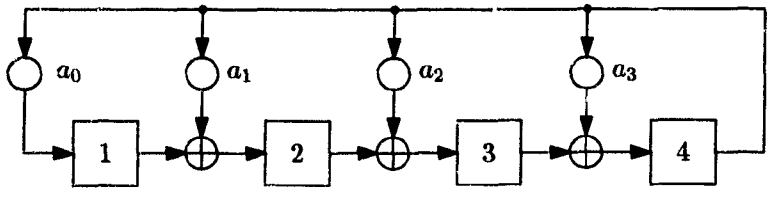
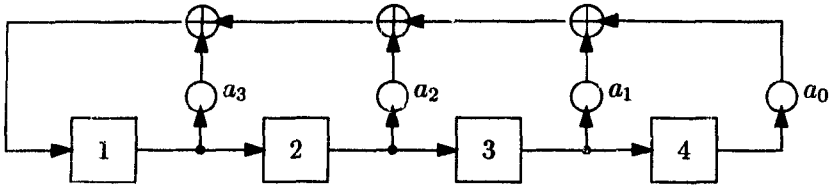


Figure 2.3: Interconnection structure of a type-2 LFSR



XOR, LFSR each cell receives input from only its left neighbour, except for the left-most cell, which receives a linear combination of the other cells. The interconnection structure of a 4-cell type-1 LFSR is shown in figure Figure 2.2, and that of a 4-cell type-2 is shown in Figure 2.3. All of the LFSRs in this dissertation are type-1.

In a CA, communication between cells is *nearest-neighbour*, meaning that each cell is connected to only its left and right neighbours. Figure 2.4 shows the interconnection structure of a CA. Thus the only information available to the next-state function is the states of cells $i-1$, i, and $i+1$ at time t. Hence for a CA

$$s_i^{t+1} = f_i(s_{i-1}^t, s_i^t, s_{i+1}^t).$$

The next-state function of a cell is called its *computation rule*, or just *rule*. Note that different cells may use different computation rules, making the CA *hybrid*. In

Figure 2.4: Null-boundary CA interconnection structure

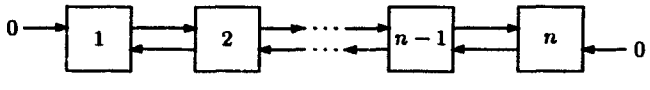


Table 2.1: Eight linear functions of three variables

$$f(a,b,c) = \begin{cases} 0 & \text{rule } 0 \\ a & \text{rule } 170 \\ b & \text{rule } 204 \\ c & \text{rule } 240 \\ a+b & \text{rule } 102 \\ a+c & \text{rule } 90 \\ b+c & \text{rule } 60 \\ a+b+c & \text{rule } 150 \end{cases}$$

contrast, all cells in a *uniform* CA use an identical computation rule (it is shown in Chapter 5 that uniform CA are not as interesting, from the point of view of this work, as hybrid CA). Also, note that the leftmost and rightmost cells behave as though their left and right neighbours, respectively, are always in state 0. This makes the CA *null-boundary*. The cells are labeled 1 through n, from left to right. A CA has a trivial symmetry, in that if the cell labels are reversed, the machine is structurally unchanged. Despite this, a CA and its reversal are considered to be different.

The remainder of this section is restricted to GF(2).

The next-state function f_i is a function of three variables, and there are $2^{2^3} = 256$ such boolean functions. However, the requirement that each next-state function be *linear* reduces this number to eight. These eight linear functions are shown in Table 2.1. The *rule number* is obtained from the decimal equivalent of the function's defining truth table.

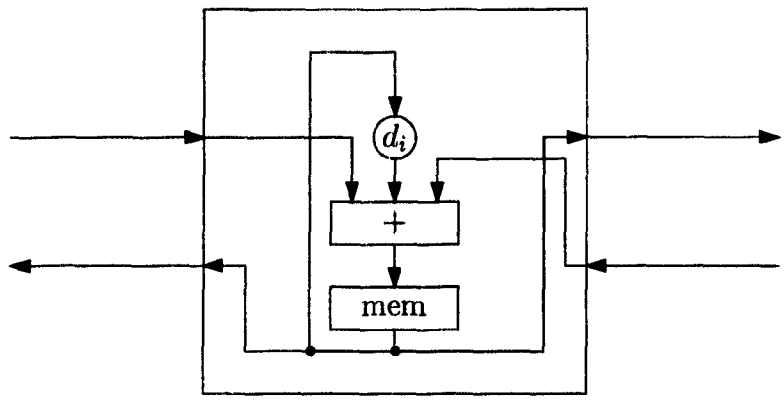
It turns out that of the eight possible linear functions of three variables, two are of primary interest. They are known as *rule 90* and *rule 150*, and are defined as:

$$\textit{rule 90:} \quad f_i(s_{i-1}^t, s_i^t, s_{i+1}^t) = s_{i-1}^t + s_{i+1}^t$$

$$\textit{rule 150:} \quad f_i(s_{i-1}^t, s_i^t, s_{i+1}^t) = s_{i-1}^t + s_i^t + s_{i+1}^t.$$

The other six linear functions of three variables give rise to CA with reducible char-

Figure 2.5: Internal structure of a CA cell



acteristic polynomials ([61], also shown in section 5.1). Because of this, almost all of the present work focuses on CA that use rules 90 and 150. Such CA are called *fully connected* in this work, because all possible connections between cells are present. It is specifically pointed out if other rules are under consideration. The internal structure of a cell of a fully-connected CA is shown in Figure 2.5. A CA Mis completely specified by which cells use rule 90 and which use rule 150. A natural form for the specification is an n-tuple d, called the *rule vector*:

$$d = [d_1, d_2, \ldots, d_n],$$

where

$$d_i = \begin{cases} 0, & \text{if cell } i \text{ uses rule 90} \\ 1, & \text{if cell } i \text{ uses rule 150.} \end{cases}$$

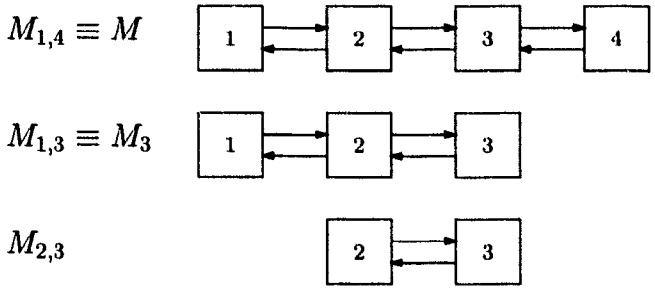
If it does not cause a loss of clarity, the term *CA* refers, interchangeably, to such a vector and to the actual machine. The *weight* of a CA is defined to be the number of 1s in its rule vector, i.e. the sum of the d_i.

Note that the encoding of rules 90 and 150 into 0 and 1, respectively, means that

$$f_i(s_{i-1}^t, s_i^t, s_{i+1}^t) = s_{i-1}^t + d_i s_i^t + s_{i+1}^t.$$

The notation $M_{i,j}, (i \leq j)$ refers to a *submachine* of the CA M. This submachine

Figure 2.6: Submachines of a 4-cell CA



consists of cells i through j, and is defined by its rule vector

$$[d_i, d_{i+1} \ldots, d_j].$$

Note that this submachine is also null-boundary; cells i and j receive 0 from their missing left and right neighbours, respectively. Since submachines that contain cell 1 are common, they are denoted M_j. Figure 2.6 shows a 4-cell CA and two of its submachines.

The *state* of an LFSM at time t is defined to be the n-tuple formed from the states of the individual cells:

$$s^t = [s_1^t, \ldots, s_n^t]^{\mathrm{T}}$$

(the superscript T represents the transpose of the vector). The *next-state function* of the LFSM is computed as

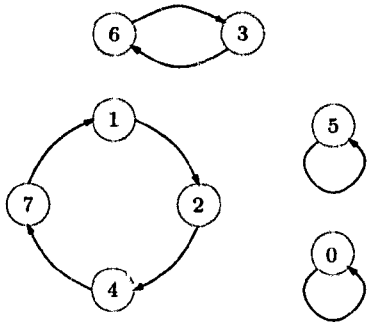
$$s^{t+1} = f(s_t) = [f_1(s^t), f_2(s^t), \ldots, f_n(s)]^{\mathrm{T}}. \tag{2.1}$$

For a CA, the next-state function is

$$s_{t+1} = f(s_t) = [f_1(0, s_1^t, s_2^t), \ldots, f_i(s_{i-1}^t, s_i^t, s_{i+1}^t), \ldots, f_n(s_{n-1}^t, s_n^t, 0)]^{\mathrm{T}}. \tag{2.2}$$

The *state diagram* of an LFSM is a graphical representation of the next-state function. Each of the 2^n states is a vertex, labeled with the decimal value of the state (most significant bit first). The arcs specify the next-state function. Figure 2.7 shows the

Figure 2.7: State diagram of a reducible LFSR



state diagram of a 3-cell (8-state) LFSR. Note that the next-state of the zero state is always the zero state.

Definition 2.8 *An n-cell LFSM has a* maximal-length cycle *if all non-zero states lie on a single cycle of length* $2^n - 1$.

Since each f_i is a linear function, f is also a linear function, mapping n-tuples to n-tuples. Linearity implies that f has an n-by-n matrix formulation A, so that the next-state function is computed as

$$s_{t+1} = f(s_t) = A \cdot s_t,$$

where the product is a matrix-vector multiplication over GF(2). This matrix is the LFSM's *transition matrix*.

The transition matrix of a type-1 LFSR has the form

$$A = \begin{bmatrix} 0 & 0 & 0 & \cdots & \cdots & 0 & a_0 \\ 1 & 0 & 0 & \ddots & & \vdots & a_1 \\ 0 & 1 & 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & & \ddots & \ddots & 0 & 0 & a_{n-3} \\ 0 & & & \ddots & 1 & 0 & a_{n-2} \\ 0 & 0 & \cdots & \cdots & 0 & 1 & a_{n-1} \end{bmatrix}.$$

A multiplier a_i being 1 denotes the presence of a feedback tap in front of cell i. This form of matrix, with (possibly) non-zero entries in the last column and 1s on the subdiagonal, is called a *companion form* matrix.

The nearest-neighbour communication to which a CA is restricted has the consequence that the transition matrix A is *tridiagonal.* The subdiagonal and the superdiagonal elements of the matrix are all 1, the main diagonal is the CA vector d, and the rest of the matrix is 0:

$$A = \begin{bmatrix} d_1 & 1 & 0 & \cdots & \cdots & 0 & 0 \\ 1 & d_2 & 1 & \ddots & & & 0 \\ 0 & 1 & d_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & d_{n-2} & 1 & 0 \\ 0 & & & \ddots & 1 & d_{n-1} & 1 \\ 0 & 0 & \cdots & \cdots & 0 & 1 & d_n \end{bmatrix} \tag{2.3}$$

It is straightforward to see that $A \cdot s^t$ is indeed the next state of the CA. The next state of cell i is the product of the ith row of A and s^t,

$$\begin{aligned} s_i^{t+1} &= (i\text{th row of } A) \cdot s^t \\ &= [0, \ldots, 0, 1, d_i, 1, 0, \ldots 0] \cdot [s_1^t, \ldots, s_{i-1}^t, s_i^t, s_{i+1}^t, \ldots, s_n^t]^{\mathrm{T}} \\ &= s_{i-1}^t + d_i s_i^t + s_{i+1}^t. \end{aligned}$$

The following lemma merely emphasises that the main diagonal of the matrix is the CA vector.

Lemma 2.9 *CA are in 1-1 correspondence with tridiagonal matrices that have subdiagonal and superdiagonal all 1.*

Example 2.10. Consider the 3-cell CA in Figure 2.8. The rule vector for this machine is $[0, 0, 1]$, and the transition matrix is

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

Figure 2.8: A 3-cell CA

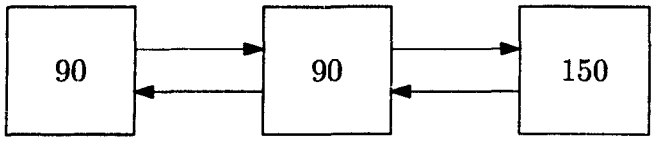
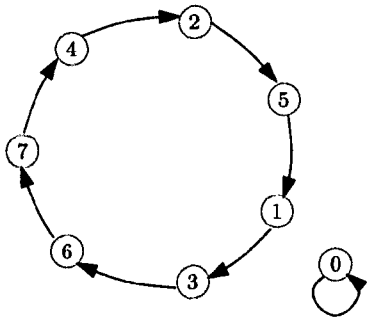


Figure 2.9: State diagram of a primitive CA



The sequence of states calculated by the CA, if started in $0, 0, 1$, is shown in the following table:

time	state
0	0, 0, 1
1	0, 1, 1
2	1, 1, 0
3	1, 1, 1
4	1, 0, 0
5	0, 1, 0
6	1, 0, 1
7	0, 0, 1

The state structure of the machine is in Figure 2.9. □

Example 2.11. Consider the 3-cell LFSR in Figure 2.10. The transition matrix is

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

The sequence of states calculated by the LFSR, if started in $1, 0, 0$, is shown in the

Figure 2.10: A 3-cell LFSR

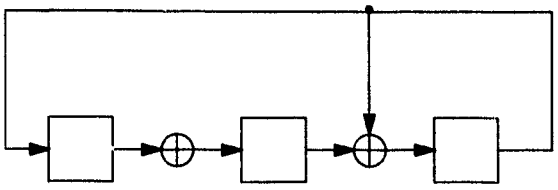
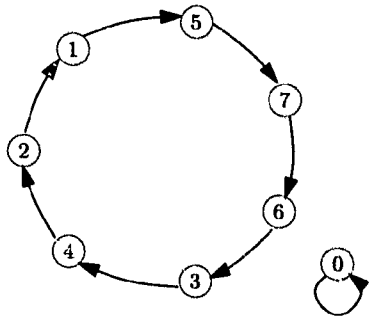


Figure 2.11: State diagram of a primitive LFSR



following table:

time	state
0	1, 0, 0
1	0, 1, 0
2	0, 0, 1
3	1, 0, 1
4	1, 1, 1
5	1, 1, 0
6	0, 1, 1
7	1, 0, 0

The state structure of the machine is in Figure 2.11. □

2.3 Characteristic polynomials

The *characteristic polynomial* p of an LFSM with transition matrix A is defined by

$$p = |xI - A|,$$

where x is an indeterminate and I is the identity matrix with dimension n. The characteristic polynomial is a degree n polynomial in x, though the indeterminate is often dropped if the context is clear. The matrix $xI - A$ is called the *characteristic matrix* of A.

The characteristic polynomial of an LFSM can determine its cycle structure, as the following theorems show.

Theorem 2.12 ([62]) *If the characteristic polynomial p of an LFSM is irreducible, then all non-zero states lie on cycles of length k, where k is the least integer such that p divides $x^k - 1$.*

Theorem 2.13 ([62]) *If the characteristic polynomial of an LFSM is primitive, then all non-zero states lie on a single maximal-length cycle.*

Definition 2.14 *A polynomial p is said to be a* CA polynomial *if it is the characteristic polynomial of some CA M. In this case, M is said to be a* realisation *of p.*

The characteristic polynomial of an LFSR can be read directly from its transition matrix. If the last column is $[a_0, a_1, \ldots, a_n]^{\mathrm{T}}$, then the characteristic polynomial is $x^n - a_{n-1}x^{n-1} - \cdots - a_1 x - a_0$.

Although the characteristic polynomial of a CA is more difficult to obtain than that of an LFSR, Chapter 3 shows that the general method of computing determinants need not be used. Appendix A lists CA and their characteristic polynomials for all CA with up to six cells.

The characteristic polynomial of a CA is denoted Δ. Recall that $M_{i,j}$ has been defined as the CA consisting of cells i through j.

Definition 2.15 *Let M be a CA. Define $\Delta_{i,j}$ to be the characteristic polynomial of the submachine $M_{i,j}$.*

Submachines consisting of cells 1 through j are commonly referred to, and so Δ_j is used to denote the characteristic polynomial of such a submachine. Note that a CA can be denoted by $M_{1,n}$, M_n or M, and that its characteristic polynomial can be denoted by $\Delta_{1,n}$, Δ_n, or Δ. Later material requires that the characteristic polynomial of an n-cell CA for $n = 0$ and $n = -1$ be defined:

$$\begin{aligned} \Delta_{-1} &= \Delta_{i,i-2} = 0 \\ \Delta_0 &= \Delta_{i,i-1} = 1. \end{aligned}$$

For brevity, a CA with an irreducible characteristic polynomial is called an *irreducible CA*. If the characteristic polynomial is primitive, the CA is called a *primitive CA* or a *maximal-length CA*. Analogous terms are used for LFSRs.

The following lemma is quite useful.

Lemma 2.16 *The degree $n-1$ coefficient of the characteristic polynomial equals the trace of the transition matrix.*

In order to better understand the context of this work, it is important to define precisely the relationship between linear operators and linear finite state machines. A *linear operator* L mapping $\mathrm{GF}(q)^n$ to $\mathrm{GF}(q)^n$ satisfies

$$\begin{aligned} &L(a+b) = L(a) + L(b) \ \ \forall a, b \in \mathrm{GF}(q)^n, \quad \text{and} \\ &L(ca) = cL(a), \ \ \forall a \in \mathrm{GF}(q)^n, c \in \mathrm{GF}(q). \end{aligned}$$

An LFSM is a *realisation* of a linear operator. A linear operator can have many LFSM realisations. Two LFSMs that realise the same linear operator are called *similar*.

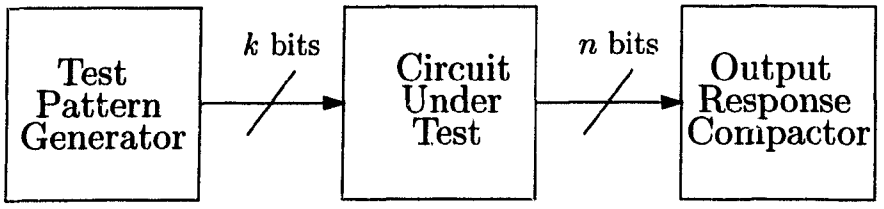
The following theorem ([44]) is used to show the invertibility of matrices that arise in the similarity transform derivations, in section 3.5 and section 4.5.

Theorem 2.17 ([44],Corollary 2.38) *Let $\alpha_1, \alpha_2, \ldots, \alpha_n$ be elements of $GF(q^n)$. Then $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ is a basis of $GF(q^n)$ over $GF(q)$ if and only if*

$$\begin{vmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^q & \alpha_2^q & \cdots & \alpha_n^q \\ \vdots & \vdots & & \vdots \\ \alpha_1^{q^{n-1}} & \alpha_2^{q^{n-1}} & \cdots & \alpha_n^{q^{n-1}} \end{vmatrix} \neq 0.$$

This concludes the technical background material. The next section discusses previous work on various aspects of CA, relating to both applications and theory.

Figure 2.12: Circuit testing



2.4 Previous work

This section describes work by other authors relating to CA and their applications. Many of the cited materials contain a mixture of applications and theory, and so these two subjects are treated together. The majority of the applications discussed here are VLSI related, the most common of which is circuit testing. Figure 2.12 shows a typical scenario, explained below.

The papers cited in this section fall into several broad categories:

- applications of CA, with experimental/bench mark results,
- theoretical results concerning CA, typically linear hybrid CA, usually from the computer science/engineering community,
- a mixture of the above two,
- theoretical work on CA (typically non-linear uniform CA), primarily from the applied mathematics community, and
- purely mathematical material, that contains no mention of CA.

The LFSMs as defined to this point have all been *autonomous.* Such machines have no input: they are initialised with a start state, and then proceed according to their next-state function. Autonomous machines can be used as *test pattern generators* or pseudorandom number generators.

A non-autonomous LFSM is one that has input. When this is the case, an external source of data is XORed to the state of the LFSM at each time step. The data source can be either a single-bit stream, in which case it is fed into a single cell of the LFSM, or a stream of width $k < n$, in which case it is fed in parallel into k of the cells. When an LFSM has input, the state diagram contains an arc for each state/input combination. In VLSI, non-autonomous machines are commonly used as *output response compactors*. The outputs from the circuit under test are fed into the LFSM. The final state of the LFSM is the *signature* of the test, and is compared to a known correct value. For applications over non-binary fields, see [59].

The resurgence of interest in CA started with the work of Wolfram. Aspects such as chaotic behaviour, self-organisation and computational complexity are explored in [69] and [70]. In [71], Wolfram studies the randomness properties of CA. The CA considered by Wolfram are uniform and (usually) nonlinear, and use various boundary configurations.

One of the early works by VLSI researchers is [57]. In this paper, Pries *et al.* study CA that are one-dimensional, linear, and uniform. The authors study conditions under which the CA is reversible. Hortensius ([36]) and Hortensius *et al.* ([35] and [34]) compare CA with LFSRs for pseudorandom number generation and signature analysis. The authors determine that CA have less cross correlation in their output stream, and hence may be better as test pattern generators. Further comparative work in this vein, for hybrid CA based on rules 90 and 150, is in [65] and [66]. The authors argue that CA have better randomness properties and VLSI layout advantages when compared to LFSRs.

The VLSI group at the University of Victoria has pursued several CA projects, primarily application-related. The early theoretical work culminated in [61], which shows that CA and LFSRs are isomorphic as single-input data compactors. The CA recurrence is proved, as well as the fact that rules other than 90 and 150 give reducible characteristic polynomials. An almost-brute-force synthesis from a polynomial algo-

rithm is presented. In [48], the fault coverages using CA and LFSRs for BIST are determined for various benchmark circuits. In [64], a scheme that shares hardware for concurrent checking and offline BIST is presented. This scheme utilises the concatenation properties of CA. Serra and Slater present a CA synthesis algorithm based on the Lanczos tridiagonalisation method in [60]. They also produced tables of CA realisations of all irreducible polynomials up to degree 16.

Kontopidi and Muzio explore the partitioning properties of CA and LFSRs ([42]). This empirical study finds the number of irreducible and primitive CA and LFSRs that can be broken into smaller machines that are irreducible or primitive. The authors consider the hardware cost of having a machine that can operate either normally or in partitioned mode, and also the effect of allowing small changes to be made to the machines. They conclude that CA have better partitioning properties than LFSRs.

Several theses and dissertations from the University of Victoria have involved CA. Hassan ([31]) performs a classification of primitive polynomials, implemented as either CA of LFSRs, in terms of signature analysis aliasing probability. Janoowalla ([37]) examines the performanc[illegible]wo-dimensional CA when used as pseudorandom number generators and test patt[illegible]n generators. Kontopidi's work ([41]) leads to the material discussed in the previous paragraph. Sun ([63]) makes use of the partition-[illegible]oncatenation properties of CA in a mixed concurrent/off0list BIST approach. Zhang ([72]) analyses the transition properties of CA for detecting sequential faults in circuits.

In [1], Bardell discusses aspects of the similarity transform between CA and LFSRs. He also conjectures that no cyclic-boundary CA has a primitive characteristic polynomial, and that for every primitive polynomial there exists a CA. The former is proved in Chapter 3, and the latter in Chapter 6. The use of discrete logarithms for analysing phase shifts in CA and LFSRs is introduced, and explored in more detail in [2].

In [18], Chen and Gupta derive necessary and sufficient conditions for a CA or

LFSR to generate maximal pattern-pair coverage. The authors conclude that CA are better than LFSRs as test pattern generators for two-pattern testing.

In [25], Das *et al.* conjecture, based on empirical evidence, that rules 90 and 150 are necessary for maximal length cycles, and that the reversal of a maximal length CA is also maximal length. The synthesis problem is stated as being reducible to solving a set of n non-linear equations. In [26], the same authors propose that CA be used as signature analysers. They show that the steady-state aliasing probability for an n-cell CA can be lower than 2^{-n}, under the assumption that the probability of error in each bit is at least $1/2$ (note that under this model, LFSRs can also achieve such aliasing probabilities). They claim that CA outperform LFSRs as signature analysers. In [24], the reversibility of CA that use mixed linear and complemented-linear rules are explored. In [23], CA that use rules 60, 90, 102 and 150 are proposed for use as test pattern generators. The authors study conditions on such CA that guarantee that the output from a subset of $m < n$ cells generates all 2^m patterns. This is carried forward in [52], where the authors consider CA of which a subset of the cells generate all possible pairs and triples of consecutive patterns.

Damarla and Sathaye ([22]) derive several properties of the characteristic polynomials of CA. The general CA recurrence is derived, and a palindromic CA is shown to have a characteristic polynomial that is a sum of squares. Also, the effect of complementing the rule used by the middle cell of a palindromic CA is given. The authors go on to describe a method of computing the period of a CA, and discuss applications for pseudo-exhaustive testing.

In Boubezari and Kaminska ([8]), nonlinear CA that produce a set of precomputed test vectors are synthesised. Similar work is done in [53], under the name "multiple weighted cellular automata."

An automated system for generating self-test hardware based on CA is explored in [67]. The CA used are linear and uniform, with the boundary inputs being constant-0 or constant-1. The authors' system guarantees 100 percent fault coverage. Similar

work by the same authors is in [68].

CA have also been used in the design of error-correcting codes. For example, [19] and [20] describe "a novel approach for designing byte error-correcting codes." The authors claim that their design is simpler and less expensive than that for Reed-Solomon codes.

Some authors have studied various aspects of linear finite state machines over non-binary finite fields. For example, it is proposed in [27] that multiple-valued multiple input shift registers perform better than binary ones for output compaction (signature analysis) of multiple-valued circuits.

Nonlinear cyclic-boundary hybrid CA restricted to rules 30 and 153 are studied by Khare and Albicki ([39]). An exhaustive examination by simulation of the state spaces of all 8-cell machines is presented. The authors allow for a "Möbius loop," where a complementation can occur on the cyclic connection. A CA was found that generates a test set for the 74181 ALU.

In a paper by Motzkin ([50]), optimisation problems for functions defined on linear graphs are examined. In the course of this work, cyclic Gauss brackets are defined, which turn out to be related to cyclic-boundary CA (see section 5.2). For unrelated but interesting theoretical material on Euclid's GCD algorithm, see [49].

In [32], the authors claim that the number of CA with characteristic polynomials that are divisible by x (such polynomials are called "infirmative") is $(2^n - (-1)^n)/3$, i.e. roughly one third. Their proof is unavailable, but the result can be shown by a fairly straightforward inductive argument. The authors also show that palindromic CA have reducible characteristic polynomials.

The existence of worst-cases for Euclid's GCD algorithm in GF(2)[x] provides the proof of the existence of CA for irreducible polynomials. The underpinnings of this area are contained in [5] and [6], both works by Baum and Sweet. The actual result is by Mesirov and Sweet, in [47]. The problem is still open for finite fields other that GF(2) ([46]), as is the conjecture that every (not just irreducible) polynomial

in GF(2)[x] has a GCD calculation in which all quotients have degree 1 or 2 (i.e. an almost worst-case calculation). A modern detailed analysis of this area can be found in [54].

Theoretical aspects of uniform CA limited to rules 60 and 90 can be found in [43] and [55], respectively.

Clementi and Impagliazzo study the reachability problem for invertible and additive CA ([21]). Jen ([38]) analyses one-dimensional non-linear uniform CA with a possibly infinite number of cells. A slightly different form of linear CA, where the transition matrix is circulant, are studied by Guan and He ([28]). These CA are uniform, but not restricted to nearest-neighbour communication. Conditions for reversibility are derived for multi-dimensional CA on finite lattices.

In the course of this research, several papers and technical reports have been written. In the conference paper [16], some basic results on characteristic polynomials of CA over GF(q) are presented. The proof of reducibility of cyclic CA characteristic polynomials is shown in [11]. The non-technical and technical descriptions of the synthesis algorithm for irreducible polynomials is contained in [12] and [13], respectively. The algorithm was applied to the 300 minimal weight primitive polynomials listed in [4], the results of which are in [14]. Recent journal submissions are [10] (a paper describing the synthesis algorithm), and [15], which contains various results for CA over GF(q). A list of minimal-cost CA with up to 500 cells is contained in [17]. This table is reproduced in Appendix D.

2.5 Computational issues

This section briefly discusses issues of computations in finite fields. The computer plays a large role in the results of this dissertation. It allows for hypothesis testing, formula verification, and the generation and analysis of empirical data.

Virtually all of the computer calculations were performed with the University of Waterloo's Maple symbolic mathematical software. At the start of this research

Maple version 4.3 was available, which had fairly good capabilities for working with finite fields. The release of Maple V contained several improvements.

- Maple is typically not efficient, as it is inherently general purpose.
- The modp1 library provides high-speed arithmetic for univariate polynomials over $\mathbb{Z}_n$. This can be used for simple arithmetic in finite fields, or for polynomial calculations over prime order fields. Most of the material in this dissertation uses modp1, as it concerns polynomials over GF(2). The following code fragments are examples of working with polynomials over prime fields, and of arithmetic in non-prime fields.

```
################## Polynomials over GF(2) #################
# define two polynomials
a := modp1( ConvertIn( x^3+x+1, x ), 2 );
                                  a := 1000000010001
b := modp1( ConvertIn( x^3+x^2+1, x ), 2 );
                                  b := 1000100000001
# multiply them, and add 1
c := modp1( Add( Multiply( a, b ), One() ), 2 );
                          c := 10001000100010000100010000
modp1( ConvertOut( c, x ), 2 );
                              6    5    4    3    2
                             x  + x  + x  + x  + x  + x

################## Arithmetic in GF(16) #################
# Define GF(16) via irreducible polynomial p
p := modp1( ConvertIn( x^4+x+1, x ), 2 );
                               p := 10000000000010001
#
c := modp1( Rem( c, p ), 2 );
                                    c := 100010001
modp1( ConvertOut( c, x ), 2 );
                                      2
                                     x  + x + 1
```

- The evalgf library supports polynomials over non-prime fields. The following example multiplies and calculates the GCD of two polynomials over GF(8):

```
readlib(evalgf);
```

```
# define GF(8) via a root of p
p := z^3+z+1;
                                        3
                                  p := z  + z + 1

# define y as an alias for a root of p
alias( y = RootOf( p, z ) mod 2 ):

# define two polynomials
a := x^3 + (y+1)*x^2 + y*x;
                                    3           2
                            a := x  + (y + 1) x  + y x

b := y*x^2 + y*x;
                                      2
                                 b := y x  + y x

# multiple them
collect( evalgf( Expand(a*b), 2 ), x );
                              5     2  4    3     2  2
                             x  y + y  x  + x  y + y  x


# calculate their GCD
evalgf( Gcd( a, b ), 2 );
                                      2
                                     x  + x
```

- Maple's lack of efficiency is amply compensated by its flexibility and reliability.

This concludes the background material for the dissertation. The following chapter is the start of the analysis of the characteristic polynomials of CA.

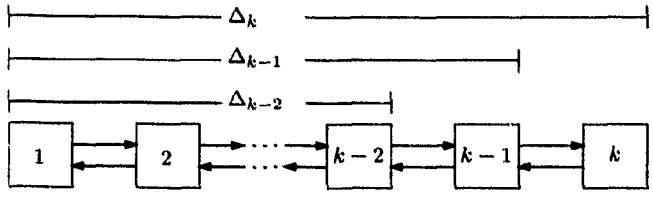
Chapter 3

Fundamentals of CA Characteristic Polynomials

This chapter explores general algebraic properties of CA characteristic polynomials. Several relationships satisfied by a CA characteristic polynomial are shown, and a similarity transform between CA and LFSRs is defined and derived. An alternative structure known as a *cyclic CA* is defined and analysed. The results in this chapter are used repeatedly in further chapters.

In section 3.1, the recurrence relation that is fundamental to the study of CA is presented. The application of this recurrence to the efficient computation of CA characteristic polynomials is discussed in section 3.2. Section 3.3 derives both a generalisation of the recurrence, and a formula that is related to greatest common divisors (GCDs). In section 3.4, it is shown that the transition matrix of a CA is non-derogatory. Similarity transforms from CA to diagonal form and companion form are presented in section 3.5. Section 3.6 explores fully-connected cyclic-boundary CA, and demonstrates how the characteristic polynomials of these machines are related to those of null-boundary CA. Further, it is shown that the characteristic polynomials of cyclic CA are always reducible (they contain large perfect squares), solving the conjecture by Bardell ([1]).

Figure 3.1: Submachines related by the CA recurrence (3.1)



3.1 CA recurrence

Central to the study of CA characteristic polynomials is the *CA recurrence*. This recurrence provides an efficient means of calculating the characteristic polynomial of a CA (discussed in section 3.2), and is the basis for many of the results in this dissertation. The recurrence was first shown in [61]; the proof is included here for completeness.

The recurrence relates the characteristic polynomials of three submachines of a CA, shown in Figure 3.1. Recall that Δ_k is the characteristic polynomial of the submachine consisting of cells 1 through k. The recurrence states that Δ_k is a function of d_k (the 90/150 control for cell k), Δ_{k-1}, and Δ_{k-2}. This means that for the $k = 1$ and $k = 2$ cases, the recurrence depends on Δ_{-1} and Δ_0. In section 2.3, these expressions are defined as

$$\begin{aligned} \Delta_{-1} &= \Delta_{i,i-2} = 0 \\ \Delta_0 &= \Delta_{i,i-1} = 1. \end{aligned}$$

The proof of the CA recurrence shows that with these definitions of Δ_{-1} and Δ_0, the recurrence is correct for $k = 1$ and $k = 2$. These two cases are handled separately, as the recurrence involves characteristic polynomials of 'empty' machines. The general case is then shown directly.

Theorem 3.1 (CA recurrence [61]) *Let M be a CA with rule vector $d = [d_1, d_2, \ldots, d_n]$. Then the characteristic polynomials of the submachines of M satisfy the*

following recurrence:

$$\begin{aligned} \Delta_{-1} &= 0 \\ \Delta_0 &= 1 \\ \Delta_k &= (x+d_k)\Delta_{k-1} + \Delta_{k-2}, \quad 1 \le k \le n. \end{aligned} \tag{3.1}$$

Proof. For $k = 1$, (3.1) gives $\Delta_1 = x + d_1$, which is the characteristic polynomial of the 1 by 1 matrix $[d_1]$. Hence the recurrence holds for $k = 1$.

For $k = 2$, the recurrence states that $\Delta_2 = (x + d_2)(x + d_1) + 1$. This is indeed the characteristic polynomial of the 2 by 2 matrix

$$\begin{bmatrix} d_1 & 1 \\ 1 & d_2 \end{bmatrix}.$$

To prove the general case, let A be the characteristic matrix of a k-cell CA,

$$A = \begin{bmatrix} x+d_1 & 1 & 0 & \cdots & \cdots & 0 & 0 \\ 1 & x+d_2 & 1 & \ddots & & & 0 \\ 0 & 1 & x+d_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & x+d_{k-2} & 1 & 0 \\ 0 & & & \ddots & 1 & x+d_{k-1} & 1 \\ 0 & 0 & \cdots & \cdots & 0 & 1 & x+d_k \end{bmatrix}.$$

By expansion along the last row,

$$\det(A) = (x + d_k)\det(B) + \det(C)$$

where

$$B = \begin{bmatrix} x+d_1 & 1 & 0 & \cdots & \cdots & 0 & 0 \\ 1 & x+d_2 & 1 & \ddots & & & 0 \\ 0 & 1 & x+d_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & x+d_{k-3} & 1 & 0 \\ 0 & & & \ddots & 1 & x+d_{k-2} & 1 \\ 0 & 0 & \cdots & \cdots & 0 & 1 & x+d_{k-1} \end{bmatrix},$$

and

$$C = \begin{bmatrix} x+d_1 & 1 & 0 & \cdots & \cdots & 0 & 0 \\ 1 & x+d_2 & 1 & \ddots & & & 0 \\ 0 & 1 & x+d_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & x+d_{k-3} & 1 & 0 \\ 0 & & & \ddots & 1 & x+d_{k-2} & 0 \\ 0 & 0 & \cdots & \cdots & 0 & 1 & 1 \end{bmatrix}.$$

Now, B is the characteristic matrix of the CA with the last cell removed, and so

$$\det(B) = \Delta_{k-1}.$$

The matrix C is expanded along the last column, resulting in the characteristic matrix of the CA with the last two cells removed. So

$$\det(C) = \Delta_{k-2},$$

and thus

$$\det(A) = (x+d_k)\Delta_{k-1} + \Delta_{k-2}.$$

□

As discussed in section 2.2, a CA has left-right symmetry, in that the machine is essentially unchanged if the cell labelings are reversed. Hence (3.1) can be stated equally well in terms of "right-side" submachines,

$$\Delta_{k,n} = (x+d_k)\Delta_{k+1,n} + \Delta_{k+2,n}. \tag{3.2}$$

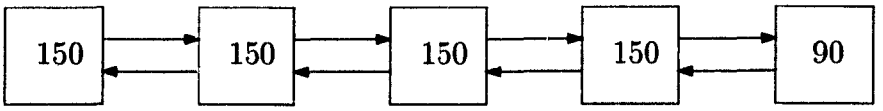
More generally, the relationship holds for any submachine of a CA ((3.1) and (3.2) are for submachines that contain cell 1 and cell n, respectively). This means that the following two generalisations hold:

$$\Delta_{i,k} = (x+d_k)\Delta_{i,k-1} + \Delta_{i,k-2}, \quad i \leq k \tag{3.3}$$

and

$$\Delta_{k,i} = (x+d_k)\Delta_{k+1,i} + \Delta_{k+2,i}, \quad k \leq i \tag{3.4}$$

Figure 3.2: A 5-cell CA



These relationships are used frequently throughout this dissertation, playing key roles in many of the results.

3.2 Computation of CA characteristic polynomials

This section demonstrates the calculation of the characteristic polynomial of a CA using the CA recurrence, and discusses the computational cost of this calculation. The general expressions for the characteristic polynomial of a CA are given for $n = 5$.

Consider the 5-cell CA pictured in Figure 3.2. The rule vector for this machine is

$$d = [d_1, d_2, d_3, d_4, d_5] = [1, 1, 1, 1, 0].$$

The computation of the characteristic polynomial starts with the definitions of $\Delta_{1,-1}$ and $\Delta_{1,0}$. Equation (3.1) is then applied to obtain $\Delta_{1,1}$, then reapplied to obtain $\Delta_{1,2}$, etc., until $\Delta_{1,n} = \Delta_n$ is obtained. This process is carried out in the following example.

Example 3.2. For the 5-cell CA with rule vector $[1, 1, 1, 1, 0]$,

$$\begin{aligned}
\Delta_{1,-1} &= 0 \\
\Delta_{1,0} &= 1 \\
\Delta_{1,1} &= (x+1)\cdot 1 + 0 && = x+1 \\
\Delta_{1,2} &= (x+1)\cdot(x+1) + 1 && = x^2 \\
\Delta_{1,3} &= (x+1)\cdot(x^2) + x + 1 && = x^3 + x^2 + x + 1 \\
\Delta_{1,4} &= (x+1)\cdot(x^3 + x^2 + x + 1) + x^2 && = x^4 + x^2 + 1 \\
\Delta_{1,5} &= (x+0)\cdot(x^4 + x^2 + 1) + x^3 + x^2 + x + 1 && = x^5 + x^2 + 1.
\end{aligned}$$

Hence the characteristic polynomial is $\Delta = \Delta_5 = x^5 + x^2 + 1$. □

For illustrative purposes, the computation of the characteristic polynomial via (3.2) is shown below. Note that the use of this equation is equivalent to reversing the

labeling of the cells.

$$\begin{aligned}
\Delta_{7,5} &= 0 \\
\Delta_{6,5} &= 1 \\
\Delta_{5,5} &= (x+0)\cdot 1 + 0 && = x \\
\Delta_{4,5} &= (x+1)\cdot (x) + 1 && = x^2 + x + 1 \\
\Delta_{3,5} &= (x+1)\cdot (x^2+x+1) + x && = x^3 + x + 1 \\
\Delta_{2,5} &= (x+1)\cdot (x^3+x+1) + x^2 + x + 1 && = x^4 + x^3 + x \\
\Delta_{1,5} &= (x+1)\cdot (x^4+x^3+x) + x^3 + x + 1 && = x^5 + x^2 + 1
\end{aligned}$$

It is evident that each application of the CA recurrence involves a polynomial multiplication and a polynomial addition. However, the polynomial multiplication always involves a degree 1 polynomial, and so it can be performed with a shift and an addition. This gives the following result.

Lemma 3.3 *The characteristic polynomial of a CA can be calculated with $2n$ polynomial additions and n polynomial shifts.*

Not surprisingly, since the CA transition matrix is so sparse, this is far less than the n^3 operations required to compute the characteristic polynomial of a general LFSM.

The characteristic polynomial can be written as an explicit function of $a_1, a_2, \ldots, a_n$, where a_i denotes $x + d_i$. This provides a closed formula, which, when evaluated for specific a_i, gives the characteristic polynomial. For example, with $n = 5$,

$$\begin{aligned}
\Delta_{-1} &= 0 \\
\Delta_0 &= 1 \\
\Delta_1 &= a_1 \\
\Delta_2 &= a_1 a_2 + 1 \\
\Delta_3 &= a_1 a_2 a_3 + a_1 + a_3 \\
\Delta_4 &= a_1 a_2 a_3 a_4 + a_1 a_2 + a_1 a_4 + a_3 a_4 + 1 \\
\Delta_5 &= a_1 a_2 a_3 a_4 a_5 + a_1 a_2 a_3 + a_1 a_2 a_5 + a_1 a_4 a_5 + a_3 a_4 a_5 + a_1 + a_3 + a_5
\end{aligned}$$

The patterns that are evident in these expressions are explored further in section 5.2.

If a_i is expanded to $x + d_i$, the expressions quickly become unwieldy:

$$\begin{aligned}
\Delta_{-1} &= 0 \\
\Delta_0 &= 1 \\
\Delta_1 &= x + d_1 \\
\Delta_2 &= x^2 + (d_1 + d_2)x + d_1d_2 + 1 \\
\Delta_3 &= x^3 + (d_1 + d_2 + d_3)x^2 + (d_1d_2 + d_1d_3 + d_2d_3)x + d_1d_2d_3 + d_1 + d_3 \\
\Delta_4 &= x^4 + (d_1 + d_2 + d_3 + d_4)x^3 \\
&\quad +(d_1d_2 + d_1d_3 + d_1d_4 + d_2d_3 + d_2d_4 + d_3d_4 + 1)x^2 \\
&\quad +(d_1d_2d_3 + d_1d_2d_4 + d_1d_3d_4 + d_2d_3d_4 + d_2 + d_3)x \\
&\quad +d_1d_2d_3d_4 + d_1d_2 + d_1d_4 + d_3d_4 + 1 \\
\Delta_5 &= x^5 + (d_1 + d_2 + d_3 + d_4 + d_5)x^4 \\
&\quad +(d_1d_2 + d_1d_3 + d_1d_4 + d_1d_5 + d_2d_3 + d_2d_4 \\
&\quad\quad + d_2d_5 + d_3d_4 + d_3d_5 + d_4d_5)x^3 \\
&\quad +(d_1d_2d_3 + d_1d_2d_4 + d_1d_2d_5 + d_1d_3d_4 + d_1d_3d_5 + d_1d_4d_5 \\
&\quad\quad +d_2d_3d_4 + d_2d_3d_5 + d_2d_4d_5 + d_3d_4d_5 + d_1 + d_5)x^2 \\
&\quad + (d_1d_2d_3d_4 + d_1d_2d_3d_5 + d_1d_2d_4d_5 + d_1d_3d_4d_5 \\
&\quad\quad +d_2d_3d_4d_5 + d_1d_3 + d_1d_4 + d_2d_3 + d_2d_5 + d_3d_4 + d_3d_5 + 1)x \\
&\quad +d_1d_2d_3d_4d_5 + d_1d_2d_3 + d_1d_2d_5 + d_1d_4d_5 + d_3d_4d_5 + d_1 + d_3 + d_5.
\end{aligned}$$

Though some structure is evident here, is would be difficult to work with these expressions directly.

3.3 Two important relations

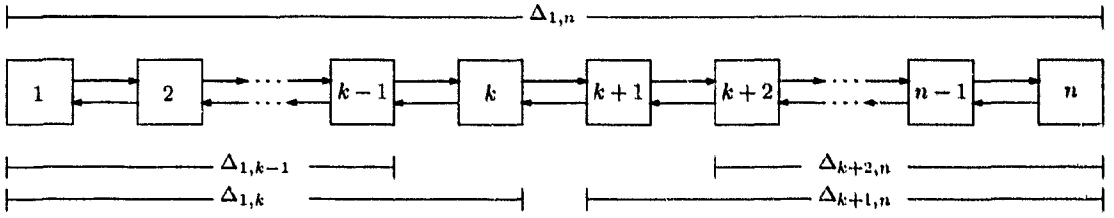
This section presents two relations that are satisfied by CA characteristic polynomials. The first describes the result of concatenating two CA, and is a generalisation of the CA recurrence. The second is a relation that, in Chapter 6, is shown to be intimately related to Euclid's GCD algorithm.

The *concatenation relation* gives the characteristic polynomial of the CA formed by concatenating two CA. The result is stated in terms of breaking an n-cell CA into components $1, 2, \ldots, k$ and $k+1, k+2, \ldots, n$ (see Figure 3.3).

Theorem 3.4 (Concatenation Relation) *For* $0 \leq k \leq n$,

$$\Delta_{1,n} = \Delta_{1,k}\Delta_{k+1,n} + \Delta_{1,k-1}\Delta_{k+2,n}. \tag{3.5}$$

Figure 3.3: Submachines related by the concatenation relation



Proof. The base case of the inductive proof is for $k = 0$. Using the definitions of $\Delta_{i,i-2}$ and $\Delta_{i,i-1}$ as 0 and 1 respectively, the theorem claims that

$$\begin{aligned} \Delta_{1,n} &= \Delta_{1,0}\Delta_{1,n} + \Delta_{1,-1}\Delta_{2,n} \\ &= 1 \cdot \Delta_{1,n} + 0 \cdot \Delta_{2,n} \\ &= \Delta_{1,n}, \end{aligned}$$

and hence is satisfied trivially.

Assuming that the theorem holds for k, it is shown that it holds for $k+1$. By the inductive hypothesis,

$$\Delta_{1,n} = \Delta_{1,k}\Delta_{k+1,n} + \Delta_{1,k-1}\Delta_{k+2,n}.$$

Applying (3.2) to $\Delta_{k+1,n}$ gives

$$\Delta_{1,n} = \Delta_{1,k}((x + d_{k+1})\Delta_{k+2,n} + \Delta_{k+3,n}) + \Delta_{1,k-1}\Delta_{k+2,n},$$

and by rearranging terms,

$$\Delta_{1,n} = \Delta_{1,k}\Delta_{k+3,n} + (x + d_{k+1})\Delta_{1,k}\Delta_{k+2,n} + \Delta_{1,k-1}\Delta_{k+2,n}.$$

Factoring out $\Delta_{k+2,n}$,

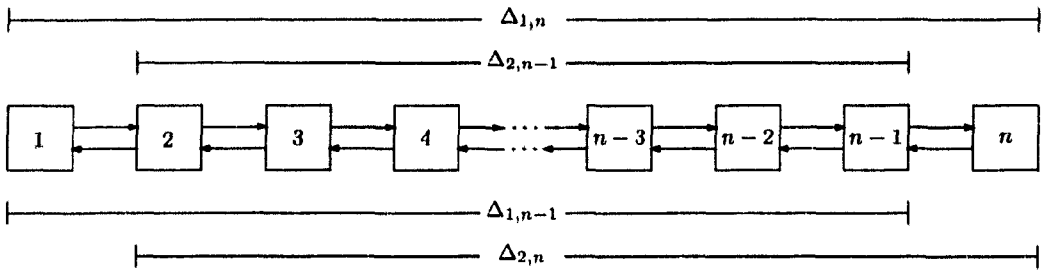
$$\Delta_{1,n} = ((x + d_{k+1})\Delta_{1,k} + \Delta_{1,k-1})\Delta_{k+2,n} + \Delta_{1,k}\Delta_{k+3,n}.$$

Applying (3.1),

$$\Delta_{1,n} = \Delta_{1,k+1}\Delta_{k+2,n} + \Delta_{1,k}\Delta_{k+3,n}.$$

Hence the formula holds for $k + 1$, given that it holds for k. □

Figure 3.4: Submachines related by the GCD relation



Example 3.5. Consider the 5-cell CA in Example 3.2 with rule vector $[1,1,1,1,0]$. With $k = 2$,

$$\begin{aligned}\Delta_{1,5} &= \Delta_{1,2}\Delta_{3,5} + \Delta_{1,1}\Delta_{4,5} \\ &= (x^2)(x^3+x+1) + (x+1)(x^2+x+1) \\ &= x^5 + x^2 + 1.\end{aligned}$$

□

The CA recurrence (3 1) is a special case of the concatenation relation. If $k = n-1$, (3.5) can be simplified as

$$\Delta_{1,n} = \Delta_{1,n-1}\Delta_{n,n} + \Delta_{1,n-2}\Delta_{n+1,n}.$$

Recalling that $\Delta_{i+1,i}$ is defined to be 1, this further simplifies to

$$\Delta_{1,n} = (x + d_n)\Delta_{1,n-1} + \Delta_{1,n-2},$$

which is the statement of Theorem 3.1.

The *GCD relation*, though similar in appearance, is actually quite different to the concatenation relation. It relates the characteristic polynomials of the submachines shown in Figure 3.4. In Chapter 6, it is shown that the relation can be derived from the connection between CA characteristic polynomials and Euclid's GCD algorithm.

Theorem 3.6 (GCD relation) *For $n \geq 1$,*

$$\Delta_{1,n}\Delta_{2,n-1} + \Delta_{1,n-1}\Delta_{2,n} = 1.$$

Proof. The proof is by induction, with the base case being $n = 1$. To this end,

$$\begin{aligned} & \Delta_{1,n}\Delta_{2,n-1} + \Delta_{1,n-1}\Delta_{2,n} \\ = \; & \Delta_{1,1}\Delta_{2,0} + \Delta_{1,0}\Delta_{2,1} \\ = \; & (x + d_1) \cdot 0 + 1 \cdot 1 \\ = \; & 1 \end{aligned}$$

proving the base case. Assuming inductively that the theorem holds for $n - 1$, it is shown that it holds for n.

$$\begin{aligned} & \Delta_{1,n-1}\Delta_{2,n-2} + \Delta_{2,n-1}\Delta_{1,n-2} = 1 \\ \Rightarrow \; & \Delta_{2,n-1}\Delta_{1,n-2} + (x + d_n)\Delta_{2,n-1}\Delta_{1,n-1} \\ & = 1 + \Delta_{1,n-1}\Delta_{2,n-2} + (x + d_n)\Delta_{2,n-1}\Delta_{1,n-1} \\ \Rightarrow \; & \Delta_{2,n-1}((x + d_n)\Delta_{1,n-1} + \Delta_{1,n-2}) = 1 + \Delta_{1,n-1}((x + d_n)\Delta_{2,n-1} + \Delta_{2,n-2}) \\ \Rightarrow \; & \Delta_{1,n}\Delta_{2,n-1} + \Delta_{1,n-1}\Delta_{2,n} = 1. \end{aligned}$$

□

Example 3.7. Consider the 5-cell CA from Example 3.5.

$$\begin{aligned} & \Delta_{1,5}\Delta_{2,4} + \Delta_{1,4}\Delta_{2,5} \\ = \; & (x^5 + x^2 + 1)(x^3 + x^2 + x + 1) + (x^4 + x^2 + 1)(x^4 + x^3 + x) \\ = \; & (x^8 + x^7 + x^6 + x^4 + x + 1) + (x^8 + x^7 + x^6 + x^4 + x) \\ = \; & 1 \end{aligned}$$

□

The connection between CA characteristic polynomials and polynomial GCDs is explored further in Chapter 6.

3.4 Non-derogatory LFSMs

This section describes a condition under which an LFSM has the property of being *non-derogatory*. This property is important for the material in section 3.5.

As discussed in section 2.3, LFSMs are often studied via their characteristic polynomials, though it is actually the *minimal polynomial* of an LFSM that determines the linear operator that the LFSM represents. However, if an LFSM is non-derogatory, these polynomials are equal. The following definitions and theorems present the necessary background and results. Unless otherwise stated, A is the transition matrix of an n-cell LFSM (and hence has dimension n).

Definition 3.8 ([33]) *The* minimal polynomial of A, *denoted by* $\min(A)$, *is the least-degree monic polynomial such that*

$$\min(A) = 0.$$

The minimal polynomial is uniquely determined by Definition 3.8. The following lemma and corollaries relate the minimal polynomial and the characteristic polynomial.

Lemma 3.9 (Cayley-Hamilton, [33]) *The minimal polynomial of A divides the characteristic polynomial of A.*

The following two corollaries follow immediately from Lemma 3.9.

Corollary 3.10 ([33]) *If the minimal polynomial of A has degree n, then it equals the characteristic polynomial of A.*

Corollary 3.11 ([33]) *If the characteristic polynomial of A is irreducible, then it equals the minimal polynomial of A.*

It is the property described in Corollary 3.10 that is of interest. This is formulated as follows.

Definition 3.12 ([33]) *A is said to be* non-derogatory *or* cyclic *if the minimal polynomial of A has degree n.*

The term *cyclic* is not used in the dissertation, to avoid confusion with *cyclic CA* (which refers to boundary conditions). Note that Corollary 3.11 follows immediately from the characteristic polynomial being degree n.

A large class of matrices that includes all of the CA and LFSR (type-1 and type-2) transition matrices can be shown to be non-derogatory. To this end, the following definition is required. Note that if "upper" and "lower" are interchanged in the following material, the results still hold.

Definition 3.13 ([9]) *A is* upper Hessenburg *if it contains only zero entries below the lower subdiagonal. That is, $(A)_{i,j} = 0$ for all $i \geq j + 2$.*

A supporting lemma for the main theorem is required.

Lemma 3.14 *If A is upper Hessenburg and all of the subdiagonal entries are 1, then for $0 \leq k \leq n - 1$,*

1. *the kth subdiagonal of A^k is all 1, and*
2. *all entries below the kth subdiagonal of A^k are 0.*

Proof. For $k = 0$, the identity matrix A^0 clearly satisfies the lemma (the 0th subdiagonal is the main diagonal). The remainder of the cases are shown by induction, with the base case $k = 1$ holding trivially from the definition of A.

Consider the element $(A^k)_{i,i-k}$ on the kth subdiagonal of A^k, formed as the product of the ith row of A^{k-1} and the $(i-k)$th column of A. The ith row of A^{k-1} has $i - k$ leading 0s followed by a 1, and the $(i-k)$th column of A has a 1 in position $i-k+1$ followed by all 0s. Hence $(A^k)_{i,i-k} = 1$. By similar observations, $(A^k)_{i,j} = 0$ if $j > i - k$. □

Theorem 3.15 *A is non-derogatory if it is upper Hessenburg and all of the subdiagonal entries are 1.*

Proof. Let A be n-by-n upper Hessenburg, with all subdiagonal entries 1. From Lemma 3.14, the first column of A^k has the properties

$$\begin{aligned}(A^k)_{k+1,1} &= 1, \quad \text{and} \\ (A^k)_{i,1} &= 0, \quad k+2 \leq i \leq n\end{aligned}$$

for $k = 0, 1, \ldots, n-1$. Thus any non-empty linear combination

$$z_{n-1}A^{n-1} + z_{n-2}A^{n-2} + \cdots + z_2A^2 + z_1A^1 + z_0I,$$

$z_i \in \mathrm{GF}(2)$, has a 1 in the $(k,1)$ position, where k is the greatest index for which $z_i \neq 0$. Therefore any polynomial of degree $n-1$ or less in $\mathrm{GF}(2)[x]$ is non-zero when evaluated at A. This means that the minimal polynomial of A has degree n, and A is non-derogatory. □

Clearly both CA and LFSR transition matrices satisfy the conditions of Theorem 3.15, and therefore are non-derogatory. Note that the lemma is dependent on the lower subdiagonal being "full" (all of the entries must be 1).

3.5 Similarity transforms between CA and LFSR matrices

This section demonstrates a *similarity transform* from the transition matrix of an irreducible CA to diagonal form. By combining this transform with the transform from diagonal form to companion form, the similarity transform from a CA transition matrix to an LFSR transition matrix is obtained. The first part of this section derives the necessary background and motivates the exploration of similarity transforms.

As discussed in section 2.2, the transition matrix A is a *representation* or *realisation* of a linear operator L. Since A has elements from $\mathrm{GF}(2)$, L is a linear operator on the vector space $(\mathrm{GF}(2))^n$.

A linear operator is uniquely determined by its minimal polynomial. The minimal polynomial of a linear operator L is defined to be the minimal polynomial of any matrix representing L. Hence two matrices A and A' represent the same linear

operator if and only if they have the same minimal polynomial. This property is called *similarity.*

Definition 3.16 ([33]) *Two matrices A and A' are* similar *if they have the same minimal polynomial.*

Similarity is often defined via the statement in the following theorem.

Theorem 3.17 ([33]) *Two matrices A and A' are similar if and only if there exists a non-singular matrix P such that*

$$PAP^{-1} = A'.$$

The matrix P in Theorem 3.17 is called a *similarity transform* from A to A'.

Let M be an n-cell CA with transition matrix A and irreducible characteristic polynomial Δ, and let L be the linear operator represented by A. There are two "natural" matrix representations of L; the first is the companion matrix of Δ (or of L), and the second is the diagonal matrix of Δ (or of L).

The companion matrix of Δ is the transition matrix of the LFSR that has characteristic polynomial Δ. The companion matrix of a polynomial always exists, since it is formed by the simple process of writing the coefficients of the polynomial in the last column of the matrix and 1s in the subdiagonal.

To work with the diagonal matrix of Δ, L must be considered as a linear operator on the vector space $(\mathrm{GF}(2^n))^n$. Importantly, L is uniquely and well defined on $(\mathrm{GF}(2^n))^n$ by its definition on $(\mathrm{GF}(2))^n$ (see [44]).

Definition 3.18 ([33]) *A linear operator L is* diagonalisable *if it can be represented by a diagonal matrix; that is, if the matrix A represents L, then A is similar to a diagonal matrix.*

Theorem 3.19 ([33]) *A linear operator L is diagonalisable if and only if the minimal polynomial of L can be written as a product of distinct linear factors. Furthermore, if L is diagonalisable, then the entries on the diagonal of the diagonalised matrix are the roots of the minimal polynomial of L.*

Consider now the characteristic polynomial Δ of the CA M. As Δ is assumed to be irreducible, it has a root α in GF(2^n). It follows that all n roots of Δ lie in GF(2^n) and are distinct. Further, the roots are given by

$$\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{n-1}}.$$

This means that Δ can be factored in GF(2^n) as

$$(x+\alpha)(x+\alpha^2)(x+\alpha^4)\cdots(x+\alpha^{2^{n-1}}),$$

a product of non-repeated linear factors. Hence if Δ is irreducible over GF(2), then L is diagonalisable over GF(2^n).

The diagonal form of L has the roots of the characteristic polynomial on the main diagonal. As these roots can appear in any order, L has multiple diagonal forms ($n!$, as the roots are distinct). Without loss of generality, the following diagonal form is used:

$$D = \begin{bmatrix} \alpha & & & & \\ & \alpha^2 & & & \\ & & \ddots & & \\ & & & \alpha^{2^{n-2}} & \\ & & & & \alpha^{2^{n-1}} \end{bmatrix}.$$

The first goal of this section is to demonstrate a similarity transform P (a matrix over GF(2^n)) such that

$$PDP^{-1} = A$$

(recall that A is transition matrix of the CA M). In the computation of the characteristic polynomial Δ of A, a sequence of polynomials $1, \Delta_{1,1}, \Delta_{1,2}, \ldots, \Delta_{1,n-1}, \Delta_{1,n}$ is formed. Each of these polynomials is mapped to an element of GF(2^n) by the usual process of evaluation at the root of an irreducible polynomial in GF(2)[x]. The root α of Δ is used, and the result of this mapping is denoted m_i:

$$m_i = \Delta_{1,i}(\alpha),\ i = 0, \ldots, n \quad \text{(evaluation)}.$$

Note that $m_0 = \Delta_{1,0}(\alpha) = 1$, and $m_n = \Delta(\alpha) = 0$. The claim is that the similarity transform is given by

$$P = \begin{bmatrix} m_0 & m_0^2 & m_0^4 & \cdots & m_0^{2^{n-1}} \\ m_1 & m_1^2 & m_1^4 & \cdots & m_1^{2^{n-1}} \\ \vdots & \vdots & \vdots & & \vdots \\ m_{n-2} & m_{n-2}^2 & m_{n-2}^4 & \cdots & m_{n-2}^{2^{n-1}} \\ m_{n-1} & m_{n-1}^2 & m_{n-1}^4 & \cdots & m_{n-1}^{2^{n-1}} \end{bmatrix}.$$

To prove this claim, the matrix P is first shown to be non-singular, and then shown to be the required transform.

Lemma 3.20 *P is non-singular.*

Proof. By Theorem 2.17, the determinant $|P|$ is not zero if and only if $\{1, m_1, m_2, \ldots, m_{n-1}\}$ is a basis of GF(2^n) over GF(2). To show that the elements $\{1, m_1, m_2, \ldots, m_{n-1}\}$ form a basis of GF(2^n) over GF(2), it suffices to show that no non-empty linear combination of the m_i is zero. Since m_i is the image of a degree i polynomial (under the mapping "evaluation at α"), m_i has the form

$$\alpha^i + c_{i-1}\alpha^{i-1} + \cdots + c_1\alpha + c_0,$$

where the $c_j, 0 \leq j < i$, are the coefficients of the polynomial $\Delta_{1,i}$. Hence a linear combination

$$z_0 + z_1 m_1 + z_2 m_2 + \cdots + z_{n-1} m_{n-1} \tag{3.6}$$

has a term α^i, where i is the greatest index such that $z_i \neq 0$. Therefore (3.6) is not zero, so long as not all z_i are zero. □

Theorem 3.21 *P is a similarity transform between A and D. That is,*

$$PDP^{-1} = A.$$

Proof. It is shown that $PD = AP$, and since P is non-singular, the theorem follows. The product of the ith row of P and the jth column of D is

$$\begin{aligned} & (PD)_{i,j} \\ = \; & [m_{i-1}, m_{i-1}^2, \ldots, m_{i-1}^{2^{n-1}}] \cdot [0, 0, \ldots, \alpha^{2^{j-1}}, \ldots, 0]^{\mathrm{T}} \\ = \; & \alpha^{2^{j-1}} m_{i-1}^{2^{j-1}} \\ = \; & (\alpha m_{i-1})^{2^{j-1}}. \end{aligned}$$

The product of the ith row of A ($i \neq 1, n$), and the jth column of P is

$$\begin{aligned} & (AP)_{i,j} \\ = \; & [0, 0, \ldots, 1, d_j, 1, \ldots, 0] \cdot [1, m_1^{2^{j-1}}, \ldots, m_{n-1}^{2^{j-1}}]^{\mathrm{T}} \\ = \; & 1 \cdot m_{i-2}^{2^{j-1}} + d_i \cdot m_{i-1}^{2^{j-1}} + 1 \cdot m_i^{2^{j-1}} \\ = \; & (m_{i-2} + d_i \cdot m_{i-1} + m_i)^{2^{j-1}}. \end{aligned}$$

As the CA polynomials satisfy the recurrence

$$\Delta_i = (x + d_i)\Delta_{i-1} + \Delta_{i-2}, \;\; 2 \leq i \leq n,$$

in $\mathrm{GF}(2^n)$

$$\begin{aligned} m_i &= (\alpha + d_i)m_{i-1} + m_{i-2} \\ &= \alpha m_{i-1} + d_i m_{i-1} + m_{i-2}. \end{aligned}$$

This gives

$$(AP)_{i,j} = (\alpha m_{i-1})^{2^{j-1}}, \;\; 2 \leq i \leq n-1, \;\; 1 \leq j \leq n.$$

The boundary cases for $i = 1$ and $i = n$ are handled separately:

$$\begin{aligned} & (AP)_{1,j} \\ = \; & [d_1, 1, 0, \ldots, 0] \cdot [1, m_1^{2^{j-1}}, \ldots, m_{n-1}^{2^{j-1}}]^{\mathrm{T}} \\ = \; & d_1 + m_1^{2^{j-1}} \\ = \; & (d_1 + m_1)^{2^{j-1}}, \\ = \; & (d_1 + \alpha + d_1)^{2^{j-1}}, \\ = \; & (\alpha)^{2^{j-1}}, \\ = \; & (\alpha m_0)^{2^{j-1}}, \end{aligned}$$

and, using the fact that m_n is 0 (Δ evaluated at its own root),

$$\begin{aligned} & (AP)_{n,j} \\ = \; & [0, \ldots, 0, 1, d_n] \cdot [1, m_1^{2^{j-1}}, \ldots, m_{n-1}^{2^{j-1}}]^{\mathrm{T}} \\ = \; & m_{n-2}^{2^{j-1}} + d_n m_{n-1}^{2^{j-1}} \\ = \; & (m_{n-2} + d_n m_{n-1})^{2^{j-1}}, \\ = \; & (m_{n-2} + d_n m_{n-1} + m_n)^{2^{j-1}}, \\ = \; & (\alpha m_{n-1})^{2^{j-1}}. \end{aligned}$$

Hence

$$(AP)_{i,j} = (\alpha m_{i-1})^{2^{j-1}} = (PD)_{i,j},$$

and $PD = AP$. □

The transform is illustrated by an example.

Example 3.22. Consider the 5-cell CA from Example 3.2, with the rule vector $[1,1,1,1,0]$. The characteristic polynomials of the submachines of this CA are

$$\begin{aligned}
\Delta_0 &= 1 \\
\Delta_1 &= x+1 \\
\Delta_2 &= x^2 \\
\Delta_3 &= x^3+x^2+x+1 \\
\Delta_4 &= x^4+x^2+1 \\
\Delta_5 &= x^5+x^2+1.
\end{aligned}$$

The characteristic polynomial, Δ_5, is irreducible. The matrices P, D, and P^{-1} are given by

$$P = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 \\
\alpha+1 & \alpha^2+1 & \alpha^4+1 & \alpha^3+\alpha^2 & \alpha^4+\alpha^3+\alpha \\
\alpha^2 & \alpha^4 & \alpha^3+\alpha^2+1 & \alpha^4+\alpha^3+\alpha+1 & \alpha \\
\alpha^3+\alpha^2+\alpha+1 & \alpha^4+\alpha^3+\alpha^2+\alpha+1 & \alpha^4+\alpha & \alpha^3+1 & \alpha^3+\alpha+1 \\
\alpha^4+\alpha^2+1 & \alpha^4+\alpha^3+\alpha^2 & \alpha^4+\alpha^2+\alpha+1 & \alpha^4+\alpha^3 & \alpha^2+\alpha+1
\end{bmatrix},$$

$$D = \begin{bmatrix}
\alpha & 0 & 0 & 0 & 0 \\
0 & \alpha^2 & 0 & 0 & 0 \\
0 & 0 & \alpha^4 & 0 & 0 \\
0 & 0 & 0 & \alpha^3+\alpha^2+1 & 0 \\
0 & 0 & 0 & 0 & \alpha^4+\alpha^3+\alpha+1
\end{bmatrix},$$

$$P^{-1} = \begin{bmatrix}
\alpha^2+1 & \alpha^3+\alpha^2+\alpha+1 & \alpha^4+\alpha^2 & \alpha^4+\alpha+\alpha^2 & \alpha^3+\alpha+1 \\
\alpha^4+1 & \alpha^4+\alpha^3+\alpha^2+\alpha+1 & \alpha^4+\alpha^3+\alpha^2+1 & \alpha^3+1+\alpha^4 & \alpha^3+\alpha^2+\alpha+1 \\
\alpha^3+\alpha^2 & \alpha^4+\alpha & \alpha^4+\alpha+\alpha^2 & \alpha^2+\alpha & \alpha^4+\alpha^3+\alpha^2+\alpha+1 \\
\alpha^4+\alpha^3+\alpha & \alpha^3+1 & \alpha^3+1+\alpha^4 & \alpha^4+\alpha^2 & \alpha^4+\alpha \\
\alpha+1 & \alpha^3+\alpha+1 & \alpha^2+\alpha & \alpha^4+\alpha^3+\alpha^2+1 & \alpha^3+1
\end{bmatrix}.$$

The product PDP^{-1} is

$$\begin{bmatrix}
1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 0
\end{bmatrix},$$

which is the transition matrix A of the CA. □

Theorem 3.21 provides a general form for a similarity transform between a CA matrix A and the diagonal matrix D. From this, a similarity between the CA and its

corresponding LFSR is obtained, by finding a similarity between an LFSR and the diagonal form. To this end, define the matrix Q as follows

$$Q = \begin{bmatrix} \alpha & \alpha^2 & \alpha^3 & \cdots & \alpha^n \\ \alpha^2 & (\alpha^2)^2 & (\alpha^3)^2 & \cdots & (\alpha^n)^2 \\ \alpha^4 & (\alpha^2)^4 & (\alpha^3)^4 & \cdots & (\alpha^n)^4 \\ \vdots & \vdots & \vdots & & \vdots \\ \alpha^{2^{n-1}} & (\alpha^2)^{2^{n-1}} & (\alpha^3)^{2^{n-1}} & \cdots & (\alpha^n)^{2^{n-1}} \end{bmatrix}.$$

Lemma 3.23 *Q is non-singular.*

Proof. By Theorem 2.17, the determinant $|Q|$ is not zero if and only if $\{\alpha, \alpha^2, \alpha^3, \ldots, \alpha^n\}$ is a basis of GF(2^n) over GF(2). Clearly, the elements $\alpha, \alpha^2, \ldots, \alpha^{n-1}$ are linearly independent. Furthermore,

$$\alpha^n = 1 + c_1\alpha + c_2\alpha^2 + \cdots + c_{n-1}\alpha^{n-1},$$

where $c_i \in$ GF(2). The assumption that the polynomial Δ is irreducible ensures that the constant term is present. In turn, this ensures that α^n is not a linear combination of $\{\alpha, \alpha^2, \ldots, \alpha^{n-1}\}$. Thus $\{\alpha, \alpha^2, \ldots, \alpha^n\}$ is a set of n linearly independent vectors, and hence forms a basis. □

Theorem 3.24 *Q is a similarity transform between D and C. That is,*

$$QCQ^{-1} = D.$$

Proof. It is shown that $QC = DQ$, and since Q is non-singular, the theorem follows.

The product of the ith row of D and the jth column of Q is

$$\begin{aligned} & (DQ)_{i,j} \\ = \; & [0, \ldots, 0, \alpha^{2^{i-1}}, 0, \ldots, 0] \cdot [\alpha^j, (\alpha^j)^2, \ldots, (\alpha^j)^{2^{i-1}}, \ldots, (\alpha^j)^{2^{n-1}}]^{\mathrm{T}} \\ = \; & \alpha^{2^{i-1}}(\alpha^j)^{2^{i-1}} \\ = \; & (\alpha^{(j+1)})^{2^{i-1}} \\ = \; & \alpha^{(j+1)2^{i-1}}. \end{aligned}$$

The product of the ith row of Q and the jth column of C, with $j < n$, is

$$\begin{aligned} & (QC)_{i,j} \\ = \; & [\alpha^{2^{i-1}}, (\alpha^2)^{2^{i-1}}, \ldots, (\alpha^n)^{2^{i-1}}] \cdot \overbrace{[0, \ldots, 0, 1, 0, \ldots, 0]^{\mathrm{T}}}^{1 \text{ in entry } j+1} \\ = \; & (\alpha^{j+1})^{2^{i-1}} \\ = \; & \alpha^{(j+1)2^{i-1}}. \end{aligned}$$

For the case $j = n$, recall that the last column of C contains the coefficients of Δ, $[p_0, p_1, \ldots, p_{n-1}]^{\mathrm{T}}$:

$$\begin{aligned}
& (QC)_{i,n} \\
= \; & [\alpha^{2^{i-1}}, (\alpha^2)^{2^{i-1}}, \ldots, (\alpha^n)^{2^{i-1}}] \cdot [p_0, p_1, \ldots, p_{n-1}]^{\mathrm{T}} \\
= \; & (\alpha p_0 + \alpha^2 p_1 + \cdots + \alpha^n p_{n-1})^{2^{i-1}} \\
= \; & (\alpha(p_0 + \alpha p_1 + \cdots + \alpha^{n-1} p_{n-1}))^{2^{i-1}} \\
= \; & (\alpha\alpha^n)^{2^{i-1}} \\
= \; & \alpha^{(n+1)2^{i-1}}.
\end{aligned}$$

□

Example 3.25. Using the CA from Example 3.22, and obtaining the companion form from Δ_5, the matrices Q, C, and Q^{-1} are given by

$$Q = \begin{bmatrix}
x & x^2 & x^3 & x^4 & x^2+1 \\
x^2 & x^4 & x^3+x & x^3+x^2+1 & x^4+1 \\
x^4 & x^3+x^2+1 & x^3+x^2+x & x^4+x^3+x+1 & x^3+x^2 \\
x^3+x^2+1 & x^4+x^3+x+1 & x^4+x^3+x^2+x & x & x^4+x^3+x \\
x^4+x^3+x+1 & x & x^4+x+1 & x^2 & x+1
\end{bmatrix},$$

$$C = \begin{bmatrix}
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0
\end{bmatrix},$$

$$Q^{-1} = \begin{bmatrix}
\alpha^4+\alpha^3+1 & \alpha^2+\alpha & \alpha^2+\alpha^4 & \alpha^4+\alpha^3+\alpha^2+1 & \alpha^4+\alpha^2+\alpha \\
\alpha^4+\alpha^3+\alpha^2+\alpha & \alpha+1+\alpha^4 & \alpha^3 & \alpha^3+\alpha & \alpha^3+\alpha^2+\alpha \\
\alpha^4+\alpha^2+\alpha & \alpha^4+\alpha^3+1 & \alpha^2+\alpha & \alpha^2+\alpha^4 & \alpha^4+\alpha^3+\alpha^2+1 \\
\alpha^3+\alpha+1 & \alpha^3+\alpha^2+\alpha+1 & \alpha^4+\alpha^3+\alpha^2+\alpha+1 & \alpha^4+\alpha & \alpha^3+1 \\
\alpha^4+\alpha^2+\alpha+1 & \alpha^4+\alpha^3 & \alpha^2+\alpha+1 & \alpha^4+\alpha^2+1 & \alpha^4+\alpha^3+\alpha^2
\end{bmatrix}.$$

The product QCQ^{-1} is

$$\begin{bmatrix}
\alpha & 0 & 0 & 0 & 0 \\
0 & \alpha^2 & 0 & 0 & 0 \\
0 & 0 & \alpha^4 & 0 & 0 \\
0 & 0 & 0 & \alpha^3+\alpha^2+1 & 0 \\
0 & 0 & 0 & 0 & \alpha^4+\alpha^3+\alpha+1
\end{bmatrix},$$

which is the diagonal matrix D of Δ. □

To obtain the similarity transform between the CA matrix and the companion matrix, consider the following derivation. From Theorem 3.21,

$$A = PDP^{-1}.$$

Substituting from Theorem 3.24

$$\begin{aligned} A &= PQCQ^{-1}P^{-1} \\ &\quad (PQ)C(PQ)^{-1}. \end{aligned}$$

Hence the similarity transform between the CA matrix and the companion matrix is the matrix PQ. The entry $(PQ)_{i,j}, 1 \leq i,j \leq n$, can be determined as follows.

$$\begin{aligned} & (PQ)_{i,j} \\ = \; & [m_{i-1}, m_{i-1}^2, \ldots, m_{i-1}^{2^{n-1}}] \cdot [\alpha^j, (\alpha^j)^2, \ldots, (\alpha^j)^{2^{i-1}}, \ldots, (\alpha^j)^{2^{n-1}}]^{\mathrm{T}} \\ = \; & m_{i-1}\alpha^j + (m_{i-1}\alpha^j)^2 + \cdots + (m_{i-1}\alpha^j)^{2^{n-1}} \\ = \; & \mathrm{Tr}(m_{i-1}\alpha^j). \end{aligned}$$

Thus the matrix PQ is given by

$$PQ = \begin{bmatrix} \mathrm{Tr}(\alpha m_0) & \mathrm{Tr}(\alpha^2 m_0) & \mathrm{Tr}(\alpha^3 m_0) & \cdots & \mathrm{Tr}(\alpha^n m_0) \\ \mathrm{Tr}(\alpha m_1) & \mathrm{Tr}(\alpha^2 m_1) & \mathrm{Tr}(\alpha^3 m_1) & \cdots & \mathrm{Tr}(\alpha^n m_1) \\ \mathrm{Tr}(\alpha m_2) & \mathrm{Tr}(\alpha^2 m_2) & \mathrm{Tr}(\alpha^3 m_2) & \cdots & \mathrm{Tr}(\alpha^n m_2) \\ \vdots & \vdots & \vdots & & \vdots \\ \mathrm{Tr}(\alpha m_{n-1}) & \mathrm{Tr}(\alpha^2 m_{n-1}) & \mathrm{Tr}(\alpha^3 m_{n-1}) & \cdots & \mathrm{Tr}(\alpha^n m_{n-1}) \end{bmatrix}.$$

Example 3.26. Continuing Example 3.25, the product $(PQ)C(PQ)^{-1}$ is

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

as expected. □

This completes the material on similarity transforms for machines over GF(2). Section 4.5 contains extensions of this area to larger finite fields.

3.6 Cyclic CA

To this point, the focus has been exclusively on null-boundary CA, in which the leftmost and rightmost cells have constant-0 input from their non-existent neighbours. This section explores *fully-connected cyclic-boundary CA* (abbreviated *cyclic*

CA), a machine structure obtained by joining the leftmost and rightmost cells of a null-boundary CA. The characteristic polynomial of a cyclic CA is related to the characteristic polynomial of a null-boundary CA, and it is demonstrated that the characteristic polynomial of a cyclic CA is always reducible. To avoid confusion, throughout this section a null-boundary CA is referred to as such, rather than by the usual shortened name *CA*.

A null-boundary CA can be viewed as a machine restricted to local communication on a linear topology. If the local-communication restriction is kept, but the topology is changed to be circular, then the resulting machines have the structure shown in Figure 3.5.

A null-boundary CA has an obvious left-right symmetry. This symmetry is broken by the selection of one of the two end cells to be labeled as cell 1, thereby forcing the other end cell to be labeled as n. In a cyclic CA, any of the cells can be labeled as cell 1, and the labeling can proceed either left or right from the selected cell 1. Hence, instead of there being two ways in which to break the symmetry, there are $2n$. This means that there are $2n$ labelings of a cyclic CA, no one of which is preferred.

For a particular labeling of a cyclic CA, define the *underlying null-boundary CA* as the null-boundary CA obtained by removing the two connections between cell 1 and cell n. The labeling of the cyclic CA then induces a labeling on the underlying null-boundary CA. The notation $\Delta_{i,j}$ refers to the characteristic polynomial of the submachine of the underlying null-boundary CA (with respect to a fixed labeling).

Having chosen a labeling of the cells, the transition matrix of a cyclic CA can be formulated. The transition matrix is denoted by A_{cyc}, and is very similar to the transition matrix of the underlying null-boundary CA. The two extra connections (from cell 1 to cell n and from cell n to cell 1) correspond, respectively, to 1s in the $(n,1)$ and $(1,n)$ elements of A_{cyc}. As with null-boundary CA, d_i denotes the "self"

Figure 3.5: Structure of a fully connected cyclic CA

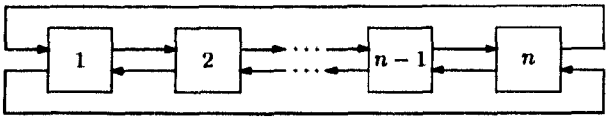
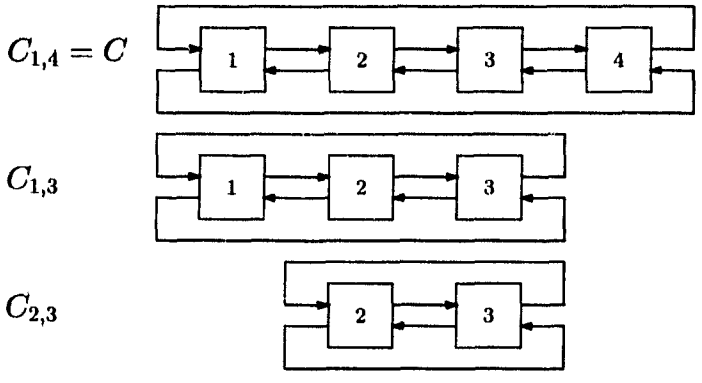


Figure 3.6: Structure of a fully connected cyclic CA



multiplier in cell i. Hence, the form of the cyclic CA transition matrix is

$$A_{\text{cyc}} = \begin{bmatrix} d_1 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 1 & d_2 & 1 & \ddots & & & 0 \\ 0 & 1 & d_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & d_{n-2} & 1 & 0 \\ 0 & & & \ddots & 1 & d_{n-1} & 1 \\ 1 & 0 & \cdots & \cdots & 0 & 1 & d_n \end{bmatrix} \tag{3.7}$$

The characteristic polynomial of a cyclic CA, denoted by $\Gamma_{1,n}$, is given by

$$\Gamma_{1,n} = |xI - A_{\text{cyc}}|.$$

As for a null-boundary CA, the notion of a *submachine* of a cyclic CA is required. The submachine consisting of cells i through j is denoted $C_{i,j}$. Figure 3.6 shows two submachines of a four cell cyclic CA. Note that the cyclic-boundary conditions are maintained for the submachines. The characteristic polynomial of the submachine consisting of cell i through j is denoted by $\Gamma_{i,j}$.

It is now shown how a cyclic CA relates to its underlying null-boundary CA.

Lemma 3.27 *The characteristic polynomial of a cyclic CA satisfies the relationship*

$$\Gamma_{1,n} = \Delta_{1,n} + \Delta_{2,n-1}.$$

Proof. Let A be the characteristic matrix of a cyclic CA.

$$A = \begin{bmatrix} a_1 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 1 & a_2 & 1 & \ddots & & & 0 \\ 0 & 1 & a_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & a_{n-2} & 1 & 0 \\ 0 & & & \ddots & 1 & a_{n-1} & 1 \\ 1 & 0 & \cdots & \cdots & 0 & 1 & a_n \end{bmatrix},$$

where a_i denotes $x + d_i$. By expanding the determinant along the first row,

$$\Gamma_{1,n} = a_1\Delta_{2,n} + |B| + |D|,$$

where

$$B = \begin{bmatrix} 1 & 1 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & a_3 & 1 & \ddots & & & 0 \\ 0 & 1 & a_4 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & a_{n-2} & 1 & 0 \\ 0 & & & \ddots & 1 & a_{n-1} & 1 \\ 1 & 0 & \cdots & \cdots & 0 & 1 & a_n \end{bmatrix}$$

and

$$D = \begin{bmatrix} 1 & a_2 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & a_3 & 1 & \ddots & & \vdots \\ \vdots & \ddots & 1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 1 & 0 \\ \vdots & & & \ddots & \ddots & a_{n-2} & 1 \\ 0 & & & & \ddots & 1 & a_{n-1} \\ 1 & 0 & \cdots & \cdots & \cdots & 0 & 1 \end{bmatrix}.$$

Now, by expanding B along the first column,

$$|B| = \Delta_{3,n} + |E|$$

where

$$E = \begin{bmatrix} 1 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ a_3 & 1 & \ddots & & & & 0 \\ 1 & a_4 & \ddots & \ddots & & & \vdots \\ 0 & 1 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & a_{n-3} & 1 & \ddots & \vdots \\ \vdots & & \ddots & 1 & a_{n-2} & 1 & 0 \\ 0 & \cdots & \cdots & 0 & 1 & a_{n-1} & 1 \end{bmatrix}.$$

By expanding D along the last row, $|D| = \Delta_{2,n-1} + |F|$, where

$$F = \begin{bmatrix} 1 & 1 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & a_3 & 1 & \ddots & & \vdots \\ \vdots & \ddots & 1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 1 & 0 \\ \vdots & & & \ddots & \ddots & a_{n-3} & 1 \\ \vdots & & & & \ddots & 1 & a_{n-2} \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & 1 \end{bmatrix}.$$

Since E and F are lower and upper triangular, respectively, with all 1s on the diagonal, $|E| = |F| = 1$. Substituting these into the expressions for B and D,

$$\begin{aligned} |B| &= \Delta_{3,n} + 1 \\ |D| &= \Delta_{2,n-1} + 1 \end{aligned}$$

and hence, by (3.2),

$$\begin{aligned} \Gamma_{1,n} &= a_1\Delta_{2,n} + |B| + |D| \\ &= (x + d_1)\Delta_{2,n} + \Delta_{3,n} + \Delta_{2,n-1} \\ &= \Delta_{1,n} + \Delta_{2,n-1}. \end{aligned}$$

□

Table 3.1: Degree 3 Cyclic CA and polynomials

w_0	w_1	CA	polynomial
even	odd	001	$(x+1)^3$
		111	$x^2(x+1)$
odd	even	000	$(x+1)^2x$
		011	x^3

Appendix B lists all cyclic CA for n up to seven, with their corresponding characteristic polynomials. By inspecting these tables, it is natural to conjecture that the characteristic polynomial of a cyclic CA is always reducible. Before making the formal statement of this, some definitions are required. Let d denote the diagonal of A_{cyc}. Let w_0 be the parity of the number of 0s in d, and let w_1 be the parity of the number of 1s in d. That is, w_0 is 0 if d contains an even number of 0s, and 1 otherwise. Similarly, w_1 is 0 if d contains an even number of 1s, and 1 otherwise.

The Cyclic CA Theorem, which demonstrates the reducibility of cyclic CA characteristic polynomials, is a crucial result in the derivation of the CA synthesis algorithm (Chapter 6). The theorem also goes beyond showing that the characteristic polynomial is reducible; it states that the polynomial contains a large perfect square.

Theorem 3.28 (Cyclic CA Theorem) *Let A_{cyc} be the transition matrix of an n-cell cyclic CA (with respect to some fixed labeling), and let w_0 and w_1 be as defined above. Then there exists a polynomial q satisfying*

$$\Gamma_{1,n} = \begin{cases} q^2 & \text{if } w_0 = 0 \text{ and } w_1 = 0 \text{ (n even)} \\ (x+1)\,q^2 & \text{if } w_0 = 0 \text{ and } w_1 = 1 \text{ (n odd)} \\ (x)\,q^2 & \text{if } w_0 = 1 \text{ and } w_1 = 0 \text{ (n odd)} \\ (x)(x+1)\,q^2 & \text{if } w_0 = 1 \text{ and } w_1 = 1 \text{ (n even)} . \end{cases}$$

Proof. The proof is by induction on n. The base cases are for $n = 3$ and $n = 4$, and are provided by Table 3.1 and Table 3.2. The inductive step consists of ten cases.

The first eight cases apply when $d_1 = d_n$ for some cyclic shift of d. When this is true, the following recurrence holds, and is derived by repeatedly using various forms

Table 3.2: Degree 4 Cyclic CA and polynomials

w_0	w_1	CA	polynomial
even	even	0000	x^4
		0011	$(x^2+x+1)^2$
		0101	$(x+1)^2x^2$
		1111	$(x+1)^4$
odd	odd	0001	$x^3(x+1)$
		0111	$(x+1)^3x$

of the CA recurrence.

$$\begin{aligned}
\Gamma_{1,n} &= \Delta_{1,n} + \Delta_{2,n-1} \\
&= (x+d_1)\Delta_{2,n} + \Delta_{3,n} + \Delta_{2,n-1} \\
&= (x+d_1)\Delta_{2,n} + (x+d_n)\Delta_{3,n-1} + \Delta_{3,n-2} + \Delta_{2,n-1} \\
&= (x+d_1)(\Delta_{2,n} + \Delta_{3,n-1}) + (\Delta_{2,n-1} + \Delta_{3,n-2}) \\
&= (x+d_1)\Gamma_{2,n} + \Gamma_{2,n-1}.
\end{aligned}$$

So in summary, if $d_1 = d_n$, then

$$\Gamma_{1,n} = (x+d_1)\Gamma_{2,n} + \Gamma_{2,n-1}. \tag{3.8}$$

These eight cases correspond to the eight possible values for d_1, w_0, and w_1. Let y_0, y_1 denote the parity of the number of 0s and 1s, respectively, on the main diagonal of $C_{2,n}$. Let z_0, z_1 denote the parity of the number of 0s and 1s, respectively, on the main diagonal of $C_{2,n-1}$.

The values of d_1, w_0, and w_1 uniquely determine the values of y_0, y_1, z_0, and z_1. Then, since $\Gamma_{2,n}$ and $\Gamma_{2,n-1}$ have degree less than n, the inductive hypothesis is applied to show that $\Gamma_{1,n}$ has the correct form.

For example, suppose that $d_1 = w_0 = w_1 = 0$. Then the vector $[d_2, d_3, \ldots, d_n]$ has an odd number of 0s and an even number of 1s, and so $y_0 = 1$ and $y_1 = 0$. By induction, $\Gamma_{2,n} = xq_y^2$ for some $q_y \in \mathrm{GF}(2)[x]$.

The vector $[d_2, d_3, \ldots, d_{n-1}]$ has an even number of 0s and an even number of 1s, implying that $z_0 = 0$ and $z_1 = 0$. Again by induction, $\Gamma_{2,n-1} = q_z^2$ for some q_z.

Combining $\Gamma_{2,n}$ and $\Gamma_{2,n-1}$ with (3.8),

$$\begin{aligned}\Gamma_{1,n} &= (x + d_1)\Gamma_{2,n} + \Gamma_{2,n-1} \\ &= x(xq_y^2) + q_z^2 \\ &= (xq_y + q_z)^2 \\ &= q^2,\end{aligned}$$

with $q = xq_y + q_z$. Hence $\Gamma_{1,n}$ is a perfect square, as is claimed in the theorem.

The eight cases are shown in the following table.

case	d_1	w_0	w_1	y_0	y_1	$\Gamma_{2,n}$	z_0	z_1	$\Gamma_{2,n-1}$	$\Gamma_{1,n}$
1	0	0	0	1	0	$(x)p^2$	0	0	q^2	$(x)(x)p^2 + q^2$ $= (xp + q)^2$
2	1	0	0	0	1	$(x+1)p^2$	0	0	q^2	$(x+1)(x+1)p^2 + q^2$ $= ((x+1)p + q)^2$
3	0	0	1	1	1	$(x)(x+1)p^2$	0	1	$(x+1)q^2$	$(x)(x)(x+1)p^2 + (x+1)q^2$ $= (x+1)(xp + q)^2$
4	1	0	1	0	0	p^2	0	1	$(x+1)q^2$	$(x+1)p^2 + (x+1)q^2$ $= (x+1)(p + q)^2$
5	0	1	0	0	0	p^2	1	0	$(x)q^2$	$(x)p^2 + (x)q^2$ $= x(p + q)^2$
6	1	1	0	1	1	$(x)(x+1)p^2$	1	0	$(x)q^2$	$(x+1)(x)(x+1)p^2 + (x)q^2$ $= x((x+1)p + q)^2$
7	0	1	1	0	1	$(x+1)p^2$	1	1	$(x)(x+1)q^2$	$(x)(x+1)p^2 + (x)(x+1)q^2$ $= (x)(x+1)(p + q)^2$
8	1	1	1	1	0	$(x)p^2$	1	1	$(x)(x+1)q^2$	$(x+1)(x)p^2 + (x)(x+1)q^2$ $= (x)(x+1)(p + q)^2$

The two remaining cases apply when $d_1 \neq d_n$ for any cyclic shift of the vector d.

This means that d must be of the form

$$d = [0, 1, 0, 1, \ldots, 0, 1, 0, 1].$$

It is assumed, without loss of generality, that $d_1 = 0$. Note that such a vector must have even length.

The following derivations hold for $n \geq 4$. Using the relation between the cyclic CA and null-boundary CA characteristic polynomials (Lemma 3.27) and the CA recurrence (version (3.2)),

$$\begin{aligned} \Gamma_{1,n} &= \Delta_{1,n} + \Delta_{2,n-1} \\ &= x\Delta_{2,n} + \Delta_{3,n} + \Delta_{2,n-1}. \end{aligned}$$

Because of the regular pattern in d, $\Delta_{3,n} = \Delta_{2,n-1}$, and so

$$\Gamma_{1,n} = x\Delta_{2,n}. \tag{3.9}$$

Again exploiting the structure of d, with two more applications of the CA recurrence,

$$\begin{aligned} \Delta_{2,n} &= (x+1)\Delta_{3,n} + \Delta_{4,n} \\ &= (x+1)(x\Delta_{4,n} + \Delta_{5,n}) + (x+1)\Delta_{5,n} + \Delta_{6,n} \\ &= x(x+1)\Delta_{4,n} + \Delta_{6,n}. \end{aligned}$$

Substituting this into (3.9) twice,

$$\begin{aligned} \Gamma_{1,n} &= x\Delta_{2,n} \\ &= x(x(x+1)\Delta_{4,n} + \Delta_{6,n}) \\ &= (x(x+1))x\Delta_{4,n} + x\Delta_{6,n} \\ &= x(x+1)\Gamma_{3,n} + \Gamma_{5,n}. \end{aligned} \tag{3.10}$$

Case 9 of the proof is for w_0 even, w_1 even, and $d = [0, 1, 0, 1, \ldots, 0, 1]$. Then $C_{3,n}$ has an odd number of 0s and an odd number of 1s, and thus by induction

$\Gamma_{3,n} = (x)(x+1)q_y^2$ for some q_y. Further, $C_{5,n}$ has an even number of 0s and an even number of 1s, and so $\Gamma_{5,n} = q_z^2$ for some q_z. Then by (3.10),

$$\begin{aligned}\Gamma_{1,n} &= x(x+1)\Gamma_{3,n} + \Gamma_{5,n} \\ &= x(x+1)(x)(x+1)q_y^2 + q_z^2 \\ &= x^2(x+1)^2q_y^2 + q_z^2 \\ &= (x(x+1)q_y + q_z)^2,\end{aligned}$$

satisfying the inductive hypothesis.

For case 10, w_0 is odd, w_1 is odd, and $d = [0,1,0,1,\ldots,0,1]$. Then $C_{3,n}$ has an even number of 0s and an even number of 1s, and $\Gamma_{3,n} = q_y^2$ for some q_y. Also, $C_{5,n}$ has an odd number of 0s and an odd number of 1s, and so $\Gamma_{5,n} = x(x+1)q_z^2$ for some q_z. By (3.10),

$$\begin{aligned}\Gamma_{1,n} &= x(x+1)\Gamma_{3,n} + \Gamma_{5,n} \\ &= x(x+1)q_y^2 + x(x+1)q_z^2 \\ &= x^2(x+1)^2q_y^2 + q_z^2 \\ &= x(x+1)(q_y + q_z)^2,\end{aligned}$$

satisfying the inductive hypothesis.

In conclusion, the inductive hypothesis is satisfied for all diagonals of length n. □

3.7 Conclusion

This chapter explores the fundamentals of characteristic polynomials of CA over GF(2). Section 3.1 presents the (previously known) CA recurrence, and section 3.2 shows how it is used to efficiently calculate the characteristic polynomial of a null-boundary CA. In section 3.3, two useful relations satisfied by the characteristic polynomials of the submachines of a CA are derived. Section 3.4 justifies the study of characteristic polynomials, by showing that the characteristic polynomial and minimal polynomial of a CA are often the same. This is important for the similarity

transform between a CA and an LFSR in section 3.5. Section 3.6 defines cyclic CA, and relates the characteristic polynomial of such a machine to the characteristic polynomial of its underlying null-boundary CA. The characteristic polynomial of a cyclic CA is always reducible.

Many of the results of this chapter are used for the analysis of special forms of CA (Chapter 5), and for the synthesis algorithm in Chapter 6. Some of the results generalise to CA over an arbitrary finite field GF(q), which is the subject of the next chapter.

Chapter 4

Fundamentals of CA Characteristic Polynomials - GF(q)

This chapter generalises some of the results of Chapter 3 to CA over the Galois field GF(q). As well as being interesting on their own, these generalisations are required for the classification of partially-connected CA over GF(2) and GF(q) in section 5.1.

Section 4.1 contains background material for CA over GF(q). The generalisation of the CA recurrence and its application to computing CA characteristic polynomials is presented in section 4.2 and section 4.3. Generalisations of the concatenation relation and the GCD relation are contained in section 4.4. Section 4.5 shows that a subclass of CA are non-derogatory, and demonstrates a similarity transform between CA and LFSRs over GF(q). In section 4.6, cyclic-boundary CA over GF(q) are defined and related to the standard null-boundary CA.

4.1 Background

A linear hybrid null-boundary cellular automaton over GF(q) (abbreviated to *CA* in this chapter) is slightly more complicated than a CA over GF(2). Each cell is connected to its left and right neighbours, with the two end cells having constant zero inputs from their missing neighbours. A cell consists of a memory device capable of holding an element of GF(q), and a next-state computation function. This function, or *rule*, is specified by three constant multipliers from GF(q); one as a self-multiplier,

Figure 4.1: Interconnection structure and multipliers of a CA

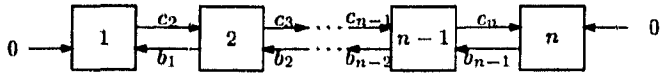
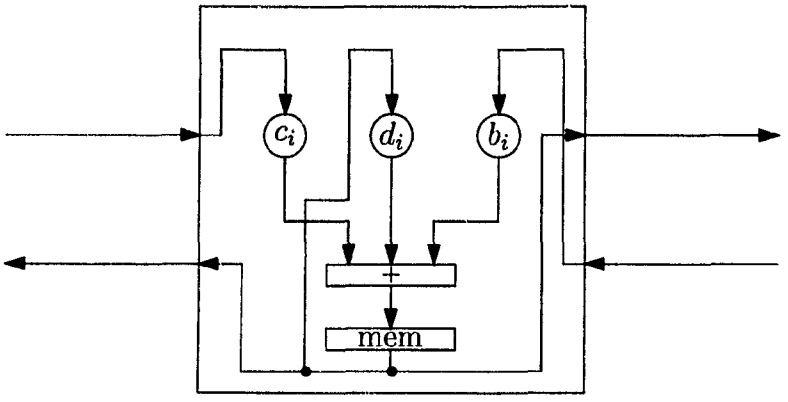


Figure 4.2: Internal structure of a CA cell



and one for each of the left and right inputs. These input multipliers are not present on GF(2) machines. The interconnection structure and multipliers of a CA are shown in Figure 4.1.

A CA over GF(q) computes the state of cell i at time $t+1$ by the function

$$s_i^{t+1} = c_i s_{i-1}^t + d_i s_i^t + b_i s_{i+1}^t, \tag{4.1}$$

where c_i is the *left input multiplier*, b_i is the *right input multiplier*, and d_i is the *self-multiplier*. The multiplication and addition are the usual finite field operations in GF(q). The internal structure of a cell is shown in Figure 4.2.

Figure 4.3: A 3-cell CA over GF(3)

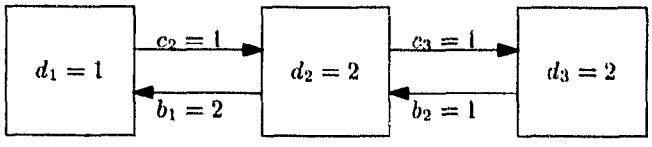


Table 4.1: State space of 3-cell CA

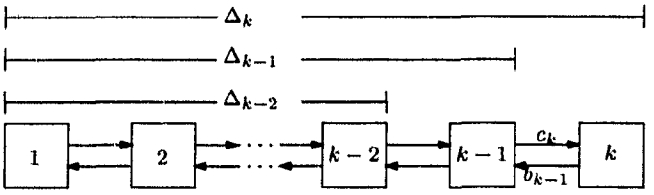
time	*state*	time	*state*
0	[1, 0, 0]	13	[2, 0, 0]
1	[1, 1, 0]	14	[2, 2, 0]
2	[0, 0, 1]	15	[0, 0, 2]
3	[0, 1, 2]	16	[0, 2, 1]
4	[2, 1, 2]	17	[1, 2, 1]
5	[1, 0, 2]	18	[2, 0, 1]
6	[1, 0, 1]	19	[2, 0, 2]
7	[1, 2, 2]	20	[2, 1, 1]
8	[2, 1, 0]	21	[1, 2, 0]
9	[1, 1, 1]	22	[2, 2, 2]
10	[0, 1, 0]	23	[0, 2, 0]
11	[2, 2, 1]	24	[1, 1, 2]
12	[0, 1, 1]	25	[0, 2, 2]

Equation (4.1) implies that the transition matrix of a CA has the following form:

$$A = \begin{bmatrix} d_1 & b_1 & 0 & \cdots & \cdots & 0 & 0 \\ c_2 & d_2 & b_2 & \ddots & & & 0 \\ 0 & c_3 & d_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & d_{n-2} & b_{n-2} & 0 \\ 0 & & & \ddots & c_{n-1} & d_{n-1} & b_{n-1} \\ 0 & 0 & \cdots & \cdots & 0 & c_n & d_n \end{bmatrix}. \tag{4.2}$$

Example 4.1. Consider the 3-cell CA M in Figure 4.3. The transition matrix for

Figure 4.4: Submachines related by the CA recurrence



this machine is

$$A = \begin{bmatrix} 1 & 2 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}.$$

Table 4.1 shows the non-zero state space of M. As can be seen in the table, all 26 non-zero states ($q^n - 1 = 3^3 - 1 = 26$) appear in a single cycle, demonstrating that M has a maximal length cycle. □

The following sections contain generalisations of the results of Chapter 3 to CA over GF(q). Appendix E contains counts of the number of CA for various restricted types of CA and various fields. Appendix F lists least-cost maximal-length CA for various prime fields, for one of the restricted types.

4.2 CA recurrence

This section contains the generalisation of the CA recurrence (Theorem 3.1) to GF(q). As in GF(2), the recurrence provides an efficient means of calculating the characteristic polynomial of a CA.

The recurrence shows that the polynomial $\Delta_{1,k}$ is a function of Δ_{k-1}, Δ_{k-2}, and the information pertaining to the addition of cell k. This information consists of the self-multiplier d_k for cell k, and the multipliers on the connections joining cell k to the rest of the CA, b_{k-1} and c_k. The submachines involved are shown in Figure 4.4.

Theorem 4.2 (CA recurrence over GF(q)) *Let M be an n-cell CA with transition matrix (4.2). Then the characteristic polynomials of the submachines of M satisfy*

the following recurrence:

$$\begin{aligned}\Delta_{-1} &= 0\\ \Delta_0 &= 1\\ \Delta_k &= (x-d_k)\Delta_{k-1} - b_{k-1}c_k\Delta_{k-2},\ \ 1 \le k \le n. \end{aligned} \tag{4.3}$$

Proof. The cases for $k = 1$ and $k = 2$ are handled separately, as they depend on the definitions of Δ_{-1} and Δ_0. The general case is then proven directly.

For $k = 1$, (4.3) gives

$$\begin{aligned}\Delta_1 &= (x-d_1) - b_0c_1\Delta_{-1}\\ &= (x-d_1) - b_0c_1 \cdot 0\\ &= (x-d_1).\end{aligned}$$

As the single cell machine M_1 has transition matrix $[d_1]$, the characteristic polynomial Δ_1 is $x-d_1$, (4.3). Note that as $\Delta_{-1} = 0$, it is not necessary for b_0 or c_1 to be defined.

For $k = 2$, (4.3) states that

$$\begin{aligned}\Delta_2 &= (x-d_2)\Delta_1 - b_1c_2\Delta_0\\ &= (x-d_2)(x-d_1) - b_1c_2.\end{aligned}$$

The transition matrix of the 2-cell machine M_2 is

$$\begin{bmatrix} d_1 & b_1 \\ c_2 & d_2 \end{bmatrix},$$

which has characteristic polynomial $(x-d_1)(x-d_2) - b_1c_2$, verifying (4.3) for $k = 2$.

For the general case, $3 \le k \le n$, let A be the characteristic matrix of the machine M_k (with a_i denoting $x - d_i$):

$$A = \begin{bmatrix} a_1 & b_1 & 0 & \cdots & \cdots & 0 & 0 \\ c_2 & a_2 & b_2 & \ddots & & & 0 \\ 0 & c_3 & a_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & a_{k-2} & b_{k-2} & 0 \\ 0 & & & \ddots & c_{k-1} & a_{k-1} & b_{k-1} \\ 0 & 0 & \cdots & \cdots & 0 & c_k & a_k \end{bmatrix}.$$

By expanding A along the last row,

$$\begin{aligned}\Delta_k &= \det(A) \\ &= (-1)^{k+1}(-1)^{k+1}a_k \det(B) + (-1)^{k+1}(-1)^k c_k \det(C) \\ &= a_k \det(B) - c_k \det(C)\end{aligned}$$

where

$$B = \begin{bmatrix} a_1 & b_1 & 0 & \cdots & \cdots & 0 & 0 \\ c_2 & a_2 & b_2 & \ddots & & & 0 \\ 0 & c_3 & a_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & a_{k-3} & b_{k-3} & 0 \\ 0 & & & \ddots & c_{k-2} & a_{k-2} & b_{k-2} \\ 0 & 0 & \cdots & \cdots & 0 & c_{k-1} & a_{k-1} \end{bmatrix},$$

and

$$C = \begin{bmatrix} a_1 & b_1 & 0 & \cdots & \cdots & 0 & 0 \\ c_2 & a_2 & b_2 & \ddots & & & 0 \\ 0 & c_3 & a_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & a_{k-3} & b_{k-3} & 0 \\ 0 & & & \ddots & c_{k-2} & a_{k-2} & 0 \\ 0 & 0 & \cdots & \cdots & 0 & c_{k-1} & b_{k-1} \end{bmatrix}.$$

Now, B is the characteristic matrix of the CA with the last cell removed, and so by the inductive hypothesis,

$$\det(B) = \Delta_{k-1}.$$

Similarly, the matrix C expanded along the last column gives

$$\det(C) = (-1)^k(-1)^k b_{k-1}\Delta_{k-2}.$$

Combining these,

$$\det(A) = a_k\Delta_{k-1} - c_k b_{k-1}\Delta_{k-2}.$$

□

Interestingly, the characteristic polynomial is invariant under changes to the b_i and c_i, so long as each product $b_{i-1}c_i$ remains unchanged. This observation is important for the derivation of the similarity transform in section 4.5.

As in section 3.1, the left-right symmetry provides the following generalisation of (4.3). This is equivalent to reversing the labeling on the CA, and then applying Theorem 4.2. The relation is

$$\Delta_{k,n} = (x - d_k)\Delta_{k+1,n} - b_k c_{k+1}\Delta_{k+2,n}, \quad 1 \leq k \leq n, \tag{4.4}$$

with the base cases defined as $\Delta_{n+1,n} = 1$ and $\Delta_{n+2,n} = 0$. In a more general form, (4.3) can be applied to a submachine of a CA, giving

$$\Delta_{i,k} = (x - d_k)\Delta_{i,k-1} - b_k c_{k+1}\Delta_{i,k-2}, \quad k \geq i, \tag{4.5}$$

and, by reversal,

$$\Delta_{k,i} = (x - d_k)\Delta_{k+1,i} - b_k c_{k+1}\Delta_{k+2,i}, \quad k \leq i. \tag{4.6}$$

For both of these relations, the bases cases are $\Delta_{i,i-1} = 1$ and $\Delta_{i,i-2} = 0$.

4.3 Computation of CA characteristic polynomials

Repeated application of Theorem 4.2 gives an efficient method of computing the characteristic polynomial of a CA. Initially, Δ_{-1} and Δ_0 are assigned 0 and 1, respectively, and (4.3) is applied to obtain Δ_1. Reapplying (4.3) to Δ_0 and Δ_1 results in Δ_2. Continuing this process, the polynomials $\Delta_3, \Delta_4, \ldots, \Delta_{n-1}, \Delta_n = \Delta$ are obtained.

Each application of (4.3) involves a polynomial multiplication, with one of the multiplicands monic and degree 1, and a polynomial addition. Hence, similar to the GF(2) case, the number of operations required to compute the characteristic polynomial is $2n$ polynomial additions, n scalar-polynomial multiplications, and n polynomial shifts.

The following examples illustrate the computation of the characteristic polynomial of a CA.

Example 4.3. Consider the 3-cell CA over GF(3) from Example 4.1. The transition matrix for this machine is

$$A = \begin{bmatrix} 1 & 2 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

The computation of the characteristic polynomial is performed in GF(3)[x], and proceeds as follows.

$$\begin{aligned} \Delta_{-1} &= 0 \\ \Delta_0 &= 1 \\ \Delta_1 &= x - 1 \\ \Delta_2 &= (x-2)(x-1) - 2 \cdot 1 \cdot 1 = x^2 \\ \Delta_3 &= (x-2)(x^2) - 1 \cdot 1 \cdot (x-1) \\ &= x^3 + x^2 + 2x + 1, \end{aligned}$$

so $\Delta = \Delta_3 = x^3 + x^2 + 2x + 1$. □

Example 4.4. This example is for a 3-cell CA over GF(4). The elements of the field are represented as polynomials in y, and arithmetic is performed modulo the polynomial $y^2 + y + 1$, irreducible over GF(2). Note that $-\alpha = \alpha$ and $2\alpha = 0$ for all α. Consider the transition matrix

$$A = \begin{bmatrix} y+1 & 1 & 0 \\ 1 & 0 & y \\ 0 & y & 1 \end{bmatrix}.$$

The computation of the characteristic polynomial is

$$\begin{aligned} \Delta_{-1} &= 0 \\ \Delta_0 &= 1 \\ \Delta_1 &= x + (y+1) \\ \Delta_2 &= (x)(x + (y+1)) + 1 \cdot 1 \cdot 1 = x^2 + x(y+1) + 1 \\ \Delta_3 &= (x+1)(x^2 + x(y+1) + 1) + y \cdot y \cdot (x + (y+1)) \\ &= (x+1)(x^2 + x(y+1) + 1) + x(y+1) + y \\ &= x^3 + (1 + (y+1))x^2 + (1 + (y+1) + (y+1))x + (y+1) \\ &= x^3 + yx^2 + x + (y+1) \end{aligned}$$

so $\Delta = \Delta_3 = x^3 + yx^2 + x + (y+1)$. This polynomial is primitive over GF(4). □

The characteristic polynomial written as an explicit function of the variables $a_i = x + d_i$, b_i, and c_i is shown below. As in the GF(2) case, such formulas quickly become

cumbersome.

$$\begin{aligned}\Delta_{-1} &= 0\\ \Delta_0 &= 1\\ \Delta_1 &= a_1\\ \Delta_2 &= a_2a_1 - b_1c_2\\ \Delta_3 &= a_1a_2a_3 - a_3b_1c_2 - a_1b_2c_3.\end{aligned}$$

If the a_i are expanded to $x + d_i$, the closed formulas for the characteristic polynomial become

$$\begin{aligned}\Delta_{-1} &= 0\\ \Delta_0 &= 1\\ \Delta_1 &= (x - d_1)\\ \Delta_2 &= (x - d_2)(x - d_1) - b_1c_2\\ &= x^2 - (d_1 + d_2)x + d_1d_2 - b_1c_2\\ \Delta_3 &= (x - d_3)(x - d_2)(x - d_1) - (x - d_3)b_1c_2 - (x - d_1)b_2c_3.\\ &= x^3 - (d_1 + d_2 + d_3)x^2 + (d_1d_2 + d_1d_3 + d_2d_3 - b_1c_2 - b_2c_3)x\\ &\quad -d_1d_2d_3 + d_3b_1c_2 + d_1b_2c_3.\end{aligned}$$

As in GF(2), these expressions would be difficult to work with directly.

4.4 Other relations

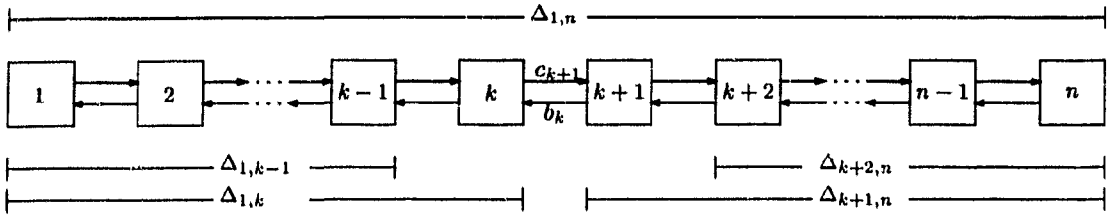
This section contains the generalisations to GF(q) of the *concatenation relation* and the *GCD relation*. The GF(2) analogues of these relations are in section 3.3. The GF(q) generalisation of the concatenation relation is used in the classification of partially-connected CA over GF(2) (section 5.1).

The concatenation relation describes the characteristic polynomial of a CA formed by concatenating two CA. In view of Theorem 3.4, it is not surprising that, in GF(q), the characteristic polynomial of the concatenated machine is a function of the characteristic polynomials of the two components and of the multipliers used to join the components (see Figure 4.5).

Theorem 4.5 *For any k with $0 \le k \le n$,*

$$\Delta_{1,n} = \Delta_{1,k}\Delta_{k+1,n} - b_kc_{k+1}\Delta_{1,k-1}\Delta_{k+2,n}.$$

Figure 4.5: Submachines related by the concatenation relation



Proof. The proof is by induction on k, with the base case being $k = 0$. Using the definitions of $\Delta_{i,i-2}$ and $\Delta_{i,i-1}$ as 0 and 1 respectively, the theorem claims that

$$\begin{aligned}\Delta_{1,n} &= \Delta_{1,0}\Delta_{1,n} - b_0c_1\Delta_{1,-1}\Delta_{2,n} \\ &= 1 \cdot \Delta_{1,n} - b_0c_1 \cdot 0 \cdot \Delta_{2,n} \\ &= \Delta_{1,n},\end{aligned}$$

and hence is satisfied trivially.

By assuming inductively that the theorem holds for k, it is shown below that is holds for $k+1$. By the inductive hypothesis,

$$\Delta_{1,n} = \Delta_{1,k}\Delta_{k+1,n} - b_kc_{k+1}\Delta_{1,k-1}\Delta_{k+2,n}.$$

Applying (4.4) to $\Delta_{k+1,n}$, this is equal to

$$\Delta_{1,n} = \Delta_{1,k}((x - d_{k+1})\Delta_{k+2,n} - b_{k+1}c_{k+2}\Delta_{k+3,n}) - b_kc_{k+1}\Delta_{1,k-1}\Delta_{k+2,n}.$$

Rearranging terms,

$$\Delta_{1,n} = -b_{k+1}c_{k+2}\Delta_{1,k}\Delta_{k+3,n} + (x - d_{k+1})\Delta_{1,k}\Delta_{k+2,n} - b_kc_{k+1}\Delta_{1,k-1}\Delta_{k+2,n},$$

and factoring out $\Delta_{k+2,n}$,

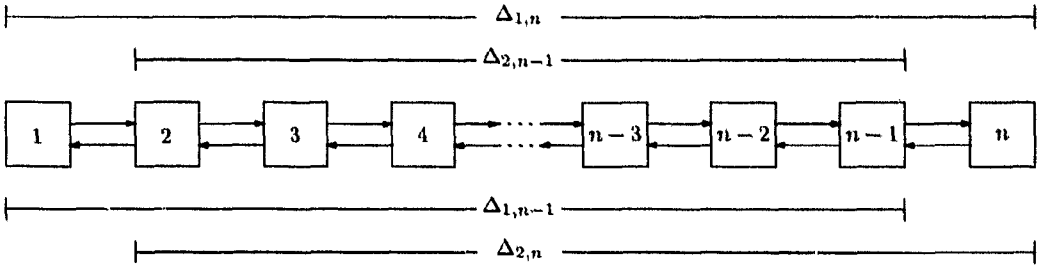
$$\Delta_{1,n} = ((x - d_{k+1})\Delta_{1,k} - b_kc_{k+1}\Delta_{1,k-1})\Delta_{k+2,n} - b_{k+1}c_{k+2}\Delta_{1,k}\Delta_{k+3,n}.$$

Applying (4.4),

$$\Delta_{1,n} = \Delta_{1,k+1}\Delta_{k+2,n} - b_{k+1}c_{k+2}\Delta_{1,k}\Delta_{k+3,n},$$

showing that the formula holds for $k+1$, given that it holds for k. □

Figure 4.6: Submachines related by the GCD relation



When k is set to $n-1$, Theorem 4.5 becomes

$$\begin{aligned}\Delta_{1,n} &= \Delta_{1,n-1}\Delta_{n,n} - b_{n-1}c_n\Delta_{1,n-2}\Delta_{n+1,n} \\ &= (x-d_n)\Delta_{1,n-1} - b_{n-1}c_n\Delta_{1,n-2}.\end{aligned}$$

This is the statement of Theorem 4.2, demonstrating that Theorem 4.5 subsumes Theorem 4.2.

Theorem 4.5 has an easy corollary, which provides a much simpler proof of a result in [16]. This corollary is important for the similarity transform derived in section 4.5.

Corollary 4.6 *If $b_i = 0$ for any $1 \leq i < n$ or $c_j = 0$ for any $1 < j \leq n$, then Δ is reducible.*

Proof. Suppose that $b_i = 0$ for some $1 \leq i < n$. From Theorem 4.2, it is clear that Δ factors as $\Delta_{1,n} = \Delta_{1,i}\Delta_{i+1,n}$, and both $\Delta_{1,i}$ and $\Delta_{i+1,n}$ have degree at least 1. The proof is analogous for $c_j = 0$. □

The GCD relation has no evident usefulness in GF(q), though the GF(2) formulation in Theorem 3.6 is crucial to the synthesis of CA over GF(2) (Chapter 6). It is included here for completeness. The submachines involved in the relation are shown in Figure 4.6.

Theorem 4.7 *For $n \geq 1$,*

$$\Delta_{1,n}\Delta_{2,n-1} - \Delta_{1,n-1}\Delta_{2,n} = -\prod_{i=1}^{n-1} b_i \prod_{i=2}^{n} c_i$$

Proof. The proof is by induction. The base case is for $n = 1$

$$\begin{aligned}
\Delta_{1,n}\Delta_{2,n-1} - \Delta_{1,n-1}\Delta_{2,n} &= \Delta_{1,1}\Delta_{2,0} - \Delta_{1,0}\Delta_{2,1} \\
&= (x - d_1) \cdot 0 - 1 \cdot 1 \\
&= -1 \\
&= -\textstyle\prod_{i=1}^{0} b_i \prod_{i=2}^{1} c_i.
\end{aligned}$$

By induction, assume that the theorem holds for $n - 1$. The following derivation shows that it holds for n.

$$\begin{aligned}
& \Delta_{1,n}\Delta_{2,n-1} - \Delta_{1,n-1}\Delta_{2,n} \\
= \; & \Delta_{2,n-1}((x - d_n)\Delta_{1,n-1} - b_{n-1}c_n\Delta_{1,n-2}) \\
& - \Delta_{1,n-1}((x - d_n)\Delta_{2,n-1} - b_{n-1}c_n\Delta_{2,n-2}) \\
= \; & -b_{n-1}c_n\Delta_{2,n-1}\Delta_{1,n-2} + (x - d_n)\Delta_{2,n-1}\Delta_{1,n-2} \\
& + b_{n-1}c_n\Delta_{1,n-1}\Delta_{2,n-2} - (x - d_n)\Delta_{2,n-1}\Delta_{1,n-1} \\
= \; & -b_{n-1}c_n\Delta_{2,n-1}\Delta_{1,n-2} + b_{n-1}c_n\Delta_{1,n-1}\Delta_{2,n-2} \\
= \; & b_{n-1}c_n(\Delta_{1,n-1}\Delta_{2,n-2} - \Delta_{2,n-1}\Delta_{1,n-2}) \\
= \; & b_{n-1}c_n(\prod_{i=1}^{n-2} b_i \prod_{i=2}^{n-1} c_i) \\
= \; & \prod_{i=1}^{n-1} b_i \prod_{i=2}^{n} c_i.
\end{aligned}$$

□

4.5 Similarity transforms between CA and LFSR matrices

In this section, a similarity transform from an irreducible CA to its corresponding LFSR is demonstrated. This transform is obtained as the composition of three simple

transforms. The first transform is from a CA to a special form of CA. The second is from this special form to the diagonal matrix, and the third is from the diagonal matrix to an LFSR. A general condition is presented, that does not require irreducibility of the CA, that shows when a similarity transform exists.

A generalisation to GF(q) of the non-derogatory result of section 3.4 is required, to determine conditions under which a similarity transform exists. The conditions are closely related to those for CA over GF(2).

Theorem 4.8 *A matrix A is non-derogatory if it is upper Hessenburg and all of the subdiagonal entries are non-zero.*

Proof. The proof is a simple generalisation of the proof of the GF(2) version of the theorem (Theorem 3.15). The non-zero entries on the subdiagonal guarantee that the kth subdiagonal of A^k is all nonzero. □

For the derivation of the similarity transform from a CA to an LFSR, a more restrictive assumption than A being upper Hessenburg is used.

For the remainder of this section, only CA with irreducible characteristic polynomials are considered. Recall that two matrices with the same irreducible characteristic polynomial are similar. Also, to ease the presentation, n is fixed at 4, and so short derivations are presented instead of full proofs. The general proofs are easily deduced from the material presented.

Consider a 4-cell CA with irreducible characteristic polynomial Δ, and suppose that the transition matrix for this CA is

$$A = \begin{bmatrix} d_1 & b_1 & & \\ c_2 & d_2 & b_2 & \\ & c_3 & d_3 & b_3 \\ & & c_4 & d_4 \end{bmatrix}.$$

By the converse of Corollary 4.6, this CA has no zero entries on the subdiagonal or superdiagonal.

The first of the three transforms provides a CA in which the upper diagonal is all 1. It utilises the property that the characteristic polynomial is invariant under changes to the b_i and c_i, so long as each product $b_{i-1}c_i$ is held fixed (see section 4.2). Consequently, the matrix

$$A' = \begin{bmatrix} d_1 & 1 & & \\ b_1c_2 & d_2 & 1 & \\ & b_2c_3 & d_3 & 1 \\ & & b_3c_4 & d_4 \end{bmatrix} \tag{4.7}$$

has the same characteristic polynomial as A, and hence there exists a similarity transform from A to A' (both are nonderogatory by Theorem 4.8). Such a similarity transform is given by

$$P_1 A P_1^{-1} = A',$$

with

$$P_1 = \begin{bmatrix} 1 & & & \\ & b_1 & & \\ & & b_1b_2 & \\ & & & b_1b_2b_3 \end{bmatrix}.$$

P_1 exists because of the observation that $b_i \neq 0$ for all i, and is clearly non-singular. To see that transform behaves as is claimed, consider the products P_1A and $A'P_1$:

$$P_1A = \begin{bmatrix} d_1 & b_1 & & \\ b_1c_2 & b_1d_2 & b_1b_2 & \\ & b_1b_2c_3 & b_1b_2d_3 & b_1b_2b_3 \\ & & b_1b_2b_3c_4 & b_1b_2b_3d_4 \end{bmatrix}$$

$$A'P_1 = \begin{bmatrix} d_1 & b_1 & & \\ b_1c_2 & b_1d_2 & b_1b_2 & \\ & b_1b_2c_3 & b_1b_2d_3 & b_1b_2b_3 \\ & & b_1b_2b_3c_4 & b_1b_2b_3d_4 \end{bmatrix}.$$

The second transform is between a matrix of the form

$$\begin{bmatrix} d_1 & 1 & & \\ b_1 & d_2 & 1 & \\ & b_2 & d_3 & 1 \\ & & b_3 & d_4 \end{bmatrix}$$

(from (4.7)) and the diagonal form. The irreducibility of Δ ensures that the diagonal form exists, and that the non-zero entries are the roots in GF(q^n) of Δ. The diagonal

form is given by

$$D = \begin{bmatrix} \alpha & & & \\ & \alpha^q & & \\ & & \alpha^{q^2} & \\ & & & \alpha^{q^3} \end{bmatrix}.$$

The transform between (4.7) and diagonal form is

$$P_2 D P_2^{-1} = A',$$

with

$$P_2 = \begin{bmatrix} m_0 & m_0^q & m_0^{q^2} & m_0^{q^3} \\ m_1 & m_1^q & m_1^{q^2} & m_1^{q^3} \\ m_2 & m_2^q & m_2^{q^2} & m_2^{q^3} \\ m_3 & m_3^q & m_3^{q^2} & m_3^{q^3} \end{bmatrix}.$$

The matrix P_2 can be shown to be non-singular by applying Theorem 2.17. It is straightforward to see that $\{1, m_1, m_2, \ldots, m_{n-1}\}$ forms a basis of GF(q^n) over GF(q) To see that P_2 is the required transform, consider the elements $(P_2 D)_{i,j}$ and $(A' P_2)_{i,j}$. First note that

$$\begin{aligned} & m_i = (\alpha - d_i) m_{i-1} - b_{i-1} c_i m_{i-2} \\ \Rightarrow \quad & \alpha m_{i-1} = b_{i-1} c_i m_{i-2} + d_i m_{i-1} + m_i. \end{aligned}$$

for $i = 1, \ldots, n$. The $i = 1$ and $i = n$ cases hold because m_{-1} and m_n are both zero.

$$\begin{aligned} (P_2 D)_{i,j} &= m_{i-1}^{q^{j-1}} \alpha^{q^{j-1}} \\ &= (\alpha m_{i-1})^{q^{j-1}}. \end{aligned}$$

For $i = 1, \ldots, n$,

$$\begin{aligned} (A' P_2)_{i,j} &= b_{i-1} c_i m_{i-2}^{q^{j-1}} + d_i m_{i-1}^{q^{j-1}} + m_i^{q^{j-1}} \\ &= (b_{i-1} c_i m_{i-2} + d_i m_{i-1} + m_i)^{q^{j-1}} \\ &= (\alpha m_{i-1})^{q^{j-1}}. \end{aligned}$$

The third transform is between the diagonal matrix D and the companion form C. The companion form (transition matrix of a 4-cell LFSR) is

$$C = \begin{bmatrix} 0 & 0 & 0 & -p_0 \\ 1 & 0 & 0 & -p_1 \\ 0 & 1 & 0 & -p_2 \\ 0 & 0 & 1 & -p_3 \end{bmatrix},$$

where the p_i are the coefficients of Δ. The similarity transform is

$$P_3 C P_3^{-1} = D,$$

with

$$P_3 = \begin{bmatrix} \alpha & \alpha^2 & \alpha^3 & \alpha^4 \\ \alpha^q & (\alpha^2)^q & (\alpha^3)^q & (\alpha^4)^q \\ \alpha^{q^2} & (\alpha^2)^{q^2} & (\alpha^3)^{q^2} & (\alpha^4)^{q^2} \\ \alpha^{q^3} & (\alpha^2)^{q^3} & (\alpha^3)^{q^3} & (\alpha^4)^{q^3} \end{bmatrix}.$$

Again, P_3 is non-singular by application of Theorem 2.17 (as in the GF(2) case, the irreducibility of Δ ensures that $\{\alpha, \alpha^2, \ldots, \alpha^n\}$ is a basis for GF(q^n) over GF(q)). Consider the element (i, j) of DP_3:

$$\begin{aligned} (DP_3)_{i,j} &= [0, \ldots, 0, \alpha^{q^{i-1}}, 0, \ldots, 0] \cdot [\alpha^j, (\alpha^j)^q, \ldots, (\alpha^j)^{q^{i-1}}, \ldots, (\alpha^j)^{q^{n-1}}]^{\mathrm{T}} \\ &= \alpha^{q^{i-1}} (\alpha^j)^{q^{i-1}} \\ &= (\alpha^{(j+1)})^{q^{i-1}} \\ &= \alpha^{(j+1)q^{i-1}}. \end{aligned}$$

Now, for $j < n$,

$$\begin{aligned} (P_3 C)_{i,j} &= [\alpha^{q^{i-1}}, (\alpha^2)^{q^{i-1}}, \ldots, (\alpha^n)^{q^{i-1}}] \cdot \overbrace{[0, \ldots, 0, 1, 0, \ldots, 0]^{\mathrm{T}}}^{1 \text{ in entry } j+1} \\ &= (\alpha^{j+1})^{q^{i-1}} \\ &= \alpha^{(j+1)q^{i-1}}. \end{aligned}$$

For the last column, the fact that the polynomial evaluated at its own root is 0 is used to obtain $\alpha^n = -p_0 - \alpha p_1 - \cdots - \alpha^{n-1} p_{n-1}$. The element $(P_3 C)_{i,n}$ is

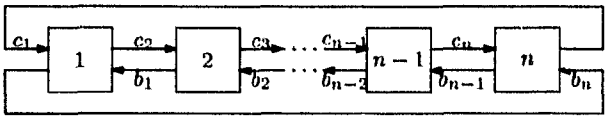
$$\begin{aligned} (P_3 C)_{i,n} &= [\alpha^{q^{i-1}}, (\alpha^2)^{q^{i-1}}, \ldots, (\alpha^n)^{q^{i-1}}] \cdot [-p_0, -p_1, \ldots, -p_{n-1}]^{\mathrm{T}} \\ &= (-\alpha p_0 - \alpha^2 p_1 - \cdots - \alpha^n p_{n-1})^{q^{i-1}} \\ &= (\alpha(-p_0 - \alpha p_1 - \cdots - \alpha^{n-1} p_{n-1}))^{q^{i-1}} \\ &= (\alpha \alpha^n)^{q^{i-1}} \\ &= \alpha^{(n+1)q^{i-1}}. \end{aligned}$$

Hence $P_3 C = D P_3$.

By composing the three transforms,

$$\begin{aligned} A &= P_1 A' P_1^{-1} \\ &= P_1 P_2 D P_2^{-1} P_1^{-1} \\ &= P_1 P_2 P_3 C P_3^{-1} P_2^{-1} P_1^{-1} \\ &= (P_1 P_2 P_3) C (P_1 P_2 P_3)^{-1}. \end{aligned}$$

Figure 4.7: Structure of a fully connected cyclic CA



This provides a direct transform from a CA to its corresponding LFSR.

4.6 Cyclic CA

This section explores *cyclic CA* defined over GF(q). A relation between the characteristic polynomial of a cyclic CA and its underlying null-boundary CA is derived. No results analogous to the reducibility of cyclic CA characteristic polynomials over GF(2) has been found for GF(q). Empirical evidence suggests that no such property exists.

A cyclic CA is formed from a null-boundary CA by the addition of connections between the two end cells. A diagram of the interconnection structure of a cyclic CA is contained in Figure 4.7.

As for cyclic CA over GF(q), the characteristic polynomial of a cyclic CA is invariant under rotations of the cell labels. For a fixed labeling, the underlying null-boundary CA is obtained by setting c_1 and b_n to 0. The characteristic polynomial of the underlying null-boundary CA is denoted $\Delta_{1,n}$, and submachines and their characteristic polynomials are defined as usual.

The transition matrix of an n-cell cyclic CA is given by

$$A_{\text{cyc}} = \begin{bmatrix} d_1 & b_1 & 0 & \cdots & \cdots & 0 & c_1 \\ c_2 & d_2 & b_2 & \ddots & & & 0 \\ 0 & c_3 & d_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & d_{n-2} & b_{n-2} & 0 \\ 0 & & & \ddots & c_{n-1} & d_{n-1} & b_{n-1} \\ b_n & 0 & \cdots & \cdots & 0 & c_n & d_n \end{bmatrix} \tag{4.8}$$

Lemma 4.9 *The characteristic polynomial of a cyclic CA satisfies the relationship*

$$\Gamma_{1,n} = \Delta_{1,n} - c_1 b_n \Delta_{2,n-1} + (-1)^{n+1}(\prod_{k=1}^{n} b_k + \prod_{k=1}^{n} c_k).$$

Proof. Consider the characteristic matrix of an n-cell CA:

$$\begin{bmatrix} a_1 & b_1 & 0 & \cdots & \cdots & 0 & c_1 \\ c_2 & a_2 & b_2 & \ddots & & & 0 \\ 0 & c_3 & a_3 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & a_{n-2} & b_{n-2} & 0 \\ 0 & & & \ddots & c_{n-1} & a_{n-1} & b_{n-1} \\ b_n & 0 & \cdots & \cdots & 0 & c_n & a_n \end{bmatrix}.$$

By expanding the determinant along the first row,

$$\Gamma_{1,n} = a_1 \Delta_{2,n} + (-b_1)|B| + (-1)^{n+1} c_1 |D|,$$

where

$$B = \begin{bmatrix} c_2 & b_2 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & a_3 & b_3 & \ddots & & & 0 \\ 0 & c_4 & a_4 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & a_{n-2} & b_{n-2} & 0 \\ 0 & & & \ddots & c_{n-1} & a_{n-1} & b_{n-1} \\ b_n & 0 & \cdots & \cdots & 0 & c_n & a_n \end{bmatrix}$$

and

$$D = \begin{bmatrix} c_2 & a_2 & b_2 & 0 & \cdots & \cdots & 0 \\ 0 & c_3 & a_3 & b_3 & \ddots & & \\ \vdots & \ddots & c_4 & \ddots & \ddots & \ddots & : \\ \vdots & & \ddots & \ddots & \ddots & b_{n-3} & 0 \\ \vdots & & & \ddots & \ddots & a_{n-2} & b_{n-2} \\ 0 & & & & \ddots & c_{n-1} & a_{n-1} \\ b_n & 0 & \cdots & \cdots & \cdots & 0 & c_n \end{bmatrix}.$$

Now, by expanding B along the first column,

$$|B| = c_2 \Delta_{3,n} + (-1)^n b_n |E|$$

where

$$E = \begin{bmatrix} b_2 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ a_3 & b_3 & \ddots & & & & 0 \\ c_4 & a_4 & \ddots & \ddots & & & \vdots \\ 0 & c_5 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & a_{n-3} & b_{n-3} & \ddots & \vdots \\ \vdots & & \ddots & c_{n-2} & a_{n-2} & b_{n-2} & 0 \\ 0 & \cdots & \cdots & 0 & c_{n-1} & a_{n-1} & b_{n-1} \end{bmatrix}.$$

By expanding D along the last row, $|D| = (-1)^n b_n \Delta_{2,n-1} + (-1)^{n+n} c_n |F|$, where

$$F = \begin{bmatrix} c_2 & a_2 & b_2 & 0 & \cdots & \cdots & 0 \\ 0 & c_3 & a_3 & b_3 & \ddots & & \vdots \\ \vdots & \ddots & c_4 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & b_{n-4} & 0 \\ \vdots & & & \ddots & \ddots & a_{n-3} & b_{n-3} \\ \vdots & & & & \ddots & c_{n-2} & a_{n-2} \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & c_{n-1} \end{bmatrix}.$$

Since E and F are lower and upper triangular, respectively, $|E| = \prod_{i=2}^{n-1} b_i$ and $|F| = \prod_{i=2}^{n-1} c_i$. Substituting these into the expressions for $|B|$ and $|D|$,

$$\begin{aligned} |B| &= c_2 \Delta_{3,n} + (-1)^n b_n \prod_{i=2}^{n-1} b_i \\ &= c_2 \Delta_{3,n} + (-1)^n \prod_{i=2}^{n} b_i \end{aligned}$$

$$\begin{aligned} |D| &= (-1)^n b_n \Delta_{2,n-1} + (-1)^{n+n} c_n \prod_{i=2}^{n-1} c_i \\ &= (-1)^n b_n \Delta_{2,n-1} + \prod_{i=2}^{n} c_i. \end{aligned}$$

$$\begin{aligned} \text{Hence, } \Gamma_{1,n} &= a_1 \Delta_{2,n} + -b_1 |B| + (-1)^{n+1} c_1 |D| \\ &= a_1 \Delta_{2,n} + -b_1 (c_2 \Delta_{3,n} \\ &\quad + (-1)^n \prod_{i=2}^{n} b_i) + (-1)^{n+1} c_1 ((-1)^n b_n \Delta_{2,n-1} + \prod_{i=2}^{n} c_i) \end{aligned}$$

$$
\begin{aligned}
&= a_1\Delta_{2,n} - b_1c_2\Delta_{3,n} + (-1)^{n+n+1}c_1b_n\Delta_{2,n-1} \\
&\quad + (-1)^{n+1}\prod_{i=1}^{n} b_i + (-1)^{n+1}\prod_{i=1}^{n} c_i \\
&= \Delta_{1,n} - c_1b_n\Delta_{2,n-1} + (-1)^{n+1}(\prod_{i=1}^{n} b_i + \prod_{i=1}^{n} c_i).
\end{aligned}
$$

This completes the proof. □

A null-boundary CA is a special case of a cyclic CA, for which $c_1 = b_n = 0$. Note that if these zero values are substituted into Lemma 4.9,

$$
\begin{aligned}
\Gamma_{1,n} &= \Delta_{1,n} - c_1b_n\Delta_{2,n-1} + (-1)^{n+1}(\prod_{k=1}^{n} b_k + \prod_{k=1}^{n} c_k) \\
&= \Delta_{1,n} - 0 \cdot 0 \cdot \Delta_{2,n-1} + (-1)^{n+1}(0 + 0) \\
&= \Delta_{1,n},
\end{aligned}
$$

as it must be.

4.7 Conclusion

This chapter defines CA that operate over any finite field GF(q). Basic results concerning the characteristic polynomials of these CA are derived, including their efficient computation. The concatenation relation and the GCD relation from Chapter 3 are extended to GF(q). A three-part similarity transform between an irreducible CA and the corresponding LFSR is derived. A cyclic CA over GF(q) is defined, and a relation between its characteristic polynomial and the characteristic polynomial of its underlying null-boundary CA is shown.

Chapter 5

Special forms of CA and other results

This chapter contains a variety of results about the characteristic polynomials of CA over GF(2). Both generalised and specialised forms of CA are defined and analysed. A few of the results are extended to apply to CA over GF(q).

Section 5.1 investigates the use of rules other than 90 and 150 in a CA. Starting from a fully-connected cyclic CA, various classes of *partially-connected* cyclic CA are obtained, one of which is the class of fully-connected null-boundary CA. In section 5.2, two alternate formulations of the characteristic polynomial are presented, one due to Knuth, and the other by Gauss. Motzkin extended the Gauss result, providing an alternate formulation for cyclic CA characteristic polynomials. Though not central to this dissertation, these formulations are an interesting side issue. Section 5.3 shows the effect on the characteristic polynomial of a CA when various changes are made to the multipliers. These changes are crucial for the derivation of the synthesis algorithm in Chapter 6. Section 5.4 defines and analyses three special types of CA, namely palindromic, self-concatenated, and uniform CA. The properties of palindromic CA are also important for the synthesis results; therefore they are the focus of much attention here.

5.1 Classification of CA

This section explores the various cases that arise when linear rules other than 90 and 150 are used in a fully-connected cyclic CA. Selective use of other rules gives rise to several CA classes, one of which is the class of null-boundary CA. For each of these classes, properties of the characteristic polynomials of the CA in the class are determined.

As discussed in section 2.2, there are eight linear functions of three variables, and hence eight possible rules for each cell. These rules are shown in the following table (note that subscript arithmetic is performed cyclicly in $\{1, 2, \ldots, n\}$.

rule #	Next state function	b_i	c_i
0	0	0	0
204	s_i	0	0
170	s_{i-1}	1	0
102	$s_{i-1} + s_i$	1	0
240	s_{i+1}	0	1
60	$s_i + s_{i+1}$	0	1
90	$s_{i-1} + s_{i+1}$	1	1
150	$s_{i-1} + s_i + s_{i+1}$	1	1

Consider a fully-connected cyclic CA, as defined in section 3.6, and its transition matrix A_{cyc}. Each cell of the fully-connected cyclic CA uses either rule 90 or rule 150. The use of a rule other than 90 or 150 by a cell i means that at least one of the input multipliers b_i or c_i is zero. The use of these alternate rules is, in turn, equivalent to 'breaking' or 'removing' some number of connections in the fully-connected cyclic CA.

In this section, the term *CA* refers to a machine in which each cell uses any of the eight possible linear rules. Hence the terms *null-boundary CA* and *fully-connected null-boundary* are used when appropriate. Six classes of CA are defined. Figure 5.1 contains interconnection diagrams and examples for each class. Descriptions of the six classes are as follows (refer to the matrix in (4.8)):

1. *Fully-connected cyclic.* Such a CA has $b_i = c_i = 1$ for $i = 1, 2, \ldots, n$, so that all

possible connections are present.

2. *Single-break cyclic.* This form of CA has a single connection removed. Without loss of generality, this connection is assumed to be c_1.

3. *Fully-connected null boundary.* A fully-connected null-boundary CA is obtained from a single-break cyclic CA by removing the connection opposite to the single break. As the single break was assumed to be c_1, the additional break is b_n.

4. *Non-opposite breaks.* A member of this class is obtained by removing a single connection on the opposite side to (but not opposite to) the single break in a single-break cyclic CA.

5. *Multiple-breaks one side, cyclic.* Such a CA is obtained by removing additional connections on the same side as the broken connection in a single-break cyclic CA.

6. *Partially-connected null-boundary.* This is a null-boundary CA with at least one more connection broken.

To aid in the classification, the concept of *fully dependent* is required. This rigorously defines the intuitive concept of information flow in a CA.

Definition 5.1 *A CA is* fully dependent *if every cell depends on every other cell after some number of time steps.*

For example, in Figure 5.3, cell 2 is dependent on cell 4 (via 3), but cell 4 is not dependent on cell 2. The following lemma gives a condition that is equivalent to fully dependent.

Lemma 5.2 *A CA is fully dependent if and only if for each i such that $c_i = 0$, it is the case that $b_j = 1$ for $j = 1, \ldots, i-2, i, \ldots, n$.*

Figure 5.1: CA classes

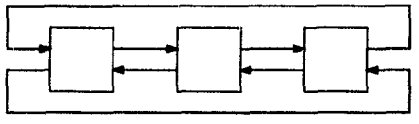
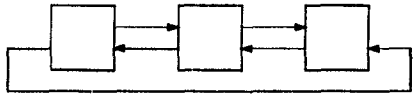
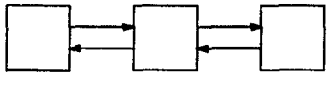
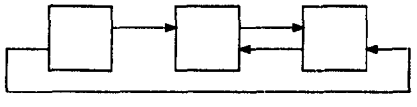
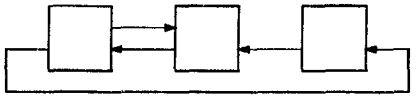
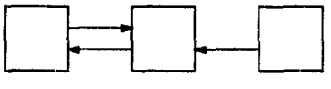
Class	Structure	Matrix
1		$\begin{bmatrix} d_1 & 1 & 1 \\ 1 & d_2 & 1 \\ 1 & 1 & d_3 \end{bmatrix}$
2		$\begin{bmatrix} d_1 & 1 & 0 \\ 1 & d_2 & 1 \\ 1 & 1 & d_3 \end{bmatrix}$
3		$\begin{bmatrix} d_1 & 1 & 0 \\ 1 & d_2 & 1 \\ 0 & 1 & d_3 \end{bmatrix}$
4		$\begin{bmatrix} d_1 & 0 & 0 \\ 1 & d_2 & 1 \\ 1 & 1 & d_3 \end{bmatrix}$
5		$\begin{bmatrix} d_1 & 1 & 0 \\ 1 & d_2 & 1 \\ 1 & 0 & d_3 \end{bmatrix}$
6		$\begin{bmatrix} d_1 & 1 & 0 \\ 1 & d_2 & 1 \\ 0 & 0 & d_3 \end{bmatrix}$

Figure 5.2: CA class relationships

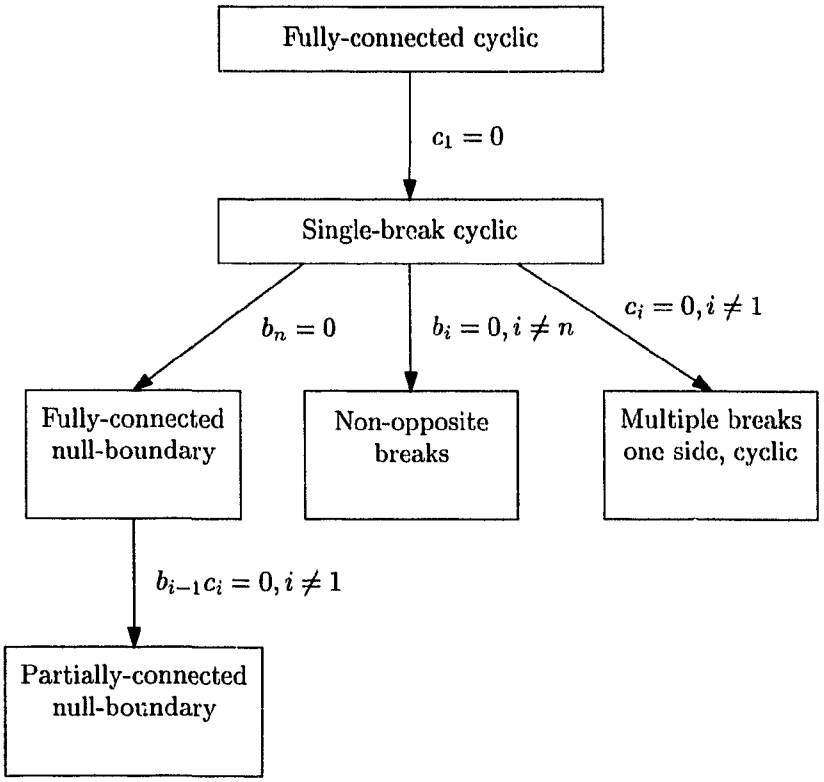
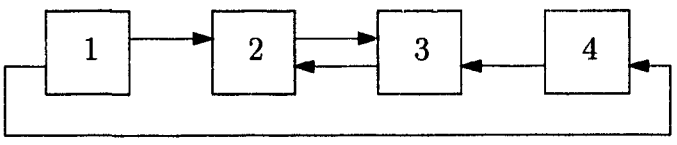


Figure 5.3: A non fully dependent CA



Proof. ($\Rightarrow$) Suppose the CA is fully dependent. If $c_i = 0$ for some i, then there is no length 1 directed path to cell i from cell $i-1$. Hence there is a path from cell $i-1$ to cell i only if the CA has the length $n-1$ path

$$\text{cell } i \to \text{cell } i+1 \to \cdots \to \text{cell } i-1,$$

which is to say $b_i = b_{i+1} = \cdots = b_{i-2} = 1$.

($\Leftarrow$) Suppose that $c_i = 0 \Rightarrow b_j = 1$ for $j = 1, \ldots, i-2, i, \ldots, n$. If $c_i = 1$ for all i, then the CA is fully dependent. If $c_i = 1$ for exactly one i, then $b_i = b_{i+1} = \cdots = b_{i-2} = 1$, and the CA is fully dependent. If $c_i = c_j = 0$ for some $i \neq j$, then $b_k = 1$ for all k, and the CA is fully dependent. □

The following result is used to show that a non fully dependent CA has a reducible characteristic polynomial.

Lemma 5.3 *If M is a CA with $b_n = c_1 = 0$ (i.e. M is null-boundary), and $b_i = 0$ for some $i \neq n$, then the characteristic polynomial of M is given by*

$$\Delta_{1,i}\Delta_{i+1,n},$$

and hence is reducible.

Proof. This result is the same as Corollary 4.6. □

Corollary 5.4 *If a CA M is not fully dependent, then its characteristic polynomial is reducible.*

Proof. If $c_i = 1$ for all i, then clearly M is fully dependent. So, without loss of generality, assume that $c_1 = 0$. By Lemma 5.2, there exists $j \neq n$ such that $b_j = 0$. Applying Lemma 4.9 to M, it is the case that the characteristic polynomial of M is

$$\Delta_{1,n} + 0 \cdot \Delta_{2,n-1} + 0 + 0.$$

In effect, this is saying that the multiplier b_n is not relevant: the characteristic polynomial of M is $\Delta_{1,n}$ no matter what the value of b_n. Thus Lemma 5.3 applies, and the corollary follows. □

Using the above results, statements about properties of the characteristic polynomial for each of the six classes are deduced. Each class has results stated in terms of the characteristic polynomial of the *corresponding null-boundary CA*, obtained by adding or removing connections as necessary.

1. *Fully-connected cyclic.* For this case, the characteristic polynomial is described by direct application of the cyclic CA results of Chapter 3. The characteristic polynomial is given by $\Delta_{1,n} + \Delta_{2,n-1}$ (Lemma 3.27), and is reducible (Theorem 3.28).

2. *Single-break cyclic.* By examining Lemma 4.9, it is evident that setting c_1 to 0 causes the $\Delta_{2,n-1}$ term and one of the products to drop out. Thus the characteristic polynomial is $\Delta_{1,n} + 1$, and can be either reducible or irreducible.

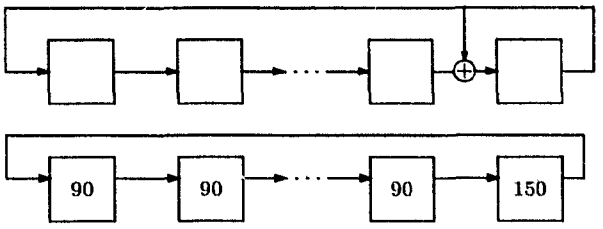
3. *Fully-connected null-boundary.* This is the "defining" class, in the sense that elements of the other classes are specified in terms of elements of this class.

4. *Non-opposite breaks.* By Lemma 5.2 this CA is not fully dependent, and hence its characteristic polynomial is reducible (Lemma 5.4).

5. *Multiple-breaks one side, cyclic.* As with a single-break cyclic CA, the characteristic polynomial is given by $\Delta_{1,n} + 1$, and can be reducible or irreducible. Also, the LFSR with characteristic polynomial $x^n + x^{n-1} + 1$ can be modeled as such a CA (see Figure 5.4).

6. *Partially-connected null-boundary.* The breaking of any connections in a null-boundary CA results in a CA to which Lemma 5.3 applies. Hence such a CA has a reducible characteristic polynomial.

Figure 5.4: A LFSM that is both a CA and an LFSR



5.2 Alternative formulations of the characteristic polynomial

The computation of CA characteristic polynomials can be modeled as a product of 2 by 2 matrices. This formulation gives better insight into the characteristic polynomials of concatenated CA, and provides an alternate proof for Theorem 3.6. The basis for this formulation can be found in [40], as part of the analysis of the complexity of Euclid's algorithm for integer greatest common divisor computation.

Consider an n-cell CA with rule vector $[d_1, d_2, \ldots, d_n]$. Define the matrix T_i as

$$T_i = \begin{bmatrix} x + d_i & 1 \\ 1 & 0 \end{bmatrix}.$$

Now denote the product of consecutively numbered matrices as

$$T_{i,j} = \prod_{k=i}^{j} T_k.$$

The following lemma describes the result of this multiplication.

Lemma 5.5

$$T_{i,j} = \begin{bmatrix} \Delta_{i,j} & \Delta_{i,j-1} \\ \Delta_{i+1,j} & \Delta_{i+1,j-1} \end{bmatrix}.$$

Proof. The proof is by straightforward induction, using (3.3) and (3.4) from section 3.1. □

To derive Theorem 3.6, calculate the determinant of $T_{1,n}$. Note that for the matrix T_i,

$$|T_i| = 1.$$

Hence

$$\begin{aligned} T_{1,n} &= \begin{vmatrix} \Delta_{1,n} & \Delta_{1,n-1} \\ \Delta_{2,n} & \Delta_{2,n-1} \end{vmatrix} \\ &= \Delta_{1,n-1}\Delta_{2,n} + \Delta_{1,n}\Delta_{2,n-1} \\ &= 1. \end{aligned}$$

Now consider $T_{1,k}$ and $T_{k+1,n}$, for the submachines $M_{1,k}$ and $M_{k+1,n}$. Since matrix multiplication is associative,

$$T_{1,n} = T_{1,k}T_{k+1,n}.$$

Alternatively, knowning $T_{1,n}$ and $T_{1,m}$ for two CA of length n and m, respectively, gives $T_{1,n+m}$ for the machine obtained by concatenating these CA.

There is another way of viewing the characteristic polynomial of a CA, due to Gauss. Let a_i denote $x+d_i$ (a diagonal element of the characteristic matrix). Consider the vector $a = [a_1, a_2, \ldots, a_n]$. Let Θ be the set of vectors obtained by removing an even number of adjacent elements from a in all possible ways. For example, with $n = 3$,

$$\Theta = \{[a_1, a_2, a_3], [a_1], [a_3]\},$$

for $n = 4$,

$$\Theta = \{[a_1, a_2, a_3, a_4][a_1, a_2], [a_1, a_4], [a_3, a_4], []\},$$

and for $n = 5$,

$$\Theta = \{[a_1, a_2, a_3, a_4, a_5], [a_1, a_2, a_3], [a_1, a_2, a_5], [a_1, a_4, a_5], [a_3, a_4, a_5], [a_1], [a_3], [a_5]\}.$$

Now consider each vector in Θ as a product of the a_i in the vector. The characteristic polynomial of the CA is the sum of these products. For $n = 3$,

$$\Delta_{1,n} = a_1a_2a_3 + a_1 + a_3,$$

for $n = 4$,

$$\Delta_{1,n} = a_1a_2a_3a_4 + a_1a_2 + a_1a_4 + a_3a_4,$$

and for $n = 5$,

$$\Delta_{1,n} = a_1a_2a_3a_4a_5 + a_1a_2a_3 + a_1a_2a_5 + a_1a_4a_5 + a_3a_4a_5 + a_1 + a_3 + a_5.$$

It is easily verified that each of these expressions is in fact the characteristic polynomial of the specified size CA. This formulation gives another way to see that the degree $n-1$ coefficient of the characteristic polynomial is equal to the weight (mod 2) of the CA. The only term of the sum that has degree at least $n-1$ is the product of all a_i, and so this product specifies the degree $n-1$ term. The product, written without substituting a_i for d_i, is

$$(x+d_1)(x+d_2)\cdots(x+d_n).$$

The degree $n-1$ coefficient of this product is $d_1+d_2+\cdots+d_n$, which is the weight of the CA.

A natural extension of the above property exists for cyclic CA, due to Motzkin and Straus ([50]). Their work concerns the optimisation of functions of finitely or infinitely many variables, where the variables are identified with the points on a linear graph. In this extension, they use expressions that are obtained by removing, in all possible ways, an even number of adjacent elements from the vector $a=[a_1,a_2,\ldots,a_n]$. However, the vector is treated as cyclic rather than linear. For example, with $n=3$,

$$\Theta_{\text{cyc}}=\{[a_1,a_2,a_3],[a_1],[a_2],[a_3]\},$$

and for $n=4$,

$$\Theta_{\text{cyc}}=\{[a_1,a_2,a_3,a_4][a_1,a_2],[a_1,a_4],[a_2,a_3],[a_3,a_4],[],\}.$$

Taking the sum of the products of the terms as before results in expressions for the characteristic polynomials of cyclic CA:

$$\Gamma_{1,3}=a_1a_2a_3+a_1+a_2+a_3,$$

and

$$\Gamma_{1,4}=a_1a_2a_3a_4+a_1a_2+a_1a_4+a_2a_3+a_3a_4.$$

5.3 Results of simple CA modification

This section explores the effect of making small modifications to a CA. More specifically, it is shown how the characteristic polynomial of a CA changes when one of the following modifications is performed:

- the self-multiplier d_n is changed to $d_n + 1$,
- a self-multiplier d_i is changed to $d_i + 1$,
- all of the self-multipliers $d_1, d_2, \ldots, d_n$ are changed, becoming $d_1 + 1, d_2 + 1, \ldots, d_n + 1$.

For each case, the characteristic polynomial of the modified CA is derived in terms of the characteristic polynomials of the unmodified CA and its submachines.

For the first case, consider a CA with rule vector $d = [d_1, d_2, \ldots, d_n]$, and characteristic polynomial $\Delta_{1,n}$. Now consider the CA with rule vector $d' = [d_1, d_2, \ldots, d_{n-1}, d_n + 1]$, obtained from d by changing the self-multiplier used in cell n. Denote the characteristic polynomial of this modified CA by $\Delta'_{1,n}$. By the CA recurrence (3.1), and using the fact that $\Delta_{1,k} = \Delta'_{1,k}$ for $k < n$,

$$\begin{aligned} \Delta'_{1,n} &= (x + d_n + 1)\Delta'_{1,n-1} + \Delta'_{1,n-2} \\ &= (x + d_n + 1)\Delta_{1,n-1} + \Delta_{1,n-2} \\ &= (x + d_n)\Delta_{1,n-1} + \Delta_{1,n-2} + \Delta_{1,n-1} \\ &= \Delta_{1,n} + \Delta_{1,n-1}. \end{aligned} \tag{5.1}$$

So changing the rule used in cell n causes the characteristic polynomial to change by an additive factor of $\Delta_{1,n-1}$. As a trivial consequence, note that the sum of $\Delta_{1,n}$ and $\Delta'_{1,n}$ is $\Delta_{1,n-1}$ $(= \Delta'_{1,n-1})$, the characteristic polynomial of the shared submachine with rule vector $[d_1, d_2, \ldots, d_{n-1}]$.

For the second case, now consider the CA with rule vector $d' = [d_1, \ldots, d_{i-1}, d_i + 1, d_{i+1}, \ldots, d_n]$, obtained by changing the self-multiplier used by cell i. Again, denote the characteristic polynomial of the modified CA by $\Delta'_{1,n}$. The concatenation relation

of Theorem 3.4 is applied to $\Delta'_{1,n}$ to obtain

$$\Delta'_{1,n} = \Delta'_{1,i}\Delta'_{i+1,n} + \Delta'_{1,i-1}\Delta'_{i+2,n}.$$

Observing that $\Delta_{1,k} = \Delta'_{1,k}$ for $k < i$ and $\Delta_{k,n} = \Delta'_{k,n}$ for $k > i$, and applying (5.1) to $\Delta'_{1,i}$,

$$\begin{aligned}
\Delta'_{1,n} &= \Delta'_{1,i}\Delta_{i+1,n} + \Delta_{1,i-1}\Delta_{i+2,n} \\
&= (\Delta_{1,i} + \Delta_{1,i-1})\Delta_{i+1,n} + \Delta_{1,i-1}\Delta_{i+2,n} \\
&= \Delta_{1,i}\Delta_{i+1,n} + \Delta_{1,i-1}\Delta_{i+2,n} + \Delta_{1,i-1}\Delta_{i+1,n} \\
&= \Delta_{1,n} + \Delta_{1,i-1}\Delta_{i+1,n}.
\end{aligned}$$

The equation for this change does not nest well. Suppose that the self-multipliers in both cells i and j are changed, with $i < j$. The characteristic polynomial of the resulting CA is

$$\Delta'_{1,n} = \Delta_{1,n} + \Delta_{1,i-1}\Delta_{i+1,n} + (\Delta_{1,j-1} + \Delta_{1,i-1}\Delta_{i+1,j-1})\Delta_{j+1,n}, \tag{5.2}$$

which cannot be significantly simplified. However, a special case of this relation is required for the results underlying the synthesis of CA in Chapter 6.

Lemma 5.6 *Let $d = [d_1, d_2, \ldots, d_n]$ be the rule vector of a CA, with characteristic polynomial $\Delta_{1,n}$. Then the characteristic polynomial of the CA with rule vector $d' = d_1 + 1, d_2, \ldots, d_{n-1}, d_n + 1]$ is*

$$\Delta'_{1,n} = \Delta_{1,n} + \Delta_{2,n} + \Delta_{1,n-1} + \Delta_{2,n-1}.$$

Proof. Substituting $i = 1$ and $j = n$ into (5.2),

$$\begin{aligned}
\Delta'_{1,n} &= \Delta_{1,n} + \Delta_{1,0}\Delta_{2,n} + (\Delta_{1,n-1} + \Delta_{1,0}\Delta_{2,n-1})\Delta_{n+1,n} \\
&= \Delta_{1,n} + 1 \cdot \Delta_{2,n} + (\Delta_{1,n-1} + 1 \cdot \Delta_{2,n-1}) \cdot 1 \\
&= \Delta_{1,n} + \Delta_{2,n} + \Delta_{1,n-1} + \Delta_{2,n-1}.
\end{aligned}$$

□

The third case is when all of the self-multipliers are changed, so that the rule vector of the modified CA is $d' = [d_1 + 1, d_2 + 1, \ldots, d_{n-1} + 1, d_n + 1]$. The characteristic polynomial $\Delta'_{1,n}$ of this CA is given by

$$\Delta'_{1,n} = \det(xI + A'),$$

where A' is the transition matrix of the modified CA. However, note that $A' = A + I$, where A is the transition matrix of the unmodified CA. Hence

$$\begin{aligned} \Delta'_{1,n} &= \det(xI + A + I) \\ &= \det((x + 1)I + A) \\ &= \Delta(x + 1), \end{aligned} \tag{5.3}$$

which is the characteristic polynomial $\Delta_{1,n}$ evaluated at $x+1$. Note that this evaluation trivially preserves irreducibility. It does not, however, preserve primitivity; consider $p(x) = x^4+x^3+1$ which is primitive over GF(2), and $p(x+1) = x^4+x^3+x^2+x+1$, which is not.

5.3.1 Extensions to GF(q)

The effect on the characteristic polynomial of modifications for machines over GF(q) is explored briefly. These results are simple generalisations of the GF(2) results above, except for the additional case of an off-diagonal multiplier change.

If a diagonal multiplier d_k is changed to $d_k + \beta$, β any element of GF(q), the resulting characteristic polynomial is

$$\Delta'_{1,n} = \Delta_{1,n} - \beta\Delta_{1,k-1}\Delta_{k+1,n}.$$

Similarly, if an off-diagonal multiplier b_k is changed to $b_k + \beta$, then the characteristic polynomial $\Delta'_{1,n}$ of the modified machine is derived by applying Theorem 4.2 and (4.5):

$$\begin{aligned} \Delta'_{1,n} &= \Delta_{1,k}\Delta_{k+1,n} - (b_k + \beta)c_k\Delta_{1,k-1}\Delta_{k+2,n} \\ &= \Delta_{1,k}\Delta_{k+1,n} - b_kc_k\Delta_{1,k-1}\Delta_{k+2,n} - \beta c_k\Delta_{1,k-1}\Delta_{k+2,n} \\ &= \Delta_{1,n} - \beta c_k\Delta_{1,k-1}\Delta_{k+2,n}. \end{aligned}$$

If all diagonal multipliers are simultaneously modified, d_k becoming $d_k + \beta$ for all k, then

$$\begin{aligned} \Delta'_{1,n} &= |xI - (A + \beta I)| \\ &= |xI - \beta I - A| \\ &= |(x - \beta)I - A|. \end{aligned}$$

Thus $\Delta'_{1,n}$ is $\Delta_{1,n}$ evaluated at $x - \beta$.

5.4 Palindromic, self-concatenated, and uniform CA

This section explores three special patterns of CA rule vectors, for fully-connected null-boundary CA. The first of these is *palindromic*, where the rule vector it identical to its own reversal. It is shown that the characteristic polynomials of such a CA is either a perfect square (n even), or is a monomial times a perfect square (n odd). The second special pattern occurs when the left half of the rule vector is equal to the right half (unreversed). Such a machine is formed when two copies of a CA are joined together, giving rise to the name *self-concatenated.* A formula is given for the characteristic polynomial of this type of CA, and can result in either a reducible or an irreducible polynomial. The third type of rule pattern is the most simple: where all of the cells use the same rule. This is known as a *uniform* CA. A closed formula for the characteristic polynomial for certain values of n is derived, and a fast general technique for calculating the characteristic polynomial is given.

5.4.1 Palindromic CA

A *palindromic* CA has a symmetric rule vector:

$$[d_1, \ldots, d_n] = [d_n, \ldots, d_1].$$

The following theorem shows that the characteristic polynomial of an even-length palindromic CA is a perfect square, and that the characteristic polynomial of an odd-length palindromic CA is the product of a monomial and a perfect square.

Theorem 5.7 *The characteristic polynomial $\Delta_{1,n}$ of a palindromic CA is given by*

$$\Delta_{1,n} = \begin{cases} (\Delta_{1,n/2} + \Delta_{1,n/2-1})^2 & \text{if } n \text{ is even, and} \\ (x + d_{(n+1)/2})(\Delta_{1,(n-1)/2})^2 & \text{if } n \text{ is odd.} \end{cases}$$

Proof. Suppose that n is even, and let $k = n/2$. The symmetry of the rule vector d implies that $[d_1, \ldots, d_n] = [d_1, \ldots, d_k, d_k, \ldots d_1]$. Hence $\Delta_{1,k} = \Delta_{k+1,n}$, and $\Delta_{1,k-1} = \Delta_{k+2,n}$. By substituting these into Theorem 3.4,

$$\begin{aligned} \Delta_{1,n} &= \Delta_{1,k}\Delta_{k+1,n} + \Delta_{1,k-1}\Delta_{k+2,n} \\ &= \Delta_{1,k}\Delta_{1,n} + \Delta_{1,k-1}\Delta_{1,k-1} \\ &= \Delta_{1,k}^2 + \Delta_{1,k-1}^2 \\ &= (\Delta_{1,k} + \Delta_{1,k-1})^2 \\ &= (\Delta_{1,n/2} + \Delta_{1,n/2-1})^2. \end{aligned}$$

Now suppose n is odd, and let $k = (n-1)/2$. Then the rule vector d has the form $[d_1, \ldots, d_n] = [d_1, \ldots, d_k, d_{k+1}, d_k, \ldots d_1]$. Hence $\Delta_{1,k} = \Delta_{k+2,n}$, and $\Delta_{1,k-1} = \Delta_{k+3,n}$. Now,

$$\begin{aligned} \Delta_{1,n} &= \Delta_{1,k}\Delta_{k+1,n} + \Delta_{1,k-1}\Delta_{k+2,n} \\ &= \Delta_{1,k}((x + d_{k+1})\Delta_{k+2,n} + \Delta_{k+3,n}) + \Delta_{1,k-1}\Delta_{k+2,n} \\ &= \Delta_{1,k}((x + d_{k+1})\Delta_{1,k} + \Delta_{1,k-1}) + \Delta_{1,k-1}\Delta_{1,k} \\ &= (x + d_{k+1})\Delta_{1,k}\Delta_{1,k} + \Delta_{1,k}\Delta_{1,k-1} + \Delta_{1,k-1}\Delta_{1,k} \\ &= (x + d_{k+1})(\Delta_{1,k})^2 \\ &= (x + d_{(n+1)/2})(\Delta_{1,(n-1)/2})^2. \end{aligned}$$

□

The characteristic polynomial of an even length palindromic CA is a perfect square. A nice corollary of this is that the square of a CA polynomial has a CA realisation.

Corollary 5.8 *If $\Delta_{1,n}$ is the CA polynomial of $[d_1, \ldots, d_n]$, then there exists a CA for $\Delta_{1,n}^2$.*

Proof. Consider the length $2n$ CA

$$[d_1, \ldots, d_{n-1}, d_n + 1, d_n + 1, d_{n-1}, \ldots, d_1].$$

Using Theorem 3.4, and applying (5.1) to the submachine $\Delta_{1,n}$, this CA has characteristic polynomial

$$\begin{aligned} \Delta_{1,2n} &= \Delta([d_1, \ldots, d_{n-1}, d_n + 1])\Delta([d_1, \ldots, d_{n-1}, d_n + 1]) + \Delta_{1,n-1}\Delta_{1,n-1} \\ &= (\Delta_{1,n} + \Delta_{1,n-1})(\Delta_{1,n} + \Delta_{1,n-1}) + \Delta_{1,n-1}\Delta_{1,n-1} \\ &= \Delta_{1,n}^2. \end{aligned}$$

Hence this CA has characteristic polynomial $\Delta_{1,n}^2$. □

It is interesting to note that

$$\begin{aligned} \Delta_{1,2n-1} &= \Delta([d_1, \ldots, d_{n-1}, d_n + 1])\Delta([d_2, \ldots, d_{n-1}, d_n + 1]) + \Delta_{1,n-1}\Delta_{2,n-1} \\ &= (\Delta_{1,n} + \Delta_{1,n-1})(\Delta_{2,n} + \Delta_{2,n-1}) + \Delta_{1,n-1}\Delta_{2,n-1} \\ &= \Delta_{1,n}\Delta_{2,n} + \Delta_{1,n}\Delta_{2,n-1} + \Delta_{1,n-1}\Delta_{2,n} + \Delta_{1,n-1}\Delta_{2,n-1} + \Delta_{1,n-1}\Delta_{2,n-2} \\ &= \Delta_{1,n}\Delta_{2,n} + \Delta_{1,n}\Delta_{2,n-1} + \Delta_{1,n-1}\Delta_{2,n} \\ &= \Delta_{1,n}\Delta_{2,n} + 1. \end{aligned}$$

A small result about square polynomials that is perhaps surprising is now presented. Above, it is shown that the characteristic polynomial of a palindromic CA is a perfect square. The following theorem gives a converse to this statement: that any CA that has a perfect square as its characteristic polynomial must be a palindrome. This provides a clean separation of CA into the classes palindromic/square and nonpalindromic/nonsquare.

Theorem 5.9 *If a polynomial p in $GF(2)[x]$ is a perfect square, then any CA with characteristic polynomial p is a palindrome.*

Proof. The proof is by induction on n, the degree of p. The base of the induction is with $n = 0$. The only degree 0 perfect square is $p = 1$, and the only CA that has this characteristic polynomial is the empty (zero length) CA. Trivially, this CA is a palindrome.

For the general case, let $d = [d_1, d_2, \ldots, d_n]$ be the rule vector of a CA with characteristic polynomial $p = \Delta_{1,n}$, and suppose $\Delta_{1,n}$ is a perfect square. It is shown that d is a palindrome.

First, note that $\Delta_{1,n}$ square implies that n is even. Second, note that $\Delta_{1,n}$ square implies that the degree $n - 1$ coefficient is 0, which in turn implies that the parity of d is even (Lemma 2.16). Using these two observations, the Cyclic CA Theorem (Theorem 3.28) states that

$$\Delta_{1,n} + \Delta_{2,n-1}$$

is a perfect square. As $\Delta_{1,n}$ is a perfect square, it must be the case that $\Delta_{2,n-1}$ is a perfect square.

The inductive hypothesis applied to $\Delta_{2,n-1}$ gives the fact that $[d_2, d_3, \ldots, d_{n-1}]$ is a palindrome. It remains to be shown that $d_1 = d_n$. However, the parity of $[d_2, d_3, \ldots, d_{n-1}]$ is even, and thus if d_1 did not equal d_n, it would be the case that the parity of d was odd. This is not so, and hence d must be a palindrome. □

The analogous result for n odd follows.

Theorem 5.10 *Suppose $p = (x + w)q^2$ for some degree $(n - 1)/2$ polynomial q and some $w \in GF(2)$. Then any CA with characteristic polynomial p is a palindrome, with $d_{(n+1)/2} = w$.*

Proof. The proof is by induction on n, the degree of p. The base of the induction is with $n = 1$, and is satisfied trivially for the polynomials x and $x + 1$.

For the general case, let $d = [d_1, d_2, \ldots, d_n]$ be the rule vector of a CA with characteristic polynomial $p = \Delta_{1,n}$, and suppose $\Delta_{1,n} = (x+w)q^2$. The degree $n-1$ coefficient of $\Delta_{1,n}$ is w, and so the weight of d is also w. Applying the Cyclic CA Theorem (Theorem 3.28) with n odd and $w = w_1 = 1$,

$$\Delta_{1,n} + \Delta_{2,n-1} = (x+w)t^2$$

for some polynomial t. Thus

$$\Delta_{2,n-1} = (x+w)q^2 + (x+w)t^2 = (x+w)(q+t)^2.$$

The inductive hypothesis applied to $\Delta_{2,n-1}$ gives the fact that $[d_2, d_3, \ldots, d_{n-1}]$ is a palindrome, with the centre cell using rule w. Since the weight of $\Delta_{2,n-1}$ is also w, is must be that $d_1 = d_n$. Therefore $[d_1, d_2, \ldots, d_n]$ is a palindrome. □

5.4.2 Self-concatenated CA

Interest has been shown in the concatenation properties of CA ([63]). A particular case that has been studied using empirical evidence is when a CA is concatenated with itself (not with its reversal, which is the palindromic case). There is an interesting expression that can be derived for this type of CA, and that involves the characteristic polynomial of a cyclic CA.

A self-concatenated CA has a rule vector of the form

$$[d_1, d_2, \ldots, d_{n-1}, d_n, d_1, d_2, \ldots, d_{n-1}, d_n].$$

The characteristic polynomial, $\Delta_{1,2n}$, is obtained by applying Theorem 3.4:

$$\Delta_{1,2n} = \Delta_{1,n}\Delta_{n+1,2n} + \Delta_{1,n-1}\Delta_{n+2,n}.$$

Noting that the self-concatenation gives $\Delta_{n+1,2n} = \Delta_{1,n}$ and $\Delta_{n+2,n} = \Delta_{2,n}$,

$$\Delta_{1,2n} = \Delta_{1,n}^2 + \Delta_{1,n-1}\Delta_{2,n}.$$

The GCD relation (Theorem 3.6) applied to $\Delta_{1,n-1}\Delta_{2,n}$ gives

$$\begin{aligned}\Delta_{1,2n} &= \Delta_{1,n}^2 + \Delta_{1,n}\Delta_{2,n-1} + 1 \\ &= \Delta_{1,n}(\Delta_{1,n} + \Delta_{2,n-1}) + 1.\end{aligned}$$

Applying the cyclic CA lemma (Lemma 3.27), the final result is obtained:

$$\Delta_{1,2n} = \Delta_{1,n}\Gamma_{1,n} + 1.$$

It is worth noting that if a single connection is added between cells 1 and n, the characteristic polynomial of the resulting machine (in terms of the characteristic polynomial $\Delta_{1,2n}$ of the unmodified machine) is

$$\Delta_{1,2n} + 1 = \Delta_{1,n}\Gamma_{1,n} + 1 + 1 = \Delta_{1,n}\Gamma_{1,n},$$

which is clearly reducible.

5.4.3 Uniform CA

A uniform CA is the simplist possible form of CA, in which all of the cells use the same rule. Formulas are derived for the characteristic polynomial of a CA with a constant-zero rule vector. The characteristic polynomial for the other case, where the rule vector is all ones, can be obtained via (5.3). Note that for a uniform CA, the characteristic polynomial of a submachine depends only on its length; for example, $\Delta_{1,n-1} = \Delta_{2,n}$.

Consider first the case of n odd, as it is more straightforward. Since a uniform CA is clearly a palindrome, Theorem 5.7 is applied to obtain

$$\Delta_{1,n} = x(\Delta_{1,(n-1)/2})^2. \tag{5.4}$$

Hence for the n-odd case, the formula for the characteristic polynomial depends only on the characteristic polynomial of the submachine $\Delta_{1,(n-1)/2}$. This gives the following clean result for a special case.

Lemma 5.11 *The characteristic polynomial of the all-0 uniform CA of length* $n = 2^k - 1 > 0$ *is* x^n.

Proof. To apply induction on k, note that the base case for $n = 1$ is satisfied as $\Delta_{1,1} = x$. For the inductive step, note that if $n = 2^k - 1$, then $(n-1)/2 = 2^{k-1} - 1$. Hence, by (5.4),

$$\Delta_{1,n} = x(\Delta_{1,(n-1)/2})^2 = x(x^{(n-1)/2})^2 = x^n.$$

□

For the n-even case, the result for palindromes (Theorem 5.7) states that

$$\Delta_{1,n} = (\Delta_{1,n/2} + \Delta_{1,n/2-1})^2.$$

Hence the calculation of the characteristic polynomial of an even length uniform CA can be reduced to two smaller calculations, of sizes $n/2$ and $n/2 - 1$. Also, one of these smaller cases is odd, and hence immediately reduces to a problem of size around $n/4$. An explicit formula exists for when $n = 2^k$, as follows.

Lemma 5.12 *The characteristic polynomial of the all-0 uniform CA of length* $n = 2^k > 0$ *is* $x^{2^k} + \sum_{i=1}^{k} x^{2^k - 2^i}$.

Proof. The base base of the inductive proof is for $k = 0$ ($n = 1$), in which case the summation is empty and the formula holds. For the general case, both the inductive hypothesis and Lemma 5.11 are used:

$$\begin{aligned}
\Delta_{1,2^k} &= (\Delta_{1,2^k/2} + \Delta_{1,(2^k/2)-1})^2 \\
&= (\Delta_{1,2^{k-1}})^2 + (\Delta_{1,2^{k-1}-1})^2 \\
&= (x^{2^{k-1}} + \sum_{i=1}^{k-1} x^{2^{k-1}-2^i})^2 + (x^{2^{k-1}-1})^2 \\
&= x^{2^k} + \sum_{i=1}^{k-1} x^{2^k - 2^{i+1}} + (x^{2^k-2}) \\
&= x^{2^k} + \sum_{i=2}^{k} x^{2^k-2^i} + (x^{2^k-2}) \\
&= x^{2^k} + \sum_{i=1}^{k} x^{2^{k-1}-2^i}.
\end{aligned}$$

Table 5.1: Characteristic polynomials of uniform CA of length 2^k

k	n	$\Delta_{1,n}$
0	1	x
1	2	x^2+1
2	4	x^4+x^2+1
3	8	$x^8+x^6+x^4+1$
4	16	$x^{16}+x^{14}+x^{12}+x^8+1$
5	32	$x^{32}+x^{30}+x^{28}+x^{24}+x^{16}+1$
6	64	$x^{64}+x^{62}+x^{60}+x^{56}+x^{48}+x^{32}+1$

□

The characteristic polynomials for the first seven values of k are given in Table 5.1.

5.5 Conclusion

The first part of this chapter generalises CA by allowing linear rules other than 90 and 150, and derives a classification scheme for these CA. Some interesting alternative formulations, based on work by other authors, of the characteristic polynomial of null-boundary and cyclic-boundary CA are presented. The effect on the characteristic polynomial of a null-boundary CA when diagonal or off-diagonal multipliers are modified is deduced. Various specialised forms of null-boundary CA are defined, these being palindromic, self-concatenated, and uniform. The palindromic CA are especially important for the synthesis algorithm in the next chapter.

Chapter 6

Synthesis of CA

This chapter presents an efficient algorithm for the synthesis of a CA for a given irreducible polynomial. This synthesis hinges on a relation between the computation of the characteristic polynomial of a CA and Euclid's greatest common divisor algorithm. It also shows that exactly two CA exist for each irreducible polynomial, solving the *CA existence conjecture.*

This chapter is organised as follows. Section 6.1 formally defines the synthesis problem, and describes previous attempts at its solution. Section 6.2 shows the relationship between synthesis and Euclid's GCD algorithm, by reformulating the synthesis problem in terms of worst-cases instances for Euclid's algorithm. A crucial result concerning the *sum of subpolynomials* is derived in section 6.3. This leads to both the necessary conditions for the existence of such worst-cases, and the *CA quadratic congruence* (contained in section 6.4). Section 6.5 presents a standard technique for solving a quadratic in characteristic 2, which completes the description of the synthesis algorithm. Section 6.6 gives a full example of the synthesis, and section 6.7 discusses implementation issues. Section 6.8 derives a result that determines the amount of symmetry that a CA possesses, as a function of its characteristic polynomial.

6.1 Background

In Chapter 3, it is shown that obtaining the characteristic polynomial of a CA is a well-defined problem with an easily computed solution. That is, the function that maps the set of CA into the set of polynomials is well-defined, via a determinant. Calculating the inverse of this function is called the *synthesis problem.*

Recall from Chapter 2 that if the CA M has characteristic polynomial p, then M is said to be a *realisation* of p. Also, if p is the characteristic polynomial of a CA, p is said to be a *CA polynomial.*

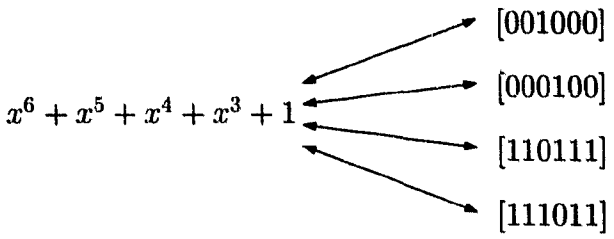
The synthesis problem has been considered by several researchers. Serra *et al.* ([61]) present a technique that is essentially a pruned search through the space of all CA. The pruning is based on knowledge of the weight (modulo 4) of the CA vector. The CA characteristic polynomial is calculated coefficient-by-coefficient, so as to allow early detection of an incorrect CA. This approach has a complexity of approximately 2^{n-2}.

A more sophisticated approach by Serra and Slater can be found in [60]. The authors derive a modification to the Lanczos tridiagonalisation algorithm. The CA is obtained by tridiagonalising the companion matrix of a polynomial. Unfortunately, the algorithm is not constructive over finite fields (it is for complex matrices). The search that is required to overcome the non-constructivity has complexity $2^{n/2}$. This is significantly better than the [61] approach, but is still limited to n around 60.

Das *et al.* ([25]) state that "[t]he problem of synthesising such an L-cell CA reduces to solving an independent set of L nonlinear equations." The authors go on to say that "[t]he problem is currently being worked out and will be reported shortly," but further material is not available.

Typically, a characteristic polynomial is not sufficient to determine uniquely the CA from which it was calculated. To start with, a CA and its reversal have the same characteristic polynomial (except in the special case that the CA is symmetric, in which case it is its own reversal). From the point of view of this work, it is preferable

Figure 6.1: Lack of 1-1 correspondence between CA and polynomials



to consider a CA and its reversal as different, so, for example, [01100] and [00110] are different CA with the same characteristic polynomial. Going further, however, it is evident that the lack of uniqueness extends beyond the issue of the reversal of a CA. Figure 6.1 shows a degree 6 polynomial with four CA realisations. Hence, there is not a one-to-one correspondence between CA polynomials and their CA realisations, no matter how reversals are handled.

Consider also that there are 2^n degree n polynomials and 2^n length n CA. The lack of 1-1 correspondence thus implies that not all polynomials have CA realisations; one such polynomial is $x^2 + x$. The counts in the GF(2) table in Appendix E indicate that, as n increases, the number of CA polynomials increases by a factor that is a little less than 2 (whereas the number of CA increases by a factor of exactly 2).

The characteristic polynomial of the submachine consisting of cells 1 through $n-1$ plays a crucial role in the material in this chapter. This motivates the following definition.

Definition 6.1 *Let M be a CA with characteristic polynomial $\Delta_{1,n}$. Then $\Delta_{1,n-1}$ is called a* CA subpolynomial, *and the pair of polynomials $\Delta_{1,n}, \Delta_{1,n-1}$ is called a* CA polynomial-subpolynomial pair.

Note that a CA subpolynomial is itself a CA polynomial.

Throughout this section, the notion of a formal derivative p' of a polynomial p over GF(2) is used. Recall that p' is defined as the usual (i.e. integer coefficient) polynomial derivative with coefficients reduced modulo 2 (Definition 2.7). For example, if $p =$

$x^5 + x^4 + x^3 + 1$, then $p' = 5x^4 + 4x^3 + 3x^2 = x^4 + x^2$.

6.2 Euclid's GCD algorithm

The calculation of the characteristic polynomial of a CA is intimately related to *Euclid's greatest common divisor (GCD) algorithm*. This relation forms the basis for the synthesis algorithm derived below. This section shows how the CA recurrence ((3.1)) satisfies the *division algorithm* for polynomials, and then demonstrates that the repeated application of the recurrence is, in effect, a reverse GCD computation.

The division algorithm for integers states that given two integers, a and b, $b \neq 0$, there exists a unique quotient and unique non-negative remainder when a is divided by b. Analogously, the division algorithm for polynomials states that given two polynomials, a and b, $b \neq 0$, there exist unique polynomials q and r, such that

$$\begin{aligned} a &= q \cdot b + r \\ \text{and} \quad r &= 0 \text{ or } \deg(r) < \deg(b). \end{aligned}$$

For example, if $x^5 + x^4 + x^3$ is divided by $x^3 + x + 1$,

$$x^5 + x^4 + x^3 = (x^2 + x)(x^3 + x + 1) + x,$$

and so $q = x^2 + x$ and $r = x$. The polynomial a is the *dividend*, b is the *divisor*, q is the *quotient*, and r is the *remainder*. Note that, if $\deg(a) \geq \deg(b)$, then the degree of the quotient is $\deg(a) - \deg(b)$. Often only the remainder r is of interest, in which case it is denoted $a \bmod b$. This is called the *modulo b* operation. For example, $x^5 + x^4 + x^3 \bmod x^3 + x + 1$ is x.

Now, recall the CA recurrence ((3.1)),

$$\Delta_k = (x + d_k)\Delta_{k-1} + \Delta_{k-2}.$$

Since $\deg(\Delta_{k-2}) = k - 2 < k - 1 = \deg(\Delta_{k-1})$, this equation satisfies the division algorithm, with

$$\begin{array}{lcll} a & = & \Delta_k, & \text{dividend,} \\ b & = & \Delta_{k-1}, & \text{divisor,} \\ q & = & x + d_k, & \text{quotient,} \\ r & = & \Delta_{k-2}, & \text{remainder.} \end{array}$$

This means that Δ_k and Δ_{k-1} uniquely determine $x \div d_k$ and Δ_{k-2}. An extremely important property is that, in this particular case, the quotient has degree 1.

It is now shown that repeated application of the division algorithm reverses the computation of the characteristic polynomial of a CA. Suppose that Δ_n and Δ_{n-1} are known. One application of the division algorithm calculates d_n and Δ_{n-2}. Hence if $\Delta = \Delta_n$ and Δ_{n-1} are known, then $x + d_n$ and Δ_{n-2} are uniquely determined, and easily calculated. If the division algorithm is then applied to Δ_{n-1} and Δ_{n-2}, it calculates $x + d_{n-1}$ and Δ_{n-3}. Continuing this process calculates the sequence of polynomials $\Delta_{n-2}, \Delta_{n-3}, \ldots, \Delta_2, \Delta_1, \Delta_0(=1), \Delta_{-1}(=0)$.

Example 6.2. To illustrate, this process is carried out for the CA with rule vector [0,1,1,0,0]. For this CA, $\Delta_n = x^5 + x^3 + 1$ and $\Delta_{n-1} = x^4 + 1$, as shown by the following calculation.

$$
\begin{array}{lcllll}
\Delta_{1,-1} & = & 0 & & & \\
\Delta_{1,0} & = & 1 & & & \\
\Delta_{1,1} & = & (x+d_1)\Delta_{1,0} + \Delta_{1,-1} & = & (x+0)1+0 & = x \\
\Delta_{1,2} & = & (x+d_2)\Delta_{1,1} + \Delta_{1,0} & = & (x+1)x+1 & = x^2+x+1 \\
\Delta_{1,3} & = & (x+d_3)\Delta_{1,2} + \Delta_{1,1} & = & (x+1)(x^2+x+1)+x & = x^3+x+1 \\
\Delta_{1,4} & = & (x+d_4)\Delta_{1,3} + \Delta_{1,2} & = & (x+0)(x^3+x+1)+x^2+x+1 & = x^4+1 \\
\Delta_{1,5} & = & (x+d_5)\Delta_{1,4} + \Delta_{1,3} & = & (x+0)(x^4+1)+x^3+x+1 & = x^5+x^3+1
\end{array}
\tag{6.1}
$$

Now consider the repeated application of the division algorithm, as described above.

dividend	divisor	quotient	remainder	cell rule
x^5+x^3+1	x^4+1	x	x^3+x+1	0
x^4+1	x^3+x+1	x	x^2+x+1	0
x^3+x+1	x^2+x+1	$x+1$	x	1
x^2+x+1	x	$x+1$	1	1
x	1	x	0	0

(6.2)

□

Note that that the divisor column is the same as the dividend column shifted up one position, and the remainder column is a shift of the divisor column. Comparing calculation (6.2) to the CA characteristic polynomial calculation (6.1), the sequence

of polynomials in (6.2) is the reverse of the sequence of intermediate polynomials in (6.1). Furthermore, calculation (6.2) yields the sequence of quotients

$$[x, x, x+1, x+1, x].$$

By taking the constant terms of these quotients and reversing, the vector

$$[0, 1, 1, 0, 0]$$

is obtained, which is the rule vector of the CA, $d = [d_1, d_2, d_3, d_4, d_5]$.

Thus, Δ_n and Δ_{n-1} determine the whole CA, as well as all of the lower degree polynomials in the CA polynomial calculation. The procedure in (6.3), stated in general terms, is

$$\begin{array}{lcllll}
\Delta_n & = & (x+d_n) & \Delta_{n-1} & + & \Delta_{n-2} \\
\Delta_{n-1} & = & (x+d_{n-1}) & \Delta_{n-2} & + & \Delta_{n-3} \\
\Delta_{n-2} & = & (x+d_{n-2}) & \Delta_{n-3} & + & \Delta_{n-4} \\
 & \vdots & & & & \\
\Delta_4 & = & (x+d_4) & \Delta_3 & + & \Delta_2 \\
\Delta_3 & = & (x+d_3) & \Delta_2 & + & \Delta_1 \\
\Delta_2 & = & (x+d_2) & \Delta_1 & + & 1 \\
\Delta_1 & = & (x+d_1) & 1 & + & 0.
\end{array} \tag{6.3}$$

The computation as a whole is Euclid's GCD algorithm applied to the polynomials Δ_n and Δ_{n-1}, and the division algorithm ensures that this computation is unique. The calculation is *worst-case* for the GCD algorithm, in that it requires the largest possible number of applications of the division algorithm (n for the GCD of a degree n polynomial and a degree $n-1$ polynomial). Typically, computing the GCD of arbitrary degree n and degree $n-1$ polynomials requires less than n applications of the division algorithm. The calculated GCD is the last non-zero remainder. For a polynomial-subpolynomial pair, the GCD is always 1, since the CA recurrence starts at 1. Furthermore, the computation provides the CA rule vector d, formed by the constant terms of the quotients.

6.3 Sum of subpolynomials

This section derives a theorem that is central to the synthesis of CA, that describes the sum of the two subpolynomials $\Delta_{1,n-1}$ and $\Delta_{2,n}$. Starting from the Cyclic CA Theorem and results from Chapter 5, the result is established.

Let w denote the weight of the rule vector of a CA (i.e. the number of cells with rule 150). The Cyclic CA Theorem (Theorem 3.28), with a slight change in notation, states that

$$\Gamma_{1,n} = \begin{cases} q^2 & \text{if } n \text{ even and } w \text{ even} \\ (x)(x+1)\,q^2 & \text{if } n \text{ even and } w \text{ odd} \\ (x)\,q^2 & \text{if } n \text{ odd and } w \text{ even} \\ (x+1)\,q^2 & \text{if } n \text{ odd and } w \text{ odd} \end{cases}, \tag{6.4}$$

for some polynomial q. Recall also the relationship between the characteristic polynomials of a cyclic CA and its underlying null-boundary CA:

$$\Gamma_{1,n} = \Delta_{1,n} + \Delta_{2,n-1}.$$

This means that the Cyclic CA Theorem is describing a property of the sum of the polynomials $\Delta_{1,n}$ and $\Delta_{2,n-1}$. This is used to derive a similar result about the sum of $\Delta_{1,n-1}$ and $\Delta_{2,n}$.

The following table summarises some results from section 5.3. These results describe the characteristic polynomial of the CA obtained by changing the rule(s) used in one or both of the end cells. The rule vectors of the machines and their characteristic polynomials (stated in terms of subpolynomials of the original machine) are:

case	rule vector	characteristic polynomial
original	$[d_1, d_2, \ldots, d_{n-1}, d_n]$	$\Delta_{1,n}$
complement d_n	$[d_1, d_2, \ldots, d_{n-1}, d_n + 1]$	$\Delta_{1,n} + \Delta_{1,n-1}$
complement d_1	$[d_1 + 1, d_2, \ldots, d_{n-1}, d_n]$	$\Delta_{1,n} + \Delta_{2,n}$
complement both	$[d_1 + 1, d_2, \ldots, d_{n-1}, d_n + 1]$	$\Delta_{1,n} + \Delta_{1,n-1} + \Delta_{2,n} + \Delta_{2,n-1}$.

(6.5)

The main theorem, which describes the sum of the subpolynomials $\Delta_{1,n-1}$ and $\Delta_{2,n}$, is now presented.

Theorem 6.3 *For any CA,*

$$\Delta_{1,n-1}+\Delta_{2,n}=\begin{cases} & (x^2+x)\Delta'_{1,n} & n \text{ even and } w \text{ even} \\ \Delta_{1,n} + & (x^2+x)\Delta'_{1,n} & n \text{ even and } w \text{ odd} \\ (x+1)\Delta_{1,n} + & (x^2+x)\Delta'_{1,n} & n \text{ odd and } w \text{ even} \\ x\Delta_{1,n} + & (x^2+x)\Delta'_{1,n} & n \text{ odd and } w \text{ odd} \end{cases}.$$

Proof. There are four cases to be considered, depending on whether n and w are even or odd. The first of these is shown in detail, and the remaining three are condensed. For convenience, f denotes the the sum of $\Delta_{1,n-1}$ and $\Delta_{2,n}$.

Case 1: n even, w even

Suppose that n is even and w is even. By (6.4), for some polynomial q_1,

$$\Delta_{1,n}+\Delta_{2,n-1}=q_1^2.$$

If d_n is complemented, the characteristic polynomial becomes $\Delta_{1,n}+\Delta_{1,n-1}$. However, $d_2,\ldots,d_{n-1}$ are unchanged and the length is still even, but the weight has become odd. Hence, by (6.4) and (6.5),

$$\Delta_{1,n}+\Delta_{1,n-1}+\Delta_{2,n-1}=(x^2+x)q_2^2 \tag{6.6}$$

for some q_2. Similarly, if d_1 is complemented,

$$\Delta_{1,n}+\Delta_{2,n}+\Delta_{2,n-1}=(x^2+x)q_3^2 \tag{6.7}$$

for some q_3. If both d_1 and d_n are complemented, the characteristic polynomial becomes $\Delta_{1,n}+\Delta_{1,n-1}+\Delta_{2,n}+\Delta_{2,n-1}$. This CA has even length and even weight, and so, for some q_4,

$$\begin{aligned} & \Delta_{1,n}+\Delta_{1,n-1}+\Delta_{2,n}+\Delta_{2,n-1}+\Delta_{2,n-1} = q_4^2 \\ \Rightarrow \quad & \Delta_{1,n}+\Delta_{1,n-1}+\Delta_{2,n} = q_4^2. \end{aligned} \tag{6.8}$$

By adding (6.6) and (6.7),

$$f=\Delta_{1,n-1}+\Delta_{2,n}=(x^2+x)(q_2+q_3)^2.$$

Differentiating both sides,

$$f'=(q_2+q_3)^2,$$

and so

$$f = (x^2 + x)f'. \tag{6.9}$$

Equation (6.8) says that $\Delta_{1,n} + f$ is a perfect square, and hence that

$$\Delta'_{1,n} = f'. \tag{6.10}$$

Combining (6.9) and (6.10),

$$f = \Delta_{1,n-1} + \Delta_{2,n} = (x^2 + x)\Delta'_{1,n}.$$

Hence an expression for $\Delta_{1,n-1} + \Delta_{2,n}$ has been derived in terms of $\Delta_{1,n}$, for n and w both even.

Case 2: n even, w odd

In this case, the relevant equations are:

$$\begin{aligned}
\Delta_{1,n} + \Delta_{2,n-1} &= (x^2 + x)q_1^2 \\
\Delta_{1,n} + \Delta_{1,n-1} + \Delta_{2,n-1} &= q_2^2 \\
\Delta_{1,n} + \Delta_{2,n} + \Delta_{2,n-1} &= q_3^2 \\
\Delta_{1,n} + \Delta_{1,n-1} + \Delta_{2,n} &= (x^2 + x)q_4^2.
\end{aligned}$$

By adding the second and third equations,

$$f' = 0.$$

By differentiating the fourth,

$$\begin{aligned}
\Delta'_{1,n} + f' &= q_4^2 \\
\Rightarrow \quad \Delta_{1,n} + f &= (x^2 + x)(\Delta'_{1,n} + f') \\
\Rightarrow \quad f &= \Delta_{1,n} + (x^2 + x)\Delta'_{1,n}.
\end{aligned}$$

Case 3: n odd, w even

In this case, the relevant equations are:

$$\Delta_{1,n} + \Delta_{2,n-1} = xq_1^2$$

$$\begin{aligned}
\Delta_{1,n} + \Delta_{1,n-1} + \Delta_{2,n-1} &= (x+1)q_2^2 \\
\Delta_{1,n} + \Delta_{2,n} + \Delta_{2,n-1} &= (x+1)q_3^2 \\
\Delta_{1,n} + \Delta_{1,n-1} + \Delta_{2,n} &= xq_4^2.
\end{aligned}$$

By adding the second and third equations,

$$\begin{aligned}
f &= (x+1)(q_2+q_3)^2 \\
\Rightarrow\ f' &= (q_2+q_3)^2 \\
\Rightarrow\ f &= (x+1)f'.
\end{aligned}$$

Substituting this into the fourth,

$$\begin{aligned}
\Delta_{1,n} + f &= xq_4^2 \\
\Rightarrow\ \Delta'_{1,n} + f' &= q_4^2 \\
\Rightarrow\ \Delta_{1,n} + f &= x(\Delta'_{1,n} + f') \\
\Rightarrow\ &= x\Delta'_{1,n} + xf' \\
\Rightarrow\ (x+1)(\Delta_{1,n} + f) &= (x^2+x)\Delta'_{1,n} + x(x+1)f' \\
\Rightarrow\ &= (x^2+x)\Delta'_{1,n} + xf \\
\Rightarrow\ f &= (x+1)\Delta_{1,n} + (x^2+x)\Delta'_{1,n}.
\end{aligned}$$

Case 4: n odd, w odd

In this case, the relevant equations are:

$$\begin{aligned}
\Delta_{1,n} + \Delta_{2,n-1} &= (x+1)q_1^2 \\
\Delta_{1,n} + \Delta_{1,n-1} + \Delta_{2,n-1} &= xq_2^2 \\
\Delta_{1,n} + \Delta_{2,n} + \Delta_{2,n-1} &= xq_3^2 \\
\Delta_{1,n} + \Delta_{1,n-1} + \Delta_{2,n} &= (x+1)q_4^2.
\end{aligned}$$

By adding the second and third equations,

$$\begin{aligned}
f &= x(q_2+q_3)^2 \\
\Rightarrow\ f' &= (q_2+q_3)^2 \\
\Rightarrow\ f &= xf'
\end{aligned}$$

and combining with the fourth

$$\begin{array}{rrcl} & \Delta_{1,n} + f & = & (x+1)q_4^2 \\ \Rightarrow & \Delta'_{1,n} + f' & = & q_4^2 \\ \Rightarrow & \Delta_{1,n} + f & = & (x+1)(\Delta'_{1,n} + f') \\ \Rightarrow & & = & (x+1)\Delta'_{1,n} + (x+1)f' \\ \Rightarrow & x(\Delta_{1,n} + f) & = & (x^2+x)\Delta'_{1,n} + (x+1)xf' \\ \Rightarrow & & = & (x^2+x)\Delta'_{1,n} + (x+1)f \\ \Rightarrow & f & = & x\Delta_{1,n} + (x^2+x)\Delta'_{1,n}. \end{array}$$

This completes the proof. □

A simple corollary to Theorem 6.3 is as follows. It turns out that this corollary is sufficient for the derivation of the synthesis algorithm.

Corollary 6.4 *For any CA,*

$$\Delta_{1,n-1} + \Delta_{2,n} = (x^2+x)\Delta'_{1,n} \bmod \Delta_{1,n}.$$

It is convenient to make an observation about the "degenerate" case of a palindromic CA:

Corollary 6.5 *A CA is a palindrome if and only if* $\Delta_{1,n-1} = \Delta_{2,n}$.

Proof. Let w be the parity of the weight of a CA.

Suppose n is even. By Theorem 5.7 and Theorem 5.9, a CA is a palindrome if and only if its characteristic polynomial $\Delta_{1,n}$ is a perfect square. Now, $\Delta_{1,n}$ is a perfect square if and only if $\Delta'_{1,n} = 0$. Applying Theorem 6.3 (noting that w is even), $\Delta'_{1,n} = 0$ if and only if $\Delta_{1,n-1} = \Delta_{2,n}$.

Now, suppose that n is odd. By Theorem 5.7 and Theorem 5.9, a CA is a palindrome if and only if $\Delta_{1,n} = (x+w)q$, where q is a perfect square. This is so if and only if $(x+w)\Delta_{1,n}$ is a perfect square, which is true if and only if $\Delta_{1,n} + (x+w)\Delta'_{1,n}$ is 0. This holds if and only if $(x+w+1)\Delta_{1,n} + (x^2+x)\Delta'_{1,n}$ is 0. By Theorem 6.3, this is 0 if and only if $\Delta_{1,n-1} = \Delta_{2,n}$. □

With Corollary 6.5, it can be assumed that $\Delta_{1,n-1} + \Delta_{2,n} \neq 0$ when an irreducible CA is under consideration. This is useful in the following sections.

Figure 6.2: The 1-1 correspondence between polynomial pairs and CA

$$\begin{array}{lll} x^6+x^5+x^4+x^3+1, & x^5+x^4+x+1 & \longleftrightarrow [001000] \\ x^6+x^5+x^4+x^3+1, & x^5+x^4+x & \longleftrightarrow [000100] \\ x^6+x^5+x^4+x^3+1, & x^5 & \longleftrightarrow [110111] \\ x^6+x^5+x^4+x^3+1, & x^5+1 & \longleftrightarrow [111011] \end{array}$$

6.4 The quadratic

This section uses the results of the previous section to derive a necessary condition for a polynomial to be a CA polynomial. It also derives the *CA congruence*, which leads directly the the solution of the synthesis problem.

The previous section showed that the pair of polynomials Δ_n and Δ_{n-1} uniquely determine the CA, via Euclid's GCD algorithm. Conversely, a CA $[d_1, d_2, \ldots, d_n]$ uniquely determines Δ_n and Δ_{n-1}. This is why it is desirable to distinguish reversals: $\Delta_{1,n-1}$ and $\Delta_{2,n}$ play an important role, and, except for the degenerate symmetric case, are different polynomials. This gives the following lemma.

Lemma 6.6 *CA polynomial-subpolynomial pairs are in one-to-one correspondence with CA.*

Figure 6.2 demonstrates this correspondence (compare to Figure 6.1).

It has been shown already that a degree n polynomial p might not be a CA polynomial. But supposing that it is, how can it be determined if p and a given polynomial q form a CA polynomial-subpolynomial pair? Fortunately, this is easy. First of all, q must be degree $n-1$, since it is the characteristic polynomial of a length $n-1$ CA. Secondly, the correspondence between Euclid's GCD algorithm and the computation of a CA characteristic polynomial provides a simple method of determination, described in the following lemma.

Lemma 6.7 *Let p be a degree n polynomial, and let q be a degree $n-1$ polynomial. Then there exists a CA with characteristic polynomial p and characteristic subpoly-*

nomial q (i.e. with $\Delta_n = p$ and $\Delta_{n-1} = q$) if and only if applying Euclid's GCD algorithm to p and q results in n degree 1 quotients.

Proof. Suppose there exists a CA with $\Delta_n = p$ and $\Delta_{n-1} = q$. The division algorithm ensures that the sequence of remainder polynomials obtained from Euclid's algorithm is $\Delta_{n-2}, \Delta_{n-3}, \ldots, \Delta_2, \Delta_1, 1, 0$. As these polynomials have degrees differing by 1, Euclid's algorithm produces n degree 1 quotients. For the converse, suppose that Euclid's algorithm is applied to a degree n polynomial p and degree $n-1$ polynomial q, and that n degree 1 quotients result, $x+d_n, x+d_{n-1}, \ldots, x+d_1$. The last remainder is always 0, and the fact that there are n quotients means that the last divisor (second last remainder) is 1. Hence the algorithm terminates with the base cases of the CA recurrence. Clearly, if the CA recurrence is applied to $[d_1, d_2, \ldots, d_n]$, it calculates the polynomials $\Delta_n = p$ and $\Delta_{n-1} = q$. Hence there exists a CA with the desired properties. □

The constant terms of the n degree 1 quotients specify the CA, as in Example 6.2.

There is an obvious algorithm for finding a CA for p, based on this lemma:

Algorithm 6.8

for every degree $n-1$ polynomial q

if GCD(p,q) yields n degree 1 quotients (Lemma 6.7), then a CA has been found.

However, there are 2^{n-1} possibilities for q, and so this algorithm does not perform much better than just enumerating all CA, computing their characteristic polynomials, and checking for p. It is clear now, though, that finding a q that satisfies Lemma 6.7 would solve the CA synthesis problem, if it could be done efficiently.

At this point, the result central to the synthesis is derived. It shows that the degree $n-1$ polynomial Δ_{n-1} satisfies an expression called the *CA quadratic congruence.*

The usefulness of the result is discussed below, and its application in the synthesis algorithm is described in section 6.5.

The GCD relation from Chapter 3 (Theorem 3.6) describes the product of $\Delta_{1,n-1}$ and $\Delta_{2,n}$:

$$\Delta_{1,n-1}\Delta_{2,n} = \Delta_{1,n}\Delta_{2,n-1} + 1.$$

Suppose that $\Delta_{1,n}$ is irreducible, and has α as a root. The function $\phi(p(x)) = p(\alpha)$ is a homomorphism from GF(2)[x] onto GF(2^n). Let $m_{i,j}$ denote $\phi(\Delta_{i,j})$. If ϕ is applied to Theorem 3.6, then

$$m_{1,n-1}m_{2,n} = 1, \tag{6.11}$$

since $m_{1,n}$ is $\Delta_{1,n}$ evaluated at its own root, and so is 0 (note that $m_{1,n-1}$ and $m_{2,n}$ are distinct; otherwise, the CA is a palindrome and thus has a reducible characteristic polynomial). Corollary 6.4 states that

$$\Delta_{1,n-1} + \Delta_{2,n} = (x^2 + x)\Delta'_{1,n} \bmod \Delta_{1,n}.$$

Under the homomorphism, this becomes

$$m_{1,n-1} + m_{2,n} = (\alpha^2 + \alpha)m'_{1,n}, \tag{6.12}$$

where $m'_{1,n}$ is the derivative of $\Delta_{1,n}$, evaluated at α. Substituting (6.12) into (6.11),

$$m_{1,n-1}(m_{1,n-1} + (\alpha^2 + \alpha)m'_{1,n}) = 1,$$

which implies that

$${m_{1,n-1}}^2 + (\alpha^2 + \alpha)m'_{1,n}m_{1,n-1} + 1 = 0.$$

This means that $m_{1,n-1}$ is a root of the quadratic

$$y^2 + (\alpha^2 + \alpha)m'_{1,n}y + 1 = 0. \tag{6.13}$$

Likewise, $m_{2,n}$ is also a root of (6.13). This gives the following necessary condition for an irreducible polynomial to have a CA.

Theorem 6.9 *Let $p \in GF(2)[x]$ be irreducible over $GF(2)$, with root α in $GF(2^n)$. Define the homomorphism ϕ as evaluation at α. If p is the characteristic polynomial of a CA, then the following hold:*

1. *the quadratic $y^2+(\alpha^2+\alpha)m'_{1,n}y+1$ in $GF(2^n)$ has two distinct roots in $GF(2^n)$, say β_1 and β_2,*
2. *β_1 and β_2 have degree $n-1$ pre-images in $GF(2)[x]$ (that is, $\phi^{-1}(\beta_i), i = 1, 2$, contains an element of degree $n-1$), say p_1 and p_2,*
3. *calculating the GCD of p and $p_i, i = 1, 2$, requires n applications of the division algorithm,*
4. *each application of the division algorithm has a degree 1 quotient $x + d_j, j = 1, \ldots, n$, and the vector formed with the d_j is the rule vector of a CA for p,*
5. *p has exactly two CA, which are reversals of each other.*

The theorem is demonstrated on the CA [00110], for which the characteristic polynomial is $\Delta_5 = x^5 + x^3 + 1$ (refer to Example 6.2). For this CA, $\Delta_{1,4} = x^4 + 1$ and $\Delta_{2,5} = x^4 + x$. Substituting $y = \alpha^4 + 1$ into the left hand side of (6.13),

$$\begin{aligned} & y^2 + (\alpha^2 + \alpha)m'_{1,n}y + 1 \\ = \; & (\alpha^4 + 1)^2 + (\alpha^2 + \alpha) \cdot (\alpha^4 + \alpha^2) \cdot (\alpha^4 + 1) + 1 \\ = \; & (\alpha^8 + 1) + (\alpha^{10} + \alpha^9 + \alpha^8 + \alpha^7 + \alpha^6 + \alpha^5 + \alpha^4 + \alpha^3) + 1 \\ = \; & \alpha^{10} + \alpha^9 + \alpha^7 + \alpha^6 + \alpha^5 + \alpha^4 + \alpha^3 \\ = \; & 0. \end{aligned}$$

Similarly, substituting $y = \alpha^4 + \alpha$ into (6.13) results in 0.

By combining Lemma 6.7 and Theorem 6.9, a characterisation is obtained for CA polynomials.

Corollary 6.10 (CA quadratic congruence) *Let p be a degree n polynomial. Then p is a CA polynomial if and only if for some solution q for y of the congruence*

$$y^2 + (x^2 + x)p'y + 1 \equiv 0 \bmod p, \tag{6.14}$$

Euclid's GCD algorithm results in n degree 1 quotients when applied to p and q.

The algorithm for finding a CA for a given polynomial p can now be refined to:

Algorithm 6.11

for each solution $q = y$ to $y^2 + (x^2 + x)p'y + 1 \equiv 0 \bmod p$ *(step 1)*

if GCD(p,q) yields n degree 1 quotients, then a CA has been found. *(step 2)*

If none of the solutions to (6.14) are successful in step 2, then p is not a CA polynomial. Also, then, if (6.14) has no solutions at all, then p is not a CA polynomial.

Theorem 6.9 is lacking in two respects. First, it does not say that all polynomials q that are solutions to (6.14) are subpolynomials of $\Delta_{1,n}$. Second, it does not say that non-CA polynomials do not have solutions to (6.14). In these respects, the theorem provides only a necessary condition for CA polynomials: that they have solutions to (6.14), and that some of these solutions are subpolynomials.

In their paper concerning the behaviour of Euclid's GCD algorithm on polynomials over GF(2) ([47]), Mesirov and Sweet prove that for any irreducible polynomial p, there exists a polynomial w such that p and w have a worst-case GCD calculation. The following theorem follows immediately from this.

Theorem 6.12 *If p is a degree n irreducible polynomial, then (6.14) has exactly two solutions, both of which result in n degree 1 quotients.*

This gives the following corollary.

Corollary 6.13 *If p is an irreducible polynomial, then p has exactly two CA realisations.*

This solves the long-outstanding problem of *the existence of CA for irreducible polynomials.* Another fascinating consequence of their work is that a reducible polynomial p has either no CA realisation or 2^m CA realisations, where m is the number of distinct irreducible factors of p different from x and $x+1$.

As discussed below, when $\Delta_{1,n}$ is irreducible, solutions to (6.14) can be found elegantly. When $\Delta_{1,n}$ is reducible, solutions can again be obtained quickly, but they may be extremely numerous. For example, the reducible polynomial x^6+1 has eight solutions to (6.14), four of which are degree $n-1$, but none are CA subpolynomials. Hence x^6+1 is not a CA polynomial.

This section has shown that to compute a CA for a polynomial p, it is necessary to find solutions to the congruence (6.14). Once again, the brute force search technique has exponential time complexity. However, there is an efficient technique for finding solutions, which is presented in the next section.

6.5 Solving quadratics

To find the CA for a given irreducible polynomial p, it is required that the quadratic (6.14) in GF(2^n) be solved. Unfortunately, the standard quadratic formula does not hold in fields of characteristic 2, and so other techniques must be used. The theory in this section is based on material from [7] and [45].

To simplify derivations, let c denote $(\alpha^2+\alpha)p'$, and assume c is nonzero (if c is 0, then p is reducible). Thus (6.14) is rewritten as

$$y^2 + cy + 1 = 0. \tag{6.15}$$

By making the change of variable $y = 1/(zc)$, (6.15) becomes

$$\begin{aligned} & (1/zc)^2 + c/zc + 1 &&= 0 \\ \Leftrightarrow \quad & z^2 + z + 1/c^2 &&= 0. \end{aligned} \tag{6.16}$$

Let γ denote $1/c^2$. This quadratic has roots in GF(2^n) if and only if the trace of γ (= trace of $1/c$) is 0. Thus, interestingly, the trace of γ is always 0.

Let θ be an element of GF(2^n) with trace 1. Such an element always exists, since one half of the elements of GF(2^n) have trace 0, and one half have trace 1. Define β by

$$\begin{aligned}\beta &= \gamma\theta^2 + (\gamma+\gamma^2)\theta^{2^2} + \cdots + (\gamma+\gamma^2+\cdots+\gamma^{2^{n-2}})\theta^{2^{n-1}}. \\ &= \sum_{i=1}^{n-1}\left(\sum_{j=0}^{i-1}\gamma^{2^j}\right)\theta^{2^i}. \end{aligned} \tag{6.17}$$

Then the roots of (6.16) are $z = \beta$ and $z = \beta + 1$. Undoing the change of variable, the roots of (6.15) are

$$y = 1/(\beta c) \text{ and } y = 1/((\beta+1)c).$$

Also, since the roots of (6.15) are multiplicative inverses, they are

$$y = 1/(\beta c) \text{ and } y = \beta c.$$

Note that if n is odd, then the trace of 1 is 1, and so $\theta = 1$ suffices. This simplifies β to

$$\beta = \gamma^2 + \gamma^8 + \cdots + \gamma^{2^{n-2}}. \tag{6.18}$$

This completes the description of solving quadratics in GF(2^n).

6.6 An example

This section contains a complete example of computing the CA for a degree five irreducible polynomial. It shows the determination of the solutions of the CA quadratic congruence, and the subsequent application of Euclid's GCD algorithm.

Let p be the irreducible (in this case, primitive) polynomial

$$p = x^5 + x^2 + 1$$

(hence $n = 5$). To start with, the formal derivative of p is calculated, with the coefficients reduced modulo 2:

$$p' = 5x^4 + 2x = x^4.$$

Now c is calculated, reduced modulo p, as

$$c = (x^2 + x)p' = x^6 + x^5 = x^3 + x^2 + x + 1.$$

Applying the extended Euclidean GCD algorithm to compute the inverse of c gives

$$1/c = x^3 + x^2 + 1.$$

To obtain g, calculate

$$g = (1/f)^2 = x^6 + x^4 + 1 = x^4 + x^3 + x + 1.$$

The trace of g can be verified to be 0, as expected. Since n is odd, formula (6.18) is used to compute β:

$$\begin{aligned} \beta &= \sum_{i=1}^{2} g^{2^{2i-1}} \\ &= g^2 + g^8. \end{aligned}$$

Computing powers of g,

$$\begin{aligned} g &= x^4 + x^3 + x + 1 \\ g^2 &= (x^4 + x^3 + x + 1)^2 = x^8 + x^6 + x^2 + 1 = x \\ g^4 &= (g^2)^2 = x^2 \\ g^8 &= (g^4)^2 = x^4, \end{aligned}$$

and so

$$\begin{aligned} \beta &= g^2 + g^8 \\ &= x^4 + x. \end{aligned}$$

Finally, a solution q to the quadratic is computed:

$$\begin{aligned} q &= (x^2 + x)p'\beta \\ &= (x^2 + x)(x^4)(x^4 + x) \\ &= x^4 + x^2 + 1. \end{aligned}$$

Euclid's algorithm is applied to determine the CA. At this point, q is Δ_{n-1}.

$$\begin{aligned} x^5+x^2+1 &= (x) \quad (x^4+x^2+1) &+ x^3+x^2+x+1 \\ x^4+x^2+1 &= (x+1)(x^3+x^2+x+1) &+ x^2 \\ x^3+x^2+x+1 &= (x+1)(x^2) &+ x+1 \\ x^2 &= (x+1)(x+1) &+ 1 \\ x+1 &= (x+1)(1) &+ 0 \end{aligned}$$

The algorithm results in the five degree 1 quotients $[x, x+1, x+1, x+1, x+1]$, and so the CA has rule vector [1,1,1,1,0].

6.7 Implementation and performance

The algorithm has been implemented in the Maple V programming language (see Appendix G). This section presents some timing results from the program, and shows the total number of required operations.

The tables in Appendix C list a *least-weight primitive* polynomial for each degree up to 300. The primitivity of a polynomial p ensures that any LFSM with characteristic polynomial p has a maximal-length cycle, and that p is irreducible. Least-weight means that p contains the least possible number of non-zero coefficients.

Appendix C lists the CA for each of these least-weight primitive polynomials. A small subset of this table is reproduced in Table 6.1, along with the amount of CPU time needed to compute the CA (due to size, the actual CA are not repeated here for the higher degrees). The times are in seconds on a SPARC 10 workstation with 32M of memory.

The table indicates that the algorithm is sufficiently fast for practical applications. This is the case since the number of operations required is not dependent on the input polynomial, only on its degree. The largest CA synthesised to date has degree 1600 (it is not in the table), and uses about 10 cpu minutes.

Table 6.2 lists the total number of operations required for the algorithm. A *mod* p operation means that the result of the operation is reduced modulo p. These are, in effect, arithmetic operations in the finite field GF(2^n). The table shows that the

Table 6.1: Example running time of CA synthesis program

Polynomial	Time	CA
$x^{20}+x^3+1$	0.06	01101011100001010110
$x^{40}+x^{21}+x^{19}+x^2+1$	0.16	11001100000110000001000101000001001100 11
$x^{60}+x+1$	0.50	11100111101001011101000010111100110100 00
		10111010010111100111
$x^{80}+x^{38}+x^{37}+x+1$	0.90	01010110010000100000101000110011101111 01
		11101010110111011110000001000010011010 10
$x^{99}+x^{47}+x^{45}+x^2+1$	0.31	
$x^{100}+x^{37}+1$	1.19	
$x^{199}+x^{34}+1$	1.00	*omitted*
$x^{200}+x^{163}+x^2+x+1$	6.65	
$x^{299}+x^{21}+x^2+x+1$	2.25	
$x^{300}+x^7+1$	16.07	

number of mod p operations required grows linearly with the degree of the polynomial, so the method does not suffer from any sort of exponential blow-up. However, note that even degree polynomials need more operations than do odd degree, as borne out by Table 6.1.

Table 6.2: Operation count for synthesis of CA

Operation type	Operations required n even	n odd	Approx. time / op
mod p shifts	2	2	n
mod p multiplications	1	1	n^2
mod p squares	$2n-1$	$(n+1)/2$	n^2
mod p inversions	1	1	n^2
polynomial additions	$2n-2$	$(n-1)/2$	n
polynomial GCD	1	1	n^2
polynomial derivative	1	1	n

6.8 Degree-of-symmetry

An inspection of the CA that correspond to least-weight primitive polynomials reveals that virtually all of the CA are "nearly palindromic," meaning that the left half of the CA is almost equal to the reverse of the right half of the CA. In this section, a result that explains this property is derived. This is formalised in the following definition.

Definition 6.14 *Let* $d = [d_1, \ldots, d_n]$ *be a CA. Define the* degree of symmetry (DOS(d)) *of the CA to be the largest* $k \leq \lfloor \frac{n}{2} \rfloor$ *for which*

$$d_1 = d_n, d_2 = d_{n-1}, \ldots, d_k = d_{n+1-k}.$$

The following preliminary definition and lemma are deduced in order to explain why the DOS is often large.

Definition 6.15 *Let* a *and* b *be polynomials in* $GF(2)[x]$. *Define* $\mathcal{P}^l(a,b)$ *to be the largest* k *such that the first* k *divisions in the calculation of the GCD of* a *and* b *have degree 1 quotients. Let* $\mathcal{P}^s(a,b)$ *denote the sequence formed with these degree 1 quotients.* $\mathcal{P}^s(a,b)$ *is subscripted to indicate an element or a subsequence of the sequence of quotients; for example,* $\mathcal{P}^s(a,b)_i$ *is the* i*th quotient.*

The following lemma describes the effect on the GCD computation when the initial divisor is modified. It shows that either the ith quotient or the ith remainder can be controlled, by a careful selection of the initial divisor, for the first half of the computation. Even though it is not deep, the proof is quite lengthy.

Lemma 6.16 *Suppose* p_n *and* p_{n-1} *are degree* n *and* $n-1$ *respectively, and suppose* $\mathcal{P}^l(p_n, p_{n-1}) \geq k$, *for some* $1 \leq k \leq \lfloor n/2 \rfloor$. *If* c *is a degree* $n-2k-1 \geq 0$ *polynomial, then*

$$\begin{array}{lcll} \mathcal{P}^s(p_n, p_{n-1})_{1,\ldots,k} & = & \mathcal{P}^s(p_n, p_{n-1}+c)_{1,\ldots,k}, & \textit{and} \\ \mathcal{P}^l(p_n, p_{n-1}) = k & \textit{iff} & \mathcal{P}^l(p_n, p_{n-1}+c) > k. & \end{array}$$

If c *is a degree* $n-2k$ *polynomial, then*

$$\begin{array}{lcll} \mathcal{P}^s(p_n, p_{n-1})_{1,\ldots,k-1} & = & \mathcal{P}^s(p_n, p_{n-1}+c)_{1,\ldots,k-1}, & \textit{and} \\ \mathcal{P}^s(p_n, p_{n-1})_k & \neq & \mathcal{P}^s(p_n, p_{n-1}+c)_k. & \end{array}$$

Proof. Consider the polynomial remainder sequence calculation of p_n and p_{n-1}:

$$\begin{array}{lcll} p_n & = & a_n p_{n-1} & + \ p_{n-2} \\ p_{n-1} & = & a_{n-1} p_{n-2} & + \ p_{n-3} \\ p_{n-2} & = & a_{n-2} p_{n-3} & + \ p_{n-4} \\ & \vdots & & \\ p_{n-k+2} & = & a_{n-k+2} p_{n-k+1} & + \ p_{n-k} \\ p_{n-k+1} & = & a_{n-k+1} p_{n-k} & + \ r. \\ & \vdots & & \end{array} \tag{6.19}$$

Note that p_i has degree i, but the degree of the remainder r is unknown. Also, $a_n, a_{n-1}, \ldots, a_{n-k+1}$ are all of degree 1. Now consider the calculation of the characteristic polynomial of the CA with rule vector $[d_n, d_{n-1}, \ldots, d_{n-k+1}]$ (d_i is $x + a_i$):

$$\begin{array}{lcll} 0 & = & a_n 1 & + \ \Delta_{n,n} \\ 1 & = & a_{n-1} \Delta_{n,n} & + \ \Delta_{n-1,n} \\ \Delta_{n,n} & = & a_{n-2} \Delta_{n,n-1} & + \ \Delta_{n-2,n} \\ & \vdots & & \\ \Delta_{n,n-k+3} & = & a_{n-k+1} \Delta_{n,n-k+2} & + \ \Delta_{n-k+1,n}. \end{array}$$

Multiplying each equation by c,

$$\begin{array}{lcll} 0 & = & a_n c & + \ c\Delta_{n,n} \\ c & = & a_{n-1} c\Delta_{n,n} & + \ c\Delta_{n-1,n} \\ c\Delta_{n,n} & = & a_{n-2} c\Delta_{n,n-1} & + \ c\Delta_{n-2,n} \\ & \vdots & & \\ c\Delta_{n,n-k+4} & = & a_{n-k+2} c\Delta_{n,n-k+3} & + \ c\Delta_{n-k+2,n} \\ c\Delta_{n,n-k+3} & = & a_{n-k+1} c\Delta_{n,n-k+2} & + \ c\Delta_{n-k+1,n}. \end{array} \tag{6.20}$$

Adding (6.19) and (6.20) line by line,

$$\begin{array}{lcll} p_n & = & a_n(p_{n-1} + c) & + \ (p_{n-2} + c\Delta_{n,n}) \\ p_{n-1} + c & = & a_{n-1}(p_{n-2} + c\Delta_{n,n}) & + \ (p_{n-3} + c\Delta_{n-1,n}) \\ p_{n-2} + c\Delta_{n,n} & = & a_{n-2}(p_{n-3} + c\Delta_{n-1,n}) & + \ (p_{n-4} + c\Delta_{n-2,n}) \\ & \vdots & & \\ p_{n-k+2} + c\Delta_{n-k+4,n} & = & a_{n-k+2}(p_{n-k+1} + c\Delta_{n-k+3,n}) & + \ (p_{n-k} + c\Delta_{n-k+2,n}) \\ p_{n-k+1} + c\Delta_{n-k+3,n} & = & a_{n-k+1}(p_{n-k} + c\Delta_{n-k+2,n}) & + \ (r + c\Delta_{n-k+1,n}). \end{array} \tag{6.21}$$

Let $[c]$ denote the degree of c, and consider a degree analysis of (6.21):

$$\begin{array}{lcl}
[n] & = & [1]([n-1]+[c])+([n-2]+[c][1]) \\
[n-1]+[c] & = & [1]([n-2]+[c][1])+([n-3]+[c][2]) \\
[n-2]+[c][1] & = & [1]([n-3]+[c][2])+([n-4]+[c][3]) \\
 & \vdots & \\
[n-k+2]+[c][k-3] & = & [1]([n-k+1]+[c][k-2])+([n-k]+[c][k-1]) \\
[n-k+1]+[c][k-2] & = & [1]([n-k]+[c][k-1])+([r]+[c][k]).
\end{array} \tag{6.22}$$

Now, in either case, the degree of c is less than or equal to $n-2k$. For each of the first $k-1$ equations of (6.22), the terms that come from (6.20) have degrees that are less than the degrees of the terms from (6.19). Hence the first $k-1$ equations of (6.22) simplify to

$$\begin{array}{lcl}
[n] & = & [1]([n-1])+([n-2]) \\
[n-1] & = & [1]([n-2])+([n-3]) \\
[n-2]] & = & [1]([n-3])+([n-4]) \\
 & \vdots & \\
[n-k+2] & = & [1]([n-k+1])+([n-k]),
\end{array}$$

and the first $k-1$ quotients are unchanged; that is, they are still $a_n, a_{n-1}, \ldots, a_{n-k+2}$.

Now, if the degree of c is $n-2k-1$, the last equation of (6.22) is

$$[n-k+1]+[n\cdot\ k-3]=[1]([n-k]+[n-k-2])+([r]+[n-k-1]).$$

The degree of r is $n-k-1$ (which is true if and only if $\mathcal{P}^l(p_n, p_{n-1}) \neq k$) if and only if the degree of $r+c\Delta_{n-k+1,n} < n-k-1$ (which is true if and only if $\mathcal{P}^l(p_n, p_{n-1}+c) = k$).

If the degree of c is $n-2k$, the last equation of (6.22) is

$$[n-k+1]+[n-k-2]=[1]([n-k]+[n-k-1])+([r]+[n-k]).$$

The degree of the remainder is $n-k$, equal to the degree of the divisor. Hence this equation does not satisfy the division algorithm, and thus the specified quotient is incorrect. This means that the kth line of the calculation is actually

$$p_{n-k+1}+c\Delta_{n,n-k+3}=(a_{n-k+1}+1)(p_{n-k}+c\Delta_{n,n-k+2})+(r+c\Delta_{n-k+1,n}+p_{n-k}+c\Delta_{n,n-k+2}),$$

which has the required degrees. □

Example 6.17. Consider the polynomials $\Delta_n = x^8 + x^6 + x^5 + x^2 + 1$ and $\Delta_{n-1} = x^7 + x^6 + x^5 + x^2 + 1$. The GCD calculation reveals that $\mathcal{P}^l(\Delta_n, \Delta_{n-1}) = 2$:

$$\begin{aligned} x^8 + x^6 + x^5 + x^2 + 1 &= (x+1)(x^7 + x^6 + x^5 + x^3 + 1) + x^6 + x^4 + x^3 + x^2 + x \\ x^7 + x^6 + x^5 + x^3 + 1 &= (x+1)(x^6 + x^4 + x^3 + x^2 + x) + x^3 + x + 1. \end{aligned}$$

Choosing a degree $n - 2k - 1 = 3$ polynomial $c = x^3 + x^2$, and calculating the GCD of Δ_n and $\Delta_{n-1} + c$:

$$\begin{aligned} x^8 + x^6 + x^5 + x^2 + 1 &= (x+1)(x^7 + x^6 + x^5 + x^2 + 1) + x^6 + x^3 + x \\ x^7 + x^6 + x^5 + x^2 + 1 &= (x+1)(x^6 + x^3 + x) + x^5 + x^4 + x^3 + x + 1. \end{aligned}$$

As can be seen, the first two quotients are still both $x+1$, and $\mathcal{P}^l(\Delta_n, \Delta_{n-1}+c) > 2$.

Choosing a degree $n - 2k = 4$ polynomial $c = x^4 + 1$, and calculating the GCD of Δ_n and $\Delta_{n-1} + c$:

$$\begin{aligned} x^8 + x^6 + x^5 + x^2 + 1 &= (x+1)(x^7 + x^6 + x^5 + x^4 + x^3) = x^6 + x^5 + x^3 + x^2 + 1 \\ x^7 + x^6 + x^5 + x^4 + x^3 &= (x)(x^6 + x^5 + x^3 + x^2 + 1) = x^5 + x. \end{aligned}$$

The first quotient is unchanged, but the second has become x. □

Using Lemma 6.16, the symmetry in the tables can now be explained. Suppose that $\Delta_{1,n-1} + \Delta_{2,n}$ is not zero (if it is zero, then the CA is a palindrome). Let l denote the degree of $\Delta_{1,n-1} + \Delta_{2,n}$. All of the CA in the tables have even weight (the degree $n-1$ coefficient is 0), in which case the following lemma applies.

Lemma 6.18 *Let $\Delta_{1,n}$ be a degree n polynomial with CA d, and suppose that the weight of d is even. Then*

$$\begin{aligned} n \text{ even} &\Rightarrow l = 1 + \text{ highest odd degree term of } p, \\ n \text{ odd} &\Rightarrow l = 1 + \text{ highest even degree term of } p. \end{aligned}$$

Proof. Notice that $\Delta_{1,n}$ is separable into odd and even degree terms as:

$$\begin{aligned} \text{odd degree terms } p_o &= x\Delta'_{1,n} \\ \text{even degree terms } p_e &= \Delta_{1,n} + x\Delta'_{1,n} \end{aligned}$$

When both n and the weight are even, Theorem 6.3 states that

$$\begin{aligned} \Delta_{1,n-1} + \Delta_{2,n} &= (x^2 + x)\Delta'_{1,n} \\ &= (x+1)p_o, \end{aligned}$$

which has degree one greater than the highest odd degree. If n is even and the weight is odd, then

$$\begin{aligned}\Delta_{1,n-1} + \Delta_{2,n} &= (x+1)\Delta_{1,n} + (x^2+x)\Delta'_{1,n} \\ &= (x+1)p_e,\end{aligned}$$

which has degree one greater than the highest even degree. □

Using this, observe that most of the polynomials in Appendix C have a small value of l. The degree of symmetry is linked to l by the following theorem.

Theorem 6.19 *Let d be a CA, with characteristic polynomial $\Delta_{1,n}$ and characteristic subpolynomials $\Delta_{1,n-1}$ and $\Delta_{2,n}$. Then*

$$DOS(d) = \frac{n-l}{2} - 1.$$

Proof. Let $c = \Delta_{1,n-1} + \Delta_{2,n}$ have degree l. Let $k = (n-l)/2$, so that $l = n - 2k$. Applying Lemma 6.16 with k and c as defined (note that $\mathcal{P}^l(\Delta_n, \Delta_{n-1}) = n$), it is the case that

$$\begin{aligned}\mathcal{P}^s(\Delta_n, \Delta_{1,n-1})_{1,\ldots,k-1} &= \mathcal{P}^s(\Delta_n, \Delta_{2,n})_{1,\ldots,k-1} \quad \text{and} \\ \mathcal{P}^s(\Delta_n, n)_{\Delta_{1,n-1}} &\neq \mathcal{P}^s(\Delta_n, \Delta_{2,n})_k.\end{aligned}$$

Hence the DOS(d) is $k - 1 = (n-l)/2 - 1$. □

Thus a small l implies that the DOS is large (for even weight CA), which explains why many of the CA listed are "nearly palindromic."

6.9 Conclusion

This chapter relates the CA characteristic polynomial calculation to Euclid's GCD algorithm, which provides the framework for the synthesis of CA for irreducible polynomials. The framework culminates in the CA quadratic congruence. The synthesis algorithm is described in detail, and a full example is presented. This chapter also shows that an irreducible polynomial has two CA realisations. The performance of the algorithm is discussed, and partial symmetry in CA rule vectors is analysed.

Chapter 7

Synthesis of CA over GF(q)

This chapter explores issues surrounding the synthesis of CA over a general finite field GF(q). A probabilistic algorithm for the synthesis is given, with an approximation to the probability of success. This approximation is compared to actual empirical tests. A more complicated, but significantly better, probabilistic algorithm is derived. In section 7.1, the connection between the computation of the CA characteristic polynomial and Euclid's GCD algorithm is shown to hold over GF(q). Section 7.2 presents a probabilistic approach to the synthesis problem, and explores the probability of success of the algorithm. In section 7.3 it is shown that there is structure in the sequences of remainders obtained from the GCD algorithm when applied to a fixed polynomial and all polynomials of lesser degree. Section 7.4 utilises this structure to derive a synthesis algorithm that has a probability of success that is the square root of the algorithm in section 7.2.

7.1 Background

The synthesis of CA over GF(q) is, depending on the viewpoint, both harder and easier than it is for GF(2). It is harder in that the CA quadratic congruence does not seem to have a generalisation in GF(q). The product of the degree $n-1$ polynomials $\Delta_{1,n-1}$ and $\Delta_{2,n}$ is known (Theorem 4.7), but their sum does not seem to be directly predictable.

The synthesis problem is easier in the sense that as q increases, the number of CA grows much more rapidly than the number of polynomials. For a "guess and check" synthesis approach this is not relevant. However, for a GCD-based approach, it means that a probabilistic method can work well. To take an extreme, the very special worst-case-GCD-computation for GF(2) is commonly referred to as the "normal case" for real-coefficient polynomials.

As in the GF(2) case, the computation of the characteristic polynomial is intimately related to Euclid's GCD algorithm, as is shown below. Recall the GF(q) CA recurrence in Theorem 4.2,

$$\Delta_k = (x - d_k)\Delta_{k-1} - b_{k-1}c_k\Delta_{k-2}.$$

Suppose that Δ_k and Δ_{k-1} are known. The division algorithm for polynomials over GF(q) states that there exist unique polynomials s and r satisfying:

$$\Delta_k = s\Delta_{k-1} + r, \quad \deg r < \deg \Delta_{k-1} = k - 1. \tag{7.1}$$

Since the degree of $-b_{k-1}c_k\Delta_{k-2}$ is $k-2$, which is less than $k-1$, the division algorithm implies that

$$\begin{aligned} s &= (x - d_k) \text{ and} \\ r &= -b_{k-1}c_k\Delta_{k-2} \end{aligned}$$

are the unique solutions to (7.1). Hence Δ_k and Δ_{k-1} uniquely determine $(x - d_k)$ and $-b_{k-1}c_k\Delta_{k-2}$. If the division algorithm is then applied to Δ_{k-1} and Δ_{k-2}, the quotient $x+d_{k-1}$ and remainder $-b_{k-2}c_{k-1}\Delta_{k-3}$ are uniquely determined. Continuing this process is equivalent to calculating the GCD of Δ_k and Δ_{k-1}, and is the basis for the synthesis. It is actually a *normalised GCD calculation*, as the divisors are always monic.

Example 7.1. Consider a 3-cell CA over GF(4) with transition matrix

$$A = \begin{bmatrix} \beta & 1 & 0 \\ 1 & 0 & \alpha \\ 0 & \alpha & 1 \end{bmatrix}.$$

The characteristic polynomial calculation is

$$\begin{aligned}
\Delta_{-1} &= 0, \\
\Delta_0 &= 1, \\
\Delta_1 &= (x-\beta)\cdot 1 + 0 = x+\beta, \\
\Delta_2 &= (x-0)(x+\beta) - 1\cdot 1\cdot 1 = x^2+\beta x+1, \\
\Delta_3 &= (x-1)(x^2+\beta x+1) - \alpha\cdot\alpha\cdot(x+\beta), \\
&= x^3+\alpha x^2+x+\beta,
\end{aligned}$$

so $\Delta_3 = x^3+\alpha x^2+x+\beta$ and $\Delta_2 = x^2+\beta x+1$. The GCD calculation of Δ_3 and Δ_2 follows. Note that the remainder is normalised after each division.

i	dividend	divisor	quotient	remainder	d_i	$-b_{i-1}c_i$
3	$x^3+\alpha x^2+x+\beta$	$x^2+\beta x+1$	$(x-1)$	$-\alpha\cdot\alpha\cdot(x+\beta)$	1	β
2	$x^2+\beta x+1$	$x+\beta$	x	$-1\cdot 1\cdot 1$	0	1
1	$x+\beta$	1	$x+\beta$	0	β	0

(7.2)

□

The (normalised) GCD procedure for calculating the CA from the polynomials $\Delta_{1,n}$ and $\Delta_{1,n-1}$ is given explicitly in Algorithm 7.2.

Algorithm 7.2

procedure **CalcCAFromPolySubpolyPair**

input: *degree n polynomial Δ_n and degree $n-1$ polynomial Δ_{n-1}*

output: *sequences (d_i), (b_i) and (c_i), $i = 1,\ldots,n$.*

$c_1 \leftarrow 1; b_n \leftarrow 1$ *(arbitrary)*

$k \leftarrow n$

repeat

$a_k \leftarrow \Delta_k \operatorname{div} \Delta_{k-1}$

$d_k \leftarrow x - a_k$

if $k = 1$ then return (d_i), (b_i) and (c_i), $i = 1,\ldots,n$

$r \leftarrow \Delta_k \bmod \Delta_{k-1}$

choose b_{k-1} and c_k such that $-b_{k-1}c_k = \operatorname{lcoeff}(r)$

$\Delta_{k-2} \leftarrow \operatorname{normal}(r)$

$k \leftarrow k-1$

end repeat

Note that the multipliers b_{k-1} and c_k are not specified uniquely; it is only their product that is determined. The functions lcoeff and normal return the leading coefficient of the polynomial and the polynomial divided by its leading coefficient, respectively.

Example 7.3. Procedure CalcCAFromPolySubpolyPair is applied to the CA of Example 4.1 (the field is GF(3)). On entry, $\Delta_3 = x^3 + x^2 + 2x + 1$ and $\Delta_2 = x^2$.

initialisation	$k \leftarrow 3$
	choose $b_3 = c_1 = 2$
iteration 1	$a_3 \leftarrow \Delta_3 \operatorname{div} \Delta_2 = x + 1$
	$d_3 \leftarrow x - a_3 = 2$
	$r \leftarrow \Delta_3 \bmod \Delta_2 = 2x + 1$
	$-b_2 c_3 = 2 \Rightarrow b_2 c_3 = 1$; choose $b_2 = c_3 = 1$
	$\Delta_1 \leftarrow \text{normal}(r) = (1/2)(2x + 1) = x + 2$
	$k \leftarrow 2$
iteration 2	$a_2 \leftarrow \Delta_2 \operatorname{div} \Delta_1 = x + 1$
	$d_2 \leftarrow x - a_2 = 2$
	$r \leftarrow \Delta_2 \bmod \Delta_1 = 1$
	$-b_1 c_2 = 1 \Rightarrow b_1 c_2 = 2$; choose $b_1 = 2,\ c_2 = 1$
	$\Delta_0 \leftarrow \text{normal}(r) = 1$
	$k \leftarrow 1$
iteration 3	$a_1 = \Delta_1 \operatorname{div} \Delta_0 = x + 2$
	$d_1 \leftarrow x - a_1 = 1$
	stop

Hence the calculated CA has transition matrix

$$A = \begin{bmatrix} 1 & 2 & 0 \\ 1 & 2 & 2 \\ 0 & 2 & 2 \end{bmatrix}.$$

□

Procedure CalcCAFromPolySubpolyPair requires that the input polynomials be Δ_n and Δ_{n-1} for some CA. If this is not the case, then the algorithm "fails," in that a computed remainder r does not have degree $k-2$ (equivalently, the next quotient is not degree 1). An updated procedure, CalcCAFromPolys, that incorporates a check for this failure is given below.

Algorithm 7.4

procedure **CalcCAFromPolys**

input: *degree n polynomial Δ_n and degree $n-1$ polynomial w*

output: *if $w = \Delta_{n-1}$ for a CA of Δ_n,*
then sequences (d_i), (b_i) and (c_i), $i = 1, \ldots, n$.
else "false"

$c_1 \leftarrow 1; b_n \leftarrow 1$ *(arbitrary)*

$k \leftarrow n$

$\Delta_{n-1} \leftarrow w$

repeat

$a_k \leftarrow \Delta_k \operatorname{div} \Delta_{k-1}$

if $\text{degree}(a_k) \neq 1$ *then return "false"*

$d_k \leftarrow x - a_k$

if $k = 1$ then return (d_i), (b_i) and (c_i), $i = 1, \ldots, n$

$r \leftarrow \Delta_k \bmod \Delta_{k-1}$

choose b_{k-1} and c_k such that $-b_{k-1}c_k = \text{lcoeff}(r)$

$\Delta_{k-2} \leftarrow \text{normal}(r)$

$k \leftarrow k - 1$

end repeat

The algorithm raises an interesting issue about the number of n-cell CA over GF(q). The number of linear rules is given by q^3, and so there are q^{3n} CA, ignoring the null-boundary. Respecting that the null-input multipliers are irrelevant reduces this to q^{3n-2}. The fact that it is the product of the multipliers b_{k-1} and c_k that is determined, rather than the actual values, means that large numbers of CA are trivially isomorphic, and the algorithm does not distinguish them. Furthermore, the algorithm demands that the product be non-zero. Hence the number of CA is the number of choices for the n self-multiplier times the number of choices for the $n-1$ inter-cell connections,

$$q^n(q-1)^{n-1}.$$

7.2 Probabilistic synthesis

Efforts to find a synthesis approach analogous to one for GF(2) have, to this time, failed. Attempts to prove the existence of CA for irreducible polynomials have met with the same fate. However, all is not lost, in that there are probabilistic methods that (may) have acceptable probabilities of success.

Perhaps the most obvious approach to probabilistic synthesis for a degree n polynomial $\Delta_{1,n}$ is as follows.

Algorithm 7.5

while(max iterations not exceeded)

choose a random n-cell CA

if the characteristic polynomial of the CA is $\Delta_{1,n}$ output CA and stop

The best approximation for the probability of success of an iteration of this algorithm is $1/q^n$ (this is discussed further in section 7.4). For n and q of any significant magnitude, this approach is extremely impractical.

The material in section 7.1 suggests the following probabilistic algorithm for finding the CA for a given characteristic polynomial Δ_n.

Algorithm 7.6

procedure **FindCA**

while(max iterations not exceeded)

choose a random degree $n-1$ polynomial w

CalcCAFromPolys(Δ_n, w)

if successful, output CA and stop

Over GF(2), this algorithm performs extremely poorly. However, it is shown below that the larger the field, the better the probability of success in each iteration. To

get an idea of how many iterations might be needed to find a CA for Δ_n, consider the probability of procedure CalcCAFromPolys failing.

Lemma 7.7 *If p is a fixed monic degree n polynomial in $GF(q)[x]$, and s is a monic degree $n-1$ polynomial in $GF(q)[x]$ with uniformly distributed random variables from $GF(q)$ as coefficients, then the degree $n-2$ coefficient of $p \bmod s$ is a uniformly distributed random variable from $GF(q)$.*

Proof. Let $p = x^n + p_{n-1}x^{n-1} + p_{n-2}x^{n-2} + \cdots + p_0$ and $s = x^{n-1} + s_{n-2}x^{n-2} + s_{n-3}x^{n-3} + \cdots + s_0$. The degree $n-2$ term of $p \bmod s$ is given by

$$p_{n-2} - s_{n-3} - s_{n-2}(p_{n-1} - s_{n-2}).$$

The linear term s_{n-3} is independent of the non-linear terms, and the result follows. □

To determine how many iterations might be needed to find a CA for Δ_n, consider the probability of procedure CalcCAFromPolys failing when applied to Δ_n and a random degree $n-1$ polynomial w. Note that the procedure never fails in the first iteration, as the first quotient a_n is guaranteed to be degree 1.

If w is a random degree $n-1$ polynomial, then the degree $n-2$ coefficient of the remainder r calculated in the first iteration is also randomly distributed. The second iteration is successful if and only if r has degree $n-2$, and so the probability that the procedure fails in the second iteration is

$$\text{prob}(\text{degree } n-2 \text{ coefficient of } r = 0) = \frac{1}{q}, \tag{7.3}$$

where q is the size of the finite field.

Assuming success of the second iteration, consider the calculation $a_{n-2} = \Delta_{n-2} \operatorname{div} \Delta_{n-3}$ in the third iteration. This quotient is degree 1 if and only if the remainder r calculated in the second iteration has degree $n-3$. Now, this remainder is not a random polynomial independent of Δ_{n-1}, as it is determined uniquely by Δ_n and

Δ_{n-1}. Despite this, the division algorithm is of such a nature that the coefficients of the remainder are seemingly randomly distributed, and the analysis proceeds under this assumption. Hence the probability of failure during the third iteration is the same as that for the second, $1/q$. Continuing the assumption of independence, the probability for success for the procedure is given by

$$\left(\frac{q-1}{q}\right)^{n-1}, \tag{7.4}$$

as n is the maximum number of iterations required (recall that the first division is guaranteed to have a degree 1 quotient).

The assumption of independence of the sequence of polynomials $\Delta_n, \Delta_{n-1}, \ldots, \Delta_1$, though certainly incorrect, is given some support by empirical evidence. In Table 7.1, the *expected %* column was calculated by (7.4). The *actual %* column was obtained by randomly choosing 100 polynomials of degree n, and for each, running procedure CalcCAFromPolys with 100 randomly chosen degree $n-1$ polynomials. The percentage of these degree $n-1$ polynomials that successfully gave a CA is reported. As expected, for a fixed field, the larger the value of n, the lower the probability of success for procedure SearchForCA.

Section 7.4 gives a different approach to the above probability estimation, and gives a more efficient synthesis algorithm. The basis for this algorithm is presented in the following section.

7.3 PRS structure

This section analyses the structure that exists in polynomial remainder sequences (PRS). This structure lends supporting evidence to the probability estimate in the previous section, and leads to a much improved probabilistic synthesis method. The following defines notation that is useful in this section and in the next.

Definition 7.8 *Let a and b be polynomials in GF(q)[x]. Define $\mathcal{P}^l(a,b)$ to be the largest k such that the first k divisions in the normalised calculation of the GCD*

Table 7.1: Probability of success of an iteration of procedure **FindCA**

$\|F\|$	n	Actual %	Expected %	$\|F\|$	n	Actual %	Expected %
3	3	43.3	44.4	3^2	4	52.1	51.2
3	4	30.4	29.6	5	5	40.5	41.0
3	5	19.5	19.7	5	6	33.3	32.8
3	6	13.6	13.2	9	3	79.3	79.0
2^2	3	55.9	56.2	9	4	71.9	70.2
2^2	4	43.0	42.0	9	5	61.7	62.4
2^2	5	32.8	31.6	11^2	50	66.0	66.6
2^2	6	23.7	23.7	17^2	50	84.2	84.4
5	3	60.9	64.0	71	50	50.3	49.9

of a and b have degree 1 quotients. Let $\mathcal{P}^s(a,b)$ denote the sequence formed with these degree 1 quotients, and $\mathcal{C}^s(a,b)$ denote the sequence formed with the leading coefficients of the remainders. Also, let $\mathcal{R}^s(a,b)$ denote the sequence of unnormalised remainders. $\mathcal{P}^s(a,b)$, $\mathcal{C}^s(a,b)$, and $\mathcal{R}^s(a,b)$ may be singly or doubly subscripted to indicate an element or subsequence, respectively.

Also, the sequence $\mathcal{C}^s(a,b)$ is referred to as the *remainder leading coefficients.*

Table 7.2 shows an example of the structure being explored, for GF(3) and $n = 5$. For example, the first entry in the table indicates that there are six irreducible degree 5 polynomials with *polynomial remainder sequence pattern* (PRSP) [81, 54, 36, 28, 28]. For each of these, the behaviour of CalcCAFromPolys applied to the polynomial and all 81 degree 4 polynomials is analysed. The PRSP [81, 54, 36, 28, 28] means that for all 81 degree 4 polynomials, the first quotient has degree 1. Of the 81, 54 result in the second quotient being degree 1, and 36 of these have the first three quotients all of degree 1. Finally, 28 have the first four quotients all of degree 1, and all 28 of these have the fifth quotient of degree 1. Thus, a polynomial with this PRSP has 28 different CA realisations, and the table indicates there are six such polynomials.

More formally, an entry in the table indicates the number of polynomials that have a particular PRSP. The ith entry in the PRSP of a degree 5 polynomial is the count of

the number of degree 4 polynomials that have $\mathcal{P}^l(\geq, i)$. Hence the last column shows the number of different CA realisations of the polynomials (that is, the number of degree 4 polynomials that have all quotients of degree 1).

The polynomials in the table are partitioned into two groups, irreducible (*irr*) and reducible (*red*). Interestingly, only three different PRSP patterns appear for the 48 irreducible degree 5 polynomials over GF(3).

Note the apparent structure in the table, that all of the sequences start with the sequence [81, 54, 36]. It is shown below that the first $\lceil n/2 \rceil$ entries are

$$q^{n-1}, q^{n-1}\left(\frac{q-1}{q}\right), q^{n-1}\left(\frac{q-1}{q}\right)^2, \ldots, q^{n-1}\left(\frac{q-1}{q}\right)^{\lceil n/2 \rceil - 1}.$$

Also note that the total number of CA can be calculated by summing the product of the *polys* column and the final entry in the PRSP column. This sum $(6{\cdot}28{+}24{\cdot}22{+}\cdots)$ is 3888, the total number of 5-cell CA over GF(3). The huge discrepancy between this number and the $q^{3n} = 14,348,907$ discussed in section 7.1 is explained by the fact that the algorithm is not sensitive to the actual multipliers b_{k-1} and c_k, but only to their product. Also, the synthesis constructs only CA that have non-zero input multipliers b_i and c_i. Viewed this way, there are q^n choices for the self-multipliers d_i and $(q-1)^{n-1}$ choices for the products $b_1c_2, b_2c_3, \ldots, b_{n-1}c_n$. Note that $3^5 \cdot 2^4 = 3888$.

The following lemma is a natural generalisation of Lemma 6.16. It shows that by modifying the initial divisor, the quotients and remainders can be selectively modified for the first half of the GCD computation.

Lemma 7.9 *Suppose p_n and p_{n-1} are polynomials of degree n and $n-1$ respectively, and suppose $\mathcal{P}^l(p_n, p_{n-1}) \geq k$, for some $1 \leq k \leq \lfloor n/2 \rfloor$. Let c be a polynomial, and let l denote the leading coefficient of c.*
If c is degree $n-2k-1 \geq 0$, then

$$\begin{aligned} \mathcal{P}^s(p_n, p_{n-1})_{1,\ldots,k} &= \mathcal{P}^s(p_n, p_{n-1}+c)_{1,\ldots,k}, \quad \textit{and} \\ \mathcal{R}^s(p_n, p_{n-1})_k &= \mathcal{R}^s(p_n, p_{n-1}+c)_k + lx^{n-k-1} + u, \end{aligned}$$

Table 7.2: PRSP analysis ($|F| = 3$, $n = 5$)

Type	Polys	PRSP	Type	Polys	PRSP
irr	6	[81, 54, 36, 28, 28]	irr	18	[81, 54, 36, 24, 24]
irr	24	[81, 54, 36, 22, 22]			
red	3	[81, 54, 36, 32, 18]	red	33	[81, 54, 36, 29, 18]
red	9	[81, 54, 36, 28, 14]	red	6	[81, 54, 36, 27, 11]
red	3	[81, 54, 36, 26, 16]	red	6	[81, 54, 36, 26, 12]
red	6	[81, 54, 36, 26, 10]	red	6	[81, 54, 36, 25, 17]
red	12	[81, 54, 36, 25, 14]	red	6	[81, 54, 36, 24, 24]
red	6	[81, 54, 36, 24, 11]	red	6	[81, 54, 36, 24, 10]
red	6	[81, 54, 36, 24, 8]	red	6	[81, 54, 36, 22, 17]
red	18	[81, 54, 36, 23, 15]	red	6	[81, 54, 36, 23, 13]
red	6	[81, 54, 36, 23, 7]	red	3	[81, 54, 36, 22, 4]
red	6	[81, 54, 36, 21, 12]	red	6	[81, 54, 36, 21, 8]
red	9	[81, 54, 36, 20, 12]	red	18	[81, 54, 36, 19, 19]
red	3	[81, 54, 36, 19, 11]	red	3	[81, 54, 36, 19, 9]
red	3	[81, 54, 36, 15, 6]			

where degree$(u) < n - k - 1$.

If c *is degree* $n - 2k$, *then*

$$\begin{aligned} \mathcal{P}^s(p_n, p_{n-1})_{1,\ldots,k-1} &= \mathcal{P}^s(p_n, p_{n-1} + c)_{1,\ldots,k-1} \quad \textit{and} \\ \mathcal{P}^s(p_n, p_{n-1})_k &= \mathcal{P}^s(p_n, p_{n-1} + c)_k - l. \end{aligned}$$

Proof. The proof is very similar to the proof of Lemma 6.16. Consider the polynomial remainder sequence calculation of p_n and p_{n-1}:

$$\begin{aligned} p_n &= a_n p_{n-1} + (-l_n) p_{n-2} \\ &\vdots \\ p_{n-k+2} &= a_{n-k+2} p_{n-k+1} + (-l_{n-k+2}) p_{n-k} \\ p_{n-k+1} &= a_{n-k+1} p_{n-k} + r. \\ &\vdots \end{aligned} \tag{7.5}$$

Let L_i denote $(\prod_{j=i}^{n} l_j)^{-1}$. Now consider the characteristic polynomial calculation, with equations reversed, and multiplied through by c:

$$\begin{array}{lcll} 0 & = & a_n c & - \; c\Delta_{n,n} \\ c \cdot 1 & = & a_{n-1} L_n c\Delta_{n,n} & - \; L_n c\Delta_{n-1,n} \\ L_n c\Delta_{n,n} & = & a_{n-2} L_{n-1} c\Delta_{n,n-1} & - \; L_{n-1} c\Delta_{n-2,n} \\ & \vdots & & \\ L_{n-k+3} c\Delta_{n,n-k+3} & = & a_{n-k+1} L_{n-k+2} c\Delta_{n,n-k+2} & - \; L_{n-k+2} c\Delta_{n-k+1,n}. \end{array} \tag{7.6}$$

When (7.6) is added to (7.5), the terms from (7.5) have strictly larger degrees in each of the first $k-1$ equations. Thus the first $k-1$ quotients and $k-1$ remainder leading coefficients are unchanged.

Suppose that c has degree $n-2k-1$. The last equation of the sum is

$$p_{n-k+1} + L_{n-k+3} c\Delta_{n,n-k+3} = a_{n-k+1}(p_{n-k} + L_{n-k+2} c\Delta_{n,n-k+2}) + r - L_{n-k+2} c\Delta_{n-k+1,n}.$$

Thus the remainder is offset by a degree $n-2k-1+k = n-k-1$ polynomial, that has leading coefficient $-L_{n-k+2} l$.

If the degree of c is $n-2k$, then the remainder in the last equation is offset by a degree $n-k$ polynomial, and becomes invalid (i.e., it does not satisfy the division algorithm). The corrected equation is

$$\begin{array}{ll} & p_{n-k+1} + L_{n-k+3} c\Delta_{n,n-k+3} \\ = & (a_{n-k+1} - L_{n-k+2} l)(p_{n-k} + L_{n-k+2} c\Delta_{n,n-k+2}) \\ + & (r - L_{n-k+2} c\Delta_{n-k+1,n}) + (L_{n-k+2} l)(p_{n-k} + L_{n-k+2} c\Delta_{n,n-k+2}). \end{array}$$

The kth quotient is changed by $-L_{n-k+2} l$. □

Example 7.10. Consider the polynomials $p_n = x^8 + 2x^6 + 2x^5 + x^2 + 1$ and $p_{n-1} = x^7 + 2x^6 + x^5 + 5x^3 + 1$ over GF(7). The following partial normalised GCD calculation shows that $\mathcal{P}^l(p_n, p_{n-1}) > 2$:

$$\begin{array}{rcl} x^8 + 2x^6 + 2x^5 + x^2 + 1 & = & (x-2)(x^7 + 2x^6 + x^5 + 5x^3 + 1) \\ & & +(-2)(x^6 + 5x^5 + 6x^4 + 2x^3 + 3x^2 + 4x + 2) \\ x^7 + 2x^6 + x^5 + 5x^3 + 1 & = & (x-3)(x^6 + 5x^5 + 6x^4 + 2x^3 + 3x^2 + 4x + 2) \\ & & +(-4)(x^5 + 3x^4 + 5x^3 + 4x^2 + x). \end{array}$$

The value $L_n = 1/2$ appears repeatedly in the example.

Letting $k = 2$, choosing a degree $n-2k-1 = 3$ polynomial $c = 4x^3$, and calculating the GCD of p_n and $p_{n-1} + c$:

$$\begin{aligned} x^8 + 2x^6 + 2x^5 + x^2 + 1 &= (x-2)(x^7 + 2x^6 + x^5 + 2x^3 + 1) \\ &\quad +(-2)(x^6 + 5x^5 + x^4 + 5x^3 + 3x^2 + 4x + 2) \\ x^7 + 2x^6 + x^5 + 2x^3 + 1 &= (x-3)(x^6 + 5x^5 + x^4 + 5x^3 + 3x^2 + 4x + 2) \\ &\quad +(-6)(x^5 + 5x^4 + 5x^2 + 3x). \end{aligned}$$

The first two quotients and the first remainder leading coefficient are unchanged, and the second remainder leading coefficient has changed from -4 to $-4 - 4/2 = -6$.

If, however, $c = 6x^3$, then

$$\begin{aligned} x^8 + 2x^6 + 2x^5 + x^2 + 1 &= (x-2)(x^7 + 2x^6 + x^5 + 4x^3 + 1) \\ &\quad +(-2)(x^6 + 5x^5 + 2x^4 + 3x^3 + 3x^2 + 4x + 2) \\ x^7 + 2x^6 + x^5 + 4x^3 + 1 &= (x-3)(x^6 + 5x^5 + 2x^4 + 3x^3 + 3x^2 + 4x + 2) \\ &\quad +(-4)(x^4 + x^3 + 4x^2 + x). \end{aligned}$$

As before, the change occurs in the second remainder. The degree 5 coefficient has changed from -4 to $-4 - 6/2 = 0$. Thus $\mathcal{P}^l(p_n, p_{n-1} + c) = 2$.

Still considering $k = 2$, and letting c be the degree $n - 2k = 4$ polynomial $2x^4$,

$$\begin{aligned} x^8 + 2x^6 + 2x^5 + x^2 + 1 &= (x-2)(x^7 + 2x^6 + x^5 + 2x^4 + 5x^3 + 1) \\ &\quad +(-2)(x^6 + 6x^5 + 4x^4 + 2x^3 + 3x^2 + 4x + 2) \\ x^7 + 2x^6 + x^5 + 2x^4 + 5x^3 + 1 &= (x-4)(x^6 + 6x^5 + 4x^4 + 2x^3 + 3x^2 + 4x + 2) \\ &\quad +(-5)(x^4 + 5x^3 + 4x^2 + 1). \end{aligned}$$

The first quotient and remainder leading coefficient are unchanged, but the second quotient has become $(x-3) + 2/(-2) = (x-4)$. □

Corollary 7.11 *For every degree n polynomial p, and every choice of $n/2$ degree 1 quotients and $n/2 - 1$ remainder leading coefficients, there is exactly one polynomial w of degree less than n such that the normalised GCD calculation of p and w gives these quotients and remainder leading coefficients.*

Proof. The existence is shown constructively. Let $a = [a_n, a_{n-1}, \ldots, a_{n/2+1}]$ and $l = [l_n, l_{n-1}, \ldots, l_{n/2+2}]$ be the desired sequences of quotients and remainder leading coefficients. The degree $n-1$ polynomial w is initially chosen at random. The first quotient is made to be a_n by changing w to become $w + c$, $c = (\mathcal{P}^s(p, w)_n - a_n)x^{n-2}$.

Then, the first reminder leading coefficient is made to be l_n by changing w to $w + c$, $c = (C^s(p, w)_n - l_n)x^{n-3}$. This process is continued, setting the second quotient, then the second remainder leading coefficient, etc.

To show the uniqueness, suppose that both w and u have the desired property. By Lemma 7.9, the GCD calculation of p and $w + (u - w) = u$ has either a quotient or a remainder leading coefficient that differs from the calculation of p and w. This is a contradiction. □

7.4 An improved algorithm

This section presents a probabilistic approach to the synthesis problem that has significantly better performance than the approach in section 7.2. To simplify the presentation, n is assumed to be even throughout this section. A CA is being sought for the fixed polynomial p, of degree n.

The argument in section 7.2, based on the probability of successive degree 1 quotients in the GCD algorithm, can be replaced with a counting-based argument. The total number of monic degree n polynomials over GF(q) is q^n, and the total number of CA is $q^n(q-1)^{n-1}$. Thus, the average number of CA for each polynomial is

$$\frac{q^n(q-1)^{n-1}}{q^n} = (q-1)^{n-1}. \tag{7.7}$$

Consider the "brute-force" synthesis approach, which is to randomly choose a CA and check if its characteristic polynomial is p. This is not affected by the fact that (7.7) is greater than 1; the probability of success is simply $1/q^n$.

Now consider the approach of section 7.2, which is to randomly choose a monic degree $n-1$ polynomial w and check if the GCD calculation of p and w results in n degree 1 quotients. If p has $(q-1)^{n-1}$ CA, then there are $(q-1)^{n-1}$ degree $n-1$ polynomials w that would lead to success. The probability of choosing one such polynomial is

$$\frac{(q-1)^{n-1}}{q^{n-1}} = \left(\frac{q-1}{q}\right)^{n-1}.$$

This is the same as the probability derived in section 7.2.

The improved probabilistic algorithm comes about from a simple observation: if the w are chosen at random from the set of degree $n-1$ polynomials that have $\mathcal{P}^l(p,w) \geq n/2$, then the probability of success is much higher. The framework to do this is as follows.

Consider a sequence of degree 1 quotients $a = [a_n, a_{n-1}, \ldots, a_{n/2+1}]$ and remainder leading coefficients $l = [l_n, l_{n-1}, \ldots, l_{n/2+2}]$. By Corollary 7.11, there is exactly one degree $n-1$ polynomial w such that $\mathcal{P}^s(p,w)_{1,n/2} = a$ and $\mathcal{C}^s(p,w)_{1,n/2} = l$. Furthermore, w can be constructed with at most $2 \cdot n/2 = n$ polynomial additions and GCD calculations.

A random choice of a and l is equivalent to a random choice of length $n/2$ CA. As a consequence, Corollary 7.11 states that one half of a CA determines the other half. This gives the following synthesis algorithm:

Algorithm 7.12

procedure **FindCA**

while(max iterations not exceeded)

choose a random length $n/2$ CA d c ' l,

determine the degree $n-1$ polynomial w such that $\mathcal{P}^s(p,w)_{1,n/2} = a$

and $\mathcal{C}^s(p,w)_{1,n/2} = l$

call CalcCAFromPolys(p,w)

if successful, output CA and stop

The number of length $n/2$ CA is

$$q^{n/2}(q-1)^{n/2-1},$$

and so the probability of selecting a particular w is the inverse of this. As before, there are $(q-1)^{n-1}$ degree $n-1$ polynomials w that would lead to success. Hence

Table 7.3: Comparison of synthesis algorithms

$\|F\|$	n	Seconds	Actual %	Expected %	Alg. 7.6
8	20	31	23.0	26.31	7.91
9	20	28	38.0	30.79	10.67
16	20	63	54.0	52.45	29.34
32	50	434	45.0	45.22	21.10
49	100	1105	34.0	35.67	12.99

the probability of success for an iteration of the algorithm is

$$\frac{(q-1)^{n-1}}{q^{n/2}(q-1)^{n/2-1}} = \frac{(q-1)^{n/2}}{q^{n/2}} = \left(\frac{q-1}{q}\right)^{n/2}.$$

Table 7.3 shows estimated and actual probabilities for an iteration of Algorithm 7.12, and the estimated probability for an iteration of Algorithm 7.6. Also shown are actual running times, using Maple V on a SPARC 5 Sun workstation, for Algorithm 7.12 to perform 100 iterations. Appendix H contains a Maple V program listing.

7.5 Conclusion

This chapter shows that the calculation of the characteristic polynomial of a CA over GF(q) is related to a normalised GCD calculation using Euclid's algorithm. As with GF(2), this forms the basis for a synthesis procedure. Unlike the GF(2) case, however, there does not seem to be relationship that allows a worst case for the GCD algorithm to be directly calculated. Instead, a randomised algorithm is used, and the probability of success for the algorithm is estimated. A detailed analysis of Euclid's algorithm leads to an improved randomised approach to synthesis, and uncovers some interesting structure in polynomial remainder sequences.

Chapter 8

Conclusion

This chapter summarises the research contributions of this dissertation, and discusses several areas for future work.

8.1 Contributions

This dissertation is a study of the characteristic polynomials of one-dimensional linear hybrid cellular automata (CA). These linear finite state machines have been proposed as an alternative to linear feedback shift registers (LFSRs) in VLSI applications such as test pattern generation and signature analysis. The characteristic polynomial of a CA determines the linear operator that it implements, which in turn provides insight into its behaviour.

This dissertation has contributed the following for CA defined over any finite field:

- an increased understanding of the basic relationships satisfied by CA characteristic polynomials,
- a constructive proof of the similarity transform between CA and LFSR transition matrices,
- a relationship between the characteristic polynomials of cyclic-boundary CA and null-boundary CA,
- a probabilistic method for synthesising CA from polynomials.

Further contributions for CA over GF(2) are:

- a proof of the reducibility of the characteristic polynomial of a cyclic-boundary CA,
- an efficient algorithm for the synthesis of CA from irreducible polynomials,
- a proof of the existence and uniqueness of CA for irreducible polynomials,
- classification of the characteristic polynomials of partially-connected CA,
- results concerning palindromic, uniform, and self-concatenated CA.

These contributions create a solid foundation for understanding the theoretical properties of CA.

8.2 Future work

There are several areas that could be explored in more depth. The determination of criteria for the existence of CA for reducible characteristic polynomials is an open problem. Understanding the polynomial remainder sequence patterns for reducible polynomials may well be the key to the solution of this problem.

The existence of CA for fields other than GF(2) is an open problem, as is a synthesis method analogous to the GF(2) algorithm. As with the reducible polynomials over GF(2), the solution likely lies in the polynomial remainder sequence patterns, even though they are not as structured as in the GF(2) case.

The study of slightly more general CA is worth pursuing. Three broad categories are:

- multi-dimensional linear hybrid cellular automata, possibly with some restrictions on the size of some dimensions (2-by-n CA, for example);
- "slightly-non-linear" CA, that allow constant-1 boundary inputs, or perhaps some number of complement operations;

- the CA studied in this dissertation, with small relaxations of the communication restrictions (for example, to allow some number of second-nearest-neighbour connections).

Bibliography

[1] P.H. Bardell. Analysis of cellular automata used as pseudorandom pattern generators. In *Proceedings of IEEE International Test Conference*, pages 762–767, 1990.

[2] P.H. Bardell. Discrete logarithms: a parallel pseudorandom pattern generator analysis method. *Journal of Electronic Testing: Theory and Applications*, 3:17–31, 1992.

[3] P.H. Bardell. Primitive polynomials of degree 301 through 500. *Journal of Electronic Testing: Theory and Applications*, 3:175–176, 1992.

[4] P.H. Bardell, W.H. McAnney, and J. Savir. *Built-In Test for VLSI.* John Wiley and Sons, New York, 1987.

[5] Leonard E. Baum and Melvin M. Sweet. Continued fractions of algebraic power series in characteristic 2. *Annals of Mathematics*, 103:593–610, 1976.

[6] Leonard E. Baum and Melvin M. Sweet. Badly approximable power series in characteristic 2. *Annals of Mathematics*, 105:573–580, 1977.

[7] E.R. Berlekamp. *Algebraic Coding Theory.* McGraw-Hill, New York, 1968.

[8] S. Boubezari and B. Kaminska. Cellular automata synthesis based on precomputed test vectors for built-in self-test. In *Proceedings of IEEE International Conference on Computer-Aided Design*, pages 578–583, 1993.

[9] R. Burden and J. Faires. *Numerical Analysis.* Prindle, Weber and Schmidt, Boston, third edition, 1985.

[10] K. Cattell and J.C. Muzio. Synthesis of one-dimensional linear hybrid cellular automata. *To appear in IEEE Transactions on Computer-Aided Design.*

[11] K. Cattell and J.C. Muzio. Characteristic polynomials of one-dimensional cyclic linear cellular automata, 1990. Technical report DCS-133-IR, Dept. of Computer Science, Univeristy of Victoria.

[12] K. Cattell and J.C. Muzio. A linear cellular automata algorithm: Algorithm summary, 1991. Technical report DCS-162-IR, Dept. of Computer Science, Univeristy of Victoria. A version was presented at the Fifth New Directions for Testing Workshop.

[13] K. Cattell and J.C. Muzio. A linear cellular automata algorithm: Theory, 1991. Technical report DCS-161-IR, Dept. of Computer Science, Univeristy of Victoria.

[14] K. Cattell and J.C. Muzio. Tables of cellular automata for lowest weight primitive polynomials for degrees up to 300, 1991. Technical report DCS-163-IR, Dept. of Computer Science, Univeristy of Victoria.

[15] K. Cattell and J.C. Muzio. Analysis of one-dimensional linear hybrid cellular automata over GF(q). *To appear in IEEE Transactions on Computers*, 1994.

[16] K. Cattell and M. Serra. The analysis of one dimensional multiple-valued linear cellular automata. In *Proceedings of the Twentieth International Symposium on Multiple-Valued Logic*, pages 402–409, 1990.

[17] K. Cattell and S. Zhang. Minimal cost one-dimensional linear hybrid cellular automata of degree through 500. *Journal of Electronic Testing: Theory and Applications*, 6:255–258, 1995.

[18] Chih-Ang Chen and Sandeep K. Gupta. BIST test pattern generation for stuck-open and delay testing. In *Proceedings of the European Design and Test Conference*, pages 289–296, 1994.

[19] D.R. Chowdhury, S. Basu, I.S. Gupta, and P.P. Chaudhuri. Design of CAUCC – cellular automata based error correcting code. *IEEE Transactions on Computers*, 43:759–764, 1994.

[20] D.R. Chowdhury, I.S. Gupta, and P.P. Chaudhuri. CA-based byte error-correcting code. *IEEE Transactions on Computers*, 44:371–382, 1995.

[21] Andrea Clementi and Russell Impagliazzo. The reachability problem for finite cellular automata. *Information Processing Letters*, 53:27–31, 1995.

[22] T. Damarla and A. Sathaye. Applications of one-dimensional cellular automata and linear feedback shift registers for pseudo-exhaustive testing. *IEEE Transactions on Computer-Aided Design*, 12:1580–1591, 1993.

[23] A.K. Das and P.P. Chaudhuri. Vector space theoretic analysis of additive cellular automata and its application for pseudoexhaustive test pattern generation. *IEEE Transactions on Computers*, 42:340–352, 1993.

[24] A.K. Das, A. Ganguly, A. Dasgupta, S. Bhawmik, and P.P. Chaudhuri. Efficient characterisation of cellular automata. *IEE Proceedings of Part E: Computers and Digital Techniques*, 137:81–87, 1990.

[25] A.K. Das, M. Pandey, A. Gupta, and P.P. Chaudhuri. Built-in self-test structures around cellular automata and counters. *IEE Proceedings of Part E: Computers and Digital Techniques*, 137:269–276, 1990.

[26] A.K. Das, B. Saha, A.B. Chowdhury, S. Misra, and P.P. Chaudhuri. Signature analysis based on additive cellular automata. In *Proceedings of IEEE International Symposium on Fault-Tolerant Computing*, pages 265–272, 1990.

[27] G. Edirisooriya and J. Robinson. Aliasing in multiple-valued test data compaction. In *Proceedings of the Twenty-second International Symposium on Multiple-Valued Logic*, pages 43–50, 1992.

[28] P. Guan and Y. He. Exact rules for deterministic cellular automata with additive rules. *Journal of Statistical Physics*, 43:463–478, 1986.

[29] T.A. Gulliver, M. Serra, and V.K. Bhargava. Primitive polynomials with independent roots and their applications. In *Canadian Conference on Electrical and Computer Engineering*, pages 818–822, 1988.

[30] T. Hansen and G.K. Mullen. Primitive polynomials over finite fields. *Math. Comp.*, 59:639–643, 1992.

[31] Mahbub Hassan. The selection of better polynomials to reduce aliasing in signature analysis. Master's thesis, University of Victoria, 1991.

[32] Andrzej Hkawicka and Michal Kopec. Properties of the rule 150/90 cellular automata-based MISRs and LFSRs used in built-in self-test. Technical report, Technical Univeristy of Gliwice, Poland.

[33] Kenneth Hoffman and Ray Kunze. *Linear Algebra.* Prentice-Hall, New Jersey, second edition, 1971.

[34] P.D. Hortensius, R.D. McLeod, and H.C. Card. Cellular automata-based signature analysis for built-in self-test. *IEEE Transactions on Computers*, 39:1273–1283, 1990.

[35] P.D. Hortensius, R.D. McLeod, W. Pries, D.M. Miller, and H.C. Card. Cellular automata-based pseudorandom number generators for built-in self-test. *IEEE Transactions on Computer-Aided Design*, 8:842–859, 1989.

[36] Peter D. Hortensius. *Parallel Computation of Non-Deterministic Algorithms in VLSI.* PhD thesis, University of Manitoba, 1987.

[37] Hassan Janoowalla. Analysis of maximal length linear two-dimensional cellular automata. Master's thesis, University of Victoria, 1992.

[38] E. Jen. Global properties of cellular automata. *Journal of Statistical Physics*, 43:219–242, 1986.

[39] M. Khare and A. Albicki. Cellular automata used for test pattern generation. In *Proceedings of IEEE International Conference on Computer Design*, pages 56–59, 1987.

[40] D.E. Knuth. *The Art of Computer Programming. Seminumerical Algorithms*, volume 2. Addison-Wesley, 1981.

[41] E. Kontopidi. The partitioning behaviour of linear finite state machines and VLSI applications. Master's thesis, University of Victoria, 1992.

[42] E. Kontopidi and J.C. Muzio. The partitioning of linear registers for testing applications. *Microelectronics Journal*, 24:533–546, 1993.

[43] H.Y. Lee and Y. Kawahara. On dynamical behaviours of cellular automata ca-60. *Bulletin of Informatics and Cybernetics*, 25:21–25, 1992.

[44] R. Lidl and H. Niederreiter. *Introduction to Finite Fields and their Applications.* Cambridge University Press, Cambridge, 1986.

[45] R.J. McEliece. *Finite Fields for Computer Scientists and Engineers.* Kluwer, Boston, 1987.

[46] J.M. Mesirov, 1994. Private communication.

[47] J.M. Mesirov and M.M. Sweet. Continued fraction expansions of rational expressions with irreducible denominators in characteristic 2. *Journal of Number Theory*, 27:144–148, 1987.

[48] D.M. Miller, J.C. Muzio, M. Serra, X. Sun, S. Zhang, and R.D. McLeod. Cellular automata techniques for compaction based BIST. In *Proceedings of IEEE International Symposium on Circuits and Systems*, pages 1893–1896, 1991.

[49] T.S. Motzkin. The Euclidean algorithm. *Bull. Amer. Math. Soc.*, pages 1142–1146, 1949.

[50] T.S. Motzkin and E.G. Straus. Some combinatorial extremum problems. *Bull. Amer. Math. Soc.*, pages 1014–1020, 1956.

[51] J.C. Muzio, 1989. Private communication.

[52] S. Nandi, B. Vamsi, S. Chakraborty, and P. P. Chaudhuri. Cellular automata as a BIST structure for testing CMOS circuits. *IEE Proceedings of Part E: Computers and Digital Techniques*, 141:41–47, January 1994.

[53] Danian J. Neebel and Charles R. Kime. Multiple weighted cellular automata. In *Proceedings of the VLSI Test Symposium*, pages 81–86, 1994.

[54] Harald Niederreiter. Rational functions with partial quotients of small degree in their continued fraction expansion. *Monatshefte fur Mathematik*, 103:269–288, 1987.

[55] M. Nohmi. On a polynomial representation of finite linear cellular automata. *Bulletin of Informatics and Cybernetics*, 24:137–145, 1991.

[56] W. Wesley Peterson and E.J. Weldon. *Error-Correcting Codes.* MIT Press, Cambridge, Ma, 1972.

[57] W. Pries, A. Thanailakis, and H. Card. Group properties of cellular automata and VLSI applications. *IEEE Transactions on Computers*, C-35:1013–1024, 1986.

[58] M. Serra. Irreducible and primitive polynomials for GF(3), 1986. Technical report, Dept. of Computer Science, Univeristy of Victoria.

[59] M. Serra. Applications of multi-valued logic to testing of binary MVL circuits. *International Journal of Electronics*, 63:197–214, 1987.

[60] M. Serra and T. Slater. A Lanczos algorithm in a finite field and its application. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 7:11–32, 1990.

[61] M. Serra, T. Slater, J.C. Muzio, and D.M. Miller. The analysis of one dimensional linear cellular automata and their aliasing properties. *IEEE Transactions on Computer-Aided Design*, 9:767–778, 1990.

[62] H.S. Stone. *Discrete Mathematical Structures and Their Applications.* Science Research Associates Inc., 1973.

[63] X. Sun. *Merging Concurrent Checking and Off-line BIST.* PhD thesis, University of Victoria, 1993.

[64] X. Sun and M. Serra. Merging concurrent checking and off-line BIST. In *Proceedings of IEEE International Test Conference*, pages 958–967, September 1992.

[65] Ph. Tsalides. Cellular automata-based built-in self-test structures for VLSI systems. *IEE Electronics Letters*, 26:1350–1352, 1990.

[66] Ph. Tsalides, T.A. York, and A. Thanailakis. Pseudorandom number generators for systems based on linear cellular automata. *IEE Proceedings of Part E: Computers and Digital Techniques*, 138:241–249, 1991.

[67] J. van Sas, F. Catthoor, and H. De Man. Cellular automata based deterministic self-test strategies for programmable data paths. *IEEE Transactions on Computer-Aided Design*, 13:940–949, 1994.

[68] J. van Sas, F. Catthoor, and H.D. Man. Optimized BIST strategies for programmable data paths based on cellular automata. In *Proceedings of IEEE International Test Conference*, pages 110–119, 1992.

[69] S. Wolfram. Statistical mechanics of cellular automata. *Rev. of Modern Physics*, 55:601–644, 1983.

[70] S. Wolfram. Universality and complexity in cellular automata. *Physica*, 10D:1–35, 1984.

[71] S. Wolfram. Random sequence generation by cellular automata. *Advances in Applied Mathematics*, 7:123–169, 1986.

[72] S. Zhang. BIST generators for faults with sequential behavior. Master's thesis, University of Victoria, 1993.

Appendix A

CA tables

This appendix lists the CA realisations of each polynomial in GF(2)[x], for degrees 1 through 6. The tables also list the polynomials that have no CA realisation. This appendix is referred to in section 2.3.

Table A.1: Characteristic polynomials of CA, $n = 1$

polynomial	irred	CA	polynomial	irred
x	yes	0		
$x+1$	yes	1		

Table A.2: Characteristic polynomials of CA, $n = 2$

polynomial	irred	CA	polynomial	irred
x^2	no	11	x^2+x	no
x^2+1	no	00		
x^2+x+1	yes	10 01		

Table A.3: Characteristic polynomials of CA, $n = 3$

polynomial	irred	CA		polynomial	irred
x^3	no	000		x^3+1	no
x^3+x	no	101		x^3+x^2+x	no
x^3+x+1	yes	110	011		
x^3+x^2	no	010			
x^3+x^2+1	yes	100	001		
x^3+x^2+x+1	no	111			

Table A.4: Characteristic polynomials of CA, $n = 4$

polynomial	irred	CA		polynomial	irred
x^4	no	1001		x^4+x^2	no
x^4+1	no	0110		x^4+x^2+x	no
x^4+x	no	1100	0011	x^4+x^2+x+1	no
x^4+x+1	yes	1010	0101	x^4+x^3	no
x^4+x^2+1	no	0000	1111	x^4+x^3+x+1	no
x^4+x^3+1	yes	1101	1011	$x^4+x^3+x^2$	no
x^4+x^3+x	no	1110	0111	$x^4+x^3+x^2+x$	no
$x^4+x^3+x^2+1$	no	1000	0001		
$x^4+x^3+x^2+x+1$	yes	0100	0010		

Table A.5: Characteristic polynomials of CA, $n = 5$

polynomial	irred	CA		polynomial	irred
x^5	no	11011		$x^5 + x + 1$	no
$x^5 + 1$	no	11101	10111	$x^5 + x^2$	no
$x^5 + x$	no	00000		$x^5 + x^2 + x$	no
$x^5 + x^2 + 1$	yes	11110	01111	$x^5 + x^2 + x + 1$	no
$x^5 + x^3 + 1$	yes	01100	00110	$x^5 + x^3$	no
$x^5 + x^3 + x$	no	01010	10001	$x^5 + x^3 + x + 1$	no
$x^5 + x^3 + x^2$	no	10100	00101	$x^5 + x^3 + x^2 + x$	no
$x^5 + x^3 + x^2 + 1$	no	10010	01001	$x^5 + x^4 + 1$	no
$x^5 + x^3 + x^2 + x + 1$	yes	11000	00011	$x^5 + x^4 + x^2$	no
$x^5 + x^4$	no	11111		$x^5 + x^4 + x^2 + 1$	no
$x^5 + x^4 + x$	no	01000	00010	$x^5 + x^4 + x^2 + x$	no
$x^5 + x^4 + x + 1$	no	00100		$x^5 + x^4 + x^3 + x$	no
$x^5 + x^4 + x^2 + x + 1$	yes	10000	00001	$x^5 + x^4 + x^3 + x^2$	no
$x^5 + x^4 + x^3$	no	10110	01101	$x^5 + x^4 + x^3 + x^2 + x$	no
$x^5 + x^4 + x^3 + 1$	no	11010	01011		
$x^5 + x^4 + x^3 + x + 1$	yes	11100	00111		
$x^5 + x^4 + x^3 + x^2 + 1$	yes	11001	10011		
$x^5 + x^4 + x^3 + x^2 + x + 1$	no	01110	10101		

Table A.6: Characteristic polynomials of CA, $n = 6$

polynomial	irred	CA		polynomial	irred
x^6	no	001100		x^6+1	no
x^6+x+1	yes	011000	000110	x^6+x	no
x^6+x^2	no	100001		x^6+x^2+x	no
x^6+x^2+1	no	010010	111111	x^6+x^3	no
x^6+x^2+x+1	no	010100	001010	$x^6+x^3+x^2+1$	no
x^6+x^3+1	yes	100010	010001	$x^6+x^3+x^2+x$	no
x^6+x^3+x	no	100100	001001	$x^6+x^3+x^2+x+1$	no
x^6+x^3+x+1	no	101000	000101	x^6+x^4+x+1	no
$x^6+x^3+x^2$	no	110000	000011	$x^6+x^4+x^2$	no
x^6+x^4	no	011110		$x^6+x^4+x^2+x$	no
x^6+x^4+1	no	000000	101101	$x^6+x^4+x^3+x$	no
x^6+x^4+x	no	110101	101011	$x^6+x^4+x^3+x^2$	no
$x^6+x^4+x^2+1$	no	110011		$x^6+x^4+x^3+x^2+1$	no
$x^6+x^4+x^2+x+1$	yes	111001	100111	$x^6+x^4+x^3+x^2+x$	no
$x^6+x^4+x^3$	no	111010	010111	x^6+x^5	no
$x^6+x^4+x^3+1$	no	110110	011011	x^6+x^5+x	no
$x^6+x^4+x^3+x+1$	yes	101110	011101	$x^6+x^5+x^2+1$	no
$x^6+x^4+x^3+x^2+x+1$	no	111100	001111	$x^6+x^5+x^2+x$	no
x^6+x^5+1	yes	011010	010110	$x^6+x^5+x^3+x^2+x$	no
x^6+x^5+x+1	no	110001	100011	$x^6+x^5+x^3+x^2+x+1$	no
$x^6+x^5+x^2$	no	011100	001110	$x^6+x^5+x^4$	no
$x^6+x^5+x^2+x+1$	yes	101001	100101	$x^6+x^5+x^4+1$	no
$x^6+x^5+x^3$	no	110010	010011	$x^6+x^5+x^4+x$	no
$x^6+x^5+x^3+1$	no	101100	001101	$x^6+x^5+x^4+x^2$	no
$x^6+x^5+x^3+x$	no	100110	011001	$x^6+x^5+x^4+x^2+x$	no
$x^6+x^5+x^3+x+1$	no	110100	001011	$x^6+x^5+x^4+x^2+x+1$	no
$x^6+x^5+x^3+x^2$	no	111000	000111	$x^6+x^5+x^4+x^3$	no
$x^6+x^5+x^3+x^2+1$	yes	101010	010101	$x^6+x^5+x^4+x^3+x^2$	no
$x^6+x^5+x^4+x+1$	yes	100000	000001	$x^6+x^5+x^4+x^3+x^2+1$	no
$x^6+x^5+x^4+x^2+1$	yes	111110	011111	$x^6+x^5+x^4+x^3+x^2+x$	no
$x^6+x^5+x^4+x^3+1$	no	001000	000100	$x^6+x^5+x^4+x^3+x^2+x+1$	no
		111011	110111		
$x^6+x^5+x^4+x^3+x$	no	111101	101111		
$x^6+x^5+x^4+x^3+x+1$	no	010000	000010		

Appendix B

Cyclic CA tables

This appendix lists all cyclic-boundary CA with up to seven cells, and the characteristic polynomial of each. As cyclic CA have both reverse and rotational symmetry, only one representative from each "equivalence class" is listed. The polynomials are shown in factored form. Define w_0 and w_1 as the parity of the number of 0s and 1s, respectively, of the CA rule vector. This appendix is referred to in section 3.6.

Table B.1: Degree 2 Cyclic CAs and polynomials

w_0	w_1	CA	polynomial
even	even	00	$(x+1)^2$
		11	x^2
odd	odd	01	x^2+x+1

Table B.2: Degree 3 Cyclic CAs and polynomials

w_0	w_1	CA	polynomial
even	odd	001	$(x+1)^3$
		111	$x^2(x+1)$
odd	even	000	$(x+1)^2x$
		011	x^3

Table B.3: Degree 4 Cyclic CAs and polynomials

w_0	w_1	CA	polynomial
even	even	0000	x^4
		0011	$(x^2+x+1)^2$
		0101	$(x+1)^2x^2$
		1111	$(x+1)^4$
odd	odd	0001	$x^3(x+1)$
		0111	$(x+1)^3x$

Table B.4: Degree 5 Cyclic CAs and polynomials

w_0	w_1	CA	polynomial
even	odd	00001	$(x+1)(x^2+x+1)^2$
		00111	$x^4(x+1)$
		01011	$(x+1)^5$
		11111	$(x+1)(x^2+x+1)^2$
odd	even	00000	$x(x^2+x+1)^2$
		00011	$(x+1)^4x$
		00101	x^5
		01111	$x(x^2+x+1)^2$

Table B.5: Degree 6 Cyclic CAs and polynomials

w_0	w_1	CA	polynomial
even	even	000000	$(x+1)^4x^2$
		000011	$(x^3+x^2+1)^2$
		000101	$x^2(x^2+x+1)^2$
		001001	$(x+1)^6$
		001111	$(x^3+x+1)^2$
		010111	$(x+1)^2(x^2+x+1)^2$
		011011	x^6
		111111	$(x+1)^2x^4$
odd	odd	000001	$(x+1)^5x$
		000111	$(x+1)x(x^2+x+1)^2$
		001011	$(x+1)^3x^3$
		010101	$(x+1)x(x^2+x+1)^2$
		011111	$x^5(x+1)$

Table B.6: Degree 7 Cyclic CAs and polynomials

w_0	w_1	CA	polynomial
even	odd	0000001	$(x+1)(x^3+x^2+1)^2$
		0000111	$(x+1)^5x^2$
		0001011	$(x+1)^3(x^2+x+1)^2$
		0010011	$x^6(x+1)$
		0010101	$(x+1)(x^3+x+1)^2$
		0011111	$(x+1)^7$
		0101111	$(x+1)^3x^4$
		0110111	$(x+1)(x^3+x^2+1)^2$
		1111111	$(x+1)(x^3+x+1)^2$
odd	even	0000000	$x(x^3+x^2+1)^2$
		0000011	x^7
		0000101	$(x+1)^4x^3$
		0001001	$x(x^3+x+1)^2$
		0001111	$(x+1)^2x^5$
		0010111	$x^3(x^2+x+1)^2$
		0011011	$(x+1)^6x$
		0101011	$x(x^3+x^2+1)^2$
		0111111	$x(x^3+x+1)^2$

Appendix C

CA for Least Weight Primitive Polynomials

This appendix lists least-weight primitive polynomials over GF(2), and their CA realisations. The polynomials are from [4], and range from degree 2 through to degree 300. This appendix is referred to in section 6.7.

Each row of the table contains

- the polynomial, written as a list of terms with non-zero coefficients,
- the time required to synthesise the CA,
- the DOS (degree of symmetry) of the CA,
- the rule vector of the CA, with the last DOS cells excluded (as they are equal to the first DOS cells, reversed).

For example, the table entry

$$7,1,0 \quad 0.01 \quad 2 \quad 10110$$

means

- the polynomial $x^7 + x + 1$ is primitive and least-weight,
- the CA required 0.01 seconds to calculate,

- the CA has DOS equal to 2,
- the CA is 1011001.

poly	time	DOS	CA
2,1,0	0.01	0	10
3,1,0	0.01	0	110
4,1,0	0.01	0	0101
5,2,0	0.00	0	01111
6,1,0	0.01	1	00011
7,1,0	0.01	2	10110
8,6,5,1,0	0.03	0	01001011
9,4,0	0.01	1	01001110
10,3,0	0.01	2	11110000
11,2,0	0.01	3	01000011
12,7,4,3,0	0.05	1	10010101001
13,4,3,1,0	0.01	3	0111001110
14,12,11,1,0	0.06	0	01000111001111
15,1,0	0.01	6	100000011
16,5,3,2,0	0.05	4	000111100100
17,3,0	0.03	7	1001100011
18,7,0	0.05	4	11000100000001
19,6,5,1,0	0.05	5	11010111011010
20,3,0	0.06	7	0110101110000
21,2,0	0.03	8	0100100110010
22,1,0	0.10	9	0100011010101
23,5,0	0.03	10	0101101010110
24,4,3,1,0	0.10	9	110100111100100
25,3,0	0.05	11	10100000111110
26,8,7,1,0	0.08	8	110011011111011110
27,8,7,1,0	0.05	8	0001101001100010111
28,3,0	0.10	11	01011011100000011
29,2,0	0.05	12	01001000100101111
30,16,15,1,0	0.11	6	101000100111001101101010
31,3,0	0.05	14	11111011011000011
32,28,27,1,0	0.13	1	00001100010001110000110000000011
33,13,0	0.06	15	111001011001000110
34,15,14,1,0	0.18	8	100011111010110100110100000
35,2,0	0.06	15	11101000101110011110
36,11,0	0.18	11	010010010001100001011111O
37,12,10,2,0	0.06	11	100010001011010000000001100
38,6,5,1,0	0.25	15	11111011011100010000010
39,4,0	0.08	16	010101110100111100011111
40,21,19,2,0	0.16	8	110011000001100000010001010000001
41,3,0	0.08	19	0111110010010000101110
42,23,22,1,0	0.23	8	11101110111111101010110001010100000
43,6,5,1,0	0.08	17	00110101000101000110011000
44,27,26,1,0	0.25	7	01110101111100010000011101010110O1000
45,4,3,1,0	0.10	19	00111000111011111101001110
46,21,20,1,0	0.31	11	011110100000101000000111011100011011
47,5,0	0.10	22	100101101010100001101101l
48,28,27,1,0	0.28	9	010100000001111101000100101101111001111
49,9,0	0.10	23	100001000001001010110111l0
50,27,26,1,0	0.30	10	011111101111111111010000100111110110100O
51,16,15,1,0	0.11	16	110001111110101000010000101100101l1
52,3,0	0.31	23	00111100000111101001111011011
53,16,15,1,0	0.13	17	011111001111001000010000001100110111
54,37,36,1,0	0.35	7	100111001111101110101100101100110011010000101O1
55,24,0	0.11	14	011110110110111001110011001000111101111110
56,22,21,1,0	0.38	16	101010110111010111110001101011110101010O
57,7,0	0.11	27	110100111001011011010110100110
58,19,0	0.40	18	100011001100111011101100010010110011001O

poly	time	DOS	CA
59,22,21,1,0	0.13	17	01110100000011000100001010010000110111111110
60,1,0	0.50	28	11100111101001011101000010111100
61,16,15,1,0	0.15	21	10100100101110111101001111001000000100101
62,57,56,1,0	0.46	1	10110111010110111010110111010111011010110111010110111010111011
63,1,0	0.13	30	10000000000000000000000000000011
64,4,3,1,0	0.50	29	10011101010011011110110110011001001
65,18,0	0.16	22	10000011110010010001111001101111001100100110
66,10,9,1,0	0.50	27	00111000011001001010100110100011100101l
67,10,9,1,0	0.16	27	00000110010110000100110101011111101100100
68,9,0	0.55	28	10101100000100001111111101011101101101l0
69,29,27,2,0	0.16	32	11000011001110011111011111010011011111
70,16,15,1,0	0.57	26	00110011111111111110001010001000011000000l101
71,6,0	0.18	31	10010110111101100111111000000010111010l1
72,53,47,6,0	0.65	8	10111110111001101010101010101011110110000101111100000000000111000000
73,25,0	0.18	35	01111111000000111101000100101111010011
74,16,15,1,0	0.66	28	01101010011010010101011101000000011101101001111
75,11,10,1,0	0.18	31	00101000000001011000000011001011000110111000
76,36,35,1,0	0.76	19	01111001111101000011001111011110000011110101111111110100
77,31,30,1,0	0.20	22	11110100001101110101011111011100001110000011110101001l0
78,20,19,1,0	0.88	28	10010010110110101010000110110101010111111010110101
79,9,0	0.21	38	01110111001001111111110011011010000101011
80,38,37,1,0	0.90	20	01010110010000100000101000110011011110111011110101011011110111110
81,4,0	0.23	37	00100010001001011100001101111101101000001001
82,38,35,3,0	1.00	22	01000011001000111011111111000011100001010100101010001001100l0
83,46,45,1,0	0.25	17	11011110000000000111111111010110101111001101111111100011011001001000
84,13,0	0.83	34	11001011011010001101000010010011011001101101111010
85,28,27,1,0	0.23	27	10011001000001010111011111101001111011011110111000070010l1
86,13,12,1,0	0.93	35	01110100100111101111001010010011100101111110100000100
87,13,0	0.23	42	10100100010001101000010110001011100100110010
88,72,71,1,0	1.14	7	00011011010111111100011110110011001001001100101100100011001111011010100000000000
89,38,0	0.25	24	00010101110100010000010001101011001000110011110000111010100101001
90,19,18,1,0	1.14	34	10001000111000000010001110001001010110011010101111110101
91,84,83,1,0	0.26	2	01010001000011111010111010101100111111011001100100001111111110001011011000101110110101011
92,13,12,1,0	0.96	38	01010111100000011000010100000100101000001111100001010l
93,2,0	0.25	44	00001001001001000111000110110101101111111110101001
94,21,0	1.05	35	11101101011011101111110010011001111101111101001101011100010
95,11,0	0.26	46	01001010010001101011000000000001100111110101010l1
96,49,47,2,0	1.31	22	11110111101100000010001001101101110101100100011011100011010103110010101110
97,6,0	0.28	44	00111111110010100011010000111000100000011110010101111
98,11,0	1.12	42	10111101100100110001111011100110010001101011011010101010
99,47,45,2,0	0.31	47	10001100111101011001011011001011011011010010000011
100,37,0	1.19	30	11111111100100000001111001011111110001101011001110100111110010000011100
101,7,6,1,0	0.30	46	11110110011000101101100100101001111010001110100101101l1
102,77,76,1,0	1.26	11	10111110011000000000011111101011101100111101100010111000111100101001101110010101101010010011
103,9,0	0.31	50	01111110111101010001000101010001101001000111110101011
104,11,10,1,0	1.60	45	10101111110011010101111111001110000111001100110001100100110
105,16,0	0.30	43	00000110001101111010100110100001010001000010000001001110110l0
106,15,0	1.35	44	10011011000100001001000100001010001101100000000010011111000101
107,65,63,2,0	0.33	51	10101010000100111011100110001011100101110100001101101111
108,31,0	1.42	37	01010011000010101100001110110100100000100100111000110101110001010011100l0
109,7,6,1,0	0.33	50	01111001100000000100110000000000011001101010110000100100000
110,13,12,1,0	1.48	47	01110001001111110110010110000001000011000011101000001000001111l
111,10,0	0.35	49	10110110011111111110010101001111110011001111011000100011011l000
112,45,43,2,0	1.56	32	01010011100000111000110001101000101000100101110000010100001001011101011101110
113,9,0	0.40	55	01101001100110110001111101110101101111101111000111111010011

poly	time	DOS	CA
114,82,81,1,0	1.85	15	01101111111100001110001001010000111001101101111100100111111100010 0010101000000011001101001100000001
115,15,14,1,0	0.36	49	00100111001101110100001000110101011000110100011011011000010000 0
116,71,70,1,0	1.71	21	10110001001001010111101110101001111111101100000001110100110001001 01111110001101111101101010111
117,20,18,2,0	0.38	47	11111101001000100011000000010110101100110111001000011011010011090 01011
118,33,0	1.75	41	10110111100000011011111000010101011000111111101101111110100101010 100001000000
119,8,0	0.36	54	11001110100001000110110101111001010110010011101010010001100010111
120,118,111,7,0	1.96	3	01000110100010111001001100000000010011100101000000100101110001011 1000001001000000010100111001001101101010011010001011
121,18,0	0.38	50	10010010111011000110011010110111110111011000001100010110110101000 010011
122,60,59,1,0	1.83	30	10100101000100110100001001111100011001101011011100010110111001011 00111011111110110110011011
123,2,0	0.40	59	00001001111110011000101010110111101110011111010110111001000110010
124,37,0	1.89	42	10000111000110001110011111101111101011001010111011000000100100011 10000100010000010
125,108,107,1,0	0.45	7	01000101000000001111010001111011111111101010000010110011101100110 10001111101011111110111100100111110011000011101010110
126,37,36,1,0	1.98	43	10000111111011100010110001101010001000111101111110111011111010011 101010101010101100
127,1,0	0.38	62	10000000000000000000000000000000000000000000000000000000000000110
128,29,27,2,0	2.04	48	01001000100010000010111110111010110011100000011000110100111101001 11110100111001
129,5,0	0.45	63	00101111101011101111110010011111011101101111100100100110101001011 0
130,3,0	2.10	62	00011110100110011001010101011011101010111011010100100111110011011 011
131,48,47,1,0	0.46	40	11101110110000111010101011001010100101110001110000101100101001001 00100011101111110000010001
132,29,0	2.23	50	10001111101110000001001010111000000001111011000110101101111101011 10111010011110000
133,52,51,1,0	0.50	39	00101001011101101111000001011100010000111101000010111011111111000 0000010000101111001110011100 0
134,57,0	2.27	37	01101001000101100111101011101001111100010000010100000010111100111 10110011010011001101001110101001
135,11,0	0.48	66	01101010101111111011011100000000110101110000111010110110100110000 1011
136,126,125,1,0	2.93	4	01100000111001110100000011011010000110000101011110100100010000100 01101100010100100110110011100100010101101010111000111100110010001 11
137,21,0	0.50	67	01000000001110011100100100010011101110010100101110110000111000111 10011
138,8,7,1,0	2.43	64	01101100100001110100111110001101000100101101011010101100000000001 011011010
139,8,5,3,0	0.50	64	01101110000010100101101010001000101001010111000111100011100100100 0001010011
140,29,0	2.52	54	10011001011101101101001110100101100101010101000101011111111000011 11010110101100101111 0
141,32,31,1,0	0.51	53	10110111011100011111100010011111100010100000111100010011110001101 10100110011111010110111
142,21,0	2.60	59	11001610011001001100101010001010001101101000011001001111010111110 111000000010011010
143,21,20,1,0	0.55	60	11001100100111110000001101110010001101110101111010111111000001110 101100000900011011
144,70,69,1,0	2.75	36	11010100001111110111000101010110110010010000101101011101010110100 11011110100011000010011100110110110111001110
145,52,0	0.55	45	00100000110101011000011001100100101100001101000010101110111101101 10101101110000110100100110100000011
146,60,59,1,0	2.81	42	10111001000110000101101000011010100011110011000110010110001100001

poly	time	DOS	CA
			11010000010101111100000100001111010101
147,38,37,1,0	0.56	53	10110001010000010111001111101011011011000000110000101010101011000010001100111010111111000011111
148,27,0	2.96	59	01011001111101101010110110100010010011101110011011110000010001010101000001111110011000011
149,110,109,1,0	0.61	18	11000011110101000011011111010101000011000101010000011100101000011010110101110011000000000010010110010010111010001100010111000010000
150,53,0	2.89	47	00100010111110010111010001010111111101000000010100110101101111011011111110010010011101100000000011101
151,3,0	0.57	74	11011111001101011000001010000110100110000111100110101000001111100101110111110
152,66,65,1,0	3.06	42	01110001000001011000100110111000010110100000000111100111101110111000100000101011011010000101000000011101110100001
153,1,0	0.60	75	01000111000110011100111011011111010000001001010011001101101011100001010110
154,129,127,2,0	3.25	11	11101001111111111010110000000010011010111110110111110101100011000011111110010011100011100010001111000011100100100100000010100001010001001100010
155,32,31,1,0	0.63	60	10000001001100011111100101101011111011110101101101100110110110101011110001001101001
156,116,115,1,0	3.29	19	10001111111000111001101100011111010010011100001011000001001101100101010111010100001001000011110111010111010000100100011011001
157,27,26,1,0	0.61	64	01100100001000100001001000100101001000010011001010011000101011011000000100011000011010011000
158,27,26,1,0	3.62	64	01110001110010110111011100011110011000110011100010000010010001011110000110101101101100010010
159,31,0	0.61	78	01111000000111010100110100010010100000100110000101100010100000111010100000110001011
160,19,18,1,0	4.17	69	11100010000110010001110101101110011011100111001100010010111111111000100000000000010111101101
161,18,0	0.68	70	11001100111100110010010101010000110100100000110011101010101011001001010100011010111101010000
162,88,87,1,0	4.36	36	11110011101001001100011101111000000000000000110000011100110001000000100111101101100001010100001001101100100111000101011001101111
163,60,59,1,0	0.71	50	01101001110010111001001010101010111100111100100100110011010010111010111001100111111000000000101110110010011001111
164,14,13,1,0	3.70	74	01100000101010111011001010101011000010111100001010111010110101001001100010010111001101000
165,31,30,1,0	0.70	66	01111100111011110001100001001001010011111110111001010000000011100011011100011010111000111000110
166,39,38,1,0	4.50	62	11010001100110110001001100110101101010110100010000010110110100001011001001010000010100100011010010100001
167,6,0	0.73	79	10100001100100011101619901110101111000010110100111010111011100111001110011011100101010110111
168,17,15,2,0	4.66	74	11110110111101111100001111101111001001100111001001001000110010110110010010000110101011001010101
169,34,0	0.73	66	01001011003000110101100110000100010111000100011100110011100000100001011111111000110010101100001110100001
170,23,0	4.00	72	11011101110100111101011011100001011011110011110101001110011100011000011000011100001111101110000001101
171,42,3,1,0	0.80	63	00000001010111011011100011001110100110001100101101101011110001000100011000010010111100011000110010111111110101
172,7,0	4.04	81	10100101111110010010111100101000001100000101101000000110111102001001000100011000011010111001
173,10,2,1,0	0.75	80	00111000111001010001101106011100010001101000011010001100011101111001010101110011001111100110
174,13,0	4.25	79	00100110000011010111110011011111111000101101010101100011100110100111101000110011100011100101010
175,6,0	0.78	83	00111110111011000000010101101101000111101010010101000010111000000010000111010101111000010000
176,43,2,1,0	4.30	65	01010000001010010110111111110001100001101100011001000000001111110

poly	time	DOS	CA
			1000100010110011110101110011100010000110100110
177,8,0	0.78	83	1001111101001110111000001010010011010101111100101000110000100011O 1101001011011111100100100101
178,87,0	4.48	44	00000000101000101001111101100101011100100100010111100001111001010 110010010100100111010101101010010000011101100111110010101111100010 0001
179,4,2,1,0	0.80	86	1000010110100000100001110001001011111000000111011000110000101011O 11011000000100101000111100000
180,52,2,1,0	5.48	88	0000001110001001011010001010001110100100001000000101101010110011 1 0001111010110100010000010011
181,89,2,1,0	0.81	88	01000000111111111110100011110001000111100001000000000000100101001 001100100011011111111101100011
182,121,2,1,0	5.65	29	11011000101001101101111110001101011010100111001000010011000000000010 01001111111001011001100000111001010001001001111101010101101111101 0101000001011111100000111
183,56,0	0.83	62	010101001111101001011101010010101000001001111110001010110101101111 01111111000110010111000010100111001101001001011110101000
184,41,3,1,0	4.83	70	1011101111000000000111101000111111001010011110110001101111101101 01000110110001100010110000010010101011011110000010
185,24,0	0.85	79	1110100001000001111110010011010100000100100000101011110010001110O 1010100110111111001010110011110100010011O
186,53,2,1,0	5.95	65	10110010110110101001110100010101111111001110111010101010000100100 1 0011010100010011110000000011001100000010110101111110010O1
187,20,2,1,0	0.86	82	011010010011001111010101110101010010000011110001000011111110011101 1000101011000111000011101101101111101111
188,186,2,1,0	6.33	92	11101111001001111111001001011110010100010010101110001110110100010 00011101011011100010111111111100
189,49,2,1,0	0.88	92	0111101011111110101010100001010001111101101110011110010111111010010 1001011000001101101100010101100O
190,47,2,1,0	5.17	70	000101100001100100001101110110011111011000100110111000011101001 1 11110111001101111110111001100010000110001011111101111100
191,9,0	0.90	94	11111100110110011001001000100001090110111100111111010111011101100 1011010011011000100101100001101 1
192,112,3,1,0	5.75	93	00111111111101111101111111011011101110010001000011101001110010101 010100010001100101110111101110111O
193,15,0	0.93	95	00101010110011101000000110111011111101010110010001000110010001010 0001100110111110111111000101101011
194,87,0	5.46	52	01101100011100100000011001111100110110100101000000010001001000001 01101101110111111010000100110011011101010001010010111001001111000 000111110001
195,37,2,1,0	0.96	95	11110000001100011110000000010111101111011000100101011011110101001 110001111011101010000000101010100 10
196,101,2,1,0	5.66	46	11010010010011001100000010001110001010011001010011101010000101010 10100001000000100000000111100100000111010010101110001010111010100 0 01010111010101010101
197,21,2,1,0	0.96	96	11111001011100100011111100001100010111001011001110111110000110010 1 0010011100001011000000101111101001001
198,65,0	5.75	65	00101000011000011001101111100010001010000000101000101001010101001 01011011100011111010100001010110000011111110110010010001110100011 10 111
199,34,0	1.00	81	11010100000100011111010110101111001110101101011011100110011001011 000010011010101001001010001110100101110100101001001 11
200,163,2,1,0	6.65	17	100000110011000000101010110011110001111111100000011001111000010001 00000011111011111101100100001000011010000011001110111111100001111 1 10010010000010010010011100111110101010111011100100011
201,14,0	1.01	92	1110011101111000000000111001110101010110010001100100010100000110 00 10110111101000100011111011011000101100111110
202,55,0	5.95	72	0000101110010100010010011110011011110000000100000111010011100000 01 101111111001101000111100000011101011011000001100111000101001111100
203,45,2,1,0	1.05	99	01001100101011111011000000001000111111000010100111010010001000010 11 0000111100111011001101010110011110111110
204,86,2,1,0	7.40	100	111011110101011100101001011010001001111010011001101011010111110 00 001000000001010110100110100111001011001 1

poly	time	DOS	CA
205,21,2,1,0	1.03	100	01101110001111011001100011010101100101011111000000111010111010110 10000010110011011110101011101011010110000
206,147,2,1,0	7.61	28	00100011011100101000011001110000111111110011100010100010000011111 11101000111000010010010111010111111010001111110000101011010101011 00111110111101110110001010100100111000111101110
207,43,0	1.08	102	01101001001111010101101000001000101100101011111000101110101111011 00000111010101011101100111111101110110
208,83,2,1,0	6.71	61	00011001010101100000100110010111011110010101101101011101011001101 11010101011100110001111111101101001011111111000001000101101110111 11100001101100110
209,6,0	1.14	100	11111000100101100101000000111100100100100011010100000000100010000 01001001111110100001101100100011101110101110
210,31,2,1,0	7.25	88	11100010010101001011001000000100000010100010001000100110100100100 11001001101011011001111001000100011001010011110000101001011
211,165,2,1,0	1.15	103	11101000011110100000101010011111111101100001111110010001001111010101 01001001000101001111001011110010111010011001
212,105,0	6.96	52	01010110006011010111001101011001111111101110110010110011101010111 00101010100001111010011011010100011000010111110011010110100011101 11011000111011000011100011111
213,62,2,1,0	1.20	74	10000010010110000101011111000001101100110100101101010111100000101 00110011101011011100001010110111010011000011100111010010000100110 0 111110111
214,87,2,1,0	7.08	6?	10101000111101001100111011000000111110101110001100110111111011101 00011011100000101001101011011011011011101111011111101000000111011101000 00100100110101011111110
215,23,0	1.14	106	01100011011100001110000100111111010101011101110101110110010010111 00100011011101110111101011111111011100010 1110
216,107,2,1,0	8.09	53	11111001101111101111110000011001011001001111010101110101110011111 00001010010010000110111001010110001000101111100111011100010000101 11010000110010110110011000011000
217,45,0	1.17	107	10001101010011001001001000111101001001010111000011111010000001011 00111101111100001110000111001100011001101101 1
218,11,0	7.29	102	01000011100111100001000011110110000110101101111001110101010110111 10000100001000109001001100100111100011100000101 1100
219,65,2,1,0	1.15	107	01011001000011000101100101111000000000011000100111010100000000000 11100010100100001111010110011011011001010010010
220,53,3,1,0	9.15	82	01011100000100110011001111101001010011110001100000011010111000010 11011101001101011100111101000000001110101011100101000000110110000 01011100
221,18,2,1,0	1.19	100	01011001100011011110001011110010111111001101000100111101011010011 10000111100000000001101010001100100011100010110010010011
222,73,2,1,0	7.96	73	10001100001101011111001110010100110111000001101001101101100001111 11000110111000110011011110000111011010100010110011111011111011010 1 01100010110001101 00
223,33,0	1.28	110	11000010111011001110101110010010010010101000110011101011000001111 00101010C9011110100100000101100110110101111000011
224,159,2,1,0	8.73	31	00101100011011010001000000001000000000111001010010100101001111000 01001000010111010110011110100101010011011001101010001100010101011 0000001011000000011011101000000001010110111010001101111011100001 1
225,32,0	1.23	95	11010101100110001110001000000001000110001000111000010111110010101 0 10000000110011001011100111101110011010100001101010100110100111100010
226,57,2,1,0	9.42	83	11000010000000111000111111110011101010101110000100111101011111010 10001010100101110011106101011111100111110011111100111000010101001 100001111010
227,21,2,1,0	1.28	111	00110001011111100110110101010101100001001110010000011100110110101 00110001000010011110101000011100101010111100111000
228,58,2,1,0	9.73	112	00101010110001110010001101011101111011000100100101011111111100011 00100000100001100001000011111000001010110100000011100
229,21,2,1,0	1.30	112	00010101110011001111000101011110010100011110011010000000010101000 00010100110111000000111010100111100000010111110000011
230,25,2,1,0	9.88	101	11601011011111110011100101111111001001000010110000000101100100100 0 00010001110100110101111111011101010111010010111000101011000111 00
231,26,0	1.33	101	00100001010101010011000111111011011111100110110000100001100011100

poly	time	DOS	CA
			11011001111011101101100001111111111001001000101100000111011011011
232,23,2,1,0	10.09	103	00100000001101010001010111001111100111110000110011001011101011111 00111001111011000001000101011000101111001010111010001000110011 01
233,74,0	1.35	78	11011100000100101000000000100000011110010000101100110001000001111 01001100100101100001011010010101110111001011011010100111110111010 00001100001101011011011 10
234,31,0	8.59	100	00110001001100001000001001001110110101111101000000001110001010000 00010011110110110100100010001010011011111101110010100010101110001 0111
235,45,2,1,0	1.37	115	01100000011010010001111100110101111011101100011111001011111111110 0111000000110011010011011110010101110001001111100101001
236,5,0	8.65	114	11011000100001100000100110001111101100111001010111011001000000101 0010111110001001011011001111100000110100101101110001111 11
237,163,2,1,0	1.40	116	00011001100100011101101110100000100110010010001011111001011101010 111010101001011100000000101001110001100001111001110110 00
238,5,2,1,0	8.88	115	01100111111111000011111101101100011100010001101011110100010110101 0110100101010110000011011010101001111101111010100000000011
239,36,0	1.42	100	00001000110010101001100101101001010110010101110110000110001110110 01111111001011011000100111101110100110011110111001000000100110 0111 100000010
240,49,3,1,0	9.19	94	11111000100111101100111000001100110110110000100111110000111110000 11101011110111101001101111010100001011111111111100001101010100011 1001110011101010
241,70,0	1.42	84	01101101000001011100101101101011000100100100010101110000011010110 01010101010101011010011011000001000110101001101101011101000010010 0010110011100100010001 10011
242,81,4,1,0	10.30	79	00111111100101000110101001111110001101000101101101001011011010001 10100000100111010001111010000110011101001000001001001010110111101 00000101011000101001001111110011 1
243,17,2,1,0	1.65	119	01101011101101111101111001001010110000110101010100011000000100100 0111001111101001011011000010011010110110001100011101011001 0
244,96,2,1,0	11.65	120	11100001000110011000100100000001010001111111111011011010001110011 1 000001001001111110101100010011011100000110000000110010100101
245,37,2,1,0	1.46	120	11111100010010011110100011111010011101010111110111111011011100011 1 110111101000110000010100101000111011011010001100100001111110
246,11,2,1,0	11.55	116	00001101010010011101111111101100101100101011011000000111010100001 1 01110000100110001110010010000010110010011111100111000111010101101
247,82,0	1 50	81	10110100100011011100011101001100010000010010111110110010000000101 00100010000111011100100101111101000001111001111111110010001001001 010011011110111101110010000101100010
248,243,2,1,0	12.55	1	00001010011100011101010110011101000100011101100100100011101111111 10110101101011010011111000001000000000111101110100010001011010101 10100010011001111100100111110010100111111001111001111011101010011 111011110001110110100010011100010001001000000001111 11
249,86,0	1.56	80	11111111010111111111000000000000000000100010001110010100111110011 10 00111111001110110111011110101100000111110100000101010001010010101 0 100001001011011011000111011010110110001
250,103,0	10.25	72	01010001101101101111000101010010001000110000000000100110110001100 01000010010110100010101010101001010101110011110011010011100100010 00101110111011000010111011100100110101000101101 1
251,45,2,1,0	1.58	123	10011011011101100011000100111010001001000111000001010100011000011 0100100000001010001001000000100110110011111111000000000010000110 00
252,67,0	10.41	91	00001110000111110001010100111100101111100101011101100110100001100 01100001000100000100001101011101011010010101110101011000101011111 01001000101010001100010101010 11
253,33,2,1,0	1.55	124	11100001000100010001110001001000000010111000111110111001101000100 1 1001001110011000100101000110111000000101100000010010101001000000 11
254,7,2,1,0	12.52	122	01011101100011001010011101100101100011101101100011001001001011101 00110010001001101111100000101000100100101101010110000011100011100 01
255,52,0	1.61	100	01010111111110101011100110001010001000110000101111111100000000000 1 10011100000101101111100010110010001011110010111010011101000101011 000000110100000000001010101

poly	time	DOS	CA
256,16,3,1,0	11.80	125	00011100111001001010011000110011110001111010011011100101010010111 11001010001000110110000111000000010110011000100001111010111110111 0
257,12,0	1.61	121	10100000101101101100111111011101101101011000111100011000101101001 00111010001010001010110011101110100001001110100110101000110111101 010000
258,83,0	11.08	86	11000000101011111011001111000101111101011110111011110000000010000011 00111000001101110011001000110001111101100110011110011110001011111 11100000101011011001000110011110000010010 01
259,254,2,1,0	1.85	1	11101010001101011001101000110001100001001010111111011100001011000 01010101010111000001010000011111111111011011000101010001101101100 11110001110000110000001101001011101001100110110011001101001000111 10110011001011111010101110111101101010100111000011101000110111 0
260,74,3,1,0	11.27	127	01111010111100011001000101111101101011000110010110000111110100001 10000100000010001010010010101101010100100101001111110000010010101 01 110
261,74,2,1,0	1.76	92	01000000011111000001100111000001100001111000111001101100110011111 01111010011011111001010101010111111000000001001110100110001101111 01011110011100001101010110011111110100 0
262,252,2,1,0	14.17	129	11111011110011101111111101010111001001011110111111110001110100001 01011000010001001101101010001001010110000010010101111000101110100 011
263,93,0	1.71	130	11000010111010100110100111100001001011011101100111100110100001001 11010111101100001001010101111010001000101110110111001010110111111 011
264,169,2,1,0	11.84	46	10001101011000100011010110110000010111100011000110111001011011011 11100011011101010010110011100101010010000100111100000100010001011 00010111011111011101101111001110101001001110011101100000110000101 0 01101100010100110100011
265,42,0	1.76	110	00010111111001101011101110011010110110011111010001000111011010011 0 01110111110110011101001111101110001010010001111001111001001000000 00001100010111111001101 00
266,47,0	11.92	108	11100101001100011110111011001100110110011011100100100101010110001 01001000111101001000101011111000000111111010010001011010001100100 10011110100100111000000010001
267,29,2,1,0	1.75	131	10110011111000000100111110101011111101010010111100111001100110000 10000111010010110011001000001010010001000001011110011101110010011 111110
268,25,0	12.23	120	00000100101110101111100110101110110000100101000111110000011011001 00111011101010000010001110101010100001111011010110001110101100001 0001110101110110 01
269,117,2,1,0	1.75	132	11000000000011000010111110111110000001011110110011100110101001001 00110101000001010000010001100111000000101011111111001000100110011 0000011
270,53,0	12.52	107	01000110010101111010111000101111010001100110110101010110000011100 11101101011111011110101100101100100111011111110110001001000011111 0 00010011001100001001101010001 0010
271,58,0	1.80	105	10100100000110110011101010000010111101000001100001110100000000100 11010010110000000011100110100011000110011101110011111110110000010 00110011110001000001110011001101 1100
272,56,3,1,0	13.88	133	10100011011110000111110001011010011110000111010110001100010100101 11101111010100110001010000101010001110010101001110101010010101101 110000101
273,23,0	1.86	135	10010100101001011000010100010011101100011100011101011101011111000 10100001101111100111101011111011001000010110010001001100010110001 01100110
274,67,0	12.96	102	10100110100010101000001100010111001110011111001110101001001010000 1 10111011111101000110100001100100101011011010000010001010111101010 10101001001001011010110111100000101011 1010
275.28,2,1,0	1.89	122	10011100101011101101001001010011001100111000100001100100000110000 10110110010111100111011100001111001100001010000010001011000010001 00110100111011110011011
276,28,2,1,0	15.80	136	00001001000000101000100110010010111011001100001011001110000010001

poly	time	DOS	CA
			10010110100000001110110001111001000000011111111000111100000101010 1110010101
277,254,5,1,0	2.08	10	1011111100010011111110110010001111101010100101001010110100110011 11110011110100101110111000011100000101100011101001101010011001110 11100111011010001001111111010001010001100101100010110010110100101 011111110101010000110000110000111000010110001000000010010100001110 1001011
278,5,0	13.30	135	00101011000101011100011101010001110101110101111010101001000101001 011101110001110011011000000000111111101011010001011011001111110010 01100000111111
279,5,0	1.96	138	11110001101011100101011010010011111110100010110001000001010001001 01100101101000111000100101011011011010100000001110010101011100110 11111111110
280,146,3,1,0	13.63	137	001010110111111001101000010101011001010000111011001000001100101001 110001001011111100100001100010101010111010001010101010100110000101 0011001111100
281,93,0	1.96	139	101011101110100111000011111011010001101011100101010000000000110100 11110010001110100010111000100001010110011000111011100111010000000 111010111011
282,35,0	13.77	122	01101000011010100110110100111101001110110010010000011000111100111 11001000001001001001011110101010000011100001110111001111000001010 101101010100010100111100101011
283,200,2,1,0	2.16	40	00101000110001101010111100010011100010100001110000101101010101110 11001010001001010101110000001111011100001000001001011011001110111 01001010111001000100010100011000010101001000111100000100011010011 111110011010010100000100110100010010001001101111
284,119,0	14.30	81	000100000000010010001011110010011010100111100001110010101100011011 11011100111111100011110010011011110001010000100000111111011101000 111011110111101101001000000010010101011011110010111010110100000000 00111011
285,77,2,1,0	2.08	140	01000111110110000010111001101111110101111011000101101110100011000 1000110000100000100000110111111111110111111111001111110010100011000 101000010110010
286,69,0	14.81	107	000101111111111000100110010101110001011001101001110011110011111101 10000110101111100011000000011010111101110011110000010100010001100 0011111001101001100011110010000001101100110001010
287,71,0	2.10	142	11000000011111111011000011110110000110011000110110011001000101100 10000010111001100110000000011001101111001110011110011010000010000 010100110100110
288,11,10,1,0	17.37	137	000100000011001001000011101001101010011100000100000010101100000000 110011101000101110010011010001100001001111110011101001010100011110 111000010000001100100
289,21,0	2.10	143	10001101111010101011101101011011010010110101000100001100110100110 01011001101010100011000011110100010010000001001101011101110110101 0100110110111110
290,5,3,2,0	17.69	141	11010101110111100010111101010010110011101100110001110111001100110 11000011011101000011001100000100110111100111011011101011000011110 1000110111101010101
291.76,2,1.0	2.20	106	10110010010111001100000001110000101111001100100010110011111111100 01100010110010110000100110110100111100000111100010111101011010001 0000111010100110001111101011011001001101010011010001110
292,97,0	15.21	96	01001001010001000100010010000101001001010111001111011001010111101 00010000101101101011011010100111101101001011000000110101101011100 001011001100100011100111011101100110101000000010000110001100100110 0
293,154,3,1,0	2.25	68	010110111111101110000110111100010111111010101110010111000100011010 10101110001101010111010101101011110101010000100000101011000001100 10100000110010110001100011110100001001001110111000001111011111001 000011010100011110010011110101
294,61,0	15.56	115	00110001110010011100100001110010101110010100101110101100010010111 01110000100001010000001010011001010100101010111011110001110001111 1100001011000010010101100001101101001101001101110
295,48,0	2.20	122	01100100011111100100111010011011110111001011010100010100011000000

poly	time	DOS	CA
			0100011011001001110001111011011000100111100111111110001111300111111 111101100001011010100101100111001100000101
296,11,9,4,0	17.21	141	011110001101111010000101001010101110010011101111000000000000001001 100011110101111010011110111001001111101111011000011101001000101011 10001011101010100000100001
297,5,0	2.23	147	011100101101100011011100110110111011100100010011001010000000100 001110111101111000101101010001101100101001101011000010011010101111 01111111101111010011
298,78,2,1,0	19.01	147	010001011000011011011101101110101011100101001010001111000111100 0 100111111011010011110101110010011100100110010111101001011001001 0110101111100010100011
299,21,2,1,0	2.25	147	010010000010001001010001000100000010101100111000010011111001011001 100010000110101100010001101111101001110010100100101110011100111101 10001110111011010111110
300,7,0	16.07	145	000010010111011111101010010000000001100000010101001010100111000111 010101010110111111100111111011010001101101101100101100000000101 1000000101001110101111001

Appendix D

Least-Weight CA over GF(2)

This appendix lists least-weight maximal-length CA, for each length up to 500. A least-weight maximal-length CA is a CA with a primitive characteristic polynomial that contains the least possible number of rule 150 cells (i.e. the least-weight rule vector d). This table also appears in [17]. This appendix is referred to in section 2.4.

The CA are specified by the numbers of the cells that use rule 150. For example, the entry $[2, 3]$ for $n = 8$ represents the CA with rule vector 01100000.

n	CA	n	CA	n	CA	n	CA	n	CA	n	CA
1	1	18	1, 17	35	1	52	2, 29	69	1	86	1
2	1	19	3	36	6	53	1	70	1, 37	87	13
3	1	20	2, 3	37	9	54	9	71	17	88	5
4	1, 3	21	1, 10	38	7	55	17	72	6, 55	89	1
5	1	22	5	39	1	56	4, 14	73	9	90	1
6	1	23	1	40	8	57	9	74	1	91	15
7	3	24	8, 12	41	1	58	17	75	7	92	3, 71
8	2, 3	25	9	42	19	59	4, 15	76	2, 22	93	33
9	1	26	1	43	3	60	2, 38	77	3, 44	94	42
10	2, 7	27	1, 20	44	4, 26	61	1, 10	78	1, 41	95	1
11	1	28	3	45	9	62	5	79	9	96	6
12	3, 7	29	1	46	2, 10	63	31	80	1, 71	97	1, 82
13	5	30	1	47	13	64	3, 5	81	1	98	8
14	1	31	11	48	15	65	1	82	1, 69	99	13
15	3	32	1, 15	49	1, 10	66	1, 19	83	1	100	1, 67
16	1, 15	33	1	50	11	67	15	84	36	101	1, 20
17	5	34	1, 19	51	1	68	8	85	1, 46	102	33

n	CA	n	CA	n	CA	n	CA	n	CA	n	CA
103	15	143	35	183	1	223	9	263	95	303	1
104	2, 40	144	13	184	2, 163	224	64	264	2, 173	304	4, 130
105	1	145	1, 46	185	61	225	33	265	69	305	73
106	30	146	1	186	1, 125	226	1, 33	266	46	306	1
107	19	147	1, 136	187	45	227	109	267	103	307	123
108	1, 35	148	8	188	2, 10	228	6	268	1, 73	308	14
109	1, 4	149	1, 108	189	1	229	21	269	1, 210	309	1
110	13	150	2, 102	190	2, 8	230	1	270	81	310	1, 43
111	27	151	27	191	1	231	1	271	5	311	7
112	2, 5	152	40	192	3, 111	232	93	272	3, 109	312	1, 295
113	1	153	13	193	9	233	1	273	1	313	57
114	22	154	66	194	14	234	1, 145	274	2, 29	314	3, 41
115	41	155	1	195	19	235	69	275	103	315	123
116	16	156	2, 24	196	32	236	61	276	81	316	1, 25
117	33	157	33	197	65	237	33	277	1, 72	317	2, 127
118	30	158	1	198	1, 145	238	35	278	44	318	16
119	1	159	61	199	41	239	1	279	69	319	21
120	3, 73	160	1, 151	200	11	240	3, 7	280	2, 47	320	3, 79
121	45	161	1, 116	201	1, 26	241	1, 162	281	1	321	97
122	14	162	73	202	24	242	4	282	3, 19	322	2, 74
123	51	163	1, 24	203	7	243	1	283	35	323	1
124	21	164	26	204	1, 103	244	2, 49	284	49	324	19
125	13	165	3, 130	205	57	245	1	285	1, 208	325	33
126	40	166	72	206	1, 71	246	40	286	2, 167	326	1
127	15	167	1, 60	207	1, 130	247	1, 102	287	109	327	25
128	1, 29	168	1, 113	208	74	248	1, 185	288	1, 61	328	80
129	49	169	81	209	1	249	81	289	101	329	1
130	1, 27	170	22	210	1	250	1, 217	290	112	330	1
131	1	171	37	211	17	251	1	291	97	331	21
132	18	172	3	212	2, 169	252	4, 31	292	5	332	1, 65
133	1, 4	173	1	213	1, 56	253	93	293	1	333	1, 52
134	26	174	16	214	59	254	1	294	1, 79	334	104
135	1	175	2, 71	215	29	255	55	295	57	335	1, 12
136	1, 97	176	1, 45	216	4, 26	256	1, 127	296	2	336	4, 35
137	1, 132	177	1, 22	217	53	257	49	297	3, 84	337	1, 100
138	28	178	1, 79	218	50	258	1, 181	298	111	338	1, 41
139	11	179	1	219	49	259	21	299	1	339	2, 9
140	8	180	19	220	47	260	4, 89	300	1, 251	340	2, 37
141	25	181	81	221	1	261	1	301	81	341	145
142	5	182	34	222	2, 12	262	3, 189	302	2, 105	342	69

n	CA	n	CA	n	CA	n	CA	n	CA	n	CA
343	99	370	75	397	113	424	2, 80	451	11	478	86
344	13	371	1	398	1	425	37	452	29	479	1, 140
345	49	372	1, 257	399	61	426	1	453	1	480	8, 239
346	32	373	5	400	198	427	2, 91	454	18	481	225
347	13	374	2, 313	401	185	428	184	455	31	482	1, 285
348	1, 25	375	1	402	154	429	1	456	2, 209	483	1
349	117	376	62	403	179	430	3, 93	457	1, 220	484	3, 237
350	2, 63	377	149	404	68	431	1	458	91	485	1, 272
351	1, 64	378	2, 12	405	117	432	1, 49	459	19	486	1, 181
352	1, 145	379	33	406	116	433	45	460	32	487	159
353	97	380	4, 102	407	95	434	86	461	1, 90	488	188
354	24	381	1, 44	408	2, 239	435	2, 281	462	138	489	109
355	69	382	138	409	4, 97	436	17	463	33	490	2, 287
356	40	383	77	410	3, 181	437	5	464	82	491	1
357	73	384	1, 215	411	1	438	187	465	1, 202	492	15
358	9	385	161	412	185	439	171	466	90	493	1, 184
359	1	386	1	413	1	440	4, 135	467	95	494	2, 3
360	1, 97	387	1, 176	414	4, 137	441	1	468	2, 378	495	1
361	1, 282	388	1, 339	415	1, 276	442	138	469	113	496	3, 69
362	2, 154	389	89	416	1, 219	443	1	470	1	497	1, 200
363	51	390	2, 56	417	129	444	184	471	2, 185	498	63
364	27	391	1, 252	418	36	445	197	472	2, 212	499	1, 174
365	85	392	173	419	1	446	1, 443	473	1	500	2, 78
366	1, 65	393	33	420	1, 323	447	223	474	61		
367	93	394	59	421	1, 108	448	1, 153	475	45		
368	139	395	1, 86	422	1, 21	449	209	476	25		
369	1, 74	396	1, 113	423	2, 221	450	2, 137	477	1, 266		

Appendix E

Counts of CA over GF(q)

This appendix (referred to in section 4.1) contains counts of various restricted forms of CA. The CA are classified into six (overlapping) types, according to restrictions on the entries in the transition matrix. The type are (F^* denotes the finite field without 0):

Form 1:	$b_i = 1$	$c_i = 1$	$a_i \in F$
Form 2:	$b_i = 1$	$c_i = -1$	$a_i \in F$
Form 3:	$b_i = 1$	$c_i = 1$	$a_i \in \{0, 1\}$
Form 4:	$b_i = 1$	$c_i = -1$	$a_i \in \{0, 1, -1\}$
Form 5:	$b_i \in F^*$	$c_i \in F^*$	$a_i \in F$
Form 6:	$b_i = 1$	$c_i = -1$	$a_i \in \{0, 1\}$

Note that Forms 1 and 2 are identical for fields of characteristic 2, as are Forms 3, 4, and 6. For GF(2), all six forms are the same.

For various fields, various n, and each form, the following numbers have been calculated or counted:

- number of CA,
- number of polynomials,
- number of polynomials realised by CA,
- number of irreducible polynomials,
- number of irreducible polynomials realised by CA,

- number of primitive polynomials,
- number of primitive polynomials realised by CA,
- the number of polynomials that have from $1, 2, \ldots, 6$ or more than six CA realisations.

Table E.1: Count table, $F = \mathrm{GF}(2)$

form	n	CA	polys	CA polys	irred	CA irred	prim	CA prim	No. of realisations 1	2	3	4	5	6	>6
1	2	4	4	3	1	1	1	1	2	1	0	0	0	0	0
	3	8	8	6	2	2	2	2	4	2	0	0	0	0	0
	4	16	16	9	3	3	2	2	2	7	0	0	0	0	0
	5	32	32	18	6	6	6	6	4	14	0	0	0	0	0
	6	64	64	33	9	9	6	6	4	28	0	1	0	0	0
	7	128	128	58	18	18	18	18	8	40	0	10	0	0	0
	8	256	256	110	30	30	16	16	2	91	0	16	0	0	1
	9	512	512	214	56	56	48	48	4	166	0	44	0	0	0
	10	1024	1024	393	99	99	60	60	4	278	0	106	0	0	5
	11	2048	2048	760	186	186	176	176	8	508	0	232	0	0	12
	12	4096	4096	1435	335	335	144	144	4	904	0	483	0	0	44

Table E.2: Count table, $F = \mathrm{GF}(3)$

form	n	CA	polys	CA polys	irred	CA irred	prim	CA prim	No. of realisations 1	2	3	4	5	6	> 6
1	2	9	9	6	3	3	2	2	3	3	0	0	0	0	0
	3	27	27	18	8	6	4	3	9	9	0	0	0	0	0
	4	81	81	42	18	12	8	6	9	30	0	3	0	0	0
	5	243	243	123	48	36	22	17	27	84	0	12	0	0	0
	6	729	729	316	116	51	48	24	24	237	3	45	0	7	0
	7	2187	2187	942	312	216	156	108	63	708	18	132	0	21	0
	8	6561	6561	2595	810	462	320	184	48	1935	21	486	6	93	6
2	2	9	9	6	3	3	2	2	3	3	0	0	0	0	0
	3	27	27	16	8	6	4	3	6	9	1	0	0	0	0
	4	81	81	42	18	15	8	6	9	30	0	3	0	0	0
	5	243	243	120	48	30	22	14	15	96	0	9	0	0	0
	6	729	729	288	116	54	48	26	9	201	12	57	0	9	0
	7	2187	2187	885	312	144	156	72	57	636	0	153	0	33	6
	8	6561	6561	2529	810	390	320	152	60	1881	12	435	9	96	36
3	2	4	9	3	3	1	2	1	2	1	0	0	0	0	0
	3	8	27	6	8	2	4	2	4	2	0	0	0	0	0
	4	16	81	10	18	3	8	1	4	6	0	0	0	0	0
	5	32	243	20	48	4	22	0	8	12	0	0	0	0	0
	6	64	729	36	116	4	48	1	8	28	0	0	0	0	0
4	2	9	9	6	3	3	2	2	3	3	0	0	0	0	0
	3	27	27	16	8	6	4	3	6	9	1	0	0	0	0
	4	81	81	42	18	15	8	6	9	30	0	3	0	0	0
	5	243	243	120	48	30	22	14	15	96	0	9	0	0	0
	6	729	729	288	116	54	48	26	9	201	12	57	0	9	0
5	2	36	9	9	3	3	2	2	0	3	0	3	0	3	0
	3	432	27	27	8	8	4	4	0	0	0	0	0	0	27
	4	5184	81	81	18	18	8	8	0	0	0	0	0	0	81
6	2	4	9	3	3	2	2	1	2	1	0	0	0	0	0
	3	8	27	5	8	2	4	0	2	3	0	0	0	0	0
	4	16	81	10	18	5	8	2	4	6	0	0	0	0	0
	5	32	243	19	48	2	22	1	6	13	0	0	0	0	0
	6	64	729	36	116	7	48	3	8	28	0	0	0	0	0

Table E.3: Count table, $F = \mathrm{GF}(4)$

form	n	CA	polys	CA polys	irred	CA irred	prim	CA prim	No. of realisations 1	2	3	4	5	6	> 6
1	2	16	16	10	6	4	4	2	4	6	0	0	0	0	0
	3	64	64	36	18	10	12	8	12	20	4	0	0	0	0
	4	256	256	106	60	28	32	14	4	78	0	24	0	0	0
	5	1024	1024	436	198	118	120	72	4	356	12	56	0	8	0
	6	4096	4096	1672	670	284	288	122	8	1332	4	284	4	32	8
3	2	4	16	3	6	0	4	0	2	1	0	0	0	0	0
	3	8	64	6	18	0	12	0	4	2	0	0	0	0	0
	4	16	256	9	60	0	32	0	2	7	0	0	0	0	0
	5	32	1024	18	198	0	120	0	4	14	0	0	0	0	0
	6	64	4096	33	670	0	288	0	4	28	0	1	0	0	0
5	2	144	16	16	6	6	4	4	0	0	0	0	0	6	10
	3	5184	64	64	18	18	12	12	0	0	0	0	0	0	64

Table E.4: Count table, $F = \mathrm{GF}(5)$

form	n	CA	polys	CA polys	irred	CA irred	prim	CA prim	No. of realisations 1	2	3	4	5	6	> 6
1	2	25	25	15	10	5	4	2	5	10	0	0	0	0	0
	3	125	125	60	40	30	20	14	5	45	10	0	0	0	0
	4	625	625	270	150	55	48	16	25	200	0	35	0	10	0
	5	3125	3125	1302	624	292	280	138	60	1000	35	140	10	55	2
	6	15625	15625	6165	2580	1240	720	356	65	4735	50	1025	0	250	40
2	2	25	25	15	10	5	4	2	5	10	0	0	0	0	0
	3	125	125	60	40	30	20	16	5	45	10	0	0	0	0
	4	625	625	270	150	55	48	16	25	200	0	35	0	10	0
	5	3125	3125	1302	624	292	280	132	60	1000	35	140	10	55	2
	6	15625	15625	6165	2580	1240	720	356	65	4735	50	1025	0	250	40
3	2	4	25	3	10	0	4	0	2	1	0	0	0	0	0
	3	8	125	6	40	2	20	0	4	2	0	0	0	0	0
	4	16	625	10	150	0	48	0	4	6	0	0	0	0	0
	5	32	3125	20	624	2	280	0	8	12	0	0	0	0	0
	6	64	15625	36	2580	0	720	0	8	28	0	0	0	0	0
4	2	9	25	6	10	2	4	0	3	3	0	0	0	0	0
	3	27	125	18	40	4	20	0	9	9	0	0	0	0	0
	4	81	625	45	150	8	48	2	9	36	0	0	0	0	0
	5	243	3125	135	624	38	280	10	27	108	0	0	0	0	0
	6	729	15625	377	2580	83	720	16	25	352	0	0	0	0	0
5	2	400	25	25	10	10	4	4	0	0	0	0	0	0	25
6	2	4	25	3	10	1	4	0	2	1	0	0	0	0	0
	3	8	125	6	40	0	20	0	4	2	0	0	0	0	0
	4	16	625	10	150	2	48	0	4	6	0	0	0	0	0
	5	32	3125	20	624	6	280	2	8	12	0	0	0	0	0
	6	64	15625	36	2580	9	720	3	8	28	0	0	0	0	0

Table E.5: Count table, $F = \mathrm{GF}(7)$

form	n	CA	polys	CA polys	irred	CA irred	prim	CA prim	No. of realisations 1	2	3	4	5	6	> 6
1	2	49	49	28	21	14	8	4	7	21	0	0	0	0	0
	3	343	343	175	112	56	36	18	49	105	0	21	0	0	0
	4	2401	2401	1015	588	266	160	66	49	791	0	140	0	35	0
2	2	49	49	28	21	14	8	6	7	21	0	0	0	0	0
	3	343	343	161	112	56	36	19	14	126	14	0	7	0	0
	4	2401	2401	1015	588	336	160	84	49	791	0	140	0	35	0
3	2	4	49	3	21	1	8	0	2	1	0	0	0	0	0
	3	8	343	6	112	0	36	0	4	2	0	0	0	0	0
	4	16	2401	10	588	0	160	0	4	6	0	0	0	0	0
	5	32	16807	20	3360	4	1120	0	8	12	0	0	0	0	0
4	2	9	49	6	21	3	8	0	3	3	0	0	0	0	0
	3	27	343	18	112	2	36	0	9	9	0	0	0	0	0
	4	81	2401	45	588	18	160	0	9	36	0	0	0	0	0
	5	243	16807	135	3360	24	1120	5	27	108	0	0	0	0	0
5	2	1764	49	49	21	21	8	8	0	0	0	0	0	0	49
6	2	4	49	3	21	2	8	0	2	1	0	0	0	0	0
	3	8	343	6	112	0	36	0	4	2	0	0	0	0	0
	4	16	2401	10	588	5	160	0	4	6	0	0	0	0	0
	5	32	16807	20	3360	4	1120	2	8	12	0	0	0	0	0

Table E.6: Count table, $F = \mathrm{GF}(8)$

form	n	CA	polys	CA polys	irred	CA irred	prim	CA prim	No. of realisations 1	2	3	4	5	6	> 6
1	2	64	64	36	27	15	18	9	8	28	0	0	0	0	0
	3	512	512	240	168	72	144	60	40	152	24	24	0	0	0
	4	4096	4096	1708	1005	525	432	222	8	1420	0	216	0	64	0
3	2	4	64	3	27	0	18	0	2	1	0	0	0	0	0
	3	8	512	6	168	0	144	0	4	2	0	0	0	0	0
	4	16	4096	9	1005	0	432	0	2	7	0	0	0	0	0
	5	32	32768	18	6546	0	5400	0	4	14	0	0	0	0	0
5	2	3136	64	64	27	27	18	13	0	0	0	0	0	0	64

Table E.7: Count table, $F = \mathrm{GF}(9)$

form	n	CA	polys	CA polys	irred	CA irred	prim	CA prim	No. of realisations 1	2	3	4	5	6	> 6
1	2	81	81	45	36	18	16	8	9	36	0	0	0	0	0
	3	729	729	321	232	102	96	40	18	225	57	18	0	3	0
	4	6561	6561	2502	1620	558	640	212	81	1710	0	612	0	90	9
2	2	81	81	45	36	18	16	8	9	36	0	0	0	0	0
	3	729	729	321	232	102	96	40	18	225	57	18	0	3	0
	4	6561	6561	2502	1620	558	640	212	81	1710	0	612	0	90	9
3	2	4	81	3	36	0	16	0	2	1	0	0	0	0	0
	3	8	729	6	232	0	96	0	4	2	0	0	0	0	0
	4	16	6561	10	1620	0	640	0	4	6	0	0	0	0	0
	5	32	59049	20	11760	0	5280	0	8	12	0	0	0	0	0
4	2	9	81	6	36	0	16	0	3	3	0	0	0	0	0
	3	27	729	16	232	0	96	0	6	9	1	0	0	0	0
	4	81	6561	42	1620	0	640	0	9	30	0	3	0	0	0
	5	243	59049	120	11760	0	5280	0	15	96	0	9	0	0	0
5	2	5184	81	81	36	36	16	16	0	0	0	0	0	0	81
6	2	4	81	3	36	0	16	0	2	1	0	0	0	0	0
	3	8	729	5	232	0	96	0	2	3	0	0	0	0	0
	4	16	6561	10	1620	0	640	0	4	6	0	0	0	0	0
	5	32	59049	19	11760	0	5280	0	6	13	0	0	0	0	0

Table E.8: Count table, $F = \mathrm{GF}(11)$

form	n	CA	polys	CA polys	irred	CA irred	prim	CA prim	No. of realisations 1	2	3	4	5	6	> 6
1	2	121	121	66	55	33	16	8	11	55	0	0	0	0	0
	3	1331	1331	616	440	220	144	69	121	385	0	110	0	0	0
2	2	121	121	66	55	33	16	8	11	55	0	0	0	0	0
	3	1331	1331	583	440	176	144	59	33	429	66	44	0	11	0
3	2	4	121	3	55	0	16	0	2	1	0	0	0	0	0
	3	8	1331	6	440	2	144	0	4	2	0	0	0	0	0
	4	16	14641	10	3630	0	960	0	4	6	0	0	0	0	0
	5	32	161051	20	32208	0	12880	0	8	12	0	0	0	0	0
4	2	9	121	6	55	5	16	0	3	3	0	0	0	0	0
	3	27	1331	18	440	0	144	0	9	9	0	0	0	0	0
	4	81	14641	45	3630	10	960	0	9	36	0	0	0	0	0
	5	243	161051	135	32208	22	12880	4	27	108	0	0	0	0	0
5	2	12100	121	121	55	55	16	16	0	0	0	0	0	0	121
6	2	4	121	3	55	3	16	0	2	1	0	0	0	0	0
	3	8	1331	6	440	0	144	0	4	2	0	0	0	0	0
	4	16	14641	10	3630	2	960	0	4	6	0	0	0	0	0
	5	32	161051	20	32208	4	12880	1	8	12	0	0	0	0	0

Table E.9: Count table, $F = \mathrm{GF}(13)$

form	n	CA	polys	CA polys	irred	CA irred	prim	CA prim	No. of realisations 1	2	3	4	5	6	> 6
1	2	169	169	91	78	39	24	12	13	78	0	0	0	0	0
	3	2197	2197	936	728	338	240	112	52	663	78	130	13	0	0
2	2	169	169	91	78	39	24	12	13	78	0	0	0	0	0
	3	2197	2197	936	728	338	240	116	52	663	78	130	13	0	0
3	2	4	169	3	78	1	24	0	2	1	0	0	0	0	0
	3	8	2197	6	728	0	240	0	4	2	0	0	0	0	0
	4	16	28561	10	7098	0	1536	0	4	6	0	0	0	0	0
	5	32	371293	20	74256	2	24752	0	8	12	0	0	0	0	0
4	2	9	169	6	78	0	24	0	3	3	0	0	0	0	0
	3	27	2197	18	728	4	240	0	9	9	0	0	0	0	0
	4	81	28561	45	7098	8	1536	0	9	36	0	0	0	0	0
	5	243	371293	135	74256	20	24752	4	27	108	0	0	0	0	0
5	2	24336	169	169	78	78	24	24	0	0	0	0	0	0	169
6	2	4	169	3	78	0	24	0	2	1	0	0	0	0	0
	3	8	2197	6	728	2	240	0	4	2	0	0	0	0	0
	4	16	28561	10	7098	1	1536	0	4	6	0	0	0	0	0
	5	32	371293	20	74256	0	24752	0	8	12	0	0	0	0	0

Appendix F

Least-Cost CA over GF(q)

This appendix contains a table of minimal cost CA that have maximal length cycles (is referred to in section 4.1). It is difficult to measure the cost of a CA, as it involves assigning relative costs to addition, multiplication by constants, and storage of elements from the field GF(q). Hence, for the purposes of the table, the CA are restricted to a specialised form, with the following properties:

- $b_i = 1, \ \ 1 \leq i \leq n-1,$
- $c_i = -1, \ \ 2 \leq i \leq n,$
- $d_i \in \{0, 1\}, \ \ 1 \leq i \leq n,$
- Δ_n is primitive,
- the number of $d_i, 1 \leq i \leq n$ that are 1 is minimal, and
- q is prime.

The field size is restricted to being prime, since any CA that has the first three properties over a non-prime order field is reducible. This is because the characteristic polynomial of such a CA has coefficients that are all in the underlying prime-order field.

The table contain one minimal-cost primitive CA for each degree up to 40, for the fields of order 2, 3, 5, 7, and 11 (note that a CA with above restrictions does not

always exist). The table lists those cells i that have $d_i = 1$; for example, the entry for $n = 6$ for GF(5) is $1, 2, 5$, which represents the CA with $d = [1, 1, 0, 0, 1, 0]$. Note also that a 40-cell maximal length CA over GF(11) passes through approximately 10^{41} different states before returning to the start state.

Table F.1: Minimal-Cost Maximal-Length CA

	Field size				
n	2	3	5	7	1 1
3	1	none	none	none	none
4	1, 3	1, 4	none	none	none
5	1	1, 3	1, 3	1, 2, 3	1, 3
6	1	1, 6	1, 2, 5	1, 2, 5, 6	1, 2
7	3	1, 2, 3, 5	1, 3, 4	1, 3, 5	none
8	2, 3	1, 4, 5	1, 3, 6	1, 2, 5, 6	1, 6, 7
9	1	1, 3	1, 2, 5	1, 2, 7	1, 3
10	2, 7	1, 6	1, 2	1, 2, 3, 10	3, 4, 5
11	1	1, 9	1, 9	1, 2, 7	1, 9
12	3, 7	1, 8	3, 4	1, 3, 8	3, 6
13	5	1, 7	3, 5	3, 7, 9	1, 11
14	1	1, 6	3, 4	1, 4, 12	1, 8
15	3	1, 13	1, 7	1, 4, 7	1, 7
16	1, 15	1, 2, 3	3, 6	1, 4, 14	1, 2, 9
17	5	3, 5	1, 2, 17	1, 2, 15	5, 9
18	1, 17	1, 2	1, 6	1, 3, 18	3, 6
19	3	1, 5	1, 3, 4	1, 3, 17	7, 9
20	2, 3	1, 2, 15	1, 4	1, 2, 18	1, 2
21	1, 10	3, 5	1, 9	1, 2, 15	1, 5
22	5	1, 18	1, 2, 7	1, 6, 22	3, 8
23	1	3, 19	5, 15	1, 9, 13	5, 13
24	8, 12	1, 8	1, 18	1, 3, 24	1, 6, 11
25	9	1, 15	3, 15	1, 2, 9	3, 7
26	1	1, 6	1, 16	1, 3, 10	1, 2, 5
27	1, 20	3, 7	1, 3, 22	1, 2, 7	1, 23
28	3	3, 14	7, 8	1, 4, 16	1, 4
29	1	1, 11	1, 15	1, 2, 15	1, 11
30	1	1, 4, 25	1, 12	1, 4, 16	1, 2
31	11	1, 5	5, 15	1, 7, 13	1, 9
32	1, 15	5, 12	3, 8	1, 2, 32	1, 4, 17
33	1	3, 9	3, 13	1, 4, 5	5, 13
34	1, 19	1, 6	1, 2, 17	1, 4, 8	1, 26
35	1	1, 13	1, 31	1, 2, 15	7, 9
36	6	3, 30	3, 32	1, 2, 6	5, 20
37	9	1, 23	3, 9	1, 3, 9	7, 17
38	7	1, 2	1, 10	1, 3, 12	7, 10
39	1	1, 9	1, 7	1, 3, 9	1, 11
40	8	3, 30	17, 18	1, 6, 12	1, 30

Appendix G

Synthesis Algorithm of GF(2)

This appendix contains the complete source listing for a Maple V program that synthesises a CA from an irreducible polynomial, for GF(2). The program is in two files. The first is a "driver" file, that simply loads the main program and runs it. The second is the main program, which also contains utility routines. This appendix is referred to in section 6.7.

File "go":

```
#
interface( quiet=true, verboseproc = 0 );
printlevel := -1;
#
read(ca_alg);
#
find_ca( x^5 + x^2 + 1 );
find_ca( x^6 + x + 1 );
#
quit;
```

The main program file contains the following routines:

simp	reduces a polynomial's coefficients to GF(2)
int_to_poly	converts an integer to a polynomial, via the binary representation of the integer
calc_GCD_quotients	calculates the GCD of two polynomials, and returns the sequence of quotients
gfqtrace	calculates the trace of an element of $GF(2^n)$
calc_CA_charpoly	calculates the characteristic polynomial of a CA
calc_beta	calculates β (see algorithm description)
find_ca	main program

File "ca_alg":

```
#
#
# This routine is used to simplify expressions after multiplication.
#
simp := proc( p ) collect( ( p ), x ) mod 2; end;
#
#
# This routine convert an integer into a polynomial, by using its
# binary representation as the list of coefficients of the poly.
#
int_to_poly := proc( k )
local l, i;
  # make k into a list
  l := convert( k, base, 2 );
  # put in the x^i's
  l := [ l[i] * x^(i-1) $ i = 1..nops(l) ],
  # make l a poly, and return it
  convert( l, '+' );
end;
#
#
# This routine calculates the GCD of two polynomials.  However,
# we are only interested in the case when all quotients have degree
# one, and so the computation stops when a non degree one quotient
# is found.
# The constant terms of the quotients are returned as a list.
#
calc_GCD_quotients := proc( p, pnm1 )
local p3, p2, p1, j, quos, q;

    p3 := p;
    p2 := pnm1;
quos := NULL;

do
  # p2=0 means we're done
  if p2 = 0 then break; fi;

  # calc the quo and remainder
  p1 := Rem( p3, p2, x, 'q' ) mod 2;

  # quo no degree 1 means we're done
  if degree( q ) <> 1 then break; fi;

  # keep a record of the quotient
  if coeff( q, x, 0 ) = 0 then
quos := quos, 0;
  else
quos := quos, 1;
  fi;
```

```
  # prepare for next iteration
  p3 := p2;
  p2 := p1;

od;

[ quos ];

end;
#
#
# Trace - first arg is field element, 2nd is defining poly of field
#
gfqtrace := proc( alpha, p )
local tr, prod, n;

  n := degree( p );
  tr := alpha;
  prod := alpha;
  from 1 to n-1 do
prod := Powmod( prod, 2, p, x ) mod 2;
tr := ( tr + prod ) mod 2;
  od;

  # return the trace
  tr;

end;
#
#
#
# This function computes the characteristic polynomial of
# a CA.  The CA is passed as a list, like [0, 0, 1, 0].
#
calc_CA_charpoly := proc( d )
local pn, pnm1, pnm2, n;
  pnm2 := 0;
  pnm1 := 1;
  for n from 1 to nops( d ) do
    pn := simp( ( x + d[n] ) * pnm1 + pnm2 );
    pnm2 := pnm1;
    pnm1 := pn;
  od;
  pn;
end;
#
#
#
#
find_ca := proc( p )
local  n, dp, c, beta, gamma, pnm1, cinv, dont_need, quotients;
```

```
  lprint('------------------');
  lprint( p );

  if not Irreduc( p ) mod 2 then
    ERROR('poly must be irreducible');
  fi;

  n .= degree( p );

  # calc derivative
  dp := diff( p, x ) mod 2;

  # calc c = (alpha^2 + alpha) * p'(alpha)
  c := Rem( ( x^2 + x ) * dp, p, x ) mod 2;

  # calc 1/c
  Gcdex( p, c, x, dont_need, cinv ) mod 2;

  # calc 1 / c^2
  gamma := Powmod( cinv, 2, p, x ) mod 2;

  # find a root beta of the quadratic y^2 + y + gamma = 0
  beta := calc_beta( gamma, p, n );

  # undo the change of variable, to get root of y^2 + cy + 1 = 0
  pnm1 := Rem( beta * c, p, x ) mod 2;

  # try to calculate the CA
  quotients := calc_GCD_quotients( p, pnm1 );

  if nops( quotients ) <> n or p <> calc_CA_charpoly( quotients ) then
      lprint( 'There seems to be an problem with your program...');
  else
      lprint('CA found');
  fi;

  lprint( quotients );

end;
#
#
#
calc_beta := proc( gamma, p, n )
local theta, t, beta, gammaprod, gammasum, thetaprod;

  if type( n/2, integer ) then

    # n is even
    # find theta:
    #   pick random polys, looking for a trace 1 element
```

```
    rand_num := rand( 2^n );
    do
      t := rand_num();
      theta := int_to_poly( t );
      if gfqtrace( theta, p ) = 1 then break; fi;
    od;

    thetaprod := Powmod( theta, 2, p, x ) mod 2;
    gammaprod := gamma;
    gammasum := gamma;

    beta := 0;
    from 1 to n-1 do
      beta := simp( beta + gammasum * thetaprod );
      # get beta back into the field
      beta := Rem( beta, p, x ) mod  2;
      thetaprod := Powmod( thetaprod, 2, p, x ) mod 2;
      gammaprod := Powmod( gammaprod, 2, p, x ) mod 2;
      gammasum :=  ( gammasum + gammaprod ) mod 2;
    od;

  else
    # n is odd
    gammaprod := Powmod( gamma, 2, p, x ) mod 2;

    beta := 0;
    from 1 by 2 to n-2 do
      beta := beta + gammaprod mod 2;
      gammaprod := Powmod( gammaprod, 4, p, x ) mod 2;
    od;

  fi;

  beta;

end;
#
```

Appendix H

Synthesis Algorithm for GF(q)

This appendix contains the complete source listing for a Maple V program that implements the randomised search algorithm for synthesising a CA from an polynomial, over any finite field. This appendix is referred to in section 7.4. Polynomial arithmetic is implemented using Maple's finite field extension capability (evalgf). The program file contains the following routines:

convertToPrimeField	utility for determining type of polynomial
prim	utility for determining type of polynomial
calcGCD	calculates GCD so long as quotients are degree 1
CArecurrence	calculates CA characteristic polynomial, for checking
randSynth	main program

```
interface( quiet = true );
printlevel := -1;
readlib(evalgf);

#
# Maple routine to synthesis gfq CAs by random w.c. gcd polys
#

# Convert poly to prime field
#
# In:  poly w over gf(p^q)
# Out: poly over gf(p) of degree q^(degree(w)), irred iff w irred,
#      prim iff w prim
#
convertToPrimeField := proc( poly, p, q )
local prod, power;
  prod := poly;
  power := poly;
  from 1 to q-1 do
    power := collect(power,x);
    # next line raises each coeff to pth power
    power := map(
              proc(i,p) evalgf(lcoeff(i,x)^p,p)*x^(degree(i,x)) end,
              power, p );
```

```
    prod := evalgf(prod*power,p);
  od;
  prod;
end;


# Irreduciblily/primitivity checker
#
# In:  poly p over gf(p^q), p prime
# Out: one of the strings: 'red', 'irred', 'prim'.
#
prim := proc( polyarg, p, q )
local factors, root, size, poly, t, f, n, w;

  poly := convertToPrimeField( polyarg, p, q );

  poly := modp1( ConvertIn( poly, x ), p );
  n := modp1( Degree(poly), 2 );

  if not modp1(Irreduc(poly),p) then
    RETURN('red   ');
  fi;

  readlib('ifactor/anm1');
  factors  = readlib(ifactors)( 'ifactor/anm1'(p,n) )[2];
  size := p^n;
  root := modp1(ConvertIn(x,x),p);

  t := map(proc(z,n) iquo(n,z[1]) end,factors,size-1);
  for f in t do
    w := modp1(Powmod(root,f,poly),p);
    if w = modp1(One(),p) then RETURN('irred ') fi
  od;
  'prim  ';
end;


# Calculate the GCD of a and b using Euclid's algorithm
#
# In:  polys a and b, underlying prime field order p
# Out: triple:
#  [1] number of leading degree 1 quotients
#  [2] reversed sequence of constant terms of degree 1 quotients
#  [3] reversed sequence of remainder leading coefficients (first is 0)
#
calcGCD := proc( a, b, p )
local p1, p2, p3, quoSeq, lcoeffSeq, len, q;

  quoSeq := NULL;
  lcoeffSeq := NULL;
```

```
  p3 := a;
  p2 := b;

  len := 0;

  while p2 <> 0 do

    q := evalgf( Quo(p3, p2, x, 'p1' ), p );

    if degree( q, x ) <> 1 then break; fi;

    # remove monic leading term
    quoSeq := evalgf(x-q, p), quoSeq;
    # make negative, for CA
    lcoeffSeq := evalgf( -1 * lcoeff(p1,x), p ), lcoeffSeq;

    # normalise remainder
    if p1 <> 0 then
      p1 := evalgf( p1 / lcoeff(p1,x), p );
    fi;

    len := len + 1;
    p3 := p2;
    p2 := p1;
  od;

  [ len, [ quoSeq ], [ lcoeffSeq ] ];

end;

# Implements "improved" algorithm.
# Given an "initial-guess" degree n-1 poly b, return a degree
# n-1 poly that has PRS length at least n/2.
#
makehalflength := proc( a, paramb, p, y )
local gcdrslt, gcdlen, n, b, termToModify;

  b := paramb;
  n := degree( a, x );

  #lprint( a );
  while true do

    gcdrslt := calcGCD( a, b, p );
    gcdlen := gcdrslt[1];
    #lprint( '    ', gcdlen, b );
    if gcdlen >= n/2 then break; fi;
    termToModify := n - 2*(gcdlen+1) +1;
    b := b + Randpoly( termToModify, x, y ) mod p;
  od;
```

```
  b;
end;


# Apply CA recurrence to calc CA char poly
# In: diagonal sequence d1...dn,
#     offdiagonal sequence b_0c_1, b_1c_2, ..., b_(n-1)c_n
#     underlying prime field order p
# Out: characteristic polynomial
#
CArecurrence := proc( d, o, p )
local p1, p2, p3, i, n;
  n := nops( d );
  p1 := 0;
  p2 := 1;
  for i to n do
    p3 := evalgf( (x-d[i])*p2 - o[i]*p1, p );
    p1 := p2;
    p2 := p3;
  od;
  collect(p3,x);
end;


# Random search for a GCD worst case
#
# In:  poly a over gf(p^q), field extension y,
#      maximum number of iterations
# Out: None.  Prints success/failure message.
#
randSynth := proc( a, p, q, y, maxIterations )
local n, iteration, rslt, b;

   lprint( 'Search for CA for poly', a, prim(a,p,q) );
   n := degree( a, x );

   for iteration to maxIterations do

      # choose random degree n-1 poly
      b := x^(n-1) + Randpoly( n-2, x, y ) mod p;

      #### Improved Algorithm ####
      # Enable following line to implement.
      # b := makehalflength( a, b, p, y );

      # check if worst case
      lprint('Checking poly', b);
      rslt := calcGCD( a, b, p );

      if rslt[1] = n then
```

```
        # do a safety check
        if CArecurrence( rslt[2],rslt[3],p ) = a then
           lprint( '   check ok, diag = ', rslt[2], 'off-diag = ', rslt[3] );
        else
           lprint( ' check failed!!' );
        fi;
        break;
      else
        lprint('   failed (length', rslt[1], ')' );
      fi;

   od;

   iteration;
end;


p := 2;
q := 3;
fieldPoly := Randprime( q, _Z ) mod p;
alias( y=RootOf( fieldPoly, _Z ) mod p );
poly := x^4 + y*x^3 + (y^2+y)*x^2 + y*x + 1;
randSynth( poly, p, q, y, 20 );

lprint('-----------------------------------------');

p := 3;
q := 2;
fieldPoly := Randprime( q, _Z ) mod p;
alias( y=RootOf( fieldPoly, _Z ) mod p );
poly := x^8 + 2*y*x^3 + (y+1)*x + 1;
randSynth( poly, p, q, y, 20 );

quit;
```

Program output:

```
Search for CA for poly   x**4+y*x**3+(y**2+y)*x**2+y*x+1   red
Checking poly   x**3+(y**2+1)*x**2+(1+y+y**2)*x+1
   failed (length   3   )
Checking poly   x**3+x**2+(y**2+y)*x+y+1
   check ok, diag =    [y, y, 1, y+1]   off-diag =    [0, 1, y+1, y+1]
-----------------------------------------
Search for CA for poly   x**8+2*y*x**3+(y+1)*x+1   red
Checking poly   x**7+2*x**6+2*y*x**5+(2*y+1)*x**4+2*x**2*y+2*x*y+2*y+1
   failed (length   2   )
Checking poly   x**7+(2*y+2)*x**6+x**5+2*x**4+2*x**2*y+(2*y+2)*x
   check ok, diag =    [y+2, 1, 2*y, 2*y+2, y, y+1, 1, 2*y+2]   off-diag =    [
0, y+2, 1, y, y+1, 2*y, 2*y, 2]
```

Appendix I

PRS patterns

This appendix contains some detailed examples of polynomial remainder sequences, for various fields. This appendix is referred to in section 7.3.

The initial example in Table I.1 shows the determination of the PRSP (polynomial remainder sequence pattern) of the primitive degree 5 polynomial $p = x^5 + x^2 + 1$ over GF(2). The first column lists each of the 16 (monic) degree 4 polynomials w over GF(2). The rest of the columns contain information about the calculation of the GCD of p and w. The second column shows $\mathcal{P}^l(p, w)$ (the number of initial-degree-1-quotients). The third column shows the constant terms of these quotients. The fourth column shows the actual sequence of remainders, including the one that results in a quotient of degree greater than 1. For example, when p is divided by x^4, the quotient is $x + 0$, and the remainder is $x^2 + 1$. Clearly the quotient of x^4 divided by $x^2 + 1$ has degree 2. The PRSP of p is obtained by forming the vector where the ith entry is the number of w that have $\mathcal{P}^l(p, w) \geq i$. For this example, the vector is $[16, 8, 4, 2, 2]$.

Table I.2 shows the PRSP for various randomly selected polynomials over GF(2), GF(4), CF(8), and GF(9).

Table I.1: Detailed PRS determination

w	$\mathcal{P}^l(p,w)$	$\mathcal{P}^s(p,w)$	$\mathcal{R}^s(p,w)$
x^4	1	0	x^2+1
x^4+1	1	0	x^2+x+1
x^4+x	1	0	1
x^4+x+1	1	0	$x+1$
x^4+x^2	2	$0,1$	$x^3+x^2+1, x+1$
x^4+x^2+1	5	$0,1,1,1,1$	$x^3+x^2+x+1, x^2, x+1, 1, 0$
x^4+x^2+x	3	$0,0,0$	$x^3+1, x^2, 1$
x^4+x^2+x+1	2	$0,0$	$x^3+x+1, 1$
x^4+x^3	2	$1,0$	x^3+x^2+1, x
x^4+x^3+1	3	$1,0,1$	$x^3+x^2+x, x^2+1, 1$
x^4+x+x^3	5	$1,1,1,1,0$	$x^3+x+1, x^2+x+1, x, 1, 0$
x^4+x^3+x+1	2	$1,1$	$x^3, x+1$
$x^4+x^2+x^3$	1	1	1
$x^4+x^3+x^2+1$	1	1	x
$x^4+x^3+x^2+x$	1	1	x^2+x+1
$x^4+x^3+x^2+x+1$	1	1	x^2

Table I.2: Some PRSPs

Field	Poly/PRSP	Type
GF(2)	$x^8+x^7+x^6+x^5+x^4+x^2+1$ $[128, 64, 32, 16, 8, 4, 2, 2]$	primitive
GF(4)	$x^6+yx^5+yx^4+(y+1)x^3+(y+1)x+y+1$ $[1024, 768, 576, 432, 336, 336]$	irred
GF(4)	$x^6+x^5+x^2+(y+1)x+1$ $[1024, 768, 576, 432, 316, 316]$	irred
GF(4)	$x^6+yx^5+(y+1)x^3+x^2+yx+1$ $[1024, 768, 576, 432, 326, 326]$	irred
GF(4)	$x^6+x^5+x^3+yx^2+x+1$ $[1024, 768, 576, 432, 314, 314]$	irred
GF(4)	$x^6+yx^5+x^4+x^3+(y+1)x^2+y+1$ $[1024, 768, 576, 432, 316, 316]$	primitive
GF(4)	$x^6+(y+1)x^5+(y+1)x^4+yx^3+(y+1)x^2+(y+1)x$ $[1024, 768, 576, 424, 322, 224]$	reduc
GF(4)	$x^6+yx^5+x^3+(y+1)x+y$ $[1024, 768, 576, 424, 326, 326]$	reduc
GF(8)	$x^5+y^2x^4+yx^3+(y^2+y)x^2+y^2x+1$ $[4096, 3584, 3136, 2750, 2750]$	irred
GF(8)	$x^5+(y^2+y+1)x^4+(y^2+y)x^3+yx^2+(y^2+1)x+y$ $[4096, 3584, 3136, 2742, 2742]$	primitive
GF(9)	$x^5+(y+2)x^4+2yx^3+(y+2)x^2+(2y+2)x+y+2$ $[6561, 5832, 5184, 4592, 4592]$	primitive