367

ROYAL HOLLOWAY AND BEDFORD NEW COLLEGE LIBRARY

ASPECTS OF LOCAL LINEAR COMPLEXITY

BY

GLYN DAVID CARTER

Thesis submitted to the University of London for the degree of Doctor of Philosophy, 1989.

Royal Holloway and Bedford New College,
University of London.

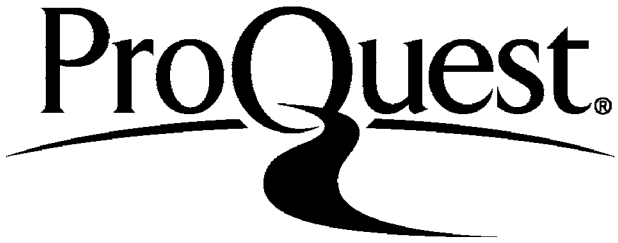
ROYAL HOLLOWAY LIBRARY
CLASS
No.

ProQuest Number: 10096244

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.



ProQuest 10096244

Published by ProQuest LLC(2016). Copyright of the Dissertation is held by the Author.

All rights reserved.
This work is protected against unauthorized copying under Title 17, United States Code.
Microform Edition © ProQuest LLC.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

ABSTRACT

The concept of linear complexity is important in cryptography, and in particular in the study of stream ciphers. There are two varieties of linear complexity; global linear complexity, which applies to infinite periodic binary sequences, and local linear complexity, which applies to binary sequences of finite length. This thesis is concerned primarily with the latter.

The local linear complexity of a finite binary sequence can be computed using the Berlekamp-Massey algorithm. Chapter 2 deals with a number of aspects of this algorithm.

The Berlekamp-Massey algorithm also yields the linear complexity profile of a binary sequence. Linear complexity profiles are discussed in Chapter 3, and a number of associated enumeration results are obtained.

In Chapter 4 it is shown that if the bits of a binary sequence satisfy certain conditions, expressible as a set of linear equations, then the linear complexity profile of the sequence will be restricted in some way. These restrictions take the form of conditions on the heights of the jumps in the profile.

The final chapter deals with the randomness testing of binary sequences. Statistical tests for randomness based on linear complexity profiles are derived, and it is demonstrated how these tests can identify the non-randomness in the sequences discussed in the preceding chapter.

CONTENTS

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Fred Piper, for all his advice and guidance in the past few years. Thanks also to Chris Mitchell and Tony Bromfield, both for their help and ideas in the past and for their comments on earlier versions of this thesis.

I am also grateful to Steve Babbage for his careful proof-reading of and helpful comments on the manuscript. Finally, I would like to thank Racal Comsec Ltd. for their support over the past four years.

CHAPTER 1

AN INTRODUCTION TO LINEAR COMPLEXITY

1.1. INTRODUCTION

The title of this thesis is "Aspects of Local Linear Complexity" and, as might be expected, local linear complexity is the central theme of the work. This first chapter introduces the reader to the concept of linear complexity, where "linear complexity" includes both "global linear complexity" and "local linear complexity". Note from the outset that global linear complexity applies to infinite periodic binary sequences, while local linear complexity relates to binary sequences of finite length.

This chapter begins with some basic definitions and a brief discussion on stream ciphers, before moving on to the subject of linear feedback shift registers. The concept of global linear complexity is then defined and discussed, and the chapter concludes with an introduction to the central theme of this thesis, local linear complexity.

1.2. STREAM CIPHERS

A cipher system provides a mechanism by which information can be disguised in such a way that it is intelligible to an authorised person but not to an unauthorised person. The information which is to be disguised is known as the plaintext (or message), the process of disguising the plaintext is known as enciphering, and the enciphered plaintext (i.e. the disguised information) is known as the ciphertext (or cryptogram).

The enciphering process is controlled by information known as a key. To encipher the plaintext, the plaintext and the key are input to an algorithm; the output from this algorithm is the ciphertext. The (enciphering) algorithm is the set of rules used to encipher the plaintext, while the (enciphering) key determines the exact transformation used. In other words, the (enciphering) key selects the enciphering transformation from the set of possibilities.

The process of retrieving the the plaintext from the ciphertext is known as deciphering, and is also controlled by a key (the deciphering key), which will often be the same as the enciphering key. Knowledge of the deciphering key allows the plaintext to be obtained

from the ciphertext, and thus this key must be kept secret from unauthorised persons if the security of the cipher system is to be maintained.

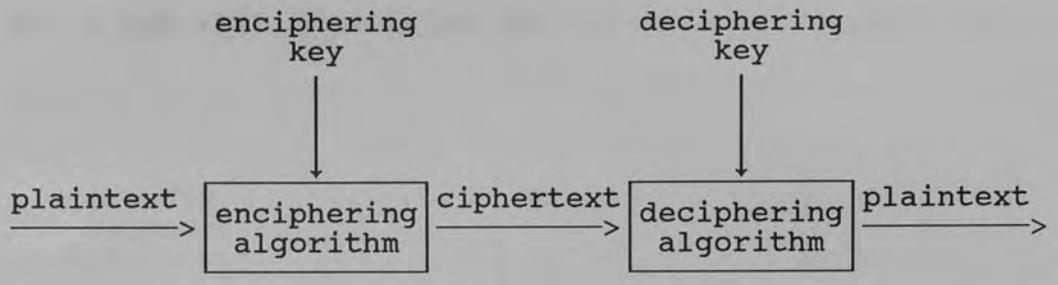


Figure 1.2.1. A cipher system

In this thesis we will be concerned with symmetric (or conventional) cipher systems, in which the deciphering key is the same as (or can be easily derived from) the enciphering key. Thus, in a symmetric cipher system the enciphering key as well as the deciphering key must be kept secret. A cipher system in which it is computationally infeasible to compute the deciphering key from the enciphering key, so that the enciphering key can be made public, is known as an asymmetric (or public key) cipher system.

One example of a symmetric cipher system is the one-time pad shown in Figure 1.2.2. In the one-time pad a message consists of at most n characters. To encipher a message $m_0m_1...m_{n-1}$ a random sequence of characters

$k_0k_1\ldots k_{n-1}$ (the key) from the same character set is mixed, character by character, with the message to produce the cryptogram $c_0c_1\ldots c_{n-1}$. To decipher the cryptogram the same random sequence $k_0k_1\ldots k_{n-1}$ is mixed with $c_0c_1\ldots c_{n-1}$, again on a character by character basis but this time using the inverse mixing operation.

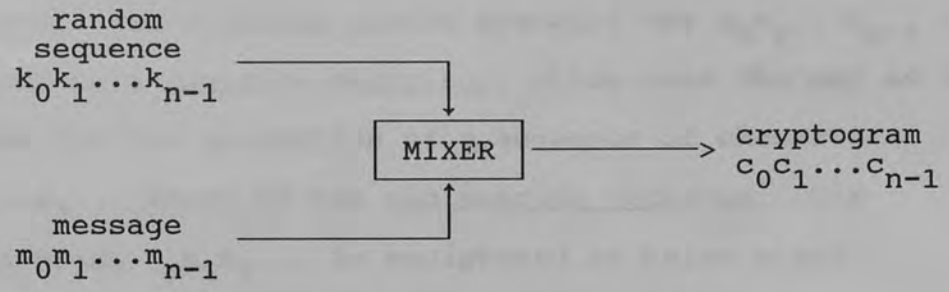


Figure 1.2.2. The one-time pad

In a common form of the one-time pad, the message, the key and the cryptogram are all binary sequences, while both mixing operations are modulo 2 addition. Thus, in this system

$$c_i = m_i + k_i \pmod 2 \quad \text{for} \quad i = 0,1,\ldots,n-1.$$

The one-time pad is theoretically unbreakable in the sense that a cryptanalyst (i.e. someone who is trying to break the system) obtains no information from intercepted ciphertext (see, for example, [2]). However, using the system often presents severe practical problems, due to the fact that the amount of

key material needed is at least as large as the amount of plaintext to be enciphered. Nevertheless, the one-time pad is still used in certain situations where the ultimate in security is required.

In applications where the one-time pad is either unnecessary, impractical or both, its principles are often imitated by using what is known as a stream cipher. In a stream cipher system a key $k_0k_1...k_{m-1}$ is input to a sequence generator, which uses the key as a seed for the generation of a sequence of characters $s_0s_1s_2...$ known as the enciphering sequence. The plaintext $m_0m_1m_2...$ is enciphered by being mixed, character by character, with the enciphering sequence. As in the one-time pad, decipherment proceeds in the same way as encipherment except that the inverse mixing operation is used. The sequence generator is often referred to as the keystream generator, a term which relates to the fact that the pseudo-random enciphering sequence (or keystream) used in a stream cipher system is emulating the random enciphering sequence (or key) used in the one-time pad.

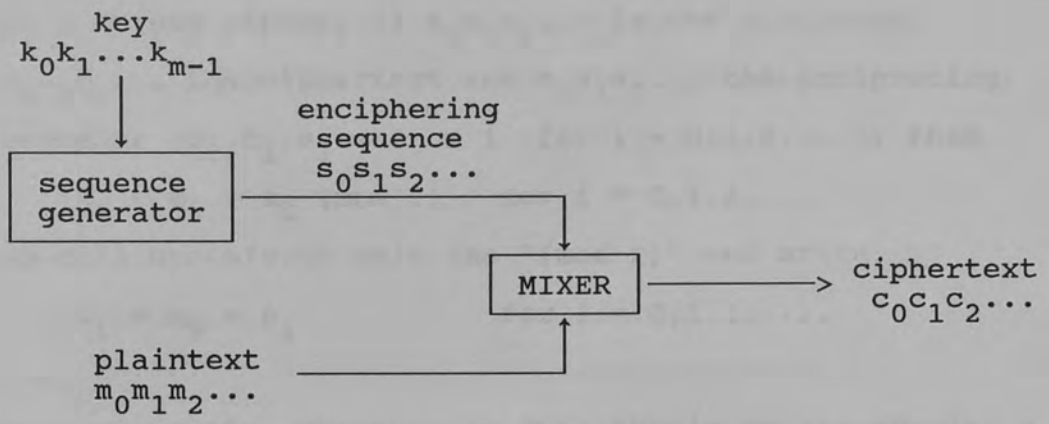


Figure 1.2.3. A stream cipher

In this thesis we will deal only with stream ciphers in which the plaintext, the ciphertext and the enciphering sequence are all binary sequences, and in which the mixing operation, both for enciphering and deciphering, is modulo 2 addition. A large proportion of the stream ciphers in use today are of this form, and some authors, when defining stream ciphers, do not mention those which use other character sets. Henceforth, when mention is made of stream ciphers they will by assumption use binary sequences and modulo 2 addition as described above.

In a stream cipher, if $m_0m_1m_2\ldots$ is the plaintext, $c_0c_1c_2\ldots$ the ciphertext and $s_0s_1s_2\ldots$ the enciphering sequence ($m_i, c_i, s_i = 0$ or 1 for $i = 0,1,2,\ldots$), then

$$c_i = m_i + s_i \pmod 2 \quad \text{for } i = 0,1,2,\ldots.$$

We will henceforth omit the "(mod 2)" and write

$$c_i = m_i + s_i \quad \text{for } i = 0,1,2,\ldots.$$

More generally, whenever in this thesis we are summing a number of bits $b_1, b_2, \ldots, b_n$ the sum will also be a bit, and so we will often write $b_1 + b_2 + \ldots + b_n$ instead of $b_1 + b_2 + \ldots + b_n \pmod 2$.

One important property of stream ciphers is that they do not propagate errors. In other words, if a single ciphertext bit is received in error, this will cause only one bit of recovered plaintext to be incorrect. This property is particularly important when the ciphertext is being transmitted over a poor quality channel, as in this situation any error propagation would increase the (already high) bit error rate and thus probably reduce the effectiveness of the system.

When designing a cipher system it is important to consider the knowledge and capabilities of a cryptanalyst trying to attack the system. It has become customary to assume that such a cryptanalyst has the following :-

(i) A complete knowledge of the cipher system

(ii) A considerable amount of ciphertext

(iii) The plaintext corresponding to a certain amount of ciphertext.

These assumptions are often referred to as the worst case conditions. The exact definitons of "considerable" and "certain" will depend on the situation under consideration.

Under the worst case conditions given above, a cryptanalyst trying to attack a stream cipher will know the enciphering and deciphering algorithms (i.e. he will have a complete knowledge of the sequence generator) and will know part of the enciphering sequence (since he will have the plaintext corresponding to a certain amount of ciphertext). Thus it is important that knowledge of a section of the enciphering sequence does not enable a cryptanalyst to determine the entire sequence (or the key, which would allow the entire sequence to be generated).

In most practical stream ciphers the enciphering sequence will be periodic. An infinite binary sequence $s_0 s_1 s_2 \ldots$ is said to be periodic if there exists an integer $p > 0$ such that $s_{i+p} = s_i$ for $i = 0,1,2,\ldots$, and the least such value of p is called the period of the

sequence. Any subsequence of the form $s_k s_{k+1} \cdots s_{k+p-1}$ for some integer $k \geq 0$ is called a cycle of $s_0 s_1 s_2 \cdots$, and $s_0 s_1 \cdots s_{p-1}$ is called the generating cycle.

If a cryptanalyst obtains p consecutive bits from an enciphering sequence of period p then he will be able to construct the entire sequence. Thus, it is essential for the security of a stream cipher that the enciphering sequence has a large period. As an absolute minimum, this period should be at least as long as any message which will be enciphered. However, a large period does not ensure that the entire enciphering sequence cannot be obtained by a cryptanalyst from a relatively small section of the sequence. We will return to this subject in the subsequent sections of this chapter.

A further requirement of an enciphering sequence is that it should "appear to be" random. The aim of this requirement is two-fold; firstly, to ensure that any statistical properties of the plaintext are not reflected in the ciphertext, and secondly, to prevent a cryptanalyst who knows a section of the enciphering sequence from successfully predicting subsequent bits of it. No infinite sequence generated by a sequence generator using a finite key can be truly random; the best that can be hoped for is that any subsequence of length less than or equal to the period should be

"indistinguishable" from a random sequence of the same length. Such a sequence is loosely termed a pseudo-random sequence.

One possible definition of a pseudo-random binary sequence was suggested by Golomb [7]. He defines a PN-sequence ("pseudo-noise" sequence) to be a binary sequence of period p which satisfies the following three randomness postulates :-

(i) The number of ones and the number of zeros in a cycle of the sequence differ by no more than 1.

(ii) In a cycle of the sequence, half the runs have length 1, a quarter have length 2, an eighth have length 3, and in general, for any k for which there are at least 2^{k+1} runs in a cycle, $(\frac{1}{2})^k$ of the runs have length k. Moreover, for each of these lengths there are equally many runs of ones and zeros.

(A run of length r is a string of r identical bits which is both preceded and succeeded by the opposite bit)

(iii) The sequence has a two-valued autocorrelation function.

(The autocorrelation function $C(\tau)$ of a binary sequence $s_0 s_1 s_2 \ldots$ of period p is defined by

$$C(\tau) := \frac{A(\tau) - D(\tau)}{p},$$

where $A(\tau)$ is the number of bit positions in

which $s_0 s_1 \ldots s_{p-1}$ and $s_\tau s_{\tau+1} \ldots s_{\tau+p-1}$ agree, and $D(\tau)$ is the number of positions in which they disagree)

We will return to Golomb's postulates in Section 1.3.

The above randomness postulates apply to a complete cycle of an enciphering sequence. Indeed, in many cases the only theoretical results that can be proved about the randomness properties of the output from a sequence generator will be concerned with "global" randomness properties (i.e. properties of an entire enciphering sequence). However, in most stream cipher systems a complete cycle of an enciphering sequence will never be used. Thus, although the global randomness properties should not be overlooked, they are probably of less importance than the "local" randomness properties of the sequence (i.e. properties of shorter subsequences of the sequence). Often the only way to test the local randomness of an enciphering sequence is by applying statistical tests to sections of the sequence. The statistical testing of binary sequences will be dealt with in more detail in Chapter 5.

1.3. LINEAR FEEDBACK SHIFT REGISTERS

A device often used in the generation of binary sequences is the linear feedback shift register. In this section we discuss linear feedback shift registers and the properties of the sequences which they generate. We begin by giving a more general definition of a shift register :-

An n-stage shift register consists of n storage elements or stages $S_0, S_1, \ldots, S_{n-1}$ which are connected in series. The contents of these stages at a given time t is known as the state of the register and is denoted by $s_0(t), s_1(t), \ldots, s_{n-1}(t)$. Periodically, the contents of the register are shifted so that the contents of S_i are transferred into S_{i-1} for $i = 1,2,\ldots,n-1$. At the same time, the new contents of S_{n-1} is formed by combining the old contents of $S_0, S_1, \ldots S_{n-1}$ using a feedback function f. In other words,

$$s_i(t+1) = \begin{cases} s_{i+1}(t) & (i = 0,1,\ldots,n-2) \\ f(s_0(t), s_1(t), \ldots, s_{n-1}(t)) & (i = n-1) \end{cases}$$

Figure 1.3.1 below shows such a shift register. Note that, in the general case shown, an input is modulo 2 added to the result of the feedback function before it enters S_{n-1}. However, in this thesis we will always take this input to be all zeros, and so we will omit it

in future. The output sequence generated by the register is the infinite binary sequence $s_0 s_1 s_2 \ldots$, where $s_t = s_0(t)$ for $t = 0,1,2,\ldots$. The state of the register $s_0(0), s_1(0), \ldots, s_{n-1}(0)$ at time 0 is known as the initial state of the register.

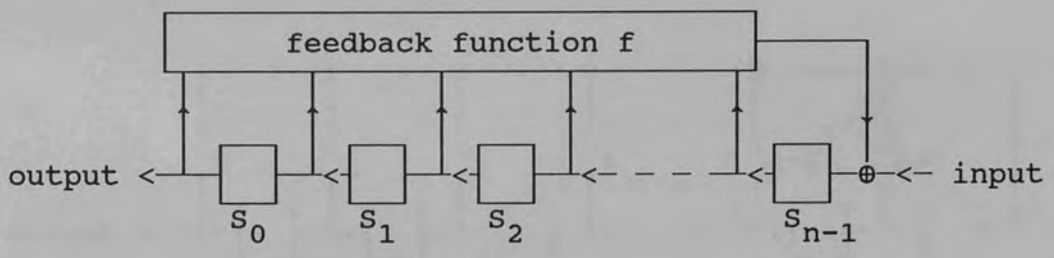


Figure 1.3.1. An n-stage shift register

In a linear feedback shift register or LFSR the feedback function f is a linear function. If we denote the state of the register by $s_0, s_1, \ldots, s_{n-1}$, then f can be written in the form $f(s_0, s_1, \ldots, s_{n-1}) = c_0 s_0 + c_1 s_1 + \ldots$ $\ldots + c_{n-1} s_{n-1}$, where $c_i = 0$ or 1 for $i = 0,1,\ldots,n-1$ and the additions (as always in the case of bits) are modulo 2. The constants $c_0, c_1, \ldots, c_{n-1}$ are called the feedback coefficients.

Figure 1.3.2 below shows an n-stage linear feedback shift register. The feedback coefficients $c_0, c_1, \ldots, c_{n-1}$ are represented by switches; if $c_i = 1$ then the corresponding switch is closed, while if $c_i = 0$ then the switch is open. Note that the "⊕" symbol indicates modulo 2 addition.

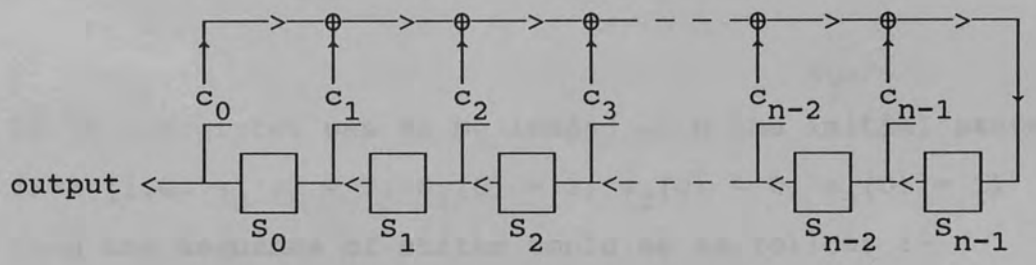


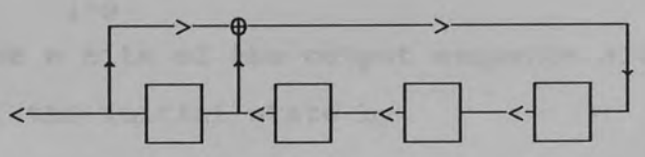
Figure 1.3.2. A linear feedback shift register

Any linear feedback shift register can be uniquely described by what is known as its characteristic polynomial. The characteristic polynomial f(x) of the n-stage LFSR with feedback coefficients $c_0, c_1 \ldots, c_{n-1}$ is defined by

$$f(x) := c_0 + c_1 x + c_2 x^2 + \ldots + c_{n-1} x^{n-1} + x^n.$$

Example 1.3.1.

As an example of a linear feedback shift register, consider the 4-stage LFSR with characteristic polynomial $1 + x + x^4$, as illustrated below :-



If this register was to be loaded with the initial state 1101 (i.e. $s_0(0) = 1$, $s_1(0) = 1$, $s_2(0) = 0$, $s_3(0) = 1$) then the sequence of states would be as follows :-

t=0	1101	t=6	1110	t=12	1001
t=1	1010	t=7	1100	t=13	0011
t=2	0101	t=8	1000	t=14	0110
t=3	1011	t=9	0001	t=15	1101
t=4	0111	t=10	0010	t=16	1010
t=5	1111	t=11	0100	etc.	

and the output sequence generated would be 11010111100010011....

In the case of a linear feedback shift register, the successive states of the register are derived from the initial state using the following relationships :-

$$s_{n-1}(t+1) = \sum_{i=0}^{n-1} c_i s_i(t)$$
$$s_k(t+1) = s_{k+1}(t) \qquad (k = 0,1,\ldots,n-2) \qquad (1.3.1)$$

Hence it can be seen that the output sequence $s_0 s_1 s_2 \ldots$ generated by the n-stage LFSR with characteristic polynomial $c_0 + c_1 x + c_2 x^2 + \ldots + c_{n-1} x^{n-1} + x^n$ must satisfy the following linear recurrence relation :-

$$s_{t+n} = \sum_{i=0}^{n-1} c_i s_{t+i} \qquad (t = 0,1,2,\ldots) \qquad (1.3.2)$$

The first n bits of the output sequence are given in terms of the initial state by

$$s_t = s_t(0) \qquad (t = 0,1,\ldots,n-1) \qquad (1.3.3)$$

Consider an n-stage LFSR with a characteristic polynomial in which the coefficient c_0 of x^0 is equal to 0. Let d be the least integer such that $c_d = 1$ (i.e. $c_0 = c_1 = \ldots = c_{d-1} = 0$, $c_d = 1$). Then equation (1.3.2) can be rewritten as

$$s_{t+n} = \sum_{i=d}^{n-1} c_i s_{t+i} \qquad (t = 0,1,2,\ldots)$$

$$\Rightarrow \quad s_{t+n-d} = \sum_{i=0}^{n-d-1} c_{i+d} s_{t+i} \qquad (t = d,d+1,d+2,\ldots)$$

Hence, $s_d s_{d+1} s_{d+2} \ldots$ can be considered as the output sequence from an (n-d)-stage LFSR with characteristic polynomial $c_d + c_{d+1} x + \ldots + c_{n-1} x^{n-d-1} + x^{n-d}$. The situation is illustrated in Figure 1.3.3 below.

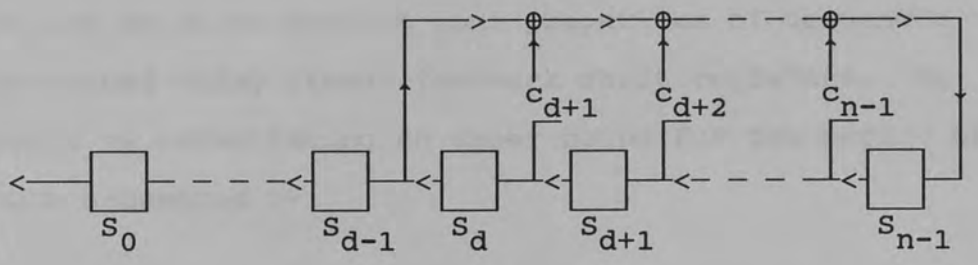


Figure 1.3.3. An n-stage LFSR with $c_0 = 0$

For this reason, in many situations it is convenient to consider only linear feedback shift registers for which the coefficient c_0 of x^0 in the characteristic polynomial is equal to 1 (see, for example, [2]). In this thesis, however, we will not make this restriction; when considering local linear complexity, for instance, we will be interested in LFSRs with $c_0 = 0$ as well as those with $c_0 = 1$ (see Section 1.5).

The set of all infinite binary sequences which can be generated using the LFSR with characteristic polynomial $f(x)$ is often denoted by $\Omega(f)$. It can be shown (see [16]) that, if $f(x)$ has degree n, then $\Omega(f)$ is an n-dimensional vector space.

We now go on to look at some properties of sequences generated using linear feedback shift registers. We begin by establishing an upper bound for the period of such sequences :-

Theorem 1.3.1.

Any infinite binary sequence generated using an n-stage LFSR must be ultimately periodic, with period at most 2^n-1.

(A sequence $s_0s_1s_2\ldots$ is ultimately periodic if $s_ks_{k+1}s_{k+2}\ldots$ is periodic for some integer k)

Proof

If any n-stage LFSR is loaded with the state consisting of n zeros, then all subsequent states will also consist of n zeros and so all subsequent output bits generated will be zeros. Hence if, during the generation of a sequence using an LFSR, the state of the register ever becomes all zeros, then the resulting sequence will be ultimately periodic with period 1.

Now consider the generation of an infinite sequence using an n-stage LFSR, and suppose that the state of the register never becomes all zeros. As soon as the state of the register is one which has already occurred the sequence of states (and hence the output sequence) will begin to repeat. But since there are only 2^n-1 possible

states which the register can take, the sequence of states (and hence the output sequence) must be ultimately periodic, with period at most 2^n-1.

□

Theorem 1.3.2.

Any infinite binary sequence which can be generated using an n-stage LFSR with $c_0 = 1$ is periodic, with period at most 2^n-1.

Proof

Let $s_0s_1s_2\ldots$ be an infinite sequence generated using an n-stage LFSR with $c_0 = 1$. Then Theorem 1.3.1 says that, for some integer $k \geqslant 0$, $s_ks_{k+1}s_{k+2}\ldots$ is periodic, with period at most 2^n-1. It is shown in [2] that the condition $c_0 = 1$ is sufficient to ensure that $k = 0$.

□

A sequence of period 2^n-1 which can be generated using an n-stage LFSR is known as an m-sequence. For instance, the sequence generated by loading the initial state 1101 (or any other non-zero state) into the 4-stage LFSR with characteristic polynomial $1 + x + x^4$ (see Example 1.3.1) has period 15 and so is an m-sequence. The following result yields a characterisation of m-sequences :-

Theorem 1.3.3.

Let f(x) be a polynomial over GF(2) of degree n, and let $s_0 s_1 s_2 \ldots$ be a non-zero sequence generated using the n-stage LFSR with characteristic polynomial f(x). Then $s_0 s_1 s_2 \ldots$ has period 2^n-1 if and only if f(x) is primitive.

(An irreducible polynomial of degree n over GF(2) is said to be primitive if 2^n-1 is the least positive integer e such that f(x) divides $1 + x^e$)

Proof

See [2].

□

Theorem 1.3.3 shows that by carefully choosing our characteristic polynomials we can generate binary sequences with large periods (e.g. m-sequences) from relatively short linear feedback shift registers. In addition, m-sequences can also be shown to have a number of desirable randomness properties. These properties are summarised in the following theorem :-

Theorem 1.3.4.

Any m-sequence is a PN-sequence.

Proof

See [2].

□

Although it was for many years widely conjectured to be true, it is not true that every PN-sequence is either an m-sequence or the complement of an m-sequence (i.e. an m-sequence with all its zeros changed to ones and ones to zeros). There are known to be at least two binary sequences of period 127 which are PN-sequences but neither m-sequences nor complements of m-sequences (see [4]).

Theorem 1.3.4 states that any m-sequence conforms to one possible definition of a pseudo-random sequence (i.e. it satisfies Golomb's randomness postulates, as given in Section 1.2), while by definition an m-sequence of large period can be generated using a relatively short LFSR. This, coupled with the fact that LFSRs are easily implemented, might suggest that LFSRs with primitive characteristic polynomials are good candidates for use as sequence generators in stream cipher systems. However, a system with such a sequence generator can be easily attacked under the worst case conditions, as will be shown in the next section.

1.4. GLOBAL LINEAR COMPLEXITY

In the previous section we discussed the generation of binary sequences using linear feedback shift registers. In fact, any periodic binary sequence can be generated in such a way, as the following theorem shows :-

Theorem 1.4.1.

Any periodic binary sequence can be generated using a linear feedback shift register.

Proof

Consider a periodic binary sequence $s_0 s_1 s_2 \ldots$ of period p. If $s_0 s_1 \ldots s_{p-1}$ is loaded into the p-stage LFSR with characteristic polynomial $1 + x^p$, then the output sequence will consist of repetitions of $s_0 s_1 \ldots s_{p-1}$, and hence $s_0 s_1 s_2 \ldots$ will be generated.

□

The global linear complexity of a periodic binary sequence $s_0 s_1 s_2 \ldots$ is defined to be the length of the shortest LFSR on which the sequence can be generated. Theorem 1.4.1 shows that, for any periodic binary sequence, the global linear complexity exists.

Moreover, it can be seen from the proof of the theorem that if $s_0 s_1 s_2 \ldots$ has period p then its global linear complexity cannot exceed p.

What we refer to here as "global linear complexity" is referred to by a variety of names in the literature, and in particular by "linear equivalence", "recursion length", or simply "linear complexity". However, each of these terms invites confusion between "global linear complexity" as defined above, which applies to infinite periodic sequences, and "local linear complexity", which we shall meet in Section 1.5 and applies to finite binary sequences. Hence, we will always retain the word "global".

The global linear complexity of a sequence can be computed from knowledge of its period and its generating cycle. Before we describe how this can be done, however, we introduce some further terminology and state an interesting intermediate result :-

If $s_0 s_1 s_2 \ldots$ is any infinite sequence, then the generating function $S(x)$ of $s_0 s_1 s_2 \ldots$ is defined to be

$$S(x) := \sum_{i=0}^{\infty} s_i x^i .$$

If $s_0 s_1 s_2 \ldots$ is a binary sequence of period p then the period polynomial s(x) of $s_0 s_1 s_2 \ldots$ is defined by

$$s(x) := s_0 + s_1 x + s_2 x^2 + \ldots + s_{p-1} x^{p-1}.$$

It can easily be shown that the generating function and period polynomial of a periodic binary sequence are related by the following equation :-

$$S(x) = s(x) / (1 + x^p) \qquad (1.4.1)$$

Now let $h(x) = h_0 + h_1 x + h_2 x^2 + \ldots + h_{d-1} x^{d-1} + x^d$ be any polynomial (of degree d) over GF(2). Then the reciprocal polynomial $h^*(x)$ of h(x) is defined by

$$h^*(x) := x^d . h(\tfrac{1}{x})$$
$$= 1 + h_{d-1} x + \ldots + h_1 x^{d-1} + h_0 x^d.$$

Note that h(x) and $h^*(x)$ have the same degree (i.e. deg h(x) = deg $h^*(x)$ = d) if and only if $h_0 = 1$; otherwise $h^*(x)$ has degree less than d.

Theorem 1.4.2.

Let S(x) be the generating function of a binary sequence $s_0 s_1 s_2 \ldots$ which can be generated using the n-stage LFSR with characteristic polynomial f(x). Then

$$S(x) = \phi(x) / f^*(x)$$

for some polynomial $\phi(x)$ of degree less than n.

Proof

See [2].

Note that the result quoted in [2] insists that the coefficient c_0 of x^0 in f(x) is equal to 1. However, the proof follows through without this restriction.

□

The proof of Theorem 1.4.2 yields an explicit expression for $\phi(x)$ in terms of the feedback coefficients and the initial state of the LFSR used in the generation of the sequence (see Section 2.6). Furthermore, for any given polynomial f(x) of degree n there is a one-to-one correspondence between the 2^n sequences which can be generated using the n-stage LFSR with characteristic polynomial f(x) (i.e. one for each initial state) and the 2^n polynomials $\phi(x)$ over GF(2) with degree less than n.

We now show how the global linear complexity of a periodic binary sequence can be found :-

Theorem 1.4.3.

Let $s_0 s_1 s_2 \ldots$ be a non-zero periodic binary sequence with period p and period polynomial s(x). Then the global linear complexity L of $s_0 s_1 s_2 \ldots$ is the degree of the polynomial f(x) given by

$$f(x) = \left(\frac{1+x^p}{(s(x), 1+x^p)} \right)^*$$

Furthermore, f(x) is the characteristic polynomial of the unique L-stage LFSR on which $s_0 s_1 s_2 \ldots$ can be generated (i.e. f(x) is the minimal polynomial of the sequence).

Proof

See [2].

(Note that (a(x),b(x)) denotes the greatest common divisor of the polynomials a(x) and b(x))

□

Example 1.4.1.

As an example of the use of Theorem 1.4.3, consider the sequence 101001010010100..., which has period 5 and period polynomial $1 + x^2$. Therefore, in this example we have

$$\frac{1+x^p}{(s(x), 1+x^p)} = \frac{1+x^5}{(1+x^2, 1+x^5)}$$

$$= \frac{(1+x).(1+x+x^2+x^3+x^4)}{1+x}$$

$$= 1 + x + x^2 + x^3 + x^4,$$

and thus, by Theorem 1.4.3, the sequence 101001010010100... has global linear complexity 4, and the unique 4-stage LFSR on which this sequence can be generated has characteristic polynomial $1 + x + x^2 + x^3 + x^4$.

We now come on to consider the subject raised at the end of Section 1.3, namely the security (or lack of security) of a stream cipher system in which the sequence generator is a linear feedback shift register. In fact, the system we shall consider will be more general than this; we will consider any sequence generator which produces a periodic enciphering sequence. We begin by demonstrating how the relationships between successive states of an LFSR can be expressed in terms of vectors and matrices :-

Consider an n-stage LFSR with characteristic polynomial $f(x) = c_0 + c_1x + c_2x^2 + \ldots + c_{n-1}x^{n-1} + x^n$. Suppose that this register is loaded with the initial state $s_0(0), s_1(0), \ldots, s_{n-1}(0)$, and let $s_0s_1s_2\ldots$ be the

output sequence generated. Then, from equations (1.3.1) and (1.3.3), the state vector $\underline{s}_t$ at time t is the n-vector given by

$$\underline{s}_t := \begin{pmatrix} s_0(t) \\ s_1(t) \\ \vdots \\ s_{n-1}(t) \end{pmatrix} = \begin{pmatrix} s_t \\ s_{t+1} \\ \vdots \\ s_{t+n-1} \end{pmatrix} \qquad (1.4.2)$$

Further, from equation (1.3.2) it can be seen that

$$\underline{s}_{t+1} = \begin{pmatrix} s_{t+1} \\ s_{t+2} \\ \vdots \\ s_{t+n-1} \\ s_{t+n} \end{pmatrix} = \begin{pmatrix} s_{t+1} \\ s_{t+2} \\ \vdots \\ s_{t+n-1} \\ c_0 s_t + \ldots + c_{n-1} s_{t+n-1} \end{pmatrix}$$

and so $\underline{s}_{t+1} = C.\underline{s}_t \qquad (1.4.3)$

where C is the n×n matrix given by

$$C = \begin{pmatrix} 0 & 1 & 0 & \ldots. & 0 \\ 0 & 0 & 1 & \ldots. & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \ldots. & 1 \\ c_0 & c_1 & c_2 & \ldots. & c_{n-1} \end{pmatrix}$$

The next theorem shows how the state vectors of a sequence generated using an LFSR can be used to test the global linear complexity of the sequence :-

Theorem 1.4.4.

Let $s_0 s_1 s_2 \ldots$ be a binary sequence generated using an n-stage LFSR. Then the following three statements are equivalent :-

(i) $s_0 s_1 s_2 \ldots$ has global linear complexity n;

(ii) there exists a set of n consecutive state vectors which are linearly independent;

(iii) every set of n consecutive state vectors is linearly independent.

Proof

See [2].

□

Example 1.4.2.

Consider again the sequence 101001010010100... of Example 1.4.1. By Theorem 1.4.1, this sequence can be generated using the 5-stage LFSR with characteristic polynomial $1 + x^5$. However, the 5 possible state vectors (1,0,1,0,0), (0,1,0,0,1), (1,0,0,1,0), (0,0,1,0,1) and (0,1,0,1,0) sum to zero, and so it is impossible to find 5 consecutive state vectors which are linearly independent. Thus, by Theorem 1.4.4, the sequence cannot have global linear complexity 5, and therefore it must be possible to generate it on a 4-stage LFSR. Now, the 4 consecutive state vectors (1,0,0,1), (0,0,1,0), (0,1,0,1) and (1,0,1,0) are

linearly independent, and thus, reapplying Theorem 1.4.4, the global linear complexity of 101001010010100... must be 4.

As previously stated, Theorem 1.4.3 yields a method for finding the global linear complexity and the minimal polynomial of a periodic binary sequence. However, this method requires the entire generating cycle of the sequence to be known, and so computing the global linear complexity in this way will be infeasible if the sequence has a sufficiently large period.

On the other hand, if a sequence $s_0s_1s_2...$ which can be generated using an L-stage LFSR has global linear complexity L, then this fact can be established using Theorem 1.4.4 from knowledge of 2L-1 consecutive bits of the sequence. Although, unlike Theorem 1.4.3, Theorem 1.4.4 does not yield the minimal polynomial of the sequence, this minimal polynomial can be obtained from knowledge of 2L consecutive bits of the sequence, as we shall now demonstrate :-

Let $s_0s_1s_2...$ be a periodic binary sequence with global linear complexity L, and suppose that 2L consecutive bits $s_ks_{k+1}...s_{k+2L-1}$ of the sequence are known. By equation (1.3.2) we have

$$\underline{s}_{k+L} = \begin{pmatrix} s_{k+L} \\ s_{k+L+1} \\ \vdots \\ s_{k+2L-1} \end{pmatrix} = \begin{pmatrix} c_0 s_k + \ldots + c_{L-1} s_{k+L-1} \\ c_0 s_{k+1} + \ldots + c_{L-1} s_{k+L} \\ \vdots \\ c_0 s_{k+L-1} + \ldots + c_{L-1} s_{k+2L-2} \end{pmatrix}$$

and so $\underline{s}_{k+L} = S.\underline{c}$ (1.4.4)

where S is the L×L matrix given by

$$S = \begin{pmatrix} s_k & s_{k+1} & \cdots & s_{k+L-1} \\ s_{k+1} & s_{k+2} & \cdots & s_{k+L} \\ \vdots & \vdots & & \vdots \\ s_{k+L-1} & s_{k+L} & \cdots & s_{k+2L-2} \end{pmatrix}$$

and $\underline{c} = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{L-1} \end{pmatrix}$ is the vector of feedback coefficients for an L-stage LFSR on which $s_0 s_1 s_2 \ldots$ can be generated.

By Theorem 1.4.4 the rows of S are linearly independent, and thus S is non-singular and invertible. And, by assumption, the elements of $\underline{s}_{k+L}$ and S are all known. Hence we can compute the feedback coefficients $c_0, c_1, \ldots, c_{L-1}$ of an L-stage LFSR on which $s_0 s_1 s_2 \ldots$ can be generated by computing the vector $\underline{c} = S^{-1}.\underline{s}_{k+L}$; these are the coefficients of the (unique) minimal polynomial of the sequence. Having obtained the feedback coefficients and a particular state $s_k s_{k+1} \ldots s_{k+L-1}$ of the L-stage LFSR on which $s_0 s_1 s_2 \ldots$ can be generated we can obviously generate all subsequent output bits

$s_k s_{k+1} s_{k+2} \ldots$. Moreover, from Theorem 1.4.3 it can be seen that the minimal polynomial of a periodic sequence always divides $1 + x^p$, and so the coefficient c_0 of x^0 is always equal to 1. This in turn means that the matrix C in expression (1.4.3) is invertible, so that (1.4.3) can be rewritten as

$$\underline{s}_t = C^{-1} . \underline{s}_{t+1} \qquad (1.4.5)$$

where $C = \begin{pmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \ldots & 1 \\ 1 & c_1 & c_2 & \ldots & c_{L-1} \end{pmatrix}$

If the minimal polynomial of $s_0 s_1 s_2 \ldots$ and a particular state of the corresponding LFSR are known (as is the case here) then expression (1.4.5) can be used to find the previous states of the register, and thus the remaining k bits $s_0 s_1 \ldots s_{k-1}$ of the sequence can be found. Thus we have proved the following result :-

Theorem 1.4.5.

Let $s_0 s_1 s_2 \ldots$ be a periodic binary sequence with global linear complexity L. Then the entire sequence $s_0 s_1 s_2 \ldots$ can be obtained from knowledge of any 2L consecutive bits.

□

Let us now look at Theorem 1.4.5 from a cryptanalytic viewpoint. Suppose that a cryptanalyst trying to attack a stream cipher system has obtained the plaintext corresponding to m bits of ciphertext. Then this attacker will be able to obtain m bits of enciphering sequence by simply modulo 2 adding the plaintext and the ciphertext. If $m \geqslant 2L$, where L is the global linear complexity of the enciphering sequence, then Theorem 1.4.5 says that the entire enciphering sequence can now be computed. The preceding discussion shows that this process involves the inversion of an L×L binary matrix, a relatively easy task which can be performed in $O(L^{2.81})$ operations (see [1]). Hence, if the global linear complexity of the enciphering sequence is not sufficiently large then the cipher system is vulnerable to a "known plaintext attack".

In particular, consider a stream cipher system in which the sequence generator consists of a single n-stage linear feedback shift register. Then, by definition, the enciphering sequence will have global linear complexity at most n, and so the attack described above will almost certainly be feasible.

1.5. LOCAL LINEAR COMPLEXITY

In Section 1.4 we discussed global linear complexity. The global linear complexity of a periodic binary sequence was defined to be the length of the shortest LFSR on which the entire infinite sequence could be generated. However, in practice a message will have only finite length, and so in a stream cipher system only a finite section of enciphering sequence will be used to encipher a given message. Moreover, in the majority of cases the length of the sections used will be significantly shorter than the period of the enciphering sequence. For this reason the "global" properties of the enciphering sequence are perhaps less important than its "local" properties, and hence we define the "local linear complexity" of a finite binary sequence as follows :-

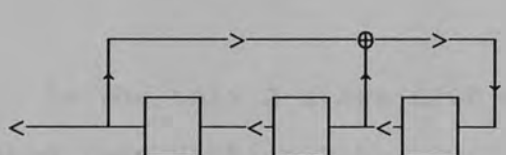
The local linear complexity of an n-bit binary sequence $s_0 s_1 \ldots s_{n-1}$ is the length of the shortest linear feedback shift register on which $s_0 s_1 \ldots s_{n-1}$ can be generated. Note that the local linear complexity of the sequence 00...0 consisting of n zeros is defined to be 0 (since the input to any LFSR is taken to be all zeros; see Section 1.3).

If we consider any n-bit sequence $s_0 s_1 \ldots s_{n-1}$, then this sequence can be generated using any n-bit LFSR by loading the register with $s_0 s_1 \ldots s_{n-1}$ as its initial

state and taking the first n output bits. Thus, the local linear complexity exists for any n-bit sequence and is at most n. Further, if we consider any periodic binary sequence $s'_0s'_1s'_2\ldots$ for which $s'_0s'_1\ldots s'_{n-1} = s_0s_1\ldots s_{n-1}$, then we know that any LFSR which can be used to generate $s'_0s'_1s'_2\ldots$ can also be used to generate $s_0s_1\ldots s_{n-1}$. Hence, the local linear complexity of $s_0s_1\ldots s_{n-1}$ must be less than or equal to the global linear complexity of any infinite periodic sequence which has $s_0s_1\ldots s_{n-1}$ as its first n bits.

Example 1.5.1.

As an example, consider the 6-bit sequence 101001. These are the first 6 bits of the periodic sequence 101001010010100... of Examples 1.4.1 and 1.4.2, which has global linear complexity 4. Hence, 101001 has local linear complexity at most 4. However, 101001 can also be generated using the 3-stage LFSR illustrated below (i.e. the LFSR with characteristic polynomial $1 + x^2 + x^3$) by loading it with the initial state 101.

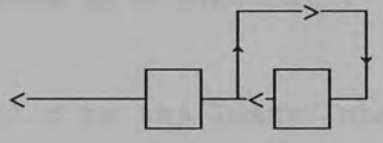


There are no 2-stage LFSRs on which 101001 can be generated, and thus this 6-bit sequence has local linear complexity 3.

In the case of an infinite periodic binary sequence, the shortest LFSR on which the sequence can be generated must have a characteristic polynomial in which the coefficient c_0 of x^0 is equal to 1. In the case of a finite binary sequence, however, this is not the case, as the following example illustrates :-

Example 1.5.2.

Consider the 7-bit sequence 0111111. The shortest LFSR on which this sequence can be generated is the 2-stage LFSR with characteristic polynomial $x + x^2$, as illustrated below :-



Note that this is the only 2-stage LFSR on which 0111111 can be generated (see Section 2.5).

We now consider what happens when we apply cryptanalytic techniques similar to those described in Section 1.4 to a message which has been stream ciphered using an n-bit

section of enciphering sequence $s_0s_1\ldots s_{n-1}$ with local linear complexity L. By the definition of local linear complexity, there must exist an L-stage LFSR (R, say) which could be used to generate $s_0s_1\ldots s_{n-1}$. Thus, $s_0s_1\ldots s_{n-1}$ must be the first n bits of an infinite binary sequence $s_0s_1s_2\ldots$ that could be generated using R. By Theorem 1.3.1, $s_0s_1s_2\ldots$ must be ultimately periodic.

Suppose that $f(x) = c_0 + c_1x + \ldots + c_{L-1}x^{L-1} + x^L$ is the characteristic polynomial of the L-stage LFSR R. If $c_0 = 1$ then $s_0s_1s_2\ldots$ is periodic (by Theorem 1.3.2) and has global linear complexity L (since $s_0s_1s_2\ldots$ can be generated using an L-stage LFSR and $s_0s_1\ldots s_{n-1}$ cannot be generated using a shorter LFSR), and so, by Theorem 1.4.5, $s_0s_1\ldots s_{n-1}$ can be obtained from knowledge of any 2L consecutive bits of the enciphering sequence (i.e. from knowledge of the plaintext corresponding to 2L consecutive bits of ciphertext).

If $c_0 = 0$, let d be the least integer such that $c_d = 1$ (i.e. $c_0 = c_1 = \ldots = c_{d-1} = 0$, $c_d = 1$). Then, from Section 1.3, $s_ds_{d+1}\ldots s_{n-1}$ can be generated using the (n-L)-stage LFSR with characteristic polynomial $1 + c_{d+1}x + \ldots + c_{L-1}x^{L-d-1} + x^{L-d}$. Thus, by a similar argument to that used in the previous paragraph for the

case $c_0 = 1$, $s_d s_{d+1} \cdots s_{n-1}$ can be obtained from knowledge of 2L-2d consecutive bits of the enciphering sequence.

By combining the two cases above it can be seen that if an n-bit message is stream ciphered using a section of enciphering sequence with local linear complexity L, then a cryptanalyst who knows the plaintext corresponding to any 2L consecutive bits of ciphertext will be able to decipher the entire message (with the possible exception of the first few bits).

CHAPTER 2

THE BERLEKAMP-MASSEY ALGORITHM

2.1. INTRODUCTION

In Section 1.5 we defined the local linear complexity of a finite binary sequence to be the length of the shortest linear feedback shift register on which the sequence could be generated. In this chapter we consider an algorithm for computing the local linear complexity L of an n-bit sequence $s_0 s_1 \ldots s_{n-1}$. This algorithm is known as the Berlekamp-Massey algorithm. The Berlekamp-Massey algorithm also yields an L-stage register on which $s_0 s_1 \ldots s_{n-1}$ can be generated.

The L-stage linear feedback shift register produced by the Berlekamp-Massey algorithm is described in terms of its connection polynomial. The L-stage linear feedback shift register in Figure 2.1.1 below has connection polynomial C(x) defined by

$$C(x) := 1 + c_{L-1}x + c_{L-2}x^2 + \ldots + c_1 x^{L-1} + c_0 x^L$$

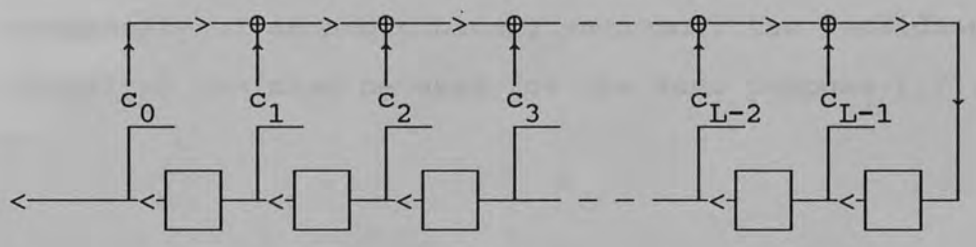


Figure 2.1.1. An L-stage linear feedback shift register

The connection polynomial C(x) as defined above should be compared with the characteristic polynomial f(x) of the same LFSR, defined by

$$f(x) := c_0 + c_1x + c_2x^2 + \ldots + c_{L-1}x^{L-1} + x^L$$

Notice that the degree of the characteristic polynomial is always equal to the length L of the register, whereas the degree of the connection polynomial is equal to L if and only if $c_0 = 1$. Notice also that $C(x) = f^*(x)$, but $f(x) = C^*(x)$ if and only if $c_0 = 1$. (Recall from Section 1.4 that the polynomials $f^*(x)$ and $C^*(x)$ are the reciprocal polynomials of f(x) and C(x) respectively).

The Berlekamp-Massey algorithm was first described as a means of finding the shortest LFSR capable of generating a given finite binary sequence by Massey in [11], although an equivalent algorithm had earlier been published by Berlekamp [3] for decoding BCH codes.

Note that the Berlekamp-Massey algorithm is not the only algorithm that can be used to compute the local linear complexity of an n-bit binary sequence; the Euclidean algorithm can also be used for the same purpose [17].

2.2. SOME PRELIMINARY RESULTS

Before we discuss the Berlekamp-Massey algorithm we first prove some preliminary results which will be used when establishing the validity of the algorithm :-

Lemma 2.2.1.

Let L(N) be the local linear complexity of the N-bit sequence $s_0 s_1 \ldots s_{N-1}$, and let L(N+1) be the local linear complexity of the N+1-bit sequence $s_0 s_1 \ldots s_N$.
Then $L(N+1) \geqslant L(N)$.

Proof

Since L(N+1) is the local linear complexity of $s_0 s_1 \ldots s_N$, there exists an L(N+1)-stage LFSR on which $s_0 s_1 \ldots s_N$ can be generated.

But $s_0 s_1 \ldots s_{N-1}$ can also be generated using this register, and L(N) is the length of the smallest LFSR on which $s_0 s_1 \ldots s_{N-1}$ can be generated.
Therefore $L(N) \leqslant L(N+1)$.

□

Lemma 2.2.2.

Let L(N) be the local linear complexity of the N-bit sequence $s_0 s_1 \ldots s_{N-1}$, let L(N+1) be the local linear complexity of the N+1-bit sequence $s_0 s_1 \ldots s_N$, and suppose that there exists an LFSR of length L(N) which can be used to generate $s_0 s_1 \ldots s_{N-1}$ but not $s_0 s_1 \ldots s_N$. Then $L(N+1) \geqslant N+1 - L(N)$.

Proof

If N = 0 then the conditions of the lemma imply that $s_0 = 1$, so that

$L(N+1) = L(1) = 1 = 1 - L(0) = N+1 - L(N)$.

Thus we can assume that $N \geqslant 1$.

From Section 1.5 we know that $L(N) \leqslant N$.

If $L(N) = N$ then $N+1 - L(N) = 1$,

and $L(N+1) \geqslant L(N) = N \geqslant 1$.

Thus the result holds when L(N) = N.

Now consider the case $L(N) < N$.

Let $C_N(x) = 1 + c_{L(N)-1}x + \ldots + c_0 x^{L(N)}$ be the connection polynomial of an L(N)-stage LFSR that can be used to generate $s_0 s_1 \ldots s_{N-1}$ but not $s_0 s_1 \ldots s_N$, let $C_{N+1}(x) = 1 + c'_{L(N+1)-1}x + \ldots + c'_0 x^{L(N+1)}$ be the

connection polynomial of an L(N+1)-stage LFSR that can be used to generate $s_0 s_1 \ldots s_N$, and suppose that

$$L(N+1) < N+1 - L(N), \quad \text{so} \quad L(N+1) \leqslant N - L(N). \qquad (2.2.1)$$

By hypothesis, since the L(N)-stage LFSR with connection polynomial $C_N(x)$ can be used to generate $s_0 s_1 \ldots s_{N-1}$ but not $s_0 s_1 \ldots s_N$, and the L(N+1)-stage LFSR with connection polynomial $C_{N+1}(x)$ can be used to generate $s_0 s_1 \ldots s_N$, we have (from equation (1.3.2))

$$\sum_{i=0}^{L(N)-1} c_i s_{j-L(N)+i} = \begin{cases} s_j & (j = L(N), \ldots, N-1) \\ s_j + 1 & (j = N) \end{cases} \qquad (2.2.2)$$

$$\sum_{k=0}^{L(N+1)-1} c'_k s_{j-L(N+1)+k} = s_j \quad (j = L(N+1), \ldots, N) \qquad (2.2.3)$$

Therefore

$$\sum_{i=0}^{L(N)-1} c_i s_{N-L(N)+i}$$

$$= \sum_{i=0}^{L(N)-1} c_i \sum_{k=0}^{L(N+1)-1} c'_k s_{N-L(N)-L(N+1)+i+k}$$

[by (2.2.3), since $L(N+1) \leqslant N-L(N)$ by (2.2.1), and so $\{N-L(N)+i : i = 0,1,\ldots,L(N)-1\} = \{N-L(N), N-L(N)+1, \ldots, N-1\}$ is a subset of $\{L(N+1), L(N+1)+1, \ldots, N\}$.]

$$= \sum_{k=0}^{L(N+1)-1} c'_k \sum_{i=0}^{L(N)-1} c_i s_{N-L(N)-L(N+1)+i+k}$$

$$= \sum_{k=0}^{L(N+1)-1} c'_k s_{N-L(N+1)+k}$$

[by (2.2.2), since $L(N) \leqslant N-L(N+1)$ by (2.2.1),
and so $\{N-L(N+1)+k : k = 0,1,\ldots,L(N+1)-1\}$
$= \{N-L(N+1), N-L(N+1)+1, \ldots, N-1\}$
is a subset of $\{L(N), L(N)+1, \ldots, N-1\}$.]

$$\Rightarrow \sum_{i=0}^{L(N)-1} c_i s_{N-L(N)+i} = s_N \qquad \text{(by (2.2.3))}$$

But (2.2.2) gives $\sum_{i=0}^{L(N)-1} c_i s_{N-L(N)+i} = s_N + 1$

and thus we have a contradiction,

proving that $L(N+1) \geqslant N+1 - L(N)$ in the case

$L(N) < N$.

□

Combining Lemmas 2.2.1 and 2.2.2 above we obtain the following result :-

Corollary 2.2.3.

Let $L(N)$ be the local linear complexity of the N-bit sequence $s_0 s_1 \ldots s_{N-1}$, let $L(N+1)$ be the local linear complexity of the N+1-bit sequence $s_0 s_1 \ldots s_N$, and suppose that there exists an LFSR of length $L(N)$ that can be used to generate $s_0 s_1 \ldots s_{N-1}$ but not $s_0 s_1 \ldots s_N$. Then $L(N+1) \geqslant \max(L(N), N+1-L(N))$.

□

2.3. THE BERLEKAMP-MASSEY ALGORITHM

As previously mentioned, the Berlekamp-Massey algorithm computes the local linear complexity L(n) of an n-bit binary sequence $s_0s_1\ldots s_{n-1}$ and the connection polynomial $C_n(x)$ of an L(n)-stage LFSR on which $s_0s_1\ldots s_{n-1}$ can be generated. As a consequence of the way in which the algorithm works it also yields, for values of k from 0 up to n-1, the local linear complexity L(k) of the first k bits $s_0s_1\ldots s_{k-1}$ of the sequence and the connection polynomial $C_k(x) = 1 + c^{(k)}_{L(k)-1}x + \ldots + c^{(k)}_1 x^{L(k)-1} + c^{(k)}_0 x^{L(k)}$ of an L(k)-stage LFSR on which $s_0s_1\ldots s_{k-1}$ can be generated.

The aim of this section is to introduce the reader to the Berlekamp-Massey algorithm. A statement of the algorithm will be given, and this will be followed by a brief discussion on the algorithm and an example of its use. The proof of the validity of the algorithm, however, will be left until Section 2.4.

The above-mentioned statement of the algorithm can be found overleaf.

The Berlekamp-Massey algorithm

$L(0) := 0$

$C_0(x) := 1$

$B(x) := 1$

$a := 1$

FOR $k := 1$ TO n DO

$$d := s_{k-1} + \sum_{i=0}^{L(k-1)-1} c_i^{(k-1)} s_{k-L(k-1)+i-1}$$

IF $d = 0$ THEN

$L(k) := L(k-1)$

$C_k(x) := C_{k-1}(x)$

$a := a + 1$

END IF

IF $d = 1$ AND $2.L(k-1) > k-1$ THEN

$L(k) := L(k-1)$

$C_k(x) := C_{k-1}(x) + x^a B(x)$

$a := a + 1$

END IF

IF $d = 1$ AND $2.L(k-1) \leqslant k-1$ THEN

$L(k) := k - L(k-1)$

$C_k(x) := C_{k-1}(x) + x^a B(x)$

$B(x) := C_{k-1}(x)$

$a := 1$

END IF

END DO

As can be seen, the algorithm is an iterative one, in which one iteration of a loop is performed for each bit of the sequence. After k-1 iterations of the loop the local linear complexity L(k-1) of the first k-1 bits $s_0 s_1 \cdots s_{k-2}$ of the sequence and the connection polynomial $C_{k-1}(x)$ of an L(k-1)-stage LFSR that could be used to generate $s_0 s_1 \cdots s_{k-2}$ will be known. On the next pass through the loop the k^{th} bit s_{k-1} of the sequence will be considered, and knowledge gained during previous iterations of the loop, together with knowledge of the value of s_{k-1}, will be used to compute the local linear complexity L(k) of $s_0 s_1 \cdots s_{k-1}$ and the connection polynomial $C_k(x)$ of an L(k)-stage LFSR that could be used to generate this subsequence.

Each iteration of the loop begins by computing a value d, where

$$d := s_{k-1} + \sum_{i=0}^{L(k-1)-1} c_i^{(k-1)} s_{k-L(k-1)+i-1}$$

If L(k-1) = 0 then d will simply be set to the value of s_{k-1}, the k^{th} bit in the sequence.
If L(k-1) > 0, let $u_0 u_1 u_2 \cdots$ be the sequence of output from the L(k-1)-stage LFSR with connection polynomial $C_{k-1}(x)$ when loaded with the initial state $s_0 s_1 \cdots s_{L(k-1)-1}$. Then we know that $u_j = s_j$ for

$j = 0,1,\ldots,k-2$, since the above-mentioned LFSR generates $s_0 s_1 \ldots s_{k-2}$. Therefore,

$$\sum_{i=0}^{L(k-1)-1} c_i^{(k-1)} s_{k-L(k-1)+i-1}$$

$$= \sum_{i=0}^{L(k-1)-1} c_i^{(k-1)} u_{k-L(k-1)+i-1} = u_{k-1},$$

and thus $d = s_{k-1} + u_{k-1}$, so that d will be set to 0 or 1 according to whether or not s_{k-1} would be the k^{th} bit output from the L(k-1)-stage LFSR with connection polynomial $C_{k-1}(x)$ if it was initially loaded with $s_0 s_1 \ldots s_{L(k-1)-1}$.

Depending on the values of d and L(k-1) one of three different actions is then taken. If $d = 0$ then the L(k-1)-stage LFSR with connection polynomial $C_{k-1}(x)$ can be used to generate $s_0 s_1 \ldots s_{k-1}$, and so the local linear complexity and the connection polynomial remain unchanged (i.e. $L(k) := L(k-1)$ and $C_k(x) := C_{k-1}(x)$).

If $d = 1$ then the above-mentioned LFSR cannot be used to generate $s_0 s_1 \ldots s_{k-1}$, and so a new LFSR must be found. If $L(k-1) > \frac{k-1}{2}$ then the local linear complexity remains unchanged (i.e. an L(k-1)-stage LFSR can be found which can be used to generate not only $s_0 s_1 \ldots s_{k-2}$ but also $s_0 s_1 \ldots s_{k-1}$). However, if $d = 1$

and $L(k-1) \leqslant \frac{k-1}{2}$ then the local linear complexity increases (i.e. $L(k) := k - L(k-1)$); no $L(k-1)$-stage LFSR exists which could be used to generate $s_0 s_1 \ldots s_{k-1}$.

It can be seen from the above remarks that if $L(k-1) > \frac{k-1}{2}$ then the local linear complexity cannot increase on the k^{th} iteration of the loop (we already know from Lemma 2.2.1 that it cannot decrease). If $L(k-1) \leqslant \frac{k-1}{2}$ then the local linear complexity will increase or remain unchanged according to whether $d = 1$ or 0. Also, if the local linear complexity does increase on the k^{th} iteration of the loop then its new value must be $k - L(k-1)$. These results follow from the proof of the validity of the Berlekamp-Massey algorithm (Section 2.4).

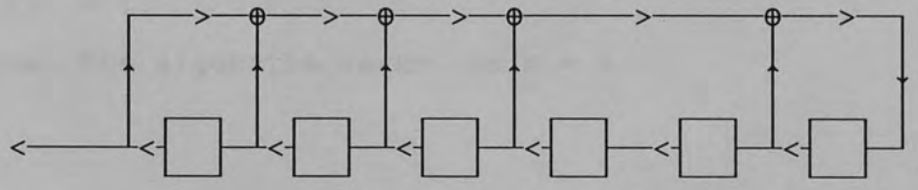
Example 2.3.1.

As an example of the use of the Berlekamp-Massey algorithm, consider the 11-bit sequence $s_0 s_1 \ldots s_{10} =$ 01111011010. We demonstrate how the algorithm might be used to compute the local linear complexity of this sequence by means of the table shown below. The "k = 0" row of the table contains the initial values assigned to

the variables used in the algorithm, while the "k = K" row (K = 1,2,...,11) shows the assignments made to these variables during the K^{th} iteration of the loop.

k	d	L(k)	$C_k(x)$	B(x)	a
0	-	0	1	1	1
1	0	0	1	1	2
2	1	2	$1+x^2$	1	1
3	1	2	$1+x+x^2$	1	2
4	1	2	$1+x$	1	3
5	0	2	$1+x$	1	4
6	1	4	$1+x+x^4$	$1+x$	1
7	0	4	$1+x+x^4$	$1+x$	2
8	1	4	$1+x+x^2+x^3+x^4$	$1+x$	3
9	1	5	$1+x+x^2$	$1+x+x^2+x^3+x^4$	1
10	0	5	$1+x+x^2$	$1+x+x^2+x^3+x^4$	2
11	1	6	$1+x+x^3+x^4+x^5+x^6$	$1+x+x^2$	1

Hence the 11-bit sequence 01111011010 can be generated using the 6-stage LFSR with connection polynomial $1 + x + x^3 + x^4 + x^5 + x^6$ (i.e. the LFSR shown below).



2.4. PROOF OF THE VALIDITY OF THE ALGORITHM

We now justify the use of the Berlekamp-Massey algorithm in computing the local linear complexity L(n) of the n-bit binary sequence $s_0 s_1 \ldots s_{n-1}$ and the connection polynomial $C_n(x)$ of an L(n)-stage LFSR on which $s_0 s_1 \ldots s_{n-1}$ can be generated.

First we show that the algorithm works for n = 1.
In this case only one iteration of the loop is required (the iteration with k = 1), and k = 1 gives d := s_0.
If $s_0 = 0$ then d := 0, giving

$L(1) := L(0) = 0$

$C_1(x) := C_0(x) = 1$

If $s_0 = 1$ then d := 1, and since $2.L(0) \leqslant 0$ this gives

$L(1) := 1 - L(0) = 1$

$C_1(x) := C_0(x) + x^1.B(x) = 1 + x$

The 1-bit sequence $s_0 = 1$ obviously has local linear complexity L(1) = 1, and can be generated using any 1-stage LFSR, including the one with connection polynomial $C_1(x) = 1 + x$.
And, by definition, the 1-bit sequence $s_0 = 0$ has local linear complexity L(1) = 0 and connection polynomial $C_1(x) = 1$.
Thus, the algorithm works for n = 1.

Now assume that the algorithm works for n = N. We will show that the algorithm also works for n = N+1.
Since the algorithm works for n = N we know that, for any N-bit sequence $s_0s_1\ldots s_{N-1}$, it gives the correct local linear complexity L(N) and the connection polynomial $C_N(x)$ of an L(N)-stage LFSR on which $s_0s_1\ldots s_{N-1}$ can be generated.

If the L(N)-stage LFSR with connection polynomial $C_N(x)$ also generates $s_0s_1\ldots s_N$ then d := 0 and so the algorithm gives

$L(N+1) := L(N)$

$C_{N+1}(x) := C_N(x)$

Hence the algorithm works for n = N+1 in this case, since $C_N(x)$ is the connection polynomial of an L(N)-stage LFSR on which $s_0s_1\ldots s_N$ can be generated, and L(N) must be the local linear complexity of $s_0s_1\ldots s_N$ since, by Lemma 2.2.1, the local linear complexity of $s_0s_1\ldots s_N$ is no less than that of $s_0s_1\ldots s_{N-1}$.

If the L(N)-stage LFSR with connection polynomial $C_N(x)$ cannot be used to generate $s_0s_1\ldots s_N$ then d := 1.
In this case the algorithm gives L(N+1) := L(N) if $2.L(N) > N$ and L(N+1) := N+1 - L(N) if $2.L(N) \leqslant N$.
i.e. the algorithm gives

$L(N+1) := \max(L(N), N+1-L(N))$.

The algorithm also gives

$$C_{N+1}(x) := C_N(x) + x^a B(x)$$

regardless of whether $2.L(N) > N$ or $2.L(N) \leqslant N$.

If no change has occurred in the local linear complexity during the first N iterations of the loop then $L(N) = 0$, $C_N(x) = 1$, $B(x) = 1$ and $a = N+1$, since $B(x)$ only alters when the local linear complexity alters and a is incremented once on each pass through the loop unless the local linear complexity changes.
Thus, in this case the algorithm gives

$$L(N+1) := N+1$$

$$C_{N+1}(x) := 1 + x^{N+1}$$

But $L(N) = 0$ implies that $s_0 s_1 \ldots s_{N-1} = 00\ldots0$, and $s_N = 1$ since the L(N)-stage LFSR with connection polynomial $C_N(x)$ cannot be used to generate $s_0 s_1 \ldots s_N$, so that $s_0 s_1 \ldots s_N = 00\ldots01$. Any N+1-bit sequence can be generated using any N+1-stage LFSR, and 00...01 cannot be generated using a shorter register (of length M, say), since to generate $s_0 s_1 \ldots s_{M-1}$ the register would have to be loaded initially with M zeros and so its output would always be zeros. Therefore the local linear complexity of $s_0 s_1 \ldots s_N$ is N+1 and the algorithm works for n = N+1 in this case.

Having dealt with the above case we can assume that at least one change has occurred in the local linear complexity during the first N iterations of the loop.

Suppose that the last such change occurred during the iteration with k = M. Then after the first N iterations of the loop we have $B(x) = C_{M-1}(x)$ and a = N-M+1, so that the algorithm gives

$$C_{N+1}(x) := C_N(x) + x^{N-M+1}C_{M-1}(x)$$

By Corollary 2.2.3 we know that

$L(N+1) \geqslant \max(L(N), N+1-L(N))$.

Therefore it is sufficient to prove that the L-stage LFSR with connection polynomial $C_N(x) + x^{N-M+1}C_{M-1}(x)$ can be used to generate $s_0 s_1 \ldots s_N$, where

$L := \max(L(N), N+1-L(N))$.

[Note that $\deg(C_N(x) + x^{N-M+1}C_{M-1}(x))$

$\leqslant \max(\deg(C_N(x)), N-M+1+\deg(C_{M-1}(x)))$

$\leqslant \max(L(N), N-M+1+L(M-1)) = L$, since

$L(N) = L(M) = M-L(M-1)$ and $L = \max(L(N), N+1-L(N))$.

Thus $C_N(x) + x^{N-M+1}C_{M-1}(x)$ is a valid connection polynomial for an L-stage LFSR.]

Let $C_N(x) + x^{N-M+1}C_{M-1}(x) =$

$$1 + c_{L-1}x + \ldots + c_1 x^{L-1} + c_0 x^L.$$

Then it is sufficient to prove that

$$\sum_{i=0}^{L-1} c_i s_{j-L+i} = s_j \qquad \text{for } j = L, L+1, \ldots, N$$

If we let

$C_N(x) = 1 + c'_{L(N)-1}x + \ldots + c'_1 x^{L(N)-1} + c'_0 x^{L(N)}$, and

$C_{M-1}(x) = 1 + c''_{L(M-1)-1}x + \ldots + c''_1 x^{L(M-1)-1} + c''_0 x^{L(M-1)}$

then $1 + c_{L-1}x + \ldots + c_0x^L$

$$= 1 + c'_{L(N)-1}x + \ldots + c'_0x^{L(N)}$$
$$+ x^{N-M+1} + c''_{L(M-1)-1}x^{N-M+2} + \ldots + c''_0x^{N-M+1+L(M-1)}$$

and so $\sum_{i=0}^{L-1} c_i s_{j-L+i}$

$$= \sum_{i=L-L(N)}^{L-1} c'_{i-L+L(N)} s_{j-L+i}$$
$$+ \sum_{i=L-N+M-1-L(M-1)}^{L-N+M-2} c''_{i-L+N-M+1+L(M-1)} s_{j-L+i} + s_{j-N+M-1}$$

$$= \sum_{i=0}^{L(N)-1} c'_i s_{j-L(N)+i}$$
$$+ \sum_{i=0}^{L(M-1)-1} c''_i s_{j-N+M-1-L(M-1)+i} + s_{j-N+M-1}$$

But the L(N)-stage LFSR with connection polynomial $C_N(x)$ can be used to generate $s_0s_1\ldots s_{N-1}$ but not $s_0s_1\ldots s_N$, and therefore

$$\sum_{i=0}^{L(N)-1} c'_i s_{j-L(N)+i} = \begin{cases} s_j & (j = L(N),\ldots,N-1) \\ s_j + 1 & (j = N) \end{cases}$$

And the L(M-1)-stage LFSR with connection polynomial $C_{M-1}(x)$ can be used to generate $s_0s_1\ldots s_{M-2}$ but not $s_0s_1\ldots s_{M-1}$, so

$$\sum_{i=0}^{L(M-1)-1} c''_i s_{j-N+M-1-L(M-1)+i} = \begin{cases} s_{j-N+M-1} \\ \quad (j = L(M-1)+N-M+1, \\ \qquad\qquad \ldots,N-1) \\ s_{j-N+M-1} + 1 \quad (j = N) \end{cases}$$

But $L(N) \leqslant L$ and $L(M-1)+N-M+1 = N+1-L(N) \leqslant L$, since $L = \max(L(N), N+1-L(N))$ and $L(N) = M-L(M-1)$.

Therefore

$$\sum_{i=0}^{L-1} c_i s_{j-L+i} = \begin{cases} s_j + s_{j-N+M-1} + s_{j-N+M-1} & (j = L,\ldots,N-1) \\ s_j + 1 + s_{j-N+M-1} + 1 + s_{j-N+M-1} & (j = N) \end{cases}$$

$$= s_j \quad (j = L, L+1, \ldots, N)$$

and hence we have shown that the L-stage LFSR with connection polynomial $C_N(x) + x^{N-M+1}C_{M-1}(x)$ can be used to generate $s_0 s_1 \ldots s_N$, as required.

□

Thus we have shown that the Berlekamp-Massey algorithm works (i.e. it correctly computes the local linear complexity L(n) of $s_0 s_1 \ldots s_{n-1}$ and the connection polynomial of an L(n)-stage LFSR on which $s_0 s_1 \ldots s_{n-1}$ can be generated).

2.5. THE MULTIPLICITY OF THE CONNECTION POLYNOMIAL

In Section 2.3 it was noted that if, in the k^{th} iteration of the loop in the Berlekamp-Massey algorithm, $d = 1$ and $L(k-1) > \frac{k-1}{2}$ then, although the $L(k-1)$-stage LFSR with connection polynomial $C_{k-1}(x)$ cannot be used to generate $s_0 s_1 \ldots s_{k-1}$, another $L(k-1)$-stage LFSR can be found which can be used to generate $s_0 s_1 \ldots s_{k-1}$. Since both registers can be used to generate $s_0 s_1 \ldots s_{k-2}$, the register given by the algorithm is not the only $L(k-1)$-stage LFSR on which $s_0 s_1 \ldots s_{k-2}$ can be generated. More precisely, the following is true :-

Theorem 2.5.1

Let $L(N)$ be the local linear complexity of the N-bit sequence $s_0 s_1 \ldots s_{N-1}$, and let $C_N(x)$, $B(x)$ and a be the values produced when the Berlekamp-Massey algorithm is applied to $s_0 s_1 \ldots s_{N-1}$.

Then

(i) If $L(N) \leqslant \frac{N}{2}$ then $C_N(x)$ is the connection polynomial of the unique $L(N)$-stage LFSR on which $s_0 s_1 \ldots s_{N-1}$ can be generated.

(ii) If $L(N) > \frac{N}{2}$ then there are $2^{2L(N)-N}$ $L(N)$-stage LFSRs on which $s_0 s_1 \ldots s_{N-1}$ can be generated, whose connection polynomials are given by

$$\{C_N(x) + q(x).x^a B(x) : \deg(q(x)) < 2L(N)-N\}.$$

Proof

(i) Suppose that an L(N)-stage LFSR with connection polynomial $C(x) = 1 + c_{L(N)-1}x + \ldots + c_0 x^{L(N)}$ can be used to generate $s_0 s_1 \ldots s_{N-1}$, where $L(N) \leqslant \frac{N}{2}$. Then

$$\sum_{i=0}^{L(N)-1} c_i s_{j-L(N)+i} = s_j \quad (j = L(N), \ldots, N-1)$$

and in particular

$$\sum_{i=0}^{L(N)-1} c_i s_{j-L(N)+i} = s_j \quad (j = L(N), \ldots, 2L(N)-1)$$

i.e. $S.\underline{c} = \underline{s}_{L(N)}$ (2.5.1)

where S is the $L(N) \times L(N)$ matrix given by

$$S = \begin{pmatrix} s_0 & s_1 & \cdots & s_{L(N)-1} \\ s_1 & s_2 & \cdots & s_{L(N)} \\ \vdots & \vdots & & \vdots \\ s_{L(N)-1} & s_{L(N)} & \cdots & s_{2L(N)-2} \end{pmatrix}$$

and $\underline{c}$ and $\underline{s}_{L(N)}$ are the vectors given by

$$\underline{c} = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{L(N)-1} \end{pmatrix} \quad \text{and} \quad \underline{s}_{L(N)} = \begin{pmatrix} s_{L(N)} \\ s_{L(N)+1} \\ \vdots \\ s_{2L(N)-1} \end{pmatrix}$$

Let $s_0 s_1 s_2 \ldots$ be the infinite binary sequence generated by loading the L(N)-stage LFSR with connection polynomial C(x) with the initial state $s_0 s_1 \ldots s_{L(N)-1}$.

Then $s_0s_1s_2\ldots$ has global linear complexity L(N), since $s_0s_1\ldots s_{N-1}$ has local linear complexity L(N) and so cannot be generated using a shorter register.

But the rows of the matrix S in (2.5.1) represent L(N) successive states of the above-mentioned LFSR when used to generate the sequence $s_0s_1s_2\ldots$. Hence, by Theorem 1.4.4, the rows of S are linearly independent and so S is invertible.

Therefore equation (2.5.1) has a unique solution $\underline{c}$ given by $\underline{c} = S^{-1}.\underline{s}_{L(N)}$, and hence there exists a unique L(N)-stage LFSR which can be used to generate $s_0s_1\ldots s_{N-1}$. But the Berlekamp-Massey algorithm gives the connection polynomial $C_N(x)$ of an L(N)-stage LFSR on which the sequence can be generated. Therefore $C_N(x)$ is the connection polynomial of the unique L(N)-stage LFSR which can be used to generate $s_0s_1\ldots s_{N-1}$.

(ii) Now suppose that $L(N) > \frac{N}{2}$, so $2.L(N) > N$. If we append 2L(N)-N bits to the sequence $s_0s_1\ldots s_{N-1}$ to obtain the 2L(N)-bit sequence $s'_0s'_1\ldots s'_{2L(N)-1}$ (where $s'_0s'_1\ldots s'_{N-1} = s_0s_1\ldots s_{N-1}$), then the Berlekamp-Massey algorithm tells us that the local linear complexity of $s'_0s'_1\ldots s'_{2L(N)-1}$ is L(N), since the local linear complexity cannot change during the k^{th} iteration of the loop in the algorithm unless $2.L(k-1) \leqslant k-1$. Thus, by (i) above there is a unique L(N)-stage LFSR on which $s'_0s'_1\ldots s'_{2L(N)-1}$ can be generated.

$s_0s_1\cdots s_{N-1}$ can be extended in $2^{2L(N)-N}$ such ways, and in each case we obtain an L(N)-stage LFSR which can be used to generate $s'_0s'_1\cdots s'_{2L(N)-1}$, and hence can also be used to generate $s_0s_1\cdots s_{N-1}$. But if an L(N)-stage LFSR can be used to generate $s_0s_1\cdots s_{N-1}$ then it can also be used to generate a sequence $s'_0s'_1\cdots s'_{2L(N)-1}$ with $s'_0s'_1\cdots s'_{N-1} = s_0s_1\cdots s_{N-1}$, and thus there are exactly $2^{2L(N)-N}$ L(N)-stage LFSRs on which $s_0s_1\cdots s_{N-1}$ can be generated.

Now consider the use of the Berlekamp-Massey algorithm to find the local linear complexity of one of the above-mentioned 2L(N)-bit sequences. For such a sequence, L(N) = L(N+1) = ... = L(2L(N)), and on the iterations of the loop with k = N+1,N+2,...,2L(N) one of the first two conditions holds (i.e. either d := 0, or d := 1 and 2.L(k-1) > k-1).
Thus, at each of these iterations the connection polynomial of the L(k)-stage LFSR given by the algorithm either remains unchanged or is modified by adding $x^aB(x)$ to it, where $B(x) = C_{m-1}(x)$, a = k-m, and m is the greatest integer such that L(m-1) < L(N).
Therefore, any L(N)-stage LFSR which generates a sequence $s'_0s'_1\cdots s'_{2L(N)-1}$ with $s'_0s_1\cdots s'_{N-1} = s_0s_1\cdots s_{N-1}$ must have connection polynomial of the form

$$C_N(x) + (\delta_0 x^{N+1-m} + \ldots + \delta_{2L(N)-N-1}x^{2L(N)-m})C_{m-1}(x)$$
$$= C_N(x) + (\delta_0 + \ldots + \delta_{2L(N)-N-1}x^{2L(N)-N-1})x^{N-m+1}C_{m-1}(x)$$

for some values δ_i (δ_i = 0 or 1, i = 0,1,...,2L(N)-N-1).

But there are $2^{L(N)-N}$ such polynomials, given by

$$\{C_N(x) + q(x).x^{N-m+1}C_{m-1}(x) : \deg q(x) < 2L(N)-N\},$$

and so each of these is the connection polynomial of an L(N)-stage LFSR that can be used to generate $s_0s_1\ldots s_{N-1}$ (since we have already shown that there are exactly $2^{L(N)-N}$ L(N)-stage LFSRs on which $s_0s_1\ldots s_{N-1}$ can be generated).

Hence, since N-m+1 and $C_{m-1}(x)$ are the values of a and B(x) produced when the Berlekamp-Massey algorithm is applied to $s_0s_1\ldots s_{N-1}$, we have proved part (ii) of the theorem.

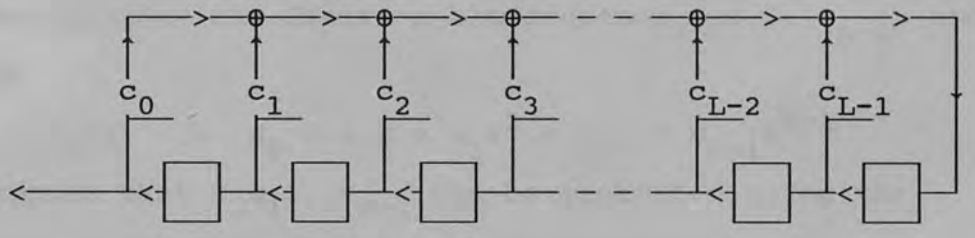
□

2.6. EXTENDED FORM OF THE ALGORITHM

The Berlekamp-Massey algorithm as stated in Section 2.3 computes the local linear complexity L(n) of an n-bit sequence $s_0 s_1 \ldots s_{n-1}$ and the connection polynomial of an L(n)-stage LFSR on which $s_0 s_1 \ldots s_{n-1}$ can be generated. In this section we will extend the Berlekamp-Massey algorithm so that it also computes a polynomial $P_n(x)$ (of degree less than L(n)), where $P_n(x)$ is the image of the initial state of the LFSR (when used to generate $s_0 s_1 \ldots s_{n-1}$) under a particular one-to-one mapping (given by (2.6.3) below) from the possible states of the register onto the set of polynomials over GF(2) of degree less than L(n). The polynomial $P_n(x)$ will be used extensively in Chapter 4.

Firstly, recall from Section 2.1 that the L-stage LFSR shown below has connection polynomial C(x) given by

$$C(x) = 1 + c_{L-1}x + c_{L-2}x^2 + \ldots + c_1 x^{L-1} + c_0 x^L$$



Let $s_0 s_1 s_2 \ldots$ be an infinite binary sequence that can be generated using the above L-stage register, and define S(x) to be the generating function of $s_0 s_1 s_2 \ldots$, i.e.

$$S(x) := s_0 + s_1 x + s_2 x^2 + \ldots$$

Then

$$C(x)S(x) = p_0 + p_1x + p_2x^2 + \ldots$$

where

$$p_j := \begin{cases} s_j + \sum_{i=L-j}^{L-1} c_i s_{j-L+i} & (j = 0,1,\ldots,L-1) \\ s_j + \sum_{i=0}^{L-1} c_i s_{j-L+i} & (j = L,L+1,\ldots) \end{cases}$$

But since $s_0s_1s_2\ldots$ can be generated using the L-stage LFSR with connection polynomial $C(x)$,

$$s_j = \sum_{i=0}^{L-1} c_i s_{j-L+i} \quad \text{for } j \geqslant L$$

and thus

$$C(x)S(x) = p_0 + p_1x + p_2x^2 + \ldots + p_{L-1}x^{L-1}$$

where $p_j = s_j + \sum_{i=L-j}^{L-1} c_i s_{j-L+i}$ $(j = 0,1,\ldots,L-1)$ (2.6.1)

Note that the polynomial $p_0 + p_1x + \ldots + p_{L-1}x^{L-1}$ above is the polynomial $\phi(x)$ of Theorem 1.4.2.

Now consider the finite n-bit sequence $s_0s_1\ldots s_{n-1}$, and let

$$S_n(x) := s_0 + s_1x + s_2x^2 + \ldots + s_{n-1}x^{n-1}$$

Suppose that $s_0s_1\ldots s_{n-1}$ can be generated using the L(n)-stage LFSR with connection polynomial

$$C_n(x) = 1 + c'_{L(n)-1}x + c'_{L(n)-2}x^2 + \ldots + c'_0x^{L(n)}$$

Then

$$C_n(x)S_n(x) = p_0 + p_1x + p_2x^2 + \ldots + p_{L(n)+n-1}x^{L(n)+n-1}$$

where

$$p_j := \begin{cases} s_j + \sum_{i=L(n)-j}^{L(n)-1} c_i' s_{j-L(n)+i} & (j = 0,1,\ldots,L(n)-1) \\ s_j + \sum_{i=0}^{L(n)-1} c_i' s_{j-L(n)+i} & (j = L(n),\ldots,n-1) \\ \sum_{i=0}^{n+L(n)-1-j} c_i' s_{j-L(n)+i} & (j = n,\ldots,n+L(n)-1) \end{cases} \qquad (2.6.2)$$

But since $s_0s_1\ldots s_{n-1}$ can be generated using the $L(n)$-stage LFSR with connection polynomial $C_n(x)$,

$$s_j = \sum_{i=0}^{L(n)-1} c_i' s_{j-L(n)+i} \quad \text{for} \quad L(n) \leqslant j \leqslant n-1$$

and thus

$$C_n(x)S_n(x) = p_0 + p_1x + \ldots + p_{L(n)-1}x^{L(n)-1} \pmod{x^n}$$

$$\text{where} \quad p_j = s_j + \sum_{i=L(n)-j}^{L(n)-1} c_i' s_{j-L(n)+i} \quad (j = 0,1,\ldots,L(n)-1) \qquad (2.6.3)$$

(c.f. (2.6.1))

Let

$$P_n(x) := p_0 + p_1x + p_2x^2 + \ldots + p_{L(n)-1}x^{L(n)-1}$$

Then

$$C_n(x)S_n(x) \equiv P_n(x) \pmod{x^n}$$

$$\Rightarrow \quad C_n(x)S_n(x) = P_n(x) + x^nG_n(x) \qquad (2.6.4)$$

for some polynomial $G_n(x)$

$$\Rightarrow \quad S_n(x) = (P_n(x) + x^n G_n(x)) \,/\, C_n(x) \qquad (2.6.5)$$

for some polynomial $G_n(x)$

(c.f. Theorem 1.4.2).

For a given L(n)-stage LFSR, (2.6.3) defines a one-to-one mapping from the possible initial states of the register onto the set of polynomials over GF(2) of degree less than L(n). Under this mapping, the initial state $s_0 s_1 \ldots s_{L(n)-1}$ is mapped onto the polynomial $P_n(x)$.

Earlier in this chapter we described the Berlekamp-Massey algorithm and showed that it can be used to compute the local linear complexity L(n) of an n-bit sequence $s_0 s_1 \ldots s_{n-1}$ and the connection polynomial of an L(n)-stage LFSR which can be used to generate $s_0 s_1 \ldots s_{n-1}$. We now show how the Berlekamp-Massey algorithm can be extended so that it also computes the corresponding polynomial $P_n(x)$. We first state the extended form of the algorithm :-

The Berlekamp-Massey Algorithm (Extended form)

$L(0) := 0$

$C_0(x) := 1$ $\qquad P_0(x) := 0$

$B(x) := 1$ $\qquad Q(x) := x^{-1}$

$a := 1$

FOR k := 1 TO n DO

$$d := s_{k-1} + \sum_{i=0}^{L(k-1)-1} c_i^{(k-1)} s_{k-L(k-1)+i-1}$$

IF d = 0 THEN

$L(k) := L(k-1)$

$C_k(x) := C_{k-1}(x)$ $\qquad P_k(x) := P_{k-1}(x)$

$a := a + 1$

END IF

IF d = 1 AND 2.L(k-1) > k-1 THEN

$L(k) := L(k-1)$

$C_k(x) := C_{k-1}(x) + x^a B(x)$ $\qquad P_k(x) := P_{k-1}(x) + x^a Q(x)$

$a := a + 1$

END IF

IF d = 1 AND 2.L(k-1) $\leqslant$ k-1 THEN

$L(k) := k - L(k-1)$

$C_k(x) := C_{k-1}(x) + x^a B(x)$ $\qquad P_k(x) := P_{k-1}(x) + x^a Q(x)$

$B(x) := C_{k-1}(x)$ $\qquad Q(x) := P_{k-1}(x)$

$a := 1$

END IF

END DO

We now justify the use of this algorithm to calculate the polynomial $P_n(x)$. We will use an inductive proof to show that, at each iteration of the loop in the algorithm, the polynomial $P_k(x)$ generated has the property that $P_k(x) = C_k(x)S_k(x) \pmod{x^k}$:-

Suppose that the first increase in local linear complexity occurs on the $a(1)^{th}$ iteration of the loop in the algorithm (i.e. $L(1) = L(2) = \ldots = L(a(1)-1) = 0$, $L(a(1)) = a(1)$, $s_0 s_1 \ldots s_{a(1)-1} = 00\ldots01$). Then the Berlekamp-Massey algorithm gives the following sequence of polynomials :-

k	$C_k(x)$	$P_k(x)$	$L(k)$
0	1	0	0
1	1	0	0
2	1	0	0
⋮	⋮	⋮	⋮
$a(1)-1$	1	0	0
$a(1)$	$1 + x^{a(1)}$	$x^{a(1)-1}$	$a(1)$

Thus, for $k < a(1)$, the algorithm gives

$C_k(x) = 1, \quad P_k(x) = 0, \quad S_k(x) = 0,$

so $C_k(x)S_k(x) \pmod{x^k} = 0 = P_k(x)$

and thus the algorithm produces the correct value of $P_k(x)$ in this case.

For k = a(1) the algorithm gives

$$C_k(x) = 1 + x^{a(1)}, \quad P_k(x) = x^{a(1)-1}, \quad S_k(x) = x^{a(1)-1}$$

so $C_k(x)S_k(x) \pmod{x^k} = (1 + x^{a(1)}).x^{a(1)-1} \pmod{x^{a(1)}}$

$$= x^{a(1)-1}$$

$$= P_k(x)$$

and thus the algorithm produces the correct value of $P_k(x)$ in this case too.

Now assume the algorithm produces the correct value of $P_k(x)$ (i.e. assume there exists a polynomial $G_k(x)$ such that $C_k(x)S_k(x) = P_k(x) + x^k G_k(x)$, where $\deg P_k(x) < L(k)$) for $k = 1,2,\ldots,N$, where $N \geq a(1)$. Consider the $N+1^{th}$ iteration of the loop in the algorithm (i.e. the iteration with k = N+1).

If d := 0 in this iteration of the loop then the algorithm gives

$$C_{N+1}(x) := C_N(x), \quad P_{N+1}(x) := P_N(x)$$

so $C_{N+1}(x)S_{N+1}(x) \pmod{x^{N+1}}$

$$= C_N(x)(S_N(x) + s_N x^N) \pmod{x^{N+1}}$$

$$= P_N(x) + x^N G_N(x) + s_N x^N C_N(x) \pmod{x^{N+1}}$$

for some $G_N(x) = g_0 + g_1 x + g_2 x^2 + \ldots$,

where $\deg P_N(x) < L(N)$

Thus $C_{N+1}(x)S_{N+1}(x) \pmod{x^{N+1}}$

$$= P_N(x) + (g_0 + s_N).x^N$$

(since $C_N(x) = 1 + c_{L(N)-1}x + \ldots + c_0 x^{L(N)}$

for some coefficients $c_0, c_1, \ldots, c_{L(N)-1}$)

But since $C_N(x)S_N(x) = P_N(x) + x^N G_N(x)$,

where $\deg P_N(x) < L(N) \leqslant N$,

$$g_0 = \text{coefficient of } x^N \text{ in } C_N(x)S_N(x) = \sum_{i=0}^{L(N)-1} c_i s_{N-L(N)+i}$$

(by (2.6.2))

And since the L(N)-stage LFSR with connection polynomial $C_N(x)$ can be used to generate $s_0 s_1 \ldots s_N$,

$$s_N = \sum_{i=0}^{L(N)-1} c_i s_{N-L(N)+i}$$

Thus $g_0 = s_N$,

and so $C_{N+1}(x)S_{N+1}(x) \pmod{x^{N+1}} = P_N(x) = P_{N+1}(x)$

(i.e. the algorithm gives the correct value of $P_{N+1}(x)$ in the case d := 0).

If d := 1 then the algorithm gives

$C_{N+1}(x) := C_N(x) + x^a B(x)$, $P_{N+1}(x) := P_N(x) + x^a Q(x)$

If the last increase in local linear complexity occurred on the m^{th} iteration of the loop then, at the end of the N^{th} iteration,

$B(x) = C_{m-1}(x)$, $Q(x) = P_{m-1}(x)$, and $a = N-m+1$,

so $C_{N+1}(x)S_{N+1}(x) \pmod{x^{N+1}}$

$= (C_N(x) + x^{N-m+1}C_{m-1}(x))(S_N(x) + s_N x^N) \pmod{x^{N+1}}$

$= C_N(x)S_N(x) + x^{N-m+1}C_{m-1}(x)S_N(x) + s_N x^N C_N(x) \pmod{x^{N+1}}$

(since $N-m+1 \geqslant 1$)

$$= P_N(x) + x^N G_N(x) + x^{N-m+1} C_{m-1}(x)(s_0 + \ldots + s_{m-1}x^{m-1}) + s_N x^N C_N(x) \pmod{x^{N+1}}$$

for some $G_N(x) = g_0 + g_1 x + g_2 x^2 + \ldots$,

where $\deg P_N(x) < L(N)$

Thus $C_{N+1}(x)S_{N+1}(x) \pmod{x^{N+1}}$

$$= P_N(x) + (g_0 + s_N).x^N + x^{N-m+1} C_{m-1}(x)(S_{m-1}(x) + s_{m-1}x^{m-1}) \pmod{x^{N+1}}$$

(since $C_N(x) = 1 + c_{L(N)-1}x + \ldots + c_0 x^{L(N)}$

for some coefficients $c_0, c_1, \ldots c_{L(N)-1}$)

$$= P_N(x) + (g_0 + s_N).x^N + x^{N-m+1}(P_{m-1}(x) + x^{m-1}G_{m-1}(x)) + s_{m-1}x^N C_{m-1}(x) \pmod{x^{N+1}}$$

for some $G_{m-1}(x) = g'_0 + g'_1 x + g'_2 x^2 + \ldots$,

where $\deg P_{m-1}(x) < L(m-1)$

$$\Rightarrow C_{N+1}(x)S_{N+1}(x) \pmod{x^{N+1}} = P_N(x) + (g_0 + s_N + g'_0 + s_{m-1}).x^N + x^{N-m+1}P_{m-1}(x) \qquad (2.6.6)$$

(since $C_{m-1}(x) = 1 + c'_{L(m-1)-1}x + \ldots + c'_0 x^{L(m-1)}$)

for some coefficients $c'_0, c'_1, \ldots, c'_{L(m-1)-1}$)

But since $C_N(x)S_N(x) = P_N(x) + x^N G_N(x)$,

where $\deg P_N(x) < L(N) \leqslant N$,

g_0 = coefficient of x^N in $C_N(x)S_N(x) = \sum_{i=0}^{L(N)-1} c_i s_{N-L(N)+i}$

(by (2.6.2))

And since d = 1, the L(N)-stage LFSR with connection polynomial $C_N(x)$ cannot be used to generate $s_0 s_1 \ldots s_N$,

and so $s_N = \sum_{i=0}^{L(N)-1} (c_i s_{N-L(N)+i}) + 1$

Thus $g_0 + s_N = 1.$ (2.6.7)

Similarly, since $C_{m-1}(x)S_{m-1}(x) = P_{m-1}(x) + x^{m-1}G_{m-1}(x)$, where deg $P_{m-1}(x) < L(m-1) \leqslant m-1$,

g_0' = coefficient of x^{m-1} in $C_{m-1}(x)S_{m-1}(x)$

$$= \sum_{i=0}^{L(m-1)-1} c_i s_{m-1-L(m-1)+i} \qquad \text{(by (2.6.2))}$$

And since L(m) > L(m-1), the L(m-1)-stage LFSR with connection polynomial $C_{m-1}(x)$ cannot be used to generate $s_0 s_1 \ldots s_{m-1}$,

and so $s_{m-1} = \sum_{i=0}^{L(m-1)-1} (c_i s_{m-1-L(m-1)+i}) + 1$

Thus, $g_0' + s_{m-1} = 1.$ (2.6.8)

Combining (2.6.6), (2.6.7) and (2.6.8) we have

$$C_{N+1}(x)S_{N+1}(x) \pmod{x^{N+1}} = P_N(x) + x^{N-m+1}P_{m-1}(x)$$
$$= P_{N+1}(x)$$

and so the algorithm gives the correct value of $P_{N+1}(x)$ in the case d := 1.

Thus we have shown that, for any integer $n \geqslant 1$, the extended form of the Berlekamp-Massey algorithm produces a polynomial $P_n(x)$ of degree less than $L(n)$ such that

$$P_n(x) = C_n(x)S_n(x) \pmod{x^n}$$

□

We now prove a relationship between the local linear complexity $L(n)$ of the n-bit sequence $s_0s_1\ldots s_{n-1}$ and the polynomials $C_n(x)$ and $P_n(x)$ produced when the Berlekamp-Massey algorithm is applied to this sequence :-

Theorem 2.6.1.

Let $L(n)$ be the local linear complexity of the n-bit sequence $s_0s_1\ldots s_{n-1}$, and let $C_n(x)$ and $P_n(x)$ be the polynomials produced when the Berlekamp-Massey algorithm (in its extended form) is applied to $s_0s_1\ldots s_{n-1}$.
Then

$$L(n) = \max (\deg C_n(x), 1 + \deg P_n(x))$$

Proof

Let $C_n(x) = 1 + c_{L(n)-1}x + \ldots + c_1x^{L(n)-1} + c_0x^{L(n)}$

and $P_n(x) = p_0 + p_1x + \ldots + p_{L(n)-1}x^{L(n)-1}$

(recall that $\deg P_n(x) \leqslant L(n)-1$)

If $c_0 = 1$ then deg $C_n(x) = L(n)$

=> max (deg $C_n(x)$, 1 + deg $P_n(x)$) = deg $C_n(x)$ = $L(n)$

and so the theorem holds in this case.

If $p_{L(n)-1} = 1$ then deg $P_n(x) = L(n)-1$

=> max (deg $C_n(x)$, 1 + deg $P_n(x)$) = 1 + deg $P_n(x)$ = $L(n)$

and so the theorem holds in this case too.

We will show that if $c_0 = 0$ then $p_{L(n)-1} \neq 0$, and so one of the two cases above must hold.

Since the L(n)-stage LFSR with connection polynomial $C_n(x)$ can be used to generate $s_0 s_1 \ldots s_{n-1}$,

$$s_j = \sum_{i=0}^{L(n)-1} c_i s_{j-L(n)+i} \quad \text{for } j = L(n),\ldots,n-1.$$

Suppose that $c_0 = 0$.

Then

$$s_j = \sum_{i=1}^{L(n)-1} c_i s_{j-L(n)+i} \quad \text{for } j = L(n),\ldots,n-1.$$

And from equation (2.6.3),

$$p_{L(n)-1} = s_{L(n)-1} + \sum_{i=1}^{L(n)-1} c_i s_{i-1}$$

$$\Rightarrow \quad s_{L(n)-1} = \sum_{i=1}^{L(n)-1} c_i s_{i-1} \quad \text{if } p_{L(n)-1} = 0$$

Thus, if $p_{L(n)-1} = 0$ then

$$s_j = \sum_{i=1}^{L(n)-1} c_i s_{j-L(n)+i} \quad \text{for } j = L(n)-1,\ldots,n-1.$$

=> $s_0 s_1 \ldots s_{n-1}$ can be generated using the $(L(n)-1)$-stage LFSR with connection polynomial $C_n(x)$

=> $s_0 s_1 \ldots s_{n-1}$ has local linear complexity $\leq L(n)-1$.

But, by definition, $s_0 s_1 \ldots s_{n-1}$ has local linear complexity $L(n)$, giving us a contradiction.

Thus, if $c_0 = 0$ then $p_{L(n)-1} \neq 0$.

□

CHAPTER 3

LINEAR COMPLEXITY PROFILES

3.1. INTRODUCTION

The Berlekamp-Massey algorithm described in Chapter 2 is usually thought of as an algorithm for computing the local linear complexity L(n) of an n-bit binary sequence $s_0s_1\cdots s_{n-1}$. However, it can be seen from Chapter 2 that this algorithm also computes, in the course of computing L(n), the local linear complexities L(1), L(2), ..., L(n-1) of the subsequences s_0, s_0s_1, ..., $s_0s_1\cdots s_{n-2}$ of $s_0s_1\cdots s_{n-1}$. Thus, the Berlekamp-Massey algorithm can be used to generate an n-vector (L(1),L(2),...,L(n)), which we call the linear complexity profile of $s_0s_1\cdots s_{n-1}$.

Alternatively, we can consider the linear complexity profile as a graph. For a given n-bit sequence $s_0s_1\cdots s_{n-1}$, consider the function L from the set of integers {1,2,...,n} into itself, where L(k) is defined to be the local linear complexity of the first k bits $s_0s_1\cdots s_{k-1}$ of $s_0s_1\cdots s_{n-1}$ (k = 1,2,...,n). The linear complexity profile of $s_0s_1\cdots s_{n-1}$ can be thought of as the graph of the function L. Strictly speaking, when this graph is plotted a point should be placed at (k,L(k)) for k = 1,2,...,n and no other points should be plotted, since the function L is defined only for the integers 1,2,...,n. However, in practice we will plot the linear complexity profile as if it was a step function defined for all real values in the interval [0,n], by taking the value y of the function at the

point x to be the local linear complexity L(k) of the subsequence $s_0s_1\ldots s_{k-1}$, where $k = \lfloor x \rfloor$, the greatest integer not exceeding x.

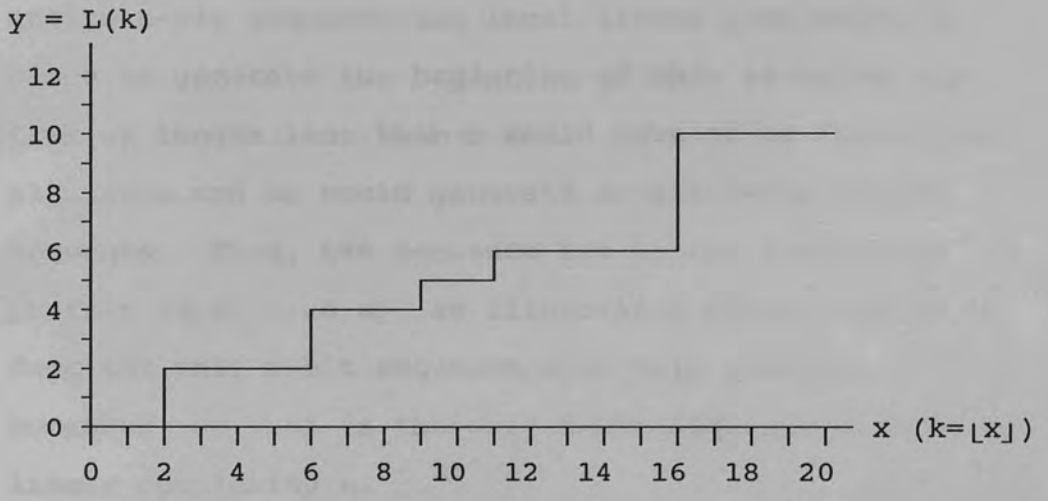
We will use the term "linear complexity profile" to denote both the vector (L(1),L(2),...,L(n)) and the graph of the corresponding step function. Note, however, that in some texts, including [15], the graph of the step function is referred to as a "staircase profile".

The above definition applies to finite binary sequences. In a similar way, we can define the linear complexity profile of an infinite binary sequence $s_0s_1s_2\ldots$ to be either the vector (L(1),L(2),L(3),...) or the graph of the corresponding step function, where L(k) is the local linear complexity of the k-bit subsequence $s_0s_1\ldots s_{k-1}$.

3.2. EXAMPLES OF LINEAR COMPLEXITY PROFILES

Example 3.2.1.

As a first example of a linear complexity profile, consider the 20-bit sequence 01111011010011010000. This sequence has linear complexity profile (0,2,2,2,2,4,4,4, 5,5,6,6,6,6,6,10,10,10,10,10), as illustrated below. Note that the local linear complexities of the first 11 subsequences were computed in Section 2.3 using the Berlekamp-Massey algorithm.

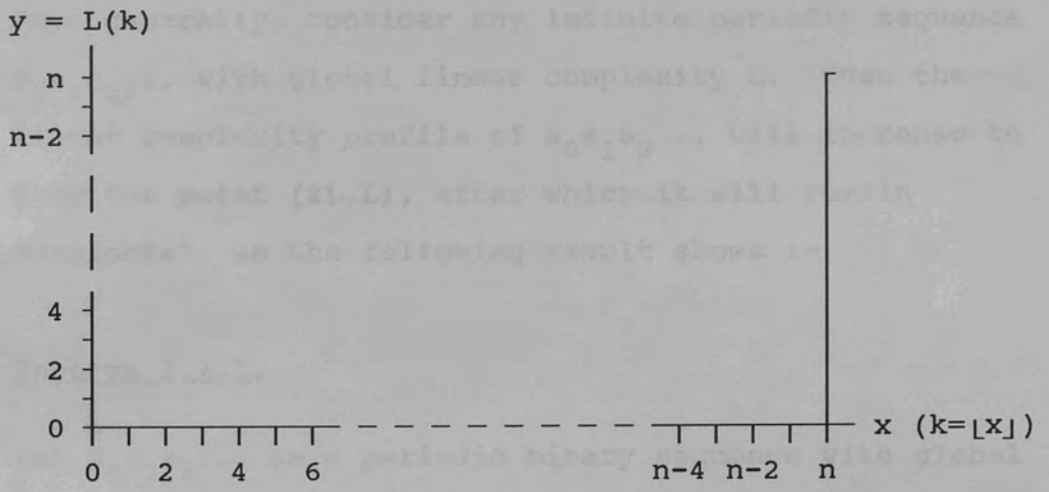


We say that the linear complexity profile of a sequence "jumps" with the k^{th} bit in the sequence if $L(k) > L(k-1)$, and we define the height of the jump to be the difference $L(k) - L(k-1)$. For instance, the

linear complexity profile in the above example jumps with the 2^{nd}, 6^{th}, 9^{th}, 11^{th} and 16^{th} bits in the sequence, and these jumps have heights 2, 2, 1, 1 and 4 respectively. We describe a horizontal section of the profile as a step. Thus, there is a step of length 4 between the first two jumps in the example above.

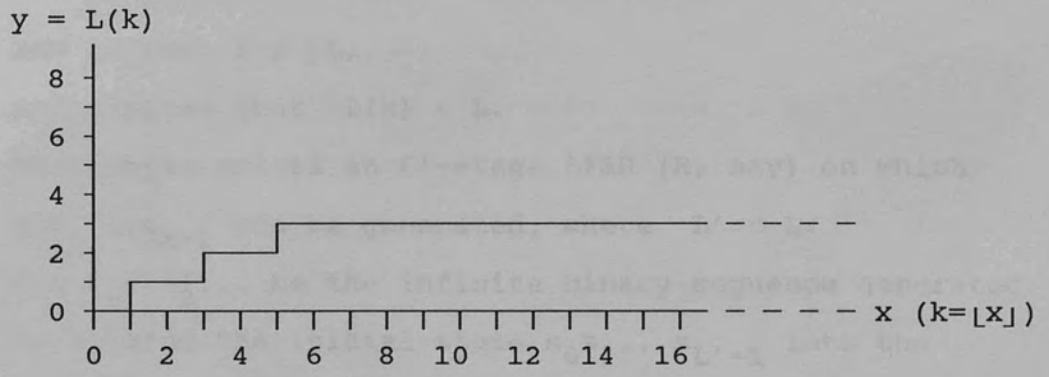
Example 3.2.2.

For any integer n > 0, consider the n-bit sequence 00...01. For k = 1,2,...,n-1, the k-bit sequence 00...0 has local linear complexity L(k) = 0; and the entire n-bit sequence has local linear complexity n, since to generate the beginning of this sequence any LFSR of length less than n would have to be loaded with all zeros and so would generate an all zeros output sequence. Thus, the sequence has linear complexity profile (0,0,...,0,n), as illustrated below, and is in fact the only n-bit sequence with this profile. Moreover, 00...01 is the only n-bit sequence with local linear complexity n.



Example 3.2.3.

Consider the infinite periodic sequence 10111001011100 1011..., which has period 7 and global linear complexity 3. The linear complexity profile of this sequence is (1,1,2,2,3,3,3,3,3,3,...), as illustrated below.



More generally, consider any infinite periodic sequence $s_0s_1s_2\ldots$ with global linear complexity L. Then the linear complexity profile of $s_0s_1s_2\ldots$ will increase to L by the point (2L,L), after which it will remain horizontal, as the following result shows :-

Theorem 3.2.1.

Let $s_0s_1s_2\ldots$ be a periodic binary sequence with global linear complexity L, and let L(k) be the local linear complexity of the k-bit subsequence $s_0s_1\ldots s_{k-1}$ of $s_0s_1s_2\ldots$.
Then $L(k) = L$ for $k \geqslant 2L$.

Proof

Since $s_0s_1s_2\ldots$ has global linear complexity L, $s_0s_1\ldots s_{k-1}$ can be generated using an L-stage LFSR for any integer $k > 0$, and so $s_0s_1\ldots s_{k-1}$ has local linear complexity $L(k) \leqslant L$.

Now assume $k \geqslant 2L$,
and suppose that $L(k) < L$.
Then there exists an L'-stage LFSR (R, say) on which $s_0s_1\ldots s_{k-1}$ can be generated, where $L' < L$.
Let $s'_0s'_1s'_2\ldots$ be the infinite binary sequence generated by loading the initial state $s_0s_1\ldots s_{L'-1}$ into the register R. Then $s_i = s'_i$ for $i = 0,1,\ldots,k-1$, and $s_j \neq s'_j$ for some $j \geqslant k$.

Now consider the infinite sequence $s''_0 s''_1 s''_2 \ldots$ formed by modulo 2 adding $s_0 s_1 s_2 \ldots$ and $s'_0 s'_1 s'_2 \ldots$. From the above discussion we know that $s''_i = 0$ for $i = 0,1,\ldots,k-1$, and $s''_j = 1$ for some $j \geqslant k$. Thus, $s''_0 s''_1 s''_2 \ldots$ cannot be generated using a k-stage LFSR (c.f. Example 3.2.2), and hence $s''_0 s''_1 s''_2 \ldots$ has global linear complexity

$$L'' > k. \qquad (3.2.1)$$

Recall from Section 1.3 that $\Omega(f)$ is the set of all infinite binary sequences that can be generated using the LFSR with characteristic polynomial f(x). From the results in [20] it can easily be seen that any infinite binary sequence which is the modulo 2 sum of a sequence in $\Omega(f)$ and a sequence in $\Omega(g)$ is in $\Omega(fg)$. Hence, an infinite binary sequence which is formed by modulo 2 adding two other such sequences, one of which can be generated using the m-stage LFSR with characteristic polynomial f(x) and the other using the m'-stage LFSR with characteristic polynomial g(x), can be generated using the (m+m')-stage LFSR with characteristic polynomial f(x)g(x).

Thus, $s''_0 s''_1 s''_2 \ldots$ can be generated using an (L+L')-stage LFSR, and so has global linear complexity

$$L'' \leqslant L+L' < 2L \leqslant k. \qquad (3.2.2)$$

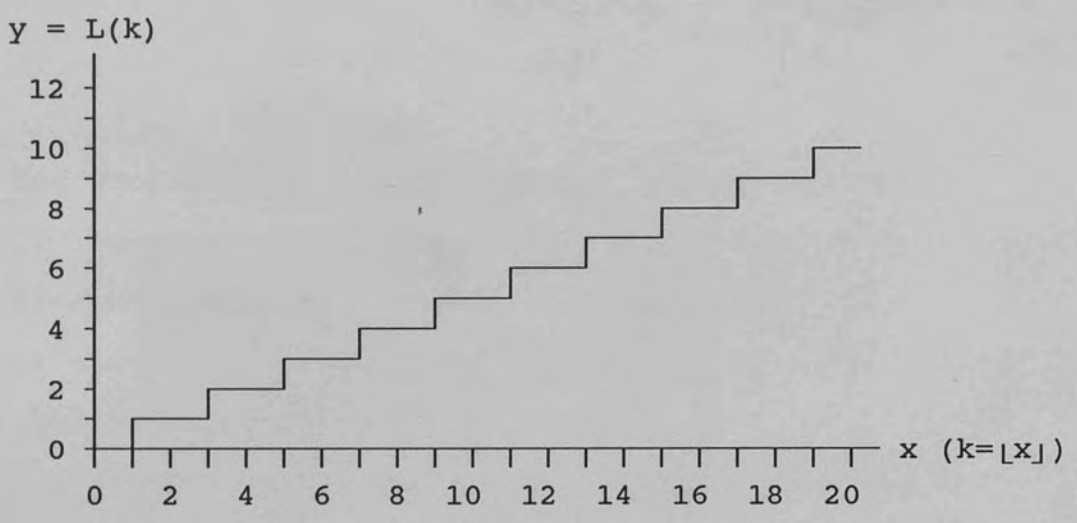
But (3.2.2) contradicts (3.2.1),

and thus $L(k) = L$ for $k \geqslant 2L$.

□

Example 3.2.4.

As a final example of a linear complexity profile, consider the 20-bit sequence 10111011001000101001. This sequence has the linear complexity profile (1,1,2,2,3,3,...,9,9,10,10), as illustrated below. Such a profile is commonly known as a "perfect" linear complexity profile.



More generally, an n-bit binary sequence is said to have a perfect linear complexity profile if

$$L(k) = \lceil k/2 \rceil \quad \text{for } k = 1,2,\ldots,n,$$

where $\lceil k/2 \rceil$ is the least integer greater than or equal to k/2.

(i.e. if it has linear complexity profile $(1,1,2,2,3,3,\ldots,\lceil n/2 \rceil)$.

Similarly, an infinite binary sequence has a perfect linear complexity profile if

$$L(k) = \lceil k/2 \rceil \quad \text{for } k = 1,2,3,....$$

We will return to the subject of perfect linear complexity profiles in Chapter 4.

3.3. PROPERTIES OF LINEAR COMPLEXITY PROFILES

In this section we look at some properties of linear complexity profiles. We begin by returning to the Berlekamp-Massey algorithm, as described in Chapter 2, and interpret some of the results from that chapter in terms of linear complexity profiles.

The first point to notice is that, by Lemma 2.2.1, the graph of the linear complexity profile of a binary sequence must be non-decreasing.

Now consider the remarks towards the end of Section 2.3. It was there stated that "if $L(k-1) > \frac{k-1}{2}$ then the local linear complexity cannot increase on the k^{th} iteration of the loop". Translated into the language of linear complexity profiles, this statement says that, if the linear complexity profile of a sequence is above the line $y = x/2$ at the point $(k-1,L(k-1))$, then the profile cannot jump with the k^{th} bit in the sequence (i.e. it cannot jump at $x = k$). Similarly, "if $L(k-1) \leqslant \frac{k-1}{2}$ then the local linear complexity will increase or remain unchanged according to whether $d = 1$ or 0" means that, if the linear complexity profile is on or below the line $y = x/2$ at the point $(k-1,L(k-1))$, then the profile will jump or not with the k^{th} bit in the sequence, according to the value of that bit.

Further, "if the local linear complexity does increase then its new value must be k-L(k-1)" implies that, if the linear complexity profile does jump at x = k, then it jumps from L(k-1) to k-L(k-1), and so the jump has height k - 2L(k-1). But $\frac{k}{2}$ - L(k-1) = (k-L(k-1)) - $\frac{k}{2}$, and hence the profile jumps the same distance above the line y = x/2 as it was below the line before it jumped.

To illustrate the above points, consider the linear complexity profile in Example 3.2.1. This profile is redrawn in Figure 3.3.1 below, this time with the line y = x/2 added.

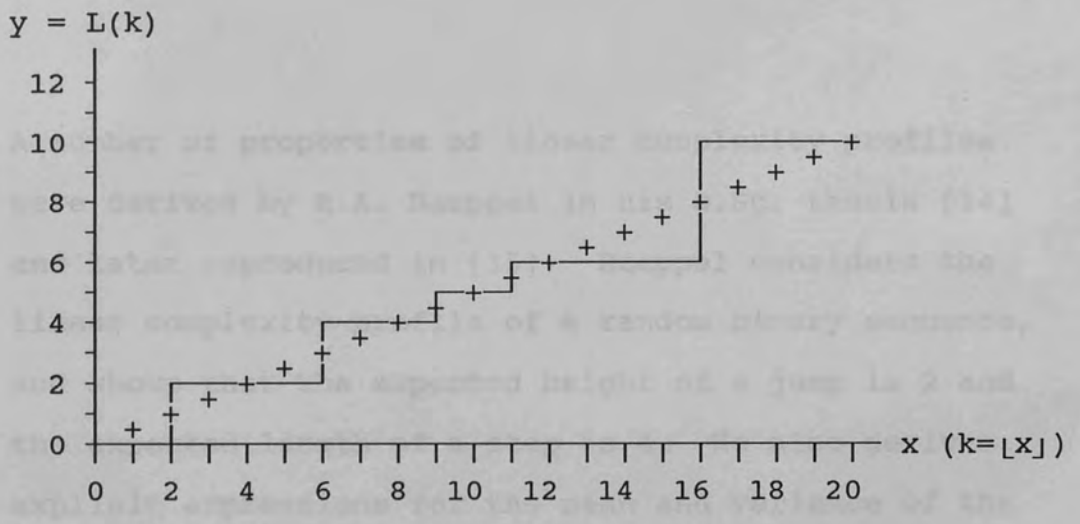


Figure 3.3.1. An example of a linear complexity profile

Immediately before the first jump in the above profile, the profile is a distance of 1 below the line $y = x/2$. Thus, given that the profile jumps, it jumps to a distance of 1 above the line $y = x/2$ (i.e. to the point (2,2)). The profile cannot jump with the 3^{rd} or 4^{th} bits in the sequence, as the profile is above the line $y = x/2$ at $y = 2$ and $y = 3$, and it does not jump at $y = 5$, even though $L(4) \leqslant 4/2$. The jump at $y = 6$ also has height 2, as the profile is again a distance of 1 below the line $y = x/2$ immediately before the jump. The next jump, however, which occurs with the 9^{th} bit in the sequence, only has height 1, as the profile is only a distance of $\frac{1}{2}$ below the line $y = x/2$ immediately before the jump.

A number of properties of linear complexity profiles were derived by R.A. Rueppel in his D.Sc. thesis [14] and later reproduced in [15]. Rueppel considers the linear complexity profile of a random binary sequence, and shows that the expected height of a jump is 2 and the expected length of a step is 4. He also derives explicit expressions for the mean and variance of the local linear complexity L(n) of the first n bits of a random sequence. These results are quoted in Theorems 3.3.1 and 3.3.2 below :-

Theorem 3.3.1.

Let E(n) be the expected local linear complexity of a random n-bit binary sequence. Then

$$E(n) = \begin{cases} \frac{n}{2} + \frac{2}{9} - 2^{-n}(\frac{n}{3} + \frac{2}{9}) & \text{(n even)} \\ \frac{n}{2} + \frac{5}{18} - 2^{-n}(\frac{n}{3} + \frac{2}{9}) & \text{(n odd)} \end{cases}$$

Theorem 3.3.2.

Let V(n) be the variance of the local linear complexity of a random n-bit binary sequence. Then

$$V(n) = \begin{cases} \frac{86}{81} - 2^{-n}(\frac{14n}{27} + \frac{82}{81}) - 2^{-2n}(\frac{n^2}{9} + \frac{4n}{27} + \frac{4}{81}) & \text{(n even)} \\ \frac{86}{81} - 2^{-n}(\frac{13n}{27} + \frac{80}{81}) - 2^{-2n}(\frac{n^2}{9} + \frac{4n}{27} + \frac{4}{81}) & \text{(n odd)} \end{cases}$$

3.4. SOME ENUMERATION RESULTS

In this section we will prove a number of enumeration results which are connected with linear complexity profiles. We wiil concern ourselves only with profiles which are "possible" in the sense that, for each possible linear complexity profile, there exists at least one binary sequence which has that profile. The set of possible linear complexity profiles is restricted by the fact that each one must satisfy the properties derived in Section 3.3 from the Berlekamp-Massey algorithm. More precisely, for the linear complexity profile $(L(1),L(2),\ldots,L(n))$ to be possible, the following two conditions must hold for each integer k such that $1 \leqslant k \leqslant n$:-

(i) if $L(k-1) > \frac{k-1}{2}$ then $L(k) = L(k-1)$

(ii) if $L(k-1) \leqslant \frac{k-1}{2}$ then

either $L(k) = L(k-1)$ or $L(k) = k-L(k-1)$.

We say that a linear complexity profile $(L(1),L(2),\ldots,L(n))$ is <u>valid</u> if it satisfies the above conditions. In Theorem 3.4.2 it will be shown that any valid linear complexity profile is possible; hence, the adjectives "possible" and "valid" are synonomous when applied to linear complexity profiles.

The first of the results in this section gives an expression for the number of n-bit sequences which have a given linear complexity profile. We begin by proving a lemma :-

Lemma 3.4.1.

Consider a given valid linear complexity profile $(L(1),L(2),\ldots,L(n))$.
There exist precisely 2^M distinct n-bit sequences which have this profile, where

$$M = |\{k : L(k) > k/2,\ 1 \leqslant k \leqslant n-1\}|.$$

(Note that M is the number of points in the profile which are above the line $y = x/2$, excluding the point $(n,L(n))$.)

Proof

We will consider what happens when we attempt to generate an n-bit sequence $s_0 s_1 \ldots s_{n-1}$ with the given linear complexity profile $(L(1),L(2),\ldots,L(n))$.
We begin by generating the first bit s_0 of the sequence.
If $s_0 = 0$ then $L(1) = 0$
and if $s_0 = 1$ then $L(1) = 1$.
Therefore $L(1)$ defines s_0 uniquely.

Now suppose that we have generated a k-bit subsequence $s_0 s_1 \ldots s_{k-1}$ with linear complexity profile $(L(1), L(2), \ldots, L(k))$ $(1 \leqslant k \leqslant n-1)$.
Consider the generation of s_k.
If $L(k) \leqslant k/2$ then, by Theorem 2.5.1, there exists a unique L(k)-stage LFSR (R, say) which can be used to generate $s_0 s_1 \ldots s_{k-1}$.
Let u_k be the $(k+1)^{th}$ bit generated by R when it is loaded with the initial state $s_0 s_1 \ldots s_{L(k)-1}$.

By the Berlekamp-Massey algorithm, if $L(k) \leqslant k/2$ then either (i) $L(k+1) = L(k)$
or (ii) $L(k+1) = k+1 - L(k)$.
(i) $L(k+1) = L(k)$
$\Rightarrow s_0 s_1 \ldots s_k$ can be generated using R
$\Rightarrow s_k = u_k$.
(ii) $L(k+1) = k+1 - L(k)$
$\Rightarrow s_0 s_1 \ldots s_k$ cannot be generated using R
$\Rightarrow s_k = u_k + 1$
Therefore, if $L(k) \leqslant k/2$ then $L(k+1)$ defines s_k uniquely, given $s_0 s_1 \ldots s_{k-1}$.

If $L(k) > k/2$ then, by the Berlekamp-Massey algorithm, $L(k+1) = L(k)$, independent of s_k. Thus, in this situation we have two choices for s_k.

By combining the cases $L(k) \leqslant k/2$ and $L(k) > k/2$ above, it can be seen that the algorithm given below can be used to generate an n-bit sequence $s_0 s_1 \ldots s_{n-1}$ with

linear complexity profile (L(1),L(2),...,L(n)). By repeatedly using the algorithm we can construct the set of all binary sequences with this profile. The algorithm is as follows :-

```
s_0 := L(1)
FOR  k = 1 TO n-1  DO
   IF  L(k) ⩽ k/2
   THEN
      IF  L(k+1) = L(k)
      THEN  s_k := u_k
      IF  L(k+1) = k+1 - L(k)
      THEN  s_k := u_k + 1
   END IF
   IF  L(k) > k/2
   THEN  Choose s_k to be either 0 or 1
END DO
```

The number of n-bit sequences with the given valid linear complexity profile (L(1),L(2)...,L(n)) is exactly the number of different sequences that can be generated by the above algorithm.
But in the algorithm, s_k is fixed when $L(k) \leqslant k/2$, and when $L(k) > k/2$ there are 2 choices for s_k.

Hence, the number of n-bit sequences with linear complexity profile $(L(1),L(2),,,,,L(n))$ is 2^M, where M is the cardinality of the set

$\{k : L(k) > k/2,\ 1 \leqslant k \leqslant n-1\}$.

□

We next use Lemma 3.4.1 to obtain our explicit expression for the number of n-bit sequences with linear complexity profile $(L(1),L(2),...,L(n))$:-

Theorem 3.4.2.

Consider a given valid linear complexity profile $(L(1),L(2),...,L(n))$.
There exist precisely 2^M distinct n-bit sequences which have this profile, where

$$M = \min(L(n),n-L(n)).$$

Proof

It is sufficient to prove that

$$|\{k : L(k) > k/2,\ 1 \leqslant k \leqslant n-1\}| = \min(L(n),n-L(n)) \quad \text{for } n \geqslant 1,$$

as the proof then follows by Lemma 3.4.1.

In other words, we will prove that the number of points in the profile which are above the line $y = x/2$, excluding the last point, is $\min(L(n), n-L(n))$.

The proof is by induction.
The result is true for $n = 1$ trivially, since $L(1) = 0$ or 1.

Suppose the result is true for $n = N$
(i.e. $|\{k : L(k) > k/2,\ 1 \leqslant k \leqslant N-1\}| = \min(L(N), N-L(N))$).
We will show that the result also holds for $n = N+1$
(i.e. $|\{k : L(k) > k/2,\ 1 \leqslant k \leqslant N\}| = \min(L(N+1), N+1-L(N+1))$).

We consider 3 cases, which cover all possibilities :-

(i) $L(N) > N/2$ (so $L(N+1) = L(N)$) :

$$
\begin{aligned}
&|\{k : L(k) > k/2,\ 1 \leqslant k \leqslant N\}| \\
&\quad = |\{k : L(k) > k/2,\ 1 \leqslant k \leqslant N-1\}| + 1 \\
&\quad = \min(L(N), N-L(N)) + 1 \\
&\quad = \min(L(N), N-L(N)+1) \qquad \text{(since } N-L(N) < L(N)\text{)} \\
&\quad = \min(L(N+1), N+1-L(N+1)).
\end{aligned}
$$

(ii) $L(N) \leq N/2$, $L(N+1) = L(N)$:

$$|\{k : L(k) > k/2,\ 1 \leq k \leq N\}|$$

$$= |\{k : L(k) > k/2,\ 1 \leq k \leq N-1\}|$$

$$= \min(L(N), N-L(N))$$

$$= \min(L(N), N-L(N)+1) \qquad \text{(since } L(N) \leq N-L(N)\text{)}$$

$$= \min(L(N+1), N+1-L(N+1)).$$

(iii) $L(N) \leq N/2$, $L(N+1) = N+1-L(N)$:

$$|\{k : L(k) > k/2,\ 1 \leq k \leq N\}|$$

$$= |\{k : L(k) > k/2,\ 1 \leq k \leq N-1\}|$$

$$= \min(L(N), N-L(N))$$

$$= \min(L(N), N-L(N)+1) \qquad \text{(since } L(N) \leq N-L(N)\text{)}$$

$$= \min(N+1-L(N+1), L(N+1)).$$

Combining the 3 cases above we see that

$$|\{k : L(k) > k/2,\ 1 \leq k \leq N\}| = \min(L(N+1), N+1-L(N+1))$$

and hence, by induction,

$$|\{k : L(k) > k/2,\ 1 \leq k \leq n-1\}| = \min(L(n), n-L(n))$$

for all $n \geq 1$.

□

Since the value M in the statement of the above theorem is non-negative, for each valid linear complexity profile there exists at least one binary sequence which has that profile. Thus, every valid linear complexity profile is possible and, since we already know that a

profile must be valid if it is possible, we have shown that a linear complexity profile is possible if and only if it is valid.

It can be seen from Theorem 3.4.2 that, for a given integer n, the number of n-bit sequences with a particular valid linear complexity profile (L(1),L(2),...,L(n)) depends only on the final value L(n) of the profile, and is independent of the values L(1),L(2),...,L(n-1). For example, for even values of n there are equally many n-bit sequences with a perfect linear complexity profile (see Example 3.2.4) as there are n-bit sequences with linear complexity profile $(0,0,\ldots,0,\frac{n}{2},\frac{n}{2},\ldots,\frac{n}{2})$. The number of sequences in each case is, in fact, $2^{n/2}$. These two profiles are illustrated in Figure 3.4.1 below for n = 10.

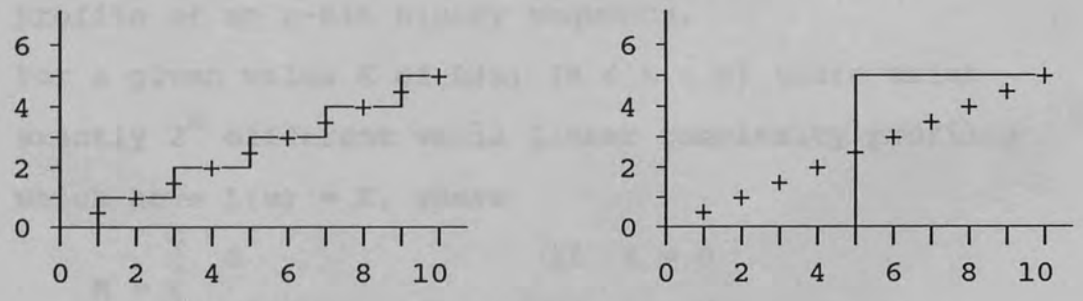


Figure 3.4.1. Two profiles with the same final value

As a corollary of Theorem 3.4.2 we can write down an expression for the number of n-bit sequences with a perfect linear complexity profile (i.e. with linear complexity profile $(1,1,2,2,3,3,\ldots,\lceil n/2 \rceil)$:-

Corollary 3.4.3.

The number of n-bit sequences with a perfect linear complexity profile is 2^M, where

$M = \lfloor n/2 \rfloor$.

□

Having enumerated the number of sequences with a given profile, we now derive an expression for the number of different profiles with the same final value :-

Theorem 3.4.4.

Let $(L(1),L(2),\ldots,L(n))$ denote the linear complexity profile of an n-bit binary sequence.
For a given value K of L(n) $(0 \leqslant K \leqslant n)$ there exist exactly 2^M different valid linear complexity profiles which have L(n) = K, where

$$M = \begin{cases} 0 & \text{if } K = 0 \\ \min(K-1,n-K) & \text{if } 1 \leqslant K \leqslant n \end{cases}$$

Proof

The proof is by induction.

Let $\#(L(n) = K)$ denote the number of distinct valid linear complexity profiles $(L(1),L(2),\ldots,L(n))$ with $L(n) = K$.

The result is true for $n = 1$ trivially,
since the only possible 1-bit sequences, 0 and 1, have linear complexity profiles (0) and (1) respectively,
and thus $\#(L(1) = 0) = 1$ and $\#(L(1) = 1) = 1$.

Suppose the result is true for $n = N$.
Then $\#(L(N) = K) = 2^M$, where

$$M = \begin{cases} 0 & \text{if } K = 0 \\ \min(K-1,N-K) & \text{if } 1 \leqslant K \leqslant N \end{cases}$$

We will show that the result also holds for $n = N+1$
(i.e. that $\#(L(N+1) = K) = 2^M$, where

$$M = \begin{cases} 0 & \text{if } K = 0 \\ \min(K-1,N+1-K) & \text{if } 1 \leqslant K \leqslant N+1 \end{cases}).$$

We consider 4 cases :-

(i) $K = 0$:

$L(N+1) = 0 \Rightarrow L(N) = 0.$

Therefore

$$\#(L(N+1) = 0) = \#(L(N) = 0) = 1.$$

(ii) $0 < K \leqslant \frac{N+1}{2}$:

$L(N+1) \leqslant \frac{N+1}{2} \Rightarrow L(N+1) = L(N)$

(since $(N+1, L(N+1))$ is on or below the line $y = x/2$).

Therefore

$$\#(L(N+1) = K) = \#(L(N) = K) = 2^M$$

where $M = \min(K-1, N-K)$

$= \min(K-1, N+1-K)$. (since $K \leqslant N+1-K$)

(iii) $\frac{N+1}{2} < K \leqslant N$:

If $\frac{N+1}{2} < L(N+1) \leqslant N$ then

either $L(N+1) = L(N)$ or $L(N+1) = N+1-L(N)$

(since $(N+1, L(N+1))$ is above the line $y = x/2$).

Therefore

$$\#(L(N+1) = K)$$

$$= \#(L(N) = K) + \#(L(N) = N+1-K)$$

(since $K \neq N+1-K$)

$$= 2^{M(1)} + 2^{M(2)}$$

where $M(1) = \min(K-1, N-K)$

and $M(2) = \min(N-K, K-1)$.

Therefore $\#(L(N+1) = K) = 2^{M}$

where $M = \min(K-1, N-K) + 1$

$= \min(K-1, N+1-K)$. (since $N-K < K-1$)

(iv) $K = N+1$:

$L(N+1) = N+1 \Rightarrow L(N) = 0$ (see Example 3.2.2).

Therefore

$$\#(L(N+1) = N+1) = \#(L(N) = 0)$$

$$= 1.$$

By combining the 4 cases above it can be seen that

$\#(L(N+1) = K) = 2^{M}$, where

$$M = \begin{cases} 0 & \text{if } K = 0 \\ \min(K-1, N+1-K) & \text{if } 1 \leqslant K \leqslant N+1 \end{cases}$$

Hence the result holds for $n = N+1$,

and therefore the result holds for all $n \geqslant 1$ by induction.

□

By combining Theorems 3.4.2 and 3.4.4 we can obtain an expression for the number of n-bit sequences with local linear complexity L. This result was proved by Rueppel [15] by a different route.

Theorem 3.4.5.

The number of n-bit binary sequences with local linear complexity L (L = 0,1,...,n) is 2^M, where

$$M = \begin{cases} 0 & \text{if } L = 0 \\ \min(2L-1, 2n-2L) & \text{if } 1 \leqslant L \leqslant n \end{cases}$$

Proof

By Theorem 3.4.2, for a fixed value of L(n), the number of n-bit sequences with linear complexity profile (L(1),L(2),...,L(n)) is the same for all possible values of L(1),L(2),...,L(n-1) for which the profile is valid. Thus, the number of n-bit sequences with local linear complexity L is the product of the number of sequences with a given valid linear complexity profile (L(1),L(2),...,L(n)), where L(n) = L, and the number of distinct valid linear complexity profiles with L(n) = L.

Hence, by Theorems 3.4.2 and 3.4.4, the number of n-bit sequences with local linear complexity L

$$= 2^{M(1)}.2^{M(2)}$$

where $M(1) = \min(L,n-L)$ for $L = 0,1,\ldots,n$

and
$$M(2) = \begin{cases} 0 & \text{for } L = 0 \\ \min(L-1,n-L) & \text{for } L = 1,2,\ldots,n \end{cases}$$

But $2^{M(1)}.2^{M(2)} = 2^{M}$, where

$$M = \begin{cases} \min(L,n-L) & \text{for } L = 0 \\ \min(L,n-L) + \min(L-1,n-L) & \text{for } L = 1,2,\ldots,n \end{cases}$$

$$= \begin{cases} 0 & \text{for } L = 0 \\ \min(2L-1,2n-2L) & \text{for } L = 1,2,\ldots,n \end{cases}$$

□

CHAPTER 4

SOME CONDITIONS ON THE LINEAR COMPLEXITY PROFILES OF CERTAIN BINARY SEQUENCES

4.1. INTRODUCTION

In Chapter 3 the linear complexity profile of an n-bit sequence $s_0 s_1 \ldots s_{n-1}$ was defined to be the vector $(L(1), L(2), \ldots, L(n))$, where $L(k)$ is the local linear complexity of $s_0 s_1 \ldots s_{k-1}$ $(k = 1,2,\ldots,n)$. The "perfect" linear complexity profile was then defined to be the linear complexity profile $(1,1,2,2,3,\ldots,\lceil n/2 \rceil)$, the profile which is "closest" to the line $y = x/2$.

In 1984 Rueppel conjectured in his thesis [14] that the sequence 110100010000000100..., defined by

$$s_i := \begin{cases} 1 & \text{if } i = 2^r - 1 \text{ for some } r \geqslant 0 \\ 0 & \text{else} \end{cases}$$

has the perfect linear complexity profile. This conjecture was later proved to be true by Dai [5]. In 1986 Wang and Massey extended the result by characterizing the set of binary sequences having the perfect linear complexity profile; in [18] they showed that an n-bit sequence $s_0 s_1 \ldots s_{n-1}$ has the perfect profile if and only if $s_0 = 1$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$. We will refer to this result as the "perfect profile characterization theorem", and return to it in Section 4.3.

In this chapter we generalise the result first proved in [18] to show that, if the bits of a binary sequence satisfy certain linear equations of a similar type to

those in the statement of the perfect profile characterization theorem, then the linear complexity profile of that sequence will be constrained in some way.

4.2. TWO FUNDAMENTAL LEMMAS

In this section we will prove two results that will themselves be used repeatedly when proving results later in this chapter. Note that most of the notation used was introduced in Chapter 2. In particular, recall that if the Berlekamp-Massey algorithm is applied to a binary sequence $s_0 s_1 \ldots s_{n-1}$ then, after k iterations the loop, the algorithm will have produced the connection polynomial $C_k(x)$ of an L(k)-stage LFSR that could be used to generate $s_0 s_1 \ldots s_{k-1}$, where L(k) is the local linear complexity of $s_0 s_1 \ldots s_{k-1}$. Recall also that the extended form of the algorithm given in Section 2.6 also produces a polynomial $P_k(x)$ of degree $< L(k)$ such that

$$P_k(x) = C_k(x)S_k(x) \pmod{x^k}$$

where $S_k(x) := s_0 + s_1 x + \ldots + s_{k-1}x^{k-1}$.

We now proceed to the first of our two results :-

Lemma 4.2.1.

Let $C_k(x)$ be the connection polynomial produced by the Berlekamp-Massey algorithm after k iterations of its loop, and let $P_k(x)$ be as defined in Section 2.6.

Suppose that two consecutive jumps in the linear complexity profile of a binary sequence occur with the $a(r)^{th}$ and $a(r+1)^{th}$ bits in the sequence.

Then

$$P_{a(r)-1}(x)C_k(x) + P_k(x)C_{a(r)-1}(x) = x^{a(r)-1} \quad (4.2.1)$$

for $a(r) \leqslant k \leqslant a(r+1)$

Proof

The proof is by induction.

Firstly, assume that the first and second jumps occur with the $a(1)^{th}$ and $a(2)^{th}$ bits in the sequence respectively. (We will assume throughout that the i^{th} jump in the profile occurs with the $a(i)^{th}$ bit in the sequence). Then the Berlekamp-Massey algorithm as described in Section 2.6 gives the following sequence of polynomials :-

k	$C_k(x)$	$P_k(x)$	$L(k)$
0	1	0	0
1	1	0	0
2	1	0	0
⋮	⋮	⋮	⋮
$a(1)-1$	1	0	0
$a(1)$	$1 + x^{a(1)}$	$x^{a(1)-1}$	$a(1)$
⋮	⋮	⋮	⋮
$a(2)-1$	$C_{a(2)-1}(x)$	$x^{a(1)-1}$	$a(1)$
$a(2)$	$C_{a(2)}(x)$	$x^{a(1)-1}$	$a(2)-a(1)$
⋮	⋮	⋮	⋮

Thus it can be seen that, for $a(1) \leqslant k \leqslant a(2)$,

$$P_{a(1)-1}(x)C_k(x) + P_k(x)C_{a(1)-1}(x) = 0.C_k(x) + x^{a(1)-1}.1$$
$$= x^{a(1)-1}$$

Hence, (4.2.1) holds for $r = 1$.

Now suppose (4.2.1) holds for $r = R-1$ $(R \geqslant 2)$

We will prove that (4.2.1) also holds for $r = R$.

Let $a(R) \leqslant k \leqslant a(R+1)$.

From the Berlekamp-Massey algorithm as stated in Section 2.6 it can be seen that, if $a(R) < k \leqslant a(R+1)$,

$$C_k(x) = C_{k-1}(x) + \delta.x^a B(x)$$

and $$P_k(x) = P_{k-1}(x) + \delta.x^a Q(x)$$

for some $\delta = 0$ or 1,

where $B(x) = C_{a(R)-1}(x)$, $Q(x) = P_{a(R)-1}(x)$

and $a = k-a(R)$.

Hence it can be seen that, if $a(R) < k \leqslant a(R+1)$,

$$C_k(x) = C_{a(R)}(x) + (\delta_1 x + \delta_2 x^2 + \ldots + \delta_{k-a(R)} x^{k-a(R)})C_{a(R)-1}$$

and $$P_k(x) = P_{a(R)}(x) + (\delta_1 x + \delta_2 x^2 + \ldots + \delta_{k-a(R)} x^{k-a(R)})P_{a(R)-1}$$

for some $\delta_1, \delta_2, \ldots, \delta_{k-a(R)} = 0$ or 1.

Thus, for all k such that $a(R) \leqslant k \leqslant a(R+1)$,

$$C_k(x) = C_{a(R)}(x) + F(x)C_{a(R)-1}(x)$$

and $$P_k(x) = P_{a(R)}(x) + F(x)P_{a(R)-1}(x)$$

for some polynomial F(x).

($F(x) = 0$ if $k = a(R)$)

Similarly, $$C_{a(R)-1}(x) = C_{a(R-1)}(x) + G(x)C_{a(R-1)-1}(x)$$

and $$P_{a(R)-1}(x) = P_{a(R-1)}(x) + G(x)P_{a(R-1)-1}(x)$$

for some polynomial G(x). (4.2.2)

Thus, $$C_k(x) = C_{a(R)}(x) + F(x)C_{a(R-1)}(x) + F(x)G(x)C_{a(R-1)-1}(x)$$

and $$P_k(x) = P_{a(R)}(x) + F(x)P_{a(R-1)}(x) + F(x)G(x)P_{a(R-1)-1}(x)$$

Therefore, $$P_{a(R)-1}(x)C_k(x) + P_k(x)C_{a(R)-1}(x)$$

$$= \left(C_{a(R)}(x) + F(x)C_{a(R-1)}(x) + F(x)G(x)C_{a(R-1)-1}(x)\right) \cdot \left(P_{a(R-1)}(x) + G(x)P_{a(R-1)-1}(x)\right)$$

$$+ \left(P_{a(R)}(x) + F(x)P_{a(R-1)}(x) + F(x)G(x)P_{a(R-1)-1}(x)\right) \cdot \left(C_{a(R-1)}(x) + G(x)C_{a(R-1)-1}(x)\right)$$

$$= \left[P_{a(R-1)}(x)C_{a(R)}(x) + P_{a(R)}(x)C_{a(R-1)}(x)\right]$$

$$+ F(x)\left[P_{a(R-1)}(x)C_{a(R-1)}(x) + P_{a(R-1)}(x)C_{a(R-1)}(x)\right]$$

$$+ F(x)G(x)\left[P_{a(R-1)-1}(x)C_{a(R-1)}(x) + P_{a(R-1)}(x)C_{a(R-1)-1}(x) + P_{a(R-1)}(x)C_{a(R-1)-1}(x) + P_{a(R-1)-1}(x)C_{a(R-1)}(x)\right]$$

$$+ G(x)\left[P_{a(R-1)-1}(x)C_{a(R)}(x) + P_{a(R)}(x)C_{a(R-1)-1}(x)\right]$$

$$+ F(x)G^2(x)\left[P_{a(R-1)-1}(x)C_{a(R-1)-1}(x) + P_{a(R-1)-1}(x)C_{a(R-1)-1}(x)\right]$$

$$= \left[P_{a(R-1)}(x)C_{a(R)}(x) + P_{a(R)}(x)C_{a(R-1)}(x)\right]$$

$$+ G(x)\left[P_{a(R-1)-1}(x)C_{a(R)}(x) + P_{a(R)}(x)C_{a(R-1)-1}(x)\right]$$

$$= \left[P_{a(R-1)}(x)C_{a(R)}(x) + P_{a(R)}(x)C_{a(R-1)}(x)\right]$$

$$+ G(x).x^{a(R-1)-1}$$

(since (4.2.1) holds for r = R-1)

From the Berlekamp-Massey algorithm as given in Section 2.6 it can also be seen that

$$C_{a(R)}(x) = C_{a(R)-1}(x) + x^{a(R)-a(R-1)}C_{a(R-1)-1}(x)$$

and $$P_{a(R)}(x) = P_{a(R)-1}(x) + x^{a(R)-a(R-1)}P_{a(R-1)-1}(x)$$

Therefore,

$$P_{a(R)-1}(x)C_k(x) + P_k(x)C_{a(R)-1}(x)$$

$$= \left(P_{a(R-1)}(x)C_{a(R)-1}(x) + P_{a(R)-1}(x)C_{a(R-1)}(x)\right)$$
$$+ x^{a(R)-a(R-1)}\left(P_{a(R-1)}(x)C_{a(R-1)-1}(x) + P_{a(R-1)-1}(x)C_{a(R-1)}(x)\right)$$
$$+ G(x).x^{a(R-1)-1}$$

$$= \left(P_{a(R-1)}(x)C_{a(R-1)}(x) + P_{a(R-1)}(x)C_{a(R-1)}(x)\right)$$
$$+ G(x)\left(P_{a(R-1)}(x)C_{a(R-1)-1}(x) + P_{a(R-1)-1}(x)C_{a(R-1)}(x)\right)$$
$$+ x^{a(R)-a(R-1)}\left(P_{a(R-1)}(x)C_{a(R-1)-1}(x) + P_{a(R-1)-1}(x)C_{a(R-1)}(x)\right)$$
$$+ G(x).x^{a(R-1)-1} \qquad \text{(by equation (4.2.2))}$$

$$= G(x).x^{a(R-1)-1} + x^{a(R)-a(R-1)}.x^{a(R-1)-1} + G(x).x^{a(R-1)}$$

(since (4.2.1) holds for r = R-1)

$$= x^{a(R)-1}$$

Therefore (4.2.1) holds for r = R.

□

We now move on to our second lemma :-

Lemma 4.2.2.

Let $C_n(x)$ be the connection polynomial produced when the Berlekamp-Massey algorithm is applied to the n-bit sequence $s_0 s_1 \ldots s_{n-1}$, and let $P_n(x)$ be as defined in Section 2.6.

Let $S(x) := s_0 + s_1 x + s_2 x^2 + \ldots ,$

$T(x) := s_0 + s_2 x + s_4 x^2 + \ldots ,$

$U(x) := s_1 + s_3 x + s_5 x^2 + \ldots .$

Suppose that

$$x^{\alpha} A(x) T(x^2) + x^{\beta} B(x) U(x^2) + x^{\gamma} C(x) S(x^2) \equiv \delta \pmod{x^m}$$

where $m \leqslant \min(\alpha+n, \beta+n-1, \gamma+2n)$

for some polynomials $A(x)$, $B(x)$, $C(x)$, and some integers α, β, $\gamma \geqslant 0$ and $\delta = 0$ or 1.

Then

$$x^{\gamma} C(x) P_n^2(x) + x^{\alpha} A(x) P_n(x) C_n(x) + (x^{\alpha+1} A(x) + x^{\beta} B(x)) [P_n(x) C_n(x)]' + \delta C_n^2(x) \equiv 0 \pmod{x^m}.$$

Proof

We begin by expressing $S(x^2)$, $T(x^2)$ and $U(x^2)$ in terms of $S(x)$:-

$$S(x^2) = s_0 + s_1 x^2 + s_2 x^4 + \ldots$$
$$= (s_0 + s_1 x + s_2 x^2 + \ldots)^2$$
$$= S^2(x)$$

$$T(x^2) = s_0 + s_2x^2 + s_4x^4 + \ldots$$
$$= (s_0 + s_1x + s_2x^2 + \ldots) + (s_1x + s_3x^3 + s_5x^5 + \ldots)$$
$$= (s_0 + s_1x + s_2x^2 + \ldots) + x(s_0 + s_1x + s_2x^2 + \ldots)'$$
$$= S(x) + xS'(x)$$

$$U(x^2) = s_1 + s_3x^2 + s_5x^4 + \ldots$$
$$= (s_0 + s_1x + s_2x^2 + \ldots)'$$
$$= S'(x)$$

By supposition,

$$x^{\alpha}A(x)T(x^2) + x^{\beta}B(x)U(x^2) + x^{\gamma}C(x)S(x^2) \equiv \delta \pmod{x^m}$$

Thus,

$$x^{\alpha}A(x)(S(x) + xS'(x)) + x^{\beta}B(x)S'(x) + x^{\gamma}C(x)S^2(x) \equiv \delta \pmod{x^m}$$

and so

$$x^{\gamma}C(x)S^2(x) + x^{\alpha}A(x)S(x) + (x^{\alpha+1}A(x) + x^{\beta}B(x))S'(x) \equiv \delta \pmod{x^m} \quad (4.2.3)$$

Now consider the n-bit sequence $s_0s_1\ldots s_{n-1}$
and recall that $S_n(x) := s_0 + s_1x + \ldots + s_{n-1}x^{n-1}$.

By inspection it can be seen that

$$x^{\gamma}C(x)S^2(x) + x^{\alpha}A(x)S(x) + (x^{\alpha+1}A(x) + x^{\beta}B(x))S'(x)$$
$$\equiv x^{\gamma}C(x)S_n^2(x) + x^{\alpha}A(x)S_n(x) + (x^{\alpha+1}A(x) + x^{\beta}B(x))S_n'(x) \pmod{x^{\min(\alpha+n,\beta+n-1,\gamma+2n)}}$$

Hence, since $m \leqslant \min(\alpha+n,\beta+n-1,\gamma+2n)$, (4.2.3) gives

$$x^{\gamma}C(x)S_n^2(x) + x^{\alpha}A(x)S_n(x) + (x^{\alpha+1}A(x) + x^{\beta}B(x))S_n'(x) \equiv \delta \pmod{x^m} \quad (4.2.4)$$

But from equation (2.6.5),

$$S_n(x) = (P_n(x) + x^n G_n(x)) / C_n(x)$$

for some polynomial $G_n(x)$

$$= D_n(x) / C_n(x)$$

where $D_n(x) = P_n(x) + x^n G_n(x)$.

Therefore

$$S_n^2(x) = (P_n(x) + x^n G_n(x))^2 / C_n^2(x)$$

$$= (P_n^2(x) + x^{2n} G_n^2(x)) / C_n^2(x)$$

(since the coefficients of our polynomials are in GF(2))

$$= E_n(x) / C_n^2(x)$$

where $E_n(x) = P_n^2(x) + x^{2n} G_n^2(x)$,

and

$$S_n'(x) = [(P_n(x) + x^n G_n(x)) / C_n(x)]'$$

$$= (C_n(x)[P_n(x) + x^n G_n(x)]'$$

$$+ C_n'(x)(P_n(x) + x^n G_n(x))) / C_n^2(x)$$

$$= (C_n(x)P_n'(x) + nx^{n-1}C_n(x)G_n(x) + x^n C_n(x)G_n'(x)$$

$$+ C_n'(x)P_n(x) + x^n C_n'(x)G_n(x)) / C_n^2(x)$$

$$= F_n(x) / C_n^2(x).$$

Thus, (4.2.4) gives

$$x^\gamma C(x)E_n(x)/C_n^2(x) + x^\alpha A(x)D_n(x)/C_n(x)$$

$$+ (x^{\alpha+1}A(x) + x^\beta B(x))F_n(x)/C_n^2(x) \equiv \delta \pmod{x^m}$$

and so

$$x^\gamma C(x)E_n(x) + x^\alpha A(x)D_n(x)C_n(x)$$

$$+ (x^{\alpha+1}A(x) + x^\beta B(x))F_n(x) + \delta C_n^2(x) \equiv 0 \pmod{x^m}$$

But $D_n(x) \equiv P_n(x) \pmod{x^n}$,

$E_n(x) \equiv P_n^2(x) \pmod{x^{2n}}$,

$F_n(x) \equiv P_n(x)C_n'(x) + P_n'(x)C_n(x) \pmod{x^{n-1}}$.

Hence, since $m \leqslant \min(\alpha+n, \beta+n-1, \gamma+2n)$,

$$x^\gamma C(x)P_n^2(x) + x^\alpha A(x)P_n(x)C_n(x) + (x^{\alpha+1}A(x) + x^\beta B(x))[P_n(x)C_n(x)]' + \delta C_n^2(x) \equiv 0 \pmod{x^m}$$

□

4.3. THE PERFECT PROFILE CHARACTERIZATION THEOREM AND AN EXTENSION

The perfect profile characterization theorem states that a binary sequence $s_0 s_1 \ldots s_{n-1}$ has the perfect linear complexity profile if and only if $s_0 = 1$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$. In this section we present a new proof of this theorem (it was originally proved in [18]), and extend the result by considering also those sequences with $s_0 = 0$ and $s_{2i} = s_{2i-1} + s_{i-1}$. Thus, we consider all sequences for which $s_{2i} = s_{2i-1} + s_{i-1}$ for $i \geqslant 1$.

We begin by proving a lemma which is a more general version of Lemma 3 of [18] :-

Lemma 4.3.1.

Let $C_n(x)$ be the connection polynomial produced when the Berlekamp-Massey algorithm is applied to the n-bit sequence $s_0 s_1 \ldots s_{n-1}$, and let $P_n(x)$ be as defined in Section 2.6.

If $s_0 = \delta$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$

then $x^2 P_n^2(x) + P_n(x) C_n(x) + (x+x^2)[P_n(x) C_n(x)]' + \delta C_n^2(x) \equiv 0 \pmod{x^n}$

Proof

Consider the infinite binary sequence $s_0 s_1 s_2 \ldots$

and let $S(x) := s_0 + s_1 x + s_2 x^2 + \ldots$,

$T(x) := s_0 + s_2 x + s_4 x^2 + \ldots$

and $U(x) := s_1 + s_3 x + s_5 x^2 + \ldots$

Then the coefficient of x^{2i} in

$T(x^2) + x^2 U(x^2) + x^2 S(x^2)$ is

$$\begin{cases} s_0 & \text{if } i = 0 \\ s_{2i} + s_{2i-1} + s_{i-1} & \text{if } i \geqslant 1 \end{cases}$$

Thus, if $s_0 = \delta$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$

then $T(x^2) + x^2 U(x^2) + x^2 S(x^2) \equiv \delta \pmod{x^n}$

We now invoke Lemma 4.2.2, taking $\alpha = 0$, $\beta = 2$, $\gamma = 2$, $A(x) = B(x) = C(x) = 1$ and $m = n$.

Hence, if $s_0 = \delta$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$

then

$$x^2 P_n^2(x) + P_n(x) C_n(x) + (x+x^2)[P_n(x) C_n(x)]' + \delta C_n^2(x) \equiv 0 \pmod{x^n}$$

□

We now present a new proof of the perfect profile characterization theorem, using Lemmas 4.2.1 and 4.3.1. Note that in this proof (and henceforth) we will often, for simplicity, write C_k and P_k instead of $C_k(x)$ and $P_k(x)$.

Theorem 4.3.2.

An n-bit sequence $s_0 s_1 \ldots s_{n-1}$ has the perfect linear complexity profile if and only if $s_0 = 1$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$.

Proof

Suppose that $s_0 = 1$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$.
Then, since $s_0 = 1$, the first jump in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ has height 1.

From Section 3.3 it can be seen that, if two consecutive jumps (the r^{th} and $(r+1)^{th}$, say) in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ occur with the $a(r)^{th}$ and $a(r+1)^{th}$ bits in the sequence, then there exists an integer m with $a(r) < 2m < a(r+1)$ and $L(2m) = m$.
Let $a(r+1) = 2m+j$,
so $L(a(r+1)) = a(r+1) - L(a(r+1)-1) = 2m+j - m = m+j$
=> the $(r+1)^{th}$ jump in the profile has height j.

Assume that the first r jumps in the profile all have height 1, and in particular that the r^{th} jump has height 1,

so $L(a(r)) = a(r) - L(a(r)-1)$

$\Rightarrow \quad m = a(r) - (m-1)$

$\Rightarrow \quad a(r) = 2m-1.$

We will show that $j = 1$, so that, by induction, all the jumps in the profile have height 1.

Since $2.L(2m+i-1) \leqslant 2m+i-1$ for $1 \leqslant i \leqslant j$, the Berlekamp-Massey algorithm as described in Section 2.6 gives the following sequence of polynomials :-

k	$C_k(x)$	$P_k(x)$	$L(k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$2m-2$	C_{2m-2}	P_{2m-2}	$m-1$
$2m-1$	C_{2m-1}	P_{2m-1}	m
$2m$	C_{2m}	P_{2m}	m
$2m+1$	C_{2m}	P_{2m}	m
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$2m+j-1$	C_{2m}	P_{2m}	m
$2m+j$	$C_{2m} + x^{j+1}C_{2m-2}$	$P_{2m} + x^{j+1}P_{2m-2}$	$m+j$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Thus, $C_{2m+j} = C_{2m} + x^{j+1}C_{2m-2}$

and $P_{2m+j} = P_{2m} + x^{j+1}P_{2m-2}$

Since $s_0 = 1$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$,

$$x^2P_{2m+j}^2 + P_{2m+j}C_{2m+j} + (x+x^2)[P_{2m+j}C_{2m+j}]' + C_{2m+j}^2$$
$$\equiv 0 \pmod{x^{2m+j}} \qquad \text{(by Lemma 4.3.1)}$$

$$\Rightarrow\ x^2(P_{2m} + x^{j+1}P_{2m-2})^2$$
$$+ (P_{2m} + x^{j+1}P_{2m-2})(C_{2m} + x^{j+1}C_{2m-2})$$
$$+ (x+x^2)[(P_{2m} + x^{j+1}P_{2m-2})(C_{2m} + x^{j+1}C_{2m-2})]'$$
$$+ (C_{2m} + x^{j+1}C_{2m-2})^2 \equiv 0 \pmod{x^{2m+j}}$$

$$\Rightarrow\ x^2P_{2m}^2 + P_{2m}C_{2m} + (x+x^2)[P_{2m}C_{2m}]' + C_{2m}^2$$
$$+ x^{2j+2}(x^2P_{2m-2}^2 + P_{2m-2}C_{2m-2} + (x+x^2)[P_{2m-2}C_{2m-2}]' + C_{2m-2}^2)$$
$$+ x^{j+1}(P_{2m}C_{2m-2} + P_{2m-2}C_{2m})$$
$$+ (x+x^2)[x^{j+1}(P_{2m}C_{2m-2} + P_{2m-2}C_{2m})]'$$
$$\equiv 0 \pmod{x^{2m+j}} \qquad (4.3.1)$$

But by Lemma 4.3.1,

$$x^2P_{2m}^2 + P_{2m}C_{2m} + (x+x^2)[P_{2m}C_{2m}]' + C_{2m}^2 \equiv 0 \pmod{x^{2m}}$$

And by Theorem 2.6.1,

$\max(\deg C_{2m},\ 1 + \deg P_{2m}) = L(2m) = m$

and so either $\deg C_{2m} = 1 + \deg P_{2m} = m$

or $\deg C_{2m} = m > 1 + \deg P_{2m}$

or $1 + \deg P_{2m} = m > \deg C_{2m}$

If $\deg C_{2m} = 1 + \deg P_{2m} = m$ then

$$\deg (x^2P_{2m}^2 + P_{2m}C_{2m} + (x+x^2)[P_{2m}C_{2m}]' + C_{2m}^2) = 2m$$

and the same result also holds if

$\deg C_{2m} = m > 1 + \deg P_{2m}$ or if

$$1 + \deg P_{2m} = m > \deg C_{2m}.$$

Therefore $x^2P_{2m} + P_{2m}C_{2m} + (x+x^2)[P_{2m}C_{2m}]' + C_{2m}^2 = x^{2m}$

Also by Lemma 4.3.1,

$$x^2P_{2m-2}^2 + P_{2m-2}C_{2m-2} + (x+x^2)[P_{2m-2}C_{2m-2}]' + C_{2m-2}^2 \equiv 0 \pmod{x^{2m-2}}$$

And by Theorem 2.6.1,

$$\max (\deg C_{2m-2}, 1 + \deg P_{2m-2}) = L(2m-2) = m-1$$

so that $\deg (x^2P_{2m-2}^2 + P_{2m-2}C_{2m-2} + (x+x^2)[P_{2m-2}C_{2m-2}]' + C_{2m-2}^2) \leqslant 2m-2$

Therefore $x^2P_{2m-2}^2 + P_{2m-2}C_{2m-2} + (x+x^2)[P_{2m-2}C_{2m-2}]' + C_{2m-2}^2 = 0$ or x^{2m-2}

$(= \alpha x^{2m-2}$, say, where $\alpha = 0$ or $1)$

By Lemma 4.2.1, $P_{2m}C_{2m-2} + P_{2m-2}C_{2m} = x^{2m-2}$

Therefore equation (4.3.1) gives

$$x^{2m} + x^{2j+2}.\alpha x^{2m-2} + x^{j+1}x^{2m-2} + (x+x^2)[x^{j+1}x^{2m-2}]' \equiv 0 \pmod{x^{2m+j}}$$

$$\Rightarrow x^{2m} + x^{2m+j-1} + (x+x^2)(j-1)x^{2m+j-2} \equiv 0 \pmod{x^{2m+j}}$$

$$\Rightarrow x^{2m} + j.x^{2m+j-1} \equiv 0 \pmod{x^{2m+j}}$$

$$\Rightarrow j = 1$$

Thus we have shown that if $s_0 = 1$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$ then all the jumps in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ have height 1.

But if all the jumps in the linear complexity profile of a binary sequence have height 1 then that sequence must have the perfect linear complexity profile, and thus $s_0 s_1 \ldots s_{n-1}$ has the perfect linear complexity profile.

To complete the proof we use a counting arguement :-

The number of n-bit sequences with $s_0 = 1$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$ is $2^{\lfloor n/2 \rfloor}$, since these equations fix all the even-indexed bits in the sequence but none of the odd-indexed ones.

And by Corollary 3.4.3, the number of n-bit sequences with the perfect linear complexity profile is also $2^{\lfloor n/2 \rfloor}$.

Thus, every sequence $s_0 s_1 \ldots s_{n-1}$ which has the perfect profile must satisfy the equations $s_0 = 1$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$.

□

We now move on to consider sequences with $s_{2i} = s_{2i-1} + s_{i-1}$, as in Theorem 4.3.2, but with $s_0 = 0$ instead of $s_0 = 1$. Firstly we prove that the linear complexity profiles of such sequences can only have jumps of certain heights :-

Theorem 4.3.3.

Let $s_0 s_1 \ldots s_{n-1}$ be an n-bit sequence with $s_0 = 0$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$.
Then the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ has no jumps of odd height > 1.

Proof

The sequence $s_0 s_1 \ldots s_{n-1} = 00\ldots0$ satisfies the theorem trivially, since its linear complexity profile has no jumps.
For any other sequence satisfying the given equations the first jump in the profile must have even height, since it must occur with bit s_{2i-1} (i.e. the $2i^{th}$ bit in the sequence) for some integer i, and thus it must have height 2i.

Suppose that two consecutive jumps in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ (the r^{th} and $(r+1)^{th}$, say) occur with the $a(r)^{th}$ and $a(r+1)^{th}$ bits in the sequence.
From Section 3.3 it can be seen that there exists an integer m with $a(r) < 2m < a(r+1)$ and $L(2m) = m$.

Let $a(r+1) = 2m+j$,

so $L(a(r+1)) = a(r+1) - L(a(r+1)-1) = 2m+j - m = m+j$

$\Rightarrow$ $(r+1)^{th}$ jump has height j.

And let $a(r) = 2m-h$,

so $L(a(r)) = a(r) - L(a(r)-1)$

$\Rightarrow$ $m = 2m-h - L(a(r)-1)$

$\Rightarrow$ $L(a(r)-1) = m-h$

$\Rightarrow$ r^{th} jump has height h.

Since $2.L(2m+i-1) \leqslant 2m+i-1$ for $1 \leqslant i \leqslant j$, the Berlekamp-Massey algorithm as described in Section 2.6 gives the following sequence of polynomials :-

k	$C_k(x)$	$P_k(x)$	$L(k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$2m-h-1$	C_{2m-h-1}	P_{2m-h-1}	$m-h$
$2m-h$	C_{2m-h}	P_{2m-h}	m
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$2m-1$	C_{2m-1}	P_{2m-1}	m
$2m$	C_{2m}	P_{2m}	m
$2m+1$	C_{2m}	P_{2m}	m
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$2m+j-1$	C_{2m}	P_{2m}	m
$2m+j$	$C_{2m} + x^{j+h}C_{2m-h-1}$	$P_{2m} + x^{j+h}P_{2m-h-1}$	$m+j$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Thus, $C_{2m+j} = C_{2m} + x^{j+h}C_{2m-h-1}$

and $P_{2m+j} = P_{2m} + x^{j+h}P_{2m-h-1}$

By Lemma 4.3.1,

since $s_0 = 0$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$,

$$x^2P_{2m+j}^2 + P_{2m+j}C_{2m+j} + (x+x^2)[P_{2m+j}C_{2m+j}]' \equiv 0 \pmod{x^{2m+j}}$$

$$\Rightarrow x^2(P_{2m} + x^{j+h}P_{2m-h-1})^2 + (P_{2m} + x^{j+h}P_{2m-h-1})(C_{2m} + x^{j+h}C_{2m-h-1}) + (x+x^2)[(P_{2m} + x^{j+h}P_{2m-h-1})(C_{2m} + x^{j+h}C_{2m-h-1})]' \equiv 0 \pmod{x^{2m+j}}$$

$$\Rightarrow x^2P_{2m}^2 + P_{2m}C_{2m} + (x+x^2)[P_{2m}C_{2m}]' + x^{2j+2h}(x^2P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (x+x^2)[P_{2m-h-1}C_{2m-h-1}]') + x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m}) + (x+x^2)[x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m})]' \equiv 0 \pmod{x^{2m+j}} \quad (4.3.2)$$

But by Lemma 4.3.1,

$$x^2P_{2m}^2 + P_{2m}C_{2m} + (x+x^2)[P_{2m}C_{2m}]' \equiv 0 \pmod{x^{2m}}$$

And by Theorem 2.6.1,

$\max(\deg C_{2m}, 1 + \deg P_{2m}) = L(2m) = m$

so that $\deg(x^2P_{2m}^2 + P_{2m}C_{2m} + (x+x^2)[P_{2m}C_{2m}]') \leqslant 2m$

Therefore $x^2P_{2m}^2 + P_{2m}C_{2m} + (x+x^2)[P_{2m}C_{2m}]'$

$= 0$ or x^{2m} $(= \alpha x^{2m}$, say, where $\alpha = 0$ or $1)$

Also by Lemma 4.3.1,

$$x^2P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (x+x^2)[P_{2m-h-1}C_{2m-h-1}]' \equiv 0 \pmod{x^{2m-h-1}}$$

And by Theorem 2.6.1,

$$\max(\deg C_{2m-h-1},\ 1 + \deg P_{2m-h-1}) = L(2m-h-1) = m-h$$

so that $\deg(x^2P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (x+x^2)[P_{2m-h-1}C_{2m-h-1}]') \leqslant 2m-2h$

Therefore $x^2P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (x+x^2)[P_{2m-h-1}C_{2m-h-1}]'$

$$= \begin{cases} 0 \text{ or } x^{2m-2} & \text{if } h = 1 \\ \quad (= \beta x^{2m-2}, \text{ say, where } \beta = 0 \text{ or } 1) & \\ 0 & \text{if } h > 1 \end{cases}$$

By Lemma 4.2.1, $P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m} = x^{2m-h-1}$

Therefore equation (4.3.2) gives

$$x^{j+h}x^{2m-h-1} + (x+x^2)[x^{j+h}x^{2m-h-1}]' + \alpha x^{2m}$$

$$+ \begin{cases} x^{2j+2}.\beta x^{2m-2} \equiv 0 \pmod{x^{2m+j}} & \text{if } h = 1 \\ 0 \equiv 0 \pmod{x^{2m+j}} & \text{if } h > 1 \end{cases}$$

$$\Rightarrow \begin{cases} x^{2m+j-1} + (x+x^2)(j-1)x^{2m+j-2} + \alpha x^{2m} + \beta x^{2m+2j} \\ \qquad \equiv 0 \pmod{x^{2m+j}} & \text{if } h = 1 \\ x^{2m+j-1} + (x+x^2)(j-1)x^{2m+j-2} + \alpha x^{2m} \\ \qquad \equiv 0 \pmod{x^{2m+j}} & \text{if } h > 1 \end{cases}$$

$$\Rightarrow \quad x^{2m+j-1} + (j-1)(x^{2m+j-1} + x^{2m+j}) + \alpha x^{2m} \equiv 0 \pmod{x^{2m+j}}$$

$$\Rightarrow \quad j.x^{2m+j-1} + \alpha x^{2m} \equiv 0 \pmod{x^{2m+j}}$$

=> either j is even

or j = 1

□

Having shown that the only odd height jumps that the linear complexity profile of sequences of this type can have are of height 1, we now show that no two consecutive jumps can have height 1 :-

Theorem 4.3.4.

Let $s_0 s_1 \ldots s_{n-1}$ be an n-bit sequence with $s_0 = 0$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$.
Then the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ cannot have two consecutive jumps of height 1.

Proof

Since $s_0 = 0$, the first jump cannot have height 1.
Assume that the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ has two consecutive jumps of height 1 (the r^{th} and $(r+1)^{th}$ jumps, say), and that these jumps occur with the $a(r)^{th}$ and $a(r+1)^{th}$ bits in the sequence.

Then, from Section 3.3 it can be seen that there exists an integer m such that $a(r) < 2m < a(r+1)$ and $L(2m) = m$.

Also $L(a(r)) = a(r) - L(a(r)-1) \Rightarrow m = a(r) - (m-1)$

$\Rightarrow a(r) = 2m-1$,

and $L(a(r+1)) = a(r+1) - L(a(r+1)-1)$

$\Rightarrow m+1 = a(r+1) - m \Rightarrow a(r+1) = 2m+1$.

Assume also that these are the first two consecutive jumps of height 1, and that the $(r-1)^{th}$ jump had height $h > 1$.

Then $L(a(r-1)) = a(r-1) - L(a(r-1)-1)$

$\Rightarrow m-1 = a(r-1) - (m-h-1)$

$\Rightarrow a(r-1) = 2m-h-2$

By Theorem 4.3.3, h is even.

Now $2.L(2m-2) \leqslant 2m-2$, $2.L(2m-1) > 2m-1$ and $2.L(2m) \leqslant 2m$, and thus the Berlekamp-Massey algorithm as described in Section 2.6 gives the following sequence of polynomials :-

k	$C_k(x)$	$P_k(x)$	$L(k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$2m-h-3$	C_{2m-h-3}	P_{2m-h-3}	$m-h-1$
$2m-h-2$	C_{2m-h-2}	P_{2m-h-2}	$m-1$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$2m-2$	C_{2m-2}	P_{2m-2}	$m-1$
$2m-1$	$C_{2m-2} + x^{h+1}C_{2m-h-3}$	$P_{2m-2} + x^{h+1}P_{2m-h-3}$	m
$2m$	$C_{2m-1} + \delta x C_{2m-2}$	$P_{2m-1} + \delta x P_{2m-2}$	m
$2m+1$	$C_{2m} + x^2 C_{2m-2}$	$P_{2m} + x^2 P_{2m-2}$	$m+1$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

where $\delta = 0$ or 1

Thus, $C_{2m+1} = C_{2m} + x^2 C_{2m-2}$

$C_{2m} = C_{2m-1} + \delta x C_{2m-2}$

$C_{2m-1} = C_{2m-2} + x^{h+1} C_{2m-h-3}$

and therefore $C_{2m+1} = C_{2m-1} + \delta x C_{2m-2} + x^2 C_{2m-2}$

$= (x^2 + \delta x + 1) C_{2m-2} + x^{h+1} C_{2m-h-3}$

Similarly $P_{2m+1} = (x^2 + \delta x + 1) P_{2m-2} + x^{h+1} P_{2m-h-3}$

By Lemma 4.3.1,

since $s_0 = 0$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$,

$$x^2 P_{2m+1}^2 + P_{2m+1} C_{2m+1} + (x+x^2)[P_{2m+1} C_{2m+1}]' \equiv 0 \pmod{x^{2m+1}}$$

$$\Rightarrow\ x^2((x^2+\delta x+1)P_{2m-2} + x^{h+1}P_{2m-h-3})^2$$
$$+ ((x^2+\delta x+1)P_{2m-2} + x^{h+1}P_{2m-h-3})((x^2+\delta x+1)C_{2m-2} + x^{h+1}C_{2m-h-3})$$
$$+ (x+x^2).[((x^2+\delta x+1)P_{2m-2} + x^{h+1}P_{2m-h-3}).((x^2+\delta x+1)C_{2m-2} + x^{h+1}C_{2m-h-3})]'$$
$$\equiv 0 \pmod{x^{2m+1}}$$

$$\Rightarrow (x^2+\delta x+1)^2(x^2P_{2m-2}^2 + P_{2m-2}C_{2m-2} + (x+x^2)[P_{2m-2}C_{2m-2}]')$$
$$+ x^{2h+2}(x^2P_{2m-h-3}^2 + P_{2m-h-3}C_{2m-h-3} + (x+x^2)[P_{2m-h-3}C_{2m-h-3}]')$$
$$+ (x^2+\delta x+1)x^{h+1}(P_{2m-2}C_{2m-h-3} + P_{2m-h-3}C_{2m-2})$$
$$+ (x+x^2)[(x^2+\delta x+1)x^{h+1}(P_{2m-2}C_{2m-h-3} + P_{2m-h-3}C_{2m-2})]'$$
$$\equiv 0 \pmod{x^{2m+1}} \qquad (4.3.3)$$

But by Lemma 4.3.1,

$$x^2P_{2m-2}^2 + P_{2m-2}C_{2m-2} + (x+x^2)[P_{2m-2}C_{2m-2}]' \equiv 0 \pmod{x^{2m-2}}$$

And by Theorem 2.6.1,

$$\max(\deg C_{2m-2},\ 1 + \deg P_{2m-2}) = L(2m-2) = m-1$$

so that $\deg(x^2P_{2m-2}^2 + P_{2m-2}C_{2m-2} + (x+x^2)[P_{2m-2}C_{2m-2}]') \leqslant 2m-2$

Therefore $x^2P_{2m-2}^2 + P_{2m-2}C_{2m-2} + (x+x^2)[P_{2m-2}C_{2m-2}]' = \alpha x^{2m-2}$, where $\alpha = 0$ or 1.

Also by Lemma 4.3.1,

$$x^2P_{2m-h-3}^2 + P_{2m-h-3}C_{2m-h-3} + (x+x^2)[P_{2m-h-3}C_{2m-h-3}]' \equiv 0 \pmod{x^{2m-h-3}}$$

And by Theorem 2.6.1,

$$\max(\deg C_{2m-h-3},\ 1 + \deg P_{2m-h-3}) = L(2m-h-3) = m-h-1$$

so that $\deg(x^2P_{2m-h-3}^2 + P_{2m-h-3}C_{2m-h-3} + (x+x^2)[P_{2m-h-3}C_{2m-h-3}]') \leqslant 2m-2h-2 \leqslant 2m-h-4$

Therefore $x^2P_{2m-h-3}^2 + P_{2m-h-3}C_{2m-h-3} + (x+x^2)[P_{2m-h-3}C_{2m-h-3}]' = 0$

Also, by Lemma 4.2.1,

$$P_{2m-2}C_{2m-h-3} + P_{2m-h-3}C_{2m-2} = x^{2m-h-3}$$

Therefore equation (4.3.3) gives

$$(x^2+\delta x+1)^2\alpha x^{2m-2} + (x^2+\delta x+1)x^{h+1}x^{2m-h-3} + (x+x^2)[(x^2+\delta x+1)x^{h+1}x^{2m-h-3}]' \equiv 0 \pmod{x^{2m+1}}$$

$$\Rightarrow (x^4+\delta x^2+1).\alpha x^{2m-2} + (x^2+\delta x+1)x^{2m-2} + (x+x^2)\delta x^{2m-2} \equiv 0 \pmod{x^{2m+1}}$$

$$\Rightarrow \alpha\delta x^{2m} + \alpha x^{2m-2} + x^{2m} + \delta x^{2m-1} + x^{2m-2} + \delta x^{2m-1} + \delta x^{2m} \equiv 0 \pmod{x^{2m+1}}$$

$$\Rightarrow (\alpha + 1).x^{2m-2} + (\alpha\delta + 1 + \delta).x^{2m} \equiv 0 \pmod{x^{2m+1}}$$

$$\Rightarrow \alpha + 1 = 0 \quad \text{and} \quad \alpha\delta + 1 + \delta = 0$$

$$\Rightarrow \alpha = 1 \quad \text{and} \quad \delta + 1 + \delta = 0$$

$$\Rightarrow 1 = 0\ !$$

Thus we have a contradiction, so the linear complexity profile of $s_0s_1\ldots s_{n-1}$ cannot have two consecutive jumps of height 1.

□

4.4. SEQUENCES WHICH SATISFY A DIFFERENT SET OF LINEAR EQUATIONS

Section 4.3 dealt with n-bit sequences $s_0 s_1 \ldots s_{n-1}$ which satisfy the equation $s_{2i} = s_{2i-1} + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n-1}{2}$. In this section we will consider sequences which satisfy a different set of linear equations - we will consider n-bit sequences $s_0 s_1 \ldots s_{n-1}$ for which $s_{2i+1} = s_{2i} + s_i$ for $1 \leqslant i \leqslant \frac{n-2}{2}$. Note that a sequence of this type can be formed by taking a subsequence consisting of the first n-1 bits of one of the sequences discussed in Section 4.3 and adding an n^{th} bit to the beginning of it.

As in Section 4.3 we begin by proving a lemma :-

Lemma 4.4.1.

Let $C_n(x)$ be the connection polynomial produced when the Berlekamp-Massey algorithm is applied to the n-bit sequence $s_0 s_1 \ldots s_{n-1}$, and let $P_n(x)$ be as defined in Section 2.6.

If $s_1 = \delta$ and $s_{2i+1} = s_{2i} + s_i$ for $1 \leqslant i \leqslant \frac{n-2}{2}$

then $P_n^2(x) + P_n(x)C_n(x) + (1+x)[P_n(x)C_n(x)]' + \delta C_n^2(x) \equiv 0 \pmod{x^{n-1}}$

Proof

Consider the infinite binary sequence $s_0 s_1 s_2 \ldots$

and let $S(x) := s_0 + s_1 x + s_2 x^2 + \ldots$

$T(x) := s_0 + s_2 x + s_4 x^2 + \ldots$

and $U(x) := s_1 + s_3 x + s_5 x^2 + \ldots$

Then the coefficient of x^{2i} in

$U(x^2) + T(x^2) + S(x^2)$ is

$s_1 + s_0 + s_0 = s_1$ if $i = 0$

$s_{2i+1} + s_{2i} + s_i$ if $i \geqslant 1$

Thus, if $s_1 = \delta$ and $s_{2i+1} = s_{2i} + s_i$ for $1 \leqslant i \leqslant \frac{n-2}{2}$

then $U(x^2) + T(x^2) + S(x^2) \equiv \delta \pmod{x^{n-1}}$.

We now invoke Lemma 4.2.2, taking $\alpha = \beta = \gamma = 0$,

$A(x) = B(x) = C(x) = 1$ and $m = n-1$.

Hence, if $s_1 = \delta$ and $s_{2i+1} = s_{2i} + s_i$

for $1 \leqslant i \leqslant \frac{n-2}{2}$

then

$$P_n^2(x) + P_n(x)C_n(x) + (1+x)[P_n(x)C_n(x)]' + \delta C_n^2(x) \equiv 0 \pmod{x^{n-1}}$$

□

Having proved Lemma 4.4.1 we will now use it to show that if a binary sequence $s_0 s_1 \ldots s_{n-1}$ satisfies $s_{2i+1} = s_{2i} + s_i$ for $1 \leqslant i \leqslant \frac{n-2}{2}$ then the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ can only have jumps of

certain heights. We will treat the sequences with $s_1 = 0$ and $s_1 = 1$ separately. Firstly we consider the case $s_1 = 0$:-

Theorem 4.4.2.

Let $s_0 s_1 \ldots s_{n-1}$ be an n-bit sequence with $s_1 = 0$ and $s_{2i+1} = s_{2i} + s_i$ for $1 \leqslant i \leqslant \frac{n-2}{2}$.
Then the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ has no jumps of even height.

Proof

The n-bit sequence 00...0 satisfies the theorem trivially, since its linear complexity profile has no jumps.
For any other sequence $s_0 s_1 \ldots s_{n-1}$ satisfying the given equations the first jump in the profile must have odd height, since it must occur with bit s_{2i} (i.e. the $(2i+1)^{th}$ bit in the sequence) for some integer i, and thus it must have height 2i+1.

Suppose that two consecutive jumps in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ (the r^{th} and $(r+1)^{th}$, say) occur with the $a(r)^{th}$ and $a(r+1)^{th}$ bits in the sequence.
From Section 3.3 it can be seen that there exists an integer m with $a(r) < 2m < a(r+1)$ and $L(2m) = m$.

Let $a(r) = 2m-h$ and $a(r+1) = 2m+j$.

Then, as in the proof of Theorem 4.3.3, the r^{th} and $(r+1)^{th}$ jumps have heights h and j respectively, and the Berlekamp-Massey algorithm as described in Section 2.6 gives

$$C_{2m+j} = C_{2m} + x^{j+h}C_{2m-h-1}$$

and $$P_{2m+j} = P_{2m} + x^{j+h}P_{2m-h-1}$$

By Lemma 4.4.1,

since $s_1 = 0$ and $s_{2i+1} = s_{2i} + s_i$ for $1 \leqslant i \leqslant \frac{n-2}{2}$,

$$P_{2m+j}^2 + P_{2m+j}C_{2m+j} + (1+x)[P_{2m+j}C_{2m+j}]' \equiv 0 \pmod{x^{2m+j-1}}$$

$$\Rightarrow (P_{2m} + x^{j+h}P_{2m-h-1})^2 + (P_{2m} + x^{j+h}P_{2m-h-1})(C_{2m} + x^{j+h}C_{2m-h-1}) + (1+x)[(P_{2m} + x^{j+h}P_{2m-h-1})(C_{2m} + x^{j+h}C_{2m-h-1})]' \equiv 0 \pmod{x^{2m+j-1}}$$

$$\Rightarrow P_{2m}^2 + P_{2m}C_{2m} + (1+x)[P_{2m}C_{2m}]' + x^{2j+2h}(P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (1+x)[P_{2m-h-1}C_{2m-h-1}]') + x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m}) + (1+x)[x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m})]' \equiv 0 \pmod{x^{2m+j-1}} \quad (4.4.1)$$

But by Lemma 4.4.1,

$$P_{2m}^2 + P_{2m}C_{2m} + (1+x)[P_{2m}C_{2m}]' \equiv 0 \pmod{x^{2m-1}}$$

And by Theorem 2.6.1,

$\max (\deg C_{2m},\ 1 + \deg P_{2m}) = L(2m) = m$

so that $\deg (P_{2m}^2 + P_{2m}C_{2m} + (1+x)[P_{2m}C_{2m}]') \leqslant 2m-1$

Therefore $P_{2m}^2 + P_{2m}C_{2m} + (1+x)[P_{2m}C_{2m}]'$

$= 0$ or x^{2m-1} $(= \alpha x^{2m-1}$, say, where $\alpha = 0$ or $1)$

Also by Lemma 4.4.1,

$$P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (1+x)[P_{2m-h-1}C_{2m-h-1}]' \equiv 0 \pmod{x^{2m-h-2}}$$

And by Theorem 2.6.1,

$\max (\deg C_{2m-h-1},\ 1 + \deg P_{2m-h-1}) = L(2m-h-1) = m-h$

so that $\deg (P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (1+x)[P_{2m-h-1}C_{2m-h-1}]') \leqslant 2m-2h-1$

Therefore $P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (1+x)[P_{2m-h-1}C_{2m-h-1}]'$

$$= \begin{cases} 0 \text{ or } x^{2m-3} & \text{if } h = 1 \\ \qquad (= \beta x^{2m-3}, \text{ say, where } \beta = 0 \text{ or } 1) & \\ 0 & \text{if } h > 1 \end{cases}$$

By Lemma 4.2.1, $P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m} = x^{2m-h-1}$

Therefore equation (4.4.1) gives

$$\alpha x^{2m-1} + x^{j+h}x^{2m-h-1} + (1+x)[x^{j+h}x^{2m-h-1}]'$$

$$+ \begin{cases} x^{2j+2}\beta x^{2m-3} \equiv 0 \pmod{x^{2m+j-1}} & \text{if } h = 1 \\ 0 \equiv 0 \pmod{x^{2m+j-1}} & \text{if } h > 1 \end{cases}$$

$$\Rightarrow \quad \alpha x^{2m-1} + x^{2m+j-1} + (1+x).(j-1).x^{2m+j-2}$$

$$+ \begin{cases} \beta x^{2m+2j-1} \equiv 0 \pmod{x^{2m+j-1}} & \text{if } h = 1 \\ 0 \equiv 0 \pmod{x^{2m+j-1}} & \text{if } h > 1 \end{cases}$$

$$\Rightarrow \quad \alpha x^{2m-1} + (j-1).x^{2m+j-2} \equiv 0 \pmod{x^{2m+j-1}}$$

$\Rightarrow$ j is odd

□

Having dealt with sequences with $s_{2i+1} = s_{2i} + s_i$ and $s_1 = 0$, we now consider sequences with $s_{2i+1} = s_{2i} + s_i$ and $s_1 = 1$:-

Theorem 4.4.3.

Let $s_0 s_1 \ldots s_{n-1}$ be an n-bit sequence with $s_1 = 1$ and $s_{2i+1} = s_{2i} + s_i$ for $1 \leqslant i \leqslant \frac{n-2}{2}$.
Then the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ has no jumps of even height > 2.

Proof

Since $s_1 = 1$, the linear complexity profile of any sequence satisfying the given equations must have at least one jump.
For any sequence which satisfies the given equations, the first jump in the profile must occur either with bit s_0 (if $s_0 = 1$) or with bit s_1 (if $s_0 = 0$), and thus it

must have either height 1 (if $s_0 = 1$) or height 2 (if $s_0 = 0$). Hence the first jump in the profile does not contradict the theorem.

Suppose that two consecutive jumps in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ (the r^{th} and $(r+1)^{th}$, say) occur with the $a(r)^{th}$ and $a(r+1)^{th}$ bits in the sequence.
From Section 3.3 it can be seen that there exists an integer m with $a(r) < 2m < a(r+1)$ and $L(2m) = m$.
Let $a(r) = 2m-h$ and $a(r+1) = 2m+j$.
Then, as in the proof of Theorem 4.3.3, the r^{th} and $(r+1)^{th}$ jumps have heights h and j respectively, and the Berlekamp-Massey algorithm as described in Section 2.6 gives

$$C_{2m+j} = C_{2m} + x^{j+h} C_{2m-h-1}$$

and $$P_{2m+j} = P_{2m} + x^{j+h} P_{2m-h-1}$$

By Lemma 4.4.1,
since $s_1 = 1$ and $s_{2i+1} = s_{2i} + s_i$ for $1 \leqslant i \leqslant \frac{n-2}{2}$,

$$P_{2m+j}^2 + P_{2m+j} C_{2m+j} + (1+x)[P_{2m+j} C_{2m+j}]' + C_{2m+j}^2 \equiv 0 \pmod{x^{2m+j-1}}$$

$$\Rightarrow \quad (P_{2m} + x^{j+h} P_{2m-h-1})^2$$
$$+ (P_{2m} + x^{j+h} P_{2m-h-1})(C_{2m} + x^{j+h} C_{2m-h-1})$$
$$+ (1+x)[(P_{2m} + x^{j+h} P_{2m-h-1})(C_{2m} + x^{j+h} C_{2m-h-1})]'$$
$$+ (C_{2m} + x^{j+h} C_{2m-h-1})^2 \equiv 0 \pmod{x^{2m+j-1}}$$

$$\Rightarrow \quad P_{2m}^2 + P_{2m}C_{2m} + (1+x)[P_{2m}C_{2m}]' + C_{2m}^2$$
$$+ x^{2j+2h}(P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1}$$
$$+ (1+x)[P_{2m-h-1}C_{2m-h-1}]' + C_{2m-h-1}^2)$$
$$+ x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m})$$
$$+ (1+x)[x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m})]'$$
$$\equiv 0 \pmod{x^{2m+j-1}} \qquad (4.4.2)$$

But by Lemma 4.4.1,

$$P_{2m}^2 + P_{2m}C_{2m} + (1+x)[P_{2m}C_{2m}]' + C_{2m}^2 \equiv 0 \pmod{x^{2m-1}}$$

And by Theorem 2.6.1,

$$\max(\deg C_{2m},\ 1 + \deg P_{2m}) = L(2m) = m$$

so that $\deg (P_{2m}^2 + P_{2m}C_{2m} + (1+x)[P_{2m}C_{2m}]' + C_{2m}^2) \leqslant 2m$

Therefore $P_{2m}^2 + P_{2m}C_{2m} + (1+x)[P_{2m}C_{2m}]' + C_{2m}^2$

$= \alpha_1 x^{2m} + \alpha_2 x^{2m-1}$, where α_1 and $\alpha_2 = 0$ or 1

Also by Lemma 4.4.1,

$$P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (1+x)[P_{2m-h-1}C_{2m-h-1}]'$$
$$+ C_{2m-h-1}^2 \equiv 0 \pmod{x^{2m-h-2}}$$

And by Theorem 2.6.1,

$$\max(\deg C_{2m-h-1},\ 1 + P_{2m-h-1}) = L(2m-h-1) = m-h$$

so that $\deg (P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1}$

$$+ (1+x)[P_{2m-h-1}C_{2m-h-1}]' + C_{2m-h-1}^2) \leqslant 2m-2h$$

Therefore

$$P_{2m-h-1}^2 + P_{2m-h-1}C_{2m-h-1} + (1+x)[P_{2m-h-1}C_{2m-h-1}]'$$

$$+ C_{2m-h-1}^2 = \begin{cases} \beta_1 x^{2m-2} + \beta_2 x^{2m-3} & \text{if } h=1 \quad (\beta_1, \beta_2 = 0 \text{ or } 1) \\ \beta_1 x^{2m-4} & \text{if } h=2 \quad (\beta_1 = 0 \text{ or } 1) \\ 0 & \text{if } h>2 \end{cases}$$

By Lemma 4.2.1, $P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m} = x^{2m-h-1}$

Therefore equation (4.4.2) gives

$$\alpha_1 x^{2m} + \alpha_2 x^{2m-1} + x^{j+h}x^{2m-h-1} + (1+x)[x^{j+h}x^{2m-h-1}]'$$

$$+ \begin{cases} x^{2j+2}(\beta_1 x^{2m-2} + \beta_2 x^{2m-3}) \equiv 0 \pmod{x^{2m+j-1}} & \text{if } h=1 \\ x^{2j+4}\beta_1 x^{2m-4} \equiv 0 \pmod{x^{2m+j-1}} & \text{if } h=2 \\ 0 \equiv 0 \pmod{x^{2m+j-1}} & \text{if } h>2 \end{cases}$$

$$\Rightarrow \alpha_1 x^{2m} + \alpha_2 x^{2m-1} + (j-1)x^{2m+j-2} \equiv 0 \pmod{x^{2m+j-1}}$$

$\Rightarrow$ either j is odd

or $j = 2$

□

4.5. A MORE GENERAL THEORY

In Sections 4.3 and 4.4 we concerned ourselves, for each theorem, with sequences which satisfied a certain set of linear equations. In this section we will derive more general results in the sense that, for each theorem, we will consider a class of sets of linear equations rather than a specific set of equations. The previous two sections involved sequences in which every other bit was the sum of the preceding bit and a bit approximately "half way back". In the more general theory of this section we will be concerned with sequences in which, roughly speaking, every other bit is the sum of a number of the preceding few bits and a number of bits approximately "half way back". It will be shown that the linear complexity profile of a sequence of the appropriate type is restricted in the sense that its profile can have no jumps of a certain parity above a certain height.

We will deal with the sequences in two groups, according to whether their "fixed" bits (i.e. the ones which can be expressed as a sum of previous bits) are the ones with odd or even indices. We begin with the sequences whose fixed bits have odd indices :-

Lemma 4.5.1.

Let $C_n(x)$ be the connection polynomial produced when the Berlekamp-Massey algorithm is applied to the n-bit sequence $s_0 s_1 \ldots s_{n-1}$, and let $P_n(x)$ be as defined in Section 2.6.

If $s_{2i+1-2w}$

$$= s_{2i+1-2x(1)} + s_{2i+1-2x(2)} + \ldots + s_{2i+1-2x(a)}$$
$$+ s_{2i-2y(1)} + s_{2i-2y(2)} + \ldots + s_{2i-2y(b)}$$
$$+ s_{i-z(1)} + s_{i-z(2)} + \ldots + s_{i-z(c)}$$

for $\min(w,z(1)) \leqslant i \leqslant \min(\frac{n}{2}+w-1, n+z(1)-1)$

where $s_\ell := 0$ for $\ell < 0$

$(w < x(1) < x(2) < \ldots < x(a),$
$w \leqslant y(1) < y(2) < \ldots < y(b),\quad z(1) < z(2) < \ldots < z(c),$
$a \geqslant 0,\ b \geqslant 0,\ c > 0)$

then

$$(x^{2z(1)} + x^{2z(2)} + \ldots + x^{2z(c)}) P_n^2(x)$$
$$+ (x^{2y(1)} + x^{2y(2)} + \ldots + x^{2y(b)}) P_n(x) C_n(x)$$
$$+ (x^{2w} + x^{2x(1)} + \ldots + x^{2x(a)}$$
$$+ x^{2y(1)+1} + \ldots + x^{2y(b)+1}) [P_n(x) C_n]'$$
$$\equiv 0 \pmod{x^{\min(2w+n-1,\, 2z(1)+2n)}}$$

Proof

Consider the infinite binary sequence $s_0 s_1 s_2 \ldots$

and let $S(x) := s_0 + s_1 x + s_2 x^2 + \ldots$

$T(x) := s_0 + s_2 x + s_4 x^2 + \ldots$

and $U(x) := s_1 + s_3 x + s_5 x^2 + \ldots$

Then the coefficient of x^{2i} in

$$(x^{2w}+x^{2x(1)}+x^{2x(2)}+ \ldots +x^{2x(a)})U(x^2)$$
$$+ (x^{2y(1)}+x^{2y(2)}+ \ldots +x^{2y(b)})T(x^2)$$
$$+ (x^{2z(1)}+x^{2z(2)}+ \ldots +x^{2z(c)})S(x^2) \quad \text{is}$$

$$s_{2i+1-2w} + s_{2i+1-2x(1)} + s_{2i+1-2x(2)} + \cdots + s_{2i+1-2x(a)}$$
$$+ s_{2i-2y(1)} + s_{2i-2y(2)} + \cdots + s_{2i-2y(b)}$$
$$+ s_{i-z(1)} + s_{i-z(2)} + \cdots + s_{i-z(c)}$$

Therefore, if the conditions of the lemma hold then

$$(x^{2w}+x^{2x(1)}+x^{2x(2)}+ \ldots +x^{2x(a)})U(x^2)$$
$$+ (x^{2y(1)}+x^{2y(2)}+ \ldots +x^{2y(b)})T(x^2)$$
$$+ (x^{2z(1)}+x^{2z(2)}+ \ldots +x^{2z(c)})S(x^2)$$
$$\equiv 0 \pmod{x^{\min(2w+n-1,2z(1)+2n)}}$$

We now invoke Lemma 4.2.2, taking

$\alpha = 2y(1)$, $\beta = 2w$, $\gamma = 2z(1)$, $\delta = 0$,

$A(x) = 1 + x^{2y(2)-2y(1)} + \ldots + x^{2y(b)-2y(1)}$,

$B(x) = 1 + x^{2x(1)-2w} + \ldots + x^{2x(a)-2w}$,

$C(x) = 1 + x^{2z(2)-2z(1)} + \ldots + x^{2z(c)-2z(1)}$,

and $m = \min(2w+n-1,2z(1)+2n)$ ($\leqslant \min(\alpha+n,\beta+n-1,\gamma+2n)$).

Hence, if the conditions of the lemma hold

then

$$(x^{2z(1)}+x^{2z(2)}+ \ldots +x^{2z(c)})P_n^2(x)$$
$$+ (x^{2y(1)}+x^{2y(2)}+ \ldots +x^{2y(b)})P_n(x)C_n(x)$$
$$+ (x^{2w}+x^{2x(1)}+ \ldots +x^{2x(a)}$$
$$+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})[P_n(x)C_n(x)]'$$
$$\equiv 0 \pmod{x^{\min(2w+n-1,2z(1)+2n)}}$$

□

Theorem 4.5.2.

Let $s_0s_1\ldots s_{n-1}$ be an n-bit sequence with

$$s_{2i+1-2w} = s_{2i+1-2x(1)} + s_{2i+1-2x(2)} + \ldots + s_{2i+1-2x(a)}$$
$$+ s_{2i-2y(1)} + s_{2i-2y(2)} + \ldots + s_{2i-2y(b)}$$
$$+ s_{i-z(1)} + s_{i-z(2)} + \ldots + s_{i-z(c)}$$

for $\min(w,z(1)) \leqslant i \leqslant \min(\frac{n}{2}+w-1, n+z(1)-1)$,

where $s_\ell := 0$ for $\ell < 0$

$(w < x(1) < x(2) < \ldots < x(a)$,
$w \leqslant y(1) < y(2) < \ldots < y(b)$, $z(1) < z(2) < \ldots < z(c)$,
$a \geqslant 0$, $b \geqslant 0$, $c > 0)$.

Then the height j of any jump in the linear complexity profile of $s_0s_1\ldots s_{n-1}$ must satisfy either (i) or (ii) below :-

(i) j odd

(ii) $j \leqslant \max(2z(c)-2w, 2y(b)-2w+1, 2x(a)-2w)$

Proof

The n-bit sequence 00...0 satisfies the theorem trivially since its linear complexity profile has no jumps.

For any other sequence $s_0 s_1 \ldots s_{n-1}$ satisfying the given equations the first jump in the profile must have odd height, since it must occur with bit s_{2i} (i.e. the $(2i+1)^{th}$ bit in the sequence) for some integer i, and thus it must have height 2i+1.

Suppose that two consecutive jumps in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ (the r^{th} and $(r+1)^{th}$, say) occur with the $a(r)^{th}$ and $a(r+1)^{th}$ bits in the sequence.

From Section 3.3 it can be seen that there exists an integer m with $a(r) < 2m < a(r+1)$ and $L(2m) = m$.

Let $a(r) = 2m-h$ and $a(r+1) = 2m+j$.

Then, as in the proof of Theorem 4.3.3, the r^{th} and $(r+1)^{th}$ jumps have heights h and j respectively, and the Berlekamp-Massey algorithm as described in Section 2.6 gives

$$C_{2m+j} = C_{2m} + x^{j+h} C_{2m-h-1}$$

and $$P_{2m+j} = P_{2m} + x^{j+h} P_{2m-h-1}$$

By Lemma 4.5.1, if the conditions of the theorem hold then

$$(x^{2z(1)}+x^{2z(2)}+ \ldots +x^{2z(c)})P_{2m+j}^2$$
$$+ (x^{2y(1)}+x^{2y(2)}+ \ldots +x^{2y(b)})P_{2m+j}C_{2m+j}$$
$$+ (x^{2w}+x^{2x(1)}+ \ldots +x^{2x(a)}$$
$$+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})[P_{2m+j}C_{2m+j}]'$$
$$\equiv 0 \pmod{x^{\min(2z(1)+4m+2j,\,2w+2m+j-1)}}$$

$$\Rightarrow (x^{2z(1)}+ \ldots +x^{2z(c)})(P_{2m}+ x^{j+h}P_{2m-h-1})^2$$
$$+ (x^{2y(1)}+ \ldots +x^{2y(b)})$$
$$\cdot(P_{2m}+ x^{j+h}P_{2m-h-1})(C_{2m}+ x^{j+h}C_{2m-h-1})$$
$$+ (x^{2w}+x^{2x(1)}+ \ldots +x^{2x(a)}+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})$$
$$\cdot[(P_{2m}+ x^{j+h}P_{2m-h-1})(C_{2m}+ x^{j+h}C_{2m-h-1})]'$$
$$\equiv 0 \pmod{x^{\min(2z(1)+4m+2j,\,2w+2m+j-1)}}$$

$$\Rightarrow A(x) + B(x) + C(x) + D(x)$$
$$\equiv 0 \pmod{x^{\min(2z(1)+4m+2j,\,2w+2m+j-1)}} \qquad (4.5.1)$$

where

$$A(x) = (x^{2z(1)}+ \ldots +x^{2z(c)})P_{2m}^2$$
$$+ (x^{2y(1)}+ \ldots +x^{2y(b)})P_{2m}C_{2m}$$
$$+ (x^{2w}+x^{2x(1)}+ \ldots +x^{2x(a)}$$
$$+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})[P_{2m}C_{2m}]'$$

$$B(x) = x^{2j+2h}((x^{2z(1)}+ \ldots +x^{2z(c)})P_{2m-h-1}^2$$
$$+ (x^{2y(1)}+ \ldots +x^{2y(b)})P_{2m-h-1}C_{2m-h-1}$$
$$+ (x^{2w}+x^{2x(1)}+ \ldots +x^{2x(a)}+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})$$
$$\cdot[P_{2m-h-1}C_{2m-h-1}]')$$

$$C(x) = (x^{2y(1)} + \ldots + x^{2y(b)}) \cdot x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m})$$

and

$$D(x) = (x^{2w} + x^{2x(1)} + \ldots + x^{2x(a)} + x^{2y(1)+1} + \ldots + x^{2y(b)+1}) \cdot [x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m})]'$$

But by Lemma 4.5.1

$$A(x) \equiv 0 \pmod{x^{\min(2z(1)+4m,\,2w+2m-1)}}$$

And by Theorem 2.6.1,

$$\max(\deg C_{2m},\ 1 + \deg P_{2m}) = L(2m) = m$$

so that

$$\deg A(x) \leqslant \max(2z(c)+2m-2,\ 2y(b)+2m-1,\ 2x(a)+2m-2)$$

Therefore, $A(x) =$

$$\alpha_1 x^{\min(2z(1)+4m,\,2w+2m-1)} + \alpha_2 x^{\min(2z(1)+4m,\,2w+2m-1)+1} + \ldots + \alpha_p x^{\max(2z(c)+2m-2,\,2y(b)+2m-1,\,2x(a)+2m-2)}$$

for some $\alpha_1, \alpha_2, \ldots, \alpha_p = 0$ or 1. (4.5.2)

Also by Lemma 4.5.1,

$$B(x) / x^{2j+2h} \equiv 0 \pmod{x^{\min(2z(1)+4m-2h-2,\,2w+2m-h-2)}}$$

$$\Rightarrow B(x) \equiv 0 \pmod{x^{\min(2z(1)+4m+2j-2,\,2w+2m+2j+h-2)}}$$

$$\Rightarrow B(x) \equiv \beta_1 x^{\min(2z(1)+4m+2j-2,\,2w+2m+2j+h-2)} + \ldots + \beta_q x^{\min(2z(1)+4m+2j-1,\,2w+2m+j-2)} \pmod{x^{\min(2z(1)+4m+2j,\,2w+2m+j-1)}}$$

for some $\beta_1, \beta_2, \ldots, \beta_q = 0$ or 1. (4.5.3)

By Lemma 4.2.1, $P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m} = x^{2m-h-1}$

Hence $C(x) = (x^{2y(1)} + \ldots + x^{2y(b)})x^{j+h}x^{2m-h-1}$ (4.5.4)

and $D(x) = (x^{2w} + x^{2x(1)} + \ldots + x^{2x(a)}$
$+ x^{2y(1)+1} + \ldots + x^{2y(b)+1}).[x^{j+h}x^{2m-h-1}]$, (4.5.5)

Combining equations (4.5.1) to (4.5.5) we obtain

$$\alpha_1 x^{\min(2z(1)+4m,\,2w+2m-1)} + \ldots$$
$$\ldots + \alpha_p x^{\max(2z(c)+2m-2,\,2y(b)+2m-1,\,2x(a)+2m-2)}$$
$$+ \beta_1 x^{\min(2z(1)+4m+2j-2,\,2w+2m+2j+h-2)} + \ldots$$
$$\ldots + \beta_q x^{\min(2z(1)+4m+2j-1,\,2w+2m+j-2)}$$
$$+ (x^{2y(1)} + \ldots + x^{2y(b)})x^{2m+j-1}$$
$$+ (x^{2w} + x^{2x(1)} + \ldots + x^{2x(a)}$$
$$+ x^{2y(1)+1} + \ldots + x^{2y(b)+1}).(j-1).x^{2m+j-2}$$
$$\equiv 0 \pmod{x^{\min(2z(1)+4m+2j,\,2w+2m+j-1)}}$$

$$\Rightarrow \alpha_1 x^{\min(2z(1)+2m,\,2w-1)} + \ldots$$
$$\ldots + \alpha_p x^{\max(2z(c)-2,\,2y(b)-1,\,2x(a)-2)}$$
$$+ \beta_1 x^{\min(2z(1)+2m+2j-2,\,2w+2j+h-2)} + \ldots$$
$$\ldots + \beta_q x^{\min(2z(1)+2m+2j-1,\,2w+j-2)}$$
$$+ (x^{2y(1)} + \ldots + x^{2y(b)}).j.x^{j-1}$$
$$+ (x^{2w} + x^{2x(1)} + \ldots + x^{2x(a)}).(j-1).x^{j-2}$$
$$\equiv 0 \pmod{x^{\min(2z(1)+2m+2j,\,2w+j-1)}}$$

$$\Rightarrow \quad \alpha_1 x^{\min(2z(1)+2m,\,2w-1)} + \ldots$$

$$\ldots + \alpha_p x^{\max(2z(c)-2,\,2y(b)-1,\,2x(a)-2)}$$

$$+ \beta_1 x^{\min(2z(1)+2m+2j-2,\,2w+2j+h-2)} + \ldots$$

$$\ldots + \beta_q x^{\min(2z(1)+2m+2j-1,\,2w+j-2)}$$

$$+ (j-1).x^{2w+j-2} \equiv 0 \pmod{x^{\min(2z(1)+2m+2j,\,2w+j-1)}}$$

(since $w < x(1) < \ldots < x(a)$

and $w \leqslant y(1) < \ldots < y(b)$)

Thus, by considering the x^{2w+j-2} term it can be seen that j must satisfy either (i), (ii) or (iii) below :-

(i) j odd

(ii) $\min(2z(1)+2m,\,2w-1) \leqslant 2w+j-2$

$\leqslant \max(2z(c)-2,\,2y(b)-1,\,2x(a)-2)$

($\Rightarrow j \leqslant \max(2z(c)-2w,\,2y(b)-2w+1,\,2x(a)-2w)$)

(iii) $\min(2z(1)+2m+2j-2,\,2w+2j+h-2) \leqslant 2w+j-2$

$\leqslant \min(2z(1)+2m+2j-1,\,2w+j-2)$

($\Rightarrow 2m+j \leqslant 2w-2z(1)$, since $2w+2j+h-2 > 2w+j-2$)

But if (iii) is true, then $2m+j \leqslant 2(w-z(1))$

and so $w-z(1) > 0$ (i.e. $z(1) < w$).

Hence, if (iii) is true then :-

$2i+1-2w < 0$ for $\min(w,z(1)) = z(1) \leqslant i < w$;

$2i+1-2x(k) < 0$ $(k = 1,2,\ldots,a)$

for $\min(w,z(1)) = z(1) \leqslant i < w$

(since $w < x(1) < x(2) < \ldots < x(a)$);

$2i-2y(k) < 0$ $(k = 1,2,\ldots,b)$

for $\min(w,z(1)) = z(1) \leqslant i < w$

(since $w \leqslant y(1) < y(2) < \ldots < y(b)$),

and thus, since $s_\ell := 0$ for $\ell < 0$:-

$s_{2i+1-2w} = 0$ for $\min(w,z(1)) = z(1) \leqslant i < w$;

$s_{2i+1-2x(k)} = 0$ $(k = 1,2,\ldots,a)$ for $\min(w,z(1)) = z(1) \leqslant i < w$;

$s_{2i-2y(k)} = 0$ $(k = 1,2,\ldots,b)$ for $\min(w,z(1)) = z(1) \leqslant i < w$.

Therefore, from the conditions of the theorem it can be seen that, since $i-z(1) > i-z(2) > \ldots > i-z(c)$,

$s_{i-z(1)} = 0$ for $\min(w,z(1)) = z(1) \leqslant i < w$;

i.e. $s_i = 0$ for $0 \leqslant i < w-z(1)$.

Thus, the first jump in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ must have height at least $w-z(1)+1$, so that $m = L(2m) = L(a(r)) \geqslant L(a(1)) \geqslant w-z(1)+1$.

Hence $2m+j > 2m \geqslant 2w-2z(1)+2$, which contradicts "$2m+j \leqslant 2w-2z(1)$", and so j cannot satisfy (iii).

Therefore j must satisfy either (i) or (ii) below :-

(i) j odd

(ii) $j \leqslant \max(2z(c)-2w, 2y(b)-2w+1, 2x(a)-2w)$

□

We now deal with sequences whose fixed bits have even indices :-

Lemma 4.5.3.

Let $C_n(x)$ be the connection polynomial produced when the Berlekamp-Massey algorithm is applied to the n-bit sequence $s_0 s_1 \ldots s_{n-1}$, and let $P_n(x)$ be as defined in Section 2.6.

If s_{2i-2w}

$$= s_{2i+1-2x(1)} + s_{2i+1-2x(2)} + \ldots + s_{2i+1-2x(a)}$$
$$+ s_{2i-2y(1)} + s_{2i-2y(2)} + \ldots + s_{2i-2y(b)}$$
$$+ s_{i-z(1)} + s_{i-z(2)} + \ldots + s_{i-z(c)}$$

for $\min(w,z(1)) \leqslant i \leqslant \min(\frac{n-1}{2}+w, n+z(1)-1)$

where $s_\ell := 0$ for $\ell < 0$

$(w < x(1) < x(2) < \ldots < x(a)$,
$w < y(1) < y(2) < \ldots < y(b)$, $z(1) < z(2) < \ldots < z(c)$,
$a \geqslant 0$, $b \geqslant 0$, $c > 0)$

then $(x^{2z(1)}+x^{2z(2)}+ \ldots +x^{2z(c)})P_n^2(x)$

$+ (x^{2w}+x^{2y(1)}+x^{2y(2)}+ \ldots +x^{2y(b)})P_n(x)C_n(x)$

$+ (x^{2w+1}+x^{2x(1)}+ \ldots +x^{2x(a)}$
$+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})[P_n(x)C_n]'$

$\equiv 0 \pmod{x^{\min(2w+n,2z(1)+2n)}}$

Proof

Consider the infinite binary sequence $s_0 s_1 s_2 \ldots$

and let $S(x) := s_0 + s_1 x + s_2 x^2 + \ldots$

$T(x) := s_0 + s_2 x + s_4 x^2 + \ldots$

and $U(x) := s_1 + s_3 x + s_5 x^2 + \ldots$

Then the coefficient of x^{2i} in

$$(x^{2w}+x^{2y(1)}+x^{2y(2)}+ \ldots +x^{2y(b)})T(x^2)$$
$$+ (x^{2x(1)}+x^{2x(2)}+ \ldots +x^{2x(a)})U(x^2)$$
$$+ (x^{2z(1)}+x^{2z(2)}+ \ldots +x^{2z(c)})S(x^2)$$

is $s_{2i-2w} + s_{2i-2y(1)} + s_{2i-2y(2)} + \ldots + s_{2i-2y(b)}$
$+ s_{2i+1-2x(1)} + s_{2i+1-2x(2)} + \ldots + s_{2i+1-2x(a)}$
$+ s_{i-z(1)} + s_{i-z(2)} + \ldots + s_{i-z(c)}$

Therefore, if the conditions of the lemma hold

then

$$(x^{2w}+x^{2y(1)}+x^{2y(2)}+ \ldots +x^{2y(b)})T(x^2)$$
$$+ (x^{2x(1)}+x^{2x(2)}+ \ldots +x^{2x(a)})U(x^2)$$
$$+ (x^{2z(1)}+x^{2z(2)}+ \ldots +x^{2z(c)})S(x^2)$$
$$\equiv 0 \pmod{x^{\min(2w+n,2z(1)+2n)}}$$

We now invoke Lemma 4.2.2, taking

$\alpha = 2w$, $\beta = 2x(1)$, $\gamma = 2z(1)$, $\delta = 0$,

$A(x) = 1 + x^{2y(1)-2w} + \ldots + x^{2y(b)-2w}$,

$B(x) = 1 + x^{2x(2)-2x(1)} + \ldots + x^{2x(a)-2x(1)}$,

$C(x) = 1 + x^{2z(2)-2z(1)} + \ldots + x^{2z(c)-2z(1)}$,

and $m = \min(2w+n, 2z(1)+2n)$ $(\leqslant \min(\alpha+n, \beta+n-1, \gamma+2n))$.

Hence, if the conditions of the lemma hold

then

$$(x^{2z(1)}+x^{2z(2)}+ \ldots +x^{2z(c)})P_n^2(x)$$
$$+ (x^{2w}+x^{2y(1)}+x^{2y(2)}+ \ldots +x^{2y(b)})P_n(x)C_n(x)$$
$$+ (x^{2w+1}+x^{2x(1)}+\ldots+x^{2x(a)}$$
$$+x^{2y(1)+1}+\ldots+x^{2y(b)+1})[P_n(x)C_n(x)]'$$
$$\equiv 0 \pmod{x^{\min(2w+n,2z(1)+2n)}}$$

□

Theorem 4.5.4.

Let $s_0s_1\ldots s_{n-1}$ be an n-bit sequence with

$$s_{2i-2w} = s_{2i+1-2x(1)} + s_{2i+1-2x(2)} + \ldots + s_{2i+1-2x(a)}$$
$$+ s_{2i-2y(1)} + s_{2i-2y(2)} + \ldots + s_{2i-2y(b)}$$
$$+ s_{i-z(1)} + s_{i-z(2)} + \ldots + s_{i-z(c)}$$

for $\min(w,z(1)) \leqslant i \leqslant \min(\frac{n-1}{2}+w, n+z(1)-1)$,

where $s_\ell := 0$ for $\ell < 0$

$(w < x(1) < x(2) < \ldots < x(a)$,
$w < y(1) < y(2) < \ldots < y(b)$, $z(1) < z(2) < \ldots < z(c)$,
$a \geqslant 0$, $b \geqslant 0$, $c > 0)$.

Then the height j of any jump in the linear complexity profile of $s_0s_1\ldots s_{n-1}$ must satisfy either (i) or (ii) below :-

(i) j even

(ii) $j \leqslant \max(2z(c)-2w-1, 2y(b)-2w, 2x(a)-2w-1)$

Proof

The n-bit sequence 00...0 satisfies the theorem trivially, since its linear complexity profile has no jumps.

For any other sequence $s_0 s_1 \ldots s_{n-1}$ satisfying the given equations the first jump in the profile must have even height, since it must occur with bit s_{2i-1} (i.e. the $2i^{th}$ bit in the sequence) for some integer i, and thus it must have height 2i.

Suppose that two consecutive jumps in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ (the r^{th} and $(r+1)^{th}$, say) occur with the $a(r)^{th}$ and $a(r+1)^{th}$ bits in the sequence.

From Section 3.3 it can be seen that there exists an integer m with $a(r) < 2m < a(r+1)$ and $L(2m) = m$.

Let $a(r) = 2m-h$ and $a(r+1) = 2m+j$.

Then, as in the proof of Theorem 4.3.3, the r^{th} and $(r+1)^{th}$ jumps have heights h and j respectively, and the Berlekamp-Massey algorithm as described in Section 2.6 gives

$$C_{2m+j} = C_{2m} + x^{j+h} C_{2m-h-1}$$

and $$P_{2m+j} = P_{2m} + x^{j+h} P_{2m-h-1}$$

By Lemma 4.5.3, if the conditions of the theorem hold then

$$(x^{2z(1)}+x^{2z(2)}+ \ldots +x^{2z(c)})P_{2m+j}^2$$
$$+ (x^{2w}+x^{2y(1)}+x^{2y(2)}+ \ldots +x^{2y(b)})P_{2m+j}C_{2m+j}$$
$$+ (x^{2w+1}+x^{2x(1)}+ \ldots +x^{2x(a)}$$
$$+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})[P_{2m+j}C_{2m+j}]'$$
$$\equiv 0 \pmod{x^{\min(2z(1)+4m+2j,2w+2m+j)}}$$

$$\Rightarrow (x^{2z(1)}+ \ldots +x^{2z(c)})(P_{2m}+ x^{j+h}P_{2m-h-1})^2$$
$$+ (x^{2w}+x^{2y(1)}+ \ldots +x^{2y(b)})$$
$$\cdot(P_{2m}+ x^{j+h}P_{2m-h-1})(C_{2m}+ x^{j+h}C_{2m-h-1})$$
$$+ (x^{2w+1}+x^{2x(1)}+ \ldots +x^{2x(a)}+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})$$
$$\cdot[(P_{2m}+ x^{j+h}P_{2m-h-1})(C_{2m}+ x^{j+h}C_{2m-h-1})]'$$
$$\equiv 0 \pmod{x^{\min(2z(1)+4m+2j,2w+2m+j)}}$$

$$\Rightarrow A(x) + B(x) + C(x) + D(x)$$
$$\equiv 0 \pmod{x^{\min(2z(1)+4m+2j,2w+2m+j)}} \qquad (4.5.6)$$

where

$$A(x) = (x^{2z(1)}+ \ldots +x^{2z(c)})P_{2m}^2$$
$$+ (x^{2w}+x^{2y(1)}+ \ldots +x^{2y(b)})P_{2m}C_{2m}$$
$$+ (x^{2w+1}+x^{2x(1)}+ \ldots +x^{2x(a)}$$
$$+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})[P_{2m}C_{2m}]'$$

$$B(x) = x^{2j+2h}((x^{2z(1)}+ \ldots +x^{2z(c)})P_{2m-h-1}^2$$
$$+ (x^{2w}+x^{2y(1)}+ \ldots +x^{2y(b)})P_{2m-h-1}C_{2m-h-1}$$
$$+ (x^{2w+1}+x^{2x(1)}+ \ldots +x^{2x(a)}+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})$$
$$\cdot[P_{2m-h-1}C_{2m-h-1}]')$$

$$C(x) = (x^{2w}+x^{2y(1)}+ \ldots +x^{2y(b)}) \cdot x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m})$$

and

$$D(x) = (x^{2w+1}+x^{2x(1)}+ \ldots +x^{2x(a)} +x^{2y(1)+1}+ \ldots +x^{2y(b)+1}) \cdot [x^{j+h}(P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m})]'$$

But by Lemma 4.5.3

$$A(x) \equiv 0 \pmod{x^{\min(2z(1)+4m,2w+2m)}}$$

And by Theorem 2.6.1,

$\max(\deg C_{2m}, 1 + \deg P_{2m}) = L(2m) = m$

so that

$$\deg A(x) \leqslant \max(2z(c)+2m-2,2y(b)+2m-1,2x(a)+2m-2)$$

Therefore, $A(x) =$

$$\alpha_1 x^{\min(2z(1)+4m,2w+2m)} + \alpha_2 x^{\min(2z(1)+4m,2w+2m)+1} + \ldots + \alpha_p x^{\max(2z(c)+2m-2,2y(b)+2m-1,2x(a)+2m-2)}$$

for some $\alpha_1,\alpha_2,\ldots,\alpha_p = 0$ or 1. (4.5.7)

Also by Lemma 4.5.3,

$$B(x) / x^{2j+2h} \equiv 0 \pmod{x^{\min(2z(1)+4m-2h-2,2w+2m-h-1)}}$$

$$\Rightarrow B(x) \equiv 0 \pmod{x^{\min(2z(1)+4m+2j-2,2w+2m+2j+h-1)}}$$

$$\Rightarrow B(x) \equiv \beta_1 x^{\min(2z(1)+4m+2j-2,2w+2m+2j+h-1)} + \ldots + \beta_q x^{\min(2z(1)+4m+2j-1,2w+2m+j-1)} \pmod{x^{\min(2z(1)+4m+2j,2w+2m+j)}}$$

for some $\beta_1,\beta_2,\ldots,\beta_q = 0$ or 1. (4.5.8)

By Lemma 4.2.1, $P_{2m}C_{2m-h-1} + P_{2m-h-1}C_{2m} = x^{2m-h-1}$

Hence $C(x) = (x^{2w}+x^{2y(1)}+ \ldots +x^{2y(b)})x^{j+h}x^{2m-h-1}$ (4.5.9)

and $D(x) = (x^{2w+1}+x^{2x(1)}+ \ldots +x^{2x(a)}$
$+x^{2y(1)+1}+ \ldots +x^{2y(b)+1})$
$.[x^{j+h}x^{2m-h-1}]'$ (4.5.10)

Combining equations (4.5.6) to (4.5.10) we obtain

$$\alpha_1 x^{\min(2z(1)+4m,2w+2m)} + \ldots$$
$$\ldots + \alpha_p x^{\max(2z(c)+2m-2,2y(b)+2m-1,2x(a)+2m-2)}$$
$$+ \beta_1 x^{\min(2z(1)+4m+2j-2,2w+2m+2j+h-1)} + \ldots$$
$$\ldots + \beta_q x^{\min(2z(1)+4m+2j-1,2w+2m+j-1)}$$
$$+ (x^{2w}+x^{2y(1)}+ \ldots +x^{2y(b)})x^{2m+j-1}$$
$$+ (x^{2w+1}+x^{2x(1)}+\ldots+x^{2x(a)}+x^{2y(1)+1}+\ldots+x^{2y(b)+1})$$
$$.(j-1).x^{2m+j-2}$$
$$\equiv 0 \pmod{x^{\min(2z(1)+4m+2j,2w+2m+j)}}$$

$$\Rightarrow \alpha_1 x^{\min(2z(1)+2m,2w)} + \ldots$$
$$\ldots + \alpha_p x^{\max(2z(c)-2,2y(b)-1,2x(a)-2)}$$
$$+ \beta_1 x^{\min(2z(1)+2m+2j-2,2w+2j+h-1)} + \ldots$$
$$\ldots + \beta_q x^{\min(2z(1)+2m+2j-1,2w+j-1)}$$
$$+ (x^{2w}+x^{2y(1)}+ \ldots +x^{2y(b)}).j.x^{j-1}$$
$$+ (x^{2x(1)}+ \ldots +x^{2x(a)}).(j-1).x^{j-2}$$
$$\equiv 0 \pmod{x^{\min(2z(1)+2m+2j,2w+j)}}$$

$$\Rightarrow \quad \alpha_1 x^{\min(2z(1)+2m,\,2w)} + \ldots + \alpha_p x^{\max(2z(c)-2,\,2y(b)-1,\,2x(a)-2)}$$

$$+ \beta_1 x^{\min(2z(1)+2m+2j-2,\,2w+2j+h-1)} + \ldots + \beta_q x^{\min(2z(1)+2m+2j-1,\,2w+j-1)}$$

$$+ j.x^{2w+j-1} \equiv 0 \pmod{x^{\min(2z(1)+2m+2j,\,2w+j)}}$$

(since $w < x(1) < \ldots < x(a)$ and $w < y(1) < \ldots < y(b)$)

Thus, by considering the x^{2w+j-1} term it can be seen that j must satisfy either (i), (ii) or (iii) below :-

(i) j even

(ii) $\min(2z(1)+2m,2w) \leqslant 2w+j-1 \leqslant \max(2z(c)-2,2y(b)-1,2x(a)-2)$

($\Rightarrow j \leqslant \max(2z(c)-2w-1,2y(b)-2w,2x(a)-2w-1)$

(iii) $\min(2z(1)+2m+2j-2,2w+2j+h-1) \leqslant 2w+j-1 \leqslant \min(2z(1)+2m+2j-1,2w+j-1)$

($\Rightarrow 2m+j \leqslant 2w-2z(1)+1$, since $2w+2j+h-1 > 2w+j-1$)

But if (iii) is true, then $2m+j \leqslant 2(w-z(1))+1$ and so $w-z(1) > 0$ (i.e. $z(1) < w$).

Hence, if (iii) is true then :-

$2i-2w < 0$ for $\min(w,z(1)) = z(1) \leqslant i < w$;

$2i+1-2x(k) < 0$ $(k = 1,2,\ldots,a)$ for $\min(w,z(1)) = z(1) \leqslant i < w$ (since $w < x(1) < x(2) < \ldots < x(a)$);

$2i-2y(k) < 0$ $(k = 1,2,\ldots,b)$ for $\min(w,z(1)) = z(1) \leqslant i < w$ (since $w < y(1) < y(2) < \ldots < y(b)$),

and thus, since $s_\ell := 0$ for $\ell < 0$:-

$s_{2i-2w} = 0$ for $\min(w,z(1)) = z(1) \leqslant i < w$;

$s_{2i+1-2x(k)} = 0$ $(k = 1,2,\ldots,a)$ for $\min(w,z(1)) = z(1) \leqslant i < w$;

$s_{2i-2y(k)} = 0$ $(k = 1,2,\ldots,b)$ for $\min(w,z(1)) = z(1) \leqslant i < w$.

Therefore, from the conditions of the theorem it can be seen that, since $i-z(1) > i-z(2) > \ldots > i-z(c)$,

$s_{i-z(1)} = 0$ for $\min(w,z(1)) = z(1) \leqslant i < w$;

i.e. $s_i = 0$ for $0 \leqslant i < w-z(1)$.

Thus, the first jump in the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ must have height at least $w-z(1)+1$, so that $m = L(2m) = L(a(r)) \geqslant L(a(1)) \geqslant w-z(1)+1$.

Hence $2m+j > 2m \geqslant 2w-2z(1)+2$, which contradicts "$2m+j \leqslant 2w-2z(1)+1$", and so j cannot satisfy (iii).

Therefore j must satisfy either (i) or (ii) below :-

(i) j even

(ii) $j \leqslant \max(2z(c)-2w-1, 2y(b)-2w, 2x(a)-2w-1)$

□

Example 4.5.1.

As an example of the way in which the general theory of this section can be applied to a binary sequence which satisfies a particular set of linear equations, consider an n-bit sequence $s_0 s_1 \ldots s_{n-1}$ which satisfies the following conditions :-

$$s_1 = 0$$

$$s_{2i+1} = s_{2i-1} + s_i + s_{i-1} \quad \text{for} \quad 1 \leqslant i \leqslant \tfrac{n}{2}-1.$$

If we let $s_\ell := 0$ for $\ell < 0$, then the above conditions can be simplified to :-

$$s_{2i+1} = s_{2i-1} + s_i + s_{i-1} \quad \text{for} \quad 0 \leqslant i \leqslant \tfrac{n}{2}-1.$$

But if we take $a = 1$, $b = 0$, $c = 2$, $w = 0$, $x(1) = 1$, $z(1) = 0$ and $z(2) = 1$ in Theorem 4.5.2, then we see that the conditions of the theorem are exactly the conditions above.

Hence, by Theorem 4.5.2, if $s_1 = 0$ and $s_{2i+1} = s_{2i-1} + s_i + s_{i-1}$ for $1 \leqslant i \leqslant \frac{n}{2}-1$, then the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ has no jumps of even height greater than 2.

The results in this section go some way towards generalising the results obtained in Sections 4.3 and 4.4 of this chapter. The question remains as to whether further generalisations are possible. For instance, might it be possible to obtain a more general version of Theorem 4.3.4? Or are there any restrictions on the

linear complexity profiles of sequences which satisfy linear equations such as $s_{3i+1} = s_{3i} + s_i$, $s_{4i+2} = s_{4i} + s_{3i+1}$, etc. These and other such questions remain to be answered.

CHAPTER 5

STATISTICAL TESTS FOR RANDOMNESS

5.1. INTRODUCTION

In Section 1.2 the need to perform statistical testing on sections of enciphering sequence for use in stream cipher systems was established. For a sequence generator to be suitable for use in a stream cipher system, it is important that sections of output from the generator should be indistinguishable from randomly generated sequences of the same length. Thus, statistical tests for local randomness must be performed on sections of sequence generator output to check whether or not this is the case.

An important property of any such statistical test for randomness is that a certain percentage of random binary sequences should pass the test. If we denote this percentage by $(100-\alpha)\%$, then $\alpha\%$ is known as the significance level of the test and is the percentage of random binary sequences that would "fail" the test.

To derive a statistical test for local randomness, a test statistic which reflects some property of a sequence is found, with the property that the distribution of the test statistic is known for the set of all binary sequences. A critical region is then defined such that, for $(100-\alpha)\%$ of all binary sequences, the test statistic lies outside the critical region. In order to perform the test on a given sequence the test statistic is computed. The sequence is said to have

failed the test if the computed statistic lies inside the critical region; otherwise the sequence is said to have passed the test at the α% significance level.

It is important to note that, even if a sequence generator has the desired statistical properties (i.e. its output sequences are indistinguishable from randomly generated sequences), α% of all output sequences will fail any given statistical test. Hence, if a single output sequence happens to fail the test, then this does not necessarily imply a weakness in the generator. Thus it can be seen that the term "fail" is perhaps rather misleading when used in this context.

In this final chapter of the thesis we discuss the statistical testing of binary sequences. We begin by briefly reviewing some previously published statistical tests for local randomness. We then derive an expression for the number of n-bit sequences with a given number of jumps in their linear complexity profiles, and use this expression to compute the mean and variance of the number of jumps in the profile of a random n-bit sequence. Next we describe a statistical test for randomness based on the number of jumps in the linear complexity profile of a binary sequence, before moving on to discuss further tests based on the distribution of jump heights.

Note that the idea of using the linear complexity profile of a sequence to construct statistical tests for randomness was also suggested by Niederreiter in [13]. However, the ideas in this thesis were developed in parallel with and completely independently of the work of Niederreiter.

5.2. SOME WELL-KNOWN STATISTICAL TESTS

Before we go on to develop some new statistical tests for randomness, we first review some of the tests which have previously been published in the literature :-

5.2.1. Frequency Test

In a randomly generated n-bit sequence we would expect approximately half of the bits in the sequence (i.e. approximately n/2 bits) to be ones and approximately half to be zeros. The frequency test checks that the number of ones in the sequence is not significantly different from n/2, the expected value.

The frequency test is normally stated in the form of a goodness of fit test. In general, goodness of fit tests are used to test whether a set of observations conform to a particular distribution. In order to do this the observations are divided into a number (M, say) of classes, and for each class the expected number of observations in the class (e_k, say) is computed for $k = 1,2,\ldots,M$. Care must be taken to ensure that the expected frequency e_k for each class is at least a certain value; this minimum value is commonly taken to be 5.

The observed frequency (f_k, say) for each class is also computed, and the test statistic S is calculated, where

$$S := \sum_{k=1}^{M} (f_k - e_k)^2 / e_k.$$

The goodness of fit test is passed at the α% significance level if and only if $S \leqslant \chi^2$, where χ^2 is the upper α%-point of the χ^2 distribution with M-1 degrees of freedom (i.e. the point such that α% of the distribution lies above χ^2).

The frequency test tests the hypothesis that zeros and ones are equally likely to occur in the sequence being tested. Under this null hypothesis, the expected number of zeros and the expected number of ones will both be n/2. If we let the number of zeros and ones in the sequence be n_0 and n_1 respectively (so $n_0 + n_1 = n$) then the test statistic S is as follows :-

$$\begin{aligned} S &= (n_0 - n/2)^2 / (n/2) + (n_1 - n/2)^2 / (n/2) \\ &= (n_0 - n_1)^2 / n. \end{aligned}$$

The frequency test is passed at the α% significance level if and only if $S \leqslant \chi^2$, where χ^2 is the upper α%-point of the χ^2 distribution with 1 degree of freedom.

It could be argued that the frequency test should use a continuity correction (see [19]). However, this section is intended as a review of published tests; the

frequency test quoted in [2] and [9] does not use a continuity correction, and so we will omit to use one here.

5.2.2. Serial Test

The serial test checks that the frequencies of the different transitions in a binary sequence (i.e. 00, 01, 10 and 11) are approximately equal. By so doing the serial test gives an indication as to whether or not the bits in the sequence are independent of their predecessors.

Let n_{00}, n_{01}, n_{10} and n_{11} be the number of occurrences of 00, 01, 10 and 11 respectively in the n-bit sequence under test, and let n_0 and n_1 be the frequencies of 0 and 1 respectively. It can be shown (see [8]) that the statistic S is approximately distributed according to the χ^2 distribution with 2 degrees of freedom, where

$$S := \sum_{i=0}^{1} \sum_{j=0}^{1} \frac{(n_{ij} - (n-1)/4)^2}{(n-1)/4} - \sum_{i=0}^{1} \frac{(n_i - n/2)^2}{n/2}$$

Hence, the serial test is passed at the α% significance level if and only if $S \leqslant \chi^2$, where χ^2 is the upper α%-point of the χ^2 distribution with 2 degrees of freedom.

5.2.3. Poker Test

In a poker test the n-bit sequence being tested is divided into F non-overlapping m-bit blocks, where $F = \lfloor n/m \rfloor$, the greatest integer not exceeding n/m. If the sequence is random then these blocks are independent, and so the frequencies of the 2^m possible blocks should be approximately equal. Thus we can apply a goodness of fit test (see 5.2.1 above) to check whether the observed frequencies of the different types of block conform to the expected uniform distribution. We call such a goodness of fit test a poker test of block size m.

More precisely, let f_k be the number of occurrences of the k^{th} block ($k = 0,1,\ldots,2^m-1$) in the sequence. Then our test statistic is

$$S = \frac{2^m}{F} \cdot \sum_{k=0}^{2^m-1} (f_k - \frac{F}{2^m})^2$$

and the poker test is passed at the α% significance level if and only if $S \leqslant \chi^2$, where χ^2 is the upper α%-point of the χ^2 distribution with 2^m-1 degrees of freedom.

As mentioned in 5.2.1 above, care must be taken to ensure that the expected frequencies ($\frac{F}{2}m$ for all k in this case) are all large enough. If this is not the case then a variant of the poker test may be used, in which the different block types are grouped into classes in such a way that the expected frequency for each class attains the minimum value.

5.2.4. Runs Test

The runs test checks that the distribution of the lengths of runs of ones and zeros in a binary sequence conforms to that which would be expected for a randomly generated sequence. The theory underlying the runs test was established by Mood in [12]. In [12], two cases are considered; one in which the numbers of ones and zeros in the sequence are fixed, and the other in which they are randomly (i.e. binomially) distributed. We will not describe the runs test in detail here, as both the test and the theory behind it are rather complex.

The statistical tests for randomness described above represent only a selection from those described in the literature. Details of other such tests can be found in [2], [9] and [10].

5.3. THE NUMBER OF SEQUENCES WITH A GIVEN NUMBER OF JUMPS IN THEIR LINEAR COMPLEXITY PROFILES

In Chapter 3 the linear complexity profile of a binary sequence was discussed. In this section we will derive an expression for the number of n-bit binary sequences with a given number of jumps in their linear complexity profiles. Before we do this, however, we prove an intermediate result :-

Theorem 5.3.1.

Let $f(n,L,J)$ be the number of n-bit sequences with local linear complexity L and J jumps in their linear complexity profiles. Then

$$f(n,0,0) = 1$$

$$f(n,L,J) = \binom{\min(L-1,n-L)}{J-1} . 2^{\min(L,n-L)}$$

$$\text{if } 1 \leqslant L \leqslant n \text{ and } 1 \leqslant J \leqslant \min(L-1,n-L)+1$$

$$f(n,L,J) = 0 \quad \text{else}$$

Proof

The n-bit sequence 00...0 is the only n-bit sequence with local linear complexity $L(n) = 0$.

Therefore $f(n,0,0) = 1$

and $f(n,0,J) = 0$ for $J \geqslant 1$ (5.3.1)

since the linear complexity profile of 00...0 has no jumps.

For any n-bit sequence, the local linear complexity L(n) $\leqslant$ n, since any n-bit sequence can be generated using any n-stage LFSR.

Therefore $f(n,L,J) = 0$ if $L > n$. (5.3.2)

Also, if the local linear complexity of a sequence is positive then its linear complexity profile must have at least one jump,

and therefore $f(n,L,0) = 0$ for $L \geqslant 1$. (5.3.3)

For any n-bit sequence, the local linear complexity L(n) of the sequence must be no less than the number of jumps J in its linear complexity profile, since each jump must have height at least 1.

And $1 \leqslant L \leqslant \frac{n+1}{2} \Rightarrow \min(L-1,n-L)+1 = L$.

Therefore for any n-bit sequence with $1 \leqslant L(n) \leqslant \frac{n+1}{2}$, $J \leqslant L(n) = \min(L(n)-1,n-L(n)) + 1$.

Also, $\frac{n+1}{2} < L \leqslant n \Rightarrow \min(L-1,n-L)+1 = n-L+1$.

And if the last jump in the linear complexity profile of an n-bit sequence occurs with the m^{th} bit in the sequence then, from Section 2.3, $L(m) = m - L(m-1)$, so $L(m-1) = m - L(m) = m - L(n) \leqslant n - L(n)$, and hence the number of jumps J in the profile is at most $n-L(n)+1$.

Therefore for any n-bit sequence with $\frac{n+1}{2} < L(n) \leqslant n$, $J \leqslant n-L(n)+1 = \min(L(n)-1,n-L(n)) + 1$.

Thus

$$f(n,L,J) = 0 \qquad (5.3.4)$$

$$\text{if} \quad 1 \leqslant L \leqslant n \quad \text{and} \quad J > \min(L-1,n-L)+1$$

Combining equations (5.3.1) to (5.3.4), we have now shown that $f(n,L,J) = 0$ "else".

We now use induction to show that

$$f(n,L,J) = \binom{\min(L-1,n-L)}{J-1} . 2^{\min(L,n-L)} \qquad (5.3.5)$$

$$\text{if} \quad 1 \leqslant L \leqslant n \quad \text{and} \quad 1 \leqslant J \leqslant \min(L-1,n-L)+1$$

Consider an n-bit sequence $s_0 s_1 \ldots s_{n-1}$, and let J be the number of jumps in its linear complexity profile and L(k) be the local linear complexity of the first k bits of the sequence.

$n = 1 \Rightarrow$ either $L(n) = 1$ and $J = 1$

or $L(n) = 0$ and $J = 0$

Thus $f(1,1,1) = 1$, and so (5.3.5) holds for n = 1.

Now suppose that (5.3.5) holds for n = N

so that $f(N,L,J) = \binom{\min(L-1,N-L)}{J-1} . 2^{\min(L,N-L)}$

$$\text{if} \quad 1 \leqslant L \leqslant N \quad \text{and} \quad 1 \leqslant J \leqslant \min(L-1,N-L)+1$$

We will show that

$$f(N+1,L,J) = \binom{\min(L-1,N+1-L)}{J-1} . 2^{\min(L,N+1-L)}$$

$$\text{if } \quad 1 \leqslant L \leqslant N+1 \quad \text{and} \quad 1 \leqslant J \leqslant \min(L-1,N+1-L)+1$$

(i.e. that (5.3.5) holds for n = N+1).

We split the values of L = L(N+1) into 4 cases :-

(i) $1 \leqslant L \leqslant \frac{N}{2}$:

No jump could have occurred with the $(N+1)^{th}$ bit in the sequence (since $L(N+1) \leqslant \frac{N+1}{2}$), and half of all the N+1-bit sequences with L(N) = L will have local linear complexity L(N+1) = L (since $L \leqslant \frac{N}{2}$; the other half will have local linear complexity L(N+1) = N+1-L).

Therefore, since the number of N+1-bit sequences with L(N) = L is twice the number of N-bit sequences with L(N) = L,

$$f(N+1,L,J) = f(N,L,J)$$

$$= \binom{\min(L-1,N-L)}{J-1} . 2^{\min(L,N-L)}$$

$$= \binom{\min(L-1,N+1-L)}{J-1} . 2^{\min(L,N+1-L)}$$

(ii) $L = \frac{N+1}{2}$:

No jump could have occurred with the $(N+1)^{th}$ bit in the sequence (since $L(N+1) \leqslant \frac{N+1}{2}$), and all of the N+1-bit sequences with $L(N) = \frac{N+1}{2}$ will have local linear complexity $L(N+1) = \frac{N+1}{2}$ (since $L(N) > \frac{N}{2}$).

Therefore $f(N+1,L,J) = 2.f(N,L,J)$

$$= 2.\binom{\min((N-1)/2,(N-1)/2)}{J-1}.2^{\min((N+1)/2,(N-1)/2)}$$

$$= \binom{(N-1)/2}{J-1}.2^{(N+1)/2}$$

$$= \binom{\min(L-1,N+1-L)}{J-1}.2^{\min(L,N+1-L)}$$

(iii) $\frac{N+2}{2} \leqslant L \leqslant N$:

Either there was no jump with the $(N+1)^{th}$ bit in the sequence, or there was a jump from N+1-L to L (since $L(N+1) > \frac{N+1}{2}$). All of the N+1-bit sequences with $L(N) = L$ have local linear complexity $L(N+1) = L$ (since $L > \frac{N}{2}$), and half of those with $L(N) = N+1-L$ will have $L(N+1) = L$ (since $N+1-L \leqslant \frac{N}{2}$).

Therefore $f(N+1,L,J) = 2.f(N,L,J) + f(N,N+1-L,J-1)$

$$= 2.\binom{\min(L-1,N-L)}{J-1}.2^{\min(L,N-L)} + \binom{\min(N-L,L-1)}{J-2}.2^{\min(N+1-L,L-1)}$$

$$= \binom{N-L}{J-1}.2^{N-L+1} + \binom{N-L}{J-2}.2^{N+1-L}$$

$$= \binom{N-L+1}{J-1} . 2^{N+1-L} \qquad \left(\text{since } \binom{n+1}{r} = \binom{n}{r} + \binom{n}{r-1}\right)$$

$$= \binom{\min(L-1,N+1-L)}{J-1} . 2^{\min(L,N+1-L)}$$

(iv) $L = N+1$:

There must have been a jump from 0 to N+1 with the $(N+1)^{th}$ bit in the sequence (see Example 3.2.2). There is only one N+1-bit sequence whose linear complexity profile has such a jump, namely 00...01.

Therefore $f(N+1,L,J) = 1$ if $J = 1$ (and 0 if $J \neq 1$)

and $\binom{\min(L-1,N+1-L)}{J-1} . 2^{\min(L,N+1-L)} = 1$ for $J = 1$

so $f(N+1,L,J) = \binom{\min(L-1,N+1-L)}{J-1} . 2^{\min(L,N+1-L)}$

for $1 \leqslant J \leqslant \min(L-1,N+1-L)+1 = 1$

□

Now let N(n,J) be the number of n-bit sequences with J jumps in their linear complexity profiles. Notice that

$$N(n,J) = \sum_{L=0}^{n} f(n,L,J) \qquad (5.3.6)$$

We will derive an explicit expression for N(n,J) :-

Theorem 5.3.2.

Let N(n,J) be the number of n-bit sequences with J jumps in their linear complexity profiles. Then

$$N(n,J) = \begin{cases} 1 & \text{if } J = 0 \\ 3.\sum_{i=J-1}^{(n/2)-1} \binom{i}{J-1}.2^i & \text{if } 1 \leqslant J \leqslant \frac{n}{2} \text{ and } n \text{ even} \\ 3.\sum_{i=J-1}^{(n-3)/2} \binom{i}{J-1}.2^i + \binom{(n-1)/2}{J-1}.2^{(n-1)/2} & \text{if } 1 \leqslant J \leqslant \frac{n+1}{2} \text{ and } n \text{ odd} \\ 0 & \text{if } J > \frac{n+1}{2} \end{cases}$$

Proof

By equation (5.3.6), $N(n,J) = \sum_{L=0}^{n} f(n,L,J)$

$J = 0 \Rightarrow f(n,L,J) = 1$ if $L = 0$ (and 0 if $L \neq 0$)

(by Theorem 5.3.1)

$$\Rightarrow N(n,0) = \sum_{L=0}^{n} f(n,L,0) = 1.$$

$J > \frac{n+1}{2} \Rightarrow J > \min(L-1,n-L)+1$ for $1 \leqslant L \leqslant n$

$\Rightarrow f(n,L,J) = 0$ for $1 \leqslant L \leqslant n$

(by Theorem 5.3.1)

$$\Rightarrow N(n,J) = \sum_{L=0}^{n} f(n,L,J) = 0.$$

Now consider the case $1 \leqslant J \leqslant \frac{n+1}{2}$.

Notice that

$1 \leqslant L \leqslant n$, $1 \leqslant J \leqslant \min(L-1,n-L)+1$ if and only if

$$1 \leqslant J \leqslant L \leqslant n-J+1.$$

Thus $$N(n,J) = \sum_{L=0}^{n} f(n,L,J)$$

$$= \sum_{L=J}^{n-J+1} \binom{\min(L-1,n-L)}{J-1} . 2^{\min(L,n-L)} \qquad (5.3.7)$$

for $1 \leqslant J \leqslant \frac{n+1}{2}$ (by Theorem 5.3.1)

We now consider the odd and even values of n separately.

$1 \leqslant J \leqslant \frac{n}{2}$ and n even :

From equation (5.3.7)

$$N(n,J) = \sum_{L=J}^{(n/2)-1} \binom{L-1}{J-1} . 2^{L} + \binom{(n/2)-1}{J-1} . 2^{n/2} + \sum_{L=(n/2)+1}^{n-J} \binom{n-L}{J-1} . 2^{n-L} + \binom{J-1}{J-1} . 2^{J-1}$$

$$= \sum_{i=J}^{(n/2)-1} \left(\binom{i-1}{J-1} + \binom{i}{J-1} \right) . 2^{i} + \binom{(n/2)-1}{J-1} . 2^{n/2} + 2^{J-1}$$

$$= \sum_{i=J-1}^{(n/2)-1} \binom{i}{J-1} . 2^{i} + \sum_{i=J}^{n/2} \binom{i-1}{J-1} . 2^{i}$$

$$= 3. \sum_{i=J-1}^{(n/2)-1} \binom{i}{J-1} . 2^{i}$$

$1 \leqslant J \leqslant \frac{n+1}{2}$ and n odd :

From equation (5.3.7)

$$N(n,J) = \sum_{L=J}^{(n-1)/2} \binom{L-1}{J-1}.2^{L} + \sum_{L=(n+1)/2}^{n-J} \binom{n-L}{J-1}.2^{n-L} + \binom{J-1}{J-1}.2^{J-1}$$

$$= \sum_{i=J}^{(n-1)/2} \left(\binom{i-1}{J-1} + \binom{i}{J-1}\right).2^{i} + 2^{J-1}$$

$$= \sum_{i=J-1}^{(n-1)/2} \binom{i}{J-1}.2^{i} + \sum_{i=J}^{(n-1)/2} \binom{i-1}{J-1}.2^{i}$$

$$= 3. \sum_{i=J-1}^{(n-3)/2} \binom{i}{J-1}.2^{i} + \binom{(n-1)/2}{J-1}.2^{(n-1)/2}$$

□

5.4. THE MEAN AND VARIANCE OF THE NUMBER OF JUMPS

In this section we use Theorem 5.3.2 to obtain explicit expressions for both the mean and variance of the number of jumps in the linear complexity profile of a random n-bit binary sequence. We begin with the mean :-

Theorem 5.4.1.

Let J_n be the random variable representing the number of jumps in the linear complexity profile of a random n-bit binary sequence, and let $\mu_n := E(J_n)$ be the mean number of jumps in the linear complexity profile of such a sequence.

Then

$$\mu_n = \begin{cases} \frac{n}{4} + \frac{1}{3} - \frac{1}{3.2^n} & \text{if } n \text{ is even} \\ \frac{n}{4} + \frac{5}{12} - \frac{1}{3.2^n} & \text{if } n \text{ is odd} \end{cases}$$

Proof

$$\mu_n = \sum_{j=0}^{n} j.\ P(J_n = j)$$

But $P(J_n = j)$ = (number of n-bit sequences with j jumps in their profiles) / (total number of n-bit sequences)

$$= N(n,j) / 2^n$$

Therefore $\mu_n = \sum_{j=0}^{n} j.\,(N(n,j) / 2^n)$

$$= \sum_{j=1}^{(n+1)/2} j.\,(N(n,j) / 2^n) \qquad (5.4.1)$$

since, by Theorem 5.3.2, $N(n,j) = 0$ for $j > \frac{n+1}{2}$.

We consider odd and even values of n separately :-

n even :

By equation (5.4.1),

$$\mu_n = \tfrac{1}{2}n . \sum_{j=1}^{n/2} j.N(n,j)$$

$$= \tfrac{3}{2}n . \sum_{j=1}^{n/2} j . \sum_{i=j-1}^{(n/2)-1} \binom{i}{j-1}.2^i \qquad \text{(by Theorem 5.3.2)}$$

$$= \tfrac{3}{2}n . \sum_{i=0}^{(n/2)-1} 2^i . \sum_{j=1}^{i+1} \binom{i}{j-1}.j$$

$\left(\text{since } 1 \leqslant j \leqslant \frac{n}{2},\ j-1 \leqslant i \leqslant \frac{n}{2}-1 \text{ if and only if } 0 \leqslant i \leqslant \frac{n}{2}-1,\ 1 \leqslant j \leqslant i+1\right)$

$$= \tfrac{3}{2}n . \sum_{i=0}^{(n/2)-1} 2^i . \sum_{j=0}^{i} \binom{i}{j}.(j+1)$$

$$= \tfrac{3}{2}n . \sum_{i=0}^{(n/2)-1} 2^i . (i.2^{i-1} + 2^i)$$

$\left(\text{since } \sum_{k=0}^{r} \binom{r}{k}.k = r.2^{r-1} \text{ and } \sum_{k=0}^{r} \binom{r}{k} = 2^r\right)$

$$= \frac{3}{2}n.\left(\frac{1}{2}.\sum_{i=0}^{(n/2)-1} i.4^i + \sum_{i=0}^{(n/2)-1} 4^i \right)$$

$$= \frac{3}{2}n.\left(\frac{1}{2}.\left[\frac{n-2}{2}.4^{(n/2)+1} - \frac{n}{2}.4^{n/2} + 4\right] / 9 + \left[4^{n/2} - 1\right] / 3\right)$$

$$\left[\text{since } \sum_{i=0}^{c} i.x^i = (c.x^{c+2} - (c+1).x^{c+1} + x) / (x-1)^2 \right.$$
$$\left. \text{and } \sum_{i=0}^{c} x^i = (x^{c+1} - 1) / (x-1)\right]$$

$$= \frac{3}{2}n.\left(\frac{1}{36}.\left[(n-2).2^{n+2} - n.2^n + 8\right] + \frac{1}{3}.\left[2^n - 1\right] \right)$$

$$= \frac{1}{12.2}n.\left[n.2^{n+2} - 2^{n+3} - n.2^n + 8 + 12.2^n - 12\right]$$

$$= \frac{n}{4} + \frac{1}{3} - \frac{1}{3.2}n$$

n odd :

By equation (5.4.1),

$$\mu_n = \frac{1}{2}n.\sum_{j=1}^{(n+1)/2} j.N(n,j)$$

$$= \frac{3}{2}n.\sum_{j=1}^{(n+1)/2} j.\sum_{i=j-1}^{(n-3)/2} \binom{i}{j-1}.2^i + \frac{1}{2}n.(2^{(n-1)/2}).\sum_{j=1}^{(n+1)/2} \binom{(n-1)/2}{j-1}.j$$

(by Theorem 5.3.2)

$$= \frac{3}{2}n \cdot \sum_{i=0}^{(n-3)/2} 2^i \cdot \sum_{j=1}^{i+1} \binom{i}{j-1} \cdot j \quad + \quad (\tfrac{1}{2})^{(n+1)/2} \cdot \sum_{j=1}^{(n+1)/2} \binom{(n-1)/2}{j-1} \cdot j$$

$$\left(\text{since } 1 \leqslant j \leqslant \frac{n+1}{2},\ j-1 \leqslant i \leqslant \frac{n-3}{2} \text{ if and only if } 0 \leqslant i \leqslant \frac{n-3}{2},\ 1 \leqslant j \leqslant i+1\right)$$

$$= \frac{3}{2}n \cdot \sum_{i=0}^{(n-3)/2} 2^i \cdot \sum_{j=0}^{i} \binom{i}{j} \cdot (j+1) \quad + \quad (\tfrac{1}{2})^{(n+1)/2} \cdot \sum_{j=0}^{(n-1)/2} \binom{(n-1)/2}{j} \cdot (j+1)$$

$$= \frac{3}{2}n \cdot \sum_{i=0}^{(n-3)/2} 2^i \cdot (i.2^{i-1} + 2^i) \quad + \quad (\tfrac{1}{2})^{(n+1)/2} \cdot \left(\frac{n-1}{2} \cdot (2^{(n-3)/2}) + 2^{(n-1)/2}\right)$$

$$\left(\text{since } \sum_{k=0}^{r} \binom{r}{k} \cdot k = r.2^{r-1} \text{ and } \sum_{k=0}^{r} \binom{r}{k} = 2^r\right)$$

$$= \frac{3}{2}n \cdot \left(\frac{1}{2} \cdot \sum_{i=0}^{(n-3)/2} i.4^i + \sum_{i=0}^{(n-3)/2} 4^i \right) + \frac{n-1}{8} + \frac{1}{2}$$

$$= \frac{3}{2}n \cdot \left(\frac{1}{2} \cdot \left(\frac{n-3}{2} \cdot 4^{(n+1)/2} - \frac{n-1}{2} \cdot 4^{(n-1)/2} + 4\right) / 9 \right.$$

$$\left. + \left(4^{(n-1)/2} - 1\right) / 3 \right) + \frac{n-1}{8} + \frac{1}{2}$$

$$\left(\text{since } \sum_{i=0}^{c} i.x^i = (c.x^{c+2} - (c+1).x^{c+1} + x) / (x-1)^2 \text{ and } \sum_{i=0}^{c} x^i = (x^{c+1} - 1) / (x-1)\right)$$

$$= \frac{3}{2}n.\left(\frac{1}{36}.\left[(n-3).2^{n+1} - (n-1).2^{n-1} + 8\right] + \frac{1}{3}.\left[2^{n-1}-1\right] \right)$$

$$+ \frac{n-1}{8} + \frac{1}{2}$$

$$= \frac{1}{12.2}n.\left[n.2^{n+1} - 3.2^{n+1} - n.2^{n-1} + 2^{n-1} + 8 + 6.2^{n} - 12\right]$$

$$+ \frac{n-1}{8} + \frac{1}{2}$$

$$= \frac{n}{4} + \frac{5}{12} - \frac{1}{3.2}n$$

□

We now compute the variance of the number of jumps in the linear complexity profile of a random n-bit sequence :-

Theorem 5.4.2.

Let σ_n^2 be the variance of the number of jumps in the linear complexity profile of a random n-bit binary sequence.

Then

$$\sigma_n^2 = \begin{cases} \frac{n}{8} - \frac{2}{9} + \frac{n}{6.2}n + \frac{1}{3.2}n - \frac{1}{9.2}2n & \text{if n is even} \\ \frac{n}{8} - \frac{1}{8} + \frac{n}{6.2}n + \frac{7}{18.2}n - \frac{1}{9.2}2n & \text{if n is odd} \end{cases}$$

Proof

Firstly we note that

$$\sigma_n^2 = E(J_n^2) - E(J_n)^2$$

$$= E(J_n^2) - \mu_n^2 \qquad (5.4.2)$$

and $E(J_n^2) = \sum_{j=0}^{n} j^2 . P(J_n = j)$

$$= \sum_{j=0}^{n} j^2 . (N(n,j)/2^n)$$

$$= \sum_{j=1}^{(n+1)/2} j^2 . (N(n,j)/2^n) \qquad (5.4.3)$$

since, by Theorem 5.3.2, $N(n,j) = 0$ for $j > \frac{n+1}{2}$.

As in Theorem 5.4.1, we consider odd and even values of n separately :-

n even :

By equation (5.4.3)

$$E(J_n^2) = \tfrac{1}{2}n . \sum_{j=1}^{n/2} j^2 . N(n,j)$$

$$= \tfrac{3}{2}n . \sum_{j=1}^{n/2} j^2 . \sum_{i=j-1}^{(n/2)-1} \binom{i}{j-1} . 2^i \qquad \text{(by Theorem 5.3.2)}$$

$$= \tfrac{3}{2}n . \sum_{i=0}^{(n/2)-1} 2^i . \sum_{j=1}^{i+1} \binom{i}{j-1} . j^2$$

$$= \frac{3}{2}n.\sum_{i=0}^{(n/2)-1} 2^i.\sum_{j=0}^{i}\binom{i}{j}.(j+1)^2$$

$$= \frac{3}{2}n.\sum_{i=0}^{(n/2)-1} 2^i.(i.(i+1).2^{i-2} + i.2^i + 2^i)$$

$$\left(\text{since } \sum_{k=0}^{r}\binom{r}{k}.k^2 = r.(r+1).2^{r-2},\ \sum_{k=0}^{r}\binom{r}{k}.k = r.2^{r-1}\right.$$

$$\left.\text{and } \sum_{k=0}^{r}\binom{r}{k} = 2^r\right)$$

$$= \frac{3}{2}n.\left(\frac{1}{4}.\sum_{i=0}^{(n/2)-1} i^2.4^i + \frac{1}{4}.\sum_{i=0}^{(n/2)-1} i.4^i + \sum_{i=0}^{(n/2)-1} i.4^i + \sum_{i=0}^{(n/2)-1} 4^i\right)$$

$$= \frac{3}{2}n.\left(\left(\left(\frac{n-2}{2}\right)^2.4^{(n+2)/2} - \left(2.\left(\frac{n-2}{2}\right)^2 + 2.\left(\frac{n-2}{2}\right) - 1\right).4^{n/2}\right.\right.$$

$$\left. + \left(\frac{n}{2}\right)^2.4^{(n-2)/2} - 5\right) / 27$$

$$+ \frac{5}{4}.\left(\frac{n-2}{2}.4^{(n+2)/2} - \frac{n}{2}.4^{n/2} + 4\right) / 9$$

$$\left. + \left(4^{n/2} - 1\right) / 3\right)$$

$$\left(\text{since } \sum_{i=0}^{c} i^2x^i = x.(c^2x^{c+2} - (2c^2+2c-1)x^{c+1}\right.$$

$$\left. + (c+1)^2x^c - x - 1) / (x-1)^3,\right.$$

$$\sum_{i=0}^{c} i.x^i = (c.x^{c+2} - (c+1).x^{c+1} + x) / (x-1)^2$$

$$\left.\text{and } \sum_{i=0}^{c} x^i = (x^{c+1} - 1) / (x-1)\right)$$

$$= \frac{3}{2}n.\left(\frac{1}{108}.(n^2-4n+4).2^{n+2} - \frac{1}{54}.(n^2-2n-2).2^n \right.$$

$$+ \frac{1}{108}.n^2.2^{n-2} - \frac{5}{27} + \frac{5}{72}.(n-2).2^{n+2} - \frac{5}{72}.n.2^n$$

$$\left. + \frac{5}{9} + \frac{1}{3}.(2^n-1) \right)$$

$$= \frac{1}{9}.(n^2-4n+4) - \frac{1}{18}.(n^2-2n-2) + \frac{1}{144}.n^2 - \frac{5}{9.2}n + \frac{5}{6}.(n-2)$$

$$- \frac{5}{24}.n + \frac{5}{3.2}n + 1 - \frac{1}{2}n$$

$$= \frac{n^2}{16} + \frac{7n}{24} - \frac{1}{9} + \frac{1}{9.2}n \qquad (5.4.4)$$

Therefore, combining equations (5.4.2) and (5.4.4) with the result of Theorem 5.4.1, we get

$$\sigma_n^2 = \frac{n^2}{16} + \frac{7n}{24} - \frac{1}{9} + \frac{1}{9.2}n - \left(\frac{n}{4} + \frac{1}{3} - \frac{1}{3.2}n \right)^2$$

$$= \frac{n}{8} - \frac{2}{9} + \frac{n}{6.2}n + \frac{1}{3.2}n - \frac{1}{9.2}2n$$

n odd :

By equation (5.4.3)

$$E(J^2) = \frac{1}{2}n. \sum_{j=1}^{(n+1)/2} j^2.N(n,j)$$

$$= \frac{3}{2}n. \sum_{j=1}^{(n+1)/2} j^2. \sum_{i=j-1}^{(n-3)/2} \binom{i}{j-1}.2^i$$

$$+ \frac{1}{2}n.(2^{(n-1)/2}). \sum_{j=1}^{(n+1)/2} \binom{(n-1)/2}{j-1}.j^2$$

(by Theorem 5.3.2)

$$= \frac{3}{2}n \cdot \sum_{i=0}^{(n-3)/2} 2^i \cdot \sum_{j=1}^{i+1} \binom{i}{j-1} \cdot j^2 \quad + \quad (\tfrac{1}{2})^{(n+1)/2} \cdot \sum_{j=1}^{(n+1)/2} \binom{(n-1)/2}{j-1} \cdot j^2$$

$$= \frac{3}{2}n \cdot \sum_{i=0}^{(n-3)/2} 2^i \cdot \sum_{j=0}^{i} \binom{i}{j} \cdot (j+1)^2$$

$$+ \quad (\tfrac{1}{2})^{(n+1)/2} \cdot \sum_{j=0}^{(n-1)/2} \binom{(n-1)/2}{j} \cdot (j+1)^2$$

$$= \frac{3}{2}n \cdot \sum_{i=0}^{(n-3)/2} 2^i \cdot (i \cdot (i+1) \cdot 2^{i-2} + i \cdot 2^i + 2^i)$$

$$+ (\tfrac{1}{2})^{(n+1)/2} \cdot \left(\frac{n-1}{2} \cdot \frac{n+1}{2} \cdot 2^{(n-5)/2} + \frac{n-1}{2} \cdot 2^{(n-1)/2} + 2^{(n-1)/2} \right)$$

$$\left(\text{since } \sum_{k=0}^{r} \binom{r}{k} \cdot k^2 = r \cdot (r+1) \cdot 2^{r-2}, \quad \sum_{k=0}^{r} \binom{r}{k} \cdot k = r \cdot 2^{r-1} \right.$$

$$\left. \text{and } \sum_{k=0}^{r} \binom{r}{k} = 2^r \right)$$

$$= \frac{3}{2}n \cdot \left(\frac{1}{4} \cdot \sum_{i=0}^{(n-3)/2} i^2 \cdot 4^i \quad + \quad \frac{1}{4} \cdot \sum_{i=0}^{(n-3)/2} i \cdot 4^i \quad + \quad \sum_{i=0}^{(n-3)/2} i \cdot 4^i \quad + \quad \sum_{i=0}^{(n-3)/2} 4^i \right)$$

$$+ \frac{(n-1)(n+1)}{32} + \frac{n-1}{4} + \frac{1}{2}$$

$$= \frac{3}{2}n.\left(\left((\frac{n-3}{2})^2.4^{(n+1)/2} - (2.(\frac{n-3}{2})^2 \right.\right.$$

$$\left.\left. + 2.(\frac{n-3}{2}) - 1).4^{(n-1)/2} + (\frac{n-1}{2})^2.4^{(n-3)/2} - 5 \right) / 27 \right.$$

$$+ \frac{5}{4}.\left(\frac{n-3}{2}.4^{(n+1)/2} - \frac{n-1}{2}.4^{(n-1)/2} + 4 \right) / 9$$

$$\left. + \left(4^{(n-1)/2} - 1 \right) / 3 \right) + \frac{(n-1)(n+1)}{32} + \frac{n-1}{4} + \frac{1}{2}$$

$$\left(\text{since } \sum_{i=0}^{c} i^2x^i = x.(c^2x^{c+2} - (2c^2+2c-1)x^{c+1} \right.$$

$$\left. + (c+1)^2x^c - x - 1) / (x-1)^3, \right.$$

$$\sum_{i=0}^{c} i.x^i = (c.x^{c+2} - (c+1).x^{c+1} + x) / (x-1)^2$$

$$\left. \text{and } \sum_{i=0}^{c} x^i = (x^{c+1} - 1) / (x-1) \right)$$

$$= \frac{3}{2}n.\left(\frac{1}{108}.(n^2-6n+9).2^{n+1} - \frac{1}{54}.(n^2-4n+1).2^{n-1} \right.$$

$$+ \frac{1}{108}.(n^2-2n+1).2^{n-3} - \frac{5}{27} + \frac{5}{72}.(n-3).2^{n+1}$$

$$\left. - \frac{5}{72}.(n-1).2^{n-1} + \frac{5}{9} + \frac{1}{3}.(2^{n-1}-1) \right)$$

$$+ \frac{(n-1)(n+1)}{32} + \frac{n-1}{4} + \frac{1}{2}$$

$$= \frac{1}{18}.(n^2-6n+9) - \frac{1}{36}.(n^2-4n+1) + \frac{1}{288}.(n^2-2n+1) - \frac{5}{9.2}n$$

$$+ \frac{5}{12}.(n-3) - \frac{5}{48}.(n-1) + \frac{5}{3.2}n + \frac{1}{2} - \frac{1}{2}n + \frac{1}{32}.(n^2+8n+7)$$

$$= \frac{n^2}{16} + \frac{n}{3} + \frac{7}{144} + \frac{1}{9.2}n \qquad (5.4.5)$$

Therefore, combining equations (5.4.2) and (5.4.5) with the result of Theorem 5.4.1, we get

$$\sigma_n^2 = \frac{n^2}{16} + \frac{n}{3} + \frac{7}{144} + \frac{1}{9.2}n - \left(\frac{n}{4} + \frac{5}{12} - \frac{1}{3.2}n \right)^2$$

$$= \frac{n}{8} - \frac{1}{8} + \frac{n}{6.2}n + \frac{7}{18.2}n - \frac{1}{9.2}2n$$

□

5.5. A STATISTICAL TEST BASED ON THE NUMBER OF JUMPS

We now go on to describe a statistical test for randomness which checks the number of jumps in the linear complexity profile of a binary sequence. This test relies on the fact that the distribution of the number of jumps is approximately normal for large n, and so we begin by justifying this claim.

We first recall a result from Chapter 3 :-

Let M(n,L) be the set of all n-bit binary sequences with local linear complexity L, and denote the cardinality of this set by m(n,L). Thus, m(n,L) is the number of n-bit sequences with local linear complexity L. Then Theorem 3.4.5 can be rephrased to give the following result :-

$$m(n,L) = \begin{cases} 1 & \text{if } L = 0 \\ 2^{\min(2L-1,2n-2L)} & \text{if } 1 \leqslant L \leqslant n \\ 0 & \text{if } L > n \end{cases}$$

Consider the set M(n,L) of n-bit binary sequences with local linear complexity L. Let $J_{n,L}$ be the random variable representing the number of jumps in the linear complexity profile of a sequence chosen at random from

M(n,L). Then, using Theorems 3.4.5 and 5.3.1 it can be seen that, if $1 \leqslant L \leqslant n$, $J_{n,L}$ has the following distribution :-

$$P(J_{n,L} = j)$$

$$= \begin{cases} 0 & \text{if } j = 0 \\ \binom{\min(L-1,n-L)}{j-1} . 2^{\min(L,n-L)} \,/\, 2^{\min(2L-1,2n-2L)} & \text{if } 1 \leqslant j \leqslant \min(L-1,n-L)+1 \\ 0 & \text{if } j > \min(L-1,n-L)+1 \end{cases}$$

(since $P(J_{n,L} = j) = f(n,L,j) \,/\, m(n,L)$)

$$\Rightarrow P(J_{n,L} - 1 = j) = \begin{cases} \binom{\min(L-1,n-L)}{j} . (\frac{1}{2})^{\min(L-1,n-L)} & \text{if } 0 \leqslant j \leqslant \min(L-1,n-L) \\ 0 & \text{else} \end{cases}$$

and so $J_{n,L} - 1$ is distributed according to the binomial distribution $B(\min(L-1,n-L),\frac{1}{2})$.
(A random variable X is distributed according to the binomial distribution $B(r,p)$ if
$P(X = k) = \binom{r}{k} . p^k (1-p)^{r-k}$ for $k = 0,1,\ldots,r$)

Note also that for large values of r the binomial distribution $B(r,\frac{1}{2})$ is approximately $N(\frac{r}{2},\frac{r}{4})$, where $N(\mu,\sigma^2)$ is the normal distribution with mean μ and variance σ^2 (see, for example, [6]).

Hence. for large values of $\min(L-1,n-L)$, $J_{n,L}-1$ is approximately $N(\min(L-1,n-L)/2,\min(L-1,n-L)/4)$.

Now consider the set $M(n,*)$ (of cardinality 2^n) of all n-bit binary sequences. By Theorem 3.4.5, if a sequence is chosen at random from $M(n,*)$ then, if $1 \leqslant L \leqslant n$,

$$P(\text{sequence has local linear complexity } L) = 2^{\min(2L-1,2n-2L)} / 2^n$$

Thus, if we let J_n denote the random variable representing the number of jumps in the linear complexity profile of a sequence chosen at random from $M(n,*)$, then the distribution of the random variable J_n-1 is given by

$$P(J_n-1 = j) = P(J_n = j+1)$$

$$= \sum_{L=1}^{n} P(\text{sequence has } j+1 \text{ jumps} \mid \text{sequence has local linear complexity } L) \cdot P(\text{sequence has local linear comlexity } L)$$

$$= \sum_{L=1}^{n} (2^{\min(2L-1,2n-2L)} / 2^n) \cdot P(J_{n,L} = j+1)$$

$$= \sum_{L=1}^{n} (2^{\min(2L-1,2n-2L)} / 2^n) \cdot P(J_{n,L}-1 = j)$$

$$= \sum_{L=1}^{n} (2^{\min(2L-1,2n-2L)} / 2^n) \cdot b(\min(L-1,n-L),\tfrac{1}{2})$$

where $b(r,p) = \binom{r}{j} . p^j (1-p)^{r-j}$ is the probability that, if X is a random variable with the binomial distribution B(r,p), then X = j.

Therefore, for n even,

$$P(J_n - 1 = j)$$

$$= (\tfrac{1}{2})^{n-1}.b(0,\tfrac{1}{2}) + (\tfrac{1}{2})^{n-3}.b(1,\tfrac{1}{2}) + \ \ldots$$

$$+ \tfrac{1}{32}.b(\tfrac{n-6}{2},\tfrac{1}{2}) + \tfrac{1}{8}.b(\tfrac{n-4}{2},\tfrac{1}{2}) + \tfrac{1}{2}.b(\tfrac{n-2}{2},\tfrac{1}{2})$$

$$+ \tfrac{1}{4}.b(\tfrac{n-2}{2},\tfrac{1}{2}) + \tfrac{1}{16}.b(\tfrac{n-4}{2},\tfrac{1}{2}) + \tfrac{1}{64}.b(\tfrac{n-6}{2},\tfrac{1}{2}) + \ \ldots$$

$$+ (\tfrac{1}{2})^{n-2}.b(1,\tfrac{1}{2}) + (\tfrac{1}{2})^{n}.b(0,\tfrac{1}{2})$$

$$= \tfrac{3}{4}.b(\tfrac{n-2}{2},\tfrac{1}{2}) + \tfrac{3}{16}.b(\tfrac{n-4}{2},\tfrac{1}{2}) + \tfrac{3}{64}.b(\tfrac{n-6}{2},\tfrac{1}{2}) + \ \ldots$$

$$+ \tfrac{3}{2}n\text{-}2 . b(1,\tfrac{1}{2}) + \tfrac{3}{2}n . b(0,\tfrac{1}{2})$$

But for large values of n the terms towards the right hand end of the above expression are insignificant, while the binomial distributions towards the left hand end are approximately normal
(since $B(r,\tfrac{1}{2}) \simeq N(\tfrac{r}{2},\tfrac{r}{4})$ for large r).

Thus, for large n,

$$P(J_n - 1 = j) \simeq \frac{3}{4}.\phi(\frac{n-2}{4}, \frac{n-2}{8}) + \frac{3}{16}.\phi(\frac{n-4}{4}, \frac{n-4}{8}) + \frac{3}{64}.\phi(\frac{n-6}{4}, \frac{n-6}{8}) + \ldots$$

$$\Rightarrow P(J_n = j) \simeq \frac{3}{4}.\phi(\frac{n+2}{4}, \frac{n-2}{8}) + \frac{3}{16}.\phi(\frac{n}{4}, \frac{n-4}{8}) + \frac{3}{64}.\phi(\frac{n-2}{4}, \frac{n-6}{8}) + \ldots$$

where $\phi(\mu,\sigma^2)$ is the probability density function of the normal distribution with mean μ and variance σ^2, evaluated at j.

By Theorems 5.4.1 and 5.4.2 respectively, for n even

$$\mu_n = E(J_n) = \frac{n}{4} + \frac{1}{3} - \frac{1}{3.2^n}$$

$$\sigma_n^2 = Var(J_n) = \frac{n}{8} - \frac{2}{9} + \frac{n}{6.2^n} + \frac{1}{3.2^n} - \frac{1}{9.2^{2n}}$$

Therefore, if we denote these values by μ and σ^2 respectively, we have

$$P((J_n-\mu)/\sigma = j) \simeq \frac{3}{4}.\phi((\frac{n+2}{4}-\mu)/\sigma, (\frac{n-2}{8})/\sigma^2) + \frac{3}{16}.\phi((\frac{n}{4}-\mu)/\sigma, (\frac{n-4}{8})/\sigma^2) + \frac{3}{64}.\phi((\frac{n-2}{4}-\mu)/\sigma, (\frac{n-6}{8})/\sigma^2) + \ldots$$

$$\simeq \frac{3}{4}.\phi(\sqrt{(\frac{2}{9n-16})}, \frac{9n-18}{9n-16}) + \frac{3}{16}.\phi(-2.\sqrt{(\frac{2}{9n-16})}, \frac{9n-36}{9n-16}) + \frac{3}{64}.\phi(-5.\sqrt{(\frac{2}{9n-16})}, \frac{9n-54}{9n-16}) + \ldots \quad \text{for large n}$$

Thus, for large n, $P((J_n-\mu)/\sigma = j) \simeq \phi(0,1)$

The same result can also be established for odd values of n in a similar fashion.

Thus we have established that, for large n, the random variable $(J_n-\mu)/\sigma$ is approximately distributed according to the standard normal distribution N(0,1). This fact can be used in the construction of a statistical test for randomness (Test 5.5.1 below), which checks that the number of jumps in the linear complexity profile of an n-bit sequence is not significantly different from the expected value. Hence, a sequence will fail Test 5.5.1 if its linear complexity profile has too many jumps (e.g. if it has the perfect linear complexity profile), or if its profile has too few jumps (e.g. if too many of its jumps are excessively large). The test proceeds as follows :-

Test 5.5.1.

Step 1. Compute the number of jumps J in the linear complexity profile of the sequence using, for example, the Berlekamp-Massey algorithm.

Step 2. Compute the mean value μ of J_n using Theorem 5.4.1.

Step 3. Compute the variance σ^2 of J_n using Theorem 5.4.2.

Step 4. Compute the test statistic $S = (J-\mu)/\sigma$.

Step 5. The test is passed at the $\alpha\%$ significance level if and only if $-C \leqslant S \leqslant C$, where C is the upper $\frac{\alpha}{2}\%$-point of the standard normal distribution N(0,1).

Alternatively, Steps 4 and 5 above can be replaced by Step 4′ below :-

Step 4′. The test is passed at the $\alpha\%$ significance level if and only if $\mu-C\sigma \leqslant J \leqslant \mu+C\sigma$, where C is the upper $\frac{\alpha}{2}\%$-point of the standard normal distribution N(0,1).

To illustrate the use of Test 5.5.1, 100 8000-bit sequences were generated. For each sequence $s_0s_1\cdots s_{7999}$, the 4000 odd-indexed bits $s_1,s_3,\ldots,s_{7999}$ were randomly generated, while the 4000 even-indexed bits $s_0,s_2,\ldots,s_{7998}$ were generated using the equations $s_0 = 1$ and $s_{2i} = s_{2i-1} + s_{i-1}$ for $i = 1,2,\ldots,3999$. By Theorem 4.3.2, these sequences all have the perfect linear complexity profile, and therefore the profile of each sequence contains $J = 4000$ jumps. Hence, all 100 sequences clearly failed Test 5.5.1; for example, at the 5% significance level the test would be passed if and only if $1939 \leqslant J \leqslant 2062$.

The above-mentioned 100 sequences were also subjected to the statistical tests described in Section 5.2, and the following results were obtained :-

	Number of sequences passing at 5% significance level
Frequency test	97
Serial test	95
Runs test	99
Poker test	
block size = 2	97
3	94
4	96
5	95
6	98
7	94
8	98

Note that, if the sequences being tested had been randomly generated then, for any given test, the expected number of passes would have been 95, and there would have been a probability of approximately 0.972 that 91 or more of the sequences passed the test.

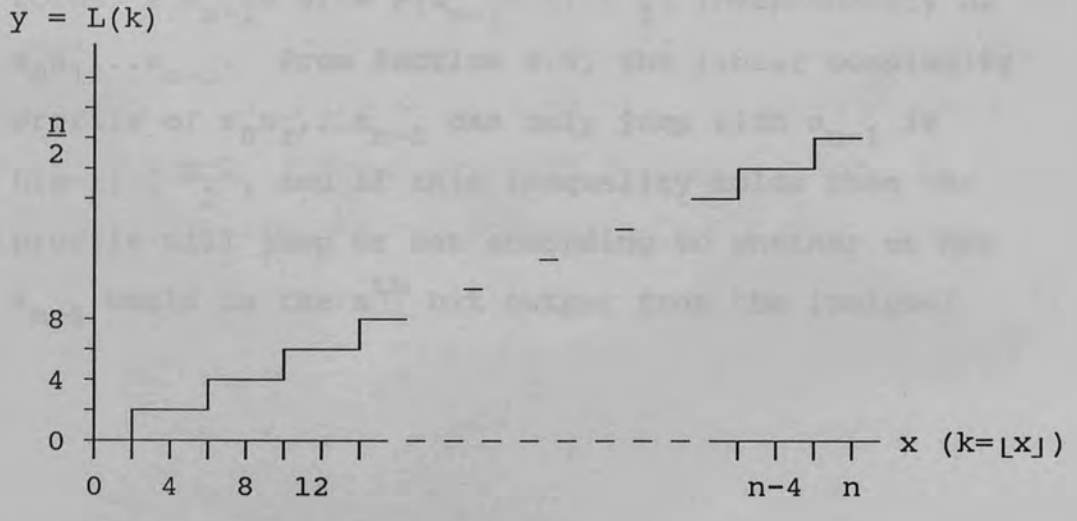
The above results suggest that Test 5.5.1 is capable of identifying in binary sequences non-randomness which would not be detected using established statistical tests for randomness.

5.6. STATISTICAL TESTS BASED ON THE DISTRIBUTION OF JUMP HEIGHTS

If a binary sequence passes Test 5.5.1 above then we know that the number of jumps in its linear complexity profile is not significantly different from the expected number of jumps in the profile of a randomly generated sequence of the same length. However, this does not necessarily imply that the linear complexity profile of this sequence is not substantially different from that which might be expected of a randomly generated sequence. The following example illustrates this point :-

Example 5.6.1.

Consider an n-bit binary sequence which has linear complexity profile $(L(1),L(2),\ldots,L(n)) = (0,2,2,2,2,4,4,4,4,6,\ldots,\frac{n}{2}-2,\frac{n}{2},\frac{n}{2},\frac{n}{2})$, as shown below. For convenience we assume that n is a multiple of 4.



This profile has exactly $\frac{n}{4}$ jumps, which is approximately the expected number for a random n-bit sequence (see Theorem 5.4.1), and hence the sequence would certainly pass Test 5.5.1. However, all $\frac{n}{4}$ of these jumps have height 2, and so the distribution of jump heights in this profile would be unlikely to be obtained from a randomly generated sequence.

In this section we will discuss statistical tests for randomness which check that the distribution of the heights of the jumps in the linear complexity profile of a binary sequence is not significantly different from the expected distribution of jump heights for a randomly generated sequence. We begin by discussing the distribution of jump heights :-

In the case of a random binary sequence $s_0 s_1 \ldots s_{n-1}$, the m^{th} bit in the sequence s_{m-1} is equally likely to be a 0 or 1, whatever the value of $s_0 s_1 \ldots s_{m-2}$. In other words, $P(s_{m-1} = 0) = P(s_{m-1} = 1) = \frac{1}{2}$, independently of $s_0 s_1 \ldots s_{m-2}$. From Section 3.3, the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ can only jump with s_{m-1} if $L(m-1) \leqslant \frac{m-1}{2}$, and if this inequality holds then the profile will jump or not according to whether or not s_{m-1} would be the m^{th} bit output from the (unique)

L(m-1)-stage LFSR on which $s_0 s_1 \cdots s_{m-2}$ can be generated if this register was loaded with the initial state $s_0 s_1 \cdots s_{L(m-1)-1}$.

Now suppose that the r^{th} jump in the linear complexity profile of a binary sequence occurs with the a^{th} bit in the sequence. Then $L(a) > \frac{a}{2}$, and so a jump cannot occur with the $(a+1)^{th}$ bit in the sequence. In fact, the next opportunity for a jump to occur will be with the $(2L(a)+1)^{th}$ bit in the sequence, since $L(i) = L(a) > \frac{i}{2}$ for $a \leqslant i \leqslant 2L(a)-1$.

From the above two paragraphs it can be seen that the linear complexity profile of a random binary sequence will jump at the first opportunity after the r^{th} jump (i.e. with the $(2L(a)+1)^{th}$ bit in the sequence) with probability $\frac{1}{2}$. Similarly, if the profile does not jump at the first opportunity then it will jump at the second opportunity (i.e. with the $(2L(a)+2)^{th}$ bit in the sequence) with probability $\frac{1}{2}$, and in general if the profile does not jump at any of the first k-1 opportunities after the r^{th} jump then it will jump at the k^{th} opportunity (i.e. with the $(2L(a)+k)^{th}$ bit in the sequence) with probability $\frac{1}{2}$. Thus, since the bits in the sequence are independent of each other, the

probability that the $(r+1)^{th}$ jump in the profile occurs at the k^{th} opportunity after the r^{th} jump (i.e. with the $(2L(a)+k)^{th}$ bit in the sequence) is $(\frac{1}{2})^k$.

If the r^{th} jump in the linear complexity profile of a binary sequence occurs with the a^{th} bit in the sequence and the $(r+1)^{th}$ jump occurs with the $(2L(a)+k)^{th}$ bit, then the $(r+1)^{th}$ jump in the profile will have height $L(2L(a)+k) - L(a)$. But $L(2L(a)+k) = 2L(a)+k - L(a)$ $= L(a)+k$ (from Section 2.3), and so the $(r+1)^{th}$ jump has height $L(a)+k - L(a) = k$ in this case. Thus, the probability that the $(r+1)^{th}$ jump has height k is $(\frac{1}{2})^k$.

Similarly, the probability that the first jump in the linear complexity profile of a random binary sequence has height k is $(\frac{1}{2})^k$, since the first jump in the linear complexity profile of $s_0s_1...s_{n-1}$ has height k if and only if $s_0s_1...s_{k-1} = 00...01$.

For a random binary sequence, the height of each jump is independent of the height of all other jumps. Thus it can be seen from the above discussion that the heights of the jumps in the linear complexity profile of a random binary sequence are independent, identically distributed random variables with distribution $G(\frac{1}{2})$, where G(p) is the geometric distribution with parameter p.

(A random variable X is distributed according to the geometric distribution G(p) if $P(X = k) = (1-p)^{k-1}.p$ for $k = 1,2,3,...$)

The above result can be used to construct a randomness test (Test 5.6.1 below) based on the heights of the jumps in the linear complexity profile of a binary sequence. This test considers the first F jumps in the profile, where F is fixed for a given sequence length n, and checks whether the heights of these jumps conform to the geometric distribution with parameter $\frac{1}{2}$. Hence, the test is capable of identifying non-randomness such as that displayed in Example 5.6.1.

Test 5.6.1 is a goodness of fit test (see Section 5.2). In the test, the jumps are divided into M classes according to their heights. Each of the first M-1 classes contains all the jumps of height k for some k ($1 \leqslant k \leqslant M-1$), while the M^{th} class contains all the jumps of height $\geqslant M$, where M is the lowest integer such that the expected number of jumps of height M is less than 5. The expected frequency e_k for each class is $F.p_k$, where p_k is the probability of a given jump in the linear complexity profile of a random binary sequence being in class k. Hence, since the jump heights for a random binary sequence are distributed according to the geometric distribution $G(\frac{1}{2})$, $p_k = (\frac{1}{2})^k$ for $k = 1,2,...,M-1$ and $p_M = (\frac{1}{2})^M + (\frac{1}{2})^{M+1} + ... = (\frac{1}{2})^{M-1}$.

Given a binary sequence $s_0 s_1 \ldots s_{n-1}$ the test proceeds as follows :-

Test 5.6.1.

Step 1. Compute the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ using, for example, the Berlekamp-Massey algorithm.

Step 2. Compute $F = \lceil \mu - C\sigma \rceil$, the least integer greater than or equal to $\mu - C\sigma$, where C, μ and σ are as in Test 5.5.1. If a sequence passes Test 5.5.1 then we know that its linear complexity profile must contain at least F jumps.

Step 3. Define M to be the smallest value of k such that $F.(\frac{1}{2})^k < 5$.

For k = 1,2,...,M compute f_k, where

f_k := number of jumps of height k in first F jumps of profile (k = 1,2,...,M-1)

f_M := number of jumps of height $\geqslant$ M in first F jumps of profile.

Step 4. Compute the test statistic

$$S := \sum_{k=1}^{M} [(f_k - F.p_k)^2 / F.p_k]$$

where $$p_k := \begin{cases} (\frac{1}{2})^k & (k = 1,2,\ldots,M-1) \\ (\frac{1}{2})^{M-1} & (k = M) \end{cases}$$

Step 5. The test is passed at the α% significance level if and only if $S \leqslant \chi^2$, where χ^2 is the upper α%-point for the χ^2 distribution with M-1 degrees of freedom.

If a single n-bit sequence $s_0 s_1 \ldots s_{n-1}$ is being tested for randomness then, strictly speaking, Test 5.6.1 can only be performed if the linear complexity profile of $s_0 s_1 \ldots s_{n-1}$ contains at least F jumps. However, if the profile has less than F jumps then the sequence will already have failed Test 5.5.1 if that test has been performed. Also, if it is a sequence generator that is being tested rather than a single sequence, then it might be possible to generate subsequent output bits $s_n s_{n+1} s_{n+2} \ldots$ until the required F jumps have been obtained.

An alternative approach to testing the distribution of jump heights is to apply the theory established by Mood in [12], as was used in the runs test (see Section 5.2) :-

Consider the n-bit sequence $s_0 s_1 \ldots s_{n-1}$ with linear complexity profile $(L(1), L(2), \ldots, L(n))$, and assume that the profile contains R jumps and that the r^{th} jump occurs with the $a(r)^{th}$ bit in the sequence ($r = 1,2,\ldots,R$). Think of the first jump in this profile (of

height L(a(1))) as a run of ones of length L(a(1)). Similarly, think of the second jump in the profile (of height L(a(2)) - L(a(2)-1)) as a run of zeros of length L(a(2)) - L(a(2)-1), the third jump (of height L(a(3)) - L(a(3)-1)) as a run of ones of length L(a(3)) - L(a(3)-1), etc. In general, think of the r^{th} jump in the profile as a run of ones or zeros (according to whether r is odd or even) of length L(a(r)) - L(a(r)-1). By considering the jumps in the linear complexity profile as runs of ones and zeros in this way, the entire profile can be thought of as a binary sequence $t_0t_1\ldots t_{L(n)-1}$ of length L(n) which contains $\lceil R/2 \rceil$ runs of ones and $\lfloor R/2 \rfloor$ runs of zeros.

If $s_0s_1\ldots s_{n-1}$ is a random sequence then, by a similar argument to that given in the preamble to Test 5.6.1, it can be seen that $t_0t_1\ldots t_{L(n)-1}$ can also be thought of as a random sequence of bits (with the possible exception of the last run, which must have length at least $2.(L(n)-\frac{n}{2})$ if $L(n) > \frac{n}{2}$).

Hence, the distribution of the heights of jumps in the linear complexity profile of $s_0s_1\ldots s_{n-1}$ can be tested by applying the theory in [12] to the L(n)-bit binary sequence $t_0t_1\ldots t_{L(n)-1}$. In particular, Corollary 5 of Section 5 of [12] and Corollary 4 of Section 8 of [12] would seem to be particularly relevant, as these results

group together runs of ones and zeros of a given length. As in Section 5.2, however, we will not give details of the tests here, as these details are rather complex.

To illustrate the use of Test 5.6.1, 100 8000-bit sequences were again generated. This time, for each sequence $s_0 s_1 \ldots s_{7999}$ the 4000 odd-indexed bits were randomly generated, while the even-indexed bits were generated so that the equations $s_0 = 0$ and $s_{2i+2} = s_{2i} + s_i$ for $i = 0,1,\ldots,3998$ were all satisfied. By Theorem 4.5.4, the linear complexity profiles of these sequences have no jumps of odd height greater than 1 (i.e. they have no jumps of heights 3,5,7,...). Hence, if we perform Test 5.6.1 on any of these sequences, then the contribution to the test statistic S from the k=3 term is $(f_3 - F.p_3)^2 / F.p_3 \simeq 242.4$ (since f_3 = number of jumps of height 3 = 0, F = 1939 from Section 5.5, and $p_3 = (\frac{1}{2})^3 = \frac{1}{8}$), and this alone is sufficient to ensure that the test is failed at the 5% significance level, since M = 9 and the upper 5%-point of the χ^2 distribution with 8 degrees of freedom is only 15.51.

The above-mentioned 100 sequences were also subjected to the statistical tests described in Section 5.2, and to Test 5.5.1. The following results were obtained :-

	Number of sequences passing at 5% significance level
Frequency test	97
Serial test	98
Runs test	97
Poker test	
block size = 2	95
3	93
4	97
5	94
6	94
7	96
8	94
Test 5.5.1	92

These results suggest that Test 5.6.1 is also capable of detecting non-randomness in sequences which would remain undetected if the sequences were tested using established statistical tests. More generally, the statistical tests based on linear complexity profiles described in this chapter would be capable of detecting

the non-randomness in most of the sequences discussed in Chapter 4, and indeed in other sequences in which non-random structure in the sequences was reflected in their linear complexity profiles.

REFERENCES

[1] **Aho, A.V., Hopcroft, J.E. and Ullman, J.D.,** *'The Design and Analysis of Cumputer Algorithms'*, Addison-Wesley, Reading, Mass., (1974).

[2] **Beker, H.J. and Piper, F.C.,** *'Cipher Systems: The Protection of Communications'*, Van Nostrand Reinhold, London, (1982).

[3] **Berlekamp, E.R.,** *'Algebraic Coding Theory'*, McGraw-Hill, New York, (1968).

[4] **Cheng, U.,** *'Properties of Sequences'*, Ph.D. Thesis, University of Southern California, (1981).

[5] **Dai, Z.D.,** 'Proof of Rueppel's Linear Complexity Conjecture'. To appear.

[6] **Feller, W.,** *'An Introduction to Probability Theory and its Applications'*, Volume 1, 3rd edition, Wiley, New York, (1968).

[7] **Golomb, S.W.,** *'Shift Register Sequences'*, revised edition, Aegean Park Press, Laguna Hills, Cal., (1982).

[8] **Good, I.J.**, 'The serial test for sampling numbers and other tests for randomness', *Proc. Camb. Phil. Soc.*, **49**, (1953), pp 276-284.

[9] **Kimberley, M.E.**, *'Statistics in Cryptology'*, M.Sc. Dissertation, Brunel University, (1986).

[10] **Knuth, D.E.**, *'The Art of Computer Programming, Volume 2: Seminumerical Algorithms'*, 2nd edition, Addison-Wesley, Reading. Mass., (1981).

[11] **Massey, J.L.**, 'Shift register synthesis and BCH decoding', *IEEE Trans. Information Theory*, **IT-15**, (1969), pp 122-127.

[12] **Mood, A.M.**, 'The distribution theory of runs', *Ann. Math. Statist.*, **11**, (1940), pp 367-392.

[13] **Niederreiter, H.**, 'The probabilistic theory of linear complexity', *Advances in Cryptology: Proceedings of Eurocrypt 88*, Springer-Verlag, Berlin, (1988), pp 191-209.

[14] **Rueppel, R.A.**, *'New Approaches to Stream Ciphers'*, D.Sc. Dissertation, Swiss Federal Institute of Technology, Zurich, (1984).

[15] **Rueppel, R.A.**, *'Analysis and Design of Stream Ciphers'*, Springer-Verlag, Berlin, (1986).

[16] **Selmer, E.S.**, *'Linear Recurrence Relations over Finite Fields'*, University of Bergen, (1966).

[17] **Sugiyama, Y., Kasahara, M., Hirasawa, S. and Namekawa, T.**, 'A method for solving key equation for decoding Goppa codes', *Information and Control*, **27**, (1975), pp 87-99.

[18] **Wang, M.Z. and Massey, J.L.**, 'The characterization of all binary sequences with perfect linear complexity profiles'. Presented at Eurocrypt 86.

[19] **Wetherill, G.B.**, *'Elementary Statistical Methods'*, 3rd edition, Chapman and Hall, London, (1982).

[20] **Zierler, N.**, 'Linear recurring sequences', *J. Soc. Ind. Appl. Math.*, **7**, (1959), pp 31-48.