On the Generation and Analysis of Pseudorandom Sequences over Arbitrary Finite Fields

Submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy
by
Suman Roy

under the supervision of
Dr. Srinivasan Krishnaswamy

Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati
Guwahati-781039, Assam, India

January 2020

To

My Parents

Mr. Sankar Roy and Mrs. Mithu Roy

and

My Guru

Dr. Srinivasan Krishnaswamy

Certificate

This is to certify that the thesis titled "**On the Generation and Analysis of Pseudorandom Sequences over Arbitrary Finite Fields**", submitted by **Suman Roy** (Roll No.:126102024), a research scholar in the Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati, for the award of the degree of **Doctor of Philosophy**, is a record of an original research work carried out by him under my supervision and guidance. The thesis has fulfilled all requirements as per the regulations of the institute and in my opinion has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Date:

Place: Guwahati

Dr. Srinivasan Krishnaswamy
Dept. of Electronics and Electrical Engg.,
Indian Institute of Technology Guwahati,
Guwahati - 781039, Assam, India.

Declaration

I declare that this written submission represents my ideas in my own words, and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Suman Roy

Roll No.: 126102024

Date:

Place: Guwahati

Acknowledgements

Ah! Finally, an edifying voyage comes to an end and this thesis becomes an evidence of all the ups and downs of my journey over the past few years. But, life is again knocking at the door with a new set of challenges and opportunities. At this juncture, I would like to thank all those wonderful people without whom this thesis would never have been written. Their unconditional constant support and inspiration made my six-year stay at IITG by far the most enjoyable experience of my life so far.

First and foremost, I will be forever grateful to my supervisor Dr. Srinivasan Krishnaswamy. He has truly inspired me and helped me a lot during these years. His patience and encouragement have been very important in addition to his thoroughness in research. I feel fortunate to have him as my friend, philosopher and guide throughout these years. Without his guidance, I would not be even able to start working in this area of research or shape my thesis.

I would like to express my deepest gratitude to all, the faculty members of Department of Electronics and Electrical Engineering, IIT Guwahati. Especially, to my doctoral committee (DC) members Dr. I. Kar, Dr. B.K. Rai and Dr. D. Kesh (Dept. of CSE) for sparing their precious time out of their busy schedule to evaluate my time to time progress and enrich this work with their invaluable comments and suggestions. I am especially grateful to Dr. D. Pal (Dept. of EE, IIT Bombay). His informative and

energetic lectures on Algebra inspired and mesmerised us all and I was also not an exception.

I want to thank all the research and technical staff members of the department, especially Mr. Mukut, Late Mr. Uday, Mr. Dasharath, Mr. Mazid ,Ms. Khurshida, and Mr. Sonowal. Without their help, I could not have completed this thesis. My special thanks to Mr. Sanjib for maintaining an excellent computing facility and various resources used for the research work.

Now, it is time to thank all my lab mates, seniors, and friends. Without their useful suggestions, support and companionship, this journey would not have been such a smooth ride. My seniors Dr. Arghya, Dr. Himadri, Dr. Karteek, Dr. Mridul, Dr. Saurabh, Dr. Shounak, Dr. Sujoy, Dr. Tousif, Dr. Vinay and Sushanta da, Mandar da along with my friends Abhradip, Ardhendu, Arijit, Arunangshu, Atanu, Brijesh, Gautam, Gourhari, Gundappa, Jitendra, Kamakshi, Karnika, Kasturi, Kasyap, Mitali, Mriganka, Pradip, Rajkumar, Subrata, Trusna, and Uddipana, I thank all of them. Also, I feel fortunate enough in getting a chance to meet some awesome people like Mousumi (IITB), Poulomee (NITD), Wg. Cdr. Uday and Sqn. Ldr. Deepak. during this period.

How can I forget my Bengali drama group at IITG? Like many, I too always love to do something creative in my free time. So, we formed 'Lubdhak'. I would like to thank each and every member of my 'Lubdhak' family for making my stay at IITG the most colourful and enjoyable. Those happy moments will be etched in my mind and heart forever.

Last but not least, I salute my Maa and Baba for their immense patience. Without their support I would never have been able to complete my Ph.D. And — my beloved Sunandita — it is difficult to express my gratitude towards her in words. She always believes in me and stands by

me to hold my hands in every hard time I faced during my Ph.D. life.

Thank you all!

Date: Suman Roy

Abstract

A sequence generated by a deterministic algorithm is called a pseudorandom sequence if it satisfies statistical properties similar to a truly random sequence. Such sequences arise in many applications starting from key generation in cryptography to spread-spectrum communication, and so on. In this thesis, we address topics related to the generation and analysis of pseudorandom sequences.

Many pseudorandom sequence generators are based on Linear Feedback Shift Registers (LFSRs). LFSR is a shift register whose input bit is a linear function of some bits of the shift register value. Nonlinear feed-forward logic can be used along with a primitive-LFSR to increase the linear complexity of the generated sequences. Such an arrangement is called a Non-Linear Feed-forward Generator (NLFG). In this work, we first have analysed the statistical distribution of sequences generated by NLFGs over arbitrary finite fields. We then move on to describe two methods of extending nonlinear feed-forward logic to word-based LFSRs. A special kind of word-based LFSRs, known as σ-LFSRs, has been considered here. The first NLFG configuration uses permutation matrices while the second one sees a σ-LFSR over an extension field. For the first scheme, we have shown using simulation that the statistical distribution of each component sequence is better than that of an existing scheme. And, for the second scheme, we have mathematically analysed the statistical distribution of the

output vector sequence and have shown that it is more balanced compared to the sequences generated by schemes available in literature.

The last part of this thesis deals with the generation of de Bruijn sequences. An n-th order binary de Bruijn sequence is a periodic sequence of length 2^n in which every n-length subsequences occurs exactly once in each period. These sequences have good statistical properties associated with randomness such as long period, balance, ideal n-tuple distribution, and high linear complexity. Due to these reasons, de Bruijn sequence generators are used in many pseudorandom number generators and are also used as building blocks for stream ciphers. Here, we describe a method of traversing a set of n-th order binary de Bruijn sequences using a series of graph-theoretic transformations.

সংক্ষিপ্তসার

[Abstract in Bengali language]

কোনো নির্ণায়ক গাণিতিক প্রণালীলব্ধ একটি ধারাবাহিক সংখ্যারাশি (sequence) যদি প্রকৃতিলব্ধ যদৃচ্ছ ধারাবাহিক সংখ্যারাশির অনুরূপ সংখ্যাতাত্ত্বিক গুণাবলী বহন করে, তবে তাকে ছদ্মযদৃচ্ছ ধারাবাহিক সংখ্যারাশি (pseudorandom sequence) বলে। সংকেতলিপিবিদ্যা থেকে শুরু করে বৈদ্যুতিন যোগাযোগ ব্যবস্থা সহ আরও অনেক ক্ষেত্রে এই ধরণের সংখ্যারাশির বহুল প্রয়োগ লক্ষণীয়। এই গবেষণামূলক নিবন্ধে আমরা এইরকম ছদ্মযদৃচ্ছ ধারাবাহিক সংখ্যারাশির উৎপাদন ও তাদের বিশ্লেষণ সংক্রান্ত বিষয়ে আলোকপাত করব।

বেশিরভাগ ছদ্মযদৃচ্ছ সংখ্যারাশির উৎপাদকগুলি Linear Feedback Shift Register (LFSR)[1] ভিত্তিক হয়ে থাকে। এইসব উৎপন্ন সংখ্যারাশির রৈখিক জটিলতা (linear complexity) বৃদ্ধির জন্য LFSR-এর সাথে nonlinear feed-forward logic নামক একটি অরৈখিক বর্তনীর ব্যবহার করা হয়। এই ধরনের সমণ্বয়কে Nonlinear Feed-Forward Generator (NLFG) বা অরৈখিক উৎপাদক বলে। এই গবেষণায়, আমরা প্রথমত একটি যথেচ্ছ finite field-এ সংজ্ঞায়িত NLFG উদ্ভূত সংখ্যারাশির পরিসংখ্যান-সংক্রান্ত বিন্যাসের (statistical distribution) বিশ্লেষণ করেছি। অতঃপর, আমরা word-based LFSR গুলির উপরও এইধরণের অরৈখিক বর্তনীগুলোকে প্রসারিত করার দুটি পদ্ধতির বর্ণনা করেছি। আমাদের এই আলোচনায় একটি বিশেষ ধরনের word-based $LFSR-$কে গণ্য করা হয়েছে যা σ-LFSR নামে পরিচিত। প্রথম পদ্ধতিতে permutation matrix ব্যবহৃত হয়েছে এবং দ্বিতীয় ক্ষেত্রে σ-LFSR$-$কে একটি extension field-এর দৃষ্টিকোন থেকে ব্যাখ্যা করা হয়েছে। প্রথম পদ্ধতিতে simulation-এর সাহায্যে উৎপাদিত প্রতিটি শাখা অনুক্রমগুলিকে (component sequences) সংখ্যাতাত্ত্বিক বিন্যাসের বিচারে পূর্বলব্ধ পদ্ধতির চেয়ে শ্রেষ্ঠতর দেখানো হয়েছে। অন্যদিকে, দ্বিতীয় পদ্ধতিতে উৎপাদিত ধারাবাহিক ভেক্টররাশির গাণিতিক বিশ্লেষণ করে দেখানো হয়েছে যে উৎপাদিত ভেক্টররাশির সংখ্যাতাত্ত্বিক বিন্যাস সাহিত্যলভ্য পদ্ধতিগুলির চেয়ে অধিকতর সুষম।

নিবন্ধের অন্তিম পর্যায়ে de Bruijn সংখ্যারাশির[2] উৎপাদন সম্পর্কে আলোচনা করা হয়েছে। একটি n ক্রমের binary de Bruijn সংখ্যারাশি হল এমন একটি 2^n দৈর্ঘ্যের পর্যায় ক্রমিক সংখ্যারাশি যার প্রতিটি পর্যায়ে n দৈর্ঘ্যের সম্ভাব্য সকল উপসংখ্যারাশিগুলি বর্তমান। এ ধরনের সংখ্যারাশি গুলির যদৃচ্ছতা সম্পর্কিত ভাল পরিসংখ্যানগত বৈশিষ্ট্য থাকে যেমন দীর্ঘ পর্যায়, সুষম বিন্যাস, উচ্চ রৈখিক জটিলতা ইত্যাদি। একারনে বিভিন্ন ছদ্মযদৃচ্ছ সংখ্যারাশির উৎপাদকগুলিতে এগুলি ব্যবহৃত হয়। এখানে আমরা একটি নির্দিষ্ট ক্রমের de Bruijn sequence-এর শ্রেণীতে অবাধে বিচরণের একটি গ্রাফ-তাত্ত্বিক পদ্ধতির বর্ণনা করেছি।

[1] Linear Feedback Shift Register একটি বিশেষ ধরণের বৈদ্যুতিক বর্তনী।

[2] এটি গণিতজ্ঞ de Bruijn এর নামানুসারে খ্যাত একটি ধারাবাহিক সংখ্যারাশি।

Contents

List of Figures

List of Tables

List of Symbols

$\lvert S\rvert$	: Cardinality of any given set S
$S \setminus \{e\}$	: Set S without the element e
$\lfloor x \rfloor$	: The greatest integer less than or equal to x
v^T	The transpose of a vector v
$\mathbb{F}_q$	: Finite field of order $q = p^n$, where p is a prime number and n is a positive integer
$\mathbb{F}_q^n$	: n-dimensional vector space over $\mathbb{F}_q$
$\mathbb{F}_q^{m\times n}$	: A matrix of order $m \times n$ with elements from $\mathbb{F}_q$
$[x_1, x_2, \ldots, x_n]$	: A matrix with rows $x_1, x_2, \ldots, x_n$
$[y_1; y_2; \ldots; y_n]$	: A matrix with columns $y_1, y_2, \ldots, y_n$
$\mathbb{F}[s]$	: Ring of polynomials in indeterminate s and coefficients from $\mathbb{F}$
$\mathbb{N}$, $\mathbb{Z}$, and $\mathbb{R}$	: Set of natural, integer, and real numbers, respectively
$DB_n(k)$	: n-th order k-ary de Bruijn sequence.
$G(V, E)$	: Directed graph with vertex set V and edge set E
G_n	: A binary De Bruijn graph of order n
$indeg(v)$	: Number of edges entering into a vertex v
$outdeg(v)$	: Number of edges leaving from a vertex v
$s(e)$	: The source vertex of the edge $e \in E$
$t(e)$	: The target vertex of the edge $e \in E$

List of Abbreviations

FSR	:	Feedback Shift Register
LCM	:	Least Common Multiple
LFSR	:	Linear Feedback Shift Register
LRR	:	Linear Recurring Relation
NLFG	:	Non-Linear Feed-forward Generator
PRNG	:	Pseudorandom Number Generator

Chapter 1

Introduction

> "Random numbers should not be generated with a method chosen at random."
>
> *Donald E. Knuth*

Sequences, generated by deterministic algorithms, which approximate the properties of truly random sequences are called pseudorandom sequences. Such sequences over finite fields find many applications ranging from cryptography [LN86, MOV96, PP09] and error-correcting codes [PW72] to spread-spectrum communication [PSM82]. In this thesis, we explore the following two topics related to the pseudorandom number generation.

- The use of Non-Linear Feed-forward Generators (NLFGs) to increase the linear complexity of sequences generated by LFSRs over arbitrary finite fields and word-based LFSRs.
- Generation of de Bruijn sequences.

In addition to the increase in linear complexity using NLFG, it is important that the statistical properties of the generated sequence are close to those of a random sequence. In this work, it has been shown that the frequency of symbols in sequences generated by an NLFG over arbitrary finite fields tends to a balanced distribution as the number of multipliers,

that are involved in the nonlinear logic, tends to infinity. Further, NLFGs over word-based σ-LFSRs that generate balanced vector sequences with other desirable statistical properties have been proposed. Lastly, a graph-theoretic method of generating any given n-th order de Bruijn sequence from any other n-th order de Bruijn sequence has been developed.

1.1 Nonlinear feed-forward generator

Due to simplicity in hardware implementation, linear feedback shift registers (LFSRs) are commonly used as basic building blocks for pseudo-random number generators (PRNGs). Sequences that are generated by primitive-LFSRs (LFSRs with primitive characteristic polynomials) have some good statistical properties associated with randomness such as balance property, run property, span-n property, and 2-level auto-correlation property [Gol81]. Thus, primitive-LFSR based PRNGs are widely used in stream ciphers, cryptographic protocols, digital signature generation algorithms, RFID systems, and many more [GG04, HJM07, PP09].

However, sequences generated by primitive-LFSRs are marred by their low linear complexity [Gol81]. This is undesirable from a cryptographic point of view. A method to overcome this shortcoming has been suggested in [Gro71]. Here, nonlinear feed-forward logic is used in conjunction with LFSRs. Such a structure is known as nonlinear feed-forward generator (NLFG). An analysis of the linear complexity of binary sequences generated by NLFGs is given in [Gro71, Key76]. Statistical properties of the generated sequences have been investigated in [DAG90, BP01, GG06, Teo13]. In this work, we have extended some of these results to NLFGs that operate over arbitrary finite fields.

In order to utilise parallelism offered by modern processors, various word-based LFSR configurations have been proposed [TV02, DP03]. In

contrast to conventional LFSRs, word-based LFSRs consist of multi-input multi-output delay blocks. The σ-LFSR, described in [ZHH07], is one such configuration. As with conventional LFSRs, vector sequences generated by σ-LFSRs with primitive characteristic polynomials have good statistical properties associated with randomness. [HPW12] discusses a way of extending the notion of nonlinear feed-forward logic to σ-LFSRs. In this thesis, it has been shown that the sequences generated by this method, however, have poor statistical properties. Also, we have proposed two new methods of extending nonlinear feed-forward logic to word-based σ-LFSRs. The first method involves the use of certain permutation matrices. In the second method, we have proposed an NLFG configuration which considers a σ-LFSR as an FSR over an extension field.

1.2 Generation of de Bruijn sequences

An n-th order k-ary de Bruijn sequence, $DB_n(k)$, is a periodic sequence of length k^n having every possible k-ary subsequence of length n exactly once in each period. An example of $DB_2(3)$ is 001021122. In [Bru46], it has been shown that there exist $((k-1)!)^{k^{n-1}} k^{k^{(n-1)}-n}$ k-ary de Bruijn sequences of order n. De Bruijn sequences satisfy many statistical properties associated with randomness such as balance property, span-n property, etc. There are several algorithms available in literature to generate de Bruijn sequences using shift registers [Fre82, Gol81]. One way of generating de Bruijn sequences is by joining shorter cycles generated by Feedback Shift Registers (FSRs) [EL84, Jan89]. Given a k-ary de Bruijn sequence of order n, other such sequences can be obtained by using cross-join pairs [Fred68, MS15]. Recursive algorithms to produce higher-order de Bruijn sequences from lower-order de Bruijn sequences have been discussed in [Lem70, Ann97]. [BK11] gives a constructive method of enumerating all

de Bruijn sequences of a given order. In this thesis, we have used this construction to develop a method by which any given n-th order de Bruijn sequence can be generated from any other n-th order de Bruijn sequence by using a set of transformations. So, this method helps to produce an arbitrary de Bruijn sequence of a given order by choosing a set of possible transformations randomly.

1.3 Outline of the thesis

The remaining part of the thesis is organised as follows:

Chapter 2 defines various mathematical preliminaries that are used in the rest of the thesis. These include concepts in abstract algebra, linear algebra, and graph theory.

In Chapter 3, we give a brief introduction of LFSRs and discuss the properties of sequences generated by them. We then go on to describe a special type of word-based LFSR, known as σ-LFSR.

Chapter 4 deals with nonlinear feed-forward logic in conjunction with LFSRs over arbitrary finite fields. In this chapter, we have analysed the statistical distribution of sequences generated by such nonlinear feed-forward generators (NLFGs).

In Chapter 5, we discuss NLFGs over σ-LFSRs. In addition to the analysis of an existing scheme, we have proposed two novel methods of extending nonlinear feed-forward logic to σ-LFSRs. Further, we compare the statistical distribution of sequences generated by the proposed scheme to those generated by the existing one.

Chapter 6 focuses on the generation of de Bruijn sequences. We show that, it is possible to transform a given de Bruijn sequence to any other one of the same order using a series of graph transformations.

Chapter 7 summarises the key contributions of this thesis and presents

future research directions related to this work.

Chapter 2

Mathematical Background

"God used beautiful mathematics in creating the world."

Paul Dirac

In this chapter, we introduce some fundamental mathematical concepts in algebra and graph theory, which will be used in the rest of this thesis. A more detailed exposition of modern algebra and graph theory can be found in [Art11, Gal13, GG04, LN86, GYZ14]. Readers acquainted with these basic facts may want to skip this chapter, and proceed to the next chapter.

2.1 Algebraic preliminaries

We start our discussion by defining some fundamental algebraic structures.

Definition 2.1.1. ***(Group)***
A nonempty set $\mathcal{G}$ associated with a binary operation $*$ is said to be a group $(\mathcal{G}, *)$, if it satisfies the following properties.

1. *For all g_1, $g_2 \in \mathcal{G}$, $g_1 * g_2 \in \mathcal{G}$ (closure property).*
2. *For all g_1, g_2, and $g_3 \in \mathcal{G}$, $g_1 * (g_2 * g_3) = (g_1 * g_2) * g_3$ (associative property).*
3. *There exist an element $e \in \mathcal{G}$, such that for all $g \in \mathcal{G}$, $g*e = e*g = g$ (identity element).*

4. *There exist an element* $g' \in \mathcal{G}$, *such that for all* $g \in \mathcal{G}$, $g * g' = g' * g = e$ *(inverse element).*

Moreover, a group is said to be commutative (or Abelian) if $*$ *is commutative. In case of an additive group,* $*$ *denotes the addition operation and for a multiplicative group* $*$ *denotes the multiplication operation.*

Example 2.1.1. *Let* n *be a nonzero positive integer. Define the set* $\mathbb{Z}_n := \{0, 1, \ldots, n-1\}$. *It can easily be verified that* $\mathbb{Z}_n$ *forms an Abelian group under the addition modulo* n *operation* $(+_n)$ *and referred to as the group of integers modulo* n *and denoted by* $(\mathbb{Z}_n, +_n)$.

Definition 2.1.2. ***(Order of a group)***
The cardinality of the set $\mathcal{G}$ *is called the order of a group* $(\mathcal{G}, *)$.

The order of $(\mathbb{Z}_n, +)$ in Example 2.1.1 is n. If the order of a group is finite then it is called a finite group otherwise, an infinite group.

Definition 2.1.3. ***(Order of an element)***
The order of an element $g \in \mathcal{G}$ *of a group is the smallest positive integer* m *such that* $g^m = e$, *where* e *denotes the identity element of the group, and* g^m *indicates* $\underbrace{g * g * \ldots * g}_{m\text{-times}}$. *If no such* m *exists,* g *is said to have infinite order.*

Example 2.1.2. *Consider the set* $\mathbb{Z}_6 = \{0, 1, 2, 3, 4, 5\}$ *under the addition modulo* 6. *Order of the group* $\mathbb{Z}_6$ *is* 6 *and order of the element* $2 \in \mathbb{Z}_6$ *is* 3.

Definition 2.1.4. ***(Subgroup)***
Let $(\mathcal{G}, *)$ *be a group and* $\mathcal{H}$ *be a subset of* $\mathcal{G}$. *If* $\mathcal{H}$ *also forms a group under the same operation* $*$, *then* $(\mathcal{H}, *)$ *is called a subgroup of* $(\mathcal{G}, *)$.

For example, $(\mathbb{Z}, +)$ is a subgroup of $(\mathbb{R}, +)$ under the addition operation.

Definition 2.1.5. ***(Cyclic group)***
A group $(\mathcal{G}, *) = (a)$ *is said to be cyclic if there is an element* $a \in \mathcal{G}$ *such that for any* $b \in \mathcal{G}$ *there is some integer* i *such that* $b = a^i = \underbrace{a * a * \ldots * a}_{i\text{-times}}$.
Moreover, a *is called a generator of the cyclic group of* $\mathcal{G}$.

A cyclic group may have one or more than one generators. For example, it can be verified that for any prime p, $(\mathbb{Z}_p, +_p)$ forms a cyclic group wherein

every element is a generator. The number of generators of a finite cyclic group can be determined by using Euler's phi function.

Definition 2.1.6. ***(Euler's phi function)***
Given a positive integer n, the Euler's phi function, $\phi(n)$, counts the number of positive integers that are less than n and coprime to n.

Now, consider a finite cyclic group $\mathcal{G}$ having cardinality k. Let a be a generator of $\mathcal{G}$. Every other generator of $\mathcal{G}$ is an element of the form a^j, where $j \leq k$ is an integer which is coprime to k. Therefore, the number of generators of a finite cyclic group $\mathcal{G}$ is equal to $\phi(|\mathcal{G}|)$.

Definition 2.1.7. ***(Ring)***
A nonempty set $\mathcal{R}$ with two binary operations $+$ and $.$ is called a ring $(\mathcal{R}, +, .)$ if it satisfies the following properties.

1. *$(\mathcal{R}, +)$ is an Abelian group under the $+$ operation.*
2. *The $.$ operation is associative. i.e.,*
$$a.(b.c) = (a.b).c$$
3. *The operation $.$ is distributive over $+$, i.e., for all $a, b,$ and $c \in \mathcal{R}$, we have*
$$a.(b + c) = a.b + a.c$$

Definition 2.1.8. ***(Subring)***
A nonempty subset of a ring $(\mathcal{R}, +, .)$ which forms a ring in itself, under the operations defined for $\mathcal{R}$, is known as a subring of $\mathcal{R}$.

1. For a ring $\mathcal{R}$, if the multiplication operation is commutative then $\mathcal{R}$ is called a commutative ring. i.e., for all $a, b \in \mathcal{R}$
$$a.b = b.a$$
2. If the $.$ operation has an identity, denoted by 1, such that for all $a \in \mathcal{R}$,
$$a.1 = 1.a = a$$
then the ring $\mathcal{R}$ is called a ring with identity.

The set of integers ($\mathbb{Z}$), the set of rationals ($\mathbb{Q}$) and the set of real numbers ($\mathbb{R}$) are all examples of commutative rings with identity. Further, $\mathbb{Z}$ is a subring of $\mathbb{Q}$ and $\mathbb{Q}$ is a subring of $\mathbb{R}$. Now, we proceed to define some important classes of ring such as integral domain, ideal and field.

Definition 2.1.9. ***(Integral Domain)***
A commutative ring with identity is called an integral domain if $a.b = 0$ *implies either* $a = 0$ *or* $b = 0$.

Definition 2.1.10. ***(Ideal)***
A subring $\mathcal{I}$ *of a ring* $\mathcal{R}$ *is called an ideal (two-sided) of* $\mathcal{R}$*, if for all* $r \in \mathcal{I}$ *and* $r' \in \mathcal{R}$ *both* $r.r'$ *and* $r'.r$ *are in* $\mathcal{I}$.

Before defining a field, we now go on to discuss about the equivalence relation on a set.

Definition 2.1.11. ***(Equivalence Relation)***
Let $\mathcal{S}$ *be a nonempty set. An equivalence relation on* $\mathcal{S}$ *is a set* $\mathcal{R}$ *of ordered pairs of elements of* $\mathcal{S}$ *with the following properties.*

1. $(x, x) \in \mathcal{R}$ *for all* $x \in \mathcal{S}$ *(reflexive property).*
2. $(x, y) \in \mathcal{R}$ *implies* $(y, x) \in \mathcal{R}$ *(symmetric property).*
3. $(x, y) \in \mathcal{R}$ *and* $(y, z) \in \mathcal{R}$ *imply* $(x, z) \in \mathcal{R}$ *(transitive property).*

If $(a, b) \in \mathcal{R}$, then we say a is related to b and write this as $a \sim b$. The set $[a]$=$\{x \in \mathcal{S} | x \sim a\}$ is referred to as equivalence class of $\mathcal{S}$ containing a. Note that, an equivalence relation $\mathcal{R}$ on a set $\mathcal{S}$ partitions the set $\mathcal{S}$ into some equivalence classes.

Given an ideal $\mathcal{I}$ of a ring $\mathcal{R}$, consider the following equivalence relation on the set R.

$$R := \{(x, y) \in \mathcal{R} \times \mathcal{R} \mid x - y \in \mathcal{I}, \quad for\ some\ n \in \mathcal{I}\}$$

The equivalence classes under the above equivalence relation R are called residue classes of R modulo $\mathcal{I}$, denoted by $R/\mathcal{I}$. Further, it can be

easily verified that $R/\mathcal{I}$ forms a ring, called the residue class ring, under the following operations:

$$[x] + [y] = [x + y]$$

$$[x].[y] = [x.y]$$

Example 2.1.3. *Let n be a positive integer. For any $x, y \in \mathbb{Z}$, we say that x is congruent modulo y, denoted by $x \equiv y \ (mod\ n)$, if n divides $x - y$. Now, consider the set $\langle n \rangle = \{0, \pm n, \pm 2n, ...\}$ where $\langle n \rangle$ denotes the smallest ideal of $\mathbb{Z}$ containing n. Define a relation, R, on the set $\mathbb{Z}$ as follows.*

$$R := \{(x, y) \in \mathbb{Z} \times \mathbb{Z} \mid x \equiv y \ (mod\ n)\}$$

It can be easily verified that R is an equivalence relation with the following equivalence classes.

$$\begin{aligned}
[0] &= \{\ldots, -2n, -n, 0, n, 2n, \ldots\} \\
[1] &= \{\ldots, -2n+1, -n+1, 1, n+1, 2n+1, \ldots\} \\
&\vdots \\
[n-1] &= \{\ldots, -n-1, -1, n-1, 2n-1, 3n-1, \ldots\}
\end{aligned}$$

These equivalence classes constitute the the residue class ring $\mathbb{Z}/\langle n \rangle$.

Definition 2.1.12. ***(Field)***
Let $(\mathbb{F}, +, .)$ be an algebraic structure containing at least two elements and with two binary operations $+$ and $.$. $(\mathbb{F}, +, .)$ is said to be a field if $\mathbb{F}$ satisfies the following:

1. *$(\mathbb{F}, +)$ is an Abelian group.*
2. *$(\mathbb{F}^*, .)$ is an Abelian group, where $\mathbb{F}^* = \mathbb{F} \setminus \{0\}$.*
3. *the multiplication $(.)$ operation distributes over addition $(+)$.*

It can be easily verified that the set of real numbers $(\mathbb{R}, +, .)$ together with addition and multiplication forms a field with infinite cardinality. If the cardinality of a field is finite then it is called a finite field. A field with cardinality q is denoted by $\mathbb{F}_q$ or $GF(q)$.

Theorem 2.1.1 (Theorem 13.2 [Gal13]). *For every prime number p, the ring of integers modulo p, $\mathbb{Z}_p$, is a finite field.*

In the following, we consider the ring of polynomials over a field $\mathbb{F}$, it is denoted by $\mathbb{F}[x]$.

Definition 2.1.13. ***(Irreducible polynomial)***
A non-constant polynomial $f(x) \in \mathbb{F}[x]$ is called irreducible over $\mathbb{F}$, whenever $f(x) = a(x)b(x) \in \mathbb{F}[x]$ implies either $a(x)$ or $b(x)$ is a constant polynomial.

It is well known that there exist at least one irreducible polynomial of degree n over $\mathbb{Z}_p$ for every prime p and degree $n > 1$, [LN86]. We now proceed to construct the finite field $\mathbb{F}_q$ with q elements, where q is a prime power.

Let n be a positive integer. To construct the finite field $\mathbb{F}_q$ with $q = p^n$ elements, we choose an irreducible polynomial $f(x)$ of degree n over $\mathbb{F}_p$. Let α be a root of $f(x)$. Consider the set

$$GF(p^n) = \{a_0 + a_1\alpha + \ldots + a_{n-1}\alpha^{n-1} | a_i \in \mathbb{F}_p\}.$$

We define the operations addition $(+)$ and multiplication $(.)$ over this set as follows. For two elements $g(\alpha) = \sum_{i=0}^{n-1} a_i\alpha^i$ and $h(\alpha) = \sum_{i=0}^{n-1} b_i\alpha^i$ in $GF(p^n)$,
Addition: $g(\alpha) + h(\alpha) = \sum_{i=0}^{n-1}(a_i + b_i)\alpha^i \in GF(p^n)$
Multiplication: $g(\alpha).h(\alpha) = r(\alpha)$, where $r(x)$ is $g(x)h(x) \bmod f(x)$.

Theorem 2.1.2. *(Theorem 3.5 [GG04]) The set $GF(p^n)$ together with the two above mentioned operations $+$ and $.$ forms a finite field, $\mathbb{F}_q$ of order $q = p^n$.*

The polynomial $f(x)$ and the element α are called the defining polynomial and defining element of $\mathbb{F}_q$ respectively. The process of constructing $\mathbb{F}_q$ by adjoining a root of an irreducible polynomial to $\mathbb{F}_p$ is known as field extension.

Example 2.1.4. *Let $p = 2$. $f(x) = x^3+x^2+1$ is an irreducible polynomial over $\mathbb{F}_2$. Let α be a root of $f(x)$. The finite field $\mathbb{F}_{2^3}$ has the following 8 elements.*

$$\{0, 1, \alpha, \alpha^2, \alpha^2 + 1, \alpha^2 + \alpha + 1, \alpha + 1, \alpha^2 + \alpha\}$$

Definition 2.1.14. ***(Characteristic of a field)***
Given a field $\mathbb{F}$, the characteristic of $\mathbb{F}$ is the least positive integer k such that $kx = 0$ for all $x \in \mathbb{F}$. If no such integer exists, we say that $\mathbb{F}$ has characteristic 0. The characteristic of $\mathbb{F}$ is denoted by char($\mathbb{F}$).

The above definition leads us to the following results.

Theorem 2.1.3. *(Corollary 1.45 [LN86])) Any finite field has a prime characteristic.*

Theorem 2.1.4. *(Corollary 3.3 [GG04]) In any finite field of characteristic p,*

$$(x + y)^{p^m} = x^{p^m} + y^{p^m}$$

for any integer $m \geq 1$.

If a subset $\mathbb{K}$ of a field $\mathbb{F}$ is a field in itself then $\mathbb{K}$ is called a subfield of $\mathbb{F}$ or $\mathbb{F}$ is called an extension (field) of $\mathbb{K}$. If every element of $\mathbb{F}$ is a root of some nontrivial polynomial with coefficients in $\mathbb{K}$ then $\mathbb{F}$ is said to be an algebraic extension of $\mathbb{K}$. If $\mathbb{F}$ is an algebraic extension of a field $\mathbb{K}$ then $\mathbb{F}$ can be seen as a vector space over $\mathbb{K}$. The dimension of $\mathbb{F}$ over $\mathbb{K}$ is denoted by $[\mathbb{F} : \mathbb{K}]$. If $[\mathbb{F} : \mathbb{K}]$ is finite then $\mathbb{F}$ is said to be a finite extension of $\mathbb{K}$. For example, the field of complex numbers $\mathbb{C}$ is an algebraic extension of the field of real numbers $\mathbb{R}$, of dimension 2.

The following theorem characterises all possible finite fields.

Theorem 2.1.5. *(Theorem 3.7 [GG04]) Let $\mathbb{F}$ be a finite field of order q with characteristic p. Then, either $\mathbb{F} = GF(p)$ when $q = p$ or $\mathbb{F}$ is a vector space over $GF(p)$ of dimension n when $q > p$; i.e., $q = p^n$.*

Let $\mathbb{F}$ and $\mathbb{G}$ be two finite fields. If there exist a one-to-one correspondence from $\mathbb{F}$ to $\mathbb{G}$ that preserves + and . operations, then $\mathbb{F}$ is said to

be *isomorphic* to $\mathbb{G}$. It can be proved that all finite fields of order p^n are isomorphic [LN86]. We denote a finite field with order q by $\mathbb{F}_q$. All nonzero elements of $\mathbb{F}_q$ form a multiplicative group, denoted by $\mathbb{F}_q^*$, where $\mathbb{F}_q^* = \mathbb{F}_q \setminus \{0\}$.

Theorem 2.1.6. *(Theorem 2.8 [LN86]) The multiplicative group $\mathbb{F}_q^*$ of every finite field $\mathbb{F}_q$ is a cyclic group.*

Definition 2.1.15. ***(Primitive Polynomial)***
If the root of a polynomial $p(x)$ of degree n is a generator of $\mathbb{F}_{p^n}^*$, then $p(x)$ is called a primitive polynomial over $\mathbb{F}_p$.

Example 2.1.5. *$x^3 + x + 1$ and $x^3 + x^2 + 1$ are two primitive polynomials over $\mathbb{F}_2$ having degree 3.*

Definition 2.1.16. ***(Order of a Polynomial)***
Let $f(x) \in \mathbb{F}_q[x]$ be a nonzero polynomial of degree n. If $f(0) \neq 0$, then there exists a least positive integer $\tau \leq q^n - 1$ for which $f(\tau)$ divides $x^\tau - 1$ is called the order (or period) of the polynomial $f(x)$.

Observe that, if the order of a monic polynomial in $\mathbb{F}_q[x]$ of degree n is $q^n - 1$ then the polynomial is primitive polynomial over $\mathbb{F}_q$. Clearly, primitive polynomials are irreducible polynomials. However, the converse is not true. For instance, $x^4 + x^3 + x^2 + x + 1$ is an irreducible polynomial of degree 4 over $\mathbb{F}_2$, although it is not primitive. Some important results related to the primitive polynomials are given as follows.

1. For any prime p or prime power $q = p^n$ and any positive integer n, there exists a primitive polynomial of degree n over $\mathbb{F}_q$.

2. The number ($\mathcal{N}$) of primitive polynomials over $\mathbb{F}_q$ is given by

$$\mathcal{N} = \frac{\phi(q^n - 1)}{n}, \text{ where } \phi \text{ is Euler's phi function}$$

2.1.1 Linear algebra over finite fields

In this subsection, we will discuss some basic concepts of linear algebra over finite fields.

Let $A \in \mathbb{F}_q^{n\times n}$ be a square matrix of order n over $\mathbb{F}_q$. We can define the following polynomials associated with the matrix A.

Definition 2.1.17. ***(Characteristic Polynomial)***
The characteristic polynomial of a matrix $A \in \mathbb{F}_q^{n\times n}$ is defined as the determinant of the polynomial matrix $sI - A$, where $I \in \mathbb{F}_q^{n\times n}$ represents the identity matrix.

Definition 2.1.18. ***(Minimal Polynomial)***
Let A be a square matrix over $\mathbb{F}_q^{m\times m}$. The minimal polynomial $p(x)$ of A is a monic polynomial with minimum degree such that $p(A) = 0$.

Consider a matrix $A \in \mathbb{F}_q^{n\times n}$ and a vector $v \in \mathbb{F}_q^n$. The minimal polynomial of v with respect to A is the lowest degree monic polynomial satisfying $p(A).v = 0$. Note that, the minimal polynomial of a vector v divides the minimal polynomial of the matrix A and the minimal polynomial of the matrix A divides its characteristic polynomial [Gal13].

Example 2.1.6. *Let $I_{n\times n}$ be the identity matrix of order n. The polynomials $(x-1)^n$ and $(x-1)$ are the characteristic and the minimal polynomial of $I_{n\times n}$ respectively.*

Consider a polynomial $f(x) = x^n - a_{n-1}x^{n-1} - a_{n-2}x^{n-2} - ... - a_0 \in \mathbb{F}_q[\mathrm{x}]$. We can define the following specially structured matrix $M \in \mathbb{F}_q^{n\times n}$. It is known as the companion matrix of $f(x)$.

$$M = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ a_0 & a_1 & a_2 & \dots & a_{n-1} \end{bmatrix} \in \mathbb{F}_q^{n\times n}$$

2.2 Basics of graph theory

Definition 2.2.1. ***(Graph)***
A graph $G = (V, E)$ consists of two sets V and E.

- *The elements of V are called vertices (or nodes).*
- *The elements of E are called edges (or arcs).*
- *Every edge has a set of one or two vertices associated to it, which are called its endpoints.*

Example 2.2.1. *Consider the graph G given in Figure 2.1.*

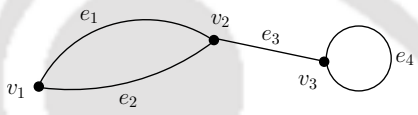

Figure 2.1: A graph G

Clearly, $V = \{v_1, v_2, v_3\}$ and $E = \{e_1, e_2, e_3, e_4\}$.

A vertex $u \in V$ is said to be adjacent to vertex $v \in V$ if they are connected by an edge. If two edges have a common endpoint then they are called adjacent edges. An edge that joins a single endpoint is called self-loop. A graph can be realised in a plane (or in space) together with a set of points (vertices)and a set of line segments (edges). However, these line segments may have a direction indicated by an arrow mark. A graph where all edges are directed is called a directed graph or digraph.

Example 2.2.2. *Figure 2.2 shows a directed graph.*

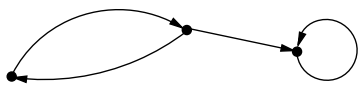

Figure 2.2: A directed graph

In a digraph, adjacent vertices are an ordered pair. Every edge $e \in E$ is directed from the source vertex s(e) to the target vertex t(e). For all $v \in V$, $indeg(v)$ and $outdeg(v)$ are the number of incoming and outgoing edges respectively. In a digraph with n vertices, $\sum_{i=1}^{n} indeg(v_i) = \sum_{i=1}^{n} outdeg(v_i) = |E|$.

Let $G = (V, E)$ be a graph. A graph $G' = (V', E')$ is said to be a subgraph of G if $V' \subseteq V$ and $E' \subseteq E$. Further, if G' and G have same vertex set, i.e., $V' = V$, then G' is a spanning subgraph of G. For instance, the following graph, given in Figure 2.3, is a directed spanning subgraph of the graph G.

Figure 2.3: A spanning subgraph

Definition 2.2.2. ***(Walk)***
In a graph G, a walk is an alternating sequence of vertices and edges,

$$W = v_1, e_1, v_2, e_2, \ldots, e_n, v_n$$

such that for $i = 1, \ldots, n$, the edge e_i has vertices v_i and v_{i+1} as the two endpoints.

Some important terminologies related to walk are given below.

- A trail in a graph is a walk such that no edge occurs more than once.
- A trail that starts and ends on the same vertex is called a cycle.
- A graph is said to be connected if between every pair of vertices there is a walk.

Definition 2.2.3. ***(Eulerian Path)***
An Eulerian path in a digraph G is a directed trail that contains each edge of G exactly once without any repetition of any internal vertex.

Definition 2.2.4. ***(Eulerian Cycle)***
A cycle that uses every edge of a graph exactly once is called an Eulerian cycle.

Definition 2.2.5. ***(Tree)***
A connected graph with no cycles (i.e., acyclic) is called a tree.

A spanning tree of a graph G is a spanning subgraph of G that is a tree. The following Figure 2.4 depicts an undirected graph and one of its spanning trees.

(a) A graph G (b) Spanning tree of G

Figure 2.4: A spanning tree of the graph G

2.3 Summary

In this chapter, we have reviewed some concepts related to basic algebraic structures like groups, rings and fields. Further, we have stated some fundamental results regarding polynomials over finite fields and graph theory which will be used in the subsequent chapters.

Chapter 3

Linear Feedback Shift Registers

> "The wheel is an extension of the foot, The book is an extension of the eye, Clothing an extension of the skin, electric circuitry, an extension of the central nervous system."
>
> *Marshall McLuhan*

In this chapter, we discuss LFSRs and mention some properties of the sequences generated by them. Finally, we briefly introduce a special word-based LFSR configuration, known as σ-LFSR.

3.1 LFSRs

A linear feedback shift register (LFSR) is an electronic circuit consisting of serially connected delay blocks (D flip-flops), feedback gain elements and adders. Figure 3.1 shows an L-stage LFSR containing L delay blocks (D_i's) along with feedback gains (a_i's). Here, $\bigotimes$ and $\bigoplus$ denote the multiplication and addition over the field $\mathbb{F}_q$ respectively.

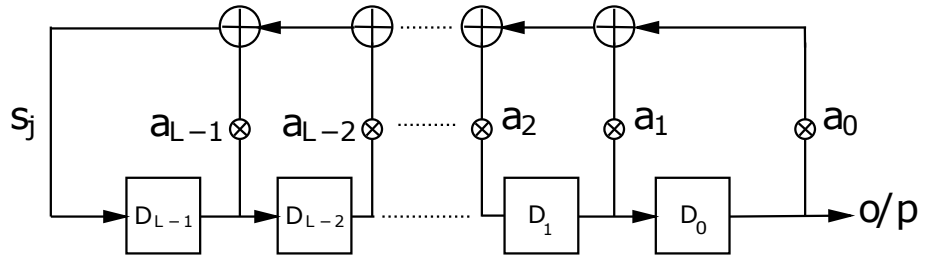

Figure 3.1: An L-stage LFSR

The output of an LFSR is an ultimately periodic sequence that satisfies a Linear Recurring Relation (LRR) of the following form.

$$s_{i+L} = a_0 s_i + a_1 s_{i+1} + \ldots + a_{L-1} s_{i+L-1} \tag{3.1}$$

where $a_i \in \mathbb{F}_q$ for $0 \leq i \leq L-1$. With such an LRR one can associate a unique monic polynomial having the same coefficients. Such a polynomial is called the characteristic polynomial of the LFSR. For example, the characteristic polynomial of the LFSR shown in Figure 3.1 is

$$f(x) = x^L - a_{L-1}x^{L-1} - \ldots - a_0 \tag{3.2}$$

Since we are dealing with periodic sequences, without loss of generality we can assume that $a_0 \neq 0$ (Theorem 6.11 [LN86]). The degree of the characteristic polynomial is known as the degree or length of the LFSR.

An LFSR can be seen as a state machine. The outputs of the delay blocks at any given time instant k constitute the state vector, $s(k)$, at that time instant. In other words, if the sequence generated by an L-stage LFSR is $\mathcal{S} = \{s_i\}_{i=0}^{\infty}$, then the state vector at any time instant k is given by $s(k) = [s_k \quad s_{k+1} \quad \ldots \quad s_{k+L-1}]^T \in \mathbb{F}_q^L$. Observe that, two consecutive state vectors are related by the equation $s(k+1) = As(k)$ where,

$$A = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ a_0 & a_1 & a_2 & \dots & a_{L-1} \end{bmatrix} \in \mathbb{F}_q^{L \times L}$$

The matrix A is called the state-transition matrix of the LFSR. Note that, A is the companion matrix of the characteristic polynomial of the LFSR. Thus, the characteristic polynomial of an LFSR and that of its state transition matrix are the same. An LFSR with a primitive characteristic polynomial is called a primitive-LFSR. For a nonzero initial state, a primitive-LFSR generates all nonzero states in a single period [LN86]. Thus, an L-stage primitive-LFSR over $\mathbb{F}_q$ generates a sequence with period $q^L - 1$. Such a sequence is called a maximal length sequence or an m-sequence. m-sequences have the following desirable statistical properties associated with randomness [Gol81].

Let $\mathbb{F}_q$ be a finite field with cardinality $q = p^r$, where p is a prime and r is a positive integer. Let $\mathcal{S}$ be a sequence over $\mathbb{F}_q$ of period τ.

1. *Balance Property*: let N_x be the number of occurrences of $x \in \mathbb{F}_q$ in a period of $\mathcal{S}$; i.e., $N_x = |\{i \ | \mathcal{S}_i = x, 0 \leq i < \tau\}|$. For each $x \neq y$, $|N_x - N_y| \leq 1$. For non-binary sequences of period $\tau = q^n - 1$, every nonzero element in $\mathbb{F}_q$ occurs q^{n-1} times and the zero element occurs $q^{n-1} - 1$ times in one period. This property is known as balance property of a non-binary sequence.

2. Span-n Property: A sliding window of length n, passed along an

m-sequence for $q^n - 1$ positions, will span every possible n length subsequence, except all-zero sequence, once and only once.

3. Run Property: For $x, y, z \in \mathbb{F}_q$ and $x \neq y$ and $y \neq z$, if the subsequence $x, \underbrace{y, y, \ldots y}_{k-times}, z$ occurs in $\mathcal{S}$, then $\underbrace{y, y, \ldots, y}_{k-times}$ is referred to as a run of y's of length k. Let $n = \lfloor log_q \tau \rfloor$, where $\lfloor x \rfloor$ denotes the floor function of x. In every period $\tau = q^n - 1$,

   (a) for $1 \leq k \leq n-2$, the runs of every element in $\mathbb{F}_q$ of length k occur $(q-1)^2 q^{n-k-2}$ times;

   (b) the runs of every nonzero element of length $n-1$ occur $q-2$ times;

   (c) the runs of zero element of length $n-1$ occur $q-1$ times; and

   (d) the run of every nonzero element of length n occurs once.

   This is referred to as the run property of nonbinary sequences.

4. Autocorrelation Property: The autocorrelation function $C(\lambda)$ of an m-sequence $\mathcal{S} = \{s_1, s_2, s_3, \ldots\}$ with period τ is two valued and is given by

$$C(\lambda) = \begin{cases} \tau & \text{if } \lambda \equiv 0 \ mod \ \tau \\ K & \text{if } \lambda \not\equiv 0 \ mod \ \tau \end{cases}$$

   where K is a constant and λ is the shifting operator. In this case, $\mathcal{S}$ is said to have an ideal 2-level autocorrelation function.

In particular, for binary case ($q = 2$), the above postulates reduce to the following.

1. *Balance property:* In a period, the number of ones is nearly equal to the number of zeros.

2. *Span-n property:* A sliding window of length n, passed along an m-sequence for $2^n - 1$ positions, will span every possible n length subsequence, except all-zero sequence, once and only once.

3. *Run property:* In a period, one-half the runs have length 1, one fourth the runs have length 2, one eighth the runs have length 3 and so on. Here, *run* indicates a sub-sequence of consecutive 1s or consecutive 0s which is neither preceded nor succeeded by the same symbol.

4. *2-level autocorrelation property:* The autocorrelation function $C(\lambda)$ of an m-sequence $\mathcal{S} = \{s_1, s_2 \ldots\}$ with period τ is two valued and is given by

$$C(\lambda) = \sum_{i=1}^{\tau} s_i s_{i+\lambda} = \begin{cases} \tau & \text{if } \lambda \equiv 0 \ mod\ \tau \\ K & \text{if } \lambda \not\equiv 0 \ mod\ \tau \end{cases}$$

where K is a constant and λ is the shifting operator.

Example 3.1.1. *Consider an LFSR over $\mathbb{F}_2$, as given in Figure 3.2, with 3 delay blocks, D_0, D_1 and D_2.*

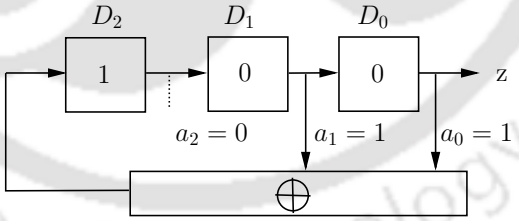

Figure 3.2: An example of a 3-bit LFSR

Let the initial state vector be $s(0) = [s_0\ s_1\ s_2]^T = [0\ 0\ 1]^T$. The corresponding LRR is $s_{i+3} = s_i + s_{i+1}$. Therefore, the characteristic polynomial of the LFSR is $p(x) = x^3 + x + 1$. Now, since $p(x)$ is a primitive polynomial over $\mathbb{F}_2$, every nonzero vector occurs exactly once in a cycle. The corresponding state diagram is depicted in Figure 3.3. As we are taking output from the last delay block, the output m-sequence, z, is 0010111 *with period* 7.

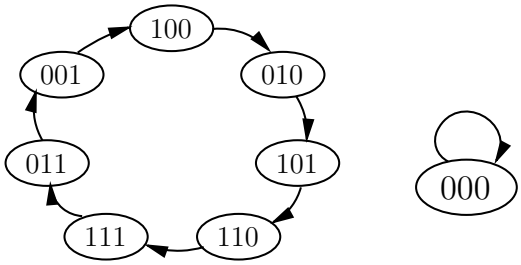

Figure 3.3: State diagram

Remark 3.1.1. *The characteristic polynomial of an LFSR being primitive ensures that all sequences generated by the LFSR are shifted versions of one another.*

Definition 3.1.1. *Any given periodic sequence, $\mathcal{S}$, over $\mathbb{F}_q$ can be generated by a set of LFSR configurations. The characteristic polynomial of the smallest degree LFSR that generates $\mathcal{S}$ is known as the minimal polynomial of $\mathcal{S}$. The degree of this minimal polynomial is called the linear complexity of the sequence $\mathcal{S}$.*

3.2 σ-LFSRs

In order to use the parallelism offered by modern processors, various word-based LFSR configurations have been proposed [TV02, DP03, ZHH07]. This section deals with a special kind of word-oriented LFSR known as a σ-LFSR. Figure 3.4 depicts an L-stage σ-LFSR configuration with multi-input multi-output delay blocks and feedback gain elements. σ-LFSRs can be seen as a realisation of the multiple-recursive matrix method due to Niederreiter [Nie95]. A detailed description of σ-LFSR can be found in [LN86, ZHH07, Kri12].

The scalar feedback gains in a conventional LFSR are replaced by gain matrices B_0, B_1, ..., $B_{L-1} \in \mathbb{F}_q^{r\times r}$. The output of a σ-LFSR with L r-input r-output delay blocks is a sequence of vectors $\{\mathbf{s}\}_{i=0}^{\infty} = \{\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2, \ldots\}$ related by the following vector LRR:

$$\mathbf{s}_{i+L} = B_0\mathbf{s}_i + B_1\mathbf{s}_{i+1} + \ldots + B_{L-1}\mathbf{s}_{i+L-1} \tag{3.3}$$

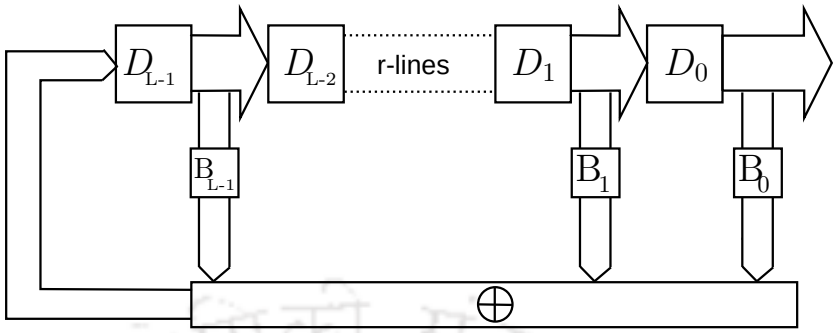

Figure 3.4: An L-stage σ-LFSR

where i=0,1,... and $\mathbf{s}_i \in \mathbb{F}_q^r$.

At a given time instant k, the state vector of a σ-LFSR, $\mathbf{s}(k)$, can be obtained by stacking the outputs of the delay blocks of the σ-LFSR one below the other. For instance,

$$\mathbf{s}(k) = \begin{bmatrix} \mathbf{s}_k \\ \mathbf{s}_{k+1} \\ \vdots \\ \mathbf{s}_{k+L-1} \end{bmatrix} \in \mathbb{F}_q^{rL}$$

The relation between two consecutive state vectors of a σ-LFSR is as follows:

$$\mathbf{s}(k+1) = A_{rL}\mathbf{s}(k), \quad \forall\ k = 0, 1, ...$$

where A_{rL} is given as follows.

$$A_{rL} = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{I} \\ B_0 & B_1 & B_2 & \dots & B_{L-1} \end{bmatrix} \in \mathbb{F}_q^{rL \times rL}$$

Here, $\mathbf{0} \in \mathbb{F}_q^{r\times r}$ is the zero matrix and $\mathbf{I} \in \mathbb{F}_q^{r\times r}$ is the identity matrix. The matrix A_{rL} is the state-transition matrix of the σ-LFSR. A_{rL} uniquely characterises the σ-LFSR. The characteristic polynomial of a σ-LFSR is the characteristic polynomial of the respective state-transition matrix. If the characteristic polynomial of a σ-LFSR is primitive then it is called a primitive σ-LFSR. For a primitive σ-LFSR, repeated multiplication of any nonzero vector by A_{rL} generates all non zero vectors in $\mathbb{F}_q^{rL}$. Therefore, the period of a nonzero vector sequence generated by a primitive σ-LFSR is $q^{rL} - 1$ which is the maximum possible period for any σ-LFSR with L r-input r-output delay blocks. Interestingly, unlike conventional LFSRs, for a given primitive characteristic polynomial we can have multiple feedback configurations for a σ-LFSR. An algorithm to design such feedback configurations is given in [KP12, Kri12].

Theorem 3.2.1. *(Theorem 6.3.2. [Kri12]) Given positive integers r and L and a primitive polynomial $p(x)$ of degree rL, the number of σ-LFSRs over $\mathbb{F}_q$ with L, r-input r-output delay blocks whose state-transition matrices have primitive characteristic polynomials is given by:*

$$q^{r(r-1)(L-1)} \frac{|GL(r, \mathbb{F}_q)|}{q^r - 1} \frac{\phi(q^{rL} - 1)}{rL}$$

where $GL(r, \mathbb{F}_q)$ denotes the general linear group of degree r over $\mathbb{F}_q$.

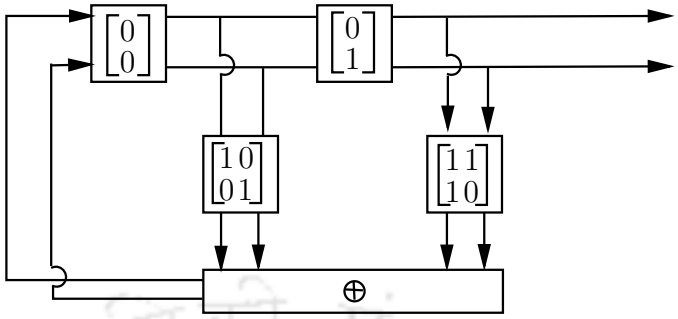

Figure 3.5: A typical example of a σ-LFSR

Example 3.2.1. *Figure 3.5 shows a typical σ-LFSR with two 2-input 2-output delay blocks, i.e., $L = 2$ and $r = 2$.*

Let the initial state vector be $\boldsymbol{s}(0) = \begin{bmatrix} \boldsymbol{s}_0 \\ \boldsymbol{s}_1 \end{bmatrix} \in \mathbb{F}_2^4$ *where* $\boldsymbol{s}_0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ *and* $\boldsymbol{s}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ *and feedback gain matrices are* $B_0 = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$ *and* $B_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.

The state transition matrix A_4 is given as follows.

$$A_4 = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ B_0 & B_1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

The characteristic polynomial of the LFSR is $p(x) = x^4 + x + 1$. $p(x)$ being primitive over $\mathbb{F}_2$ ensures that the output vector sequence is a periodic sequence with period 15. The output sequence generated by the σ-LFSR is:

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

A vector sequence generated by a σ-LFSR can be seen as an r-tuple of scalar sequences, each satisfying the LRR corresponding to the characteristic polynomial of the σ-LFSR. These scalar sequences are known as the component sequences of the output vector sequence. Note that, the LCM of minimal polynomials of the component sequences is the characteristic

polynomial of the σ-LFSR. Therefore, the σ-LFSR being primitive ensures that the component sequences are shifted versions of each other.

Consider a σ-LFSR with a primitive characteristic polynomial of degree rL. The state vector of each component sequence is defined as the set of rL consecutive bits of that sequence. For example, if $\{s_{i,j}\}_{j=0}^{\infty}$ is the i-th component sequence of the output sequence of a primitive σ-LFSR of degree rL, then the vector $x_i(k) = [s_{i,k}, s_{i,k+1}, \ldots, s_{i,k+rL-1}]$, is its state vector at the k-th time instant.

Lemma 3.2.2. *([LN86]) Consider a primitive σ-LFSR with L r-input r-output delay blocks. Let $x_1(k), x_2(k), \ldots, x_r(k)$ be the state vectors of its component sequences at time instant k. The vectors $x_1(k), x_1(k+1), \ldots, x_1(k+L-1)$,$x_2(k), x_2(k+1), \ldots, x_2(k+L-1)$,...,$x_r(k), x_r(k+1), \ldots, x_r(k+L-1)$ are linearly independent.*

Proof. Let A_{rL} be the state transition matrix of the σ-LFSR and $p(x)$ be its characteristic polynomial. Let $x(k)$ be its state vector at time instant k. Now, the minimal polynomial of $x(k)$ with respect to A_{rL} divides $p(x)$. However, since $p(x)$ is primitive (hence irreducible), the minimal polynomial of $x(k)$ with respect to A_{rL} is $p(x)$. Therefore, $x(k), x(k+1) = A_{rL}x(k), \ldots, x(k+rL-1) = A_{rL}^{rL-1}x(k)$ are linearly independent, i.e., the matrix $P = [x(k), x(k+1), \ldots, x(k+rL-1)] \in \mathbb{F}_q^{rL \times rL}$ has full rank. Observe that $P = [x_1(k); x_2(k); \ldots; x_r(k); x_1(k+1); x_2(k+1); \ldots; x_r(k+1); \ldots; x_1(k+L-1); x_2(k+L-1); \ldots; x_r(k+L-1)]$. Therefore, the vectors $x_1(k), x_1(k+1), \ldots, x_1(k+L-1)$,$x_2(k), x_2(k+1), \ldots, x_2(k+L-1)$,...,$x_r(k), x_r(k+1), \ldots, x_r(k+L-1)$ are linearly independent. □

In particular when $q = 2$, as a direct consequence of the above lemma we have the following result.

Corollary 3.2.2.1. *Consider a primitive σ-LFSR with L r-input r-output delay blocks. Let σ be a shift operator which shifts a sequence once to the right. If the component sequences of the LFSR are $\mathcal{S}, \sigma^{\gamma_1}\mathcal{S}, \sigma^{\gamma_2}\mathcal{S}, \ldots, \sigma^{\gamma_{r-1}}\mathcal{S}$, where $\gamma_1, \gamma_2, \ldots, \gamma_{r-1} > 0$, then*

1. $L - 1 < \gamma_i < 2^{rL} - L$ $(1 \leq i \leq r-1)$
2. $L - 1 < |\gamma_i - \gamma_j| < 2^{rL} - L$ $(1 \leq i \leq r-1)$

3.3 Summary

In this chapter, we have introduced LFSRs and a special kind of word-oriented LFSR known as a σ-LFSR.

Chapter 4

Nonlinear Feed-forward Generators

> "Life is really simple, but we insist on making it complicated."
>
> *Confucius*

This chapter deals with the use of nonlinear feed-forward logic along with LFSRs over arbitrary finite fields. This is a method of increasing the linear complexity (see Definition 3.1.1) of the generated sequences. Further, we analyse the statistical distribution of sequences generated by such scheme.

4.1 NLFGs

The linear complexity of a sequence generated by an LFSR can at most be equal to the number of delay blocks in that LFSR. Let the linear complexity of a binary sequence be n. The characteristic polynomial of its generating n-degree LFSR can be determined by observing merely $2n$ consecutive bits of the sequence [MOV96, PP09]. Basically, with the help of Equation 3.1 along with the first $2n$ bits, we can get that characteristic polynomial by solving a system of n linear equations with n unknowns. Thus, the next subsequent bits so that the entire sequence can be easily generated. This is

undesirable from a cryptographic point of view. One way of increasing the linear complexity of such sequences is by the use of nonlinear feedforward logic. Nonlinear feedforward logic is a multiplier assembly consisting of a set of 2-input multipliers (see Figure 4.1)[Gro71, Key76]. A sequence generator consisting of an LFSR and a nonlinear feed-forward logic is known as nonlinear feed-forward generator (NLFG). Figure 4.1 shows an NLFG consisting of 2-input multipliers.

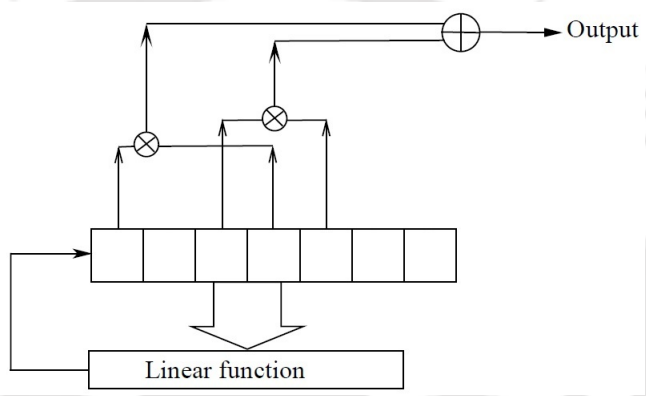

Figure 4.1: NLFG with an underlying LFSR

In the multiplier assembly, the output bits of some of the delay blocks are multiplied with each other and the resulting products are then added to generate the output sequence. The output of each delay block can act as an input to at most one multiplier. The input span of a multiplier is defined as the number of delay blocks between the two bits it multiplies. Multiplication and addition are as defined in $\mathbb{F}_q$. For $q = 2$, multiplication and addition translate to AND and XOR operations respectively. [Gro71] and [Key76] specifically deal with NLFGs over $\mathbb{F}_2$. An example of such a scheme is shown in Figure 4.1.

In [Key76], it has been shown that the linear complexity of sequences generated by an NLFG is at most $\sum_{i=1}^{k} \binom{L}{i}$, where L denotes the length of the LFSR and k is the algebraic degree of the nonlinear feed-forward

logic. Here, k is the maximum input cardinality of the multiplier blocks.

Remark 4.1.1. *(Key's Bound)*
Let $p(x)$ be a primitive polynomial of degree L over $\mathbb{F}_q$ and α be a root of $p(x)$. Any sequence, $\{a_i\}_{i=0}^{\infty} = (a_0, a_1, a_2 \ldots)$, having minimal polynomial $p(x)$ with period $q^L - 1$ can be written as $a_n = \sum_{i=0}^{L-1} A_i(\alpha^{q^i})^n$, where a_n denotes the member of the sequence at n-th instance and $A_i \in \mathbb{F}_q$. Multiplication of two such sequences generate a sequence whose n-th term can be written as follows,

$$b_n = a_{n,1}a^*_{n,2} = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} B_i\alpha^{(q^i+q^j)n}$$

where $B_i = A_{i,1}A_{j,2}$, $0 \leq i, j \leq L-1$. For binary sequences, when $q = 2$, the exponent of α becomes $2^i + 2^j$. There are L such cases when $i = j$ and there are $L(L-1)$ such cases (with each case repeated twice) when $i \neq j$. So, the total number of distinct powers of α present in the above expression is $L + \frac{L(L-1)}{2} = \frac{L(L+1)}{2}$. Therefore, for a particular sequence, if all the coefficients (B_i's) are nonzero then linear complexity of the sequence will be at most $\frac{L(L+1)}{2} = \sum_{i=1}^{2}\binom{L}{i}$. A similar argument can also be applied for the case when r sequences are multiplied. The upper bound of linear complexity of the product sequence is given by $\sum_{i=1}^{r}\binom{L}{i}$, r is the algebraic order of the nonlinearity [Key76].

In addition to the increase in linear complexity, it is important that the statistical properties of the generated sequence are close to those of a random sequence. It has been empirically observed that the statistical properties of a sequence generated using a nonlinear feedforward logic scheme are satisfactory if the span of each multiplier is distinct and the number of multipliers is maximised [Gro71]. One way of doing this is by using Langford sequences.

4.1.1 Langford sequence

Definition 4.1.1. *A Langford sequence of length $2n$ is a sequence of $2n$ positive integers in which every integer between 1 and n occurs twice and for each $i \leq n$, there are i elements of the sequence between the two occurrences of i.*

For example, for $n = 4$, $\{4, 1, 3, 1, 2, 4, 3, 2\}$ is a Langford sequence. Langford sequences exist only for $n \equiv 0$ *or* 3 $(mod\ 4)$ [Lan58]. The numbers of different Langford arrangements for $n = 1, 2, 3, \ldots$, counting any sequence as being the same as its reversal, are $0, 0, 1, 1, 0, 0, 26, 150, 0, 0, 17792, 108144, \ldots$ (Sequence A014552 in OEIS). However, no formula is known for the number of Langford sequences of a given order $n \equiv 0$ *or* 3 $(mod\ 4)$ till date.

A Langford sequence of length $2n$ can be used to design a nonlinear feedforward scheme with n multipliers for an LFSR having $L = 2n$ delay blocks as follows:

1. The delay blocks of the LFSR are labeled according to the Langford sequence. Thus, corresponding to each positive integer $i \leq n$ there exists a pair of delay blocks labeled by i having i delay blocks between them.

2. The outputs of delay blocks labeled by the same integer are given as inputs to the multipliers.

3. The output sequence is obtained by adding the outputs of the n multipliers.

For example, for an underlying primitive-LFSR, the nonlinear feedforward logic scheme corresponding to the Langford sequence $\{2, 3, 1, 2, 1, 3\}$ is as shown in Figure 4.2.

In the following section, we have discussed about nonlinear feed-forward generators that generate sequences over arbitrary finite fields. We have also analysed the frequency of symbols in sequences generated by such configurations. We assume that the underlying FSR generates a sequence wherein all nonzero states occur once in every period (as in a primitive-LFSR). Our arguments, however, do not require the FSR to be linear.

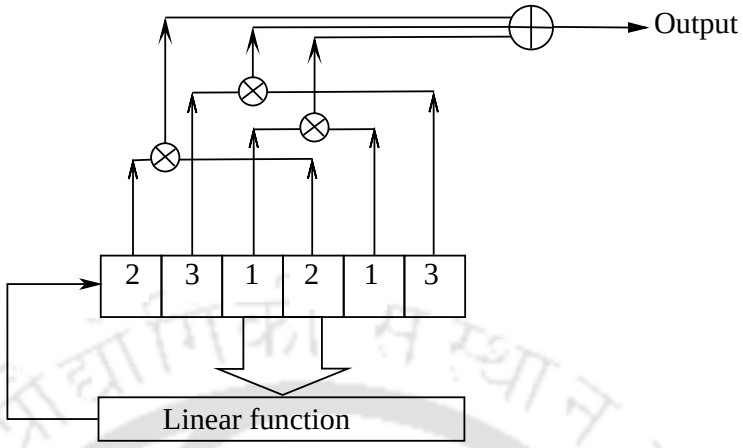

Figure 4.2: NLFG with a Langford arrangement

4.2 Analysis of the sequences generated by NLFGs

Consider an NLFG having an FSR with L delay blocks and a multiplier assembly with $m \leq \lfloor \frac{L}{2} \rfloor$ multipliers (Figure 4.3).

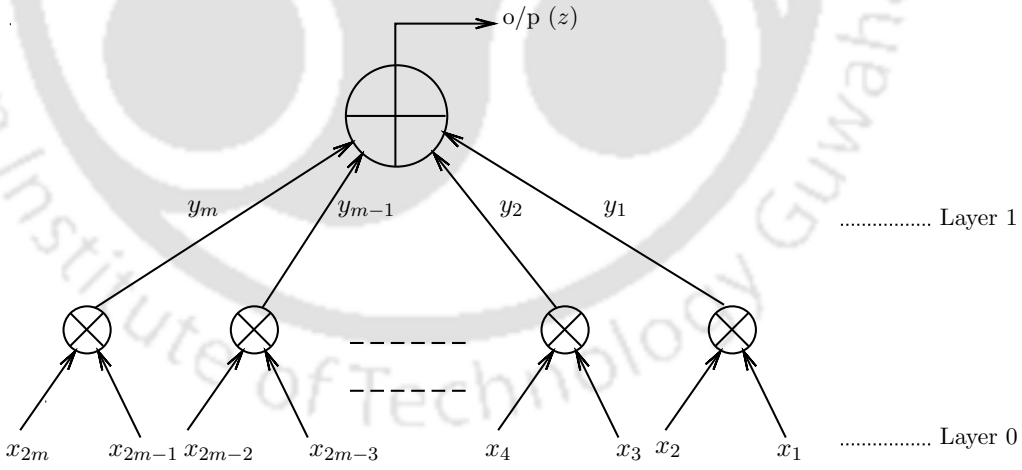

Figure 4.3: Multiplier assembly of an NLFG with m multipliers

Let $\psi_m(K)$ denote the number of possible inputs to the multiplier assembly that generate the number K at the output. When $m = 1$, the

output of the multiplier will be 0 if either of its inputs are zero. Thus,

$$\psi_1(0) = 2q - 1 \ . \tag{4.1}$$

Lemma 4.2.1. *$\psi_1(K) = (q-1)$, for all $K \in \mathbb{F}_q \backslash \{0\}$.*

Proof. Given any $K_1 \in \mathbb{F}_q \backslash \{0\}$, there exists a unique $K_2 \in \mathbb{F}_q \backslash \{0\}$ such that $K_1.K_2 = K$. Since there are $q-1$ possible values for K_1, $\psi_1(K) = (q-1)$. □

Equation 4.1 and Lemma 4.2.1 show that $\psi_1(K)$ does not depend upon the value of K but only on whether K is zero or nonzero. Therefore, in the remainder of the paper we denote $\psi_1(K)$ by ψ_{nz} when $K \neq 0$ and by ψ_z when $K = 0$.

Let $\mathcal{N}_m^L(K)$ be the number of nonzero state vectors of the underlying LFSR that generate K at the output. Each of the $q^{2m} - 1$ nonzero inputs to the multiplier assembly occurs q^{L-2m} times. Therefore,

$$\mathcal{N}_m^L(K) = \begin{cases} q^{L-2m}.\psi_m(K), & \text{when } K \neq 0 \\ q^{L-2m}.\psi_m(0) - 1, & \text{when } K = 0 \end{cases} \tag{4.2}$$

In the expression for $\mathcal{N}_m^L(0)$, one is deducted to account for the absence of the zero state. Thus, deriving an expression for $\mathcal{N}_m^L(\cdot)$ reduces to finding a formula for $\psi_m(\cdot)$.

Definition 4.2.1. *An m partition of K over $\mathbb{F}_q$ is defined as an m-tuple of nonzero elements in $\mathbb{F}_q$ whose sum (as defined in $\mathbb{F}_q$) is K. We denote the set of m-partitions of K by $\mathcal{S}_m(K)$.*

$$\mathcal{S}_m(K) := \left\{ \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_m \end{bmatrix} \in \mathbb{F}_q^m \mid \sum_{i=1}^{m} y_i = K \text{ and } y_i \neq 0 \right\} .$$

where $i = 1, 2, \ldots, m$.

Clearly, $|\mathcal{S}_0(K)| = 0$ *and* $|\mathcal{S}_1(K)| = 1$. For $m > 1$, $|\mathcal{S}_m(K)|$ can be recursively calculated as follows.

Lemma 4.2.2. $|\mathcal{S}_m(K)| = (q-1)^{m-1} - |\mathcal{S}_{m-1}(K)|$ *where* $K \in \mathbb{F}_q \backslash \{0\}$.

Proof. We can arbitrarily choose $m-1$ nonzero elements from $\mathbb{F}_q$ in $(q-1)^{m-1}$ possible ways. If the sum of these $m-1$ elements is not equal to K then there exists a unique nonzero element in $\mathbb{F}_q$ which gives K when added with this sum. If the sum of these $m-1$ elements is equal to K then this $(m-1)$-tuple is a member of the set $\mathcal{S}_{m-1}(K)$. Hence, $|\mathcal{S}_m(K)| = (q-1)^{m-1} - |\mathcal{S}_{m-1}(K)|$. □

Using the above recursion, the closed-form expression for $|\mathcal{S}_m(K)|$ is derived as follows.

Lemma 4.2.3. $|\mathcal{S}_m(K)| = \frac{1}{q}.\{(q-1)^m - (-1)^m\}$, *where* $K \in \mathbb{F}_q \backslash \{0\}$.

Proof. We shall prove the lemma using induction.
Now, $|\mathcal{S}_1(K)| = 1 = \frac{1}{q}(q-1+1)$. Thus, the statement of the lemma is true for $m = 1$.

Let the statement be true for $m = l$, i.e., $|\mathcal{S}_l(K)| = \frac{1}{q}.\{(q-1)^l - (-1)^l\}$. We now proceed to prove that the statement is true for $m = l+1$.

$$\begin{aligned}
|\mathcal{S}_{l+1}(K)| &= (q-1)^l - |\mathcal{S}_l(K)| \quad [\textit{using lemma } 4.2.2] \\
&= (q-1)^l - \frac{1}{q}\{(q-1)^l - (-1)^l\} \\
&= (q-1)^l - \frac{(q-1)^l}{q} + \frac{(-1)^l}{q} \\
&= (q-1)^l \left[1 - \frac{1}{q}\right] + \frac{(-1)^l}{q} \\
&= \frac{(q-1)^{l+1}}{q} + \frac{(-1)^l}{q} \\
&= \frac{1}{q}\{(q-1)^{l+1} - (-1)^{l+1}\}
\end{aligned}$$

□

Since $\psi_{nz} = q-1$, we can write $|\mathcal{S}_m(K)|$ as follows.

$$|\mathcal{S}_m(K)| = \frac{1}{q}.\{\psi_{nz}^m - (-1)^m\}, \textit{ where } K \in \mathbb{F}_q \backslash \{0\} \tag{4.3}$$

Assume that at a particular time instant, the outputs of i of the m multipliers are zero. These i multipliers can be chosen in $\binom{m}{i}$ ways. Each of these multipliers can have ψ_z possible pairs of inputs. Now, there are $|\mathcal{S}_{m-i}(K)|$ possible sets of outputs from the remaining $m-i$ multipliers such that the output of the adder is K. For each such set each multiplier can have ψ_{nz} possible pairs of inputs. Therefore,

$$\psi_m(K) = \sum_{i=0}^{m-1} \binom{m}{i} \psi_z^i \psi_{nz}^{m-i} |\mathcal{S}_{m-i}(K)|, \ \textit{where } K \in \mathbb{F}_q \backslash \{0\}. \tag{4.4}$$

Now, we simplify the above above formula to derive a closed form expression for $\psi_m(\cdot)$.

Theorem 4.2.4. *For a multiplier assembly with m multipliers and for all $K \in \mathbb{F}_q$.*

$$\psi_m(K) = \begin{cases} q^{m-1}(q^m - 1), & \textit{when } K \neq 0 \\ q^{m-1}(q^m + q - 1), & \textit{when } K = 0 \end{cases}$$

Proof. Let $K \neq 0$. Substituting the formula for $|\mathcal{S}_{m-i}(K)|$ from Equation 4.3 in Equation 4.4 we get -

$$\begin{aligned} \psi_m(K) &= \frac{1}{q} \sum_{i=0}^{m-1} \binom{m}{i} \psi_z^i \psi_{nz}^{m-i} \{\psi_{nz}^{m-i} - (-1)^{m-i}\} \\ &= \frac{1}{q} \left\{ \sum_{i=0}^{m-1} \binom{m}{i} \psi_z^i \psi_{nz}^{2(m-i)} - \sum_{i=0}^{m-1} \binom{m}{i} \psi_z^i (-\psi_{nz})^{(m-i)} \right\} \end{aligned}$$

Now, $\sum_{i=0}^{m-1} \binom{m}{i} \psi_z^i \psi_{nz}^{2(m-i)} = \sum_{i=0}^{m} \binom{m}{i} \psi_z^i \psi_{nz}^{2(m-i)} - \psi_z^m$ and $\sum_{i=0}^{m-1} \binom{m}{i} \psi_z^i (-\psi_{nz})^{(m-i)} = \sum_{i=0}^{m} \binom{m}{i} \psi_z^i (-\psi_{nz})^{(m-i)} - \psi_z^m$.

Therefore,

$$\begin{aligned}\psi_m(K) &= \frac{1}{q}\Bigg[\Bigg\{\sum_{i=0}^{m}\binom{m}{i}\psi_z^i\psi_{nz}^{2(m-i)} - \psi_z^m\Bigg\} \\ &\quad - \Bigg\{\sum_{i=0}^{m}\binom{m}{i}\psi_z^i(-\psi_{nz})^{(m-i)} - \psi_z^m\Bigg\}\Bigg] \\ &= \frac{1}{q}\{(\psi_z+\psi_{nz}^2)^m - (\psi_z-\psi_{nz})^m\}\end{aligned}$$

Substituting the values of ψ_z and ψ_{nz} from Equation 4.1 and Lemma 4.2.1 we get -

$$\begin{aligned}\psi_m(K) &= \frac{1}{q}\Big[\{(2q-1)+(q-1)^2\}^m - \{(2q-1)-(q-1)\}^m\Big] \\ &= \frac{1}{q}(q^{2m}-q^m) = \frac{q^m}{q}(q^m-1) \\ &= q^{m-1}(q^m-1)\end{aligned}$$

Since there are $(q-1)$ nonzero elements in $\mathbb{F}_q$, there are $(q-1)q^{m-1}(q^m-1)$ input combinations that generate a nonzero output from the NLFG. Therefore,

$$\begin{aligned}\psi_m(0) &= q^{2m} - (q-1).q^{m-1}(q^m-1) \\ &= q^{2m-1} + q^m - q^{m-1} \\ &= q^{m-1}(q^m+q-1)\end{aligned}$$

This concludes the proof of our theorem. □

Substituting the formula for $\psi_m(\cdot)$ derived in Theorem 4.2.4 in Equation 4.2 we get -

Corollary 4.2.4.1.

$$\mathcal{N}_m^L(K) = \begin{cases} q^{L-m-1}(q^m-1) & \text{where } K \neq 0. \\ q^{L-m-1}(q^m+q-1)-1 & \text{where } K = 0. \end{cases}$$

Remark 4.2.1. *It can be easily verified that* $\mathcal{N}_m^L(0) + (q-1)\mathcal{N}_m^L(K) = q^L - 1$.

Remark 4.2.2. *The Theorem 3 in [DAG90] is a special case of the Theorem 4.2.4 where* $q = 2$.

For $K \neq 0$, the ratio $\xi(L,m,q) := \frac{\mathcal{N}_m^L(0)}{\mathcal{N}_m^L(K)}$ is an indicator of how balanced a sequence is. For a completely balanced sequence $\xi(L,m,q) = 1$. We now go on to show that the distribution of elements in the output sequence of an NLFG tends to a balance distribution as the number of delay blocks and the number of multipliers tends to infinity.

Corollary 4.2.4.2.

$$\lim_{m\to\infty} \frac{\mathcal{N}_m^L(K)}{q^L - 1} = \frac{1}{q}, \quad \textit{where } K \in \mathbb{F}_q.$$

Proof. In the case, when $K \neq 0$ then -

$$\lim_{m\to\infty} \frac{\mathcal{N}_m^L(K)}{q^L - 1} = \lim_{m\to\infty} \frac{q^{L-m-1}(q^m - 1)}{q^L - 1} = \frac{1}{q}.$$

If $K = 0$ then -

$$\begin{aligned}
\lim_{m\to\infty} \frac{\mathcal{N}_m^L(0)}{q^L - 1} &= \lim_{m\to\infty} \frac{q^{L-m-1}(q^m + q - 1) - 1}{q^L - 1} \\
&= \lim_{m\to\infty} \frac{q^L}{(q^L - 1)} . \frac{q^{L-1} + q^{L-m} - q^{L-m-1} - 1}{q^L} \\
&= \lim_{m\to\infty} \frac{q^L}{(q^L - 1)} . \lim_{m\to\infty} \left[\frac{1}{q} + \frac{1}{q^m} - \frac{1}{q^{m+1}} - \frac{1}{q^L}\right] \\
&= \frac{1}{q}
\end{aligned}$$

□

Remark 4.2.3. *For any nonzero K, $\xi(L,m,q) = 1 + \frac{q}{q^m - 1} - \frac{1}{\mathcal{N}_m^L(K)} \approx 1 + q^{-(m-1)}$. Thus, the ratio of the number of occurrences of the zero vector to that of any nonzero vector in a single period approaches 1 at a rate which is exponential in m. The following table shows the values for $\xi(L,m,q)$ for different values of L, m, and q.*

(L,m,q)	(16,8,2)	(32,16,2)	(64,32,2)	(32,16,3)
ξ	1.01	1.00003	1.0000000003	1.00000007

4.3 A generalisation

In this section, we have generalised the results derived in section 4.2. Here, we have considered NLFG configurations where the multipliers can have more than 2 inputs. Consider an NLFG with L delay blocks with a multiplier assembly consisting of m ℓ-input multipliers. Clearly, $m \leq \lfloor \frac{L}{\ell} \rfloor$. Let $\widehat{\psi}_{m,\ell}(K)$ and $\widehat{\mathcal{N}}^L_{m,\ell}(K)$ be the number of possible inputs to the multiplier assembly that generate the number $K \in \mathbb{F}_q$ at the output and the number of nonzero state vectors of the underlying LFSR that generate K at the output, respectively. Similar to Equation 4.1 and Lemma 4.2.1, observe that, for an ℓ-input multiplier.

- The number of possible inputs to a multiplier that results in a particular nonzero number $K \in \mathbb{F}_q$ at its output is

$$\widehat{\psi}_{nz,\ell} = \widehat{\psi}_{1,\ell}(K) = (q-1)^{(\ell-1)} \tag{4.5}$$

and

- The number of possible inputs to a multiplier that generates zero at its output is

$$\widehat{\psi}_{z,\ell} = \widehat{\psi}_{1,\ell}(0) = q^{\ell} - (q-1)^{\ell} \tag{4.6}$$

Similar to Equation 4.2,

$$\widehat{\mathcal{N}}^L_{m,\ell}(K) = \begin{cases} q^{L-\ell m}.\widehat{\psi}_{m,\ell}(K), & \text{when } K \neq 0 \\ q^{L-\ell m}.\widehat{\psi}_{m,\ell}(0) - 1, & \text{when } K = 0 \end{cases} \tag{4.7}$$

Let $K \in \mathbb{F}_q \setminus \{0\}$. Observe that, for a multiplier assembly with m ℓ-input multipliers, Equation 4.4 transforms to the following.

$$\widehat{\psi}_{m,\ell}(K) = \sum_{i=0}^{m-1} \binom{m}{i} \widehat{\psi}^{i}_{z,\ell}.\widehat{\psi}^{m-i}_{nz,\ell}.|\mathcal{S}_{m-i}(K)| \tag{4.8}$$

Substituting the values of $\widehat{\psi}_{z,\ell}$, $\widehat{\psi}_{nz,\ell}$, and $\mathcal{S}_{m-i}(K)$ we get the following.

$$\begin{aligned}
\widehat{\psi}_{m,\ell}(K) &= \frac{1}{q}\sum_{i=0}^{m-1}\binom{m}{i}\left(q^{\ell}-(q-1)^{\ell}\right)^{i}.(q-1)^{(\ell-1)(m-i)}.\left\{(q-1)^{m-i}-(-1)^{m-i}\right\} \\
&= \frac{1}{q}\Bigg[\left\{\sum_{i=0}^{m-1}\binom{m}{i}.\left(q^{\ell}-(q-1)^{\ell}\right)^{i}.(q-1)^{\ell(m-i)}\right. \\
&\quad \left.-\sum_{i=0}^{m-1}\binom{m}{i}.\left(q^{\ell}-(q-1)^{\ell}\right)^{i}.(-(q-1))^{\ell(m-i)}\right\}\Bigg]
\end{aligned}$$

By adding the term, $(q^{\ell}-(q-1)^{\ell-1})^{m}$, to both the summations, we get

$$\begin{aligned}
\widehat{\psi}_{m,\ell}(K) &= \frac{1}{q}\Bigg[\left\{\sum_{i=0}^{m-1}\binom{m}{i}.\left(q^{\ell}-(q-1)^{\ell}\right)^{i}.(q-1)^{\ell(m-i)}+(q^{\ell}-(q-1)^{\ell-1})^{m}\right\} \\
&\quad -\left\{\sum_{i=0}^{m-1}\binom{m}{i}.\left(q^{\ell}-(q-1)^{l}\right)^{i}.(-(q-1))^{\ell(m-i)}+(q^{\ell}-(q-1)^{\ell-1})^{m}\right\}\Bigg] \\
&= \frac{1}{q}\Bigg[\left\{\sum_{i=0}^{m}\binom{m}{i}.\left(q^{\ell}-(q-1)^{\ell}\right)^{i}.(q-1)^{\ell(m-i)}\right\} \\
&\quad -\left\{\sum_{i=0}^{m}\binom{m}{i}.\left(q^{\ell}-(q-1)^{l}\right)^{i}.(-(q-1))^{\ell(m-i)}\right\}\Bigg]
\end{aligned}$$

Therefore, by applying binomial theorem

$$\begin{aligned}
\widehat{\psi}_{m,\ell}(K) &= \frac{1}{q}\left\{\left(q^{\ell}-(q-1)^{\ell}+(q-1)^{\ell}\right)^{m}-\left(q^{\ell}-(q-1)^{\ell}-(q-1)^{\ell-1}\right)^{m}\right\} \\
&= \frac{1}{q}\left(q^{\ell m}-q^{m}(q^{\ell-1}-(q-1)^{\ell-1})^{m}\right) \\
&= \left(q^{\ell m-1}-q^{m-1}(q^{\ell-1}-(q-1)^{\ell-1})^{m}\right) \\
&= q^{m-1}\left(q^{m(\ell-1)}-(q^{\ell-1}-(q-1)^{\ell-1})^{m}\right)
\end{aligned}$$

Since there are $(q-1)$ nonzero elements in $\mathbb{F}_q$, there are $(q-1).\widehat{\psi}_{m,\ell}(K)$ input combinations that generate a nonzero output. Thus, for any $K \in \mathbb{F}_q \setminus \{0\}$

$$\begin{aligned}\widehat{\psi}_{m,\ell}(0) &= q^{\ell m} - (q-1)\widehat{\psi}_{m,\ell}(K) \\ &= q^{\ell m} - \Bigg[\Big\{q^{\ell m} - q^m(q^{\ell-1} - (q-1)^{\ell-1})^m\Big\} \\ &\quad - q^{\ell m-1} + q^{m-1}(q^{\ell-1} - (q-1)^{\ell-1})^m\Bigg]\end{aligned}$$

$$\begin{aligned}&= q^{\ell m} - \Bigg[q^{\ell m} - q^m(q^{\ell-1} - (q-1)^{\ell-1})^m \\ &- q^{\ell m-1} + q^{m-1}(q^{\ell-1} - (q-1)^{\ell-1})^m\Bigg] \\ &= q^m(q^{\ell-1} - (q-1)^{\ell-1})^m + q^{\ell m-1} - q^{m-1}(q^{\ell-1} - (q-1)^{\ell-1})^m \\ &= (q^{\ell-1} - (q-1)^{\ell-1})^m(q^m - q^{m-1}) + q^{\ell m-1} \\ &= q^{m-1}\Big(q^{m(\ell-1)} + (q-1)(q^{\ell-1} - (q-1)^{\ell-1})^m\Big)\end{aligned}$$

Theorem 4.3.1. *For a multiplier assembly with m ℓ-bit multipliers and for all $K \in \mathbb{F}_q$.*

$$\widehat{\psi}_{m,\ell}(K) = \begin{cases} q^{m-1}\Big(q^{m(\ell-1)} - (q^{\ell-1} - (q-1)^{\ell-1})^m\Big), & \text{when } K \neq 0 \\ q^{m-1}\Big(q^{m(\ell-1)} + (q-1)(q^{\ell-1} - (q-1)^{\ell-1})^m\Big), & \text{when } K = 0 \end{cases}$$

Using Theorem 4.3.1 and Equation 4.7, we can calculate $\widehat{\mathcal{N}}^L_{m,\ell}(K)$.

Remark 4.3.1. *It can easily be verified that for $\ell = 2$, these results translate to the results derived in section 4.2.*

Remark 4.3.2. *As in the case of NLFG with 2-input multipliers, it can be easily shown that, at the output sequences, here also the ratio $\widehat{\xi}(L, m, \ell, q) := \frac{\widehat{\mathcal{N}}^L_{m,\ell}(0)}{\widehat{\mathcal{N}}^L_{m,\ell}(K)}$ tends to 1 as $m \to \infty$ for any $K \in \mathbb{F}_q \setminus \{0\}$.*

4.4 Generating a balanced sequence

In this section, we show how an NLFG can be used to generate a sequence that satisfies the balance property.

Consider an NLFG with L delay blocks, where the i-th delay block is not connected to the multiplier assembly. We claim that if the output of this delay block is added to the output of the NLFG, the resulting sequence will satisfy the balance property (see Figure 4.4).

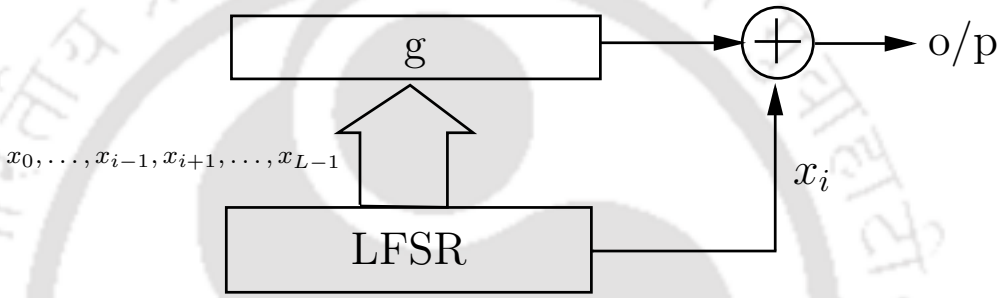

Figure 4.4: Balanaced NLFG

Let $x_0, x_1, \ldots, x_{L-1}$ denote the outputs of the corresponding delay blocks. Consequently, the multiplier assembly implements a polynomial $g(x_0, x_1, \ldots x_{i-1}, x_{i+1}, \ldots x_{L-1})$. Therefore, adding the output of the i-th delay block to the multiplier assembly, implements the following polynomial.

$$h(x_0, x_1, \ldots, x_{L-1}) = x_i + g(x_0, x_1, \ldots x_{i-1}, x_{i+1}, \ldots x_{L-1}) \tag{4.9}$$

For each choice of $(x_0, x_1, \ldots x_{i-1}, x_{i+1}, \ldots x_{L-1})$, $h(x_0, x_1, \ldots, x_{L-1})$ takes q values, each corresponding to a different value of x_i. Since the LFSR is primitive, each non zero vector in $\mathbb{F}_q^L$ occurs exactly once as a state vector in every period. Thus, each of the $q^{L-1} - 1$ non zero choices of $(x_0, x_1, \ldots x_{i-1}, x_{i+1}, \ldots x_{L-1})$ occur q times in a period. Each such instance will result in a different value of $h(x_0, x_1, \ldots, x_{L-1})$. These state vectors account for $q^{L-1}-1$ occurrences of each element of $\mathbb{F}_q$ in a period of

the output sequence. Further, there are $q-1$ non zero state vectors where $(x_0, x_1, \ldots x_{i-1}, x_{i+1}, \ldots x_{L-1})$ are all zero. For each such state vector, the value of x_i, and hence $h(x_0, x_1, \ldots, x_{L-1})$, is a distinct nonzero element of $\mathbb{F}_q$. This accounts for another occurrence of each non zero element of $\mathbb{F}_q$. As a consequence, in a single period, each non zero element of $\mathbb{F}_q$ occurs q^{L-1} times and zero occurs $q^{L-1}-1$ times. Therefore, the output sequence is balanced.

4.5 Summary

In this Chapter, we have extended the notion of NLFGs to arbitrary finite fields and have analysed the balance property of the sequences generated by such NLFGs.

Chapter 5

σ-LFSR based NLFGs

"May be linearity is beauty but non-linearity is handsome."

Anonymous

In this chapter, we discuss methods by which nonlinear feed-forward logic can be extended to σ-LFSRs. First, we discuss an existing scheme and analyse the statistical distribution of the sequences generated by that scheme. We then go on to propose two novel schemes for σ-LFSR based NLFGs. Further, we compare the statistical distribution of the sequences generated by the proposed schemes with the scheme available in literature [HPW12].

5.1 NLFG over word-based LFSR

The vector sequences generated by a primitive σ-LFSR, having L r-input r-output delay blocks, have some good statistical properties. However, the linear complexity of such sequences can at most be rL. Word-based NLFG configurations analogous to the one suggested in Chapter 4 can be used to increase the linear complexity of these sequences. Here, the multiplication of the outputs of delay blocks is done bit-wise. A schematic diagram of the σ-LFSR based NLFG is given in Figure 5.1. In spite of

having high linear complexity, these generated sequences are unbalanced (with the higher number of zeros).

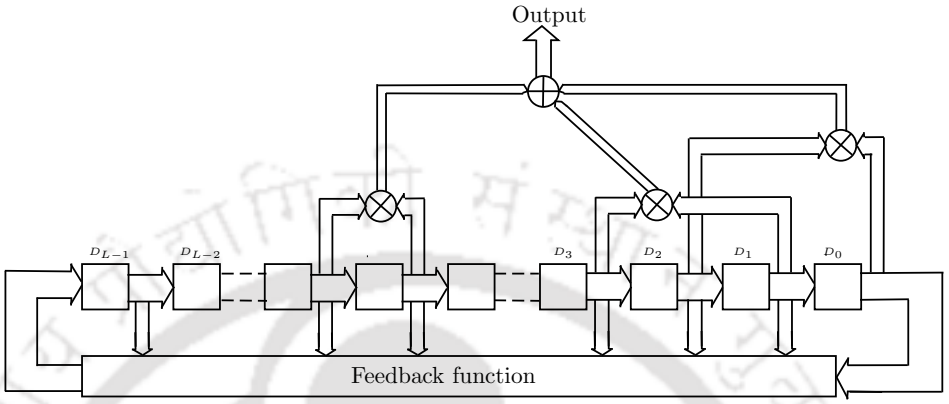

Figure 5.1: Nonlinear feedforward generator based on σ-LFSR

We now analyse the statistical distribution of vector sequences generated by the scheme given in [HPW12]. Although the scheme deals only with the binary case, in our analysis we consider the NLFG to be over an arbitrary finite field $\mathbb{F}_q$. The outputs of the delay blocks are multiplied bit-wise as shown in Figure 5.2.

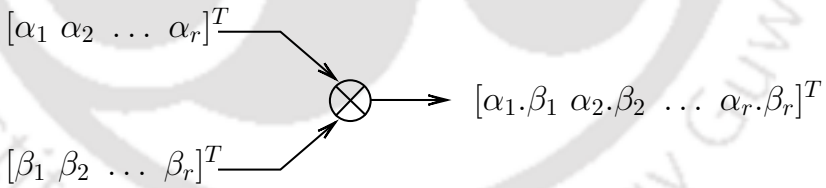

Figure 5.2: Element-wise multiplication operation

Note that, the bit-wise multiplication is not equivalent to multiplication over a finite field. For example, in $\mathbb{F}_3^4$ the bit-wise product of two nonzero vectors $v_1 = [0\ 1\ 2\ 0]^T,\ v_2 = [2\ 0\ 0\ 1]^T$ is zero which is not possible over a finite field.

Theorem 5.1.1. *Consider an element-wise NLFG having L r-input r-output delay blocks and $m \leq \lfloor \frac{L}{2} \rfloor$ multipliers. For a given nonzero vector*

$\mathbf{v} \in \mathbb{F}_q^r$, *the number,* $\mathbf{\Psi_m(v)}$, *of inputs to the multiplier assembly that generate* $\mathbf{v}$ *at the output is given by*

$$\mathbf{\Psi_m(v)} = (q^{m-1})^r (q^m - 1)^{\kappa_v} (q^m + q - 1)^{r-\kappa_v}$$

where κ_v *is the number of nonzero elements in* $\mathbf{v}$.

Proof. Since addition and multiplication are performed element-wise, the i-th entry v_i of the output vector sequence is a function of only the i-th outputs of the delay blocks of the σ-LFSR. Further, from Lemma 1 in [Nie95], it can be inferred that each component sequence of a primitive σ-LFSR can be seen to be generated by a scalar LFSR whose characteristic polynomial is the same as that of the σ-LFSR. Therefore, the i-th bit of the output sequence of the NLFG can be generated by a scalar NLFG with a primitive scalar LFSR having rL delay blocks and a multiplier assembly with m multipliers. From Theorem 4.2.4, the number of inputs to this multiplier assembly that generates a vector $\mathbf{v}$ with v_i at the output is given by

$$\psi_m(v_i) = \begin{cases} q^{m-1}(q^m - 1) & when\ v_i \neq 0. \\ q^{m-1}(q^m + q - 1) & when\ v_i = 0. \end{cases}$$

Therefore, the total number of possible inputs to the multiplier assembly that generates a given vector v having κ_v nonzero elements is given by

$$\begin{aligned} \mathbf{\Psi_m(v)} &= \left\{q^{m-1}(q^m - 1)\right\}^{\kappa_v} \left\{q^{m-1}(q^m + q - 1)\right\}^{r-\kappa_v} \\ &= (q^{m-1})^r (q^m - 1)^{\kappa_v} (q^m + q - 1)^{r-\kappa_v} \end{aligned}$$

□

Remark 5.1.1. *Clearly, in the case when* $r = 1, \kappa_v = 1$ *and* $r = 1, \kappa_v = 0$, *Theorem 5.1.1 translates to Theorem 4.2.4 .*

For an NLFG having L r-input r-output delay blocks and $m \leq \lfloor L/2 \rfloor$ multipliers, let $\mathfrak{N}_m^L(v)$ denote the number of times the vector $v \in \mathbb{F}_q^r$ occurs at the output of the NLFG in a single cycle.

Corollary 5.1.1.1.

$$\mathfrak{N}_m^L(v) = \begin{cases} q^{r(L-m-1)}(q^m - 1)^{\kappa_v}(q^m + q - 1)^{r-\kappa_v} & , v \neq 0. \\ q^{r(L-m-1)}(q^m + q - 1)^r - 1 & , v = 0. \end{cases}$$

Proof. Since every nonzero state vector occurs exactly once in every period of the underlying primitive σ-LFSR, $\mathfrak{N}_m^L(v)$ is equal to the number of nonzero states of the σ-LFSR that generate $v \in \mathbb{F}_q^r$ at the output of the NLFG. Clearly, for each input to the multiplier assembly there are q^{L-2m} possible state vectors of the σ-LFSR (since $L-2m$ of the delay blocks are not connected to the multiplier assembly). Therefore, the number of times a nonzero vector v occurs at the output of the NLFG in a single period is equal to $q^{r(L-2m)}\mathbf{\Psi_m(v)}$. Now, among the states of the σ-LFSR that result in zero at the output of the NLFG is the zero state. However, this state does not occur in any nonzero cycle. Therefore, the number of times the zero vector occurs at the output of the NLFG in a single period is equal to $q^{r(L-2m)}\mathbf{\Psi_m(0)} - 1$. Thus,

$$\mathfrak{N}_m^L(v) = \begin{cases} q^{r(L-2m)}\mathbf{\Psi_m(v)} & when\ v \neq 0. \\ q^{r(L-2m)}\mathbf{\Psi_m(0)} - 1 & when\ v = 0. \end{cases}$$

Substituting the value of $\Psi_m(v)$ from Theorem 5.1.1, we get-

$$\mathfrak{N}_m^L(v) = \begin{cases} q^{r(L-m-1)}(q^m-1)^{\kappa_v}(q^m+q-1)^{r-\kappa_v} & , v \neq 0. \\ q^{r(L-m-1)}(q^m+q-1)^r - 1 & , v = 0. \end{cases} \tag{5.1}$$

□

This result shows that the output sequences generated by the scheme given in [HPW12] have a disproportionately high number of zeros.

5.2 NLFGs with permutation matrices

In this section, we propose an alternate nonlinear feedforward scheme for primitive σ-LFSRs using Langford sequences. As we have already seen, each component sequence generated by the σ-LFSR has a primitive characteristic polynomial with degree rL. However, the number of multipliers per component sequence in the scheme described in the previous section is at most $\frac{L}{2}$. The aim of the proposed scheme is to increase this number and thereby make each component sequence more balanced. Further, to ensure good statistical properties, we attempt to keep the spans of these multipliers distinct.

Consider a σ-LFSR with L r-input r-output delay blocks. Let L' be the largest integer less than or equal to L such that $\frac{L'}{2} \equiv 0 \text{ or } 3 \ (mod\ 4)$. The outputs of the first L' blocks of the σ-LFSR are the inputs to the multiplier assembly. We choose r Langford sequences of length L', one for each component sequence. The output of each delay block is multiplied with a permutation matrix P. For generating the i-th output component sequence, the delay blocks are first labelled according to the i-th Langford sequence. Then, the output of the delay block corresponding to the first occurrence of an integer in the Langford sequence is bit-wise multiplied with the permuted output of the delay block corresponding to the second occurrence of the same integer. The output of these vector multipliers are then added to generate a vector whose entries are then added to obtain a scalar sequence. This scalar sequence constitutes the i-th component sequence of the output vector sequence. For a σ-LFSR with six delay blocks the scalar sequence corresponding to the Langford sequence 312132 is generated as shown in Figure 5.3. Here the symbol $\sum$ denotes the addition of the elements of a vector.

At any instant of time, the outputs of the delay blocks give us L consecutive bits of each component sequence of the LFSR. As we have seen in chapter 3, the characteristic polynomial of the σ-LFSR being primitive ensures that the component sequences are shifted versions of the each other. This means that at any given instant of time we have r disjoint blocks of L consecutive bits of a scalar sequence $\mathcal{S}$ whose minimal polynomial is equal to the characteristic polynomial of the σ-LFSR. The fact that they are disjoint follows from Corollary 3.2.2.1. Consider an operator σ which shifts a sequence once to the left. Therefore, if the first component sequence generated by a σ-LFSR is $\mathcal{S} = \{s_i\}_{i=0}^{\infty}$, then there exist positive integers $\gamma_1, \gamma_2 \ldots \gamma_{r-1}$ such that the i-th component

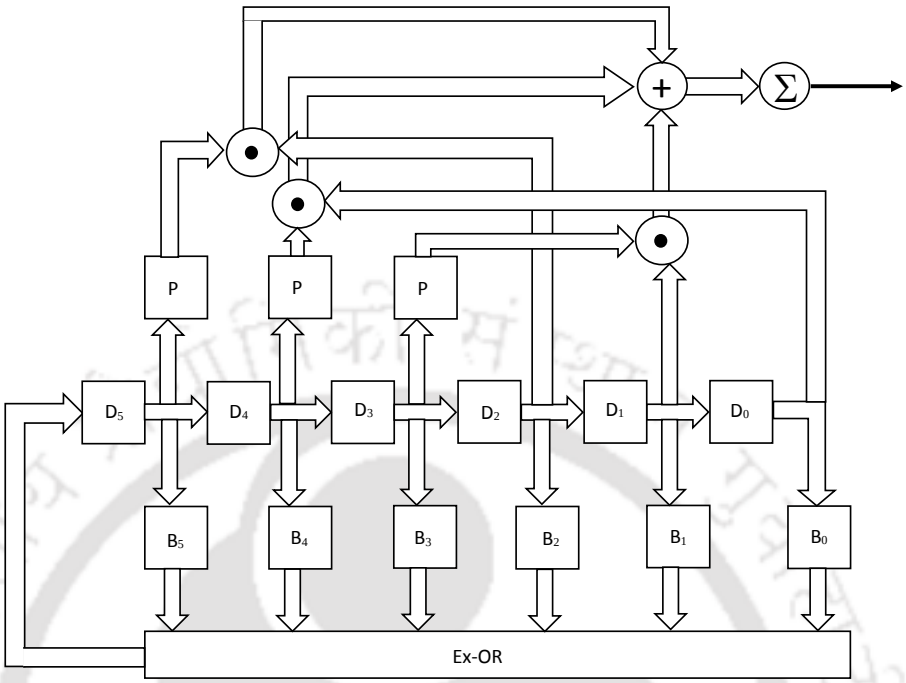

Figure 5.3: Generation of a component sequence

sequence is $\sigma^{\gamma_{i-1}}\mathcal{S}$. Thus, at time instant k, the blocks available to us are: $(s_k, s_{k+1}, \ldots, s_{k+L-1}), (s_{k+\gamma_1}, s_{k+\gamma_1+1}, \ldots, s_{k+\gamma_1+L-1}), \ldots (s_{k+\gamma_{r-1}}, s_{k+\gamma_{r-1}+1}, \ldots, s_{k+\gamma_{r-1}+L-1})$. The permutation matrix, P, determines the way in which the elements of these blocks are multiplied with one another. For example, if P is an identity matrix, the multipliers multiply the elements of the same block with each other. The span of a multiplier is defined as a vector, each entry of which gives us the minimum number of bits in the sequence $\mathcal{S}$ between the two bits that are multiplied. For example, consider a σ-LFSR with six 2-input 2-output delay blocks (Figure 5.3). Let the permutation matrix be $P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ and the component sequences be $\mathcal{S}$ and $\sigma^{\gamma_1}\mathcal{S}$. Consider a multiplier that multiplies the output of the first delay block with the permuted output of the fourth delay block. The first element of the multiplier multiplies s_k with $s_{k+\gamma_1+3}$ and the second

multiplier multiplies s_{k+3} with $s_{k+\gamma_1}$. Therefore, the span of the multiplier is $\begin{pmatrix} min(\gamma_1+2, 2^{2L}-3-(\gamma_1+2)) \\ \\ min(\gamma_1-4, 2^{2L}-3-(\gamma_1-4)) \end{pmatrix}$.

As we have already seen in the previous chapter, the output sequence of a scalar NLFG has good statistical properties if the span of the multipliers are distinct. Based on this evidence, for a σ-LFSR, it can be conjectured that if the entries of the span vectors are distinct then the output sequence will have good statistical properties. As we will see in the following lemma, this is almost always ensured when $r = 2$ and $P = \begin{bmatrix} 0 & 1 \\ \\ 1 & 0 \end{bmatrix}$.

Lemma 5.2.1. *([RKG16]) Consider a primitive σ-LFSR with L 2-input 2-output delay blocks. Let its component sequences be $\mathcal{S}$ and $\sigma^{\gamma_1}\mathcal{S}$ respectively. For an NLFG configuration with permutation matrix $P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, all the entries of the span vector will be distinct except when $\frac{2^{2L}-3-L}{2} \leq \gamma_1 \leq \frac{2^{2L}-3+L}{2}$.*

Proof. Observe that, the entries of the span vectors will either be of the form $\gamma_1 + i$ or $2^{2L} - 3 - (\gamma_1 + i)$ where $|i| \leq \frac{L}{2}$. Therefore, two of these entries will be equal only if

$$\gamma_1 + i = 2^{2L} - 3 - (\gamma_1 + j) \tag{5.2}$$

for some i, j such that $|i|, |j| \leq \frac{L}{2}$.

Clearly, $|i + j| \leq L$. Hence, Equation 5.2 can be satisfied only if $\frac{2^{2L}-3-L}{2} \leq \gamma_1 \leq \frac{2^{2L}-3+L}{2}$.

Therefore, in case of a σ-LFSR over $\mathbb{F}_2$, the fact that they are disjoint follows from Corollary 3.2.2.1. □

5.2.1 Simulation results of the proposed NLFG

Two configurations of the proposed scheme have been simulated in MATLAB. The first configuration has six 2-input 2-output delay blocks with a

characteristic polynomial $1 + x + x^3 + x^5 + x^6 + x^7 + x^{12}$ while the second one has eight 2-input 2-output delay blocks with a characteristic polynomial $1 + x^2 + x^3 + x^5 + x^{16}$. The resulting output sequences have been analysed for their linear complexity (see Table 5.1). In both cases the linear complexity of the resulting sequence is equal to $\frac{rL(rL-1)}{2}$. Further, we have calculated the frequency of occurrence of k consecutive zeros or ones for various values of k (for the six delay block case). These results have been compared with the results of the scheme proposed in [HPW12](see Table 5.2). Here, Y_1 and Y_2 are the component sequences generated by the scheme proposed in this paper while Z_1 and Z_2 are the component sequences generated by the NLFG scheme described in [HPW12]. Clearly from Table 5.1 and 5.2, for the cases considered, the performance of the proposed scheme is far superior to the scheme given in literature.

The autocorrelation sequences of the generated binary sequences have been plotted. As we can see from Figures 5.4 and 5.5 the generated sequences are very close to having a two level autocorrelation property.

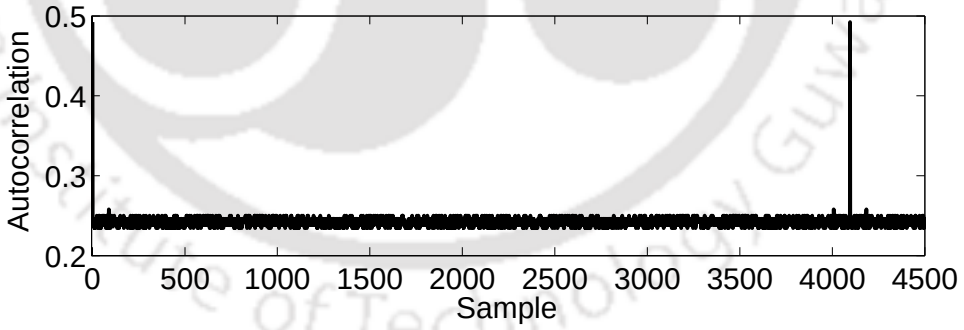

Figure 5.4: Plot of the autocorrelation function for the case $L = 6$

In the following section, we have proposed another nonlinear feedforward logic configuration wherein the outputs of the delay blocks of a σ-LFSR are multiplied as elements in $\mathbb{F}_{q^r}$. This is in contrast to the above scheme and the scheme given in [HPW12], where multiplications are done

Case	L	Langford Arrangement	Minimal Polynomial	Lin. Com.
1	6	2 3 1 2 1 3 and 3 1 2 1 3 2	$x^{66}+x^{64}+x^{63}+x^{61}+x^{60}+x^{59}$ $+x^{54}+x^{53}+x^{52}+x^{49}+x^{37}+x^{35}$ $+x^{32}+x^{31}+x^{30}+x^{29}+x^{27}+x^{26}$ $+x^{24}+x^{23}+x^{22}+x^{19}+x^{18}+x^{17}$ $+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{3}+1$	66
2	8	2 3 4 2 1 3 1 4 and 4 1 3 1 2 4 3 2	$x^{120}+x^{119}+x^{117}+x^{116}+x^{114}+$ $x^{113}+x^{112}+x^{111}+x^{110}+x^{106}+$ $x^{104}+x^{103}+x^{102}+x^{101}+x^{99}+x^{96}$ $+x^{95}+x^{93}+x^{91}+x^{89}+x^{84}+x^{83}+$ $x^{82}+x^{74}+x^{73}+x^{71}+x^{70}+x^{65}+$ $x^{62}+x^{59}+x^{58}+x^{53}+x^{52}+x^{50}+$ $x^{46}+x^{44}+x^{41}+x^{40}+x^{39}+x^{35}+$ $x^{34}+x^{33}+x^{32}+x^{30}+x^{26}+x^{25}+$ $x^{24}+x^{23}+x^{21}+x^{19}+x^{16}+x^{12}+$ $x^{11}+x^{8}+x^{7}+1$	120

Table 5.1: Linear complexity of NLFG sequences

k	$(\mathbf{Z_1})$		$(\mathbf{Z_2})$		$(\mathbf{Y_1})$		$(\mathbf{Y_2})$	
k	Consecutive 0's	Consecutive 1's	Consecutive 0's	Consecutive 1's	Consecutive 0's	Consecutive 1's	Consecutive 0's	Consecutive 1's
1	2304	1792	2304	1792	2080	2016	2080	2016
2	1407	896	1407	896	1055	992	1055	992
3	958	448	958	448	534	448	534	448
4	701	224	701	224	277	232	277	232
5	524	112	525	112	147	110	147	110
6	383	56	385	56	75	56	75	56
7	270	27	272	27	41	28	41	28
8	185	10	187	10	23	14	23	14
9	133	3	134	3	12	6	12	6
10	94	1	94	1	7	1	7	1
11	66	-	65	-	4	-	4	-
12	46	-	44	-	3	-	3	-
13	30	-	28	-	2	-	2	-
14	18	-	16	-	1	-	1	-
15	11	-	9	-	-	-	-	-
16	6	-	4	-	-	-	-	-
17	5	-	3	-	-	-	-	-
18	4	-	2	-	-	-	-	-
19	3	-	1	-	-	-	-	-
20	2	-	-	-	-	-	-	-
21	1	-	-	-	-	-	-	-

Table 5.2: Occurrence of chains of length k in one period

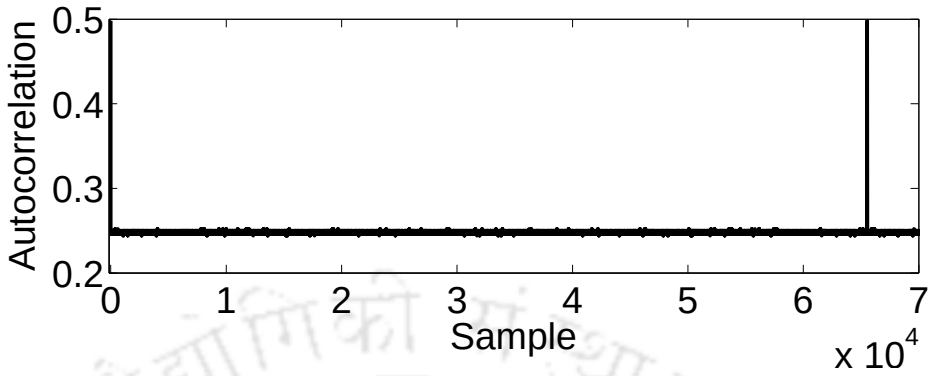

Figure 5.5: Plot of the autocorrelation function for the case $L = 8$

bit-wise.

5.3 NLFGs with modulo multipliers

Since $\mathbb{F}_q^r$ is known to be isomorphic to $\mathbb{F}_{q^r}$, a σ-LFSR can be seen as an FSR over the field $\mathbb{F}_{q^r}$ [LN86]. Thus, each state vector of a σ-LFSR can be seen as a vector in $\mathbb{F}_{q^r}^L$. The characteristic polynomial of the σ-LFSR being primitive ensures that all non zero vectors in $\mathbb{F}_{q^r}^L$ occur as state vectors exactly once in every period.

Let $p(x)$ be a primitive polynomial of degree r. Now, $\mathbb{F}_{q^r}$ can be seen as the residue class ring ${}^{\mathbb{F}_q[x]}/_{<p(x)>}$. The set $\{[1], [x], \ldots, [x^{r-1}]\}$ is a basis of ${}^{\mathbb{F}_q[x]}/_{<p(x)>}$. Given a polynomial $f(x) \in \mathbb{F}_q[x]$, the equivalence class of $f(x)$ has a unique representative element with degree less than r. We therefore have the following map $\mathcal{M} : \mathbb{F}_{q^r} \to \mathbb{F}_q^r$.

$$\mathcal{M}\big(f_0[1] + f_1[x] + \ldots + f_{r-1}[x^{r-1}]\big) = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{r-1} \end{bmatrix}$$

Clearly, the above map is a vector space homomorphism. Using this map, we define multiplication of two elements in $\mathbb{F}_q^r$, denoted as $\times$, as follows.

$$v_1 \times v_2 = \mathcal{M}\big([\mathcal{M}^{-1}(v_1).\mathcal{M}^{-1}(v_2)]\big)$$

where $v_1, v_2 \in \mathbb{F}_q^r$. Let $v_1 = \mathcal{M}([f_1(x)])$ and $v_2 = \mathcal{M}([f_2(x)])$. Therefore, $v_1 \times v_2$ is a vector whose entries are the coefficients of the polynomial $g(x) = f_1 f_2 \ mod \ p(x)$. If f_1 and f_2 are the unique elements in their respective equivalence classes having degree less than r then $f_1(x)f_2(x)$ is a polynomial with degree less than $2r$. Let $v \in \mathbb{F}_q^{2r-1}$ be a vector whose entries are the coefficients of $f_1 f_2$. Now, $v_{f_1 f_2} = v_1 * v_2$ where $*$ denotes convolution over $\mathbb{F}_q$. Observe that $v_1 \times v_2 = \mathcal{Q} v_{f_1 f_2} = \mathcal{Q}(v_1 * v_2)$ where $\mathcal{Q} \in \mathbb{F}_q^{r \times (2r-1)}$ is the following matrix.

$$\mathcal{Q} = [I^{r \times r} \vdots \mathcal{M}([x^r]) \vdots \ \ldots \ \vdots \mathcal{M}([x^{2r-2}])] \tag{5.3}$$

Example 5.3.1. *Consider the vectors* $v_1 = \mathcal{M}([1 + 2x + 3x^2]) = [1\ 2\ 3]^T$ *and* $v_2 = \mathcal{M}([2 + 3x + 4x^2]) = [2\ 3\ 4]^T$. *Let* $p(x) = x^3 + 4x + 2$ *be a primitive polynomial having degree* 3 *over* $\mathbb{F}_5$. *From Equation 5.3, the* $\mathcal{Q}$ *matrix is as follows.*

$$\mathcal{Q} = \begin{bmatrix} 1\ 0\ 0\ 2\ 0 \\ 0\ 1\ 0\ 4\ 2 \\ 0\ 0\ 1\ 0\ 4 \end{bmatrix}_{3 \times 5}$$

Now, $v_1 * v_2 = [2\ 2\ 1\ 2\ 2]^T \in \mathbb{F}_5^5$. *Therefore,* $v_1 \times v_2 = \mathcal{Q}(v_1 * v_2) = [1\ 4\ 4]^T$ *(i.e.* $\mathcal{M}([1 + 4x + 4x^2])$*).*

The schematic diagram for the scheme described here is similar to the one shown in Figure 5.1. Figure 5.1, the proposed scheme consists of a σ-LFSR and a multiplier assembly with $m \leq \lfloor \frac{L}{2} \rfloor$ multipliers. Each multiplier takes the output of two distinct r-input r-output delay blocks, convolves them and multiplies the result with the matrix $\mathcal{Q}$ given in Equation 5.3. It thus implements the map '$\times$' described above. The outputs of the multipliers are then added to generate the output vector sequence. As in other NLFG configurations, the output of each delay block can act as an input to at most one multiplier. The output sequence of a primitive σ-LFSR satisfies the span-L property. Further, outputs of the delay blocks are multiplied as elements in $\mathbb{F}_q^r$. Therefore, the analysis given in Section 4.2 is valid for this scheme.

Let $\mathbf{N}_m^L(v)$ be the number of occurrences of a vector v in a single cycle of the sequence generated by the proposed NLFG. From Corollary 4.2.4.1, $\mathbf{N}_m^L(v)$ is given by;

$$\mathbf{N}_m^L(v) = \begin{cases} q^{r(L-m-1)}(q^{rm}-1) & when\ v \neq 0 \\ q^{r(L-m-1)}(q^{rm}+q^r-1)-1 & when\ v = 0 \end{cases} \tag{5.4}$$

It is clearly seen that the output sequence of this NLFG configuration has a lesser bias towards the all-zero vectors as compared to the scheme described in Section 5.1.

Example 5.3.2. *Let $q = 2, L = 5, m = 2$ and $r = 3$. The number of occurrences of $v_1 = [0\ 0\ 0]^T$, $v_2 = [0\ 1\ 0]^T$ and $v_3 = [1\ 1\ 1]^T$ in the sequence generated by the scheme given in [HPW12] are $7999, 4800$ and 1728 respectively. However, the number of occurrences of the vectors v_1, v_2 and v_3 in a nonzero sequence generated by the scheme proposed in this section are $4543, 4032$ and 4032 respectively.*

Based on the arguments given in the previous chapter, it can be inferred that if the output of a delay block not connected to the multiplier

assembly is added to the output of the NLFG, the resulting vector sequence is balanced.

5.4 Summary

We have discussed an existing scheme and have proposed two additional scheme by which nonlinear feed-forward logic can be extended to σ-LFSRs. We have shown that component sequences generated by the first proposed scheme are more balanced that those generated by the existing one. Further, simulations indicate that these component sequences are also superior with respect to other properties associated with randomness. For the second scheme, we have derived the statistical distribution of the generated vector sequences and have shown that these are more balanced than those generated by the scheme in literature.

Chapter 6

Generating de Bruijn Sequences

> "Begin at the beginning," the King said, very gravely, "and go on till you come to the end: then stop."
>
> *Lewis Carroll, Alice in Wonderland*

In this chapter, we describe a method of traversing the set of n-th order binary de Bruijn sequences [1] using a series of graph theoretic transformations.

6.1 De Bruijn graphs

Recall, An n-th order k-ary de Bruijn sequence, $DB_n(k)$, is a periodic sequence of length k^n having every possible k-ary subsequence of length n exactly once in each period. Such sequences are closely associated with special directed Eulerian graphs known as de Bruijn graphs [Bru46].

In the following, we restrict our discussion to the binary case only.

[1] The oldest instance of a de Bruijn sequence comes from a sanskrit prosody or chandas: *yamātārājabhānasalagām* [Ste10]

Definition 6.1.1. *A binary de Bruijn graph of order n, denoted as G_n, is a directed graph with 2^n vertices, each labeled with a unique n bit string. Each edge of the graph is labeled with a binary string of length $(n+1)$. The edge labeled as $s_0 s_1 \dots s_n$ connects the source vertex labeled $s_0 s_1 \dots s_{n-1}$ with the target vertex labeled $s_1 s_2 \dots s_n$.*

Example 6.1.1. *Figure 6.1 represents a second order binary de Bruijn graph G_2.*

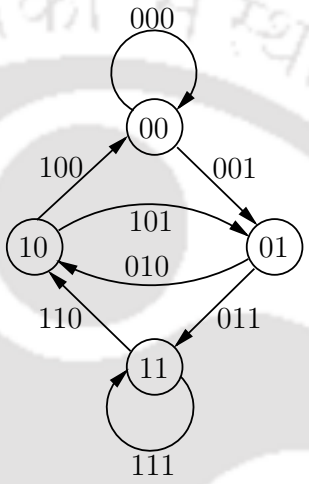

Figure 6.1: G_2: De Bruijn graph of order 2.

In G_n, each vertex $v = (s_0 s_1 \dots s_{n-1})$ has two out-edges $(s_0 s_1 \dots s_{n-1} 1)$ and $(s_0 s_1 \dots s_{n-1} 0)$; these edges are known as the one-edge and the zero-edge of v respectively. Since de Bruijn graphs are connected and balanced they always contain an Eulerian cycle. Observe that, we can obtain an $(n+1)$-th order de Bruijn sequence by considering the sequence of most significant bits of edges in an Eulerian cycle of G_n. For example, consider the 2^{nd} order de Bruijn graph in Example 6.1.1. It has two Eulerian cycles, one of them is given in Figure 6.2 and the corresponding de Bruijn sequence is 00011101. Clearly, there exists a one-to-one correspondence between Eulerian cycles of G_n and $(n+1)$-th order de Bruijn sequences. We now proceed to briefly describe the process of generating Eulerian cycles of G_n given in [BK11]. All de Bruijn sequences that are cyclic shifts

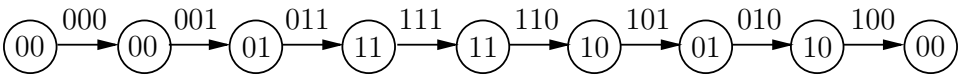

Figure 6.2: Eulerian path of the de Bruijn graph G_2.

of each other shall be considered equivalent.

An oriented spanning tree is an acyclic subgraph of a directed graph $G = (V, E)$ which has a vertex $r \in V$ known as root vertex such that $outdeg(r) = 0$ and there exists a path from every vertex $v \in V \backslash \{r\}$ to r. For a directed graph shown in Figure 6.1, an oriented spanning tree T rooted at 00 is given in Figure 6.3.

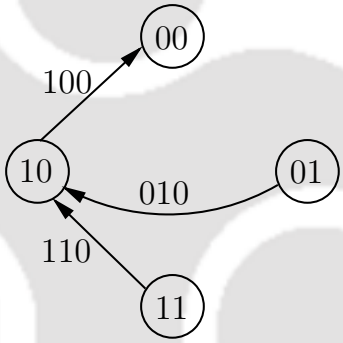

Figure 6.3: Spanning tree of G_2 rooted at 00.

Let T be a spanning tree of $G_n = (V, E)$ where V is the set of vertices and E is the edge set in G_n. Now, for all $v \in V \setminus \{r\}$, l_v is the list of its outgoing edges in G_n. The last element of l_v is the outgoing edge of v which occurs in the spanning tree T. In case of the root vertex r, last element of l_r is the symbol Ω and the first entry of l_r can be any of its outgoing edges in G_n. This array of lists is known as tree array (Example 6.1.2).

Example 6.1.2. *Consider the de Bruijn graph G_2 shown in Figure 6.1. A spanning tree of G_2 is shown in Figure 6.3. A tree array l_v corresponding to T is given as follows:*

$$l_{00} = \{(000), \quad \Omega \ \}$$
$$l_{01} = \{(011), (010)\}$$
$$l_{10} = \{(101), (100)\}$$
$$l_{11} = \{(111), (110)\}$$

Now, let T be a spanning tree of G_n rooted at the vertex r and consider its corresponding tree array. A method of obtaining the Eulerian cycle of G_n is as follows. Starting with the unique edge of G_n which does not lie in the tree array, each edge x is followed by the first unused outgoing edge of $t(x)$ in the tree array. This process stops when the only unused entry of the tree array is Ω. Thus, given a spanning tree and a tree array we can generate an Eulerian cycle in G_n. Further, it has been shown in [BK11] that the correspondence between a spanning tree - tree array pair and Eulerian cycles is one-to-one and we can easily obtain one from the other.

Example 6.1.3. *Consider the G_2 given in Figure 6.1 and its spanning tree T rooted at* 00 *shown in Figure 6.3. A tree array for the spanning tree is given in Example 6.1.2. We start with the edge* 001 *followed by the edge* 011 *which is the first unused outgoing edge of the vertex* $01 = t(001)$. *By repeating this process, we get the Eulerian cycle of G_2 shown in Figure 6.4. The corresponding* 3*-rd order de Bruijn sequence is* 00111010.

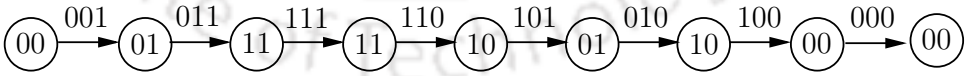

Figure 6.4: Eulerian cycle of the de Bruijn graph G_2.

6.2 The construction

A binary de Bruijn graph G_n having a vertex $s = (s_0 s_1 \dots s_{n-1})$ has two out-edges, namely zero-edge and one-edge, which connect s to its successor

vertex $(s_1 s_2 \dots s_{n-1}0)$ and $(s_1 s_2 \dots s_{n-1}1)$ respectively. These vertices are called conjugates of each other and corresponding edges are called conjugate edges. In a spanning tree of G_n, any vertex s is connected to one of its successor vertices. The all-zero and all-one vertices have only one possible successor in the spanning tree. Therefore, we consider these vertices merged with their respective successor vertices in the spanning tree as a single vertex. For example, consider the spanning tree of G_3 shown in Figure 6.5. Here 000 and 001 (similarly, 111 and 110) are jointly treated as a single vertex $000 - 001$ $(111 - 110)$.

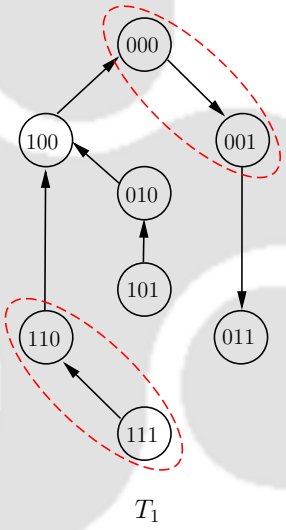

Figure 6.5: A spanning tree of G_3 rooted at 011.

Note that fixing the root vertex and its unique outgoing edge that does not lie in the tree array essentially fixes the starting edge of the Eulerian cycle (therefore, the first n-bits of the de Bruijn sequence). Therefore, if we fix the entry corresponding to the root vertex in the tree array, we have a one-to-one correspondence between de Bruijn sequences with a given initial state and oriented spanning trees of G_n. In the remainder of this section we will show that from a given spanning tree of G_n rooted at a particular vertex we can generate any other spanning tree of G_n having

the same root by a sequence of transformations.

The process of replacing the outgoing edge of one vertex in the spanning tree by its conjugate edge is known as edge switching. In a spanning tree T, a leaf node is a node that does not have any incoming edge. For example, consider one of the binary spanning trees of G_2, T, rooted at 00 given in Figure 6.3. Here, 01 and 11 are the leaf nodes.

Lemma 6.2.1. *Switching the outgoing edge of a leaf node in a spanning tree of G_n generates another spanning tree.*

Proof. Let T be a spanning tree of G_n. Suppose an outgoing edge e of a vertex $v \in T$ is replaced by its conjugate edge e', then this switching results in a cycle only if $t(e)$ is a vertex from which there exists a path to v. Otherwise the resulting graph will be an another spanning tree. Now, if the node $v \in T$ is a leaf node then there exists no path to v from any other vertex. Therefore, when the outgoing edge of a leaf node is switched the resulting graph is another spanning tree of G_n. □

Example 6.2.1. *Consider the spanning tree, T_1, of G_3 as given in Figure 6.5. T_1 has two leaf nodes viz.* 101 *and the merged pair* $111-110$. *Switching of these edges,* 101 *and* $111-110$, *produce two different spanning trees (see Figure 6.6).*

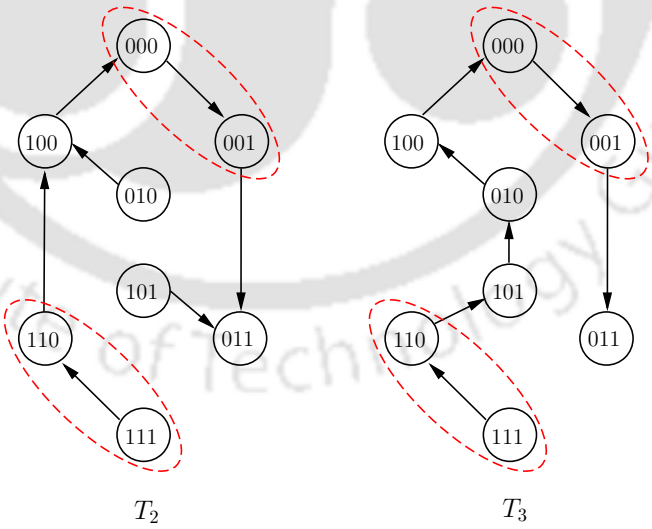

Figure 6.6: Spanning trees of G_3 rooted at 011.

In a directed spanning tree T a node x is known as an ancestor of a node v if there exists a path from x to v in T. For example, in Figure

6.5 the ancestors of the node 100 are 101, 010 and the merged vertex pair $111-110$. Now, we proceed to prove our main result.

Theorem 6.2.2. *Given a spanning tree T of G_n rooted at r, any other spanning tree T' of G_n having the same root can be obtained from T by a sequence of leaf nodes edge switching.*

Proof. Let the number of nodes in T whose outgoing edges are same as that in T' be κ. Now, consider a node $v \neq r$ in T whose outgoing edge is different from that in T' and all of whose ancestor nodes have the same out-edges as in T'. Now, we apply post-order depth first traversal on the sub-tree of T rooted at v and switch the nodes in the order in which they occur in this traversal. Clearly, this is a sequence of leaf node edge switchings culminating in the switching of the outgoing edge of v. Once the node v is switched, we switch all the other nodes that we had switched before v in the reverse order. We now have a new graph wherein the number of nodes in T whose outgoing edges are same as that in T' is $\kappa+1$. We keep repeating this process till the outgoing edges of all vertices in T become same as T'. Thus, we transform the spanning tree T into T'. □

Given a de Bruijn sequence we can construct the corresponding spanning tree and tree array. Now, by randomly performing a sequence of leaf node switches we can get a random spanning tree of G_n and therefore a random de Bruijn sequence.

Example 6.2.2. *Consider a spanning tree, T_1, of the 3-rd order de Bruijn graph shown in Figure 6.7(a). Let T_1' be any other spanning tree of G_3 rooted at the same vertex as shown in Figure 6.7(g). Here $\kappa = 5$ and except 101 and 100 all other nodes in T_1 have the same outgoing edges as in T_1'. Now, 101 is a leaf node and switching its edge gives us a new spanning tree shown in Figure 6.7(b). This spanning tree has 6 nodes whose outgoing edges are same as in T_1'. Now, the vertex 100 is not a leaf node. By applying depth first traversal algorithm on the sub-graph rooted at node 100 we get the list $\{111-110, 010, 100\}$. We now switch the nodes in the order $111-110, 010$ and 100. This gives us the spanning tree shown in Figure 6.7(e). We now again switch the nodes 010 and $111-110$. This gives us the spanning tree T_1'. These switching operations are depicted in Figure 6.7 (a-g). The corresponding tree arrays and 4-th order de Bruijn sequences of the spanning trees T_1 and T_1' are tabulated in Table 6.1.*

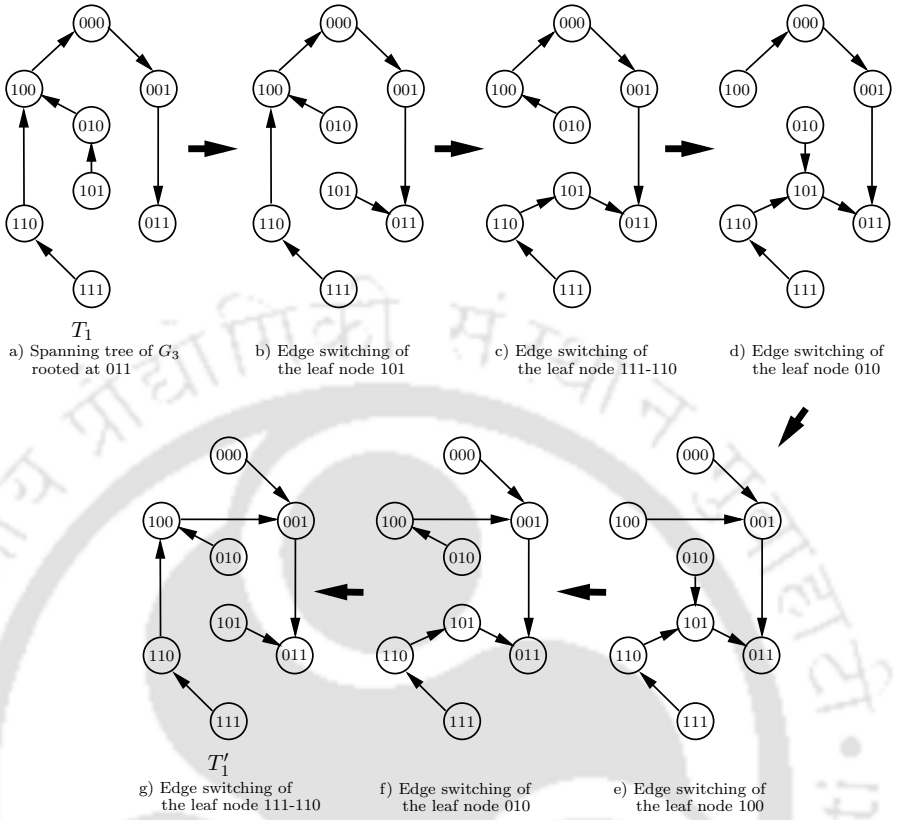

Figure 6.7: Switching of leaf node edges in a spanning tree of G_3.

Remark 6.2.1. *A sequence of Leaf node transformations can be represented by a string of vertices whose outgoing edges are switched. It is interesting to note that the set of such strings form a group under string concatenation where the zero element is the empty string and the inverse of any string is a string where the same nodes occur in the reverse order.*

These results can be easily extended for k-ary de Bruijn graphs. As in the binary case, there will be a unique outgoing edge of the root vertex that is missing in the tree array. Corresponding to each spanning tree and the missing outgoing edge of the root vertex, we will get a set of de Bruijn sequences. Here, an edge switching will take us from one such set to another. Observe that, the proof of Theorem 6.2.2 is also be valid in this case.

	Spanning tree (T_1)	Spanning tree (T_1')
Tree Array	$l_{000} = \{(0000), (0001)\}$ $l_{001} = \{(0010), (0011)\}$ $l_{010} = \{(0101), (0100)\}$ $l_{011} = \{(0110), \quad \Omega \quad \}$ $l_{100} = \{(1001), (1000)\}$ $l_{101} = \{(1010), (1011)\}$ $l_{110} = \{(1101), (1100)\}$ $l_{111} = \{(1111), (1110)\}$	$l_{000} = \{(0000), (0001)\}$ $l_{001} = \{(0010), (0011)\}$ $l_{010} = \{(0101), (0100)\}$ $l_{011} = \{(0110), \quad \Omega \quad \}$ $l_{100} = \{(1000), (1001)\}$ $l_{101} = \{(1010), (1011)\}$ $l_{110} = \{(1101), (1100)\}$ $l_{111} = \{(1111), (1110)\}$
de Bruijn seq.	0111101011001000	0111101011000010

Table 6.1: Tree arrays and de Bruijn sequences of T_1 and T_1'

6.3 Summary

In this chapter, we have shown that one can go from a given de Bruijn sequence to another by a sequence of leaf node edge switchings. By randomly choosing the sequence of leaf node edge switches we can generate a random spanning tree of G_n and therefore a random de Bruijn sequence.

Chapter 7

Conclusion

"It's for you to begin, it's for me to end,
You and me together fulfil the blend."

Rabindranath Tagore

In this chapter, we summarise our contribution and provide possible directions for future research.

7.1 Contribution

In this work, we have looked into the generation and analysis of pseudo-random sequences. The following is a list of our results.

- We have extended the notion of NLFGs to arbitrary finite fields and have analysed the statistical distribution of symbols in the sequences generated by them.
- Two novel methods of implementing NLFGs with σ-LFSR have been proposed.
    1. The first scheme involves the use of permutation matrices. The component sequences generated by such a scheme are shown to be more balanced than those generated by an existing scheme.

Further, these sequences also perform better with respect to other statistical parameters.

2. In the second scheme, the outputs of the delay blocks are multiplied as elements of an extension field. We have shown that the vector sequence generated by such a scheme is statistically more balanced than the scheme available in literature

- Finally, we have discussed a method of going from one n-th order de Bruijn sequence to another by a series of graph transformations.

7.2 Future work

In this thesis, we have analysed sequences generated by NLFGs only for their balance property. However, the performance of these sequences is still to be analysed for other statistical properties such as span-L, ideal 2-level autocorrelation etc. Further, such an analysis could help in designing cryptographically secured NLFG configurations.

It will be interesting to explore the NLFG configurations where the number of inputs to each multiplier is different.

Bibliography

[Ann97] F.S. Annexstein. *Generating de Bruijn sequences: An efficient implementation.* IEEE Trans. on Computers, 46(**2**), pp. 198-200, 1997.

[Art11] M. Artin. *Algebra.* Pearson Prentice Hall, 2011.

[BK11] H. Bidkhori and S. Kishore. *A Bijective Proof of a Theorem of Knuth.* Combinatorics, Probability and Computing, 20(**1**), pp. 11-25, 2011.

[BP01] S. Bedi and N. Pillai. *Cryptanalysis of the nonlinear feedforward generator.* In *Proceedings of the Progress in Cryptology INDOCRYPT 2001*,2247 of Lecture Notes in Computer Science, pp. 188-194, Springer Berlin Heidelberg, 2001.

[Bru46] N.G. de Bruijn. *A combinatorial Problem.* In *Proceedings of Section of Sciences of the KoninklijkeNederlandse Akademie van Wetenschappen te Amsterdam*, 49(**7**), pp. 758-764, 1946.

[DAG90] E. Dawson, J. Asenstorfer, and P. Gray. *Cryptographic properties of groth sequences.* Australian Journal of Combinatorics, **1**:53-65, 1990.

[DP03] M. Dewar and D. Panario. *Linear transformation shift registers.* IEEE Trans. on Inf. Theory, 49(**8**), pp. 2047-2052, 2003.

[EL84] T. Etzion and A. Lempel. *Algorithms for the generation of full-length shift-register sequences.* IEEE Trans. on Inf. Theory, 30(**2**), pp. 480-484, 1984.

[Fre82] H. Fredricksen. *A survey of full length nonlinear shift register cycle algorithms.* SIAM review, 24(**2**), pp. 195-221, 1982.

[Fred68] H. Fredricksen. *Disjoint cycles from the de Bruijn Graph.* DTIC Document, 1968.

[Gal13] J.A. Gallian. *Contemporary Abstract Algebra.* Cengage learning, 8 edition, 2013.

[GG04] S.W. Golomb and G. Gong. *Signal Design for Good Correlation: For Wireless Communication, Cryptography, and Radar.* Cambridge University Press, New York, NY, USA, 2004.

[GG06] B.M. Gammel and R. Göttfert. *Linear filtering of nonlinear shift-register sequences.* In *Proceedings of the Coding and Cryptography*, pp. 354-370, Springer, 2006.

[Gol81] S.W. Golomb *Shift Register Sequences.* Aegean Park Press, 1981.

[Gro71] E. Groth. *Generation of binary sequences with controllable complexity.* IEEE Trans. on Inf. Theory, 17:288-296, 1971.

[GYZ14] J.L. Gross, J. Yellen, and P. Zhang. *Handbook of Graph Theory*. CRC Press, 2 edition, 2014.

[HJM07] M. Hell, T. Johansson, and W. Meier. *Grain: a stream cipher for constrained environments*. International Journal of Wireless and Mobile Computing, 2(1), 2007.

[HPW12] S.U. Hasan, D. Panario, and Q. Wang. *Word-oriented transformation shift registers and their linear complexity*. In *Proceedings of the Sequences and Their Applications-SETA2012*,Springer Berlin Heidelberg, 7280, pp. 190-201, Berlin, Heidelberg, 2012.

[Jan89] C.J.A. Jansen. *Investigations on nonlinear stream cipher systems: construction and evaluation methods*. Ph.D. dissertation, Technische Universiteit Delft, 1989.

[Key76] E.L. Groth. *An analysis of the structure and complexity of nonlinear binary sequence generators*. IEEE Trans. on Inf. Theory, 22(6):732-736, 1976.

[KP12] S. Krishnaswamy and H. Pillai. *On the number of linear feedback shift registers with a special structure*. IEEE Trans. on Inf. Theory, 58(3):1783-1790, 2012.

[Kri12] S. Krishnaswamy. *On Multisequences and Applications*. PhD dissertation, Indian Institute of Technology Bombay, 2012.

[Lan58] C.D. Langford. *Problem*. Mathematical Gazette, **42**, 228, 1958.

[Lem70] A. Lempel. *On a homomorphism of the de Bruijn graph and its applications to the design of feedback shift registers.* IEEE Trans. on Inf. Theory, 100(**12**), pp. 1204-1209, 1970.

[LN86] R. Lidl and H. Niederreiter. *Introduction to Finite Fields and their Applications.* Cambridge University Press, 1986.

[MOV96] A. Menezes, P.V. Oorschot, and S. Vanstone. *Handbook of applied cryptography, discrete mathematics and its applications.* CRC Press, 1996.

[MS15] J. Mykkeltveit and J. Szmidt. *On cross joining de Bruijn sequences.* Topics in Finite Fields, 632(**2**), pp. 335-346, 2015.

[Nie95] H. Niederreiter. *The multiple-recursive matrix method for pseudorandom number generation.* Finite Fields and Their Applications, 1(1):3-30, 1995.

[PP09] C. Paar and J. Pelzl. *Understanding Cryptography: A Textbook for Students and Practitioners.* Springer Berlin Heidelberg, 2009.

[PSM82] R. Pickholtz, D. Schilling, and L. Milstein. *Theory of spread-spectrum communications-A tutorial.* IEEE Trans. on Comm., 30(**5**), pp. 855-884, 1982.

[PW72] W. Peterson and E. Weldon. *Error-correcting codes.* MIT Press, 1972.

[RKG16] S. Roy, S. Krishnaswamy, and P. Goyal. *On nonlinear feedforward logic for σ-lfsrs.* In *Proceedings of the 22nd International Symposium on Mathematical Theory of Networks and Systems*, pp. 366-372, Minneapolis, MN, USA, 2016.

[Ste10] S.K. Stein. *Mathematics: The Man-Made Universe*. Dover Publications, 3 edition, 2010.

[Teo13] S.G. Teo. *Analysis of nonlinear sequences and stream ciphers*. PhD dissertation, Queensland University of Technology, 2013.

[TV02] B. Tsaban and U. Vishne. *Efficient linear feedback shift registers with maximal period*. Finite fields and their applications, 8(**2**), pp. 256-267, 2002.

[ZHH07] G. Zeng, W. Han, and K. He. *High efficiency feedback shift register: σ-lfsr*. IACR Eprint archive, 2007.

List of publications

Journal

1. Suman Roy and Srinivasan Krishnaswamy, "On the Frequency of Symbols in Sequences Generated by Nonlinear Feedforward Generators", *Cryptography and Communications*, Springer, 2019.

International Conference

1. Suman Roy, Srinivasan Krishnaswamy, and Pushpinder Goyal, "On Nonlinear Feedforward Logic for σ-LFSRs", In *Proceedings of the 22nd International Symposium on Mathematical Theory of Networks and Systems (MTNS)*, Minneapolis, MN, USA, July 2016.

2. Suman Roy, Srinivasan Krishnaswamy, and Potnuru V. Kumar,"On Leaf Node Edge Switchings in Spanning Trees of De Bruijn Graphs", In *Proceedings of the 4th International Conference on Mathematics and Computing (ICMC)*, IIT BHU, Varanasi, UP, India, January 2018.

Index