CRYPTOGRAPHIC AND COMBINATORIAL PROPERTIES OF BOOLEAN FUNCTIONS AND S-BOXES

Thesis submitted to Indian Statistical Institute

by

Kishan Chand Gupta

Applied Statistics Unit
Indian Statistical Institute
2004

Cryptographic and Combinatorial Properties of Boolean Functions and S-boxes

Thesis submitted to Indian Statistical Institute in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy

To my wife

by

Kishan Chand Gupta
Applied Statistics Unit
Indian Statistical Institute
203, B. T. Road, Calcutta 700 108, INDIA
e-mail : kishan_t@isical.ac.in

under the supervision of

Dr. Palash Sarkar
Applied Statistics Unit
Indian Statistical Institute
203, B. T. Road, Calcutta 700 108, INDIA
e-mail : palash@isical.ac.in

To my wife.

Acknowledgements

A number of people have inspired me in starting and continuing my research. It is indeed a pleasure for me to acknowledge the help received from various persons. I am particularly grateful to Prof. Bimal Roy. I was led into cryptology by joining a project headed by him. I thank him for his constant encouragement and inspiration during my research work. I owe a considerable debt to Prof. Rana Barua for his help in many ways. I am grateful to Prof. K. S. Vijayan for his loving and helping attitude.

My supervisor and friend Dr. Palash Sarkar has always been a great source of inspiration. It was due to him I was able to come back to academics after a break of seven years. His thoughts on the subject greatly influence my own understanding of the subject and are reflected to some extent in the present work.

I thank Dr. Subhamoy Maitra for his help and encouragement. He always motivated me and provided thoughtful suggestions. I wish to thank Jim Massey, Willi Meier, Kaisa Nyberg, Vincent Rijmen and Johan Wallén for providing comments on some of the papers on which this thesis is based.

I am thankful to Dr. Arup Pal, Dr. Pinakpani Pal and Dr. Punam Kumar Saha for their support and encouragement. I am grateful to Prof. Aditya Bagchi, Mr. Subhasis Kumar Pal and Mr. Dibyendu Bose for their generous support and help. I thank Sandeepan Chowdhury for numerous help and discussions in the early stages of my work and for being a humourous and good friend at all times. Also, I must thank all my friends Pradeep, Tanmay, Avishek, Sourav, Arindam, Joydeep, Sanjit, Partha, Sagarnil, Madhu, Chandan, Deepak, Rajasekhar, Ratna, ..., who stood by me during all good and hard times. I also thank all the members of Cryptology Research Group, Indian Statistical Institute for their fruitful interactions. I am grateful to all the staff of ASU for providing all kinds of facilities whenever needed.

I express my thanks to my family. My mother, father and sisters have always been a great source of inspiration. I thank Sankar, Manoj, Vijay, Dipak and their wives for their support and encouragement. I also thank Kartick, Baby and Sahleja who always inspired me. I thank Ridhika, Sidhika, Shital and Rama who always kept me refreshed. My daughter Shruti is always nice.

Last but not the least I am grateful to all the teachers and staff of ISI.

Contents

1 Introduction **9**

1.1 Thesis Plan 11

2 Definitions and Preliminaries **15**

2.1 Boolean Functions 15

2.1.1 Representation of Boolean Functions 16

2.1.2 Walsh Transform 17

2.1.3 Nonlinearity 19

2.1.4 Resiliency 20

2.1.5 Strict Avalanche Criteria 20

2.2 S-Boxes 21

2.2.1 S-Box Representation 21

2.2.2 Nonlinearity of S-boxes 22

2.2.3 Resiliency of S-boxes 23

2.2.4 SAC of S-boxes 23

2.3 Cryptographic Motivation 23

2.3.1 Significance of Correlation Immunity 23

2.3.2 Significance of Nonlinearity 26

2.3.3 Significance of SAC 28

2.3.4 Significance of Degree 30

3 Computing Partial Walsh Transform from ANF **34**

3.1 Introduction 34

3.2 Walsh Transform 36

3.2.1 Case $\mathbf{g(x_1, x_2, \cdots, x_m) = x_1 x_2 \cdots x_m}$ 37

3.2.2 Arbitrary $\mathbf{g}$ 38

3.3 Simplifying C_v 43

3.4 Algorithm 46

3.5 Experimental Results 51

3.6 Possible Improvements 54

3.7 Conclusion 55

4 A General Correlation Theorem **56**

4.1 Introduction 56

4.2 Convolution and Composition Theorems 57

4.3 Correlation Theorem 59

4.4 Nyberg's Correlation Theorems 63

4.5 Applications 65

4.5.1 Brick Layering 65

4.5.2 Substitute-and-Add 67

4.6 Conclusion 69

5 Construction of Perfect Nonlinear Functions Satisfying Higher Order SAC **70**

5.1 Introduction 71

5.2 Basic Results 72

5.3 Construction of $(n, 2, \frac{n}{2} - 2)$ S-box 74

5.4 Relation With One Factorization of a Complete Graph 78

5.4.1 Improvements for Two Output S-Boxes 80

5.5 Construction of $(n, 3, k)$ S-boxes 80

5.6 Maximally Nonlinear Functions 81
5.7 Improving Algebraic degree 83
5.8 A Concrete Example 85
5.9 Conclusion 86

6 Construction of High Degree Resilient S-Boxes With Improved Nonlinearity 88
6.1 Introduction 88
6.2 Coding Theory Results 89
6.3 Construction of (n, m, t)-Resilient S-box with Degree $> m$. 89
6.4 Comparison 91
6.5 Second Construction of (n, m, t)-Resilient S-box with Degree $> m$. 92
6.6 Degree Comparison Based on MZZ Construction 95
6.7 Conclusion 96

7 Improved Construction of Nonlinear Resilient S-Boxes 97
7.1 Introduction 97
7.2 A description of the Maiorana-McFarland Construction 98
7.2.1 Some Recent Constructions 99
7.3 A Construction to Obtain High Nonlinearity 100
7.4 Nonlinearity Comparison Based on Construction-I 107
7.5 Conclusion 109

8 Software Implementation of Resilient Maiorana-McFarland S-Boxes 110
8.1 Applicability of Resilient S-boxes in Nonlinear Combiner Model 110
8.2 Algebraic Attacks 113
8.3 Maiorana-McFarland Construction for S-boxes 114
8.4 A Concrete Example 114
8.4.1 Construction of $L(C)$ 114

8.4.2 Construction of (16, 4, 2) Resilient S-Box 115
8.4.3 Implementation 118
8.4.4 Computing the Output 118
8.5 General Methodology 119
8.6 Conclusion 120

Chapter 1

Introduction

In this thesis we study combinatorial aspects of Boolean functions and S-boxes with important cryptographic properties and construct new functions possesing such properties. These have possible applications in the design of private key (symmetric key) cryptosystems.

Symmetric key cryptosystems are broadly divided into two classes.

1. Stream Ciphers,
2. Block Ciphers.

Some recent proposals of stream ciphers are SNOW [37], SCREAM [52], TURING [98], MUGI [117], HBB [102], RABBIT [9], HELIX [38] and some proposals of block ciphers are DES, AES, RC6 [97], MARS [12], SERPENT [6], TWOFISH [104].

In *stream cipher cryptography* a pseudorandom sequence of bits of length equal to the message length is generated. This sequence is then bitwise XOR-ed (addition modulo 2) with the message sequence and the resulting sequence is transmitted. At the receiving end, deciphering is done by generating the same pseudorandom sequence and again bitwise XOR-ing the cipher bits with the random bits. The seed of the pseudorandom bit generator is obtained from the secret key.

Linear Feedback Shift Registers (LFSRs) are important building blocks in stream cipher systems. A standard model (see Figure 1) of stream cipher [109, 110, 34] combines the outputs of several independent LFSR sequences using a nonlinear Boolean function to produce the keystream. Design and analysis of practical stream cipher was kept confidential for a long time. An important boost occurred in the 1970's, when several research papers on the design

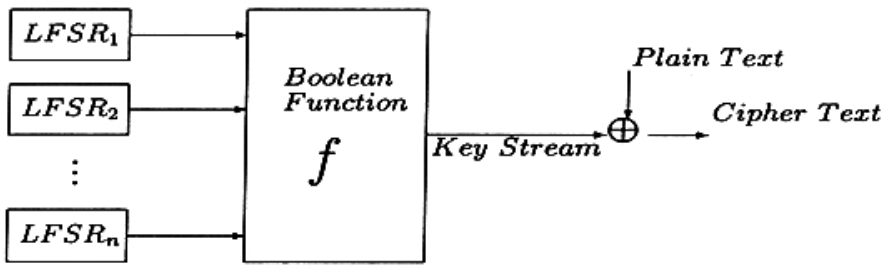


Figure 1.1: LFSR based encryption scheme

of LFSR-based stream cipher occurred. As LFSRs are linear, some form of nonlinearity is introduced by using nonlinear Boolean functions (see [100]).

Properties of the nonlinear combining Boolean function received a lot of attention in literature for the last two decades and it is now possible to get good Boolean functions which resist many of the known attacks. In this thesis we have not considered algebraic attacks as this class of attacks have become known only very recently. We consider *balancedness, nonlinearity, algebraic degree, correlation immunity and resiliency* of Boolean functions and S-boxes for use in stream ciphers model based on Figure 1.

A Boolean function used in stream cipher should be balanced, which is required for the pseudorandomness of generated keystream. In the stream cipher model, the combining Boolean function is so chosen that it increases the linear complexity [100] of the resulting key stream. High algebraic degree provides high linear complexity [101, 34]. Therefore high algebraic degree is desirable in stream ciphers. A Boolean function should have high nonlinearity to be used in stream ciphers. A function with low nonlinearity is prone to linear approximation attack. Linear approximation means approximating the combining function by a linear function. To resist *divide-and-conquer* attack a Boolean function in stream cipher should be correlation immune of higher order [109, 110]. See Chapter 2.3.1 for a more detailed discussion of these properties.

We can not achieve all the desirable properties of our liking, so there will be some trade of between these properties. Depending on the application we have to decide which properties are more important.

In *block cipher cryptography*, the message bits are divided into blocks and each block is separately enciphered using the same key and transmitted. Most of the modern day block ciphers are iterated ciphers and use substitution boxes (S-boxes) as the nonlinear part in the

scheme. The security of the block ciphers greatly depends on the strength of the substitution boxes.

Matsui [76] introduced linear cryptanalysis method for block cipher. Linear cryptanalysis means approximating a linear combination of the component functions of an S-box, used in a block cipher by a linear function of the input variables. To resist linear cryptanalysis S-boxes should have higher nonlinearity.

Differential cryptanalysis [7] is a chosen-plaintext attack and involves comparing the XOR of two inputs to the XOR of corresponding two outputs. A nonuniform output distribution will be the basis for a successful differential attack.

Webster and Tavares [118] introduced the concept of strict avalanche criteria (SAC). Propagation Characteristic (PC) and SAC are two important cryptographic properties for S-boxes to resist differential cryptanalysis. To get uniform output distribution, S-boxes in block ciphers should have PC(l) of higher order for $l \geq 1$. SAC(k) is PC(1) of order k. S-boxes having PC(l) of order k with large l and with very high nonlinearity and algebraic degree are hard to find. Therefore sometimes we may have to be satisfied with S-boxes of higher order SAC.

Jakobsen and Knudsen [58] identified interpolation attack on block ciphers with S-boxes having small algebraic degree. Later Canteaut and Videau [15] provided higher order differential attack on block ciphers using S-boxes with low algebraic degree. So algebraic degree of S-boxes should be high to resist such attacks.

Again, as stated for Boolean functions used in stream ciphers, we can not achieve all the desirable properties for S-boxes used in block ciphers. We have to decide which properties are more important depending on the application.

In writing this introduction, we have benefitted from Willi Meier's tutorial in National Workshop on Cryptology 2003 organized at Anna University, India.

1.1 Thesis Plan

This thesis is based on six papers [46, 47, 48, 49, 50, 51]. We provide a brief summary of the chapters which appear in the thesis. *Chapter 1* contains the introduction. In *Chapter 2* we provide the necessary preliminary material required in the later chapters.

In *Chapter 3,* we consider the problem of computing Walsh transform (WT) of a Boolean function from its algebraic normal form (ANF). Two standard ways of representing a Boolean

function are its truth table representation and ANF representation. Given a truth table we can compute WT by using the fast WT [73] which has complexity $O(m2^m)$, where m is the number of input variables. Hence it is useful if m is around 40 or less. Some Boolean functions have very compact ANF. An example is $g(x_1, \cdots x_{2k}) = x_1x_2 \oplus x_3x_4 \oplus \cdots \oplus x_{2k-1}x_{2k}$ which is a bent function. In general nothing can be said about the WT of a 50-variable Boolean function using the fast WT. We try to provide some answers to this problem. Clearly, it is not possible to compute the WT at all 2^m points. Suppose we want to compute WT at a small set of points. We show that this is possible in certain cases where the Boolean function has a compact ANF.

We obtain a formula for the WT of a Boolean function at a certain point in terms of parameters derived from the ANF. We simplify this formula and develop an algorithm to evaluate it to compute the WT at any point. In certain cases, It is possible to run our algorithm for 50 to 100 variable functions having a few hundred terms in their ANF. For such functions it is possible to compute the WT for a small set of points. This provides some useful information about the function such as the size of its support and an estimate of its nonlinearity. Note that for small number of variables, the fast WT is faster than our algorithm. Hence we do not provide a substitute for the fast WT; rather we provide a tool to analyse a Boolean function in situations where the fast WT cannot be used.

An important cryptanalytic method for the DES algorithm is the linear cryptanalysis method presented by Matsui in [76], which makes use of correlations between the input and output of the round functions of DES. Nyberg [86] presented three correlation theorems and applied them to partial cryptanalysis of several symmetric ciphers.

In *Chapter 4,* we continue the work of Nyberg [86] in a more theoretical direction. We consider a general functional form and obtain its WT. Two of Nyberg's correlation theorems are seen to be special cases of our general functional form. S-box look-up, addition modulo 2^{2k} and X-OR are three frequently occurring operations in the design of symmetric ciphers. We consider two methods of combining these operations and in each case apply our main result to obtain the WT. Our result have possible applications to analysis of reduced round block ciphers.

In *Chapter 5,* we construct perfect nonlinear multi-output Boolean functions satisfying higher order SAC. Our first construction is an infinite family of 2-output perfect nonlinear functions satisfying higher order SAC. This construction is achieved using the theory of bilinear forms and symplectic matrices. Next we build on a known connection between 1-factorization of a complete graph and SAC to construct more examples of 2 and 3-output perfect nonlinear functions. In certain cases, the constructed S-boxes have optimal trade-off

between the following parameters: numbers of input and output variables, nonlinearity and order of SAC. In case the number of input variables is odd, we modify the construction for perfect nonlinear S-boxes to obtain a construction for maximally nonlinear S-boxes satisfying higher order SAC. Our constructions present the first examples of perfect nonlinear and maximally nonlinear multioutput S-boxes satisfying higher order SAC. Lastly, we present a simple method for improving the degree of the constructed functions with a small sacrifice in nonlinearity and the SAC property. This yields functions which have possible applications in the design of block ciphers.

The next three chapters consider construction of resilient S-boxes and their implementation. We briefly describe each of them below.

In *Chapter 6,* we describe two constructions. In the first construction we describe a simple method using $[n-d-1, m, t+1]$ linear binary code to construct an n-input, m-output, t-resilient function with degree $d > m$ and nonlinearity $2^{n-1} - 2^{n-\lceil (d+1)/2 \rceil} - (m+1)2^{n-d-1}$. For any fixed values of parameters n, m, t and d, with $d > m$, the nonlinearity obtained by our construction is higher than the nonlinearity obtained by the only previously known construction which provides $d > m$ (due to Cheon [24]). The second method is a simple modification of a construction due to Zhang and Zheng [122] and constructs n-input, m-output resilient S-boxes with degree $d > m$. We prove by an application of the Griesmer bound for linear error correcting codes that the modified Zhang-Zheng construction is superior to the method of [24].

In *Chapter 7,* we use a sharpened version of the Maiorana-McFarland technique to construct nonlinear resilient S-boxes. The nonlinearity obtained by our construction is better than previously known construction methods. The idea is to use affine functions on small number of variables to construct nonlinear resilient functions on larger number of variables. For Boolean functions, the Maiorana-McFarland technique to construct resilient functions was introduced by Camion et al [14]. Nonlinearity calculation for the construction was first performed by Seberry, Zhang and Zheng [107]. This technique was later sharpened by Chee et al [23] and Sarkar-Maitra [103]. For S-boxes, this technique has been used by [63] and [89]. Here we develop and sharpen the technique of affine function concatenation to construct nonlinear resilient S-boxes. This leads to significant improvement in nonlinearity over that obtained in [89] and provides S-boxes with currently best known nonlinearity.

In *Chapter 8,* we consider implementation aspects of resilient S-boxes. We first discuss the applicability of resilient S-boxes to the nonlinear combiner model of stream ciphers. Next we consider the software implementation of Maiorana-McFarland resilient S-boxes. Most papers on construction of resilient Maiorana-McFarland Boolean functions and S-boxes

provide mathematical descriptions which are not sufficient for implementation purposes. Moreover, the mathematical description do not bring out the fact that in most cases such S-boxes can be efficiently implemented using a small amount of memory. Our work shows that these S-boxes can be implemented using a small amount of memory and the output of an S-box can be evaluated using very little computation.

Chapter 2

Definitions and Preliminaries

Let $\mathbb{F}_2 = GF(2)$, be the finite field of two elements. We consider the domain of an m-variable Boolean function to be the vector space $(\mathbb{F}_2^m, \oplus)$ over $\mathbb{F}_2$, where $\oplus$ is used to denote the addition operator over both $\mathbb{F}_2$ and the vector space $\mathbb{F}_2^m$. The inner product of two vectors $u, v \in \mathbb{F}_2^m$ will be denoted by $\langle u, v \rangle$. The (Hamming) distance between two vectors $x = (x_1, x_2, \cdots, x_m)$ and $y = (y_1, y_2, \cdots, y_m)$ is the number of places where they differ and is denoted by $d(x, y)$. The bitwise complement of a bit string x will be denoted by $\overline{x}$. A Boolean function g is said to be degenerate on the variable x_i if

$$g(x_1, \cdots, x_{i-1}, 1, x_{i+1}, \cdots, x_m) = g(x_1, \cdots, x_{i-1}, 0, x_{i+1}, \cdots, x_m).$$

The function g is said to be degenerate if it is degenerate on some variable, else it is said to be non-degenerate.

2.1 Boolean Functions

An m-variable Boolean function is a map $g : \mathbb{F}_2^m \to \mathbb{F}_2$. The *support* of a Boolean function g is denoted by $\mathsf{Sup}(g)$ and is defined to be $\mathsf{Sup}(g) = \{x : g(x) = 1\}$. The weight of g is denoted by $\mathsf{wt}(g)$ and is defined to be $\mathsf{wt}(g) = |\mathsf{Sup}(g)|$. A function g is said to be *balanced* if $\mathsf{wt}(g) = 2^{m-1}$. It is easy to see that there are $\binom{2^m}{2^{m-1}}$ balanced functions in the set of all m-variable Boolean functions. The (Hamming) distance $d(f, g)$ between two m-variable Boolean functions f and g is defined as

$$d(f, g) = |\{x \in \mathbb{F}_2^m : f(x) \neq g(x)\}| = \mathsf{wt}(f(x) \oplus g(x)).$$

To illustrate the different definitions we take a small example.
Example 1 : Let g be a 3-variable Boolean function whose support is given by, $\mathsf{Sup}(g) = \{(000), (001), (011), (100)\}$ so $\mathsf{wt}(g) = |\mathsf{Sup}(g)| = 4 = 2^{3-1}$ and hence g is balanced.

2.1.1 Representation of Boolean Functions

Two standard ways of representing a Boolean function are its truth table representation and ANF representation. There is an alternative representation of Boolean functions – the numerical normal form (NNF) [19]. We will not require the NNF in this thesis and hence we do not define it. In Chapter 3, we make a brief reference to the NNF.

Truth Table Representation

If we fix an enumeration of the elements of $\mathbb{F}_2^m$, then an m-variable Boolean function can be uniquely represented by a binary string of length 2^m. One standard enumeration of $\mathbb{F}_2^m$ is $\sigma(0), \ldots, \sigma(2^m - 1)$, where $\sigma(i)$ is the m-bit binary representation of i.

A truth table is tabulation of all possible combinations of input values and their corresponding outputs. The following provide an example of a 3-variable Boolean function. Note that the input variables x_3, x_2, x_1 are tabulated in each row. The function is represented in the rightmost column. For an m-variable Boolean function the truth table contains m columns for inputs, 1 column for output and 2^m rows for all the enumerations of the input variables.

x_3	x_2	x_1	f
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Table 1 : Truth table representation.

Example 1 (continued) : Table 1 provides truth table representation of the Boolean function of Example 1.

ANF Representation

An m-variable Boolean function $g(x_1, x_2, \cdots, x_m)$ can be uniquely (upto a permutation of the monomials) written as

$$g(x_1, x_2, \ldots, x_m) = \bigoplus_{(\alpha_1,\alpha_2,\ldots,\alpha_m)\in \mathbb{F}_2^m} A_g(\alpha_1, \alpha_2, \ldots, \alpha_m) x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_m^{\alpha_m} \quad (2.1)$$

where $A_g(x_1, x_2, \ldots, x_m)$ is a Boolean function. This representation is called the *algebraic normal form (ANF)* of g. The degree of the polynomial is called the algebraic degree or simply the *degree* of g and is denoted by $\deg(g)$. We will also call the Boolean function A_g the ANF of g. The maximum algebraic degree achievable for an m-variable Boolean function is m. However, such a function is not balanced. The maximum algebraic degree of a balanced function is $m-1$ (see Reed-Muller codes [73]). This can also be proved easily without going into Reed-Muller codes. An m-variable affine function is of the form $l_{u,b}(x) = \langle u, x\rangle \oplus b$, where $u \in \mathbb{F}_2^m$ and $b \in \mathbb{F}_2$. If $b = 0$, the function is called linear.

Example 1 (continued) : ANF of Boolean function of Example 1 is $1 \oplus x_2 \oplus x_1x_3 \oplus x_2x_3$ and the degree of the Boolean function is two.

2.1.2 Walsh Transform

The Walsh transform (WT) of an m-variable Boolean function g is an integer valued function $W_g : \{0,1\}^m \to [-2^m, 2^m]$ defined by (see [73, page 414])

$$W_g(u) = \sum_{w\in \mathbb{F}_2^m} (-1)^{g(w)\oplus\langle u,w\rangle}. \quad (2.2)$$

The WT is called the *spectrum* of g. We here like to mention that the Walsh transform of $g(x)$ is actually the Fourier transform of $(-1)^{g(x)}$. The conservation law for the spectral values of g is known as *Parseval's Theorem* (see [34]), which say that sum of the square of Walsh transform is constant, i.e., $\sum_{u\in F_2^m} W_f^2(u) = 2^{2m}$.

Given a truth table we can compute WT by using the fast WT algorithm [73] which has complexity $O(m2^m)$, where m is the number of input variables.

Now we explain the Fast WT (FWT) by a small example. Let us compute the FWT of the function given in Table 1. First we map the values in the truth table by $0 \to 1$ and $1 \to -1$, i.e., $g(x) \to (-1)^{g(x)}$. Next, each pair of elements is modified by an "in-place butterfly"; that is, the values in each pair produce two results which replace the original

pair, wherever they were originally located. The left result will be the two values added; the right will be the first less the second. That is, $(a', b') = (a + b, a - b)$ where (a, b) is the original pair. So for the values $(-1, 1)$, we get $(-1 + 1, -1 - 1)$ which is just $(0, -2)$. We start out pairing adjacent elements, then every other element, then every 4th element, then every eighth element and so on until the correct pairing is impossible, as shown in Figure 2.1.

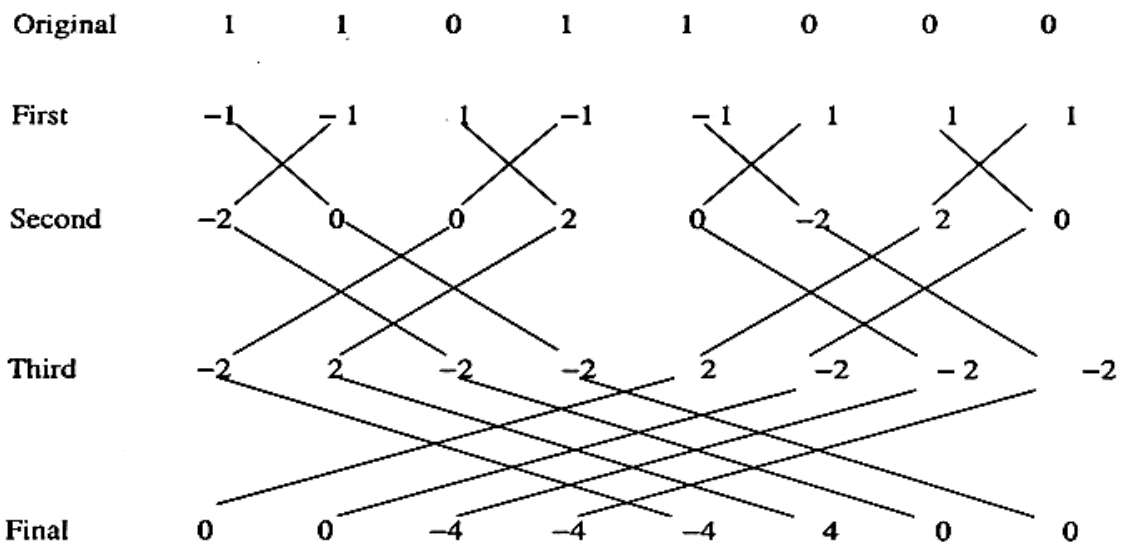


Figure 2.1: An 8-Element Fast Walsh Transform (FWT) of Example 1.

So we have the following WT.

Example 1 (continued) : WT of Example 1 is given by

u	$W_f(u)$
000	0
001	0
010	−4
011	−4
100	−4
101	4
110	0
111	0

The inverse Walsh transform is given by

$$(-1)^{g(u)} = \frac{1}{2^m} \sum_{w \in \mathbb{F}_2^m} W_g(w)(-1)^{\langle u,w \rangle}. \tag{2.3}$$

The inverse Walsh transform is also computed by a butterfly network. Let H_m be the $2^m \times 2^m$ Hadamard matrix defined recursively as (see [73, page 44]).

$$\begin{aligned} H_1 &= \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ H_m &= H_1 \otimes H_{m-1} \quad \text{for } m > 1, \end{aligned}$$

where $\otimes$ denotes the Kronecker product of two matrices. Considering the rows and columns of H_m to be indexed by the elements of F_2^m, we obtain $[H_m]_{(u,v)} = (-1)^{\langle u,v \rangle}$. Then we can write

$$[(-1)^{g(0)}, \ldots, (-1)^{g(2^m-1)}]H_m = [W_g(0), \ldots, W_g(2^m - 1)]. \tag{2.4}$$

2.1.3 Nonlinearity

A parameter of fundamental importance in cryptography is the *nonlinearity* of a Boolean function. This quantity measures the distance of a Boolean function from the set of all affine functions. Let A_m be the set of all m-variable affine functions. The nonlinearity $\mathsf{nl}(f)$ of an m-variable Boolean function is defined as $\mathsf{nl}(f) = \min_{l \in A_m} d(f, l)$. The maximum nonlinearity achievable by an m-variable Boolean function is $2^{m-1} - 2^{(m-2)/2}$. Functions achieving this value of nonlinearity are called bent and can exist only when m is even [99]. In other words, for even m, an m-variable function f is called *bent* if $W_f(u) = \pm 2^{\frac{m}{2}}$, for all $u \in \mathbb{F}_2^m$ (see [99]). When m is odd, the maximum nonlinearity achievable by an m-variable Boolean function is not known. However, functions achieving a nonlinearity of $2^{m-1} - 2^{(m-1)/2}$ are easy to construct and are called almost optimally nonlinear [16].

It is sometime convenient to express nonlinearity in terms of the spectrum of a Boolean function. The nonlinearity $\mathsf{nl}(f)$ of an m-variable Boolean function f, can be written as

$$\mathsf{nl}(f) = 2^{m-1} - \frac{1}{2} \max_{u \in \mathbb{F}_2^m} |W_f(u)|.$$

Example 1 (continued) : Nonlinearity of the Boolean function of Example 1 is $4 - 2 = 2$.

2.1.4 Resiliency

An m-variable function is called *correlation immune* of order t (t-CI) if $W_g(u) = 0$ for all u with $1 \leq \mathrm{wt}(u) \leq t$ [109, 119]. Further the function is balanced if and only if $W_g(0) = 0$. A balanced t-CI function is called *t-resilient*. For t-CI functions the algebraic degree d is bounded by $d \leq m - t$ [109]. Also for t-resilient Boolean functions the algebraic degree d is bounded by $d \leq m - t - 1$ [109].

Example 1 (continued) : The Boolean function of Example 1 is not *correlation immune* as $W_g(001) = 0$.

If two functions are highly correlated, then they are "close" to each other and can be approximated one for the other. The correlation between two m-variable Boolean functions f and g is defined in the following manner (see for example [86]).

$$c(f, g) = 2^{-m} \sum_{x \in \mathbb{F}_2^m} (-1)^{f(x) \oplus g(x)}. \tag{2.5}$$

We have the following relationship $c(f, g) = 2^{-m} W_{f \oplus g}(0)$ and $c(f, l_v) = 2^{-m} W_f(v)$, where l_v is the linear function defined as $l_v(x) = \langle v, x \rangle$ for $x \in \mathbb{F}_2^m$. Thus correlation between a Boolean function and a linear function can be conveniently studied using Walsh transform analysis.

2.1.5 Strict Avalanche Criteria

Webster and Tavares [118] introduced the concept of strict avalanche criteria (SAC). A more general concept is Propagation Characteristic (PC) [94]. An m-variable Boolean function f satisfies

1. SAC if $f(x) \oplus f(x \oplus \alpha)$ is balanced for any $\alpha \in \mathbb{F}_2^m$ such that $\mathrm{wt}(\alpha) = 1$ [118].
2. SAC(k), if every subfunction obtained from $f(x_1, \cdots, x_m)$ by keeping at most k input bits constant satisfies SAC.
3. PC(l), if $f(x) \oplus f(x \oplus \alpha)$ is balanced for any α such that $1 \leq \mathrm{wt}(\alpha) \leq l$.
4. PC(l) of order k, if any function obtained from f by keeping any k input bits constant satisfies PC(l).

Example 1 (continued) : The Boolean function of Example 1 satisfies SAC as $f(x) \oplus f(x \oplus \alpha)$ is balanced for $\alpha = 001, \alpha = 010, \alpha = 100$. Also f does not satisfy SAC(1) since if we fix $x_3 = 0$ in truth table of example 1, and take $\alpha = 001$, then $f(x) \oplus f(x \oplus \alpha)$ is not balanced.

2.2 S-Boxes

An (n, m) S-box (or vectorial function) is a map $f : \{0,1\}^n \rightarrow \{0,1\}^m$. An S-box $f : \{0,1\}^n \rightarrow \{0,1\}^m$ has component functions $f_1, \ldots, f_m$. Let $f : \{0,1\}^n \rightarrow \{0,1\}^m$ be an S-box and $g : \{0,1\}^m \rightarrow \{0,1\}$ be an m-variable Boolean function. The composition of g and f, denoted by $g \circ f$ is an n-variable Boolean function defined by $(g \circ f)(x) = g(f(x))$.

2.2.1 S-Box Representation

Truth table representation and ANF representation of component functions of S-boxes.

S-box Representation by Truth Table

A truth table is tabulation of all possible combinations of input values and their corresponding outputs. The following provide an example of a 3-input, 2-output S-box. Input variables are x_3, x_2, x_1 and component functions are f_1 and f_2.

x_3	x_2	x_1	f_1	f_2
0	0	0	1	1
0	0	1	1	0
0	1	0	0	0
0	1	1	1	1
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	0	0

We give another example of $(5, 4)$ S-box G with component functions G_1, G_2, G_3 and G_4. The output of G of any 5-bit input is given by a 4-bit string which can be represented as a hexadecimal digit. We can write G as a 32-tuple of hexadecimal digits having the following

form:

$$G = (0, 0, 4, 7, 5, F, B, 2, D, C, 1, 3, 8, F, 2, A, 7, 9, B, 6, C, 8, A, D, 1, E, 5, 9, 6, 3, 8, D)$$

S-box Representation by ANF

Let G be the $(5, 4)$ S-box above with component functions G_1, G_2, G_3 and G_4. The ANFs of its component functions are

$$\begin{aligned}
G_1(y_1, \cdots, y_5) &= y_4 \oplus y_1y_3 \oplus y_1y_5 \oplus y_2y_3 \oplus y_2y_4 \oplus y_2y_5 \oplus y_3y_4 \oplus y_3y_5 \oplus y_4y_5 \\
G_2(y_1, \cdots, y_5) &= y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_1y_5 \oplus y_3y_4 \oplus y_3y_5 \\
G_3(y_1, \cdots, y_5) &= y_5 \oplus y_1y_2 \oplus y_1y_3 \oplus y_2y_3 \oplus y_1y_5 \oplus y_3y_5 \oplus y_4y_5 \\
G_4(y_1, \cdots, y_5) &= y_3 \oplus y_4 \oplus y_5 \oplus y_1y_2 \oplus y_1y_4 \oplus y_4y_5
\end{aligned}$$

We define the degree of an (n, m) S-box f to be the minimum of the degrees of $l \circ f$, where l ranges over all non constant m-variable linear functions. This definition is more meaningful to cryptography than the definition where the degree of an S-box is taken to be the maximum of the degrees of all the component functions. The later definition has been used in [15]. In the example above the *degree* of G is 2.

2.2.2 Nonlinearity of S-boxes

Let f be an (n, m) S-box. The nonlinearity of f is defined to be

$$\mathsf{nl}(f) = \min\{\mathsf{nl}(l \circ f) : l \text{ is a non-constant } m\text{-variable linear function}\}.$$

The maximum achievable nonlinearity of an n-variable function is $2^{n-1} - 2^{(n-2)/2}$ and S-boxes achieving this value of nonlinearity are called perfect nonlinear S-boxes. For $m = 1$, perfect nonlinear S-boxes are bent functions described earlier. Such S-boxes exist only if n is even and $m \leq (n/2)$ [83]. For odd n and $m = n$, the maximum possible nonlinearity achievable is $2^{n-1} - 2^{(n-1)/2}$ and S-boxes achieving this value of nonlinearity are called maximal nonlinear S-boxes. For odd n and $1 \leq m < n$, the maximum possible achievable nonlinearity is an open problem. However, for odd n, $1 < m < n$, and quadratic (degree two) functions the maximum possible achievable nonlinearity is $2^{n-1} - 2^{(n-1)/2}$. We will also call such functions to be maximally nonlinear. In the example above the *nonlinearity* of G is $2^{5-1} - 2^{(5-1)/2} = 16 - 4 = 12$.

2.2.3 Resiliency of S-boxes

An (n, m) S-box f is said to be balanced if $l \circ f$ is balanced for every non-constant m-variable linear function l. An (n, m) S-box f is said to be t-CI, if $l \circ f$ is t-CI for every non-constant m-variable linear function l (see [122]). Further, if f is balanced then f is called t-resilient. By an (n, m, t) S-box we mean t-resilient (n, m) S-box. Similarly by an (n, m, t, d) S-box (or (n, m, t, d)-resilient function) we mean t-resilient (n, m) S-box with algebraic degree d.

2.2.4 SAC of S-boxes

An (n, m) S-box f is said to satisfy SAC(k), if $(l \circ f)$ is SAC(k) for every non-constant m-variable linear function l. Now we give an example of an $(8, 2)$ S-box $f = (f_1, f_2)$ which is SAC(3) with algebraic degree two. The component functions are

$$\begin{aligned} f_1(x_1, \ldots, x_8) &= x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_1x_5 \oplus x_1x_7 \oplus x_2x_3 \oplus x_2x_4 \oplus x_2x_6 \oplus x_2x_8 \oplus x_3x_5 \\ &\oplus x_3x_7 \oplus x_3x_8 \oplus x_4x_6 \oplus x_4x_7 \oplus x_4x_8 \oplus x_5x_6 \oplus x_5x_7 \oplus x_5x_8 \oplus x_6x_7 \oplus x_6x_8. \end{aligned}$$

and

$$\begin{aligned} f_2(x_1, \ldots, x_8) &= x_1x_2 \oplus x_1x_4 \oplus x_1x_6 \oplus x_1x_8 \oplus x_2x_3 \oplus x_2x_5 \oplus x_2x_7 \oplus x_3x_4 \oplus x_3x_6 \oplus x_3x_7 \\ &\oplus x_4x_5 \oplus x_4x_6 \oplus x_5x_6 \oplus x_5x_7 \oplus x_5x_8 \oplus x_6x_8 \oplus x_7x_8. \end{aligned}$$

2.3 Cryptographic Motivation

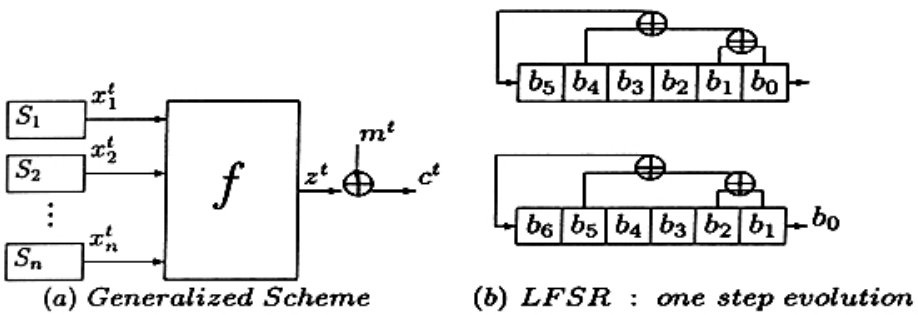
We consider cryptographic significance of resiliency, nonlinearity, SAC and algebraic degree.

2.3.1 Significance of Correlation Immunity

To resist *divide-and-conquer* attack a function in stream cipher should be correlation immune of higher order [109, 110]. First we explain the basic model (nonlinear combiner model) and then describe the idea of correlation attack. We also explain LFSRs used in this model.

Nonlinear Combiner Model

The cryptographic paradigm we consider here is a private key system. The sender and receiver both have the same key. The sender encrypts the message with a key and the



(a) Generalized Scheme *(b) LFSR : one step evolution*

Figure 2.2: LFSR based encryption scheme

receiver decrypts the cipher with the same key. The attacker taps the channel between the sender and receiver and grabs a portion of the cipher. The target of the attacker is then to recover the key from the cipher he possesses. See Figure 2.2a for the scheme. At the sender side, the message text is encoded to binary stream using some coding scheme. Depending on the key a binary keystream is generated. Each bit of the message is XOR-ed to a corresponding keystream bit and a cipher bit is generated. This cipher bit stream is transmitted through the communication channel. The receiver side possesses the same key as the sender. Hence at the receiver side same keystream bit sequence is generated. Each bit of the cipher stream is XOR-ed with the corresponding keystream bit and thus the message is recovered.

LFSRs are used as running key subgenerators in stream ciphers. See Figure 2.2b for one step evolution of an LFSR. Note that $b_i \in \{0, 1\}$. The LFSR in the Figure 2.2b is of length 6. It implements a recurrence relation of the form $x_n = x_{n-2} \oplus x_{n-5} \oplus x_{n-6}$ where $x_i \in \{0, 1\}$ and $\oplus$ represents the GF(2) sum (addition mod 2). The initial condition of the LFSR is $b_5b_4b_3b_2b_1b_0$. The state after one clock is $b_6b_5b_4b_3b_2b_1$ where $b_6 = b_4 \oplus b_1 \oplus b_0$ and the output is b_0. The recurrence relation may also be represented by the *connection polynomial* [82, Page 196], which is $1 + x^2 + x^5 + x^6$ for the LFSR in Figure 2.2b.

The n subgenerators S_i, (Figure 2.2a), $i = 1, 2, \ldots, n$ are usually realized by LFSRs, where the feedback connection polynomials are taken to be primitive [34, 100] over GF(2). The initial conditions of n subgenerators (LFSRs) are $k_1, k_2, \ldots, k_n$ respectively, where each k_i is a bit string of length d_i.

A Boolean function f is used as a nonlinear combining function on n inputs $x_1, x_2, \ldots x_n$ with $z = f(x_1, x_2, \ldots, x_n)$ (Figure 2.2a). We will denote the n input sequences to f by

$x_1^t, \ldots, x_n^t$. The keystream sequence $z^0, z^1, z^2, z^3, \ldots$ is determined by $z^t = f(x_1^t, \ldots, x_n^t)$ for $t \geq 0$. The plaintext message sequence is m^t. This message sequence m^t is bitwise XOR-ed with z^t to produce the ciphertext sequence c^t. Deciphering is done by performing a bitwise XOR of ciphertext sequence with the keystream sequence produced by the same f and same initial conditions $k_1, k_2, \ldots, k_n$ of the LFSRs $S_1, S_2, \ldots, S_n$. Thus, the encryption is $c^t = m^t \oplus z^t$ and the decryption is $m^t = c^t \oplus z^t$.

Correlation attack

In the stream cipher model of Figure 2.2, ciphertext only attacks need to identify the initial conditions $k_1, k_2, \ldots, k_n$ for the n LFSRs $S_1, S_2, \ldots, S_n$. Let M_i be the number of different initial conditions k_i for the LFSR S_i. So the total number of different initial conditions are $M = \prod_{i=1}^{n} M_i$. The size of the ith LFSR being d_i, $M_i = 2^{d_i} - 1$ (all zero condition need not be considered). This implies that it is practically infeasible to check all the keys by a brute force attack [110]. Siegenthaler [110] proposed a divide and conquer attack on this kind of stream cipher model. If the combining function f is not correlation immune, a cryptanalysis scheme may separately attack each keystream generator S_i individually to find the initial condition k_i. So in this case the number of trials is $M' = \sum_{i=1}^{n} M_i$, which may be realistic.

Now we describe the basic idea of correlation attack which was originally proposed by Siegenthaler [110] and then continue with Example 1. For $u \in \mathbb{F}_2^n$, let $l_u(x_1, \ldots, x_n)$ be a linear function defined as

$$l_u(x_1, \ldots, x_n) = \langle u, (x_1, \ldots, x_n) \rangle.$$

Suppose $W_f(u) \neq 0$ for some $u \in \mathbb{F}_2^n$, with $\mathsf{wt}(u) = 1$. Let i be such that $u_i = 1$ and for $j \neq i$, $u_j = 0$. In this situation first order correlation attacks are applicable. The function $l_u(x_1, \ldots, x_n)$ is equal to x_i. The basic concept is to use the bias $\beta_u = |\mathsf{Prob}(l_u = f) - \frac{1}{2}| = \frac{|W_f(u)|}{2^{n+1}}$ to estimate the sequence x_i^t from the sequence z_i^t or even from the cipher sequence c_i^t. Let $\mathsf{Prob}(l_u = f) = p_1 \neq \frac{1}{2}$ (since $W_f(u) \neq 0$) and $\mathsf{Prob}(m^t = 0) = p_2$, then $\mathsf{Prob}(c^t = x_i^t) = p_1 p_2 + (1 - p_1)(1 - p_2)$. If $p_2 \neq \frac{1}{2}$ then we can apply correlation attack. The message text is encoded to binary stream using some coding scheme and generally $p_2 \neq \frac{1}{2}$.

Example 1 (continued) : Note $W_f(010) = -4 \neq 0$, $\beta_u = \frac{1}{4} \neq 0$ and $p_1 = \frac{1}{4} \neq \frac{1}{2}$. So

$$\mathsf{Prob}(c^t = x_2^t) = p_1 p_2 + (1 - p_1)(1 - p_2) = \frac{3}{4} - \frac{1}{2} p_2 \neq \frac{1}{2}$$

assuming $p_2 \neq \frac{1}{2}$.

If $\beta_u = 0$ i.e., $W_f(u) = 0$ for all u with $\mathsf{wt}(u) = 1$, then we can not estimate any x_i^t from z^t. Then a higher order correlation attack can be applied. Suppose f is m-CI but not $(m+1)$-CI. Then there exists $u \in \mathbb{F}_2^n$ with $\mathsf{wt}(u) = m+1$ such that $W_f(u) \neq 0$. Let $i_1, \ldots, i_{m+1}$ be such that $u_{i_1} = \cdots = u_{i_{m+1}} = 1$ and $u_j = 0$ for $j \notin \{i_1, \ldots, i_{m+1}\}$. Then the β_u is used to estimate the sequence $y^t = x_{i_1}^t \oplus \cdots \oplus x_{i_{m+1}}^t$. The individual sequences $x_{i_1}^t, \cdots, x_{i_{m+1}}^t$ can be obtained from y^t by solving a system of linear equation.

2.3.2 Significance of Nonlinearity

Linear cryptanalysis [76] is a very powerful cryptanalytic method for block ciphers. The study of correlation between linear combinations of input and output of an S-box is therefore very important. A function with low nonlinearity is prone to linear approximation attack. Linear approximation means approximating the combining function by a linear function. Thus for symmetric cipher applications we need functions (both Boolean functions and S-boxes) with high nonlinearity.

Apart from its importance in cryptography, highly nonlinear Boolean functions are important combinatorial objects by themselves and have close relationship with coding theory [73].

Linear Cryptanalysis and Nonlinearity in Block Cipher

The discussion in this section is based on [114]. Nonlinearity is an important parameter to resist linear cryptanalysis. We describe the strategy behind linear cryptanalysis. It is possible to find a probabilistic linear relationship between a subset of plaintext bits and a subset of state bits immediately preceding the substitution performed in the last round. When the nonlinearity properties of the S-box are enumerated, it is possible to develop linear approximations between sets of input and output bits in the S-box. Consequently it is possible to concatenate linear approximations of the S-boxes together so that intermediate state bits cancel out and we are left with a linear expression which has a large bias and involves only plaintext and last round input bits. In other words, there exists a subset of bits whose XOR behaves in nonrandom fashion. Assume that an attacker has a large number of plaintext-ciphertext pairs, all of which are encrypted using the same unknown key. For each of the plaintext-ciphertext pairs, we will begin to decrypt the ciphertext, using all possible candidate keys for the last round of the cipher. For each candidate key, we compute the values of the relevant state bits involved in the linear relationship, and determine if the above mentioned linear relationship holds. Whenever it does, we increment a counter

corresponding to the particular candidate key. At the end we hope that the candidate key that has a frequency count that is furthest away from $\frac{1}{2}$ times the number of pairs contains the correct values for the key bits.

Piling-up Lemma

Let $X_1, X_2, \ldots$ be independent random variables taking on the values from the set $\{0, 1\}$ and suppose $\mathsf{Prob}(X_i = 0) = p_i$ for $i = 1, 2, \ldots$

It is often convenient to express probability distribution of a random variable in terms of the bias of the distribution. The bias of X_i is defined to be the quantity $\epsilon_i = p_i - \frac{1}{2}$ (see [114]).

Let $\epsilon_{i_1,i_2,\ldots i_k}$ denote the bias of the random variable $X_{i_1} \oplus \cdots \oplus X_{i_k}$. The following result is known as *piling-up lemma* [114]. Let $\epsilon_{i_1,i_2,\ldots i_k}$ denote the bias of the random variable $X_{i_1} \oplus \cdots \oplus X_{i_k}$. Then

$$\epsilon_{i_1,i_2,\ldots i_k} = 2^{k-1} \prod_{j=1}^{k} \epsilon_{i_j}.$$

Consider an S-box $f : \{0,1\}^n \to \{0,1\}^m$. We write input n-tuple as $x = (x_1, \ldots, x_n)$. Each coordinate x_i defines a random variable X_i having bias $\epsilon_i = 0$. Also these random variables are independent. Let output n-tuple be $y = (y_1, \ldots y_m)$. Each coordinate y_j defines a random variable Y_j. These m random variables are in general not independent. The probability that the random variable

$$X_{i_1} \oplus \cdots \oplus X_{i_k} \oplus Y_{j_1} \oplus \cdots \oplus Y_{j_l}$$

takes on value 0 can be computed by counting the number of rows of the truth table of the S-box in which $X_{i_1} \oplus \cdots \oplus X_{i_k} \oplus Y_{j_1} \oplus \cdots \oplus Y_{j_l} = 0$ and then dividing it by 2^n.

An S-box S_i is said to be *active* in round k if S_i has nonzero input and is used in linear approximation. Let $T_{i_1}, \ldots, T_{i_k}$ be the random variables associated with active S-boxes and $\epsilon_{i_1}, \ldots, \epsilon_{i_k}$ be there biases. We can compute the bias (say ϵ) of $T_{i_1} \oplus \cdots \oplus T_{i_k}$ by piling-up lemma.

In general a linear attack based on a linear approximation having bias ϵ will be successful if the number of plaintext-ciphertext pairs, is approximately $c\epsilon^{-2}$, for some small constant c. The motivation of using S-boxes with high nonlinearity is that S-boxes having high nonlinearity will have low value of ϵ. Hence linear cryptanalysis may require very large number of plaintext-ciphertext with same secret key and may be practically impossible.

Cryptanalysis and Nonlinearity in Stream Cipher

In nonlinear combiner model the combining Boolean function f should have high nonlinearity. As discussed in subsection 2.3.1 suppose the combining Boolean function is m-CI but not $(m+1)$-CI. Then there exists a u with $\mathsf{wt}(u) = m+1$ such that $W_f(u) \neq 0$. The notations used here are explained in subsection 2.3.1. Let $i_1, \ldots, i_{m+1}$ be such that $u_{i_1} = \cdots = u_{i_{m+1}} = 1$ and $u_j = 0$ for $j \notin \{i_1, \ldots, i_{m+1}\}$. The attacker will choose the bias at point u so that $\beta_u = |\mathsf{Prob}(l_u = f) - \frac{1}{2}| = \frac{|W_f(u)|}{2^{n+1}}$, is maximum, and is used to estimate the sequence x_i^t from the sequence z_i^t or even with the cipher sequence c_i^t. In other words the β_u is used to estimate the sequence $y^t = x_{i_1}^t \oplus \cdots \oplus x_{i_{m+1}}^t$. The individual sequences $x_{i_1}^t, \cdots, x_{i_{m+1}}^t$ can be obtained from y^t by solving a system of linear equation.

To resist the above attack the value $\beta_u = \frac{|W_f(u)|}{2^{n+1}}$ should be low. To ensure this, the nonzero values of the Walsh spectra should be more or less uniform, so that the maximum absolute value of Walsh spectra is minimized, as Parseval's theorem [34] say that sum of the square of Walsh transform is constant. If the maximum of absolute values of Walsh spectra is low the Boolean function has high nonlinearity. So to resist this kind of attack f should have high nonlinearity.

2.3.3 Significance of SAC

The security of block ciphers can be analysed by viewing the S-boxes as a set of Boolean functions. To resist differential cryptanalysis S-boxes should have higher order of SAC.

Overview of Differential Attack

This discussion of differential attack is mainly from [114]. We turn our focus to the application of differential cryptanalysis to the basic substitution-permutation network (SPN) [114] cipher. Differential cryptanalysis exploits the high probability of certain occurrences of plaintext differences and differences into the last round of the cipher. For example, consider a system with input $x = [x_1, x_2, \ldots, x_n]$ and output $y = [y_1, y_2, \ldots y_n]$. Let two inputs to the system be x and x^* with the corresponding outputs y and y^* respectively. The input difference is given by $x' = x \oplus x^*$. Similarly $y' = y \oplus y^*$ is the output difference.

In an ideally randomizing cipher, the probability that a particular output difference y' occurs given a particular input difference x' is $\frac{1}{2^n}$ where n is the number of bits of x. Differential cryptanalysis seeks to exploit a scenario where a particular y' occurs given a

particular input difference x' with a very high probability (i.e., much greater than $\frac{1}{2^n}$). The pair (x', y') is referred to as a *differential.*

Differential cryptanalysis is a chosen plaintext attack, meaning that the attacker is able to select inputs and examine outputs in an attempt to derive the key. For differential cryptanalysis, the attacker will select pairs of inputs, x and x^* , to satisfy a particular x', knowing that for a specific x' value, a particular y' value occurs with high probability.

Now we provide some definitions.

Let $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$, be an S-box and let $a, a^* \in \mathbb{F}_2^n$. The difference $a' = a \oplus a^*$ is said to propagate to the difference $b' = f(a) \oplus f(a')$ though f. This is denoted by $a' \xrightarrow{f} b'$. An expression of the form $a' \xrightarrow{f} b'$ is called a *differential.*

The *propagation ratio* R_p of the differential $a' \xrightarrow{f} b'$ is defined by

$$R_p(a', b') = 2^{-n} |\{x \in \mathbb{F}_2^n | f(x) \oplus f(x \oplus a') = b'\}|.$$

A table that shows the distribution of the input XORs and output XORs of all the possible pairs of an S-box is called the *difference distribution table* of the S-box.

Given an S-box it is easy to construct difference distribution table and hence it is simple to calculate propagation ratio. Suppose we find propagation ratios for differentials in consecutive rounds of the SPN, such that input XOR of a differential in any round is the same as the permuted output XORs of the differentials in the previous round. Then these differentials can be combined to form a *differential trail.* It is assumed that the various propagation ratios in a differential trail are independent. This assumption allows us to multiply the propagation ratios of the differentials in order to obtain the propagation ratio of the differential trail.

An S-box S_i is said to be *active* in round k if it has non-zero input XOR in round k. The less is the number of active S-boxes the higher is propagation ratio of the differential trail.

Differential Cryptanalysis and SAC Property

The *motivation* of SAC is the following. Suppose the S-boxes satisfy SAC. Also suppose some S-boxes in differential trail have input XOR of weight one. Then the output of these S-boxes will have uniform distribution i.e., propagation ratios will be low, making the overall propagation ratio of the differential trail less and this will resist some form of differential attack. The generalization is to use S-box with PC(l). In that case that weight of input

XOR of any active S-box should be at least l, otherwise output distribution will be uniform which is almost impossible for higher value of l.

The motivation of using S-boxes satisfying SAC(k) is that, if we fix k or less bits of input XOR in first round of SPN, even then the output XOR will be uniformly distributed and hence decrease the propagation ratio. Thus by using S-boxes satisfying SAC(k) with higher value of k we can resist differential cryptanalysis.

2.3.4 Significance of Degree

High algebraic degree resists certain attacks and is therefore desirable in both stream and block ciphers.

Cryptanalysis and Algebraic degree in Stream Cipher

In the stream cipher model, the combining function f is so chosen that it increases the linear complexity [100] of the resulting key stream. High algebraic degree provides high linear complexity [101, 34]. The linear complexity of a binary keystream is the length of the shortest LFSR which is able to generate the keystream. For secure stream cipher systems large linear complexity is necessary.

As shown in Figure 2.2a, let the lengths of n LFSRs be $l_1, l_2, \ldots, l_n$ where the l_i's are distinct. A Boolean function f is used as a nonlinear combining function on n inputs $x_1, x_2, \ldots x_n$ with $z = f(x_1, x_2, \ldots, x_n)$. The linear complexity of the output sequence z is equal to $f(l_1, l_2, \ldots, l_n)$ where the ANF of f is evaluated over integers (see [101]). The above fact shows that $(1 + l_1) \ldots (1 + l_n)$ is an upper bound on the maximum possible linear complexity in the nonlinear combiner model.

Now we provide a simple example (see [100, 101]) for more details). Let us consider a function $f(x_1, x_2, x_3, x_4) = x_1x_2 \oplus x_3x_4$. If the linear complexities of four LFSRs (as in Figure 2.2a) are 5, 6, 7, 8 respectively, then the linear complexity of the output sequence z will be $5 \times 6 + 7 \times 8 = 86$. Suppose we consider another function $g(x_1, x_2, x_3, x_4) = x_1 \oplus x_2x_3x_4$. then the linear complexity of the output sequence z will be $5 + 6 \times 7 \times 8 = 341$. This clearly underlines the need for high algebraic degree of a Boolean function.

Algebraic attacks [28] are a new type of attack on stream ciphers. These attacks exploit the fact that even if a function may have high degree it may have a low degree multiple. In this thesis, we have not considered algebraic attacks. In Chapter 8, we make a brief reference

to the algebraic attacks.

Cryptanalysis and Algebraic degree in Block Cipher

The differentials used by Biham and Shamir [7] in their differential attacks correspond to the first order derivative. Later Knudsen [64] used higher order differentials to cryptanalyze ciphers presumably secure against conventional differential attacks, i.e., attacks based on first order differentials.

A round function f in iterated block cipher leads to an upper bound on the degree of the function $f \circ f$ which grows much slower than $\deg(f)^2$ [15]. Therefore iterated block ciphers may be vulnerable to higher order differential attack if the round function has low degree.

Let f be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$. The derivative of f with respect to a is the function

$$D_a f(x) = f(x) \oplus f(x \oplus a).$$

The i-th derivative of f at the point $a_1, \ldots, a_i$ is defined as

$$D^{(i)}_{a_1,\ldots,a_i} f(x) = D_{a_i}(D^{i-1}_{a_1,\ldots,a_{i-1}} f(x)).$$

It is proved [71] that if $D^{(i)}_{a_1,\ldots,a_i} f(x)$ is not constant then $\deg(D^{(i)}_{a_1,\ldots,a_i} f(x)) \leq \deg(f) - i$. *The design principle for the S-boxes to be used in block cipher* should be such that, for each small i, the non trivial i-th derivatives of function should take on each possible value uniformly i.e., i-th derivatives of function should take on each possible value with probability about 2^{i-n} [71].

Higher Order Differential Cryptanalysis [15, 64]

For any i-dimensional vector subspace V of $\mathbb{F}_2^n$, the i-th derivative of F with respect to V is the function

$$D_V f(x) = D^{(i)}_{a_1,\ldots,a_i} f(x)$$

where $a_1, \ldots, a_i$ is any basis of V. We know (see [71]) for any $x \in \mathbb{F}_2^n$,

$$D_V f(x) = \bigoplus_{v \in V} f(x \oplus v).$$

Suppose we consider r round iterated block cipher with block size n and round function f. Reduced cipher is the cipher obtained by removing the final round of the original cipher.

The reduced cipher correspond to the function $g = f_{K_{r-1}} \circ \ldots \circ f_{K_1}$, where K_i is the ith round key. Suppose g has degree at most i, then there exists a i-dimensional subspace V of $\mathbb{F}_2^n$, such that $D_V g(x) = c$ for all $x \in \mathbb{F}_2^n$ (see [71]) where c is a constant in $\mathbb{F}_2^n$ which does not depend on the round keys $K_1, \cdots, K_{r-1}$. Then for any round keys the reduced cipher g satisfies

$$\bigoplus_{x \in V} g(x \oplus v) = c \tag{2.6}$$

for all $x \in \mathbb{F}_2^n$.

This gives us the following chosen plaintext attack.

1. Select a random plaintext $x_0 \in \mathbb{F}_2^n$ and get the plaintexts $x_0 \oplus v$ and ciphertexts c_v corresponding to all $v \in V$.
2. Compute c by equation 2.6 to the reduced cipher with $K_1 = \cdots = K_{r-1} = 0$.
3. For each candidate round key k_r, compute

$$\sigma(k_r) = \bigoplus_{x \in V} f_{k_r}^{-1}(c_v).$$

The key k_r with $\sigma(k_r) = c$ is the correct last round key with high probability. If the attack returns several round keys, it could be repeated with different values of x_0. The attack corresponds to 2^{m+i} evaluations of f^{-1}, where m is the size of the last round key. It requires the knowledge of 2^i chosen plaintexts. The maximum value of i is degree of f. So the degree of S-boxes used in iterated block ciphers should be high to resist higher order differential attacks.

Interpolation attack [58]

Let F be a field and let $2s$ elements $x_1, \ldots, x_s, y_1, \ldots, y_s \in F$ be given, where x_i's are distinct. Define

$$f(x) = \sum_{i=1}^{s} y_i \prod_{1 \le j \le s, j \ne i} \frac{x - x_j}{x_i - x_j} \tag{2.7}$$

Then $f(x)$ is the polynomial over the field F of degree at most $s-1$ such that $f(x_i) = y_i$ for $i = 1, \ldots, s$. This formula is known as Lagrange interpolation formula. Now we explain the

outline of interpolation attack. Consider an m-bit secret key block cipher. The ciphertext y can be described as a polynomial $p(x) \in GF(2^m)[x]$. If the number of coefficients of this polynomial is sufficiently low, one can reconstruct it with sufficiently many plaintext-ciphertext pair by solving a set of linear equations. Hence we have an algorithm which can encrypt and decrypt plaintexts and ciphertexts, without knowledge of secret key.

Now we consider key-recovery variant of interpolation attack. We express the output from the reduced cipher as a polynomial $p(x) \in GF(2^m)[x]$. Let this polynomial be of degree d and we have $(d+1)$ plaintext-ciphertext pairs. Then for all possible values of last round key one decrypts the d ciphertexts one round and tries to construct the polynomial. With $(d+1)$th plaintext-ciphertext pair one checks whether the polynomial is correct. If this is the case, then the correct value of the last round key has been found with a high probability. It is known [58], that there exists an interpolation attack of average time complexity $2^{b-1}(s+1)$ requiring $s+1$ chosen plaintexts, where b denotes the number of last round key bits and s denotes the number of coefficients in the polynomial. The maximum number of coefficients is $d+1$, where d is the degree of the polynomial.

The above attacks motivate the requirement for high algebraic degree of S-boxes for block cipher applcations.

Chapter 3

Computing Partial Walsh Transform from ANF

In this chapter we study the relationship between the Walsh transform and the algebraic normal form of a Boolean function. At first we carry out a combinatorial analysis to obtain a formula for the Walsh transform at a certain point in terms of parameters derived from the algebraic normal form. The second part is devoted to simplify this formula and develop an algorithm to evaluate it. Our algorithm can be applied in situations where it is practically impossible to use the fast Walsh transform algorithm. Experimental results show that under certain conditions it is possible to execute our algorithm to evaluate the Walsh transform (at a small set of points) of functions on a few scores of variables having a few hundred terms in the algebraic normal form.

3.1 Introduction

One of the most useful tools for the study of Boolean functions is the Walsh transform, which is essentially the Fourier transform applied to the function $(-1)^{g(x)}$. The Walsh transform measures the correlations between an m-variable Boolean function and all the m-variable linear functions. These correlations uniquely determine the function and hence it is possible to work entirely with the Walsh transform. In fact, many properties of Boolean functions are most easily stated in terms of Walsh transform.

For practical applications it is often useful to be able to compute the Walsh transform of a Boolean function. It turns out that there is an excellent algorithm to do so, namely,

the fast Walsh transform [1] (see Section 2.1.2 for an exposition of the fast Walsh transform algorithm). For an m-variable function the fast Walsh transform takes time $O(m2^m)$ and hence can be used for functions of around 30 variables. The fast Walsh transform is computed from the description of the function itself. More precisely, the fast Walsh transform takes as input the bit string $f(\sigma_0)\ldots f(\sigma_{2^m-1})$ of length 2^m, where for $0 \leq i \leq 2^m - 1$, σ_i is the m-bit binary representation of i.

There is another way to uniquely represent a Boolean function, namely by its algebraic normal form, which expresses a Boolean function as a multivariate polynomial over $\mathbb{F}_2$, the finite field of two elements (see Section 2.1.1). The number of nonzero terms in the polynomial can be 2^m in the worst case. However, many interesting classes of Boolean functions have compact algebraic normal form representation. (For example, an $m = 2k$-variable bent function can have as few as k many terms in their ANF.)

In this chapter, we study the relationship between the algebraic normal form and the Walsh transform of a Boolean function. We obtain a formula for the Walsh transform of a Boolean function at a point $v \in \mathbb{F}_2^m$ in terms of certain parameters derived from the algebraic normal form. We present an efficient algorithm to evaluate the formula and hence compute the Walsh transform at v.

Our algorithm can be used to compute the Walsh transform in cases where it is not possible to use the fast Walsh transform. For example, it is possible in certain cases to run our algorithm for 50 to 100 variable functions having a few hundred terms in their algebraic normal form. For such functions it is possible to compute the Walsh transform for a small set of points v. Note that it is practically impossible to compute the Walsh transform of an m-variable function at all points in $\mathbb{F}_2^m$ if m is around 50 or more. Our algorithm provides a method to probe the spectral domain of large variable functions. This will provide some useful information about the function such as the size of its support and an estimate of its nonlinearity. Note that for small number of variables, the fast Walsh transform is faster than our algorithm. Hence we do not provide a substitute for the fast Walsh transform; rather we provide a tool to analyse a Boolean function in situations where the fast Walsh transform cannot be used.

Carlet and Guillot [19] study an alternative representation of Boolean functions - the numerical normal form (NNF). In Theorem 5 of [19], they obtain a formula for computing the NNF from the ANF representation and in Equation (6) of [19], they obtain a formula for computing the Walsh transform from the NNF representation. Using these two results, it is possible to obtain a formula for computing the Walsh transform from the ANF representation. In fact, in principle this formula can be used to derive the explicit relationship

between the Walsh transform and ANF that we obtain in this chapter. However, carrying out this task appears to be a non-trivial exercise. More importantly, we do more than just obtain the relationship between the Walsh transform and the ANF. We analyse this relationship and ultimately obtain an algorithm to compute the Walsh transform from the ANF. Obtaining this algorithm is the major motivation of this chapter. We note that our analysis and algorithm is not present in [19]. (Actually, the purpose of [19] is to study the NNF and its relationship with the other representations.)

Boolean functions are studied extensively from different perspectives – coding theory [73], circuit complexity [72] and cryptography [20] are some examples. In all these areas, the Walsh transform is the main tool in the analysis of Boolean functions. However, to the best of our knowledge, the only previously known algorithm for computing the Walsh transform is the fast Walsh transform [1]. Hopefully the present work will motivate researchers to study the algorithmic issues of the Walsh transform more deeply.

To illustrate different definitions, terms and and notations we take two small examples.
Example 1 : $g_1(x_1, x_2, x_3) = x_1x_2 \oplus x_2x_3 \oplus x_3x_1$, **and**
$\mathsf{Sup}(A_{g_1}) = \{(1,1,0), (0,1,1), (1,0,1)\}$.
Example 2 : $g_2(x_1, x_2, x_3) = x_1x_2 \oplus x_2x_3 \oplus x_1 \oplus x_2$, **and**
$\mathsf{Sup}(A_{g_2}) = \{(1,1,0), (0,1,1), (1,0,0), (0,1,0)\}$.

3.2 Walsh Transform

In this section, we express the Walsh transform of a Boolean function in terms of certain parameters derived from its ANF. The approach that we take is the following. Equation (2.3) expresses the relation between $(-1)^{g(x)}$ and $W_g(u)$. Our first task is to obtain a formula for $(-1)^{g(x)}$ in terms of the ANF of $g(x)$. Then using Equation (2.3) we obtain the desired relationship between the Walsh transform and the ANF.

In obtaining a formula for $(-1)^{g(x)}$ in terms of the ANF, we first tackle the special case when $g(x) = g(x_1, \ldots, x_m) = x_1 \ldots x_m$. The result of this case is used to analyse the general case.

3.2.1 Case $g(x_1, x_2, \cdots, x_m) = x_1 x_2 \cdots x_m$

For $x = (x_1, \cdots, x_m) \in \{0,1\}^m$, define

$$N_r^{(m)}(x) \triangleq N_r^{(m)}(x_1, \cdots, x_m) \triangleq \sum_{1 \le i_1 < i_2 < \cdots < i_r \le m} (-1)^{x_{i_1} \oplus \cdots \oplus x_{i_r}}.$$

Lemma 1 *Let $x = (x_1, \ldots, x_m) \in \mathbb{F}_2^m$ be such that $\mathsf{wt}(x) = k$. Then*

$$N_r^{(m)}(x) = \sum_{j=0}^{r} (-1)^j \binom{k}{j} \binom{m-k}{r-j}.$$

Consequently, $N_r^{(m)}(x)$ is the Krawtchouk polynomial $p_r(k, m)$ [73, page 130].

Proof : In the vector $(x_1, \cdots, x_m)$, k of the x_i's are 1 and $(m-k)$ of the x_i's are 0. Hence the number of terms of the type $(x_{i_1} \oplus \cdots \oplus x_{i_r})$ with j number of 1's $(0 \le j \le r)$ and $(r-j)$ number of 0's is equal to $\binom{k}{j}\binom{m-k}{r-j}$. Since j of the x_i's are ones, we have, $(-1)^{x_{i_1} \oplus \cdots \oplus x_{i_r}} = (-1)^j$. Hence we get $N_r^{(m)}(x_1, \cdots, x_m) = \sum_{j=0}^{r} (-1)^j \binom{k}{j}\binom{m-k}{r-j}$ which is the Krawtchouk polynomial $p_r(k, m)$ [73, page 130]. ∎

Corollary 2 *1. $\sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(x_1, \cdots, x_m) = 0$ for any vector $x = (x_1, \cdots, x_m) \neq (1, \cdots, 1)$.*

2. $\sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(1, \cdots, 1) = -2^m$.

Proof : From [73, Equation (16), page 130] we have $\sum_{r=0}^{m} (-1)^{r-1} p_r(k, m) = 0$ for $0 \le k < m$ and $\sum_{r=0}^{m} (-1)^{r-1} p_r(k, m) = -2^m$ for $k = m$. Hence using Lemma 1 the result follows. ∎

Theorem 3 *Let $g(x) = x_1 x_2 \cdots x_m$. Then*

$$(-1)^{g(x)} = \frac{1}{2^{m-1}} \left(2^{m-1} + \sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(x) \right).$$

Proof : *Case 1:* $x \in \{0,1\}^m$ such that $x = (1, \cdots, 1)$. Then L.H.S $= (-1)^1 = -1$. By Corollary 2(2) we have $\sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(x) = \sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(1, \cdots, 1) = -2^m$. Hence R.H.S $= \frac{1}{2^{m-1}}(2^{m-1} - 2^m) = -1$.

Case 2: $x \in \{0,1\}^m$ such that $x \neq (1, \cdots, 1)$. Then L.H.S $= (-1)^0 = 1$. By Corollary 2(1) we have $\sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(x) = 0$. Hence R.H.S $= \frac{1}{2^{m-1}}(2^{m-1} - 0) = 1$. ∎

3.2.2 Arbitrary g

Let g be a Boolean function and A_g be its ANF. For $\alpha, x \in \mathbb{F}_2^m$ and $r \in [m] = \{1, \ldots, m\}$. define

$$N_r^{(m)}(\alpha, x) \triangleq \sum_{1 \le i_1 < i_2 < \cdots < i_r \le m} (-1)^{x_{i_1}{}^{\alpha_{i_1}} \oplus \cdots \oplus x_{i_r}{}^{\alpha_{i_r}}}.$$

For $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m) \in \mathbb{F}_2^m$ and $x = (x_1, \cdots, x_m)$, define $x^\alpha \triangleq x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_m^{\alpha_m}$. Note that if $\alpha_i = 1$, then $x_i^{\alpha_i} = x_i$ else $x_i^{\alpha_i} = 1$.

Proposition 4

$$(-1)^{x^\alpha} = \frac{1}{2^{m-1}} \left(2^{m-1} + \sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(\alpha, x) \right)$$

Proof : Let $\mathbf{1} = (1, \ldots, 1)$ and note that $N_r^{(m)}(\mathbf{1}, x) = N_r^{(m)}(x)$. We define $y = (y_1, \cdots, y_m)$ as follows. For $1 \le i \le m$, if $\alpha_i = 1$ then $y_i = x_i$ else $y_i = 1$. Now it is easy to check the following

1. $x_i^{\alpha_i} = y_i$ and hence $(-1)^{x^\alpha} = (-1)^y$.
2. $N_r^{(m)}(\alpha, x) = N_r^{(m)}(y)$.

From this the result follows. ∎

The next result expresses $(-1)^{g(x)}$ in terms of A_g.

Proposition 5

$$(-1)^{g(x)} = \left(\frac{1}{2^{m-1}} \right)^{\mathsf{wt}(A_g)} \prod_{\alpha : A_g(\alpha) = 1} \left(2^{m-1} + \sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(\alpha, x) \right)$$

Proof :

$$(-1)^{g(x)} = (-1)^{\bigoplus_{\alpha \in \mathbb{F}_2^m} A_g(\alpha) x^\alpha} = \prod_{\alpha \in \mathbb{F}_2^m} (-1)^{A_g(\alpha) x^\alpha} = \prod_{\alpha \in \mathbb{F}_2^m} \left((-1)^{x^\alpha} \right)^{A_g(\alpha)} = \prod_{\alpha : A_g(\alpha) = 1} (-1)^{x^\alpha}$$

$$= \prod_{\alpha : A_g(\alpha) = 1} \frac{1}{2^{m-1}} \left(2^{m-1} + \sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(\alpha, x) \right)$$

$$= \left(\frac{1}{2^{m-1}} \right)^{\mathsf{wt}(A_g)} \prod_{\alpha : A_g(\alpha) = 1} \left(2^{m-1} + \sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(\alpha, x) \right)$$

■

For $\alpha = (\alpha_1, \cdots, \alpha_m)$ and $u = (u_1, \cdots, u_m)$ we define $u \leq \alpha$ if $u_i \leq \alpha_i$ for $1 \leq i \leq m$. *In the rest of the chapter we will denote* $(\mathsf{wt}(u) \bmod 2)$ *by* $\mathsf{wt}_2(u)$. In the next two results we present a two step simplification of the sum $\sum_{r=0}^{m}(-1)^{r-1}N_r^{(m)}(\alpha, x)$ which occurs in Proposition 5.

Proposition 6

$$N_r^{(m)}(\alpha, x) = (-1)^r \sum_{\{u \in \mathbb{F}_2^m : u \leq \alpha, \mathsf{wt}(u) \leq r\}} \binom{m-\mathsf{wt}(\alpha)}{r-\mathsf{wt}(u)} (-1)^{\langle u, x\rangle \oplus \mathsf{wt}_2(u)}$$

Proof : Note ${x_{i_1}}^{\alpha_{i_1}} = x_{i_1}$ if $\alpha_{i_1} = 1$ and ${x_{i_1}}^{\alpha_{i_1}} = 1$ if $\alpha_{i_1} = 0$. So ${x_{i_1}}^{\alpha_{i_1}} = 1 \oplus (1 \oplus x_{i_1})\alpha_{i_1} = 1 \oplus (\overline{x}_{i_1})\alpha_{i_1}$ and hence ${x_{i_1}}^{\alpha_{i_1}} \oplus \cdots \oplus {x_{i_r}}^{\alpha_{i_r}} = (r \bmod 2) \oplus \langle (\overline{x}_{i_1}, \cdots, \overline{x}_{i_r}), (\alpha_{i_1}, \cdots, \alpha_{i_r})\rangle$. We have

$$\begin{aligned} N_r^{(m)}(\alpha, x) &= \sum_{1 \leq i_1 < i_2 < \cdots < i_r \leq m} (-1)^{{x_{i_1}}^{\alpha_{i_1}} \oplus \cdots \oplus {x_{i_r}}^{\alpha_{i_r}}} \\ &= \sum_{1 \leq i_1 < i_2 < \cdots < i_r \leq m} (-1)^{(r \bmod 2) \oplus \langle (\overline{x}_{i_1}, \cdots, \overline{x}_{i_r}), (\alpha_{i_1}, \cdots, \alpha_{i_r})\rangle}. \\ &= (-1)^r \sum_{1 \leq i_1 < i_2 < \cdots < i_r \leq m} (-1)^{\langle (\overline{x}_{i_1}, \cdots, \overline{x}_{i_r}), (\alpha_{i_1}, \cdots, \alpha_{i_r})\rangle}. \end{aligned}$$

Given $\alpha = (\alpha_1, \ldots, \alpha_m)$ and $1 \leq i_1 < i_2 < \ldots < i_r \leq m$, define an m-bit vector u in the following manner. For $1 \leq j \leq m$, if $j \in \{i_1, \ldots, i_r\}$, then $u_j = \alpha_j$ else $u_j = 0$. We say that $\{i_1, \ldots, i_r\}$ *produces* u from α. If $\{i_1, \ldots, i_r\}$ produces u from α, then it is easy to verify the following relations.

1. $\langle (\overline{x}_{i_1}, \cdots, \overline{x}_{i_r}), (\alpha_{i_1}, \cdots, \alpha_{i_r})\rangle = \langle (\overline{x}_1, \ldots, \overline{x}_m), (u_1, \ldots, u_m)\rangle$
   $= \langle (x_1, \ldots, x_m), (u_1, \ldots, u_m)\rangle \oplus \mathsf{wt}_2(u)$;
2. $u \leq \alpha$ and $\mathsf{wt}(u) \leq r$.

It is possible that two distinct sets $\{i_1, i_2, \ldots, i_r\}$ and $\{i'_1, i'_2, \ldots, i'_r\}$ produce the same $u \leq \alpha$. This will happen if and only if $\mathsf{wt}(\alpha_{i_1}, \ldots, \alpha_{i_r}) = \mathsf{wt}(\alpha_{i'_1}, \ldots, \alpha_{i'_r}) = \mathsf{wt}(\alpha_{j_1}, \ldots, \alpha_{j_s})$, where $\{j_1, \ldots, j_s\} = \{i_1 \ldots, i_r\} \cap \{i'_1, \ldots, i'_r\}$. We now claim that the number of distinct sets $\{i_1, i_2, \ldots, i_r\}$ which produce the same u is $\binom{m-k}{r-l}$, where $k = \mathsf{wt}(\alpha)$ and $l = \mathsf{wt}(u)$. To see this fix $u \leq \alpha$ with $\mathsf{wt}(u) = l \leq r$. Let $j_1, \ldots, j_l$ be such that $u_{j_1} = \ldots = u_{j_l} = 1$ and $u_j = 0$ for $j \notin \{j_1, \ldots, j_l\}$. If $\{i_1, \ldots, i_r\}$ produces u, we must have $\{j_1, \ldots, j_l\} \subseteq \{i_1, \ldots, i_r\}$ and $\alpha_j = 0$ for $j \in S = \{i_1, \ldots, i_r\} \setminus \{j_1, \ldots, j_l\}$

Thus the number of sets $\{i_1, \ldots, i_r\}$ which produce u is the number of ways we can choose the set S of $r-l$ elements such that $\alpha_j = 0$ for $j \in S$. Since $\mathsf{wt}(\alpha) = k$, the number of $j \in \{1, \ldots, m\}$ such that $\alpha_j = 0$ is $m-k$. Since S has $r-l$ elements, the number of possible sets S is $\binom{m-k}{r-l}$, which proves our claim. Hence we can write

$$N_r^{(m)}(\alpha, x) = (-1)^r \sum_{\{u \in \mathbb{F}_2^m : u \leq \alpha, \mathsf{wt}(u) \leq r\}} \binom{m-k}{r-l} (-1)^{\langle u, x \rangle \oplus \mathsf{wt}_2(u)}$$

This completes the proof. ∎

Proposition 7

$$\sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(\alpha, x) = (-1) \times \sum_{\{u \in \mathbb{F}_2^m : u \leq \alpha\}} 2^{m - \mathsf{wt}(\alpha)} (-1)^{\langle u, x \rangle \oplus \mathsf{wt}_2(u)}$$

Proof : By Proposition 6

$$\begin{aligned}
\sum_{r=0}^{m} (-1)^{r-1} N_r^{(m)}(\alpha, x) &= (-1)^{2r-1} \sum_{r=0}^{m} \sum_{\{u \in \mathbb{F}_2^m : u \leq \alpha, \mathsf{wt}(u) \leq r\}} \binom{m - \mathsf{wt}(\alpha)}{r - \mathsf{wt}(u)} (-1)^{\langle u, x \rangle \oplus \mathsf{wt}_2(u)} \\
&= (-1) \sum_{\{u \in \mathbb{F}_2^m : u \leq \alpha\}} \sum_{r=\mathsf{wt}(u)}^{m} \binom{m - \mathsf{wt}(\alpha)}{r - \mathsf{wt}(u)} (-1)^{\langle u, x \rangle \oplus \mathsf{wt}_2(u)} \\
&= - \sum_{\{u \in \mathbb{F}_2^m : u \leq \alpha\}} 2^{m - \mathsf{wt}(\alpha)} (-1)^{\langle u, x \rangle \oplus \mathsf{wt}_2(u)}.
\end{aligned}$$

This completes the proof. ∎

Let $\mathsf{Sup}(A_g) = \{\alpha^{(1)}, \cdots, \alpha^{(p)}\}$ and $k_i = \mathsf{wt}(\alpha^{(i)})$. Define

$$V(\mathsf{Sup}(A_g)) \triangleq V(\{\alpha^{(1)}, \cdots, \alpha^{(p)}\})$$

$$\triangleq \{u^{(i_1)} \oplus \cdots \oplus u^{(i_r)} : u^{(i_j)} \leq \alpha^{(i_j)}, \{i_1, \ldots, i_r\} \subseteq [p], 1 \leq j \leq r\}.$$

Example 1 (continued) For Boolean function $g_1(x_1, x_2, x_3) = x_1x_2 \oplus x_2x_3 \oplus x_3x_1$, $\mathsf{Sup}(A_{g_1}) = \{(1,1,0), (0,1,1), (1,0,1)\}$ and $k_1 = k_2 = k_3 = 2$.

Note that $V(\mathsf{Sup}(A_g))$ is a subspace of $(\mathbb{F}_2^m, \oplus)$. It is easy to verify that $V(\mathsf{Sup}(A_g)) = \mathbb{F}_2^m$ if and only if g is non-degenerate. Suppose $v = u^{(i_1)} \oplus \cdots \oplus u^{(i_r)}$ where $u^{(i_j)} \leq \alpha^{(i_j)}$ then we say that $R = \{(u^{(i_1)}, \alpha^{(i_1)}), \cdots, (u^{(i_r)}, \alpha^{(i_r)})\}$ is *a representation of* v. For $v \in \mathbb{F}_2^m$, define

$$\mathcal{S}(v) \triangleq \text{set of all representations of } v \text{ in the vector space } V(\mathsf{Sup}(A_g)).$$

Example 1 (continued)

For $v = 000$, we have $\mathcal{S}(v) = \mathcal{S}(000) = \{R_1, R_2, R_3, \cdots, R_{15}, R_{16}, R_{17}\}$ where

$R_1 = \{((0,0,0),(1,1,0))\}$, $R_2 = \{((0,0,0),(0,1,1))\}$, $R_3 = \{((0,0,0),(1,0,1))\}$,

$R_4 = \{((0,0,0),(1,1,0)),((0,0,0),(0,1,1))\}$, $R_5 = \{((0,0,0),(0,1,1)),((0,0,0),(1,0,1))\}$,

$R_6 = \{((0,0,0),(1,1,0)),((0,0,0),(1,0,1))\}$, $R_7 = \{((0,1,0),(1,1,0)),((0,1,0),(0,1,1))\}$,

$R_8 = \{((1,0,0),(1,1,0)),((1,0,0),(1,0,1))\}$, $R_9 = \{((0,0,1),(0,1,1)),((0,0,1,1,0,1))\}$,

$R_{10} = \{((0,0,0),(1,1,0)),((0,0,0),(0,1,1)),((0,0,0),(1,0,1))\}$,

$R_{11} = \{((0,1,0),(1,1,0)),((0,1,0),(0,1,1)),((0,0,0),(1,0,1))\}$,

$R_{12} = \{((1,0,0),(1,1,0)),((0,0,0),(0,1,1)),((1,0,0),(1,0,1))\}$,

$R_{13} = \{((0,0,0),(1,1,0)),((0,0,1),(0,1,1)),((0,0,1),(1,0,1))\}$,

$R_{14} = \{((1,1,0),(1,1,0)),((0,1,1),(0,1,1)),((1,0,1),(1,0,1))\}$,

$R_{15} = \{((1,1,0),(1,1,0)),((0,1,0),(0,1,1)),((1,0,0),(1,0,1))\}$,

$R_{16} = \{((0,1,0),(1,1,0)),((0,1,1),(0,1,1)),((0,0,1),(1,0,1))\}$,

$R_{17} = \{((1,0,0),(1,1,0)),((0,0,1),(0,1,1)),((1,0,1),(1,0,1))\}$.

Note:

1. $\mathcal{S}(v) = \emptyset$ if and only if $v \notin V(\mathsf{Sup}(A_g))$.
2. If $v = (0, \cdots, 0)$, then $\mathcal{S}(v) \neq \emptyset$.

Let $v \in \mathbb{F}_2^m$ and $R = \{(u^{(i_1)}, \alpha^{(i_1)}), \cdots, (u^{(i_r)}, \alpha^{(i_r)})\} \in \mathcal{S}(v)$. Define

$$\left.\begin{array}{lcl} C(R) & \triangleq & \frac{(-1)^r}{2^{(k_{i_1}-1)+\cdots+(k_{i_r}-1)}}. \\ C_v & \triangleq & (-1)^{\mathsf{wt}_2(v)} \sum_{R \in \mathcal{S}(v)} C(R). \end{array}\right\} \tag{3.1}$$

Example 1 (continued)

$$\begin{aligned} C_{000} &= C(R_1) + C(R_2) + C(R_3) + \cdots + C(R_{15}) + C(R_{16}) + C(R_{17}) \\ &= \frac{(-1)^1}{2^{2-1}} + \frac{(-1)^1}{2^{2-1}} + \frac{(-1)^1}{2^{2-1}} + \frac{(-1)^2}{2^{4-2}} + \frac{(-1)^2}{2^{4-2}} + \frac{(-1)^2}{2^{4-2}} + \frac{(-1)^3}{2^{6-3}} + \frac{(-1)^2}{2^{4-2}} + \frac{(-1)^2}{2^{4-2}} + \\ &\quad \frac{(-1)^2}{2^{4-2}} + \frac{(-1)^3}{2^{6-3}} + \frac{(-1)^3}{2^{6-3}} + \frac{(-1)^3}{2^{6-3}} + \frac{(-1)^3}{2^{6-3}} + \frac{(-1)^3}{2^{6-3}} + \frac{(-1)^3}{2^{6-3}} + \frac{(-1)^3}{2^{6-3}} \\ &= -1 \end{aligned}$$

Note that if $\mathcal{S}(v) = \emptyset$, then $\sum_{R \in \mathcal{S}(v)} C(R) = 0$ and hence, if $v \notin V(\mathsf{Sup}(A_g))$ then $C_v = 0$.

The parameters C_v for $v \in \mathbb{F}_2^m$ are derived entirely from the ANF of g. Our next result expresses $(-1)^{g(x)}$ in terms of C_v.

Theorem 8

$$(-1)^{g(x)} = 1 + \sum_{v \in \mathbb{F}_2^m} C_v (-1)^{\langle v,x \rangle}$$

where C_v is as defined by Equation 3.1.

Proof : As before let $\mathsf{Sup}(A_g) = \{\alpha^{(1)}, \ldots, \alpha^{(p)}\}$ where $\mathsf{wt}(\alpha^{(i)}) = k_i$. Using Proposition 5 and Proposition 7 we can write.

$$(-1)^{g(x)} = \prod_{\alpha : A_g(\alpha)=1} \left(1 - \frac{1}{2^{m-1}} \sum_{\{u \in \mathbb{F}_2^m : u \leq \alpha\}} 2^{m - \mathsf{wt}(\alpha)} (-1)^{\langle u,x \rangle \oplus \mathsf{wt}_2(u)} \right)$$

The first term in the expansion of the above expression is clearly 1. For $1 \leq r \leq p = \mathsf{wt}(A_g)$, the general term is of the form

$$\left(\frac{-1}{2^{m-1}}\right)^r \sum_{u^{(i_1)} \leq \alpha^{(i_1)}} \cdots \sum_{u^{(i_r)} \leq \alpha^{(i_r)}} 2^{(m-k_{i_1}) + \cdots + (m-k_{i_r})} (-1)^{\langle u^{(i_1)} \oplus \cdots \oplus u^{(i_r)}, x \rangle \oplus \mathsf{wt}_2(u^{(i_1)} \oplus \cdots \oplus u^{(i_r)})}$$

Let $v = u^{(i_1)} \oplus \cdots \oplus u^{(i_r)}$. Then $R = \{(u^{(i_1)}, \alpha^{(i_1)}), \cdots, (u^{(i_r)}, \alpha^{(i_r)})\}$ is a representation of v. Therefore the general term is of the form

$$\sum_{v \in \mathbb{F}_2^m} \sum_{R \in \mathcal{S}(v)} C(R) (-1)^{\langle v,x \rangle \oplus \mathsf{wt}_2(v)} = \sum_{v \in \mathbb{F}_2^m} C_v (-1)^{\langle v,x \rangle}.$$

This gives us the desired result. ∎

Now we are in a position to state the main result of this section which relates $W_g(v)$ to C_v.

Theorem 9 *If g is an m-variable Boolean function then*

$$W_g(v) = 2^m (C_v + \delta_v) \tag{3.2}$$

where $\delta_v = 1$ if $v = 0$ else $\delta_v = 0$.

Proof : From Theorem 8 we have

$$(-1)^{g(x)} - 1 = \sum_{v \in \mathbb{F}_2^m} C_v (-1)^{\langle v,x \rangle}$$

So $[(-1)^{g(0)} - 1, \ldots, (-1)^{g(2^m-1)} - 1] = [C_0, \ldots, C_{2^m-1}]H_m$, where H_m is the $(2^m \times 2^m)$ Hadamard matrix (see Section 2.1.2). Post multiplying both sides by H_m and noting that $H_m H_m = 2^m I_{2^m}$ we get

$$[(-1)^{g(0)}, \ldots, (-1)^{g(2^m-1)}]H_m - [1, \ldots, 1]H_m = 2^m[C_0, \ldots, C_{2^m-1}].$$

i.e. $[W_g(0), \ldots, W_g(2^m - 1)] - [2^m, 0, \ldots, 0] = 2^m[C_0, \ldots, C_{2^m-1}]$. Hence $2^m C_v = W_g(v) - 2^m\delta_v$ ■

3.3 Simplifying C_v

Theorem 9 relates $W_g(v)$ to C_v. Thus to compute $W_g(v)$ it is sufficient to compute C_v. However, the definition of C_v given by 3.1 is in purely algebraic terms. We need to obtain a formula for C_v which can be computed by an algorithm. In this section we perform this task of simplifying C_v.

We start by defining certain terms. As in Section 3.2, we assume $\mathsf{Sup}(A_g) = \{\alpha^{(1)}, \ldots, \alpha^{(p)}\}$. Given $\emptyset \neq S \subseteq [p] = \{1, 2, \cdots, p\}$ and $v \in \mathbb{F}_2^m$, we define $\Delta(S, v)$ as follows. Let $v = (v_1, \cdots, v_m)$ and $S = \{i_1, \cdots, i_r\}$. Write $\alpha^{(i_j)} = (\alpha_{j,1}, \cdots, \alpha_{j,m})$ where $\alpha_{j,k} \in \{0, 1\}$ for $1 \leq j \leq r$, $1 \leq k \leq m$. Let $\phi(S, v)$ be a Boolean formula which is true if and only if there is a $k \in [m]$, such that $\alpha_{j,k} = 0$ for all $j \in [r]$ and $v_k = 1$. Let

$$\left.\begin{array}{lcll} \sigma(S) & = & \bigvee_{i \in S} \alpha^{(i)} & = \alpha^{(i_1)} \vee \cdots \vee \alpha^{(i_r)}); \\ \mu(S) & = & \mathsf{wt}(\sigma(S)). & \end{array}\right\} \tag{3.3}$$

Here $\vee$ represents the bitwise logical OR of two binary strings of the same length. Define

$$\left.\begin{array}{lcll} \Delta(S, v) & = & 0 & \text{if } \phi(S, v) = 1 \\ & = & \frac{1}{2^{\mu(S)-|S|}} & \text{otherwise.} \end{array}\right\} \tag{3.4}$$

Remark: If $\Delta(S, v) > 0$, then the value of $\Delta(S, v)$ is independent of v and depends only on S.

Theorem 10

$$C_v = (-1)^{\mathsf{wt}_2(v)} \sum_{\emptyset \neq S \subseteq [p]} (-1)^r \Delta(S, v).$$

Proof : Fix a set S, such that, $\emptyset \neq S = \{i_1, \ldots, i_r\} \subseteq [p]$ and a vector $v = (v_1, \ldots, v_m) \in \mathbb{F}_2^m$. Define $\Gamma(S) = \frac{2^r}{2^{k_{i_1}+\cdots+k_{i_r}}}$ where $k_{i_j} = \mathsf{wt}(\alpha^{(i_j)})$. As before, let $\alpha^{(i_j)} = (\alpha_{j,1}, \cdots, \alpha_{j,m})$, $1 \leq j \leq r$ and define $\lambda_k = \sum_{j=1}^{r} \alpha_{j,k}$. For $1 \leq k \leq m$, define

$$\left.\begin{array}{rcll} b_k & = & 2^{\lambda_k - 1} & \text{if } \lambda_k \neq 0 \\ & = & 1 & \text{if } \lambda_k = 0, v_k = 0 \\ & = & 0 & \text{if } \lambda_k = 0, v_k = 1 \end{array}\right\} \tag{3.5}$$

Define $n(S, v) = b_1 \cdots b_m$.

Claim 1: $\Delta(S, v) = n(S, v)\Gamma(S)$

Proof of Claim 1: There are two cases to consider.

Case $\phi(S, v) = 1$: In this case, $\Delta(S, v) = 0$ by Equation 3.4. Also $\phi(S, v) = 1$ implies that there is a $k \in [m]$ such that $\alpha_{j,k} = 0$ for all $j \in [r]$ and $v_k = 1$. This implies that $\lambda_k = 0$ and $v_k = 1$. Hence $b_k = 0$ and so $n(S, v) = 0$. Thus in this case we have $\Delta(S, v) = n(S, v)\Gamma(S)$.

Case $\phi(S, v) = 0$: In this case $\Delta(S, v) = \frac{1}{2^{m_1 - r}}$ by Equation 3.4, where $m_1 = \mu(S)$ and $r = |S|$. Also $b_k = 2^{\lambda_k - z_k}$ where $z_k = 0$ if $\lambda_k = 0$; and $z_k = 1$ if $\lambda_k > 0$. So

$$n(S, v) = (2^{\lambda_1 - z_1})(2^{\lambda_2 - z_2}) \cdots (2^{\lambda_m - z_m}) = 2^{(\lambda_1 + \cdots + \lambda_m) - (z_1 + \cdots + z_m)}.$$

Since $m_1 = \mu(S) = \mathsf{wt}(\alpha^{(i_1)} \vee \cdots \vee \alpha^{(i_r)})$ we have $(z_1 + z_2 + \cdots + z_m) = m_1$. Also we have

$$\sum_{k=1}^{m} \lambda_k = \sum_{k=1}^{m} \sum_{j=1}^{r} \alpha_{j,k} = \sum_{j=1}^{r} \sum_{k=1}^{m} \alpha_{j,k} = \sum_{j=1}^{r} k_{i_j}.$$

So we get

$$n(S, v) = 2^{(\lambda_1 + \cdots + \lambda_m) - m_1} = \frac{2^{(k_{i_1} + \ldots + k_{i_r})}}{2^{m_1}}.$$

By definition $\Gamma(S) = \frac{2^r}{2^{k_{i_1}+\cdots+k_{i_r}}}$ and so $\Delta(S, v) = \frac{1}{2^{m_1 - r}} = n(S, v)\Gamma(S)$. This completes the proof of Claim 1.

Claim 2:

$$C_v = (-1)^{\mathsf{wt}_2(v)} \sum_{\emptyset \neq S \subseteq [p]} (-1)^r n(S, v)\Gamma(S).$$

Proof of Claim 2: From Equation 3.1, we have $C_v = (-1)^{\mathsf{wt}_2(v)} \sum_{R \in \mathcal{S}(v)} C(R)$.

Given $S = \{i_1, \cdots, i_r\}$, the vectors $\alpha^{(i_1)}, \ldots, \alpha^{(i_r)}$ are fixed. Let $D(S, v)$ be the set of all representations of v of the form $\{(u^{(i_1)}, \alpha^{(i_1)}), \ldots, (u^{(i_r)}, \alpha^{(i_r)})\}$. The value of $C(R)$ for any such representation R is equal to $\frac{2^r}{2^{k_{i_1}+\cdots+k_{i_r}}}$ and depends only on S. From definition, this

value is equal to $\Gamma(S)$. In evaluating C_v we have to sum over all representations of v. The contribution of the set S to this sum is clearly $|D(S,v)|\Gamma(S)$. Thus we obtain

$$C_v = (-1)^{\mathsf{wt}_2(v)} \sum_{\emptyset \neq S \subseteq [p]} |D(S,v)|\Gamma(S).$$

From this it is sufficient to show that $n(S,v) = |D(S,v)|$. There are two cases to consider.

Case $b_k = 0$ for some $k \in [m]$:
This implies that v cannot be represented as $\{(u^{(i_1)}, \alpha^{(i_1)}), \ldots, (u^{(i_r)}, \alpha^{(i_r)}\}$ for any choice of $u^{(i_j)} \leq \alpha^{(i_j)}$. Hence $D(S,v) = \emptyset$ and so we have $n(S,v) = 0 = |D(S,v)|$.

Case $b_k > 0$ for all $k \in [m]$:
In this case $D(S,v) \neq \emptyset$. Suppose $R = \{(u^{(i_1)}, \alpha^{(i_1)}), \ldots, (u^{(i_r)}, \alpha^{(i_r)}\} \in D(S,v)$, where for $1 \leq j \leq r$, $u^{(i_j)} = (u_{j,1}, \ldots, u_{j,m}) \in \mathbb{F}_2^m$ and $\alpha^{(i_j)} = (\alpha_{j,1}, \ldots, \alpha_{j,m}) \in \mathbb{F}_2^m$.

Define two $r \times m$ matrices M_1 and M_2 in the following manner. The (j,k)th entry of M_1 (resp. M_2) is $\alpha_{j,k}$ (resp. $u_{j,k}$). The conditions $u^{(i_j)} \leq \alpha^{(i_j)}$ is equivalent to $M_2 \leq M_1$, where the matrices are compared entrywise. Since R is a representation of v, we must have,

$$u_{1,k} \oplus \ldots \oplus u_{r,k} = v_k \text{ for each } k \in [m]. \tag{3.6}$$

Thus the problem of enumerating the set $D(S,v)$ is equivalent to enumerating matrices M_2 such that $M_2 \leq M_1$ and Equation (3.6) holds. Let c_k be the number of possible choices for the kth column of M_2. Then $|D(S,v)| = c_1 \ldots c_m$ and it is sufficient to show that $c_k = b_k$ for each $k \in [m]$. There are two cases to consider.

Subcase $\lambda_k = 0$: In this case $v_k = 0$ and $b_k = 1$. Since $\lambda_k = 0$, the kth column of M_1 is the all zero column. Hence the only possible choice for the kth column of M_2 is also the all zero column and so $c_k = 1 = b_k$.

Subcase $\lambda_k > 0$: In this case $b_k = 2^{\lambda_k - 1}$. We first observe that the XOR of the ones in the kth column of M_2 must be equal to v_k. Thus we have to consider two cases according as $v_k = 0$ or $v_k = 1$. First suppose $v_k = 0$. The kth column of M_1 contains λ_k ones. Since $v_k = 0$, we can choose an even number of these ones to form a column for the matrix M_2. Thus the number of choices for the kth column of M_2 is $c_k = \binom{\lambda_i}{0} + \binom{\lambda_i}{2} + \cdots + \binom{\lambda_i}{2\lfloor \frac{\lambda_i}{2} \rfloor} = 2^{\lambda_k - 1} = b_k$. A similar argument holds when $v_k = 1$.

This completes the proof of Claim 2. Theorem 10 is a direct consequence of Claim 1 and Claim 2. ∎

Theorem 10 provides a method for computing C_v. However, this requires an algorithm to consider all possible subsets of $[p] = \{1, \ldots, p\}$. If p is even moderately large, this yields an

impractical algorithm. Our next task is to show that it is actually not required to consider all the subsets of $[p]$. In this section we present one such special case and in the next section we present a general algorithm to compute C_v without generating all the subsets of $[p]$.

Theorem 11 *Let* $\mathsf{Sup}(A_g) = \{\alpha_1, \cdots, \alpha_p\}$ *and* $t > 0$ *be such that for any* $\{i_1, \ldots, i_{t+1}\} \subseteq \{1, \ldots, p\}$, *we have* $\mathsf{wt}(\alpha_{i_1} \vee \cdots \vee \alpha_{i_{t+1}}) = m$. *Then*

$$C_v = (-1)^{\mathsf{wt}_2(v)} \left(\sum_{j=t+1}^{p} \frac{(-1)^j}{2^{m-j}} \binom{p}{j} + \sum_{\emptyset \neq S \subseteq [p], |S| \leq t} (-1)^{|S|} \triangle (S, v) \right)$$

Proof : We have from Theorem 10

$$\begin{aligned} C_v &= (-1)^{\mathsf{wt}_2(v)} \sum_{\emptyset \neq S \subseteq [p]} (-1)^{|S|} \triangle (S, v) \\ &= (-1)^{\mathsf{wt}_2(v)} \left(\sum_{\emptyset \neq S \subseteq [p], |S| \geq t+1} (-1)^{|S|} \triangle (S, v) + \sum_{\emptyset \neq S \subseteq [p], |S| \leq t} (-1)^{|S|} \triangle (S, v) \right). \end{aligned}$$

Under the given condition if $|S| \geq t + 1$ then using Equation 3.4 we have $\mu(S) = m$. Hence

$$\begin{aligned} C_v &= (-1)^{\mathsf{wt}_2(v)} \left(\sum_{\emptyset \neq S \subseteq [p], |S| \geq t+1} (-1)^{|S|} \frac{1}{2^{m-r}} + \sum_{\emptyset \neq S \subseteq [p], |S| \leq t} (-1)^{|S|} \triangle (S, v) \right) \\ &= (-1)^{\mathsf{wt}_2(v)} \left(\sum_{j=t+1}^{p} (-1)^j \frac{1}{2^{m-r}} \binom{p}{j} + \sum_{\emptyset \neq S \subseteq [p], |S| \leq t} (-1)^{|S|} \triangle (S, v) \right) \end{aligned}$$

This completes the proof. ∎

Under the condition of Theorem 11, to evaluate C_v we only have to consider all non empty subsets of $[p]$ of cardinality at most t. If t is reasonably small, this is much better than considering all non empty subsets of $[p]$. In the next section, we develop this idea to obtain an algorithm to compute C_v.

3.4 Algorithm

In general we are interested in computing the Walsh transform of g at all the points $u \in \mathbb{F}_2^m$. However, if m is relatively large (say around 50), then it will be practically impossible to compute the Walsh transform at all the 2^m points. In such a situation, it will be of interest

to compute the Walsh transform at a particular point or for a small set of points. The fast Walsh transform takes time $O(m2^m)$ and computes the Walsh transform at all the 2^m points. In fact, to the best of our knowledge, there is no known algorithm which can compute the Walsh transform at a particular point in time less than 2^m.

Our approach is to design an algorithm that consists of two parts. In the first part, the algorithm does a certain amount of preprocessing and prepares a list. In the second part, the algorithm takes as input a particular $v \in \mathbb{F}_2^m$ and computes C_v. (Using Theorem 9 this also gives us $W_g(v)$). Once the preprocessing is complete, the second part can be run for different v without running the first part. This makes it efficient to compute $W_g(v)$ for a set of v.

We start by defining certain parameters. For $\emptyset \neq S \subseteq [p]$ recall from equation (3.3) that $\sigma(S) = \bigvee_{i \in S} \alpha^{(i)}$ and $\mu(S) = \mathsf{wt}(\sigma(S))$. For $j = 0, \cdots, m$ define

$$B_j \triangleq \sum_{\mu(S)=j} (-2)^{|S|} \text{ and } B \triangleq \sum_{j=0}^{m-1} \sum_{\mu(S)=j} (-1)^{|S|} \Delta(S, v).$$

Note that if $S = \{i\}$ and $\alpha^{(i)} = (0, \ldots, 0)$, then $\mu(S) = 0$. The vector $(0, \ldots, 0) \in \mathsf{Sup}(A_g)$ implies that the constant term in the ANF of g is equal to 1. The values of $B_0, \ldots, B_m$ are independent of v and only the value of B depends on v.

Theorem 12 *For any $v \in \mathbb{F}_2^m$, we have*

$$C_v = (-1)^{\mathsf{wt}_2(v)} \left[\frac{(-1)^p - 1 - B_0 - \cdots - B_{m-1}}{2^m} + B \right]. \tag{3.7}$$

Proof : Define

$$A \triangleq \sum_{\emptyset \neq S \subseteq [p]} (-2)^{|S|} = \sum_{j=1}^{p} (-2)^j \binom{p}{j} = -1 + (-1)^p.$$

Then $A = \sum_{j=0}^{m} B_j$. Consequently, $B_m = A - B_0 - \cdots - B_{m-1}$. To see this note that $\sum_{j=0}^{m} B_j = \sum_{j=0}^{m} \sum_{\mu(S)=j} (-2)^{|S|} = \sum_{\emptyset \neq S \subseteq [p]} (-2)^{|S|} = A$.

From Theorem 10 we have

$$\begin{aligned} C_v &= (-1)^{\mathsf{wt}_2(v)} \sum_{\emptyset \neq S \subseteq [p]} (-1)^{|S|} \Delta(S, v) \\ &= (-1)^{\mathsf{wt}_2(v)} \left[\sum_{j=0}^{m} \sum_{\mu(S)=j} (-1)^{|S|} \Delta(S, v) \right] \end{aligned}$$

$$= (-1)^{\mathsf{wt}_2(v)} \left[\sum_{\mu(S)=m} (-1)^{|S|} \Delta(S,v) + \sum_{j=0}^{m-1} \sum_{\mu(S)=j} (-1)^{|S|} \Delta(S,v) \right]$$
$$= (-1)^{\mathsf{wt}_2(v)} \left[\frac{B_m}{2^m} + B \right]$$
$$= (-1)^{\mathsf{wt}_2(v)} \left[\frac{A - B_0 - \cdots - B_{m-1}}{2^m} + B \right]$$

This completes the proof. ∎

Our algorithm is based on Equation (3.7). The intuition behind the algorithm is the following. For most sets S, the value of $\mu(S)$ will be equal to m and will be accounted for by B_m. Thus if we can avoid computing B_m directly, then we will be saving a lot of computation. We will compute $B_0, \ldots, B_{m-1}$ and then use Equation (3.7) to compute the value of C_v. However, the value of B has to be computed. We next describe how this is done.

Define

$$\left.\begin{array}{rcl} \mathcal{S}_1 & = & \{S \subseteq [p] : \mu(S) < m\}; \\ \mathcal{S}_2 & = & \sigma(\mathcal{S}_1) = \{\sigma(S) : S \in \mathcal{S}_1\}. \end{array}\right\} \tag{3.8}$$

Example 2 (continued) From now onward, we consider the Boolean function $g_2(x_1, x_2, x_3) = x_1x_2 \oplus x_2x_3 \oplus x_1 \oplus x_2$, so $\mathsf{Sup}(A_{g_2}) = \{(1,1,0), (0,1,1), (1,0,0), (0,1,0)\}$. This Boolean function is taken different from the previous one so that $|\mathcal{S}_1|$ and $|\mathcal{S}_2|$ are different. Here $p = 4$, $m = 3$. It is easy to see

$$\mathcal{S}_1 = \{\{1\}, \{2\}, \{3\}, \{4\}, \{1,3\}, \{1,4\}, \{2,4\}, \{3,4\}, \{1,3,4\}\},$$

$$\mathcal{S}_2 = \{\sigma(\{1\}) = (1,1,0), \sigma(\{2\}) = (0,1,1), \sigma(\{3\}) = (1,0,0), \sigma(\{4\}) = (0,1,0)\}$$

and hence $|\mathcal{S}_1| = 9$ and $|\mathcal{S}_2| = 4$. Note by definition of σ, $\sigma(\{1,3\}) = (1,1,0) \vee (1,0,0) = (1,1,0) = \sigma(\{1\})$. And this is the reason for $|\mathcal{S}_2| \leq |\mathcal{S}_1|$.

Write $\mathcal{S}_2 = \{str_1, \ldots, str_n\}$ for some $n > 0$. For $1 \leq i \leq n$ define $val_i = \sum_{\sigma(S)=str_i} (-2)^{|S|}$. In the first part of our algorithm we prepare the list $\mathcal{L} = ((str_1, val_1), \ldots, (str_n, val_n))$. Note that this part does not depend on v.

Example 2 (continued) $\mathcal{S}_2 = \{str_1 = (1,1,0), str_2 = (0,1,1), str_3 = (1,0,0), str_4 = (0,1,0)\}$. Now $val_1 = \sum_{\sigma(S)=str_1} (-2)^{|S|} = (-2)^1 + (-2)^2 + (-2)^2 + (-2)^2 + (-2)^3 = -2$. Similarly $val_2 = 2$, $val_3 = -2$ and $val_4 = -2$

For $str, v \in \mathbb{F}_2^m$, define $\psi(str, v) \triangleq \overline{str} \wedge v$ where $\overline{str}$ is bitwise complement of str and $\wedge$ is the bitwise logical AND. The operation $\wedge$ is performed bitwise on str and v.

Suppose $\emptyset \neq S \subseteq [p]$ such that $\sigma(S) = str$ and $v \in \mathbb{F}_2^m$. Then $\Delta(S, v) > 0$ if and only if $\psi(str, v) = (0, \ldots, 0)$. Thus

$$B = \sum_{\psi(str_i, v) = (0, \ldots, 0)} \frac{val_i}{2^{\mathsf{wt}(str_i)}}.$$

Also for $j = 0, \ldots, m-1$,

$$B_j = \sum_{\mathsf{wt}(str_i) = j} val_i.$$

Hence once the list $\mathcal{L}$ is prepared, it is easy to compute $B_0, \ldots, B_{m-1}$ and B.

Example 2 (continued) We take $v = (1, 0, 0)$, then $\psi((1, 1, 0), (1, 0, 0)) = (0, 0, 1) \wedge (1, 0, 0) = (0, 0, 0)$. Similarly $\psi((0, 1, 1), (1, 0, 0)) = (1, 0, 0)$, $\psi((1, 0, 0), (1, 0, 0)) = (0, 0, 0)$ and $\psi((0, 1, 0), (1, 0, 0)) = (1, 0, 0)$. So $B = \frac{val_1}{2^{\mathsf{wt}(str_1)}} + \frac{val_3}{2^{\mathsf{wt}(str_3)}} = \frac{2}{4} + \frac{-2}{2} = -\frac{1}{2}$ Again from definition $B_0 = 0, B_1 = val_3 + val_4 = -2 + (-2) = -4, B_2 = val_1 + val_2 = 2 + 2 = 4$. Now $4 = -1 + (-1)^p = 0$. So $C_v = (-1)^{\mathsf{wt}_2(v)} \left[\frac{A - B_0 - B_1 - B_2}{2^m} + B\right] = -\left[\frac{0 - 0 + 4 - 4}{2^3} + \frac{1}{2}\right] = -\frac{1}{2}$.

Now we describe a method for preparing the list $\mathcal{L}$. First we describe a rooted directed tree $\mathcal{T}$ whose nodes are the subsets of $[p] = \{1, \ldots, p\}$. The root node of $\mathcal{T}$ is the empty set. The children of a set S are the sets $S_1, \ldots, S_k$, where for $1 \leq i \leq k$, $S_i = S \cup \{\max(S) + i\}$ and $k = p - \max(S)$. This ensures that if S' is a node in the subtree rooted at S, then $S \subset S'$.

Our algorithm will traverse all the nodes S of $\mathcal{T}$ for which $\mu(S) < m$. (Note that $\mathcal{T}$ has 2^p nodes and if an algorithm is required to traverse all the nodes of $\mathcal{T}$, then the algorithm will be exponential in p.) From the structure of $\mathcal{T}$ we know that if $\mu(S) = m$ for some node S, then $\mu(S') = m$ for all nodes S' in the subtree rooted at S. This crucial fact makes the traversal particularly efficient. If during the traversal we reach a node S with $\mu(S) = m$, then we need not visit any of the nodes in the subtree rooted at S. This means that we are effectively *pruning* the subtree rooted at S from $\mathcal{T}$. The more we encounter this pruning effect, the more efficient is our algorithm.

While traversing $\mathcal{T}$ we prepare the list $\mathcal{L}$ in the following manner. Initially $\mathcal{L}$ is the empty list. Let $\mathsf{First}(\mathcal{L}) = \{str : (str, val) \in \mathcal{L}\}$. Suppose we have reached a node S with $\mu(S) < m$. If $\sigma(S) \notin \mathsf{First}(\mathcal{L})$, then we add $(\sigma(S), (-2)^{|S|})$ to $\mathcal{L}$. On the other hand, if $\sigma(S) \in \mathsf{First}(\mathcal{L})$, then we update val to $val + (-2)^{|S|}$.

Thus the operations on $\mathcal{L}$ are search and insert. We implement $\mathcal{L}$ using a height balanced binary tree (see [57]). Hence each search/insert operation requires time $O(\log \mathcal{L})$. One such search or insert operation is required for each S such that $\mu(S) < m$. Also the total time spent at any node which is visited is $O(m)$. Hence the total time required by the algorithm

is $O(m|\mathcal{S}_1|\log(|\mathcal{S}_2|))$. We next present the algorithm for computing C_v.

Algorithm ComputeCv

Inputs :

1. $sup(A_g)$, where g is an m-variable Boolean function.
2. $v \in \mathbb{F}_2^m$.

Output : C_v.

Part 1 : Computation of list $\mathcal{L}$.
Set $\mathcal{L}$ equal to the empty list.
Set str equal to the empty string.
Traverse(str).
Assume $\mathcal{L} = ((str_1, val_1), \ldots, (str_n, val_n))$ at the end of Traverse.
Part 2 : Computation of C_v.
Set $B_0 = \cdots = B_{m-1} = 0$.
For $i = 1$ to n do
if $(\mathsf{wt}(str_i) = j))$ then $B_j = B_j + val_i$.
if $(\psi(str_i, v) = (0, \ldots, 0))$ then $B = B + \frac{val_i}{2^{\mathsf{wt}(str_i)}}$.
End For.
$C_v = (-1)^{\mathsf{wt}_2(v)}(\frac{1}{2^m}((-1)^p - 1 - \sum_{i=0}^{m-1} B_i) + B)$.
Return C_v.
End Algorithm ComputeCv.

The subroutine Traverse() performs a depth first search on the tree $\mathcal{T}$ described before. The details of the algorithm Traverse() are given below. In the algorithm we use the notation $\sigma(tstr)$ for a p-bit string $tstr$ to mean $\sigma(S)$ where $S = \{i : tstr_i = 1\}$.

Algorithm Traverse(str)

Input : a binary string str of length at most p.

For $i = |str|$ to $p - 1$ do
Set $j = i - |str|$.
Set $tstr = str||0^j||1$.
If $(\mu(tstr) < m)$ then
If $(\sigma(tstr) \notin \mathsf{First}(\mathcal{L})$, then
Add $(\sigma(tstr), (-2)^{\mathsf{wt}(tstr)}$ to $\mathcal{L}$.
Else suppose $(\sigma(tstr), val)$ is present in $\mathcal{L}$.

Set $val = val + (-2)^{\mathsf{wt}(tstr)}$.
End If.
If ($|tstr| < p$), then Traverse($tstr$).
End If.
End For.
End Algorithm Traverse

From the above discussion we obtain the following result.

Theorem 13 *Let $v_1, \ldots, v_t \in \mathbb{F}_2^m$ and g be an m-variable Boolean function. Then $\{W_g(v_i) : 1 \leq i \leq t\}$ can be computed in time $O(m(|\mathcal{S}_1| \log(|\mathcal{S}_2|) + t|\mathcal{S}_2|))$.*

Proof : Part 1 of Algorithm ComputeCv has to be executed only once for all the vectors $v_1, \ldots, v_t$. This takes time $O(|\mathcal{S}_1| \log(|\mathcal{S}_2|))$. Part 2 of Algorithm ComputeCv has to run once for each of the vectors $v_1, \ldots, v_t$. Each execution of Part 2 takes time $O(|\mathcal{S}_2|)$. This gives the time complexity of the algorithm. Correctness of the algorithm follows from the previous discussion. ∎

Remark : It is important to note that for small t it is possible to have $|\mathcal{S}_1| \log(|\mathcal{S}_2|) + t|\mathcal{S}_2| \ll \min(2^p, 2^m)$. In such situations it is possible to efficiently compute the Walsh transform of g for a small set of points. Cardinality of $|\mathcal{S}_1|$ and $|\mathcal{S}_2|$ depends on ANF and in the worst case can be $O(2^p)$. To keep $|\mathcal{S}_1|$ and $|\mathcal{S}_2|$ within a controllable limit, the ANF should be a sparse multinomial and support of ANF should contain terms with very high weights (nearly m). In Section 3.5, we provide some experimental data to support this intuition.

3.5 Experimental Results

Similar to Section 3.2, let $\mathsf{Sup}(A_g) = \{\alpha^{(1)}, \ldots, \alpha^{(p)}\}$. We write $\alpha^{(i)} = (\alpha_{i,1}, \cdots, \alpha_{i,m})$ where $\alpha_{i,j} \in \{0, 1\}$ for $1 \leq i \leq p$, $1 \leq j \leq m$. For $1 \leq j \leq m$, define $\rho_j = p - \sum_{i=1}^{p} \alpha_{i,j}$. Let $\rho_{max} = max_{j \in [m]}(\rho_j)$, $\rho_{avg} = \sum_{i=1}^{m} \frac{\rho_i}{m}$ and $w_{avg} = \sum_{i=1}^{p} \frac{\mathsf{wt}(\alpha^{(i)})}{p}$. We use the notation $\mathcal{S}_1$ and $\mathcal{S}_2$ as defined in Equation 3.8.

Tables 1 and 2 consider the cases $p > m$ and $p < m$ respectively. (Here we note that the class of Boolean functions for which p is less than m is also very rich. For example this class includes an important subset of the class of bent functions.) The results in these two tables show that ρ_i's (specially ρ_{max}) are crucial to the complexity of the algorithm. If ρ_{max} is comparatively larger than the other ρ_i's then the size of $|\mathcal{S}_1|$ is exponential in ρ_{max}. Also note that in general $|\mathcal{S}_1| \geq 2^{\rho_{max}}$.

m, p	$\lvert\mathcal{S}_2\rvert$	$\lvert\mathcal{S}_1\rvert, 2^{\rho_{max}}$	$\rho_{max}, \rho_{avg}, w_{avg}$
100, 250	546	2135470, 1048576	20, 15, 94
100, 250	547	3184044, 2097152	21, 15, 94
100, 250	549	5288601, 4194304	22, 15, 94
100, 250	549	9482897, 8388608	23, 15, 94
100, 250	550	17871500, 16777216	24, 15, 94
100, 250	552	34648698, 33554432	25, 15, 94
100, 250	553	68203122, 67108864	26, 15, 94

Table 1

m, p	$\lvert\mathcal{S}_2\rvert$	$\lvert\mathcal{S}_1\rvert, 2^{\rho_{max}}$	$\rho_{max}, \rho_{avg}, w_{avg}$
100, 50	2258	365465, 131072	17, 15, 70
100, 50	2392	1340716, 1048576	20, 15, 70
100, 50	2429	8678491, 8388608	23, 15, 70
100, 50	2465	67386792, 67108864	26, 15, 70

Table 2

From the definition, the parameters m, p, ρ_{avg} and w_{avg} satisfy the following relationship: $m \times \rho_{avg} = p \times (m - w_{avg})$. For fixed m and p, ρ_{avg} decreases as w_{avg} increases. Similarly, for fixed m and ρ_{avg}, as p increases so does w_{avg}. In Table 3, we show the change of w_{avg} with p for $m = 100$ and $\rho_{avg} = 16$.

m, p	$\lvert\mathcal{S}_2\rvert$	$\lvert\mathcal{S}_1\rvert, 2^{\rho_{max}}$	$\rho_{max}, \rho_{avg}, w_{avg}$
100, 50	9976	5313186, 524288	19, 16, 68
100, 100	2934	6482400, 524288	19, 16, 84
100, 200	1414	4609688, 524288	19, 16, 92
100, 400	827	6644436, 524288	19, 16, 96

Table 3

It is possible to run the algorithm in cases where m is large but ρ_{max} is small. Table 4 provides some data which illustrates this fact.

m, p	$\lvert\mathcal{S}_2\rvert$	$\lvert\mathcal{S}_1\rvert, 2^{\rho_{max}}$	$\rho_{max}, \rho_{avg}, w_{avg}$
200, 200	725	96213, 32768	15, 12, 188
400, 200	2175	207969, 16384	14, 12, 376
800, 200	1938	528605, 65536	16, 12, 752
3200, 100	2290	243188, 16384	14, 12, 2816
3200, 100	2502	4436706, 4194304	22, 12, 2816

Table 4

In the above examples we see that $|\mathcal{S}_2|$ is small. In Table 5 we provide some examples for which $|\mathcal{S}_2|$ is large but w_{avg} is small.

m, p	$\lvert\mathcal{S}_2\rvert$	$\lvert\mathcal{S}_1\rvert, 2^{\rho_{max}}$	$\rho_{max}, \rho_{avg}, w_{avg}$
50, 25	134260	21432167, 1048576	20, 16, 18
60, 30	150554	48852312, 8388608	23, 16, 28
70, 35	92849	61196173, 16777216	24, 16, 38
80, 40	13403	17968626, 33554432	26, 16, 48
90, 45	22728	585495610, 134217728	27, 16, 58

Table 5

As long as ρ_{max} is small (say less than 25), it is possible to run the algorithm for quite large values of both m and p. Table 6 provides some evidence of this fact.

m, p	$\lvert\mathcal{S}_2\rvert$	$\lvert\mathcal{S}_1\rvert, 2^{\rho_{max}}$	$\rho_{max}, \rho_{avg}, w_{avg}$
400, 800	1578	1442226, 32768	15, 14, 393
400, 800	1588	2489671, 1048576	20, 14, 393

Table 6

Table 7 provides the actual running times taken by a simple C language implementation (Linux operating system on Pentium IV, 2.4 GHz CPU, 1 GB RAM).

m, p	$\lvert\mathcal{S}_2\rvert$	$\lvert\mathcal{S}_1\rvert$	time in seconds
100, 50	9976	5313186	50
100, 100	625	103455	2
100, 200	1297	71257808	830
400, 800	1588	2489671	1059

Table 7

Note that the following are true for the experiments performed above.

1. $|\mathcal{S}_1|$ is exponential in ρ_{max} and hence the run time of the algorithm is exponential in ρ_{max}.

2. The bounds on p are $0 \leq p \leq 2^m$. But for our algorithm to be useful the value of p can not be very large because in that case ρ_{max} can not be restricted to low values (say 25) and our algorithm will not work.

3. The value of w_{avg} should be high i.e., the ANF should be a very sparse multinomial and support of ANF should contain terms with very high weight.

4. In most cases, $|\mathcal{S}_2|$ is small compared to $|\mathcal{S}_1|$. Also $|\mathcal{S}_2|$ tends to grow as w_{avg} decreases.

5. In all the examples given above, the chosen functions were non-degenerate on all the variables.

Finally, in summary we note that for certain types of Boolean functions on large number of variables algorithm ComputeCv can be used to compute the Walsh transform for a small set of points. This can provide some useful information about the Boolean function. For example, one can obtain the weight (or size of support) of g by computing $W_g(0)$. Also the ability to probe the spectral domain of the function at random points can provide an estimate of the nonlinearity of the function.

3.6 Possible Improvements

In this section we discuss possible ways of improving the algorithm. From Theorem 13, we see that the complexity of the algorithm depends on $|\mathcal{S}_1|$ and $|\mathcal{S}_2|$. These two parameters are determined by the ANF of the function. We discuss two ways of improving the efficiency.

Let g be the function under consideration whose ANF is A_g. Suppose we apply an affine transformation to the variables of g to obtain f such that $|sup(A_f)| < |sup(A_g)|$. In such a situation it might be possible to improve the run time of the algorithm. The problem is in obtaining an affine transformation which will perform this task.

Another approach to improve the efficiency is to look for a more general and compact algebraic representation of a function. We briefly discuss this approach below and indicate the difficulty of this method.

Let g be of the form

$$g(x_1, x_2, \ldots, x_m) = \bigoplus_{(\alpha,\beta)\in list} (x_1 \oplus \beta_1)^{\alpha_1} \cdots (x_m \oplus \beta_m)^{\alpha_m}$$

where $list = ((\alpha^{(1)}, \beta^{(1)}), \cdots, (\alpha^{(p)}, \beta^{(p)}))$ and $\alpha^{(i)}, \beta^{(i)} \in \mathbb{F}_2^m$. The ANF of g is a special case of the above form where $\beta_1 = \cdots = \beta_m = 0$.

Let r be such that $1 \leq r \leq |list|$ and suppose $v = u^{(i_1)} \oplus \cdots \oplus u^{(i_r)}$ where $u^{(i_j)} \leq \alpha^{(i_j)}$ then we say $R = \{(u^{(i_1)}, \alpha^{(i_1)}, \beta^{(i_1)}), \cdots, (u^{(i_r)}, \alpha^{(i_r)}, \beta^{(i_r)})\}$ is a representation of v .

Let $R \in \mathcal{S}(v)$, such that $R = \{(u^{(i_1)}, \alpha^{(i_1)}, \beta^{(i_1)}), \cdots, (u^{(i_r)}, \alpha^{(i_r)}, \beta^{(i_r)})\}$ where $u^{(i_j)} \leq \alpha^{(i_j)}$ and for $1 \leq j \leq r$, let $k_{i_j} = \mathsf{wt}(\alpha^{(i_j)})$. Define

$$C(R) = \frac{(-1)^r}{2^{(k_{i_1}-1)+\cdots+(k_{i_r}-1)}}(-1)^{\langle u^{(i_1)},\beta^{(i_1)}\rangle \oplus \cdots \oplus \langle u^{(i_r)},\beta^{(i_r)}\rangle}.$$

Now it is simple to check that Theorem 9 holds. But in this situation it seems difficult to simplify the formula and obtain an algorithm to compute C_v.

3.7 Conclusion

In this chapter we have developed an algorithm to compute the Walsh transform of Boolean function at a point from its algebraic normal form. This can also be extended to evaluate the Walsh transform for a set of points. The advantage of our method is that in certain situations it is possible to run our algorithm to evaluate the Walsh transform (at a small set of points) of Boolean functions with large number of variables and hence can be used in cases where the fast Walsh transform is not applicable.

An important parameter in the study of a Boolean function is its nonlinearity. To compute nonlinearity, it is necessary to compute the Walsh transform at all the points. For Boolean functions on a large number of variables ($\geq$ 40), it is practically impossible to perform this task. In certain situations, our algorithm can be used to evaluate the Walsh transform at any particular point. In such situations, it is possible to use our algorithm in conjunction with some randomized heuristic (simulated annealing/ hill climbing/ genetic algorithms) to estimate the nonlinearity of the function.

Chapter 4

A General Correlation Theorem

In 2001, Nyberg proved three important correlation theorems and applied them to several cryptanalytic contexts. We continue the work of Nyberg in a more theoretical direction. We consider a general functional form and obtain its Walsh transform. Two of Nyberg's correlation theorems are seen to be special cases of our general functional form. S-box look-up, addition modulo 2^{2k} and X-OR are three frequently occurring operations in the design of symmetric ciphers. We consider two methods of combining these operations and in each case apply our main result to obtain the Walsh transform.

4.1 Introduction

Symmetric ciphers are a basic cryptographic primitive. In practice, symmetric ciphers are designed using nonlinear Boolean functions and S-boxes. One of the most effective methods of attacking symmetric ciphers is the technique of linear cryptanalysis [76] (see Section 2.3.2 for a brief description of this technique). The efficacy of this technique depends upon the ability to obtain good linear approximations of the constituent Boolean functions and S-boxes.

Linear approximations are studied using the technique of Walsh transform analysis. While it is usually easy to apply the Walsh transform to an individual constituent of a symmetric cipher, in general it is more difficult to apply the technique when a combination of primitives are used. This requires the development of a general methodology of Walsh transform applications.

One such important work has been done by Nyberg in [86]. This work unifies some of the previous approaches and obtains three key results on the Walsh transform of various functional forms. These are then applied to several typical cryptanalytic context.

The purpose of the current chapter is to continue the direction of research initiated in [86]. We obtain the Walsh transform for the following general functional form:

$$h(x_1, x_2, \ldots, x_{t+1}) \quad = \quad f(g_1(x_1, x_2), g_2(x_2, x_3), \ldots, g_t(x_t, x_{t+1}))$$

where each g_i is a map from $\mathbb{F}_2^{m_i+m_{i+1}}$ to $\mathbb{F}_2^{n_i}$ and f is a Boolean function from $\mathbb{F}_2^{n_1+\cdots+n_t}$ to $\mathbb{F}_2$. We obtain a closed form expression for the Walsh transform of h in terms of the Walsh transform of f and $g_1, \ldots, g_t$. We show that two of Nyberg's results in [86] are special cases of our theorem. This underlines the importance of our result in the context of symmetric cipher cryptanalysis.

We also consider two applications of our result. The operations of S-box look-up, X-OR and addition modulo 2^{2k} typically occur in the design of block and stream ciphers. We consider two possible ways of combining these operations. The first method is the situation where an S-box is applied to the X-OR of two outputs of the application of the S-box to two input bit strings. The second method considers the situation where an S-box is combined with addition modulo 2^{2k}. In both cases, we obtain complete expressions for the Walsh transform.

4.2 Convolution and Composition Theorems

The Convolution for two n-variable Boolean functions is well known (see for example [31]).

Theorem 14 (Convolution Theorem) *Let f and g be n-variable Boolean functions and $h(x) = g(x) \oplus f(x)$. Then*

$$W_h(u) \quad = \quad \frac{1}{2^n} \sum_{v \in \mathbb{F}_2^n} W_g(v) W_f(v \oplus u). \tag{4.1}$$

We next prove a generalization of the Convolution Theorem.

Theorem 15 (Generalized Convolution Theorem) *Let $g_1, \ldots, g_k$ be n-variable Boolean functions and $h(x) = g_1(x) \oplus \cdots \oplus g_k(x)$. Then for $u \in \mathbb{F}_2^n$*

$$W_h(u) \quad = \quad \frac{1}{2^s} \sum_{v \in \mathbb{F}_2^s} W_{g_k}(u \oplus u_{k-1}) W_{g_{k-1}}(u_{k-1} \oplus u_{k-2}) \cdots W_{g_2}(u_2 \oplus u_1) W_{g_1}(u_1) \tag{4.2}$$

$$= \frac{1}{2^s} \sum_{v \in \mathbb{F}_2^s} \prod_{i=1}^{k} W_{g_i}(u_i \oplus u_{i-1}) \tag{4.3}$$

where $s = n(k-1)$, $v = (u_1, \ldots, u_{k-1})$, $u_k = u$ *and* $u_0 = (0, \ldots, 0)$ *with each* $u_i \in \mathbb{F}_2^n$.

Proof : We prove the above Theorem by induction on k. For $k = 2$, the result follows from the Convolution Theorem. Assume that result holds for $(k-1) \geq 2$. We now apply the Convolution Theorem on the functions $g_k(x)$ and $f(x) = g_1(x) \oplus \cdots \oplus g_{k-1}(x)$. This gives

$$W_h(u) = \frac{1}{2^n} \sum_{u_{k-1} \in \mathbb{F}_2^n} W_f(u_{k-1}) W_{g_k}(u \oplus u_{k-1}) = \frac{1}{2^n} \sum_{u_{k-1} \in \mathbb{F}_2^n} W_f(u_{k-1}) W_{g_k}(u_k \oplus u_{k-1})$$

Now we invoke the induction hypothesis for $(k-1)$ on the function f to get

$$\begin{aligned} W_h(u) &= \frac{1}{2^n} \sum_{u_{k-1} \in \mathbb{F}_2^n} W_{g_k}(u_k \oplus u_{k-1}) \left(\frac{1}{2^{n(k-2)}} \sum_{(u_1, \ldots, u_{k-2})} \prod_{i=1}^{k-1} W_{g_i}(u_i \oplus u_{i-1}) \right) \\ &= \frac{1}{2^s} \sum_{v \in \mathbb{F}_2^s} \prod_{i=1}^{k} W_{g_i}(u_i \oplus u_{i-1}). \end{aligned}$$

This proves the result. ∎

Now we provide the Walsh transform of composition of an S-box and a Boolean function. A similar result is stated in [31] in terms of correlation matrices.

Theorem 16 (Composition Theorem) *Let* $g : \{0,1\}^n \to \{0,1\}^m$ *and* $f : \{0,1\}^m \to \{0,1\}$. *Then for any* $w \in \mathbb{F}_2^n$,

$$W_{(f \circ g)}(w) = \frac{1}{2^m} \sum_{v \in \mathbb{F}_2^m} W_f(v) W_{(l_v \circ g)}(w)$$

where $(l_v \circ g)(x) = \langle v, g(x) \rangle$.

Proof : From the inverse Walsh transform (2.3) we know

$$(-1)^{f(x)} = \frac{1}{2^m} \sum_{v \in \mathbb{F}_2^m} W_f(v) (-1)^{\langle v, x \rangle}.$$

Let $y = g(x)$. Then

$$\begin{aligned}(-1)^{(f\circ g)(x)} = (-1)^{f(g(x))} = (-1)^{f(y)} &= \frac{1}{2^m}\sum_{v\in\mathbb{F}_2^m} W_f(v)(-1)^{\langle v, y\rangle} \\ &= \frac{1}{2^m}\sum_{v\in\mathbb{F}_2^m} W_f(v)(-1)^{\langle v, g(x)\rangle} \\ &= \frac{1}{2^m}\sum_{v\in\mathbb{F}_2^m} W_f(v)(-1)^{(l_v\circ g)(x)}.\end{aligned}$$

So we have

$$\begin{aligned}W_{f\circ g}(w) &= \sum_{x\in\mathbb{F}_2^n}(-1)^{(f\circ g)(x)\oplus\langle w, x\rangle} \\ &= \frac{1}{2^m}\sum_{x\in\mathbb{F}_2^n}\sum_{v\in\mathbb{F}_2^m} W_f(v)(-1)^{(l_v\circ g)(x)\oplus\langle w, x\rangle} \\ &= \frac{1}{2^m}\sum_{v\in\mathbb{F}_2^m} W_f(v)\sum_{x\in\mathbb{F}_2^n}(-1)^{(l_v\circ g)(x)\oplus\langle w, x\rangle} \\ &= \frac{1}{2^m}\sum_{v\in\mathbb{F}_2^m} W_f(v)W_{(l_v\circ g)}(w).\end{aligned}$$

This proves the result. ∎

4.3 Correlation Theorem

In this section we prove the main correlation theorem. Let $g_1, \ldots, g_t$ be S-boxes, where for $1 \le i \le t$, $g_i : \mathbb{F}_2^{m_i+m_{i+1}} \to \mathbb{F}_2^{n_i}$. Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function, where $n = n_1 + \cdots + n_t$. Let $m = m_1 + \cdots + m_{t+1}$ and define a Boolean function $h : \mathbb{F}_2^m \to \mathbb{F}_2$ in the following manner.

$$h(x_1, \ldots, x_{t+1}) = f(g_1(x_1, x_2), g_2(x_2, x_3), \ldots, g_t(x_t, x_{t+1})) \tag{4.4}$$

where $x_i \in \mathbb{F}_2^{m_i}$ for $1 \le i \le t+1$. Our task in this section is to compute $W_h(u)$ for $u \in \mathbb{F}_2^m$. In Theorem 20 (see later) we show that $W_h(w)$ is given by the following expression.

$$W_h(w) = \frac{2^{m_1+m_{t+1}}}{2^{n+m}}\sum_{v\in\mathbb{F}_2^n} W_f(v)\sum_{u\in\mathbb{F}_2^M}\prod_{i=1}^{t} W_{(l_{v_i}\circ g_i)}(w_{t,i}\oplus w_{i-1,i}, w_{i,i+1})$$

where $v = (v_1, \ldots, v_t)$ with $v_i \in \mathbb{F}_2^{n_i}$, $M = m - m_1 - m_{t+1}$, $w = w_t = (w_{t,1}, \ldots, w_{t,t+1})$ with $w_{t,i} \in \mathbb{F}_2^{m_i}$, $u = (w_{1,2}, \ldots, w_{t-1,t})$ and for $1 \leq i \leq t$, $w_{i-1,i} \in \mathbb{F}_2^{m_i}$ with $w_{0,1} = (0, \ldots, 0)$. This result is obtained through a series of simplifications. Let $g' : \mathbb{F}_2^m \to \mathbb{F}_2^n$ be an S-box defined by

$$g'(x_1, \ldots, x_{t+1}) = (g_1(x_1, x_2), \ldots, g_t(x_t, x_{t+1})). \tag{4.5}$$

Using the Composition Theorem we have the following result.

Proposition 17 *Let h be defined by (4.4). Then*

$$W_h(u) = \frac{1}{2^n} \sum_{v \in \mathbb{F}_2^n} W_f(v) W_{(l_v \circ g')}(u).$$

For $v \in \mathbb{F}_2^n$, write $v = (v_1, \ldots, v_t)$, where $v_i \in \mathbb{F}_2^{n_i}$. We can write

$$\begin{aligned}
(l_v \circ g')(x_1, \ldots, x_{t+1}) &= l_v(g'(x_1, \ldots, x_{t+1}) \\
&= l_v((g_1(x_1, x_2), \ldots, g_t(x_t, x_{t+1})) \\
&= l_{v_1}(g_1(x_1, x_2)) \oplus \cdots \oplus l_{v_t}(g_t(x_t, x_{t+1})) \\
&= (l_{v_1} \circ g_1)(x_1, x_2)) \oplus \cdots \oplus (l_{v_t} \circ g_t)(x_t, x_{t+1})).
\end{aligned} \tag{4.6}$$

For $1 \leq i \leq t$, we define

$$h_i'(x_1, \ldots, x_{t+1}) = (l_{v_i} \circ g_i)(x_i, x_{i+1}). \tag{4.7}$$

Given bit strings $u_1, \ldots, u_k$ we define $\delta(u_1, \ldots, u_k) = 1$ if each $u_1, \ldots, u_k$ are all-zero bit strings; otherwise $\delta(u_1, \ldots, u_k) = 0$. The Walsh transform of h_i' is given by the following proposition.

Proposition 18 *Let h_i' be defined by (4.7). Then*

$$W_{h_i'}(u) = 2^{M_i} W_{l_{v_i} \circ g_i}(u_i, u_{i+1}) \delta(\omega_i)$$

where $u = (u_1, \ldots, u_{t+1})$, $M_i = m - m_i - m_{i+1}$ and $\omega_i = (u_1, \ldots, u_{i-1}, u_{i+2}, \ldots, u_{t+1})$.

Proof : We compute as follows.

$$W_{h_i'}(u) = \sum_{x \in \mathbb{F}_2^m} (-1)^{h_i'(x) \oplus \langle u, x \rangle}$$

$$
\begin{aligned}
&= \sum_{x \in \mathbb{F}_2^m} (-1)^{l_{v_i}(g_i(x_i, x_{i+1})) \oplus \langle (u_1, \ldots, u_{t+1}), (x_1, \ldots, x_{t+1}) \rangle} \\
&= \sum_{y_i \in \mathbb{F}_2^{M_i}} (-1)^{\langle \omega_i, y_i \rangle} \times \sum_{(x_i, x_{i+1}) \in \mathbb{F}_2^{m_i + m_{i+1}}} (-1)^{l_{v_i}(g_i(x_i, x_{i+1})) \oplus \langle (u_i, u_{i+1}), (x_i, x_{i+1}) \rangle} \\
&= W_{(l_{v_i} \circ g_i)}(u_i, u_{i+1}) \sum_{y_i \in \mathbb{F}_2^{M_i}} (-1)^{\langle \omega_i, y_i \rangle} \\
&= 2^{M_i} W_{(l_{v_i} \circ g_i)}(u_i, u_{i+1}) \delta(\omega_i)
\end{aligned}
$$

where $y_i = (x_1, \ldots, x_{i-1}, x_{i+2}, \ldots, x_{t+1})$. The last statement follows from the fact that $\sum_{z \in \mathbb{F}_2^r} (-1)^{\langle w, z \rangle} = 2^{|w|} \delta(w)$. This completes the proof. ∎

Now we obtain the Walsh transform of $(l_v \circ g')$.

Lemma 19 *Let g' be defined as in (4.5). Then*

$$
W_{(l_v \circ g')}(w) = \frac{2^{m_1 + m_{t+1}}}{2^m} \sum_{u \in \mathbb{F}_2^M} \prod_{i=1}^{t} W_{(l_{v_i} \circ g_i)}(w_{t,i} \oplus w_{i-1,i}, w_{i,i+1})
$$

where $M = m - m_1 - m_{t+1}$, $w = w_t = (w_{t,1}, \ldots, w_{t,t+1})$ with $w_{t,i} \in \mathbb{F}_2^{m_i}$, $u = (w_{1,2}, \ldots, w_{t-1,t})$ and for $1 \leq i \leq t$, $w_{i-1,i} \in \mathbb{F}_2^{m_i}$ with $w_{0,1} = (0, \ldots, 0)$.

Proof : Using (4.6), (4.7) and the Generalized Convolution Theorem, we have

$$
W_{(l_v \circ g')}(w_t) = \frac{1}{2^{m(t-1)}} \sum_{(w_1, \ldots, w_{t-1})} \prod_{i=1}^{t} W_{h_i'}(w_i \oplus w_{i-1})
$$

where $w_i \in \mathbb{F}_2^m$ for $0 \leq i \leq t-1$ and $w_0 = (0, \ldots, 0)$. For $0 \leq i \leq t$, write $w_i = (w_{i,1}, \ldots, w_{i,t+1})$ with $w_{i,j} \in \mathbb{F}_2^{m_j}$. Set $\alpha_{i,j} = w_{i,j} \oplus w_{i-1,j}$ and $\alpha_i = (\alpha_{i,1}, \ldots, \alpha_{i,t+1}) = w_i \oplus w_{i-1}$.
Set $\beta_i^t = (\alpha_{i,1}, \ldots, \alpha_{i,i-1}, \alpha_{i,i+2}, \ldots, \alpha_{i,t+1})$. Using Proposition 18, we obtain

$$
\begin{aligned}
W_{(l_v \circ g')}(w_t) &= \frac{1}{2^{m(t-1)}} \sum_{(w_1, \ldots, w_{t-1})} \prod_{i=1}^{t} W_{h_i'}(\alpha_i) \\
&= \frac{1}{2^{m(t-1)}} \sum_{(w_1, \ldots, w_{t-1})} \prod_{i=1}^{t} 2^{M_i} W_{l_{v_i} \circ g_i}(\alpha_{i,i}, \alpha_{i,i+1}) \delta(\beta_i^t) \\
&= \frac{2^{m_1 + m_{t+1}}}{2^m} \sum_{(w_1, \ldots, w_{t-1})} \prod_{i=1}^{t} W_{l_{v_i} \circ g_i}(\alpha_{i,i}, \alpha_{i,i+1}) \delta(\beta_i^t)
\end{aligned}
$$

The proof now follows from the following claim.

Claim: For $t \geq 2$, we have

$$\sum_{(w_1,\ldots,w_{t-1})} \prod_{i=1}^{t} W_{l_{v_i} \circ g_i}(\alpha_{i,i}, \alpha_{i,i+1}) \delta(\beta_i^t) \quad = \quad \sum_{(w_{1,2},\ldots,w_{t-1,t})} \prod_{i=1}^{t} W_{(l_{v_i} \circ g_i)}(w_{t,i} \oplus w_{i-1,i}, w_{i,i+1})$$

Proof : The claim is proved by induction on $t \geq 2$. Let $\mathcal{L}_t$ (resp. $\mathcal{R}_t$) be the left (resp. right) side of the claim.

Base : Case $t = 2$. In this case, the left side becomes

$$\mathcal{L}_2 = \sum_{w_1} W_{l_{v_1} \circ g_1}(\alpha_{1,1}, \alpha_{1,2}) \delta(\beta_1^2) W_{l_{v_2} \circ g_2}(\alpha_{2,2}, \alpha_{2,3}) \delta(\beta_2^2).$$

Note that $\beta_1^2 = \alpha_{1,3} = w_{1,3} \oplus w_{0,3} = w_{1,3}$ and $\beta_2^2 = \alpha_{2,1} = w_{2,1} \oplus w_{1,1}$. Substituting the value of α's, we write the above expression as

$$\begin{aligned}
\mathcal{L}_2 &= \sum_{w_{1,1}} \sum_{w_{1,2}} \sum_{w_{1,3}} W_{l_{v_1} \circ g_1}(w_{1,1}, w_{1,2}) W_{l_{v_2} \circ g_2}(w_{2,2} \oplus w_{1,2}, w_{2,3} \oplus w_{1,3}) \delta(w_{1,3}) \delta(w_{2,1} \oplus w_{1,1}) \\
&= \sum_{w_{1,1}} \sum_{w_{1,2}} W_{l_{v_1} \circ g_1}(w_{1,1}, w_{1,2}) W_{l_{v_2} \circ g_2}(w_{2,2} \oplus w_{1,2}, w_{2,3}) \delta(w_{2,1} \oplus w_{1,1}) \\
&= \sum_{w_{1,2}} \sum_{w_{1,1}} W_{l_{v_1} \circ g_1}(w_{1,1}, w_{1,2}) W_{l_{v_2} \circ g_2}(w_{2,2} \oplus w_{1,2}, w_{2,3}) \delta(w_{2,1} \oplus w_{1,1}) \\
&= \sum_{w_{1,2}} W_{l_{v_1} \circ g_1}(w_{2,1}, w_{1,2}) W_{l_{v_2} \circ g_2}(w_{2,2} \oplus w_{1,2}, w_{2,3})
\end{aligned}$$

which is equal to $\mathcal{R}_2$, i.e., the right side of the claim for $t = 2$.

Induction hypothesis : Assume that the claim is true for $t - 1$.

Induction step : We now prove the result for t. We will use the induction hypothesis in this case. The main difficulty arises from the fact that the length of the w's and the α's increases by one. We have to take care of this while applying the induction hypothesis. First note that for $1 \leq i \leq t-1$, we have $\delta(\beta_i^t) = \delta(\beta_i^{t-1}) \delta(\alpha_{i,t+1})$. Now we compute

$$\begin{aligned}
\mathcal{L}_t &= \sum_{(w_1,\ldots,w_{t-2})} \prod_{i=1}^{t-1} W_{l_{v_i} \circ g_i}(\alpha_{i,i}, \alpha_{i,i+1}) \delta(\beta_i^t) \sum_{w_{t-1}} W_{l_{v_t} \circ g_t}(\alpha_{t,t}, \alpha_{t,t+1}) \delta(\beta_t^t) \\
&= \sum_{(w_1,\ldots,w_{t-2})} \prod_{i=1}^{t-1} W_{l_{v_i} \circ g_i}(\alpha_{i,i}, \alpha_{i,i+1}) \delta(\beta_i^{t-1}) \\
&\quad \times \sum_{(w_{1,t+1},\ldots,w_{t-2,t+1})} \delta(\alpha_{1,t+1}) \ldots \delta(\alpha_{t-2,t+1})
\end{aligned}$$

$$\times \sum_{w_{t-1}} W_{l_{v_t} \circ g_t}(\alpha_{t,t}, \alpha_{t,t+1}) \delta(\beta_t^t)$$

$$= \sum_{(w_1,\ldots,w_{t-2})} \prod_{i=1}^{t-1} W_{l_{v_i} \circ g_i}(\alpha_{i,i}, \alpha_{i,i+1}) \delta(\beta_i^{t-1})$$
$$\times \sum_{(w_{t-1,1},\ldots,w_{t-1,t})} \delta(\alpha_{t,1}) \delta(\alpha_{t,2}) \ldots \delta(\alpha_{t,t-1})$$
$$\times \sum_{(w_{1,t+1},\ldots,w_{t-2,t+1},w_{t-1,t+1})} W_{l_{v_t} \circ g_t}(\alpha_{t,t}, \alpha_{t,t+1}) \delta(\alpha_{1,t+1}) \ldots \delta(\alpha_{t-2,t+1}).$$

Note that $\alpha_{1,t+1} = w_{1,t+1} \oplus w_{0,t+1} = w_{1,t+1}$ and for $i > 1$, $\alpha_{i,t+1} = w_{i,t+1} \oplus w_{i-1,t+1}$. Thus the expression within the last sum evaluates to $W_{l_{v_t} \circ g_t}(\alpha_{t,t}, \alpha_{t,t+1})$ only under the condition $w_{1,t+1} = \cdots = w_{t-2,t+1} = (0, \ldots, 0)$. Also the expression within the second sum evaluates to 1 only under the condition $w_{t,i} = w_{t-1,i}$ for $1 \leq i \leq t-1$. We invoke the induction hypothesis on the first sum to obtain.

$$\begin{aligned} \mathcal{L}_t &= \sum_{(w_{1,2},\ldots,w_{t-2,t-1})} \prod_{i=1}^{t-1} W_{(l_{v_i} \circ g_i)}(w_{t-1,i} \oplus w_{i-1,i}, w_{i,i+1}) \\ &\quad \times \sum_{w_{t-1,t}} W_{l_{v_t} \circ g_t}(\alpha_{t,t}, \alpha_{t,t+1}) \\ &= \mathcal{R}_t. \end{aligned}$$

This completes the proof of the claim and the lemma. ∎

Finally using Proposition 17 and Lemma 19 we obtain the Walsh transform of W_h.

Theorem 20 *Let h be defined as in (4.4). Then*

$$W_h(w) = \frac{2^{m_1+m_{t+1}}}{2^{n+m}} \sum_{v \in \mathbb{F}_2^n} W_f(v) \sum_{u \in \mathbb{F}_2^M} \prod_{i=1}^{t} W_{(l_{v_i} \circ g_i)}(w_{t,i} \oplus w_{i-1,i}, w_{i,i+1}) \tag{4.8}$$

where $v = (v_1, \ldots, v_t)$ with $v_i \in \mathbb{F}_2^{n_i}$, $M = m - m_1 - m_{t+1}$, $w = w_t = (w_{t,1}, \ldots, w_{t,t+1})$ with $w_{t,i} \in \mathbb{F}_2^{m_i}$, $u = (w_{1,2}, \ldots, w_{t-1,t})$ and for $1 \leq i \leq t$, $w_{i-1,i} \in \mathbb{F}_2^{m_i}$ with $w_{0,1} = (0, \ldots, 0)$.

4.4 Nyberg's Correlation Theorems

In [86], Nyberg stated three correlation theorems – Theorems 3, 4 and 5 – which have important applications to cryptanalysis. Of these, Theorem 4 has been proved in [85] and the other two theorems are proved in [86]. In this section, we show that Nyberg's correlation

theorems – Theorem 3 and 5 of [86] – can be obtained as special cases of Theorem 20. First we rewrite Theorem 20 for $t = 2$.

Theorem 21 *Let h be defined as in (4.4) and $t = 2$. Then $W_h(w_{2,1}, w_{2,2}, w_{2,3})$*

$$= \frac{2^{m_1+m_3}}{2^{m+n}} \sum_{v \in \mathbb{F}_2^n} W_f(v) \sum_{w_{1,2} \in \mathbb{F}_2^{m_2}} W_{l_{v_1} \circ g_1}(w_{2,1}, w_{1,2}) W_{l_{v_2} \circ g_2}(w_{2,2} \oplus w_{1,2}, w_{2,3}) \tag{4.9}$$

where $v = (v_1, v_2)$ with $v_i \in \mathbb{F}_2^{n_i}$.

A special case is obtained when f is the linear function $f(a_1, \ldots, a_n) = a_1 \oplus \cdots \oplus a_n$. In this case $W_f(1, \ldots, 1) = 2^n$ and $W_f(v) = 0$ for $v \in \mathbb{F}_2^n \setminus \{(1, \ldots, 1)\}$. Also $v_1 = (1, \ldots, 1) \in \mathbb{F}_2^{n_1}$ and $v_2 = (1, \ldots, 1) \in \mathbb{F}_2^{n_2}$. We denote by $\mathbf{1}_k$ the all one vector of length k. When the value of k is clear from the context, we will simply write $\mathbf{1}$. We have the following corollary to Theorem 21.

Corollary 22 *Let $h(x_1, x_2, x_3) = \langle \mathbf{1}_n, (g_1(x_1, x_2), g_2(x_2, x_3)) \rangle$. Then*

$$W_h(w_{2,1}, w_{2,2}, w_{2,3}) = \frac{2^{m_1+m_3}}{2^m} \sum_{w_{1,2} \in \mathbb{F}_2^{m_2}} W_{l_{1_{n_1}} \circ g_1}(w_{2,1}, w_{1,2}) W_{l_{1_{n_2}} \circ g_2}(w_{2,2} \oplus w_{1,2}, w_{2,3}).$$

Substituting $n_1 = n_2 = 1$ we obtain $h(x_1, x_2, x_3) = g_1(x_1, x_2) \oplus g_2(x_2, x_3)$. Further, substituting $w_{2,2} = 0$ in Corollary 22 we obtain the the following result of Nyberg [86] (stated in terms of Walsh transform).

Theorem 23 (Nyberg [86], Theorem 5) *Let $h(x_1, x_2, x_3) = g_1(x_1, x_2) \oplus g_2(x_2, x_3)$. Then*

$$W_h(w_{2,1}, 0, w_{2,3}) = \frac{2^{m_1+m_3}}{2^m} \sum_{w_{1,2} \in \mathbb{F}_2^{m_2}} W_{g_1}(w_{2,1}, w_{1,2}) W_{g_2}(w_{1,2}, w_{2,3}).$$

Now we turn to Theorem 3 of Nyberg [86]. For this we make the following substitution in the definition of h: $m_2 = 0$, $g = g_2$, $g_1(x) = x$ and hence $n_1 = m_1$. Thus h is now of the form $h(x_1, x_3) = f(x_1, g(x_3))$. In this situation, we have

$$\sum_{w_{1,2} \in \mathbb{F}_2^{m_2}} W_{l_{v_1} \circ g_1}(w_{2,1}, w_{1,2}) W_{l_{v_2} \circ g_2}(w_{2,2} \oplus w_{1,2}, w_{2,3}) = W_{l_{v_1} \circ g_1}(w_{2,1}) W_{l_{v_2} \circ g_2}(w_{2,3}).$$

Since $g_1(x) = x$, we have $(l_{v_1} \circ g_1)(x) = l_{v_1}(g_1(x)) = l_{v_1}(x) = \langle v_1, x \rangle$ and hence $W_{l_{v_1} \circ g_1}(w_{2,1}) = 2^{m_1}\delta(v_1 \oplus w_{2,1}) = 2^{n_1}\delta(v_1 \oplus w_{2,1})$, since $m_1 = n_1$. Thus $W_{l_{v_1} \circ g_1}(w_{2,1}) = 2^{n_1}$ if $v_1 = w_{2,1}$ and is equal to 0 otherwise. Let the right side of (4.9) be $\mathcal{A}$. In this case $\mathcal{A}$ becomes

$$\begin{aligned}
\mathcal{A} &= \frac{1}{2^{n_1+n_2}} \sum_{v \in \mathbb{F}_2^n} W_f(v) \sum_{w_{1,2} \in \mathbb{F}_2^{m_2}} W_{l_{v_1} \circ g_1}(w_{2,1}, w_{1,2}) W_{l_{v_2} \circ g_2}(w_{2,2} \oplus w_{1,2}, w_{2,3}) \\
&= \frac{1}{2^{n_1+n_2}} \sum_{(v_1,v_2) \in \mathbb{F}_2^n} W_f(v_1, v_2) W_{l_{v_1} \circ g_1}(w_{2,1}) W_{l_{v_2} \circ g_2}(w_{2,3}) \\
&= \frac{1}{2^{n_2}} \sum_{v_2 \in \mathbb{F}_2^{n_2}} W_f(w_{2,1}, v_2) W_{l_{v_2} \circ g}(w_{2,3}).
\end{aligned}$$

Now we have the following result of Nyberg [86] again stated in terms of Walsh transform.

Theorem 24 (Nyberg [86], Theorem 3) *Let $h(x_1, x_3) = f(x_1, g(x_3))$. Then*

$$W_h(w_{2,1}, w_{2,3}) = \frac{1}{2^{n_2}} \sum_{v_2 \in \mathbb{F}_2^{n_2}} W_f(w_{2,1}, v_2) W_{l_{v_2} \circ g}(w_{2,3}).$$

4.5 Applications

In this section, we consider two other applications of Theorem 20. These are based on operations which typically occur in design of symmetric ciphers, namely S-box look-up, addition modulo 2^{2k} and the X-OR operation. We consider two possible ways of combining these operations and obtain the Walsh transform in each case.

4.5.1 Brick Layering

In this section, we consider a map of the form

$$h(x, y, z) = g(g(x, y) \oplus g(y, z)). \tag{4.10}$$

Figure 4.1 gives a diagrammatic view of the map. The term brick layering was used in [31] to denote a map which consists of several parallel applications of different S-boxes on disjoint subsets of the inputs which also form a partition of the input. In our case the X-OR and the second application of g is used to "glue" the outputs of the first two applications of g. In block cipher applications g is usually a $2k$-bit to $2k$-bit S-box possibly the inverse function

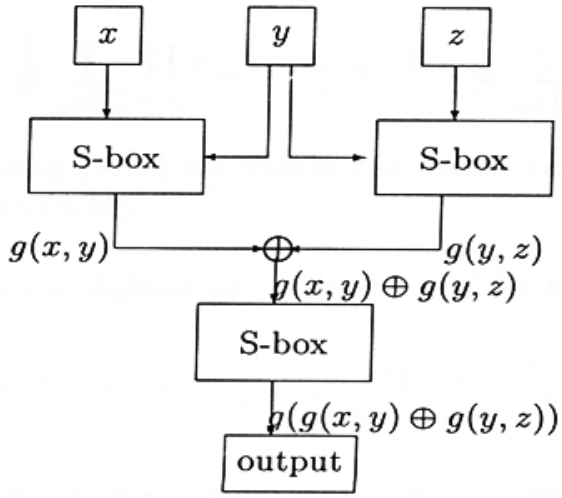


Figure 4.1: Brick layering transformation.

over $GF(2^{2k})$. Hence we assume that g is an $2k$-bit to $2k$-bit map and x, y and z are all k-bit strings. Let $g_1, \cdots, g_{2k}$ be the component functions of g. Let $\nu_i(x, y, z) = g_i(x, y) \oplus g_i(y, z)$ and $\nu = (\nu_1, \ldots, \nu_{2k})$. Applying Corollary 22 with $n_1 = n_2 = 1$ we obtain

$$W_{\nu_i}(\delta_1, \delta_2, \delta_3) = \frac{1}{2^k} \sum_{u \in \mathbb{F}_2^k} W_{g_i}(u \oplus \delta_2, \delta_3) W_{g_i}(u, \delta_1) \tag{4.11}$$

Let $h_1, \ldots, h_{2k}$ be the component functions of h. We have

$$h_i(x, y, z) = g_i(\nu_1(x, y, z), \ldots, \nu_{2k}(x, y, z)).$$

Now applying the Composition Theorem we get for $(\gamma_1, \gamma_2, \gamma_3) \in \{0, 1\}^{3k}$,

$$W_{h_i}(\gamma_1, \gamma_2, \gamma_3) = \frac{1}{2^{2k}} \sum_{v \in \mathbb{F}_2^{2k}} W_{g_i}(v) W_{(l_v \circ \nu)}(\gamma_1, \gamma_2, \gamma_3) \tag{4.12}$$

The next step is to compute the Walsh transform of $(l_v \circ \nu)$. Let v be of weight r (i.e., $\mathsf{wt}(v) = r$) with the bits in the $j_1, \ldots, j_r$th positions to be 1 and all others to be 0. Then $(l_v \circ \nu)(x, y, z) = \mu_1(x, y, z) \oplus \cdots \oplus \mu_r(x, y, z)$, where $\mu_i = \nu_{j_i}$ for $1 \leq i \leq r$. Further, let $s = 3k(\mathsf{wt}(v) - 1)$, $w = (u_1, \ldots, u_{r-1})$, with each $u_i \in \mathbb{F}_2^{3k}$, $u_0 = (0, \ldots, 0) \in \mathbb{F}_2^{3k}$ and

$u_r = (\gamma_1, \gamma_2, \gamma_3)$. Now applying the Generalized Convolution Theorem, we have

$$W_{(l_\nu \circ \nu)}(\gamma_1, \gamma_2, \gamma_3) = \frac{1}{2^s} \sum_{w \in \mathbb{F}_2^s} \prod_{i=1}^{r} W_{\mu_i}(u_i \oplus u_{i-1}) = \frac{1}{2^s} \sum_{w \in \mathbb{F}_2^s} \prod_{i=1}^{r} W_{\nu_{j_i}}(u_i \oplus u_{i-1}).$$

Substituting in (4.12) and using (4.11) we obtain the Walsh transform for any component function of the map defined in (4.10).

Theorem **25** *Let h be an S-box defined as in (4.10) and let $h_1, \ldots, h_{2k}$ be its component functions. Then*

$$W_{h_i}(\gamma_1, \gamma_2, \gamma_3) = \frac{1}{2^{2k}} \sum_{v \in \mathbb{F}_2^{2k}} W_{g_i}(v) \frac{1}{2^{s+kr}}$$

$$\times \sum_{(u_1, \ldots, u_{r-1})} \prod_{i=1}^{r} \sum_{w \in \mathbb{F}_2^k} W_{g_{j_i}}(w \oplus u_{i,2} \oplus u_{i-1,2}, u_{i,3} \oplus u_{i-1,3}) W_{g_{j_i}}(w, u_{i,1} \oplus u_{i-1,1})$$

where $u_i = (u_{i,1}, u_{i,2}, u_{i,3})$ with $u_{i,j} \in \mathbb{F}_2^k$ and for each $v \in \mathbb{F}_2^{2k}$, $r = \mathsf{wt}(v)$, $u_r = (\gamma_1, \gamma_2, \gamma_3)$.

Theorem 25 provides the complete expression for the Walsh transform of any h_i.

4.5.2 Substitute-and-Add

In this section, we consider the map obtained by alternate application of an S-box and sum modulo 2^{2k}. More precisely, we consider the following map.

$$h(x, y) = g(x) \boxed{+} g(y). \tag{4.13}$$

The map is shown diagrammatically in Figure 4.2. We obtain the complete Walsh transform of each component function of h. As before we consider g to be an $2k$-bit to $2k$-bit S-box whose component functions are $g_1, \ldots, g_{2k}$.

For the sake of convenience, we will denote $\nu(x, y) = x \boxed{+} y$. The map ν is conveniently described by separating the carry part. Suppose $(a_1, \ldots, a_{2k})$ and $(b_1, \ldots, b_{2k})$ are the inputs to $\boxed{+}$. Then the carry in the ith position is given by the function $c(a_1, \ldots, a_i, b_1, \ldots, b_i)$ and $\nu_i = a_i \oplus b_i \oplus c_i$. The Walsh transform of the carry function has been described in [116]. Let $h_1, \ldots, h_{2k}$ be the component functions of h. We can now write

$$h_i(x, y) = g_i(x) \oplus g_i(y) \oplus c_i(g_1(x), \ldots, g_i(x), g_1(y), \ldots, g_i(y)).$$

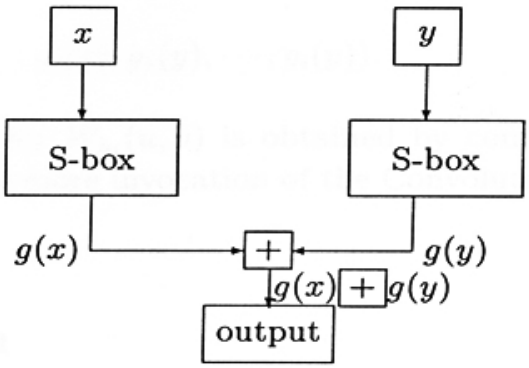


Figure 4.2: Substitute and add.

Let $f_i(x,y) = c_i(g_1(x), \cdots, g_i(x), g_1(y), \cdots, g_i(y))) = c_i(\mu_i(x,y))$ where

$$\mu_i : \{0,1\}^{4k} \to \{0,1\}^{2i}$$

is defined as $\mu_i(x,y) = (g_1(x), \cdots, g_i(x), g_1(y), \cdots, g_i(y))$. Applying the Composition Theorem for $(u,v) \in \{0,1\}^{2k}$, we have

$$W_{f_i}(u,v) = \frac{1}{2^{2i}} \sum_{w \in \mathbb{F}_2^{2i}} W_{c_i}(w) W_{(l_w \circ \mu_i)}(u,v)$$

Let $\lambda_i(x,y) = g_i(x) \oplus g_i(y)$ and hence $W_{\lambda_i}(u,v) = W_{g_i}(u) W_{g_i}(v)$. We have $h_i(x,y) = \lambda_i(x,y) \oplus f_i(x,y)$. Using the Convolution Theorem we have

$$\begin{aligned} W_{h_i}(u,v) &= \frac{1}{2^{4k}} \sum_{(u_1,v_1) \in \mathbb{F}_2^{2k+2k}} W_{\lambda_i}(u_1,v_1) W_{f_i}(u \oplus u_1, v \oplus v_1) \\ &= \frac{1}{2^{4k}} \sum_{(u_1,v_1) \in \mathbb{F}_2^{2k+2k}} W_{g_i}(u_1) W_{g_i}(v_1) W_{f_i}(u \oplus u_1, v \oplus v_1) \\ &= \frac{1}{2^{4k+2i}} \sum_{(u_1,v_1) \in \mathbb{F}_2^{2k+2k}} W_{g_i}(u_1) W_{g_i}(v_1) \sum_{w \in \mathbb{F}_2^{2i}} W_{c_i}(w) W_{(l_w \circ \mu_i)}(u \oplus u_1, v \oplus v_1). \end{aligned}$$

Thus we get the following result.

Theorem 26 *Let h be defined as in (4.13) and $h_1, \ldots, h_{2k}$ be its component functions. Then*

$$W_{h_i}(u,v) = \frac{1}{2^{4k+2i}} \sum_{(u_1,v_1) \in \mathbb{F}_2^{2k+2k}} W_{g_i}(u_1) W_{g_i}(v_1) \sum_{w \in \mathbb{F}_2^{2i}} W_{c_i}(w) W_{(l_w \circ \mu_i)}(u \oplus u_1, v \oplus v_1)$$

where $\mu_i(x, y) = (g_1(x), \cdots, g_i(x), g_1(y), \cdots, g_i(y))$.

The complete expression for $W_{h_i}(u, v)$ is obtained by computing the Walsh transform of $(l_w \circ \mu_i)$. This requires one more invocation of the Convolution Theorem and hence involves another summation.

4.6 Conclusion

We have proved a result which provides the Walsh transform of a general functional form. As special cases, we obtain two of Nyberg's correlation theorems proved in [86]. We consider two applications of our results. These applications combine S-box look-up with addition modulo 2^{2k} and the X-OR operation. In each case we obtain complete expressions for the Walsh transform. A possible future research problem is to apply our techniques to perform reduced round cryptanalysis of practical block ciphers.

Chapter 5

Construction of Perfect Nonlinear Functions Satisfying Higher Order SAC

In this chapter we consider the problem of constructing perfect nonlinear multi-output Boolean functions satisfying higher order strict avalanche criteria (SAC). Our first construction is an infinite family of 2-output perfect nonlinear functions satisfying higher order SAC. This construction is achieved using the theory of bilinear forms and symplectic matrices. Next we build on a known connection between 1-factorization of a complete graph and SAC to construct more examples of 2 and 3-output perfect nonlinear functions satisfying higher order SAC. In certain cases, the constructed S-boxes have optimal trade-off between the following parameters: numbers of input and output variables, nonlinearity and order of SAC. In case the number of input variables is odd, we modify the construction for perfect nonlinear S-boxes to obtain a construction for maximally nonlinear S-boxes satisfying higher order SAC. Our constructions present the first examples of perfect nonlinear and maximally nonlinear multioutput S-boxes satisfying higher order SAC. Lastly, we present a simple method for improving the degree of the constructed functions with a small trade-off in nonlinearity and the SAC property. This yields functions which have possible applications in the design of block ciphers.

5.1 Introduction

Multi-output Boolean functions are usually called S-boxes (see Section 2.2) and are used as basic primitives for designing symmetric ciphers. For example, the S-boxes used in DES have $n = 6$ and $m = 4$ and the S-box used in the design of AES has $n = m = 8$. For $m > 1$, construction of perfect nonlinear S-boxes satisfying SAC(k) for $k > 0$ has been a open problem. In this chapter, we (partially) solve this problem by providing constructions of perfect nonlinear S-boxes with $m = 2, 3$ and satisfying SAC(k) for $k \geq 1$. Our contributions are the following.

- Construction of an infinite family of 2-output perfect nonlinear S-boxes satisfying higher order SAC. More precisely, for each even $n \geq 6$, we construct a 2-output perfect nonlinear S-box satisfying SAC($(n/2) - 2$).
- In an earlier paper [68], a 1-factorization of the complete graph on n-vertices was used to construct S-boxes satisfying higher order SAC. However, the S-boxes constructed in [68] did not satisfy perfect nonlinearity. We make a more detailed analysis of the connection between 1-factorization and higher order SAC to construct 2 and 3-output *perfect nonlinear* S-boxes satisfying higher order SAC.
- In certain cases, the functions that we construct achieve the best possible trade-off among the following parameters: number of input variables, number of output variables, nonlinearity and order of SAC. Hence for such functions, it is not possible to improve any one parameter without changing some other parameter.
- For small n, our constructions provide S-boxes which cannot be obtained from the currently known constructions [68, 69, 103]. Some examples of such functions are the following.
  - 8-input, 2-output perfect nonlinear S-box satisfying SAC(3).
  - 8-input, 3-output perfect nonlinear S-box satisfying SAC(1).
  - 10-input, 3-output perfect nonlinear S-box satisfying SAC(3).

  The first and third S-boxes achieve the best possible trade-off.
- Our constructions are based on bilinear forms and symplectic matrices used in the study of second order Reed-Muller code. We show that if n is odd, then the construction for $(n + 1)$ can be modified to obtain maximally nonlinear S-boxes satisfying higher order SAC.

- We provide a simple technique for improving the degree of an S-box with a small sacrifice in nonlinearity and the SAC property. This results in S-boxes which have possible applications in the design of symmetric ciphers

5.2 Basic Results

An n-variable Boolean function g of degree ≤ 2 can be written as (see [73, page 434]) $g(x) = xQx^T \oplus Lx^T \oplus b$ where $Q = (q_{ij})$ is an upper triangular $n \times n$ binary matrix, $L = (l_1, \cdots, l_n)$ is a binary vector and b is 0 or 1. The expression xQx^T is called a quadratic form and Lx^T is called a linear form. Let $B = Q \oplus Q^T$. Then B is a binary symmetric matrix with zero diagonal. Such a matrix is called a symplectic matrix (see [73, page 435]). Thus from a quadratic Boolean function we can define a symplectic matrix. *Conversely, given a symplectic matrix B we can construct a quadratic Boolean function by reversing the above steps. We denote this Boolean function by f_B.*

It is known that the rank of a symplectic matrix is always even [73, page 436]. The nonlinearity of the Boolean function g is related to the rank of B by the following result [73, page 441].

Proposition 27 *Let g be a quadratic n-variable Boolean function and B be its associated symplectic form. Then the nonlinearity of g is equal to $2^{n-1} - 2^{n-h-1}$, where the rank of B is $2h$.*

Consequently, a quadratic Boolean function is bent if and only if the associated symplectic matrix is of full rank.

We will be interested in nonlinear quadratic functions satisfying higher order SAC. From Proposition 27, a convenient way to study the nonlinearity of quadratic functions is through the rank of the associated symplectic matrix. We now develop the basic relationships between the nonlinearity and SAC property of a quadratic S-box and the symplectic matrices associated with the component functions.

Proposition 28 *Let f be a quadratic Boolean function and B its associated symplectic matrix. Then f satisfies SAC(k) if and only if for all $1 \leq i \leq n$, we have $\mathsf{wt}(r^{(i)}) \geq k + 1$, where $r^{(i)}$ is the i^{th} row of B. (Since B is symmetric, a similar property holds for the columns of B.)*

Proof : Let $f(x) = xQx^T \oplus Lx^T \oplus b$. Let α be such that only the ith component of α is 1 and all other components are zero. Further, let the ith column of Q be $a^{(i)}$ and the ith row of Q be $b^{(i)}$. Then $r^{(i)} = (a^{(i)})^T \oplus b^{(i)}$. We have

$$\begin{aligned} f(x) \oplus f(x \oplus \alpha) &= xQx^T \oplus (x \oplus \alpha)Q(x \oplus \alpha)^T \oplus L\alpha^T \\ &= xQ\alpha^T \oplus \alpha Qx^T \oplus L\alpha^T \oplus \alpha Q\alpha^T \\ &= \langle b^{(i)} \oplus (a^{(i)})^T, x \rangle \oplus L\alpha^T \oplus \alpha Q\alpha^T \\ &= \langle r^{(i)}, x \rangle \oplus L\alpha^T \oplus \alpha Q\alpha^T \end{aligned}$$

Note that $L\alpha^T \oplus \alpha Q\alpha^T$ is a constant. Now suppose $\mathsf{wt}(r^{(i)}) \geq k+1$. Let $g(x)$ be a function obtained by setting any k bits of $f(x) \oplus f(x \oplus \alpha)$ to constant values. Then $\langle r^{(i)}, x \rangle$ is a non constant linear function and hence $g(x)$ is balanced. Conversely, if $\mathsf{wt}(r^{(i)}) \leq k$, then we can set k variables to constant values in such a manner that $g(x)$ is a constant function. This proves the result. ∎

Let $f = (f_1, \cdots, f_m)$ be an (n, m) quadratic S-box. Then each of the component functions f_i is an n-variable quadratic Boolean function. For $1 \leq i \leq m$, let B_i be the symplectic matrix associated with the component function f_i. Clearly, any linear combination of symplectic matrices is also a symplectic matrix. We have the following extension of Proposition 28.

Lemma 29 *Let f be an (n, m) S-box with quadratic component functions f_i and associated symplectic forms B_i for $1 \leq i \leq m$. Then f satisfies $SAC(k)$ if and only if the weight of every row in any nonzero linear combination of the B_i's is at least $k+1$.*

A similar result for nonlinearity can be stated by extending Proposition 27.

Lemma 30 *Let f be an (n, m) S-box with quadratic component functions f_i and associated symplectic forms B_i for $1 \leq i \leq m$. The nonlinearity of f is $2^{n-1} - 2^{n-h-1}$, where $2h$ is the minimum of the ranks of any nonzero linear combination of the B_i's. Consequently for even n, the S-box f is perfect nonlinear if and only if every nonzero linear combination of the B_i's has full rank. Similarly, for odd n, the S-box f is maximally nonlinear if and only if every nonzero linear combination of the B_i's has rank $(n-1)$.*

Lemmas 29 and 30 will be used in proving the correctness of our constructions in the next sections.

5.3 Construction of $(n, 2, \frac{n}{2} - 2)$ S-box

Our construction will be via symplectic matrices. Given any (n, r) quadratic S-box, it is clear from the above discussion that the symplectic matrices associated with the output component function defines the S-box. Thus to describe the construction, it is sufficient to define these symplectic matrices and use Lemmas 29 and 30 to prove the correctness of the construction.

In this section, we describe the construction of $(n, 2)$ S-boxes. Hence it is sufficient to define two symplectic matrices. We proceed to do this as follows. For each even $n \geq 6$, we define two sequences of $n \times n$ matrices and show that these matrices are the symplectic matrices required in the construction. For the rest of this chapter, we will use the following notation.

- For each $n \geq 1$, define v_n to be a string of length n which is the alternating sequence of 0's and 1's starting with a 0. For example, $v_4 = 0101$ and $v_5 = 01010$. Define $w_n = 1\overline{v_{n-1}}$.
- For each even $n \geq 2$, define u_n as $u_n = \underbrace{1 \ldots 1}_{(n/2)} \underbrace{0 \ldots 0}_{(n/2)}$. For odd $n \geq 3$, define $x_n = 1\overline{u_{n-1}}$.

Define $M_4 = [0010, 0010, 1101, 0010]^T$ and $N_4 = [0101, 1011, 0101, 1110]^T$. Further, for even $n > 4$ define

$$\begin{aligned} M_n &= \begin{bmatrix} 0 & v_{n-2} & 0 \\ v_{n-2}^T & M_{n-2} & v_{n-2}^T \\ 0 & v_{n-2} & 0 \end{bmatrix}, \quad F_n = \begin{bmatrix} 0 & v_{n-2} & 0 \\ v_{n-2}^T & M_{n-2} & u_{n-2}^T \\ 0 & u_{n-2} & 0 \end{bmatrix}; \\ N_n &= \begin{bmatrix} 0 & \overline{v_{n-2}} & 1 \\ \overline{v_{n-2}^T} & N_{n-2} & \overline{v_{n-2}^T} \\ 1 & \overline{v_{n-2}} & 0 \end{bmatrix}, \quad G_n = \begin{bmatrix} 0 & \overline{v_{n-2}} & 1 \\ \overline{v_{n-2}^T} & N_{n-2} & \overline{u_{n-2}^T} \\ 1 & \overline{u_{n-2}} & 0 \end{bmatrix}. \end{aligned} \tag{5.1}$$

The following result is easy to prove by induction on even $n \geq 6$.

Lemma 31 *F_n, G_n and $F_n \oplus G_n$ are symplectic matrices, where F_n and G_n are defined by equation 5.1.*

The matrices F_n and G_n are our required symplectic matrices which define the two output component functions of the required $(n, 2)$ S-box. In particular, we have the following result.

Theorem 32 *Let $n \geq 6$ be an even integer. The S-box $f : \mathbb{F}_2^n \to \mathbb{F}_2^2$ defined by $f(x) = (f_{F_n}(x), f_{G_n}(x))$ is a perfect nonlinear S-box satisfying $SAC(\frac{n}{2} - 2)$.*

We now turn to the proof of correctness of Theorem 32. The proof is in two parts – in the first part we prove the statement about SAC and in the second part we prove the statement about nonlinearity.

Lemma 33 *The S-box f defined in Theorem 32 satisfy $SAC(\frac{n}{2} - 2)$.*

Proof : Let r_j denote the j-th row of M_n. We make the following claim which can be routinely proved by induction on even $n \geq 4$.

$$\left.\begin{array}{lllll} \mathsf{wt}(r_j) & \geq & \frac{n}{2} - 1 & \text{if } 1 \leq j \leq \frac{n}{2} & \text{and } j \text{ is odd;} \\ \mathsf{wt}(r_j) & \geq & \frac{n}{2} - 2 & \text{if } 1 \leq j \leq \frac{n}{2} & \text{and } j \text{ is even;} \\ \mathsf{wt}(r_j) & \geq & \frac{n}{2} & \text{if } \frac{n}{2} + 1 \leq j \leq n & \text{and } j \text{ is odd;} \\ \mathsf{wt}(r_j) & \geq & \frac{n}{2} - 1 & \text{if } \frac{n}{2} + 1 \leq j \leq n & \text{and } j \text{ is even.} \end{array}\right\} \quad (5.2)$$

We will use the notation r'_j for j-th row which is obtained by dropping first and last column of M_n. Let s_j denote the jth row of F_n. We now have several cases.

Case 1 : $1 \leq j \leq \frac{n}{2}$ and j odd: There are two subcases.
Subcase 1(a) : $j = 1$. In this case $\mathsf{wt}(s_j) = \mathsf{wt}(v_{n-2}) = \frac{n-2}{2} = \frac{n}{2} - 1$.
Subcase 1(b) : $j > 1$. In this case $\mathsf{wt}(s_j) = 1 + 1 + \mathsf{wt}(r'_j) \geq 2 + \frac{n-2}{2} - 2 = \frac{n}{2} - 1$.

Case 2 : $1 \leq j \leq \frac{n}{2}$ and j even: In this case $\mathsf{wt}(s_j) = 1 + \mathsf{wt}(r'_j) \geq 1 + \frac{n-2}{2} - 1 = \frac{n}{2} - 1$.

Case 3 : $\frac{n}{2} + 1 \leq j \leq n$ and j odd: In this case $\mathsf{wt}(s_j) = 1 + \mathsf{wt}(r'_j) \geq 1 + \frac{n-2}{2} - 1 = \frac{n}{2} - 1$.

Case 4 : $\frac{n}{2} + 1 \leq j \leq n$ and j even: There are two subcases.
Subcase 4(a) : $j < n$. In this case $\mathsf{wt}(s_j) = \mathsf{wt}(r'_j) \geq \frac{n-2}{2} = \frac{n}{2} - 1$.
Subcase 4(b) : $j = n$. In this case $\mathsf{wt}(s_j) = \mathsf{wt}(u_{n-2}) = \frac{n-2}{2} = \frac{n}{2} - 1$.

This proves that the weight of each row of F_n is at least $(n/2) - 1$ and hence the corresponding Boolean function satisfies SAC$((n/2) - 2)$. By a similar argument the Boolean function associated with G_n also satisfies SAC$((n/2) - 2)$. Also note

$$F_n \oplus G_n = \begin{bmatrix} 0 & J_{n-2} & 1 \\ J_{n-2}^T & M_{n-2} \oplus N_{n-2} & J_{n-2}^T \\ 1 & J_{n-2} & 0 \end{bmatrix},$$

where J_n is all 1 vector. From this it is simple to verify by induction that $F_n \oplus G_n$ satisfies SAC$(\frac{n}{2} - 2)$. Now using Lemma 29 we obtain the required result. ∎

We next turn to the nonlinearity of the S-box defined in Theorem 32.

Lemma **34** *For even $n \geq 6$, the rank of F_n is n.*

Proof : First we prove that the rank of M_n is $n-2$. It is easy to check that the rank of M_4 is 2. Assume that the rank of M_{n-2} is $n-4$. It is clear that 1-st column and n-th column of M_n are identical. Likewise 1-st column and $n-2$-th column of M_{n-2} are identical. Consider the matrix

$$M'_n = \begin{bmatrix} 0 & v_{n-2} \\ v_{n-2}^T & M_{n-2} \end{bmatrix}.$$

From the definition of v_n, we have that the first bit of v_{n-2} is 0 and $(n-2)$-th bit is 1. So v_{n-2} is linearly independent of rows of M_{n-2}. So rank of M'_n is at least $n-4+1 = n-3$. But M'_n is symplectic matrix and hence its rank must be even (see [73, page 436]). So the rank of M'_n (and hence M_n) is $n-2$.

Now we turn to the rank of F_n. As M'_n has rank $n-2$, the rank of F_n is at least $n-2$. It is simple to verify by induction that $\frac{n}{2}$-th column and $(\frac{n}{2}+2)$-th column of M_n are identical. From definition, the $\frac{n}{2}$-th bit of $0u_{n-2}0$ is 1 and the $(\frac{n}{2}+2)$-th bit is 0. Hence the last row $0u_{n-2}0$ of F_n is linearly independent of the previous $(n-1)$ rows. Thus the rank of F_n is at least $n-2+1 = n-1$. But F_n is a binary symplectic matrix and hence its rank must be even. Hence the rank of F_n is n. ∎

Lemma **35** *For even $n \geq 6$, the rank of $F_n \oplus G_n$ is n.*

Proof : Note $M_4 \oplus N_4 = [0111, 1001, 1000, 1100]^T$ and hence the rank of $M_4 \oplus N_4$ is 4. Assume that the rank of $M_{n-2} \oplus N_{n-2}$ is $n-2$. Note

$$F_n \oplus G_n = M_n \oplus N_n = \begin{bmatrix} 0 & J_{n-2} & 1 \\ J_{n-2}^T & M_{n-2} \oplus N_{n-2} & J_{n-2}^T \\ 1 & J_{n-2} & 0 \end{bmatrix},$$

where J_n is the all 1 vector. The row $1J_{n-2}0$ is linearly independent of rows of matrix $J_{n-2}^T(M_{n-2} \oplus N_{n-2})J_{n-2}^T$. So rank of

$$\begin{bmatrix} J_{n-2}^T & M_{n-2} \oplus N_{n-2} & J_{n-2}^T \\ 1 & J_{n-2} & 0 \end{bmatrix}$$

is at least $n-2+1 = n-1$ and hence the rank of $F_n \oplus G_n$ is at least $n-1$. Again since $F_n \oplus G_n$ is a symplectic matrix its rank must be even. Hence its rank is n. ∎

We define $T_5 = [01010, 10101, 01011, 10101, 01110]^T$,

$$T_n = \begin{bmatrix} 0 & \overline{v_{n-2}} & 0 \\ \overline{v_{n-2}^T} & T_{n-2} & w_{n-2}^T \\ 0 & w_{n-2} & 0 \end{bmatrix} \text{ for odd } n > 5 \text{ and}$$
$$H_n = \begin{bmatrix} T_{n-1} & x_{n-1}^T \\ x_{n-1} & 0 \end{bmatrix} \text{ for even } n \geq 6. \tag{5.3}$$

First we prove the following result.

Lemma 36 *$G_n = H_n$ for all even $n \geq 6$.*

Proof : We first prove the following statement by induction on n.

$$T_n = \begin{bmatrix} 0 & \overline{v_{n-1}} \\ \overline{v_{n-1}^T} & N_{n-1} \end{bmatrix} \text{ for odd } n \geq 5 \text{ and}$$
$$N_n = \begin{bmatrix} T_{n-1} & w_{n-1}^T \\ w_{n-1} & 0 \end{bmatrix} \text{ for even } n \geq 6. \tag{5.4}$$

It is easy to verify that $\mathbf{T_5} = \begin{Bmatrix} 0 & \overline{v_4} \\ \overline{v_4^T} & \mathbf{N_4} \end{Bmatrix}$ and $\mathbf{N_6} = \begin{Bmatrix} T_5 & w_5^T \\ w_5 & \mathbf{0} \end{Bmatrix}$. **Assume that (5.4)** holds for $(n-1)$. By definition and using $\overline{v_{n-1}} = \overline{v_{n-2}}0$ we have that for odd $n \geq 7$, $\begin{bmatrix} 0 & \overline{v_{n-1}} \\ \overline{v_{n-1}^T} & N_{n-1} \end{bmatrix} = \begin{bmatrix} 0 & \overline{v_{n-2}} & 0 \\ \overline{v_{n-2}^T} & T_{n-2} & w_{n-2}^T \\ 0 & w_{n-2} & 0 \end{bmatrix} = T_n$. Similarly, by definition and using $1\overline{v_{n-2}} = \overline{w_{n-1}}0$ we have that for even $n \geq 8$, $\begin{bmatrix} T_{n-1} & w_{n-1}^T \\ w_{n-1}^T & 0 \end{bmatrix} = \begin{bmatrix} 0 & \overline{v_{n-2}} & 1 \\ \overline{v_{n-2}^T} & N_{n-2} & \overline{v_{n-2}^T} \\ 1 & \overline{v_{n-2}} & 0 \end{bmatrix} = N_n$. This completes the proof of (5.4). Now to prove $G_n = H_n$ it is sufficient to show $T_{n-1} = \begin{bmatrix} 0 & \overline{v_{n-2}} \\ \overline{v_{n-2}^T} & N_{n-2} \end{bmatrix}$ and $x_{n-1}0 = 1\overline{u_{n-2}}0$. The first statement follows from (5.4) and the second statement follows from the definition of x_n. ∎

Lemma 37 *For odd $n \geq 5$, the following statements hold for T_n.*

1. *The first column of T_{n-2} is v_{n-2}^T and the second column is $\overline{v_{n-2}^T}$.*
2. *The $\lfloor \frac{n}{2} \rfloor$-th column and $(\lfloor \frac{n}{2} \rfloor + 2)$-th column of T_n are identical.*

3. *The rank of* T_n *is* $(n-1)$.

Proof : All three statements are proved using induction on odd $n \geq 5$. We only describe the proof for the third statement. For $n = 5$ it is easy to verify that the rank of T_5 is 4. Assume that the rank of T_{n-2} is $n-3$. Consider the matrix $A_n = \begin{bmatrix} \overline{v_{n-2}^T} & T_{n-2} \\ 0 & w_{n-2} \end{bmatrix}$. By the first statement of the lemma, the first and third columns of the matrix $[\overline{v_{n-2}^T} \;\; T_{n-2}]$ are identical. At the same time the first and third bits of the vector $0w_{n-2}$ are 0 and 1 respectively. So the last row of A_n is linearly independent of other rows. Hence the rank of A_n is $n-3+1 = n-2$. Consequently, T_n has rank at least $n-2$. Again since T_n is a symplectic matrix, its rank must be even and hence must be $n-1$. ∎

Now we are in a position to prove that G_n is of full rank.

Lemma 38 *The rank of* G_n *is* n.

Proof : Consider $G_n = H_n = \begin{bmatrix} T_{n-1} & x_{n-1}^T \\ x_{n-1} & 0 \end{bmatrix}$. Since the rank of T_{n-1} is $(n-2)$ the rank of H_n is at least $n-2$. Again from Lemma 37, the $\lfloor \frac{n-1}{2} \rfloor$-th column and the $(\lfloor \frac{n-1}{2} \rfloor + 2)$-th column of T_{n-1} are identical. But the $\lfloor \frac{n-1}{2} \rfloor$-th and the $(\lfloor \frac{n-1}{2} \rfloor + 2)$-th bits of x_{n-1} are 0 and 1 respectively. Hence x_{n-1} is linearly independent of T_{n-1}. Thus the rank of G_n is at least $n-2+1 = n-1$. Again since G_n is a symplectic matrix its rank must be even and hence its rank is n. ∎

Thus we have the following result which completes the proof of Theorem 32.

Lemma 39 *The S-box* f *defined in Theorem 32 is a perfect nonlinear S-box.*

Proof : Using Lemmas 34, 35 and 38, we know that F_n, G_n and $F_n \oplus G_n$ have full rank. Hence the Boolean functions f_{F_n}, f_{G_n} and $f_{F_n} \oplus f_{G_n} = f_{F_n \oplus G_n}$ are bent. Thus the function f defined in Theorem 32 is a perfect nonlinear function. ∎

5.4 Relation With One Factorization of a Complete Graph

A one-factor of a graph G is a one-regular spanning subgraph of G. A one-factorization of G is a partition of the edges of G into one-factors.

Let K_n be the complete graph with n vertices. For even $n \geq 2$, it is well known that K_n can be decomposed into $(n-1)$ edge disjoint, one-factors [10]. One such decomposition of K_n is described as follows. For even n and $1 \leq i \leq n-1$, define

$$\begin{aligned}\mathcal{F}_i^n &= \{(n,i)\} \cup \{((n-2-j+i) \bmod (n-1)+1, (i+j-1) \bmod (n-1)+1) : \\ &\quad 1 \leq j \leq \frac{n}{2}-1\}\end{aligned}$$

The collection $\mathcal{T}_n = \{\mathcal{F}_1^n, \ldots, \mathcal{F}_{n-1}^n\}$ is a one factorization of K_n where the vertices are labeled by the integers $1, \ldots, n$. When n is clear from the context we will write $\mathcal{F}_i$ instead of $\mathcal{F}_i^n$. The elements of $\mathcal{T}_8$ (i.e. a one factorization of K_8) are given below.

$$\begin{aligned}\mathcal{F}_1 &= \{(8,1),(7,2),(6,3),(5,4)\} & \mathcal{F}_2 &= \{(8,2),(1,3),(7,4),(6,5)\} \\ \mathcal{F}_3 &= \{(8,3),(2,4),(1,5),(7,6)\} & \mathcal{F}_4 &= \{(8,4),(3,5),(2,6),(1,7)\} \\ \mathcal{F}_5 &= \{(8,5),(4,6),(3,7),(2,1)\} & \mathcal{F}_6 &= \{(8,6),(5,7),(4,1),(3,2)\} \\ \mathcal{F}_7 &= \{(8,7),(6,1),(5,2),(4,3)\}\end{aligned}$$

In [68], one factorization of K_n was used as a tool for construction of S-boxes satisfying SAC. We point out the connection of the construction of Section 5.3 to the one factorization of K_n. This connection will be developed in later sections to obtain other constructions of perfect nonlinear S-boxes satisfying higher order SAC.

Suppose $\mathcal{S} \subseteq \mathcal{T}_n$. We use $\mathcal{S}$ to define a symplectic matrix $B_{\mathcal{S}}$ in the following manner: For $1 \leq k, l \leq n$, the entry $B_{\mathcal{S}}[k,l] = 1$ if and only if either (k,l) or (l,k) is in $\mathcal{F}_i^n$ for some $\mathcal{F}_i^n \in \mathcal{S}$.

Theorem 40 *Let $n \geq 4$ be an even integer, $\mathcal{S}_1 = \{\mathcal{F}_2, \ldots, \mathcal{F}_{\frac{n}{2}}\}$ and $\mathcal{S}_2 = \mathcal{T} \setminus \mathcal{S}_1$. Let $B_{\mathcal{S}_1}$ and $B_{\mathcal{S}_2}$ be the symplectic matrices associated with $\mathcal{S}_1$ and $\mathcal{S}_2$ respectively. Then*

1. *F_n is obtained from $B_{\mathcal{S}_1}$ by changing the zeros in positions $(\frac{n}{2}+1, \frac{n}{2})$ and $(\frac{n}{2}, \frac{n}{2}+1)$ to ones.*

2. *G_n is obtained from $B_{\mathcal{S}_2}$ by changing the zeros in positions $(\frac{n}{2}+1, \frac{n}{2}+2)$ and $(\frac{n}{2}+2, \frac{n}{2}+1)$ to ones.*

Theorem 40 shows the relationship between one factorization and two output S-boxes of Section 5.3. This can be generalized to more than two output S-boxes. In fact, the earlier work of [68] provides such a generalization. However, there is one major difficulty with the generalization. It becomes very difficult to ensure that the resulting S-box is a perfect nonlinear S-box. Thus while the generalization of [68] ensures the SAC property, it results

in functions with quite weak nonlinearity. On the other hand, our motivation is to obtain *perfect nonlinear* S-boxes satisfying higher order SAC. The rest of the chapter is devoted to identifying other perfect nonlinear S-boxes satisfying higher order SAC.

5.4.1 Improvements for Two Output S-Boxes

We know from [68] that for an $(n, 2, k)$-SAC function, $k \leq \lfloor \frac{2(n-1)}{3} \rfloor - 1$. Thus the construction in Section 5.3 is suboptimal with respect to the SAC property. (However, it is optimal with respect to nonlinearity).

Here we provide some examples of two output S-boxes with higher order SAC. All these examples were obtained using experimental method. The constructions are based on the relationship between the symplectic matrices and one factorization described above. These examples are summarized in Table 1. The interpretation of the entries in Table 1 is as follows. Each row describes a construction for the particular value of n. The second column describes two subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ of $\mathcal{T}_n$. Let $B_{\mathcal{S}_1}$ and $B_{\mathcal{S}_2}$ be the symplectic matrices associated with these two sets. We set $B_1 = B_{\mathcal{S}_1}$ and B_2 is $B_{\mathcal{S}_2}$ with the following modification: If (k, l) is in the third column, then $B_{\mathcal{S}_2}[k, l]$ and $B_{\mathcal{S}_2}[l, k]$ are changed from 0 to 1. The desired S-box $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^2$ is given by $f(x) = (f_{B_1}(x), f_{B_2}(x))$. *Each of these S-boxes is a perfect nonlinear S-box.* The fourth column provides the order of SAC that is achieved by the corresponding S-box. The fifth column provides the maximum order of SAC that can be achieved by an $(n, 2)$ S-box. *In the situation where this maximum is equal to the achieved order of SAC, the construction provides optimal trade-off among the following parameters : nonlinearity, order of SAC, number of input variables, number of output variables. None of these parameters can be improved without changing some other parameter.*

5.5 Construction of $(n, 3, k)$ S-boxes

We describe constructions of $(n, 3, k)$ perfect nonlinear S-boxes. These constructions were also obtained by experimental methods. Some of the constructions seem to have a general pattern, though it has not been possible to prove a general result. There are several cases in the construction though the description of the constructions in all the cases is similar. We first identify three subsets $\mathcal{S}_1, \mathcal{S}_2$ and $\mathcal{S}_3$ of $\mathcal{T}_n$. These three subsets define three symplectic matrices $B_{\mathcal{S}_1}, B_{\mathcal{S}_2}$ and $B_{\mathcal{S}_3}$. These matrices are then modified by changing a number of zeros to ones to obtain three other symplectic matrices B_1, B_2 and B_3. The positions where the

n	Description	Modification	k	max k
8	$\mathcal{S}_1 = \{\mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4, \mathcal{F}_5, \mathcal{F}_6\}$	–	3	3
	$\mathcal{S}_2 = \{\mathcal{F}_1, \mathcal{F}_5, \mathcal{F}_6, \mathcal{F}_7\}$	(5,6)		
10	$\mathcal{S}_1 = \{\mathcal{F}_1, \mathcal{F}_3, \mathcal{F}_5, \mathcal{F}_8, \mathcal{F}_9\}$	–	4	5
	$\mathcal{S}_2 = \{\mathcal{F}_4, \mathcal{F}_6, \mathcal{F}_7, \mathcal{F}_8, \mathcal{F}_9\}$	(6,9)		
12	$\mathcal{S}_1 = \{\mathcal{F}_1, \mathcal{F}_4, \mathcal{F}_6, \mathcal{F}_7, \mathcal{F}_8, \mathcal{F}_9, \mathcal{F}_2\}$	(2,7)	6	6
	$\mathcal{S}_2 = \{\mathcal{F}_1, \mathcal{F}_3, \mathcal{F}_5, \mathcal{F}_8, \mathcal{F}_9, \mathcal{F}_{10}, \mathcal{F}_{11}\}$	–		
14	$\mathcal{S}_1 = \{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4, \mathcal{F}_5, \mathcal{F}_{10}, \mathcal{F}_{11}, \mathcal{F}_{12}, \mathcal{F}_{13}\}$	–	7	7
	$\mathcal{S}_2 = \{\mathcal{F}_6, \mathcal{F}_7, \mathcal{F}_8, \mathcal{F}_9, \mathcal{F}_{10}, \mathcal{F}_{11}, \mathcal{F}_{12}, \mathcal{F}_{13}\}$	(8,9)		
16	$\mathcal{S}_1 = \{\mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4, \mathcal{F}_5, \mathcal{F}_6, \mathcal{F}_7, \mathcal{F}_8, \mathcal{F}_9, \mathcal{F}_{10}\}$	–	8	9
	$\mathcal{S}_2 = \{\mathcal{F}_1, \mathcal{F}_8, \mathcal{F}_9, \mathcal{F}_{10}, \mathcal{F}_{11}, \mathcal{F}_{12}, \mathcal{F}_{13}, \mathcal{F}_{14}, \mathcal{F}_{15}\}$	(3,9)		

Table 1 : Improved and Optimal Constructions of Two Output S-boxes.

changes are to be made are given by the third column. If (k, l) is in the third column, then $B_{\mathcal{S}_j}[k, l]$ and $B_{\mathcal{S}_j}[l, k]$ $(1 \leq j \leq 3)$ are changed from 0 to 1. The required $(n, 3)$ S-box $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^3$ is obtained from these three matrices in the following manner: $f(x) = (f_{B_1}(x), f_{B_2}(x), f_{B_3}(x))$. There are three cases.

1. Table 2 describes several cases of constructions for $n \equiv 0 \bmod 8$. For $n > 8$, there is a general heuristic which provides the required construction. For $n = 8$, a special construction is required.

2. Table 3 describes constructions for $n \equiv 4 \bmod 8$. These constructions have a general pattern.

3. Table 4 describes several constructions for $n \equiv 2 \bmod 4$. There does not appear to be any general pattern for these constructions.

The constructions for $n = 10, 22$ provide optimal trade-off between the following parameters: numbers of input and output variables, nonlinearity and the order of SAC. Further, for $n = 12, 16, 20$ and 24 the achieved value of k is only one less than the upper bound on k.

5.6 Maximally Nonlinear Functions

The constructions described so far hold when the number of input bits n is even. In case n is odd, there do not exist any perfect nonlinear S-boxes. In this section, we describe a simple

n	Description	Modification	k	$\max k$
8	$\mathcal{S}_1 = \{\mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4\}$	(4,5)	1	2
	$\mathcal{S}_2 = \{\mathcal{F}_4, \mathcal{F}_5, \mathcal{F}_6\}$	–		
	$\mathcal{S}_3 = \{\mathcal{F}_1, \mathcal{F}_4, \mathcal{F}_7\}$	(4,7)		
16,	$\mathcal{S}_1 = \{\mathcal{F}_2, \mathcal{F}_3, \ldots, \mathcal{F}_{\frac{n}{2}}\}$	$(\frac{n}{2}, \frac{n}{2}+1)$	$\frac{n}{2}-2$	min(
24,	$\mathcal{S}_2 = \{\mathcal{F}_{\frac{n}{4}+2}, \ldots, \mathcal{F}_{\frac{3n}{4}}\}$	$(\frac{n}{2}, \frac{n}{4}+1), (\frac{n}{2}, \frac{3n}{4}+1)$		$\lfloor \frac{4(n-1)}{7} \rfloor - 1$,
32	$\mathcal{S}_3 = \{\mathcal{F}_1, \mathcal{F}_{\frac{n}{4}+2}, \ldots, \mathcal{F}_{\frac{n}{2}}, \mathcal{F}_{\frac{3n}{4}+1}, \ldots, \mathcal{F}_{n-1}\}$	$(\frac{n}{2}, \frac{3n}{4})$		$2\lfloor \frac{2n}{7} \rfloor - 1)$

Table 2 : Constructions for $n \equiv 0 \bmod 8$.

n	Description	Modification	k	$\max k$
12,20,28	$\mathcal{S}_1 = \{\mathcal{F}_2, \mathcal{F}_3, \ldots, \mathcal{F}_{\frac{n}{2}}\}$	$(\frac{n}{2}, \frac{n}{2}+1)$	$\frac{n}{2}-2$	min(
	$\mathcal{S}_2 = \{\mathcal{F}_{\frac{n}{4}+2}, \ldots, \mathcal{F}_{\frac{3n}{4}}\}$	$(\frac{n}{2}, \frac{n}{4}+2)$		$\lfloor \frac{4(n-1)}{7} \rfloor - 1$,
	$\mathcal{S}_3 = \{\mathcal{F}_1, \mathcal{F}_{\frac{n}{4}+2}, \ldots, \mathcal{F}_{\frac{n}{2}}, \mathcal{F}_{\frac{3n}{4}+1}, \ldots, \mathcal{F}_{n-1}\}$	$(\frac{n}{2}, \frac{3n}{4}+1)$		$2\lfloor \frac{2n}{7} \rfloor - 1)$

Table 3 : Constructions for $n \equiv 4 \bmod 8$.

modification of the previously described constructions which provide maximally nonlinear S-boxes.

Theorem 41 *Let f be a $(2r, m, k)$ perfect nonlinear quadratic S-box where the symplectic matrices associated with the component functions are $B_1, \ldots, B_m$. For $1 \leq i \leq m$, let B_i' be obtained from B_i by deleting the first row and column. Then the S-box $f' : \mathbb{F}_2^{2r-1} \to \mathbb{F}_2^m$ defined by $f'(x) = (f_{B_1'}(x), \ldots, f_{B_m'}(x))$ is a $(2r-1, m, k-1)$ maximally nonlinear quadratic S-box.*

Proof : There are two things to be proved – the nonlinearity and the order of SAC. Since f is a perfect nonlinear S-box, each nonzero linear combination of the B_i's has full rank (see Lemma 30). Dropping one row and one column decreases the rank by two for symplectic matrices. Hence the rank of any nonzero linear combination of the B_i''s is $2r-2$ and the nonlinearity of the corresponding Boolean function is $2^{2r-2} - 2^{r-1}$. Now using Lemma 30 we have that f' is a maximally nonlinear S-box.

Further, since f satisfies SAC(k), the number of ones in any nonzero linear combination of the B_i's is at least $k+1$. Dropping one row and one column decreases the number of ones in any row (or column) by at most one. Again using Lemma 29, it follows that the S-box f' satisfies SAC($k-1$). ∎

n	Description	Modification	k	max k
10	$\mathcal{S}_1 = \{\mathcal{F}_2, \mathcal{F}_4, \mathcal{F}_8, \mathcal{F}_9\}$	(6,9)	3	3
	$\mathcal{S}_2 = \{\mathcal{F}_1, \mathcal{F}_3, \mathcal{F}_5, \mathcal{F}_8, \mathcal{F}_9\}$	–		
	$\mathcal{S}_3 = \{\mathcal{F}_6, \mathcal{F}_7, \mathcal{F}_8, \mathcal{F}_9\}$	(5,6)		
14	$\mathcal{S}_1 = \{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_4, \mathcal{F}_5, \mathcal{F}_{12}, \mathcal{F}_{13}\}$	(1,6)	4	6
	$\mathcal{S}_2 = \{\mathcal{F}_1, \mathcal{F}_3, \mathcal{F}_7, \mathcal{F}_{10}, \mathcal{F}_{11}, \mathcal{F}_{12}, \mathcal{F}_{13}\}$	–		
	$\mathcal{S}_3 = \{\mathcal{F}_1, \mathcal{F}_6, \mathcal{F}_8, \mathcal{F}_9, \mathcal{F}_{12}, \mathcal{F}_{13}\}$	(1,9)		
18	$\mathcal{S}_1 = \{\mathcal{F}_2, \mathcal{F}_4, \mathcal{F}_{10}, \mathcal{F}_{11}, \mathcal{F}_{12}, \mathcal{F}_{13}, \mathcal{F}_{14}, \mathcal{F}_{15}\}$	(5,10)	7	8
	$\mathcal{S}_2 = \{\mathcal{F}_1, \mathcal{F}_3, \mathcal{F}_5, \mathcal{F}_{12}, \mathcal{F}_{13}, \mathcal{F}_{14}, \mathcal{F}_{15}, \mathcal{F}_{16}, \mathcal{F}_{17}\}$	–		
	$\mathcal{S}_3 = \{\mathcal{F}_6, \mathcal{F}_7, \mathcal{F}_8, \mathcal{F}_9, \mathcal{F}_{12}, \mathcal{F}_{13}, \mathcal{F}_{14}, \mathcal{F}_{15}\}$	(9,10)		
22	$\mathcal{S}_1 = \{\mathcal{F}_1, \mathcal{F}_4, \mathcal{F}_5, \mathcal{F}_6, \mathcal{F}_7, \mathcal{F}_{14}, \mathcal{F}_{15}, \mathcal{F}_{16}, \mathcal{F}_{17}, \mathcal{F}_{18}\}$	(1,5)	9	9
	$\mathcal{S}_2 = \{\mathcal{F}_1, \mathcal{F}_3, \mathcal{F}_9, \mathcal{F}_{14}, \mathcal{F}_{15}, \mathcal{F}_{16}, \mathcal{F}_{17}, \mathcal{F}_{18}, \mathcal{F}_{19}, \mathcal{F}_{20}, \mathcal{F}_{21}\}$	–		
	$\mathcal{S}_3 = \{\mathcal{F}_8, \mathcal{F}_{10}, \mathcal{F}_{11}, \mathcal{F}_{12}, \mathcal{F}_{13}, \mathcal{F}_{14}, \mathcal{F}_{15}, \mathcal{F}_{16}, \mathcal{F}_{17}, \mathcal{F}_{18}\}$	(1,16)		

Table 4 : Constructions for $n \equiv 2 \bmod 4$.

5.7 Improving Algebraic degree

The constructions described in the previous sections provide quadratic functions. In this section, we describe a method of improving the degree of the constructed functions with a small trade-off in the nonlinearity and the SAC property. We first need to relax the notion of SAC. (See [67] for the notion of almost PC(l) of order k functions.)

Definition 42 *An n-variable Boolean function f is said to be (ϵ, k)-SAC if the following property holds: Let g be an $(n-i)$-variable Boolean function obtained from f by fixing $i \leq k$ input variables to constants. Then $\left|\frac{wt(g(x)\oplus g(x\oplus\alpha))}{2^{n-i}} - \frac{1}{2}\right| \leq \epsilon$ for any α of weight 1. An (n,m) S-box is said to be (n,m,ϵ,k)-SAC if every nonzero linear combination of the component functions is an (ϵ,k)-SAC function.*

The next result shows how to convert an (n,m,k) S-box into an (n,m,ϵ,k) S-box for a small ϵ and with a small change in nonlinearity.

Theorem 43 *Let $f = (f_1, \ldots, f_m)$ be an (n,m,k) S-box where the degree of any f_i is less than $(n-1)$. Then it is possible to construct an (n,m,ϵ,k) S-box g with algebraic degree $n-1$, $\epsilon = \frac{m+1}{2^{n-k-1}}$ and $\mathsf{nl}(g) \geq \mathsf{nl}(f) - (m+1)$ if m is odd; $\mathsf{nl}(g) \geq \mathsf{nl}(f) - m$ if m is even.*

Proof : We construct an (n,m) S-box g with component functions $g_1, g_2, \ldots, g_m$ in the following manner. For $1 \leq i \leq m$, define $g_i(x_1, \cdots, x_n) = f_i(x_1, \cdots, x_n) \oplus x_1 \ldots x_{i-1}x_{i+1} \ldots x_n$.

By construction, the algebraic degree of any g_i is $n-1$. Further, the degree $(n-1)$ terms in the g_i's are distinct. Hence any nonzero linear combination of the g_i's also has degree $(n-1)$. Thus the degree of g is $(n-1)$.

We now prove the nonlinearity. The term $x_1 \ldots x_{i-1}x_{i+1} \ldots x_n$ which is XORed to f_i to obtain g_i changes exactly two output values of f_i. Thus $\mathsf{nl}(g_i) = \mathsf{nl}(f_i) - 2$. Further, the inputs for which the outputs are changed are the all one vector and the vector with a zero only in the ith position. Thus if h (resp. h') is a linear combination of i of the g_i's (resp. f_i's), then h and h' differ in at most $(i+1)$ positions. Since $1 \leq i \leq m$, we have $\mathsf{nl}(g) \geq \mathsf{nl}(f) - (m+1)$ when m is odd. Since the nonlinearity of a function of n variables and degree $< n$ is always even we have $\mathsf{nl}(g) \geq \mathsf{nl}(f) - m$ when m is even.

Now suppose that $h_1(x)$ (resp. $h_1'(x)$) is obtained from $h(x)$ (resp. $h'(x)$) by fixing at most j $(1 \leq j \leq k)$ input bits to constant values. Since $h(x)$ and $h'(x)$ differ in exactly $(i+1)$ positions, it follows that $h_1(x)$ and $h_1'(x)$ differ in at most $(i+1)$ positions. Further, since $h_1(x)$ and $h_1'(x)$ differ in at most $(i+1)$ positions, so does $h_1(x \oplus \alpha)$ and $h_1'(x \oplus \alpha)$. Let $\mu(x) = h(x) \oplus h(x \oplus \alpha)$ and $\mu'(x) = h'(x) \oplus h'(x \oplus \alpha)$. Then it follows that $\mu(x)$ and $\mu'(x)$ differ in at most $2(i+1)$ positions. Since f satisfies SAC(k), it follows that $\mu'(x)$ is balanced and has weight 2^{n-j-1}. Also since $1 \leq i \leq m$ and $1 \leq j \leq k$, we obtain $\left|\frac{wt(\mu(x))}{2^{n-j}} - \frac{1}{2}\right| = \left|\frac{wt(\mu(x))}{2^{n-j}} - \frac{wt(\mu'(x))}{2^{n-j}}\right| = \left|\frac{wt(\mu(x)|-wt(\mu'(x))}{2^{n-j}}\right| \leq \frac{2(i+1)}{2^{n-j}} \leq \frac{m+1}{2^{n-k-1}}$. This completes the proof. ∎

Table 5 provides some examples to illustrate Theorem 43. The interpretation of Table 5

n	degree	$m=2$	$m=3$
8	7	(3, 0.1875, 118)	(1, 0.0625, 116)
9	8	(3, 0.0938, 238)	(2, 0.0625, 236)
10	9	(4, 0.0938, 494)	(3, 0.0625, 492)
11	10	(5, 0.0938, 990)	(3, 0.0313, 988)
12	11	(6, 0.0938, 2014)	(4, 0.0313, 2012)

Table 5 : Values of k, ϵ and nonlinearity for 2 and 3 output S-boxes for different values of n (see Theorem 43).

is as follows. Each entry is of the form (k, ϵ, x), where k is the order of SAC, ϵ is defined in Theorem 43 and x is the nonlinearity of the modified function. (When m is even, the value of nonlinearity is one more than the lower bound given in Theorem 43.) Note that in each case the algebraic degree is $n-1$. The drop in nonlinearity is very small; for example for $n = 8$, the lower bound from Theorem 43 is 117 while the maximum possible nonlinearity

s 120. Similarly, in each of the above cases, the value of ϵ is small. Hence the deviation from perfect nonlinearity and the (perfect) SAC property is small. On the other hand, the degree increases to the maximum possible. Thus such S-boxes are amply suited for use in the design of practical block cipher algorithms.

5.8 A Concrete Example

Here we provide a step by step procedure to construct a $(8, 2, 3)$ perfect nonlinear quadratic S-box. Then we modify it to get higher degree with some sacrifice in nonlinearity and SAC property. We take (see Table 1) $\mathcal{S}_1 = \{\mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4\, \mathcal{F}_5, \mathcal{F}_6\}$ and $\mathcal{S}_2 = \{\mathcal{F}_1, \mathcal{F}_5, \mathcal{F}_6, \mathcal{F}_7\}$. We construct symplectic matrix $B_{\mathcal{S}_1}$ in the following manner: the entry $B_{\mathcal{S}_1}[k, l] = 1$ if and only if either (k, l) or (l, k) is in $\mathcal{F}_i$ for some $\mathcal{F}_i \in \mathcal{S}_1$. Similarly we construct symplectic matrix $B_{\mathcal{S}_2}$.

$$B_{\mathcal{S}_1} = \begin{bmatrix} 01111010 \\ 10110101 \\ 11001011 \\ 11000111 \\ 10100111 \\ 01011011 \\ 10111100 \\ 01111100 \end{bmatrix}, \qquad B_{\mathcal{S}_2} = \begin{bmatrix} 01010101 \\ 10101010 \\ 01010110 \\ 10101100 \\ 01010x11 \\ 1011x001 \\ 01101001 \\ 10001110 \end{bmatrix}.$$

Initially $x = 0$ in matrix $B_{\mathcal{S}_2}$. We modify $B_{\mathcal{S}_2}$ by changing the value of x from 0 to 1. Symplectic matrix $B_{\mathcal{S}_1}$ can uniquely be written as $Q_1 \oplus Q_1^T$, where Q_1 is an upper triangular matrix with zero diagonal. Similarly for modified $B_{\mathcal{S}_2}$ we have Q_2. Where

$$Q_1 = \begin{bmatrix} 01111010 \\ 00110101 \\ 00001011 \\ 00000111 \\ 00000111 \\ 00000011 \\ 00000000 \\ 00000000 \end{bmatrix}, \qquad Q_2 = \begin{bmatrix} 01010101 \\ 00101010 \\ 00010110 \\ 00001100 \\ 00000111 \\ 00000001 \\ 00000001 \\ 00000000 \end{bmatrix}.$$

Associated with B_{S_1} and modified B_{S_2} we have two quadratic component functions $f_1 = xQ_1x^T$ and $f_2 = xQ_2x^T$. So

$$\begin{aligned} f_1(x_1,\ldots,x_8) &= x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_1x_5 \oplus x_1x_7 \oplus x_2x_3 \oplus x_2x_4 \oplus x_2x_6 \oplus x_2x_8 \oplus x_3x_5 \\ &\oplus x_3x_7 \oplus x_3x_8 \oplus x_4x_6 \oplus x_4x_7 \oplus x_4x_8 \oplus x_5x_6 \oplus x_5x_7 \oplus x_5x_8 \oplus x_6x_7 \oplus x_6x_8. \end{aligned}$$

Similarly

$$\begin{aligned} f_2(x_1,\ldots,x_8) &= x_1x_2 \oplus x_1x_4 \oplus x_1x_6 \oplus x_1x_8 \oplus x_2x_3 \oplus x_2x_5 \oplus x_2x_7 \oplus x_3x_4 \oplus x_3x_6 \oplus x_3x_7 \\ &\oplus x_4x_5 \oplus x_4x_6 \oplus x_5x_6 \oplus x_5x_7 \oplus x_5x_8 \oplus x_6x_8 \oplus x_7x_8. \end{aligned}$$

The S-box $f = (f_1, f_2)$ is an $(8, 2, 3)$ perfect nonlinear S-box with algebraic degree two. To increase the algebraic degree we define S-box $g = (g_1, g_2)$ by $g_1(x_1,\ldots,x_8) = f_1(x_1,\ldots,x_8) \oplus x_2x_3\ldots x_8$ and $g_2(x_1,\ldots,x_8) = f_1(x_1,\ldots,x_8) \oplus x_1x_3x_4\ldots x_8$. The 8-input, 2-output S-box g has degree 7, nonlinearity 118 and $\epsilon = 0.1875$ (see Table 5).

Now we give an example of $(8, 3, 1)$ perfect nonlinear S-box (see Table 2). Let f_1, f_2, f_3 be the component functions, then associated symplectic matrices B_1, B_2, B_3 are given below .

$$\begin{bmatrix} 00101010 \\ 00010101 \\ 10001001 \\ 01001011 \\ 10110100 \\ 01001010 \\ 10010100 \\ 01110000 \end{bmatrix}, \quad \begin{bmatrix} 01010010 \\ 10100100 \\ 01001010 \\ 10000101 \\ 00100011 \\ 01010001 \\ 10101000 \\ 00011100 \end{bmatrix}, \quad \begin{bmatrix} 00000111 \\ 00001110 \\ 00011100 \\ 00101011 \\ 01110000 \\ 11100000 \\ 11010001 \\ 10010010 \end{bmatrix}.$$

As in the previous example we can compute the algebraic normal form of the component functions and can also increase degree sacrificing a little bit of nonlinearity and SAC property.

5.9 Conclusion

In this chapter, we have considered the problem of constructing perfect nonlinear S-boxes satisfying higher order SAC. Previous work in this area [68] also provided constructions of

S-boxes satisfying higher order SAC. However, the nonlinearity obtained was lower. To the best of our knowledge, we provide the first examples of S-boxes satisfying higher order SAC *and* perfect nonlinearity. Some of the constructed S-boxes also achieve optimal trade-off between the numbers of input and output variables, nonlinearity and the order of SAC. Our construction uses bilinear forms and symplectic matrices and yields quadratic functions. We show that the degree can be significantly improved by a small sacrifice in nonlinearity and the SAC property. This yields S-boxes which have possible applications in the design of block ciphers. To increase degree we use a method by which we get an S-box which is very close to the original S-box. One research problem is to investigate other methods for increasing the degree. Lastly, we would like to remark that more research is necessary to generalize our construction using symplectic matrices to more than 3 outputs and also to obtain direct constructions of higher degree S-boxes which satisfy higher order SAC and perfect nonlinearity.

Chapter 6

Construction of High Degree Resilient S-Boxes With Improved Nonlinearity

Resilient S-boxes were introduced by Chor et al [27] and Bennett et al [2]. The study of other important cryptographic properties of resilient S-boxes such as high nonlinearity and algebraic degree have been performed in [24, 63, 70, 89, 122]. In [24], Cheon used an $[n-d-1, m, t+1]$ code to construct an n-input, m-output, t-resilient S-box with degree $d > m$ and nonlinearity $(2^{n-1} - 2^{n-d-1}\lfloor\sqrt{2^n}\rfloor + 2^{n-d-2})$. *Till date this is the only known method which can be used to obtain degree $d > m$.* The construction of Cheon uses the algebraic structure of linearized polynomials and the nonlinearity calculation is based on the Hasse-Weil bound for higher genus curves.

6.1 Introduction

In this chapter we provide two new constructions. We first describe a *simple* construction of nonlinear resilient S-boxes with high algebraic degree. Given an $[n-d-1, m, t]$ code we construct an n-input, m-output, t-resilient S-box with degree $d > m$ and nonlinearity $2^{n-1} - 2^{n-\lceil\frac{d+1}{2}\rceil} - (m+1)2^{n-d-1}$. Further we *prove* that for any fixed values of the parameters n, m, t and d with $d > m$, the nonlinearity obtained by our method is in all cases higher than the nonlinearity obtained by Cheon's method.

The second construction is a simple modification of the Zhang-Zheng method [122]. To get algebraic degree $d > m$, we start with an $[n, d+1, t+1]$ code. Then we apply Zhang-Zheng construction to obtain a nonlinear S-box. Finally we drop $d+1-m$ output columns to obtain

an (n, m, t)-resilient S-box (see Section 6.5). This simple modification is powerful enough to improve upon the best known construction with algebraic degree greater than m [24]. Our contribution is to apply the Griesmer bound for linear error correcting codes to *prove* that the modified Zhang-Zheng construction is superior to the best known construction [24].

6.2 Coding Theory Results

We will use some standard coding theory results and terminology all of which can be found in [73]. An $[n, k, d]$ binary linear code is a subset of $\mathbb{F}_2^n$ which is a vector space of dimension k over $\mathbb{F}_2$ having minimum distance d. We here mention the Griesmer bound (see [73, page 546]). For an $[n, k, d]$ linear code let $N(k, d)$ = length of the shortest binary linear code of dimension k and minimum distance d.
The Griesmer bound states (see [73, page 547])

$$N(k, d) \geq \sum_{i=0}^{k-1} \left\lceil \frac{d}{2^i} \right\rceil. \tag{6.1}$$

We say that the parameters n, k, d satisfy the Griesmer bound with equality if $n = \sum_{i=0}^{k-1} \left\lceil \frac{d}{2^i} \right\rceil$. There is a general construction (see [73, page 550]) which gives large class of codes meeting the Griesmer bound with equality. Given d and k , define $s = \lceil \frac{d}{2^{k-1}} \rceil$ and $d = s2^{k-1} - \sum_{i=1}^{p} 2^{u_i - 1}$ where $k > u_1 > \cdots > u_p \geq 1$. Given d and k, there is an $[n = s(2^k - 1) - \sum_{i=1}^{p}(2^{u_i} - 1), k, d]$ code meeting the Griesmer bound with equality if $\sum_{i=1}^{min(s+1,p)} u_i \leq sk$ (see [73, page 552]). This condition is satisfied for most values of d and k.

6.3 Construction of (n, m, t)-Resilient S-box with Degree $> m$.

We will be interested in (n, m) S-boxes with maximum possible nonlinearity. The following result is well known(see for example [63]).

Theorem 44 *Let C be an $[n, m, t+1]$ binary linear code. Then we can construct an (n, m, t)-resilient function.*

The following result is a slight generalization of Theorem 44.

Lemma 45 *Let $f : \{0,1\}^n \to \{0,1\}^m$ be an S-box and $f_1, f_2, \cdots, f_m$ are the component functions. Let $\mathcal{C}$ be a $[p, m, t+1]$ binary linear code. We can construct a t-resilient S-box $h : \{0,1\}^{n+p} \to \{0,1\}^m$ with component functions $h_1, h_2, \cdots, h_m$, where $\mathsf{nl}(h) = 2^p \mathsf{nl}(f)$ and algebraic degree of $h(x)$ is same as algebraic degree of $f(x)$.*

Proof : A binary linear code $[p, m, t+1]$ is a vector space of dimension m over $\mathbb{F}_2$. Let $\{C_1, C_2, \cdots, C_m\}$ be a basis where $C_i = (c_{i_1}, c_{i_2}, \cdots, c_{i_p}) \in \mathbb{F}_2^p$. We construct the S-box $h : \{0,1\}^{n+p} \to \{0,1\}^m$ as follows. For $1 \leq i \leq m$, we define,

$$h_i(x_1, \cdots, x_n, x_{n+1}, \cdots, x_{n+p}) = f_i(x_1, \cdots, x_n) \oplus \langle C_i, (x_{n+1}, x_{n+2}, \cdots, x_{n+p}) \rangle.$$

Let h' be any nonzero linear combination of the component functions $h_1, \cdots, h_m$. So h' can be written as $h' = d_1 h_1 \oplus \cdots \oplus d_m h_m$ for some nonzero vector $(d_1, \cdots, d_m) \in \mathbb{F}_2^m$. Hence

$$\begin{aligned} h' &= d_1 f_1 \oplus \cdots \oplus d_m f_m \oplus \langle d_1 C_1 \oplus \cdots \oplus d_m C_m, (x_{n+1}, \cdots, x_{n+p}) \rangle \\ &= d_1 f_1 \oplus \cdots \oplus d_m f_m \oplus \langle C', (x_{n+1}, \cdots, x_{n+p}) \rangle \end{aligned}$$

where $C' = d_1 C_1 \oplus \cdots \oplus d_m C_m$. We have weight of vector $C' \geq t+1$ since $\mathcal{C}$ is a $[p, m, t+1]$ linear code. Hence h' is t-resilient and so $h(x_1, \cdots, x_n, x_{n+1}, \cdots, x_{n+p})$ is t-resilient. As we are adding p new variables, $\mathsf{nl}(h) = 2^p \mathsf{nl}(f)$. Clearly $\deg(h) = \deg(f)$. ∎

The next result provides a simple method to construct a $(d+1, m)$ S-box with degree d and very high nonlinearity.

Theorem 46 *It is possible to construct a $(d+1, m)$ S-box with degree $d > m$ and nonlinearity $\mathsf{nl}(h) \geq 2^d - 2^{\lfloor \frac{d+1}{2} \rfloor} - (m+1)$.*

Proof : Let f be a $(d+1, d+1)$ maximally nonlinear S-box whose component functions are $f_1, f_2, \cdots, f_{d+1}$. We construct a $(d+1, m)$ S-box h with component functions $h_1, h_2, \cdots, h_m$ in the following manner. For $1 \leq i \leq m$, define $\mu_i(x_1, \ldots, x_{d+1}) = x_1 \ldots x_{i-1} x_{i+1} \ldots x_{d+1}$, and

$$h_i(x_1, \cdots, x_{d+1}) = f_i(x_1, \cdots, x_{d+1}) \oplus \mu_i(x_1, \ldots, x_{d+1}).$$

By construction algebraic degree of S-box $h : \{0,1\}^{d+1} \to \{0,1\}^m$ is d. It is known that $\mathsf{nl}(f) \geq 2^d - 2^{\lfloor \frac{d+1}{2} \rfloor}$ [36]. We show that $\mathsf{nl}(h) \geq \mathsf{nl}(f) - (m+1)$. Let ϵ_i be the identity vector which has a one at the ith position and zero elsewhere. Let $\mathbf{1} = (1, \ldots, 1)$. From the definition of μ_i it is clear that $\mathsf{Sup}(\mu_i) = \{(x_1, \ldots, x_{d+1}) : \mu_i(x_1, \ldots, x_{d+1}) = 1\} = \{\mathbf{1}, \bar{\epsilon_i}\}$. A nonzero linear combination h' of the component functions $h_1, \ldots, h_m$ can be written as

$$h' = f_{i_1} \oplus \cdots \oplus f_{i_r} \oplus \mu_{i_1} \oplus \ldots \oplus \mu_{i_r}, \text{ for some } \{i_1, i_2, \cdots, i_r\} \subseteq \{1, 2, \cdots, m\}.$$

We have $\cup_{j=1}^{r} \mathsf{Sup}(\mu_{i_j}) = \{\mathbf{1}, \epsilon_{i_1}, \ldots, \epsilon_{i_r}\}$ and so the weight of the function $\mu_{i_1} \oplus \ldots \oplus \mu_{i_r}$ is at most $r+1$. Hence $\mathsf{nl}(h') \geq \mathsf{nl}(f) - (r+1)$. Since $r \leq m$, it follows that $\mathsf{nl}(h) \geq \mathsf{nl}(f) - (m+1)$ which gives us the required result. ■

Now we are ready to describe our construction method.

Construction-I .

1. Input: Parameters n, m, t and d with $d > m$.
2. Output: An (n, m, t, d)−resilient function.

Procedure

1. Construct a $(d+1, m)$ S-box using Theorem 46.
2. Let $\mathcal{C}$ be a $[n-d-1, m, t+1]$ code. If no such code exists, then stop. The function cannot be constructed using this method.
3. Apply Lemma 45 on h and $\mathcal{C}$ to construct the required S-box g.

Theorem 47 *If an* $[n-d-1, m, t+1]$ *code exists, then Construction-I constructs an* (n, m, t, d) *S-box* g *with nonlinearity* $2^{n-1} - 2^{n-\lceil \frac{d+1}{2} \rceil} - (m+1)2^{n-d-1}$.

Proof : By Theorem 46, $\mathsf{nl}(h) = 2^d - 2^{\lfloor \frac{d+1}{2} \rfloor} - (m+1)$ and $\deg(h) = d$. By Lemma 45, g is t-resilient, $\mathsf{nl}(g) = 2^{n-d-1}\mathsf{nl}(h) = 2^{n-1} - 2^{n-\lceil \frac{d+1}{2} \rceil} - (m+1)2^{n-d-1}$ and $\deg(g) = \deg(h) = d$.■

6.4 Comparison

In [24, Theorem 5], Cheon proved the following result.

Theorem 48 *For any non-negative integer* d, *if there exists* $[n-d-1, m, t+1]$ *linear code then there exists a* (n, m, t)*-resilient function with degree* d *and nonlinearity* $(2^{n-1} - 2^{n-d-1}\lfloor \sqrt{2^n} \rfloor + 2^{n-d-2})$.

The nonlinearity calculation in the above theorem is based on Hasse-Weil bound for higher genus curves. Till date, this is the only construction which provides (n, m, t) nonlinear resilient S-boxes with degree greater than m. In the next Theorem we prove that nonlinearity obtained by Construction-I is higher than nonlinearity obtained by Cheon's construction.

Theorem 49 *Let* f *be an* (n, m, t, d)*-resilient function* f *with* $d > m$ *and nonlinearity* n_1 *constructed by Cheon's method. Then it is possible to construct an* (n, m, t, d)*-resilient function* g *with nonlinearity* n_2 *using Construction-I. Further* $n_2 > n_1$.

Proof : As (n, m, t)-resilient function f is constructed by Cheon's method, there exists an $[n-d-1, m, t+1]$ code. Construction-I can be applied to obtain an (n, m, t)-resilient function g with degree d and nonlinearity $\mathsf{nl}(g) = n_2 = 2^{n-1} - 2^{n-\lceil\frac{d+1}{2}\rceil} - (m+1)2^{n-d-1}$. It remains to show that $n_2 > n_1$, which we show now. Recall $n_1 = 2^{n-1} - 2^{n-d-1}\lfloor\sqrt{2^n}\rfloor + 2^{n-d-2}$. Hence $n_2 - n_1 = -2^{n-\lceil\frac{d+1}{2}\rceil} + 2^{n-d-1}\lfloor\sqrt{2^n}\rfloor - 2^{n-d-2} - (m+1)2^{n-d-1}$. Thus we have $n_2 > n_1$ if $2^{n-d-1}\lfloor 2^{\frac{n}{2}}\rfloor > 2^{n-\lceil\frac{d+1}{2}\rceil} + 2^{n-d-2} + (m+1)2^{n-d-1}$. The last condition holds if and only if $\lfloor 2^{\frac{n}{2}}\rfloor > \frac{1}{2} + (m+1) + 2^{d+1-\lceil\frac{d+1}{2}\rceil}$. So $n_2 > n_1$ if $2^{\frac{n}{2}} - 1 > \frac{3}{2} + m + 2^{\frac{d+1}{2}}$, i.e., if $2^{\frac{n}{2}} > \frac{5}{2} + m + 2^{\frac{d+1}{2}}$. It is simple to check that the last condition holds for $n \geq d+2$ for all positive integers n and $d > m$. The maximum possible degree of an S-box is $(n-1)$, so $d \leq n-1$. For $d = n-1$ the required code is $[0, m, t+1]$. Clearly such a code does not exist. Hence $n_2 > n_1$ holds for all d. ∎

Remark : We note that Cheon's method does not provide any nonlinearity for $d \leq \frac{n-1}{2}$, whereas Construction-I provides positive nonlinearity for $d > 2$.

6.5 Second Construction of (n, m, t)-Resilient S-box with Degree $> m$.

In this section we modify an elegant construction by Zhang and Zheng [122] to obtain high degree nonlinear resilient S-boxes.

Modified Zhang-Zheng (MZZ) Construction.

1. Input: Number of output columns $= m$, degree $= d \geq m$ and resiliency $= t$.
2. Output: An $(n, m, t)-$resilient function with degree d and nonlinearity $2^{n-1} - 2^{n-\lceil\frac{d+1}{2}\rceil}$.

Procedure

1. Choose an $[n, d+1, t+1]$ code to obtain a linear $(n, d+1, t)$-resilient function f.
2. Define $g = G \circ f$, where $G : \{0,1\}^{d+1} \to \{0,1\}^{d+1}$ is a bijection and $\deg(G) = d$, $\mathsf{nl}(G) = 2^d - 2^{\lfloor\frac{d+1}{2}\rfloor}$. Then $\mathsf{nl}(g) \geq 2^{n-d-1}(2^d - 2^{\lfloor\frac{d+1}{2}\rfloor}) = 2^{n-1} - 2^{n-\lceil\frac{d+1}{2}\rceil}$ and $\deg(g) = d$.
3. Drop $(d+1-m)$ columns from the output of g to obtain an(n, m, t)-resilient function with degree d and nonlinearity $2^{n-1} - 2^{n-\lceil\frac{d+1}{2}\rceil}$.

In Step 2 above, we choose the function G to be the inverse function over $GF(2^{d+1})$ (with respect to a fixed irreducible polynomial). Then the nonlinearity of G is $2^d - 2^{\lfloor\frac{d+1}{2}\rfloor}$

and is given in [84]. There are other bijections by which we get the same value of $\mathsf{nl}(G)$ but $\deg(G) = d$ is achieved only for G obtained from the inverse map over $GF(2^{d+1})$ (see [21]). The fact that $g = G \circ f$ is t-resilient if f is t-resilient is given in a more general form in [13] and also appears in [122].

The modification to the Zhang-Zheng construction is really simple. If we want degree d, then we start with an $[n, d+1, t+1]$ code. Then we apply the main step of Zhang-Zheng construction to obtain a nonlinear S-box. Finally we drop $d+1-m$ output columns to obtain an (n, m, t)-resilient S-box. Though simple, this modification is powerful enough to improve upon the best known construction with high algebraic degree [24].

Theorem 50 *Let n, m, d, t be such that the following two conditions hold.*

1. *Either (a) $d < m$ or (b) $d \geq m \geq \log_2(t+1)$.*
2. *The parameters $n, d+1, t+1$ meet the Griesmer bound with equality. Then it is not possible to construct an (n, m, t)-resilient function f with degree d using Cheon [24] method.*

Proof : We have from Theorem 48, given any $[N, M, T+1]$ and a non negative integer D, the Cheon construction produces an $(N+D+1, M, T)$-resilient function with degree D. Thus if f is obtained by the Cheon construction we must have $n = N + D + 1$, $m = M$, $t = T$ and $d = D$.
This means that an $[n-d-1, m, t+1]$ code will be required by the Cheon construction. Since the parameters $n, d+1, t+1$ satisfies Griesmar bound with equality we have $n = \sum_{i=0}^{d} \lceil \frac{t+1}{2^i} \rceil$.
Claim : If (a) $d < m$ or (b) $d \geq m \geq \log_2(t+1)$ then $n - d - 1 < \sum_{i=0}^{m-1} \lceil \frac{t+1}{2^i} \rceil$.
Proof of the claim: Since $n = \sum_{i=0}^{d} \lceil \frac{t+1}{2^i} \rceil$ we have that $n - d - 1 < \sum_{i=0}^{m-1} \lceil \frac{t+1}{2^i} \rceil$ if and only if

$$\sum_{i=0}^{d} \left\lceil \frac{t+1}{2^i} \right\rceil - d - 1 < \sum_{i=0}^{m-1} \left\lceil \frac{t+1}{2^i} \right\rceil .$$

If $d < m$, then the last mentioned condition is trivially true. So suppose $d \geq m \geq \log_2(t+1)$. Then the above inequality holds if and only if

$$\sum_{i=m}^{d} \left\lceil \frac{t+1}{2^i} \right\rceil < d + 1.$$

Since $m \geq \log_2(t+1)$, $\sum_{i=m}^{d} \lceil \frac{t+1}{2^i} \rceil = d - m + 1 < d + 1$ for $m \geq 1$. This completes the proof of the claim.

Since $n-d-1 < \sum_{i=0}^{m-1} \lceil \frac{t+1}{2^i} \rceil$, the parameters $n-d-1, m, t+1$ violate the Griesmer bound and hence an $[n-d-1, m, t+1]$ code do not exist. Thus Cheon method cannot be used to construct the function f. ∎

The following result is a consequence of Theorem 50 and the MZZ construction.

Theorem 51 *Let n, m, d, t be such that the following two conditions hold.*

1. *Either (a) $d < m$ or (b) $d \geq m \geq \log_2(t+1)$.*
2. *An $[n, d+1, t+1]$ code meeting the Griesmer bound with equality exist. Then it is possible to construct an (n, m, t)-resilient function f with degree d by the MZZ method which cannot be constructed using Cheon [24] method.*

Remark: As mentioned in [73, page 550] there is a large class of codes which meet the Griesmer bound with equality. Further, the condition $d \geq m \geq \log_2(t+1)$ is quite weak. Hence there exists a large class of (n, m, t)-resilient functions which can be constructed using MZZ construction but cannot be constructed using Cheon [24] construction. See Section 6.6 for some concrete examples.

Given any $[N, M, T+1]$ and a non negative integer D, the Cheon construction produces an $(N+D+1, M, T)$-resilient function with degree D (see Theorem 48). *Nonlinearity in Cheon method is $(2^{N+D} - 2^N \lfloor \sqrt{2^{N+D+1}} \rfloor + 2^{n-1})$ which is positive if $D \geq N+1$ for $N \geq 2$. So for $D \leq N$, Cheon method does not provide any nonlinearity. Thus Cheon method may provide high algebraic degree but it does not provide good nonlinearity.* In fact, in the next theorem we prove that nonlinearity obtained by MZZ method is larger than nonlinearity obtained by Cheon method.

Theorem 52 *Let f be an (n, m, t)-resilient function f of degree d and nonlinearity n_1 constructed by Cheon method. Suppose there exists a linear $[n, d+1, t+1]$ code. Then it is possible to construct an (n, m, t)-resilient function g with degree d and nonlinearity n_2 using MZZ method . Further $n_2 \geq n_1$.*

Proof : Since $[n, d+1, t+1]$ code exists, the MZZ construction can be applied to obtain an (n, m, t)-resilient function g with degree d and nonlinearity $\mathsf{nl}(g) = n_2 = 2^{n-1} - 2^{n-\lfloor \frac{d+1}{2} \rfloor}$. It remains to show that $n_2 \geq n_1$, which we show now. Recall $n_1 = 2^{n-1} - 2^{n-d-1} \lfloor \sqrt{2^n} \rfloor + 2^{n-d-2}$. Hence

$$n_2 - n_1 \geq -2^{n-\frac{d+1}{2}} + 2^{n-d-1} \lfloor \sqrt{2^n} \rfloor - 2^{n-d-2}.$$

Thus we have $n_2 \geq n_1$ if $-2^{\frac{-(d+1)}{2}} + 2^{-(d+1)}\lfloor\sqrt{2^n}\rfloor - 2^{-(d+2)} \geq 0$. The last condition holds if and only if

$$\lfloor\sqrt{2^n}\rfloor \geq 2^{d+1}(\frac{1}{2^{\frac{d+1}{2}}} + \frac{1}{2^{d+2}}).$$

So $n_2 \geq n_1$ if $\sqrt{2^n} - 1 \geq 2^{\frac{d+1}{2}} + 2^{-1}$, i.e, if $2^{\frac{n}{2}} \geq 2^{\frac{d+1}{2}} + \frac{3}{2}$. Again the last condition hold for $1 \leq d \leq n-3$. Hence $n_2 \geq n_1$ for $1 \leq d \leq n-3$. The maximum possible degree of an S-box is $n-1$. For $d = n-1$ and $d = n-2$, Cheon construction requires $[0, m, t+1]$ and $[1, m, t+1]$ codes respectively. Clearly such codes do not exist. Hence $n_2 \geq n_1$ holds for all d. ∎

Lemma 53 *Let f be an (n, m, t)-resilient function f of degree $d \geq m$ constructed by Cheon method and $m \geq \log_2(t+1)$. Then the parameters $n, d+1, t+1$ satisfy the Griesmer bound.*

Proof : Since f has been obtained from Cheon method, there exists an $[n-d-1, m, t+1]$ code. Hence the parameters $n-d-1, m$ and $t+1$ satisfy the Griesmar bound. Since $n-d-1, m$ and $t+1$ satisfy the Griesmar bound we have

$$n - d - 1 \geq \sum_{i=0}^{m-1} \left\lceil \frac{t+1}{2^i} \right\rceil,$$

i.e, we have $n \geq d+1+\sum_{i=0}^{m-1}\lceil\frac{t+1}{2^i}\rceil$. As $m \geq \log_2(t+1)$ we have $\lceil\frac{t+1}{2^i}\rceil = 1$ for $i \geq m$. Hence

$$n \geq (d+1) - (d-m+1) + \sum_{i=m}^{d} \left\lceil \frac{t+1}{2^i} \right\rceil + \sum_{i=0}^{m-1} \left\lceil \frac{t+1}{2^i} \right\rceil.$$

This shows $n \geq m + \sum_{i=0}^{d}\lceil\frac{t+1}{2^i}\rceil$ and consequently $n \geq \sum_{i=0}^{d}\lceil\frac{t+1}{2^i}\rceil$. Thus the parameters $n, d+1, t+1$ satisfy the Griesmer bound. ∎

Remark: Since the parameters $n, d+1$ and $t+1$ satisfy the Griesmer bound, in most cases it is possible to obtain an $[n, d+1, t+1]$ code (see [73, page 550]) and apply Theorem 52. *In fact we do not know any case where a function can be constructed using the Cheon method but not by the MZZ method.* Theorems 51 and 52 *prove* the clear advantage of the MZZ method over the Cheon construction. Thus MZZ method is the currently known best method to construct $[n, m, t]$-resilient function with degree $d > m$.

6.6 Degree Comparison Based on MZZ Construction

We present examples to show the advantage of the MZZ method over the Cheon method. Cheon method cannot construct (n, m, t)-resilient function of degree $d \geq m \geq 2$ if the fol-

lowing two conditions hold.

(1)

t	1	2 to 3	4 to 7	8 to 15	16 to 31
m	$m \geq 1$	$m \geq 2$	$m \geq 3$	$m \geq 4$	$m \geq 5$

(2) The parameters $n, d+1, t+1$ satisfy Griesmer bound with equality.
We next present some examples of n, m, d and t satisfying condition (1) and (2) such that the MZZ method can be used to construct (n, m, t)-resilient function with degree d.

(a) $t = 1$, $2 \leq m \leq d$, $n = d + 2$. It is easy to check that a $[d+2, d+1, 2]$ code exists.

(b) $t = 2$, $2 \leq m \leq d$, $(n, d) = (6, 2), (7, 3), (8, 4), (9, 5), (10, 6), (11, 7)$. In each case an $[n, d+1, t+1]$ code exists.

(c) $t = 3$, $2 \leq m \leq d$, $(n, d) = (7, 2), (8, 3), (11, 6), (12, 7), (13, 8)$. In each case an $[n, d+1, t+1]$ code exists.
In (a) to (c) above an (n, m, t)-resilient function with degree d can be constructed using MZZ method, but cannot be constructed using Cheon method(see Theorem 51). Now we present some examples where both MZZ and Cheon method construct (n, m, t)-resilient function with degree d and compare their nonlinearity using Theorem 52. An (n, m, d, t) S-box is an (n, m, t)-resilient S-box with degree d.

Function	$(10, 3, 1, 5)$	$(18, 4, 2, 10)$	$(24, 5, 2, 15)$	$(24, 7, 3, 12)$	$(28, 6, 4, 14)$
Cheon [24, Theorem 5]	8	$2^{16} + 2^9$	$2^{23} - 2^{20} + 2^7$	2^{10}	2^{12}
MZZ	$2^9 - 2^7$	$2^{17} - 2^{12}$	$2^{23} - 2^{16}$	$2^{23} - 2^{17}$	$2^{27} - 2^{20}$

Table 1 : Comparison of nonlinearity obtained by MZZ Construction to that obtained by Cheon [24].

We see that in each case the nonlinearity obtained by the MZZ method is superior to that obtained by the Cheon method.

6.7 Conclusion

In this chapter we have presented two simple construction of nonlinear resilient S-boxes with algebraic degree greater than m. We *proved* that for any fixed values of the parameters n, m, t and d, the nonlinearity obtained by our constructions is superior to the nonlinearity obtained by the only previously known construction [24] which can provide degree $d > m$.

Chapter 7

Improved Construction of Nonlinear Resilient S-Boxes

Our construction for nonlinear resilient functions uses a sharpened version of the Maiorana-McFarland technique. The nonlinearity obtained by our construction is better than previously known construction methods.

7.1 Introduction

The current state-of-art in resilient S-box design can be classified into the following two approaches.

1. Construction of (n, m, t)-resilient functions with degree $d > m$ and high nonlinearity.
2. Construction of (n, m, t)-resilient functions with very high nonlinearity.

The first problem has already been discussed in Chapter 6. The second problem has been studied in [122, 70, 63, 89]. The currently best known results are obtained using the construction described in [89], though in certain cases, for small number of variables, the search technique of [63] yields better results.

In this chapter, we first prove that the correlation immunity of a resilient function is preserved under composition with an arbitrary Boolean function. This property is useful for possible application of resilient S-boxes in designing secure stream ciphers. Our contribution in this chapter consists of describing constructions for the second class of of problems.

As in the case of Boolean functions (see Section 2.1.1), if we fix the enumeration σ of the set $\mathbb{F}_2^n$, then an (n, m) S-box f is uniquely defined by a $2^n \times m$ matrix M_f, where $f(\sigma(i))$ is given by the ith row of M_f. Given a sequence of S-boxes $f_1, \cdots, f_k$; where f_i is an (n_i, m) S-box we define the *concatenation* of $f_1, \cdots, f_k$ to be the matrix

$$M = \begin{bmatrix} M_{f_1} \\ M_{f_2} \\ \vdots \\ M_{f_k} \end{bmatrix}.$$

If $2^{n_1} + \cdots + 2^{n_k} = 2^n$ for some n, then the matrix M uniquely defines an (n, m) S-box f. In this case we say f is the *concatenation* of $f_1, \cdots, f_k$.

7.2 A description of the Maiorana-McFarland Construction

Maiorana-McFarland construction essentially consists of obtaining a nonlinear resilient function by concatenating small affine functions and was introduced in [14]. Later work have helped in developing and sharpening the idea. Our discussion is based on [103].

We start by providing a brief description of the construction as applied to Boolean functions and then discuss the modifications required for applying to S-boxes. An n-variable Boolean function may be represented uniquely by a binary string of length 2^n. Thus to describe a Boolean function it is sufficient to describe its representation as a string of length 2^n. The construction consists of several ideas.

First idea : Let $f_1, \cdots, f_{2^{n-r}}$ be 2^{n-r} affine functions, where each f_i is represented by a string of length 2^r and is non-degenerate on at least $t+1$ variables. Consider the concatenation of the string representation of the functions $f_1, \cdots, f_{2^{n-r}}$. The resulting string is of length 2^n and hence represent an n-variable Boolean function f.

Second idea : Let $g(x_n, \ldots, x_{r+1})$ be a nonlinear function and let $h(x_r, \ldots, x_1)$ be a linear function which is non-degenerate on at least $t+1$ variables. Define

$$f(x_n, \ldots, x_1) = g(x_n, \ldots, x_{r+1}) \oplus h(x_r, \ldots, x_1).$$

Now consider the string representation of f. Suppose we fix the values of $x_n, \cdots, x_{r+1}$ to some constants $c_n, \cdots, c_{r+1}$. Then $f(c_n, \cdots, c_{r+1}, x_r, \cdots, x_1) = g(c_n, \cdots, c_{r+1}) \oplus h(x_r, \cdots, x_1)$.

Since $g(c_n, \cdots, c_{r+1})$ is a constant, the substring of f as $x_r, \cdots, x_1$ runs over 2^r possible values is equal to either the string representation of h or the string representation of $1 \oplus h$. Thus the entire string f is a concatenation of h and $1 \oplus h$.

Generalization : Let $R_1 + \cdots + R_k = 2^n$ where R_i is a multiple of 2^{r_i} and is of the form $R_i = M_i \times 2^{r_i} = (2^{s_{i,1}} + \cdots + 2^{s_{i,k_i}})2^{r_i} = 2^{s_{i,1}}2^{r_i} + \cdots + 2^{s_{i,k_i}}2^{r_i}$. Let $g_{i,j}$ be a nonlinear function of $s_{i,j}$ variables and l_i is a linear function of r_i variables disjoint from the previous $s_{i,j}$ variables. Let $h_{i,j} = g_{i,j} \oplus l_i$. By the second idea $h_{i,j}$ is a concatenation of l_i or $\overline{l_i}$. Consider concatenation of $h_{i,1}, \cdots, h_{i,k_i}$ and call it f_i. Each f_i may not represent a Boolean function since its string representation may not be a power of 2. Let f be concatenation of $f_1, \cdots, f_k$. This f is an n-variable Boolean function which is ultimately a concatenation of the l_is and the $(1 \oplus l_i)$s.

The Maiorana-McFarland technique is a well known method to construct nonlinear resilient functions. The idea is to use affine functions on small number of variables to construct nonlinear resilient functions on larger number of variables. We provide a construction to generate functions of the first type using a sharpened version of the Maiorana-McFarland method. For Boolean functions, the Maiorana-McFarland technique to construct resilient functions was introduced by Camion et al [14]. Nonlinearity calculation for the construction was first performed by Seberry, Zhang and Zheng [107]. This technique was later sharpened by Chee et al [23] and Sarkar-Maitra [103]. For S-boxes this technique has been used by [63] and [89], though [63] uses essentially a heuristic search technique. Here we develop and sharpen the technique of affine function concatenation to construct nonlinear resilient S-boxes. This leads to significant improvement in nonlinearity over that obtained in [89]. Thus we obtain better results than [89] which currently provides the best known nonlinearity results for most choices of input parameters n, m, t.

7.2.1 Some Recent Constructions

Here we summarize the previous construction results.

1. Zhang and Zheng [122]: This is the first paper to provide a general construction of nonlinear resilient S-boxes. The main idea of the construction was also stated in [13]. The result proved is the following [122, Corollary 6]. If there exists a linear (n, m, t)-resilient function, then there exists a nonlinear (n, m, t)-resilient function with algebraic degree $(m-1)$ and nonlinearity $\geq (2^{n-1} - 2^{n-\frac{m}{2}})$.

2. Kurosawa, Satoh and Yamamoto [70, Theorem 18]: For any even l such that $l \geq 2m$,

if there exists an $(n-l, m, t)$-resilient function , then there exists an (n, m, t)-resilient function, whose nonlinearity is at least $2^{n-1} - 2^{n-\frac{l}{2}-1}$.

3. Johansson and Pasalic [63]: They use a linear error correcting code to build a matrix A of small affine functions. Resiliency and nonlinearity is ensured by using non-intersecting codes along with the matrix A. The actual non-intersecting codes used were obtained by a heuristic search technique. It becomes difficult to carry out this search technique for $n > 12$.

4. Pasalic and Maitra [89]: They use the matrix A of the previous method (3) along with highly nonlinear functions for their construction. The nonlinearity obtained is higher than the previous methods, except in certain cases, where the search technique of (3) yields better results.

7.3 A Construction to Obtain High Nonlinearity

In this section we concentrate on obtaining (n, m, t)-resilient S-boxes with high nonlinearity only. We present a construction method which improves the nonlinearity obtainable by the previously known methods. We start by mentioning the following result which is restatement of Lemma 7 in [63].

Theorem 54 *Let C be a $[u, m, t+1]$ code. Then it is possible to construct $(2^m-1)\times m$ matrix D with entries from C, such that, $\{c_1 D_{i,1} \oplus \cdots \oplus c_m D_{i,m} : 1 \leq i \leq 2^m - 1\} = C \setminus \{(0, \cdots, 0)\}$ for each nonzero vector $(c_1, \cdots, c_m) \in \mathbb{F}_2^m$.*

Let D be the matrix in Theorem 54. For $(1 \leq i \leq 2^m - 1)$ and $(1 \leq j \leq m)$ define a u-variable linear function $L_{i,j}(x_1, \cdots, x_u) \triangleq \langle D_{i,j}, (x_1, \cdots, x_u)\rangle$. Given the code C we define a $(2^m - 1) \times m$ matrix $L(C)$ whose entries are u-variable linear functions by defining the i, j th entry of $L(C)$ to be $L_{i,j}(x_1, \cdots, x_u)$. We have the following result which follows directly from Theorem 54.

Proposition 55 *Let $c \in \mathbb{F}_2^m$ be a nonzero row vector. Then all the entries of the column vector $L(C)c^T$ are distinct.*

For positive integers k, l with $k \leq l$, we define $L(C, k, l)$ to be the submatrix of $L(C)$ consisting of the rows k to l. Thus $L(C, 1, 2^m-1) = L(C)$. Let $G(y_1, \cdots, y_p)$ be a (p, m) S-box

whose component functions are $G_1, \cdots, G_m$. We define $G \oplus L(C,k,l)$ to be an $(l-k+1) \times m$ matrix whose i,j th entry is

$$G_j(y_1, \cdots, y_p) \oplus L_{k+i-1,j}(x_1, \cdots, x_u)$$

for $1 \leq i \leq l-k+1$ and $1 \leq j \leq m$. If $l-k+1 = 2^r$ for some r then $G \oplus L(C,k,l)$ defines an S-box $F : \{0,1\}^{r+p+u} \rightarrow \{0,1\}^m$ in the following manner.

$$F_j(z_1, \cdots, z_r, y_1, \cdots, y_p, x_1, \cdots, x_u) = G_j(y_1, \cdots, y_p) \oplus L_{k+i-1,j}(x_1, \cdots, x_u)$$

where $1 \leq j \leq m$, $1 \leq i \leq 2^r$, $F_1, \cdots, F_m$ are the component functions of F and $z_1 \cdots z_r$ is the binary representation of $i-1$. By $F = G \oplus L(C,k,l)$ we will mean the above representation of the S-box F. *Note that the function F is t-resilient, since each $L_{i,j}(x_1, \cdots, x_u)$ is non-degenerate on at least $(t+1)$ variables and hence t-resilient.*

In the matrix $M = G(y_1, \cdots, y_p) \oplus L(C,k,l)$ we say that the row $L_{i,*}$ of $L(C)$ is repeated 2^p times. Let $G(y_1, \cdots, y_p)$ and $H(y_1, \cdots, y_q)$ be (p,m) and (q,m) S-boxes respectively and $M_1 = G \oplus L(C,k,l)$, $M_2 = H \oplus L(C,k,l)$. Then we say that the row $L_{i,*}$ of $L(C)$, $(k \leq i \leq l)$ is repeated a total of $2^p + 2^q$ times in the matrix $[M_1 \ M_2]^T$.

Proposition 55 has also been used by [89] in the construction of resilient S-boxes. However we improve upon the construction of [89] by utilizing the following two ideas.

1. We use all the $2^m - 1$ rows of the matrix $L(C)$. In contrast, [89] uses at most 2^{m-1} rows of $L(C)$.

2. We allow a row of $L(C)$ to be repeated 2^{r_1} or $2^{r_1} + 2^{r_2}$ or $2^{r_1} + 2^{r_2} + 2^{r_3}$ times as required. On the other hand, the number of times a row of $L(C)$ can be repeated in [89] is of the form 2^r.

It turns out that a proper utilization of the above two techniques result in significant improvement in nonlinearity. We will require (r,m) S-boxes with very high nonlinearity. For this we propose to use the best known results which we summarize in the following definition.

Definition 56 *Let G be an (r,m) S-box satisfying the following.*

1. *If $r < m$, G is a constant S-Box.*

2. *If $m \leq r < 2m$, G is a maximally nonlinear S-Box [36].*

3. *If $r \geq 2m$ and r is even, G is a perfect nonlinear S-Box [84].*

4. *If $r \geq 2m$ and r is odd, G is concatenation of two perfect nonlinear S-Boxes(see Section 2.2).*

Then we say that G is a PROPER S-box.

The following result summarizes the best known results on the nonlinearity of PROPER S-boxes.

Proposition 57 *Let G be an (r, m) PROPER S-box. Then*

1. *If $r < m$, $\mathsf{nl}(G) = 0$.*
2. *If $m \leq r < 2m$, then $\mathsf{nl}(G) = 2^{r-1} - 2^{\frac{r-1}{2}}$ if r is odd and $\mathsf{nl}(G) \geq 2^{r-1} - 2^{\frac{r}{2}}$ if r is even.*
3. *If $r \geq 2m$, then $\mathsf{nl}(G) = 2^{r-1} - 2^{\frac{r}{2}-1}$ if r is even and $\mathsf{nl}(G) = 2^{r-1} - 2^{\frac{r-1}{2}}$ if r is odd.*

Now we are in a position to describe a new construction of resilient S-boxes. The construction has two parts. In Part-A, we compute the number of rows of $L(C)$ to be used and the number of times each row is to be repeated. The output of Part-A is a *list* of the form $list = \langle (n_1, R_1), (n_2, R_2), \cdots, (n_k, R_k) \rangle$ which signifies that n_i rows of $L(C)$ are to be repeated R_i times each. Part-A also computes a variable called effect which determines the nonlinearity of the S-box (see Theorem 58). In Part-B of the construction, we choose PROPER functions based on *list* and describe the actual construction of the S-box.

Construction-I.

1. Input: Positive integers (n, m) and t.
2. Output: A nonlinear (n, m, t)-resilient S-box F.

Part-A

1. Obtain minimum u such that $[u, m, t+1]$ code C exists.
2. Case: $n - u \leq 0$, then function cannot be constructed using this method.
Hence stop.
3. Case: $n - u \geq 0$
   (a) $0 \leq n - u < m$; $list = \langle (2^{n-u}, 1) \rangle$ and effect $= 1$.
   (b) $m \leq n - u < 2m - 1$; $list = \langle (2^{m-1}, 2^{n-u-m+1}) \rangle$
   and effect$= 2^{n-u-m+1}$.

(c) $n-u=2m-1$; $list=\langle(2^{m-1},2^m)\rangle$ and effect= $2^{\lfloor\frac{m}{2}\rfloor+1}$.

(d) $2m\leq n-u<3m$.

(i) $n-u=2m+2e$; m even; $0\leq e<\frac{m}{2}$;
$list=\langle(1,2^{m+2e+1}),(2^m-2,2^{m+2e})\rangle$ and effect= $2^{e+1+\frac{m}{2}}$.

(ii) $n-u=2m+2e+1$; m even; $0\leq e\leq\frac{m}{2}-1$;

- $0\leq e\leq\frac{m}{2}-2$;
  $list=\langle(2,2^{m+2e+1}+2^{2e+1}+2^{2e}),(2^m-3,2^{m+2e+1}+2^{2e+1})\rangle$
  and effect= $2^{2e+1}+2^{2e}+2^{e+1+\frac{m}{2}}$.
- $e=\frac{m}{2}-1$; $list=\langle(2^{m-1},2^{2m})\rangle$ and effect= 2^m.

(iii) $n-u=2m+2e+1$; m odd; $0\leq e\leq\lfloor\frac{m}{2}\rfloor-1$;
$list=\langle(1,2^{m+2e+2}),(2^m-2,2^{m+2e+1})\rangle$ and effect= $2^{\frac{m+2e+3}{2}}$.

(iv) $n-u=2m+2e$; m odd; $0\leq e<\lfloor\frac{m}{2}\rfloor$;
$list=\langle(2^m-2,2^{m+2e}+2^{2e+1}),(1,2^{2e+2})\rangle$
and effect= $2^{2e+1}+2^{\frac{m+2e+1}{2}}$.

(v) $n-u=3m-1$; m odd;
$list=\langle(2^{m-1},2^{2m})\rangle$ and effect= 2^m.

(e) $n-u\geq 3m$.

(i) $n-u=3m+2e+1$; $e\geq 0$;
$list=\langle(2^{m-1},2^{2m+2e+2})\rangle$ and effect= 2^{m+e+1}.

(ii) $n-u=3m+2e$; (m even; $e\geq\frac{m}{2}$) or (m odd; $0\leq e<\lfloor\frac{m}{2}\rfloor$);
$list=\langle(2,2^{2m+2e}+2^{m+2e}+2^{m+2e-1}),(2^m-3,2^{2m+2e}+2^{m+2e})\rangle$
and effect= $2^{m+e}+2^{e+1+\frac{m}{2}}$.

(iii) $n-u=3m+2e$; m even; $0\leq e<\frac{m}{2}$
$list=\langle(2^m-2,2^{2m+2e}+2^{m+2e+1}),(1,2^{m+2e+2})\rangle$
and effect= $2^{m+e}+2^{e+1+\frac{m}{2}}$.

(iv) $n-u=3m+2e$; m odd; $e\geq\lfloor\frac{m}{2}\rfloor$
$list=\langle(2^m-2,2^{2m+2e}+2^{m+2e+1}),(1,2^{m+2e+2})\rangle$
and effect= $2^{m+e}+2^{e+\frac{m+1}{2}}$.

Part-B

1. If $list=\langle(2^s,2^r)\rangle$;

- Obtain $L(C,1,2^s)$ from $L(C)$ by selecting first 2^s rows of $L(C)$.
- Let G be an (r,m) PROPER S-box.
- Define $F=G\oplus L(C,1,2^s)$.
- This covers cases 3.(a),(b),(c),(d)(ii) second item, (d)(v)

and e(i) of Part-A.

Case: 3(d)(i) of Part-A

- Let G_1 and G_2 be $(m+2e+1, m)$ and $(m+2e, m)$ PROPER S-boxes.
- Define $F_1 = G_1 \oplus L(C, 1, 1)$, $F_2 = G_2 \oplus L(C, 2, 2^m - 1)$.
- F is the concatenation of F_1 and F_2 .

Case: 3(d)(ii) first item of Part-A and $e = 0$

- Let G_1 and G_2 be $(m+1, m)$ and $(1, m)$ PROPER S-boxes.
- Define $F_1 = G_1 \oplus L(C)$, $F_2 = G_2 \oplus L(C)$, $F_3 = L(C, 1, 2)$.
- F is the concatenation of F_1, F_2 and F_3 .

Case: 3(d)(ii) first item of Part-A and $e \neq 0$

- Let G_1, G_2 and G_3 be $(m+2e+1, m)$, $(2e+1, m)$ and $(2e, m)$ PROPER S-boxes.
- Define $F_1 = G_1 \oplus L(C)$, $F_2 = G_2 \oplus L(C)$, $F_3 = G_3 \oplus L(C, 1, 2)$.
- F is the concatenation of F_1, F_2 and F_3 .

Case: 3(d)(iii) of Part-A

- Let G_1 and G_2 be $(m+2e+2, m)$ and $(m+2e+1, m)$ PROPER S-boxes.
- Define $F_1 = G_1 \oplus L(C, 1, 1)$, $F_2 = G_2 \oplus L(C, 2, 2^m - 1)$.
- F is the concatenation of F_1 and F_2 .

Case: 3(d)(iv) of Part-A

- Let G_1, G_2 and G_3 be $(m+2e, m)$, $(2e+2, m)$ and $(2e+1, m)$ PROPER S-boxes.
- Define $F_1 = G_1 \oplus L(C, 1, 2^m - 2)$, $F_2 = G_2 \oplus L(C, 2^m - 1, 2^m - 1)$, $F_3 = G_3 \oplus L(C, 1, 2^m - 2)$.
- F is the concatenation of F_1, F_2 and F_3 .

Case: 3(e)(ii) of Part-A

- Let G_1, G_2 and G_3 be $(2m+2e, m)$, $(m+2e, m)$ and $(m+2e-1, m)$ PROPER S-boxes.
- Define $F_1 = G_1 \oplus L(C)$, $F_2 = G_2 \oplus L(C)$, $F_3 = G_3 \oplus L(C, 1, 2)$.
- F is the concatenation of F_1, F_2 and F_3 .

Case: 3(e)(iii) and 3(e)(iv) of Part-A

- Let G_1, G_2 and G_3 be $(2m+2e, m)$, $(m+2e+2, m)$ and $(m+2e+1, m)$ PROPER S-boxes.
- Define $F_1 = G_1 \oplus L(C, 1, 2^m - 2)$, $F_2 = G_2 \oplus L(C, 2^m - 1, 2^m - 1)$, $F_3 = G_3 \oplus L(C, 1, 2^m - 2)$.
- F is the concatenation of F_1, F_2 and F_3 .

Theorem 58 *Construction-I provides a nonlinear (n, m, t)-resilient S-box with nonlinearity* $= (2^{n-1} - 2^{u-1} \times$ effect*), where* effect *is as computed in Part-A.*

Proof : There are several things to be proved.

(a) The output function F is an (n, m) S-box. *(b)* F is t-resilient. *(c)* $\mathsf{nl}(f) = (2^{n-1} - 2^{u-1} \times$ effect$)$.

Proof of (a) The output of Part-A is a *list* $= \langle (n_1, R_1), (n_2, R_2), \cdots, (n_k, R_k) \rangle$. Part-B ensures that for $1 \leq i \leq k$, n_i rows of $L(C)$ are repeated R_i times each. It is easy to verify that in each case of Part-A we have $\sum_{i=1}^{k} n_i R_i = 2^{n-u}$. Since each row $L_{i,*}$ of $L(C)$ defines a (u, m) S-box, ultimately F is an (n, m) S-box.

Proof of (b) Each row $L_{i,*}$ of $L(C)$ defines a t-resilient (u, m) S-box. F is formed by concatenating the rows of $L(C)$ one or more times. Hence F is t-resilient.

Proof of (c) The nonlinearity calculation is similar for all the cases. As an example, we perform the calculation for Case 3(e)(ii). In this case, Part-A computes

$$list = \langle (2, 2^{2m+2e} + 2^{m+2e} + 2^{m+2e-1}), (2^m - 3, 2^{2m+2e} + 2^{m+2e}) \rangle.$$

Let $R_1 = 2^{2m+2e} + 2^{m+2e} + 2^{m+2e-1}$ and $R_2 = 2^{2m+2e} + 2^{m+2e}$. Rows $L_{1,*}$ and $L_{2,*}$ of $L(C)$ are repeated R_1 times each and each of the rows $L_{3,*}$ to $L_{2^m-1,*}$ is repeated R_2 times each. Part-B uses three PROPER functions G_1, G_2 and G_3 to construct S-boxes F_1, F_2 and F_3 respectively. F is the concatenation of F_1, F_2 and F_3. We have to show that if ν is a non constant m-variable linear function and λ is an n-variable linear function, then $d(\nu \circ F, \lambda) \geq (2^{n-1} - 2^{u-1} \times$ effect$)$. We write λ as

$$\lambda(y_1, \cdots, y_{n-u}, x_1, \cdots, x_u) = \lambda_1(y_1, \cdots, y_{n-u}) \oplus \lambda_2(x_1, \cdots, x_u).$$

Let $\nu(z_1, \cdots, z_m) = \langle (c_1, \cdots, c_m), (z_1, \cdots, z_m) \rangle$ for some non-zero vector $c = (c_1, \cdots, c_m) \in \mathbb{F}_2^m$. The Boolean function $\nu \circ F$ is a concatenation of Boolean functions $\nu \circ F_1$, $\nu \circ \mathbb{F}_2$ and $\nu \circ F_3$. For $1 \leq i \leq 2$,

$$\nu \circ F_i = (\nu \circ G_i) \oplus (L(C)c^T)$$

and

$$\nu \circ F_3 = (\nu \circ G_3) \oplus (L(C, 1, 2)c^T).$$

Using Proposition 55, we know that all the entries of the column vector $L(C)c^T$ are distinct u-variable linear functions. Let $L(C)c^T = [\mu_1, \cdots, \mu_{2^m-1}]^T$. The function $\nu \circ F$ is a concatenation of the μ_i's and their complements. Further, μ_1 and μ_2 are repeated R_1 times and $\mu_3, \cdots, \mu_{2^m-1}$ are repeated R_2 times in the construction of $\nu \circ F$. If $\lambda \notin \{\mu_1, \cdots, \mu_{2^m-1}\}$ then $d(\lambda_2, \mu_i) = 2^{u-1}$ for each $1 \leq i \leq 2^m - 1$ and hence $d(\nu \circ F, \lambda) = 2^{n-u}(2^{u-1}) = 2^{n-1}$. Now suppose $\lambda_2 = \mu_i$ for some $i \in \{1, \cdots, 2^m - 1\}$. In this case $d(\nu \circ F, \lambda)$ will be less than 2^{n-1} and the actual value is determined by the repetition factors R_1 and R_2. There are two cases to consider.

Case 1: $\lambda_2 = \mu_1$ or μ_2. Without loss of generality we assume $\lambda_2 = \mu_1$, the other case being similar. Since $\lambda_2 = \mu_1$, we have $d(\lambda_2, \mu_i) = 2^{u-1}$ for $2 \leq i \leq 2^m - 1$. The function μ_2 is repeated R_1 times and each of the functions $\mu_3, \cdots, \mu_{2^m-1}$ is repeated R_2 times. So the total contribution of $\mu_2, \mu_3, \cdots, \mu_{2^m-1}$ to $d(\nu \circ F, \lambda)$ is $2^{u-1}(R_1 + (2^m - 3)R_2)$. We now have to compute the contribution of μ_1 to $d(\nu \circ F, \lambda)$. The function μ_1 is repeated in $\nu \circ F_i$ by XORing with $\nu \circ G_i$. Hence the contribution of μ_1 to $d(F, \lambda)$ is equal to

$$2^u(\mathsf{nl}(\nu \circ G_1) + \mathsf{nl}(\nu \circ G_2) + \mathsf{nl}(\nu \circ G_3)) = 2^u(\mathsf{nl}(G_1) + \mathsf{nl}(G_2) + \mathsf{nl}(G_3))$$

since $\mathsf{nl}(\nu \circ G_i) = \mathsf{nl}(G_i)$. Each G_i is a PROPER function whose nonlinearity is given by Proposition 57.

Hence,

$$\begin{aligned} d(\nu \circ F, \lambda) &= 2^{u-1}(R_1 + (2^m - 3)R_2 + 2(\mathsf{nl}(G_1) + \mathsf{nl}(G_2) + \mathsf{nl}(G_3)) \\ &= 2^{u-1}(2^{n-u} - (R_1 - 2(\mathsf{nl}(G_1) + \mathsf{nl}(G_2) + \mathsf{nl}(G_3)))) \\ &= 2^{n-1} - 2^{u-1}(R_1 - 2(\mathsf{nl}(G_1) + \mathsf{nl}(G_2) + \mathsf{nl}(G_3))) \end{aligned}$$

From the given conditions, it is easy to verify that $\mathsf{effect} = R_1 - 2(\mathsf{nl}(G_1) + \mathsf{nl}(G_2) + \mathsf{nl}(G_3))$ and so $d(\nu \circ F, \lambda) = (2^{n-1} - 2^{u-1} \times \mathsf{effect})$.

Case 2: $\lambda_2 = \mu_i$ for some $i \in \{3, \cdots, 2^m - 1\}$. In this case we proceed as in the previous case to obtain

$$\begin{aligned} d(\nu \circ F, \lambda) &= 2^{u-1}(2R_1 + (2^m - 4)R_2) + 2^u(\mathsf{nl}(G_1) + \mathsf{nl}(G_2)) \\ &= 2^{u-1}(2R_1 + (2^m - 4)R_2 + 2(\mathsf{nl}(G_1) + \mathsf{nl}(G_2)) \\ &= 2^{u-1}(2^{n-u} - R_2 + 2(\mathsf{nl}(G_1) + \mathsf{nl}(G_2)) \\ &= 2^{n-1} - 2^{u-1}(R_2 - 2(\mathsf{nl}(G_1) + \mathsf{nl}(G_2))) \\ &> 2^{n-1} - 2^{u-1} \times \mathsf{effect} \end{aligned}$$

since $\mathsf{effect} = R_1 - 2(\mathsf{nl}(G_1) + \mathsf{nl}(G_2) + \mathsf{nl}(G_3)) > R_2 - 2(\mathsf{nl}(G_1) + \mathsf{nl}(G_2))$.

By *Case 1 and Case 2* above it follows that $\mathsf{nl}(\nu \circ F) = 2^{n-1} - 2^{u-1} \times$ effect. Hence $\mathsf{nl}(F) = 2^{n-1} - 2^{u-1} \times$ effect. ∎

7.4 Nonlinearity Comparison Based on Construction-I

We compare the nonlinearity obtained by Construction-I to the nonlinearity obtained in Theorem 4 of [89]. The nonlinearity obtained in [89] is better than the nonlinearity obtained by other methods. Hence we do not compare our method with the other methods. It is to be noted that in certain cases the search technique of [63] provides better nonlinearity than [89].

Our first observation is that *the nonlinearity obtained by Construction-I is at least as large the nonlinearity obtained in [89].* The intuitive reason is that we use all the rows of the matrix $L(C)$ and hence the repetition factor is less than that of [89]. The detailed verification of the superiority of Construction-I over [89] is straightforward but tedious. In the next table we summarize the cases under which Construction-I yields higher nonlinearity than [89].

Case	Nonlinearity of [89]	Construction-I nonlinearity	
$2m \leq n-u < 3m-3$, π even	$2^{n-1} - 2^{(n+u-m+1)/2}$	$2^{n-1} - 2^{(n+u-m-1)/2} - 3 \times 2^{n-2m-2}$ $2^{n-1} - 2^{(n+u-m-1)/2} - 2^{n-2m}$	(1) (2)
$2m \leq n-u < 3m-3$, π odd	$2^{n-1} - 2^{(n+u-m+2)/2}$	$2^{n-1} - 2^{(n+u-m)/2}$	(3)
$n-u = 3m-3$	$2^{n-1} - 2^{(u+m-1)}$	$2^{n-1} - \frac{11}{16} 2^{(u+m-1)}$	(4)
$n-u \geq 3m$, π odd	$2^{n-1} - 2^{(n+u-m)/2}$	$2^{n-1} - 2^{(n+u-m)/2}(\frac{1}{2} + \frac{1}{2^{m/2}})$ $2^{n-1} - 2^{(n+u-m)/2}(\frac{1}{2} + \frac{1}{2^{((m+1)/2)}})$	(5) (6)

Table 1 : Comparison of Construction-I nonlinearity with the nonlinearity of [89].

We list the different cases of Part-A corresponding to the different rows of the table.
(1) Case 3(d)(ii)first item; **(2)** Case 3(d)(iv); **(3)** Case 3(d)(i) and Case 3(d)(iii); **(4)** Case 3(d)(ii)first item; **(5)** Case 3(e)(iii), $m > 2$ and Case 3(e)(ii), $m > 2$; **(6)** Case 3(e)(iv), $m > 1$.

In Tables 2 to 4 we provide some concrete examples of cases where the nonlinearity obtained by Construction-I is better than that obtained by [89]. Each entry of Tables 3 to 5 is of the form (a, b), where a is the nonlinearity obtained by [89] and b is the nonlinearity obtained by Construction-I.

The linear codes used in Table 2 are $[5, 4, 2], [7, 4, 3]$ and $[8, 4, 4]$. The 2nd, 4th, and 6th rows give the nonlinearity of (n, m, t)-resilient functions corresponding to the codes $[5, 4, 2], [7, 4, 3]$ and $[8, 4, 4]$ respectively for different values of n.

$n=13$	$n=14$	$n=17$	$n=19$
$(2^{12}-2^8),(2^{12}-2^7)$	$(2^{13}-2^8),(2^{13}-\frac{11}{16}2^8)$	$(2^{16}-2^9),(2^{16}-\frac{3}{4}2^9)$	$(2^{18}-2^{10}),(2^{18}-\frac{3}{4}2^{10})$
$n=15$	$n=16$	$n=19$	$n=21$
$(2^{14}-2^{10}),(2^{14}-2^9)$	$(2^{15}-2^{10}),(2^{15}-\frac{11}{16}2^{10})$	$(2^{18}-2^{11}),(2^{18}-\frac{3}{4}2^{11})$	$(2^{20}-2^{12}),(2^{20}-\frac{3}{4}2^{12})$
$n=16$	$n=17$	$n=20$	$n=22$
$(2^{15}-2^{11}),(2^{15}-2^{10})$	$(2^{16}-2^{11}),(2^{16}-\frac{11}{16}2^{11})$	$(2^{19}-2^{12}),(2^{19}-\frac{3}{4}2^{12})$	$(2^{21}-2^{13}),(2^{21}-\frac{3}{4}2^{18})$

Table 2 : Comparison of Construction-I nonlinearity with [89] for $m=4$ and resiliency $=1,2,3$.

The linear codes used in Table 3 are $[6,5,2]$, $[9,5,3]$ and $[10,5,4]$.

$n=16$	$n=17$	$n=18$	$n=21$
$(2^{15}-2^9),(2^{15}-\frac{5}{8}2^9)$	$(2^{16}-2^{10}),(2^{16}-2^9)$	$(2^{17}-2^{10}),(2^{17}-\frac{11}{16}2^{10})$	$(2^{20}-2^{11}),(2^{20}-\frac{5}{8}2^{11})$
$n=19$	$n=20$	$n=21$	$n=24$
$(2^{18}-2^{12}),(2^{18}-\frac{5}{8}2^{12})$	$(2^{19}-2^{13}),(2^{19}-2^{12})$	$(2^{20}-2^{13}),(2^{20}-\frac{11}{16}2^{13})$	$(2^{23}-2^{14}),(2^{23}-\frac{5}{8}2^{14})$
$n=18$	$n=19$	$n=20$	$n=25$
$(2^{17}-2^{11}),(2^{17}-\frac{5}{8}2^{11})$	$(2^{18}-2^{12}),(2^{18}-2^{11})$	$(2^{19}-2^{12}),(2^{19}-\frac{11}{16}2^{12})$	$(2^{24}-2^{15}),(2^{24}-\frac{5}{8}2^{15})$

Table 3: Comparison of Construction-I nonlinearity with [89] for $m=5$ and resiliency $=1,2,3$.

The linear codes used in Table 4 are $[7,6,2]$, $[10,6,3]$ and $[10,6,4]$.

$n=19$	$n=20$	$n=21$	$n=22$
$(2^{18}-2^{11}),(2^{18}-2^{10})$	$(2^{19}-2^{11}),(2^{19}-\frac{19}{32}2^{11})$	$(2^{20}-2^{12}),(2^{20}-2^{11})$	$(2^{21}-2^{12}),(2^{21}-\frac{11}{16}2^{12})$
$n=22$	$n=23$	$n=24$	$n=25$
$(2^{21}-2^{14}),(2^{21}-2^{13})$	$(2^{22}-2^{14}),(2^{22}-\frac{19}{32}2^{14})$	$(2^{23}-2^{15}),(2^{23}-2^{14})$	$(2^{24}-2^{15}),(2^{24}-\frac{11}{16}2^{15})$
$n=22$	$n=23$	$n=24$	$n=25$
$(2^{21}-2^{14}),(2^{21}-2^{13})$	$(2^{22}-2^{14}),(2^{22}-\frac{19}{32}2^{14})$	$(2^{23}-2^{15}),(2^{23}-2^{14})$	$(2^{24}-2^{15}),(2^{24}-\frac{11}{16}2^{15})$

Table 4 : Comparison of Construction-I nonlinearity with [89] for $m=6$ and resiliency $=1,2,3$.

Nonlinearity of $(36,8,t)$ resilient S-boxes has been used as very important examples in [70, 63, 89]. Now we compare our nonlinearity with those.

t	7	6	5	4	3	2	1
[70]	$2^{35}-2^{27}$	$2^{35}-2^{27}$	$2^{35}-2^{26}$	$2^{35}-2^{25}$	$2^{35}-2^{24}$	$2^{35}-2^{23}$	$2^{35}-2^{22}$
[63]	$2^{35}-2^{22}$	-	$2^{35}-2^{23}$	$2^{35}-2^{22}$	$2^{35}-2^{22}$	$2^{35}-2^{21}$	$2^{35}-2^{21}$
[89]	$2^{35}-2^{25}$	$2^{35}-2^{24}$	$2^{35}-2^{23}$	$2^{35}-2^{23}$	$2^{35}-2^{20}$	$2^{35}-2^{20}$	$2^{35}-2^{18}$
Ours	$2^{35}-2^{24}$	$2^{35}-\frac{35}{64}2^{24}$	$2^{35}-\frac{19}{32}2^{23}$	$2^{35}-2^{22}$	$2^{35}-2^{20}$	$2^{35}-\frac{9}{16}2^{20}$	$2^{35}-2^{18}$
Codes	$[20,8,8]$	$[19,8,7]$	$[17,8,6]$	$[16,8,5]$	$[13,8,4]$	$[12,8,3]$	$[9,8,2]$

Table 5 : Comparison of nonlinearity of $(36,8,t)$-resilient S-boxes using different methods.

The results of [63] are not constructive. They show that resilient S-box with such parameter exist. *Note that, except for resiliencies of order* 1 *and* 3 *our nonlinearity is better than nonlinearity of [89].* It should also be noted that in all the cases we provide construction with currently best known nonlinearity.

7.5 Conclusion

In this chapter we considered the construction of nonlinear resilient S-boxes. We proved that the correlation immunity of a resilient S-box is preserved under composition with an arbitrary Boolean function. Our main contribution is to construct methods for nonlinear resilient S-boxes. Our construction is based on concatenation of small affine function to build nonlinear resilient S-boxes. We sharpen the technique to construct (n, m, t)-resilient S-boxes with the currently best known nonlinearity.

Chapter 8

Software Implementation of Resilient Maiorana-McFarland S-Boxes

In this chapter we consider implementation of resilient Maiorana-McFarland S-boxes. Such S-boxes have application in the design of stream ciphers and their efficient software implementation is important for software implementation of the corresponding stream ciphers. Most papers on construction of resilient Maiorana-McFarland S-boxes provide mathematical descriptions which are not sufficient for implementation purposes. Moreover, the mathematical descriptions do not bring out the fact that in most cases such S-boxes can be efficiently implemented using a small amount of memory. Our work shows that these S-boxes can be implemented using a small amount of memory and the output of an S-box can be evaluated using very little computation.

8.1 Applicability of Resilient S-boxes in Nonlinear Combiner Model

In the nonlinear combiner model (see Section 2.3.1 of Chapter 2) we can use S-boxes in place of Boolean functions to increase throughput. (Note that the criteria of resiliency is perhaps not that important for the nonlinear filter model and hence we do not consider this model in this chapter.) Now we provide some justification as to why we can use S-boxes as multioutput combining functions.

We consider the composition of an (n, m) S-box and an m-variable Boolean function.

The Walsh transform of the composition is already proved in the Theorem 16 of Chapter 4.

Let $f : \{0,1\}^n \to \{0,1\}^m$ and $g : \{0,1\}^m \to \{0,1\}$. Then for any $w \in \mathbb{F}_2^n$,

$$W_{(g \circ f)}(w) = \frac{1}{2^m} \sum_{v \in \mathbb{F}_2^m} W_g(v) W_{(l_v \circ f)}(w)$$

where $l_v = \langle v, x \rangle$ and $(l_v \circ f)(x) = \langle v, f(x) \rangle$.

Corollary 59 *Let $f : \{0,1\}^n \to \{0,1\}^m$ be a balanced S-box. Let g be an m-variable Boolean function. Then $(g \circ f)$ is balanced if and only if g is balanced.*

Proof : Since f is balanced, $W_{(l_v \circ f)}(w) = 0$ for all nonzero $v \in \mathbb{F}_2^m$.
Thus $W_{g \circ f}(0) = \frac{1}{2^m} W_g(0) 2^m = W_g(0)$. ∎

Remark: It is possible for $(g \circ f)$ to be balanced even when either only f is unbalanced or both f and g are unbalanced. We present examples for these cases. Let $f : \{0,1\}^3 \to \{0,1\}^2$ be an unbalanced S-box and f_1, f_2 are component functions.

(a) Let $f_1(x_1, x_2, x_3) = x_1 \oplus x_2 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_2x_3$ and $f_2(x_1, x_2, x_3) = x_2 \oplus x_1x_2 \oplus x_2x_3 \oplus x_1x_3 \oplus x_1x_2x_3$ and $g(x_1, x_2) = x_1 \oplus x_2$. Here f is unbalanced but g is balanced. Observe $(g \circ f)(x_1, x_2, x_3) = f_1(x_1, x_2, x_3) \oplus f_2(x_1, x_2, x_3) = x_1 \oplus x_2x_3$ is balanced.

(b) Let $f_1(x_1, x_2, x_3) = x_3 \oplus x_1x_2 \oplus x_1x_2x_3$ and $f_2(x_1, x_2, x_3) = x_2 \oplus x_3 \oplus x_1x_2 \oplus x_2x_3 \oplus x_1x_2x_3$ and $g(x_1, x_2) = x_1x_2$. Here both f and g are unbalanced. Observe $(g \circ f)(x_1, x_2, x_3) = f_1(x_1, x_2, x_3) f_2(x_1, x_2, x_3) = x_3$, which is balanced.

Theorem 16 and Corollary 59 provide the following theorem.

Theorem 60 *Let f be a t-resilient S-box and g be any arbitrary Boolean function then $(g \circ f)$ is t-CI. Further $(g \circ f)$ is t-resilient if and only if g is balanced.*

Theorem 60 shows that correlation immunity of an (n, m, t)-resilient S-box is preserved under composition with an *arbitrary* m-variable Boolean function. This is an important security property for the use of resilient S-boxes in stream cipher design. Thus to obtain correlation between an arbitrary combination of the output variables and a non trivial linear function of the input variables, we must choose at least $(t + 1)$ input variables.

The work of Zhang and Chan [121] provide an upper bound on the correlation between a linear combination of input variables and any arbitrary (i.e., not necessarily linear) combination of the output variables. This upper bound is $2^{\frac{m}{2}}$ times more than the correlation determined by the nonlinearity of f (see [121, Theorem 4]).

Theorem 4 of [121] states: Let F be a (n, m) S-box and $c(f, g)$ is the correlation coefficient of f and g. Let maximum correlation coefficient between F and linear function l_w be defined as

$$C_F(w) = \max_{g \in \Omega_m} c(g \circ F, l_w)$$

where Ω_m is set of all m-variable Boolean functions. The Walsh transform of $l_v \circ F$ at point w is defined as

$$W_F(v, w) = \sum_{x \in \mathbb{F}_2^n} (-1)^{\langle v, F(x) \rangle \oplus \langle w, x \rangle}.$$

Then

$$C_F(w) \leq \frac{2^{m/2}}{2^n} \max_{v \in \mathbb{F}_2^m} |W_F(v, w)|. \tag{8.1}$$

Thus, as observed in above theorem, the value of m must be small compared to that of n. Increasing the value of m will leak extra information about the input. On the other hand increasing the value of m increases throughput. *So as stated in [121] there is a trade-off against correlation attacks and throughput of the keystream sequences in the design of S-boxes for stream ciphers.*

In the example we describe in Section 8.4, we have $n = 16$ and $m = 4$. The correlation between any nonzero linear combination of at least $(t + 1)$ input variables and any nonzero linear combination of output variables is at most $\frac{11}{2^9}$. If we choose any arbitrary combination of the output variables, then from Equation 8.1, the upper bound for the correlation is $\frac{11}{2^7}$, which is not significantly more than $\frac{11}{2^9}$. There is also a related issue, which we discuss next.

The resistance of the system to correlation attacks depends not only on maximum correlation coefficient but also on the lengths of the LFSRs. Thus to achieve a target resistance one has to correspondingly choose the lengths of the LFSRs. Now we compute the amount by which the length of LFSRs are to be increased to get same level of security considering the increase in value of maximum correlation coefficient. Let N be the number of key bits required to successfully carry out correlation attack as has been done in [26]. From [26] we have

$$N \simeq \frac{1}{4}.(2kj! \ln 2)^{\frac{1}{j}}.\beta^{-2}.2^{\frac{L-k}{j}},$$

where k, j are algorithm parameters, L is effective length of LFSRs and $\beta = \max_{w \in \mathbb{F}_2^n} C_F(w)$ is the bias. In the attack of [26] the complexity of the precomputation phase is approximately $N^{\lceil (j-1)/2 \rceil}$ and requires $N^{\lfloor (j-1)/2 \rfloor}$ memory. The number of parity check equations that need to be stored is roughly $\frac{N^j}{j!}.2^{-(L-k)}$. To successfully carry out the attack the values of $N^{\lceil (j-1)/2 \rceil}$ and $\frac{N^j}{j!}.2^{-(L-k)}$ should not be very large and hence the value of j is small (≤ 6).

Now consider the effect of replacing a Boolean function by an S-box. Let $\beta' \geq \beta$ be the modified bias. To keep the required number of ciphertext bits N same for both Boolean functions and S-boxes, suppose L is increased to L'. Then

$$\beta^{-2} \times 2^{\frac{L-k}{j}} = (\beta')^{-2} \times 2^{\frac{L'-k}{j}},$$

and from this $L' = L + 2j.\log_2 \frac{\beta'}{\beta}$. In our example of Section 8.4, we have $\frac{\beta'}{\beta} = 4$. So for same value of N we have $L' = L + 4j \leq L + 24$. Thus if we increase the effective length of LFSR by just 24 bits, we get same level of security. In our example $t = 2$ i.e., the S-box is 2-resilient. To carry out correlation attack at least 3 LFSRs have to be taken. So length of each LFSR should increase by $\frac{24}{3} = 8$ bits. As $n = 16$, total increase in length of LFSRs will be $8 \times 16 = 128$ bits and throughput will increase $m = 4$ times. So there is a trade-off between increase in length and increase in throughput.

8.2 Algebraic Attacks

Algebraic attacks [28, 29] are a new type of attacks on stream cipher. It recovers the secret key by solving an overdefined system of multivariate equations. These attacks exploit the fact that even if a function may have high degree it may have a low degree multiple. It is shown in [28] that if f be any Boolean with n inputs then there is a Boolean function $g \neq 0$ of degree at most $\lceil \frac{n}{2} \rceil$ such that $f * g$ is of degree at most $\lceil \frac{n}{2} \rceil$.

So we can get multiple of a Boolean function which has multiple of degree at most $\lceil \frac{n}{2} \rceil$. In [77] it is shown that Maiorana-McFarland class of n-variable Boolean functions inherently has an annihilator of degree $(n-r+1)$, where r is the number of variables of affine functions which are concatenated to construct the n-variable Boolean function. If r is very close to n then $(n-r+1)$ is small and the attacker gains. If $r > \frac{n}{2}$ the annihilator has degree $< \frac{n}{2}$. On the other hand, if $r \leq \lfloor \frac{n}{2} \rfloor$, the degree of annihilator of [77] is $\geq \lceil \frac{n}{2} \rceil$. In such a situation, the multiplier of [28] is more useful, since its degree is at most $\lceil \frac{n}{2} \rceil$.

In our construction in Chapter 7, generally $r < \frac{n}{2}$ and hence degree of annihilator is $> \frac{n}{2}$. In our example $n = 16$, $r = 7$ so $n - r + 1 = 10$ which is greater than $\frac{n}{2} = 8$. So by just considering inherent annihilator of degree $(n - r + 1)$ (as in [77]) one cannot conclude that Maiorana-McFarland constructions are prone to algebraic attacks. To apply algebraic attack on Maiorana-McFarland class of S-boxes one has to find low degree multiple as with other classes of Boolean functions.

8.3 Maiorana-McFarland Construction for S-boxes

A description of Maiorana-McFarland resilient S-box in Chapter 7 consists of the following information.

1. The matrix $L(C)$. (For definition of $L(C)$ see Theorem 54 in Chapter 7.)
2. A list $((n_1, R_1), (n_2, R_2), \cdots, (n_k, R_k))$ which signifies that n_i rows of $L(C)$ are to be repeated R_i times each. For $1 \leq i \leq k$, either $n_i = 2^{r_i}$ or $n_i = 2^{r_{i,1}} + 2^{r_{i,2}}$ or $n_i = 2^{r_{i,1}} + 2^{r_{i,2}} + 2^{r_{i,3}}$.
3. For $1 \leq i \leq k$ if $n_i = 2^{r_i}$, then G_i is a nonlinear (r_i, m) S-box otherwise $G_{i,j}$ is a nonlinear $(r_{i,j}, m)$ S-box where $2 \leq j \leq 3$.

The method described in Chapter 7 consists of around 15–20 different cases depending on the values of the parameters n, m and t. The cases are chosen to maximise the nonlinearity. Each case consists of a description of the above form. Hence the actual implementation method for any specific function can be described by a general method. In the next section, we illustrate this general method by considering a specific example, which is sufficiently general enough to cover all the cases mentioned in Chapter 7.

8.4 A Concrete Example

In this section we provide description of the construction of a $(16, 4, 2)$ S-box. The purpose of this example is to illustrate the methodology behind the general description provided later.

8.4.1 Construction of $L(C)$

We use $[7, 4, 3]$ linear Hamming code C. Let $c_0 = 1101000, c_1 = 1010100, c_2 = 0110010, c_3 = 1110001$ be a basis of C. Let β be a root of primitive polynomial $x^4 + x + 1$ and $(1, \beta, \beta^2, \beta^3)$ be a polynomial basis of $GF(2^4)$. Any element γ of $GF(2^4)$ can be written as $\gamma = \gamma_0 + \gamma_1\beta + \gamma_2\beta^2 + \gamma_3\beta^3$ where $\gamma_0, \gamma_1, \gamma_2, \gamma_3 \in \{0, 1\}$. We define a bijection $\phi : GF(2^4) \mapsto C$ by (see Lemma 7 in [63])

$$\phi(a_0 + a_1\beta + a_2\beta^2 + a_3\beta^3) = a_0c_0 \oplus a_1c_1 \oplus a_2c_2 \oplus a_3c_3.$$

Then

$$\phi(1) = 1101000, \quad \phi(\beta) = 1010100, \quad \phi(\beta^2) = 0110010, \quad \phi(\beta^3) = 1110001, \quad \phi(\beta^4) = 0111100,$$
$$\phi(\beta^5) = 1100110, \quad \phi(\beta^6) = 1000011, \quad \phi(\beta^7) = 1001101, \quad \phi(\beta^8) = 1011010, \quad \phi(\beta^9) = 0100101,$$
$$\phi(\beta^{10}) = 0001110, \quad \phi(\beta^{11}) = 0010111, \quad \phi(\beta^{12}) = 1111111, \quad \phi(\beta^{13}) = 0101011, \quad \phi(\beta^{14}) = 0011001.$$

We have constructed the 15×4 matrix D as given below (see Theorem 54 and Lemma 7 in [63]).

$$D = \begin{bmatrix} \phi(1) & \phi(\beta) & \phi(\beta^2) & \phi(\beta^3) \\ \phi(\beta) & \phi(\beta^2) & \phi(\beta^3) & \phi(\beta^4) \\ \vdots & \vdots & \vdots & \vdots \\ \phi(\beta^{14}) & \phi(1) & \phi(\beta) & \phi(\beta^2) \end{bmatrix}.$$

As defined after Theorem 54 we have a 15×4 matrix $L(C)$ whose entries are 7 variable linear functions.

$$L(C) = \begin{bmatrix} x_1 \oplus x_2 \oplus x_4 & x_1 \oplus x_3 \oplus x_5 & x_2 \oplus x_3 \oplus x_6 & x_1 \oplus x_2 \oplus x_3 \oplus x_7 \\ x_1 \oplus x_3 \oplus x_5 & x_2 \oplus x_3 \oplus x_6 & x_1 \oplus x_2 \oplus x_3 \oplus x_7 & x_2 \oplus x_3 \oplus x_4 \oplus x_5 \\ x_2 \oplus x_3 \oplus x_6 & x_1 \oplus x_2 \oplus x_3 \oplus x_7 & x_2 \oplus x_3 \oplus x_4 \oplus x_5 & x_1 \oplus x_2 \oplus x_5 \oplus x_6 \\ x_1 \oplus x_2 \oplus x_3 \oplus x_7 & x_2 \oplus x_3 \oplus x_4 \oplus x_5 & x_1 \oplus x_2 \oplus x_5 \oplus x_6 & x_1 \oplus x_6 \oplus x_7 \\ x_2 \oplus x_3 \oplus x_4 \oplus x_5 & x_1 \oplus x_2 \oplus x_5 \oplus x_6 & x_1 \oplus x_6 \oplus x_7 & x_1 \oplus x_4 \oplus x_5 \oplus x_7 \\ x_1 \oplus x_2 \oplus x_5 \oplus x_6 & x_1 \oplus x_6 \oplus x_7 & x_1 \oplus x_4 \oplus x_5 \oplus x_7 & x_1 \oplus x_3 \oplus x_4 \oplus x_6 \\ x_1 \oplus x_6 \oplus x_7 & x_1 \oplus x_4 \oplus x_5 \oplus x_7 & x_1 \oplus x_3 \oplus x_4 \oplus x_6 & x_2 \oplus x_5 \oplus x_7 \\ x_1 \oplus x_4 \oplus x_5 \oplus x_7 & x_1 \oplus x_3 \oplus x_4 \oplus x_6 & x_2 \oplus x_5 \oplus x_7 & x_4 \oplus x_5 \oplus x_6 \\ x_1 \oplus x_3 \oplus x_4 \oplus x_6 & x_2 \oplus x_5 \oplus x_7 & x_4 \oplus x_5 \oplus x_6 & x_3 \oplus x_5 \oplus x_6 \oplus x_7 \\ x_2 \oplus x_5 \oplus x_7 & x_4 \oplus x_5 \oplus x_6 & x_3 \oplus x_5 \oplus x_6 \oplus x_7 & x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 \\ x_4 \oplus x_5 \oplus x_6 & x_3 \oplus x_5 \oplus x_6 \oplus x_7 & x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 & x_2 \oplus x_4 \oplus x_6 \oplus x_7 \\ x_3 \oplus x_5 \oplus x_6 \oplus x_7 & x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 & x_2 \oplus x_4 \oplus x_6 \oplus x_7 & x_3 \oplus x_4 \oplus x_7 \\ x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 & x_2 \oplus x_4 \oplus x_6 \oplus x_7 & x_3 \oplus x_4 \oplus x_7 & x_1 \oplus x_2 \oplus x_4 \\ x_2 \oplus x_4 \oplus x_6 \oplus x_7 & x_3 \oplus x_4 \oplus x_7 & x_1 \oplus x_2 \oplus x_4 & x_1 \oplus x_3 \oplus x_5 \\ x_3 \oplus x_4 \oplus x_7 & x_1 \oplus x_2 \oplus x_4 & x_1 \oplus x_3 \oplus x_5 & x_2 \oplus x_3 \oplus x_6 \end{bmatrix}.$$

8.4.2 Construction of $(16, 4, 2)$ Resilient S-Box

Following Chapter 7 the description of a $(16, 4, 2)$-resilient S-box is as follows (see Case: 3(d)(ii) first item of Part-A and Part B in Chapter 7.

1. Matrix $L(C)$ as constructed above.

2. The list $\langle(2, 2^{m+1}+2^1+2^0), (2^m-3, 2^{m+1}+2^1)\rangle = \langle(2, 2^5+2^1+2^0), (2^4-3, 2^5+2^1)\rangle = \langle(2, 35), (13, 34)\rangle$. This implies that 13 rows (along with their complements) of $L(C)$ are to be repeated 34 times each and 2 rows (along with their complements) are to be repeated 35 times each.

3. A nonlinear $(5, 4)$ and an $(1, 4)$ S-Box G_1 and G_2 respectively as described below.

We construct a maximally nonlinear $(5,4)$ S-box G_1. For this we define a bijection on $GF(2^5)$ by $x \mapsto x^3$ (see [36]). We represent this bijection as a map from $\{0,1\}^5$ to $\{0,1\}^4$ by representing $GF(2^5)$ using the primitive polynomial x^5+x+1. The S-box G_1 is obtained by dropping one bit of the output of this bijection. Let $G_{1,1}, G_{1,2}, G_{1,3}, G_{1,4}$ be the four component functions of G_1. Each $G_{1,j}$ is a 5-variable Boolean function as given below.

$$\begin{aligned}
G_{1,1}(y_1,\cdots,y_5) &= y_4 \oplus y_1y_3 \oplus y_1y_5 \oplus y_2y_3 \oplus y_2y_4 \oplus y_2y_5 \oplus y_3y_4 \oplus y_3y_5 \oplus y_4y_5 \\
G_{1,2}(y_1,\cdots,y_5) &= y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_1y_5 \oplus y_3y_4 \oplus y_3y_5 \\
G_{1,3}(y_1,\cdots,y_5) &= y_5 \oplus y_1y_2 \oplus y_1y_3 \oplus y_2y_3 \oplus y_1y_5 \oplus y_3y_5 \oplus y_4y_5 \\
G_{1,4}(y_1,\cdots,y_5) &= y_3 \oplus y_4 \oplus y_5 \oplus y_1y_2 \oplus y_1y_4 \oplus y_4y_5
\end{aligned}$$

The output of G_1 is given by a 4-bit string which can be represented as a hexadecimal digit. We can write G_1 as a 32-tuple of hexadecimal digits having the following form:

$$G_1 = (0,0,4,7,5,F,B,2,D,C,1,3,8,F,2,A,7,9,B,6,C,8,A,D,1,E,5,9,6,3,8,D)$$

This representation of G_1 is useful for a table look-up implementation.

The $(1,4)$ S-box G_2 is taken to be the constant function $G_2(y_5) = (0,0,0,0)$. Thus the four component functions $G_{2,j}(y_5)$ $(1 \le j \le 4)$ are single variable constant functions. We now define

$$\begin{aligned}
F_1(y_1,\ldots,y_5,x_1,\ldots,x_7) &= G_1(y_1,\ldots,y_5) \oplus L(C) \\
F_2(y_5,x_1,\ldots,x_7) &= G_2(y_5) \oplus L(C) \\
F_3(x_1,\ldots,x_7) &= L(C,1,2)
\end{aligned}$$

Thus $F_1(y_1, \ldots, y_5, x_1, \ldots, x_7)$ is of the form.

$$\begin{bmatrix}
G_{1,1} \oplus x_1 \oplus x_2 \oplus x_4 & G_{1,2} \oplus x_1 \oplus x_3 \oplus x_5 & G_{1,3} \oplus x_2 \oplus x_3 \oplus x_6 & G_{1,4} \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_7 \\
G_{1,1} \oplus x_1 \oplus x_3 \oplus x_5 & G_{1,2} \oplus x_2 \oplus x_3 \oplus x_6 & G_{1,3} \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_7 & G_{1,4} \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \\
G_{1,1} \oplus x_2 \oplus x_3 \oplus x_6 & G_{1,2} \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_7 & G_{1,3} \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 & G_{1,4} \oplus x_1 \oplus x_2 \oplus x_5 \oplus x_6 \\
G_{1,1} \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_7 & G_{1,2} \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 & G_{1,3} \oplus x_1 \oplus x_2 \oplus x_5 \oplus x_6 & G_{1,4} \oplus x_1 \oplus x_6 \oplus x_7 \\
G_{1,1} \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 & G_{1,2} \oplus x_1 \oplus x_2 \oplus x_5 \oplus x_6 & G_{1,3} \oplus x_1 \oplus x_6 \oplus x_7 & G_{1,4} \oplus x_1 \oplus x_4 \oplus x_5 \oplus x_7 \\
G_{1,1} \oplus x_1 \oplus x_2 \oplus x_5 \oplus x_6 & G_{1,2} \oplus x_1 \oplus x_6 \oplus x_7 & G_{1,3} \oplus x_1 \oplus x_4 \oplus x_5 \oplus x_7 & G_{1,4} \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_6 \\
G_{1,1} \oplus x_1 \oplus x_6 \oplus x_7 & G_{1,2} \oplus x_1 \oplus x_4 \oplus x_5 \oplus x_7 & G_{1,3} \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_6 & G_{1,4} \oplus x_2 \oplus x_5 \oplus x_7 \\
G_{1,1} \oplus x_1 \oplus x_4 \oplus x_5 \oplus x_7 & G_{1,2} \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_6 & G_{1,3} \oplus x_2 \oplus x_5 \oplus x_7 & G_{1,4} \oplus x_4 \oplus x_5 \oplus x_6 \\
G_{1,1} \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_6 & G_{1,2} \oplus x_2 \oplus x_5 \oplus x_7 & G_{1,3} \oplus x_4 \oplus x_5 \oplus x_6 & G_{1,4} \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_7 \\
G_{1,1} \oplus x_2 \oplus x_5 \oplus x_7 & G_{1,2} \oplus x_4 \oplus x_5 \oplus x_6 & G_{1,3} \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_7 & G_{1,4} \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 \\
G_{1,1} \oplus x_4 \oplus x_5 \oplus x_6 & G_{1,2} \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_7 & G_{1,3} \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 & G_{1,4} \oplus x_2 \oplus x_4 \oplus x_6 \oplus x_7 \\
G_{1,1} \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_7 & G_{1,2} \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 & G_{1,3} \oplus x_2 \oplus x_4 \oplus x_6 \oplus x_7 & G_{1,4} \oplus x_3 \oplus x_4 \oplus x_7 \\
G_{1,1} \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 & G_{1,2} \oplus x_2 \oplus x_4 \oplus x_6 \oplus x_7 & G_{1,3} \oplus x_3 \oplus x_4 \oplus x_7 & G_{1,4} \oplus x_1 \oplus x_2 \oplus x_4 \\
G_{1,1} \oplus x_2 \oplus x_4 \oplus x_6 \oplus x_7 & G_{1,2} \oplus x_3 \oplus x_4 \oplus x_7 & G_{1,3} \oplus x_1 \oplus x_2 \oplus x_4 & G_{1,4} \oplus x_1 \oplus x_3 \oplus x_5 \\
G_{1,1} \oplus x_3 \oplus x_4 \oplus x_7 & G_{1,2} \oplus x_1 \oplus x_2 \oplus x_4 & G_{1,3} \oplus x_1 \oplus x_3 \oplus x_5 & G_{1,4} \oplus x_2 \oplus x_3 \oplus x_6
\end{bmatrix}.$$

We have $F_2(y_5, x_1, \ldots, x_7) = L(C)$, where each entry of $L(C)$ is treated as degenerate function of y_5. Lastly $F_3(x_1, \ldots, x_7)$ is defined as

$$\begin{bmatrix}
x_1 \oplus x_2 \oplus x_4 & x_1 \oplus x_3 \oplus x_5 & x_2 \oplus x_3 \oplus x_6 & x_1 \oplus x_2 \oplus x_3 \oplus x_7 \\
x_1 \oplus x_3 \oplus x_5 & x_2 \oplus x_3 \oplus x_6 & x_1 \oplus x_2 \oplus x_3 \oplus x_7 & x_2 \oplus x_3 \oplus x_4 \oplus x_5
\end{bmatrix}.$$

Remarks :

1. Each entry in the 15×4 matrix F_1 is a 12-variable Boolean function. This accounts for 32 repetitions of each of the 15 rows of $L(C)$.
2. Each entry in the 15×4 matrix F_2 is an 8-variable Boolean function. This accounts for 2 repetitions of each of the 15 rows of $L(C)$.
3. Each entry in the 2×4 matrix F_3 is a 7-variable Boolean function. This accounts for 1 repetition of each of the first two rows of $L(C)$.

Thus the first two rows of $L(C)$ are repeated 35 times and the next 13 rows of $L(C)$ are repeated 34 times. The desired $(16, 4, 2)$ S-box F is the concatenation of F_1, F_2 and F_3 as we explain below. We use the notation $F_{k,i,j}$ to denote the (i, j)th entry of matrix F_k and $F_{k,i}$ to be the ith row of matrix F_k. For a binary string $a_1, \ldots, a_k$, we define $\delta(a_1, \cdots, a_k) = a_1 2^k + a_2 2^{k-1} + \cdots + a_{k-1} 2 + a_k + 1$. We write $\overline{x} = (x_1, \ldots, x_7)$, $\overline{y} = (y_1, \ldots, y_5)$ and $\overline{z} = (z_1, \ldots, z_4)$.

$$\left.\begin{array}{rcll}
F(\overline{z}, \overline{y}, \overline{x}) & = & F_{1,\delta(z_1,z_2,z_3,z_4)}(\overline{y}, \overline{x}) & \text{if } \overline{z} \neq (1111) \\
& = & F_{2,\delta(y_1,y_2,y_3,y_4)}(y_5, \overline{x}) & \text{if } \overline{z} = (1111) \text{ and } (y_1, y_2, y_3, y_4) \neq (1111) \\
& = & F_{3,y_5}(\overline{x}) & \text{if } \overline{z} = (y_1, y_2, y_3, y_4) = (1111)
\end{array}\right\} \quad (8.2)$$

The above defined F is our $(16, 4, 2)$ S-box. Next we explain how to implement the above $(16, 4, 2)$ S-box efficiently and to compute the 4-bit output when 16-bit input is given.

8.4.3 Implementation

For implementing the S-box we need to store the following.

1. A 15×4 table D whose each entry is a 7-bit binary vector. Thus each entry of D represents a 7-variable linear function.
2. An array G of length 32 whose each entry is a 4-bit vector.

Thus the number of bits of storage required is $15 \times 4 \times 7 + 32 \times 4 = 488$ bits.

Remark : Note that the number of bits required to store the representation is only 488 bits which is much less than directly storing F as a look-up table of size $4 \times 2^{16} = 2^{18}$ bits.

8.4.4 Computing the Output

The question now is the following: Given the representation of F as defined in Section 8.4.3, how to compute the 4-bit output when a 16-bit input is given? Here we provide the answer to this question. Let the input be $(z_1, z_2, z_3, z_4, y_1, y_2, y_3, y_4, y_5, x_1, x_2, x_3, x_4, x_5, x_6, x_7)$. We will denote the kth row of D by D_k and the kth entry of G by G_k. Note that D_k is a 4-tuple of 7-bit strings and G_k is a 4-bit string. By $D_{k,j}$ $(1 \leq j \leq 4)$ we will denote the (k, j)th entry of D.

1. If $(z_1, z_2, z_3, z_4) = (y_1, y_2, y_3, y_4) = (1111)$ then the output is the 4-bit vector given by

$$(\langle D_{y_5,1}, \overline{x} \rangle, \langle D_{y_5,2}, \overline{x} \rangle, \langle D_{y_5,3}, \overline{x} \rangle, \langle D_{y_5,4}, \overline{x} \rangle)$$

   The cost is one table lookup for D, 4 inner product computations and one comparison to check $z_1 z_2 z_3 z_4 y_1 y_2 y_3 y_4 = 11111111$.

2. If $(z_1, z_2, z_3, z_4) = 1111$ and $(y_1, y_2, y_3, y_4) \neq (1111)$ then we compute $k = \delta(y_1, y_2, y_3, y_4)$ and the output is the 4-bit vector given by

$$(\langle D_{k,1}, \overline{x} \rangle, \langle D_{k,2}, \overline{x} \rangle, \langle D_{k,3}, \overline{x} \rangle, \langle D_{k,4}, \overline{x} \rangle).$$

   The cost is one table lookup for D, 4 inner product computations and two comparisons to verify $(z_1, z_2, z_3, z_4) = 1111$ and $(y_1, y_2, y_3, y_4) \neq (1111)$.

3. If $(z_1, z_2, z_3, z_4) \neq (1111)$ then we compute $k = \delta(z_1, z_2, z_3, z_4)$, $l = \delta(y_1, y_2, y_3, y_4, y_5)$ and the output is the 4-bit vector given by

$$G_l \oplus (\langle D_{k,1}, \overline{x} \rangle, \langle D_{k,2}, \overline{x} \rangle, \langle D_{k,3}, \overline{x} \rangle, \langle D_{k,4}, \overline{x} \rangle).$$

The cost is one table lookup each for D and G, 4 inner product computations, one XOR of 4-bit strings and one comparison to check $(z_1, z_2, z_3, z_4) \neq (1111)$.

The maximum cost occurs in the last case, which is 2 table look-ups, 4 inner product computations, one XOR and one bit string comparison. Alternatively, if F is stored directly as a look-up table of 4×2^{16} bits then the cost is one table look-up. However, this table look-up is into a large table (of size 4×2^{16} bits) whereas the two table look-ups into D and G are into much smaller sized tables.

8.5 General Methodology

The example in Section 8.4 illustrates the method for software implementation of the resilient S-boxes described in Chapter 7. The basic points behind the example generalize quite easily. In this section, we briefly describe this generalization.

As mentioned in Section 8.3, a description of an S-box constructed using the method of Chapter 7 consists of the matrix $L(C)$, the list of pairs indicating repetition pattern of linear functions and the required nonlinear S-boxes G_i's and/or $G_{i,j}$'s.

The matrix $L(C)$ is stored as a $((2^m - 1) \times m)$ matrix D whose each entry is a u-bit vector. Thus storing $L(C)$ requires $u \times m \times (2^m - 1)$ bits. Each G_i is an (r_i, m) S-box and each $G_{i,j}$ is an $(r_{i,j}, m)$ S-box. Storing these as look-up tables require $m \times 2^{r_i}$ and $m \times 2^{r_{i,j}}$ bits respectively. Nothing else requires to be stored. Since the r_i's and the $r_{i,j}$'s are much smaller compared to n, the total amount of storage space required will be much smaller than $m \times 2^n$ bits.

The constructed S-box F can be expressed as in (8.2). The strategy for evaluating the output of F depends on the nature of this expression. Moreover, it is clear that such a strategy can be formulated as described in Section 8.4.4. The cost of evaluating the output will be m inner product computation of u-bit strings; one table look-up into matrix D; a few table look-ups into the G_i's and the $G_{i,j}$'s and some bit vector comparisons to determine the case which is applicable for the given input vector. Given a particular S-box, the actual strategy is easy to design as discussed in Section 8.4.4.

Maiorana-McFarland construction consists of concatenation of linear/affine S-boxes. The order of concatenation does not affect the construction. Thus in general one can consider any permutation of the functions to be concatenated. We will permute the rows of matrix D to achieve this task. The method of changing the order of concatenation is simple. We will take an array $A[1, \ldots, 2^{m-1}]$ of length 2^{m-1}. In array A the desired permutation of $\{1, \ldots, 2^{m-1}\}$ is stored. Considering the matrix D as described in Section 8.4.4 we will take $D_{A[i],j}$ instead of $D_{i,j}$ in computing the output. Thus we see that with this simple data structure using an array A of length 2^{m-1} we can get $(2^{m-1})!$ different S-boxes. Another way of doing this has been described by Nyberg [83] for Maiorana-McFarland perfect nonlinear functions. The implementation for an (n, m) S-box require one $\frac{n}{2}$ bit shift register and need m inner product computations and $(m-1)$ shifts.

8.6 Conclusion

In this chapter, we have described a method for software implementation of resilient Maiorana-McFarland S-boxes from Chapter 7. The main features of our implementation are small amount of memory and fast evaluation of the output. These features show that Maiorana-McFarland S-boxes are suitable candidates for use in software implementation of nonlinear combiner model of stream ciphers.

Bibliography

[1] K. Beauchamp. *Applications of Walsh and Related Functions.* Academic Press, 1984.

[2] C. Bennett, G. Brassard and J. Robert. Privacy Amplification by Public Discussion. *SIAM Journal of Computing*, volume 17, pages 210–229, 1988.

[3] E. R. Berlekamp. *Algebraic Coding Theory.* McGraw Hill, New York, 1968.

[4] E. R. Berlekamp and L. R. Welch. Weight distributions of the cosets of the (32, 6) Reed-Muller code. *IEEE Transactions on Information Theory*, IT-18(1):203–207, January 1972.

[5] J. Bierbrauer, K. Gopalakrishnan and D. R. Stinson. Bounds on resilient functions and orthogonal arrays. In *Advances in Cryptology – CRYPTO'94*, number 839 in Lecture Notes in Computer Science, pages 247–256. Springer Verlag, 1994.

[6] E. Biham, R. J. Anderson and L. R. Knudsen. Serpent: A New Block Cipher Proposal. In *Fast Software Encryption, FSE 1998*, pages 222–238. Springer-Verlag, 1998.

[7] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. In *Advances in Cryptology - CRYPTO'90*, Lecture Notes in Computer Science, pages 2–21. Springer-Verlag, 1991.

[8] L. Blum, M. Blum, and M. Shub. A simple unpredictable random number generator. *SIAM Journal on Computing*, 15:364–383, 1986.

[9] M. Boesgaard, M. Vesterager, T. Pedersen, J. Christiansen and O. Scavenius. Rabbit: A New High-Performance Stream Cipher. In *Fast Software Encryption, FSE 2003*, pages 307–329. Springer-Verlag, 2003.

[10] J. A. Bondy and U. S. R. Murthy. *Graph Theory with Applications.* London, Macmillan Press, 1977.

[11] E. F. Brickell, J. H. Moore, and M. R. Purtill. Structures in the S-boxes of the DES. In *Advances in Cryptology – CRYPTO'86*, Lecture Notes in Computer Science, pages 3–8. Springer-Verlag, 1987.

[12] C. Burwick, D. Coppersmith, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, S.M. Matyas, L. O'Connor, M. Peyravian, D. Safford, and N. Zunic. "MARS - A Candidate Cipher for AES," *NIST AES Proposal, Jun 98.* http://citeseer.nj.nec.com/burwick99mars.html.

[13] P. Camion and A. Canteaut. Construction of t-Resilient Functions over a Finite Alphabet. *In Advances in Cryptology - EUROCRYP 1996*, pages 283–293, Lecture Notes in Computer Science, Springer-Verlag, 1996.

[14] P. Camion, C. Carlet, P. Charpin, and N. Sendrier. On correlation immune functions. In *Advances in Cryptology – CRYPTO'91*, pages 86–100. Springer-Verlag, 1992.

[15] A. Canteaut and M. Videau. Degree of composition of highly nonlinear functions and applications to higher order differential cryptanalysis. *Advances in Cryptology – Eurocrypt 2002*, LNCS 2332, pages 518–533.

[16] A. Canteaut, C. Carlet, P. Charpin and C. Fontaine. Propagation Characteristics and Correlation-Immunity of Highly Nonlinear Boolean Functions. In *Advances in Cryptology – Eurocrypt 2000*, pages 507–522, Lecture Notes in Computer Science, Springer-Verlag, 2000.

[17] A. Canteaut and M. Trabbia. Improved fast correlation attacks using parity checks equations of weight 4 and 5. In *Advances in Cryptology – Eurocrypt 2000*, pages 573–588, Lecture Notes in Computer Science, Springer-Verlag, 2000.

[18] C. Carlet. On cryptographic propagation criteria for Boolean functions. *Information and Computation*, 151:32–56, 1999.

[19] C. Carlet and P. Guillot. A new representation of Boolean functions. *Proceedings of AAECC'13*, Lecture Notes in Computer Science, 1719, pp 94–103, 1999.

[20] C. Carlet and P. Sarkar. Spectral Domain Analysis of Correlation Immune and Resilient Boolean Functions. *Finite Fields and their Applications*, Volume 8, Number 1, January 2002, Pages 120–130.

[21] F. Chabaud, S. Vaudenay. Links Between Differential and Linear Cryptoanalysis. *In Advances in Cryptology - EUROCRYPT 1994*, pages 356–365, Lecture Notes in Computer Science, Springer-Verlag, 1995.

[22] S. Chee, S. Lee, and K. Kim. Semi-bent functions. In *Advances in Cryptology, Asiacrypt'94*, number 917 in Lecture Notes in Computer Science, pages 107–118. Springer-Verlag, 1995.

[23] S. Chee, S. Lee, D. Lee, and S. H. Sung. On the correlation immune functions and their nonlinearity. In *Advances in Cryptology, Asiacrypt 96*, number 1163 in Lecture Notes in Computer Science, pages 232–243. Springer-Verlag, 1996.

[24] J. H. Cheon. Nonlinear Vector Resilient Functions. In *Advances in Cryptology – CRYPTO 2001*, pages 458–469, Lecture Notes in Computer Science, Springer-Verlag, 2001.

[25] V. Chepyzhov and B. Smeets. On a fast correlation attack on certain stream ciphers. In *Advances in Cryptology – EUROCRYPT'91*, volume 547, pages 176–185. Springer-Verlag, 1991.

[26] V. Chepyzhov, T. Johansson and B. Smeets. A simple algorithm for fast correlation attacks on stream ciphers. In *Fast Software Encryption – FSE 2000*, pp 181 –195, Lecture Notes in Computer Science, Springer-Verlag, 2000.

[27] B. Chor, O. Goldreich, J. Hastad, J. Friedman, S. Rudich, and R. Smolensky. The bit extraction problem or t-resilient functions. In *26th IEEE Symposium on Foundations of Computer Science*, pages 396–407, 1985.

[28] N. Courtois and W. Meier. Algebraic Attacks on Stream Ciphers with Linear Feedback, *In Advances in Cryptology – EUROCRYPT 2003*, 345–359.

[29] N. Courtois and W. Meier. Algebraic Attacks on Stream Ciphers with Linear Feedback, Extended version, available at http://www.cryptosystem.net/stream/,2003.

[30] T. W. Cusick. Boolean functions satisfying higher order strict avalanche criterion. In *Advances in Cryptology – EUROCRYPT'93*, number 765 in Lecture Notes in Computer Science, pages 102–117. Springer-Verlag, 1994.

[31] J. Daemen and V. Rijmen. *The Design of Rijndael: AES* - The Advanced Encryption Standard (Information Security and Cryptography). Spriner-Verlag 2002.

[32] E. Dawson and C. K. Wu. Construction of correlation immune Boolean functions. In *Information and Communications Security*, Lecture Notes in Computer Science, pages 170–180. Springer-Verlag, 1997.

[33] W. Diffe and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(5):644–654, 1976.

[34] C. Ding, G. Xiao, and W. Shan. *The Stability Theory of Stream Ciphers.* Number 561 in Lecture Notes in Computer Science. Springer-Verlag, 1991.

[35] H. Dobbertin. Construction of bent functions and balanced Boolean functions with high nonlinearity. In *Fast Software Encryption*, number 1008 in Lecture Notes in Computer Science, pages 61–74. Springer-Verlag, 1994.

[36] H. Dobbertin, Almost Perfect Nonlinear Power Functions on $GF(2^n)$: The Welch Case. *IEEE Transactions on Information Theory*, Vol 45 , No 4, pp. 1271–1275 , 1999.

[37] P. Ekdahl and T. Johansson. A New Version of the Stream Cipher SNOW. In *Selected Areas in Cryptography, SAC 2003*, pages 47–61.

[38] N. Ferguson, D. Whiting, B. Schneier, J. Kelsey, S. Lucks and T. Kohno. Helix: Fast Encryption and Authentication in a Single Cryptographic Primitive. In *Fast Software Encryption, FSE 2003*, pages 330–346. Springer-Verlag, 2003.

[39] E. Filiol and C. Fontaine. Highly nonlinear balanced Boolean functions with a good correlation-immunity. In *Advances in Cryptology - EUROCRYPT'98.* Springer-Verlag, 1998.

[40] C. Fontaine. On some cosets of the first-order Reed-Muller code with high minimum weight. *IEEE Transactions on Information Theory*, 45(4):1237–1243, 1999.

[41] R. Forré. The strict avalanche criterion : Spectral properties of Boolean functions and an extended definition. In *Advances in Cryptology - CRYPTO'88*, Lecture Notes in Computer Science, pages 450–468. Springer-Verlag, 1990.

[42] J. Friedman. On the bit extraction problem. In *33rd IEEE Symposium on Foundations of Computer Science*, pages 314–319, 1982.

[43] J. Dj. Golic, M. Salmasizadeh, L. Simpson, and E. Dawson. Fast correlation attacks on nonlinear filter generators. *Information Processing Letters*, 64(1):37–42, 1997.

[44] S. W. Golomb. *Shift Register Sequences.* San Fransisco, CA, Holden-Day, 1967.

[45] K. Gopalakrisnan and D. R. Stinson. Three characterizations of non-binary correlation-immune and resilient functions. *Designs, Codes and Cryptography*, 5:241–251, 1995.

[46] K. C. Gupta and P. Sarkar. Improved Construction of Nonlinear S-Boxes. *In Advances in Cryptology - Asiacrypt 2002*, pp 466–483, Lecture Notes in Computer Science, Springer-Verlag, 2002.

[47] K. C. Gupta and P. Sarkar. Construction of Perfect Nonlinear and Maximally Nonlinear Multi-output Boolean Functions Satisfying Higher Order Strict Avalanche Criteria. In *Progress in Cryptology - Indoocrypt 2003*, pp 107–120, Lecture Notes in Computer Science, Springer-Verlag, 2003.

[48] K. C. Gupta and P. Sarkar. Computing Partial Walsh Transform from the Algebraic Normal Form of a Boolean Function. *Electronic Notes in Discrete Mathematics, Volume 15, http://www.elsevier.nl/gej-ng/31/29/24/75/23/show/Products/notes/index.htt.* Full version of the paper accepted for the special R. C. Bose issue of Discrete Mathematics.

[49] K. C. Gupta and P. Sarkar. A General Correlation Theorem. *Cryptology ePrint Archive: Report 2003/124, http://eprint.iacr.org/2003/124.*

[50] K. C. Gupta and P. Sarkar. Software Implementation of Resilient Maiorana-McFarland S-Boxes, preprint.

[51] K. C. Gupta and P. Sarkar. Construction of High Degree Resilient S-Boxes With Improved Nonlinearity. *Indian Statistical Institute, Technical Report No. ASD/2003/4.*

[52] S. Halevi, D. Coppersmith and C. S. Jutla. Scream: A Software-Efficient Stream Cipher. In *Fast Software Encryption - FSE 2002*, pp 195 –209, Lecture Notes in Computer Science, Springer-Verlag, 2002.

[53] R. W. Hamming. *Coding And Information Theory.* Prentice Hall Inc., 1980.

[54] C. Harpes, G. G. Kramer and J. L. Massey. A Generalization of Linear Cryptanalysis and the Applicability of Matsui's Piling-up Lemma. In *Advances in Cryptology - EUROCRYPT 1995*, pages 24–38, Lecture Notes in Computer Science, Springer-Verlag, 1995.

[55] M. Hermelin and K. Nyberg. Correlation Properties of the Bluetooth Combiner Generator. *The Second International Conference on Information Security and Cryptology of ICISC '99*, pages 17–29, Lecture Notes in Computer Science, Springer-Verlag, 2000.

[56] T. Honda, T. Satoh, T. Iwata, and K. Kurosawa. Balanced Boolean functions satisfying PC(2) and very large degree. In *SAC'97*, pages 64–72, January 1997.

[57] E. Horowitz and S. Sahni. *Fundamentals of Data Structures*, W. H. Freeman and Co., 1983.

[58] T. Jakobsen and L. R. Knudsen. The interpolation attack on block ciphers. In *SAC'97*, pages 28–40, January 1997.

[59] T. Johansson. Reduced complexity correlation attacks on twoclock-controlled generators. In *Advances in Cryptology - ASIACRYPT'98*, Lecture Notes in Computer Science. Springer-Verlag, 1998.

[60] T. Johansson. A simple algorithm for fast correlation attacks on stream ciphers. In *Fast Software Encryption'2000*, Lecture Notes in Computer Science. Springer-Verlag, 2000.

[61] T. Johansson and F. Jonsson. Fast correlation attacks based on turbo code techniques. In *Advances in Cryptology - CRYPTO'99*, number 1666 in Lecture Notes in Computer Science, pages 181–197. Springer-Verlag, August 1999.

[62] T. Johansson and F. Jonsson. Improved fast correlation attacks on stream ciphers via convolutional codes. In *Advances in Cryptology - EUROCRYPT'99*, number 1592 in Lecture Notes in Computer Science, pages 347–362. Springer-Verlag, May 1999.

[63] T. Johansson and E. Pasalic. A construction of resilient functions with high nonlinearity. *International Symposium on Information Theory*, 2000.

[64] L. R. Knudsen. Truncated and Higher Order Differentials. In *Fast Software Encryption'1995*, Lecture Notes in Computer Science, pages 196–211, Springer-Verlag, 1995.

[65] D. E. Knuth. *The Art of Computer Programming*, volume 2. Addison Wesley, 1969.

[66] I. Krasikov. On integral zeros of Krawtchouk polynomials. *Journal of Combinatorial Theory, Series A*, 74:71–99, 1996.

[67] K. Kurosawa. Almost security of cryptographic Boolean functions. Cryptology e-print archive, http://eprint.iacr.org/2003/075.

[68] K. Kurosawa and T. Satoh. Generalization of higher order SAC to vector output Boolean functions. In *Advances in Cryptology - ASIACRYPT'96*, Lecture Notes in Computer Science, pages 218–231. Springer-Verlag, 1996.

[69] K. Kurosawa and T. Satoh. Design of SAC/PC(l) of order k Boolean functions and three other cryptographic criteria. In *Advances in Cryptology - EUROCRYPT'97*, Lecture Notes in Computer Science, pages 434–449. Springer-Verlag, 1997.

[70] K. Kurosawa, T. Satoh and K. Yamamoto. Highly nonlinear t-resilient functions . Journal of Universal Computer Science, vol.3, no. 6, pp. 721–729, Springer Publishing Company, 1997.

[71] X. Lai. Higher Order Derivative and Differential Cryptanalysis. In *Communication and Cryptography'1994*, Kluwer Academic Publisher, 1994.

[72] N. Linial, Y. Mansour and N. Nisan. Constant Depth Circuits, Fourier Transform, and Learnability. *Journal of the ACM*, 40(3): 607–620 (1993).

[73] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error Correcting Codes.* North Holland, 1977.

[74] S. Maitra and P. Sarkar. Highly nonlinear resilient functions optimizing Siegenthaler's inequality. In *Advances in Cryptology - CRYPTO'99*, number 1666 in Lecture Notes in Computer Science, pages 198–215. Springer Verlag, August 1999.

[75] J. L. Massey. Shift register synthesis and bch decoding. *IEEE Transactions on Information Theory*, IT-15:122–127, January 1969.

[76] M. Matsui. Linear cryptanalysis method for DES cipher. In *Advances in Cryptology - EUROCRYPT'93*, Lecture Notes in Computer Science, pages 386–397. Springer-Verlag, 1994.

[77] W. Meier, E. Pasalic and C. Carlet. Algebraic attacks and decomposition of Boolean Functions. To be published In *Advances in Cryptology - EUROCRYPT'04*.

[78] W. Meier and O. Staffelbach. Fast correlation attack on stream ciphers. In *Advances in Cryptology - EUROCRYPT'88*, volume 330, pages 301–314. Springer-Verlag, May 1988.

[79] W. Meier and O. Staffelbach. Nonlinearity criteria for cryptographic functions. In *Advances in Cryptology – EUROCRYPT'89*, pages 549–562. Springer-Verlag, 1990.

[80] W. Meier and O. Stafflebach. Fast correlation attacks on certain stream ciphers. *Journal of Cryptology*, 1:159–176, 1989.

[81] W. Meier and O. Staffelbach. Correlation Properties of Combiners with Memory in Stream Cipher. J. Cryptology. 5 (1) (1992) 67–86.

[82] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1997.

[83] K. Nyberg. Perfect Nonlinear S-boxes. In *Advances in Cryptology – EUROCRYPT 1991*, pages 378–386, Lecture Notes in Computer Science, Springer-Verlag, 1991.

[84] K. Nyberg. Differentially uniform mapping for cryptography. In *Advances in Cryptology – EUROCRYPT 1993*, pages 55–65, Lecture Notes in Computer Science, Springer-Verlag, 1994.

[85] K. Nyberg. Linear Approximation of Block Ciphers. In *Advances in Cryptology – EUROCRYPT 1994*, pages 439–444, Lecture Notes in Computer Science, Springer-Verlag, 1995.

[86] K. Nyberg. Correlation Theorems in Cryptanalysis. Discrete Applied Mathematics 111 (2001) 177–188.

[87] S. Palit and B. K. Roy. Cryptanalysis of LFSR-encrypted codes with unknown combining functions. In *Advances in Cryptology – ASIACRYPT'99*, number 1716 in Lecture Notes in Computer Science, pages 306–320. Springer Verlag, November 1999.

[88] E. Pasalic and T. Johansson. Further results on the relation between nonlinearity and resiliency of Boolean functions. In *IMA Conference on Cryptography and Coding*, number 1746 in Lecture Notes in Computer Science, pages 35–45. Springer-Verlag, 1999.

[89] E. Pasalic and S. Maitra. Linear Codes in Generalized Construction of Resilient Functions with Very High Nonlinearity. *IEEE Transactions on Information Theory*, Vol 48, No 8, pp. 2182–2191, August 2002.

[90] N. J. Patterson and D. H. Wiedemann. The covering radius of the $(2^{15}, 16)$ Reed-Muller code is at least 16276. *IEEE Transactions on Information Theory*, IT-29(3):354–356, 1983.

[91] N. J. Patterson and D. H. Wiedemann. Correction to - the covering radius of the $(2^{15}, 16)$ Reed-Muller code is at least 16276. *IEEE Transactions on Information Theory*, IT-36(2):443, 1990.

[92] W. Penzhorn. Correlation attacks on stream ciphers : Computing low weight parity checks based on error correcting codes. In *Fast Software Encryption, FSE'96*, volume 1039, pages 159–172. Springer-Verlag, 1996.

[93] B. Preneel. Analysis and design of cryptographic hash functions, doctoral dissertation, K.U. Leuven, 1993.

[94] B. Preneel, W. Van Leekwijck, L. Van Linden, R. Govaerts, and J. Vandewalle. Propagation characteristics of Boolean functions. In *Advances in Cryptology - EUROCRYPT'90*, Lecture Notes in Computer Science, pages 161–173. Springer-Verlag, 1991.

[95] B. Preneel, R. Govaerts, and J. Vandewalle. Boolean functions satisfying higher order propagation criteria. In *Advances in Cryptology – EUROCRYPT'91*, Lecture Notes in Computer Science, pages 141–152. Springer-Verlag, 1991.

[96] B. Preneel, W. Van Leekwijck, L. Van Linden, R. Govaerts, and J. Vandewalle. Propagation characteristics of Boolean functions. In *Advances in Cryptology – EUROCRYPT'90*, Lecture Notes in Computer Science, pages 161–173. Springer-Verlag, 1991.

[97] R. L. Rivest, M. J. B. Robshaw and Y. L. Yin. RC6 as the AES. In *AES Candidate Conference 2000*, pp 337–342.

[98] G. G. Rose and P. Hawkes. Turing: A Fast Stream Cipher. In *Fast Software Encryption, FSE 2003*, pages 290–306. Springer-Verlag, 2003.

[99] O. S. Rothaus. On bent functions. *Journal of Combinatorial Theory, Series A*, 20:300–305, 1976.

[100] R. A. Rueppel. *Analysis and Design of Stream Ciphers*. Springer Verlag, 1986.

[101] R. A. Rueppel and O. J. Staffelbach. Products of linear recurring sequences with maximum complexity. *IEEE Transactions on Information Theory*, IT-33:124–131, January 1987.

[102] P. Sarkar. Hiji-bij-bij: A New Stream Cipher with a Self-Synchronizing Mode of Operation. In *Progress in Cryptology – Indoocrypt 2003*, pp 36–51, Lecture Notes in Computer Science, Springer-Verlag, 2003.

[103] P. Sarkar and S. Maitra. Construction of Nonlinear Boolean Functions with Important Cryptographic Properties. *In Advances in Cryptology – EUROCRYPT 2000*, pages 485–506, Lecture Notes in Computer Science, Springer-Verlag, 2000.

[104] B. Schneier, J. Kelsey, D. Whiting, D. Wagner and N. Ferguson. Comments on Twofish as an AES Candidate. In *AES Candidate Conference 2000*, pages 355–356.

[105] J. Seberry and X. M. Zhang. Highly nonlinear 0-1 balanced Boolean functions satisfying strict avalanche criterion. In *Advances in Cryptology, Auscrypt'92*, number 718 in Lecture Notes in Computer Science. Springer-Verlag, 1993.

[106] J. Seberry, X. M. Zhang, and Y. Zheng. Nonlinearly balanced Boolean functions and their propagation characteristics. In *Advances in Cryptology – CRYPTO'93*, pages 49–60. Springer-Verlag, 1994.

[107] J. Seberry, X. M. Zhang, and Y. Zheng. On constructions and nonlinearity of correlation immune Boolean functions. In *Advances in Cryptology – EUROCRYPT'93*, pages 181–199. Springer-Verlag, 1994.

[108] J. Seberry, X. M. Zhang, and Y. Zheng. Structures of cryptographic functions with strong avalanche characteristics. In *Advances in Cryptology, Asiacrypt 94*, number 917 in Lecture Notes in Computer Science, pages 119–132. Springer-Verlag, 1995.

[109] T. Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications. *IEEE Transactions on Information Theory*, IT-30(5):776–780, September 1984.

[110] T. Siegenthaler. Decrypting a class of stream ciphers using ciphertext only. *IEEE Transactions on Computers*, C-34(1):81–85, January 1985.

[111] O. Staffelbach and W. Meier. Cryptographic Significance of the Carry for Ciphers Based on Integer Addition. In *Advances in Cryptology – CRYPT0 1990*, pages 601–615, Lecture Notes in Computer Science, Springer-Verlag, 1991.

[112] D. R. Stinson. Resilient functions and large sets of orthogonal arrays. *Congressus Numerantium*, 92:105–110, 1993.

[113] D. R. Stinson. *Cryptography , Theory and Practice.* CRC Press, 1995.

[114] D. R. Stinson. *Cryptography , Theory and Practice, Second Edition.* CRC Press, 2002.

[115] D. R. Stinson and J. L. Massey. An infinite class of counterexamples to a conjecture concerning nonlinear resilient functions. *Journal of Cryptology*, 8(3):167–173, 1995.

[116] J Wallén. Linear Approximation of Addition Modulo 2^n. *Tenth Annual Workshop on Fast Software Encryption* , February 24–26, 2003, ÅF-Borgen, Lund, Sweden, pages 277–290, Pre-proceedings.

[117] D. Watanabe, S. Furuya, H. Yoshida, K. Takaragi and B. Preneel. A New Keystream Generator MUGI. In *Fast Software Encryption, FSE 2002*, pages 179–194. Springer-Verlag, 2002.

[118] A. F. Webster and S. E. Tavares. On the design of S-boxes. In *Advances in Cryptology - CRYPTO'85*, Lecture Notes in Computer Science, pages 523–534. Springer-Verlag, 1986.

[119] G. Xiao and J. L. Massey. A spectral characterization of correlation-immune combining functions. *IEEE Transactions on Information Theory*, pages 569–571, 1988.

[120] M. Zhang. Maximum Correlation Analysis of Nonlinear Combining Functions in Stream Ciphers. *Journal of Cryptology*, 13(3): 301–314, 2000.

[121] M. Zhang and A. H. Chan. Maximum Correlation Analysis of Nonlinear S-boxes in Stream Ciphers. In *Advances in Cryptology - CRYPTO 2000*, Lecture Notes in Computer Science, pages 501–514. Springer-Verlag, 2000.

[122] X. M. Zhang and Y. Zheng. Cryptographically resilient functions. *IEEE Transactions on Information Theory*, 43(5):1740–1747, 1997.

INDIAN STATISTICAL INSTITUTE LIBRARY

Processing slip

Source.....P/E/G date 23:12:05 Acc. no. T 159

Author. Gupta, Kisham Chand;

Title. Cryptographic and combinatorial properties of boolean functions and S-boxes

CHECKING : Call no.

1 In library/earlier ed./later ed. [✓]

2 Earlier volumes/nos. of the series in lib. [ ]

3 New title [ ]

Suggested/Approved by

Note :

23.12.05
Signature & date

CLASSIFICATION :

Class no.....x-ref.....

Scrutiny report : Signature & date

Class no.....x-ref.....

Note :

Signature & date

CATALOGUING ;

1 Main Card [ ]

2 Supp. Cards (Subject/Title/Series/Added entries/Shelf List) [Total no. of cards].....

3 Mechanical processing completed on.....

4 Cards filed [ ]

Note :

Signature & date