M.Tech Dissertation Report

on

Investigation on HN-Spectra of S-Boxes

Submitted in the fulfillment of the requirements
for the award of degree
Master of Technology
in
Computer Science and Engineering

Submitted By:

SNEHA CHAUHAN

Enrollment Number: 16535041

Under the supervision of

DR. SUGATA GANGOPADHYAY

Associate Professor, Dept. of Computer Science and Engineering

Department of Computer Science and Engineering

Indian Institute of Technology Roorkee

May, 2018

AUTHOR'S DECLARATION

I declare that the work presented in this dissertation with title **"Investigation on HN-Spectra of S-Boxes"** towards fulfillment of the requirement for the award of the degree of **Master of Technology** in **Computer Science & Engineering** submitted in the **Department of Computer Science & Engineering, Indian Institute of Technology Roorkee, India** is an authentic record of my own work carried out during the period of **May 2017 to May 2018** under the supervision of **Dr. Sugata Gangopadhyay**, Associate Professor, Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee, India. The content of this dissertation has not been submitted by me for the award of any other degree of this or any other institute.

Date: .....
Place: ROORKEE

SNEHA CHAUHAN
M.TECH CSE (16535041)
Indian Institute of Technology Roorkee

CERTIFICATE

This is to certify that the statement made by the candidate is correct to the best of my knowledge and belief.

Date: .....
Place: .....

Sign: .....
DR. SUGATA GANGOPADHYAY
(Associate Professor)
Indian Institute of Technology Roorkee

ACKNOWLEDGEMENTS

Dedicated to my family and friends, for standing by me through thick and thin, without whom I would not have gotten this far. I would like to express my sincere gratitude to my advisor Dr. Sugata Gangopadhyay for the continuous support of my study and research, for his patience, motivation, enthusiasm and immense knowledge. His guidance helped me in all time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my study. I would like to express my sincere appreciation and gratitude towards Mr. Nishant Sinha for guidance and invaluable suggestions at the time I needed the most.

I am also grateful to the Department of Computer Science and Engineering, IIT Roorkee for providing valuable resources to aid my research.

SNEHA CHAUHAN

ABSTRACT

Internet is today's buzz. It brings with it, many advantages and disadvantages. One such disadvantage is security. The communication done through the Internet must be secure so that no third party could read the information being shared by the two communicating parties. Lot of research is being done in the field of cryptology to find such functions which can have high nonlinearity so that they cannot be approximated to linear functions. In this report, some major transforms are discussed which classify the boolean functions into bent and negabent functions. An introduction on Boolean function and its use in cryptology is described. Then, the different types of transforms are explained along with their algorithms and its complexity. We have used the HN-Spectra to find the quadratic approximation for the S-Boxes. At the end, the report is summarized stating the further work still being done in this field.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Communication is a process which includes sending and receiving of information between two or more people. Information shared among the people needs to be secured from attacks. In order to secure information, information needs to be protected from unauthorized access. Modification of contents of the message should not be possible by the third party other than the sender and receiver.

From the past two decades, internet has become an important part of our life. Nowdays, all the business and communications are done through Internet. So keeping our information secure has become a major goal as with increase in technology different types of attacks are also possible. Sending and receiving information over communication channel, gives rise to the risk of eavesdropping by a third party called as adversaries.

Cryptography provides secure communication where two parties can communicate excluding the third party. Cryptography is the science of information hiding by applying complex mathematics and logic to transform the meaningful messages to meaningless text[9]. The original message to be communicated is known as plain text and the coded message which is transmitted over the communication channel is called cipher text. Encryption is the process of converting plain text to cipher text. Decryption is the process of obtaining the plain text back from cipher text. The techniques used for encryption fall into the area of cryptography and the techniques used for decrypting a cipher text constitute the area of cryptanalysis. Cryptography and Cryptanalysis together are called Cryptology[16]. At the sender's end, a plain text is converted to cipher text by applying encryption techniques. At the receiver's end, the same cipher text is decrypted back into plain text. Thus preventing the message from getting recognised by the attacker.

Modern cryptography is based on mathematical theory and computer science concepts. With increase in technology, cryptanalysis has also become common. So the design of cryptographic algorithms should be such that they are computationally hard to break. One such algorithm is Advanced Encryption standard(AES) [13] which is based on the design principle of Substitution-Permutation network.

Many properties of boolean function are used in stream and block cipher. The S-box construction is done using modular arithmetic and affine transformation. It was designed to resist linear and differential cryptanalysis. The boolean functions are used for this, so that the nonlinearity is high for resisting linear/affine approx-

imations. To prevent divide and conquer style correlation attacks, the correlation between two functions must be minimal that is their hamming distance should be almost equivalent to zero.

Chapter 2

Definitions and Notations

Let $\mathbb{F}_2$ be the finite field of order 2, defined as the set Z_2 of integers $\{0, 1\}$ together with the arithmetic operations modulo 2 [16].
Let $\mathbb{V}_n$ be a vector space of n-dimension over the 2-element field $\mathbb{F}_2$. Let $\mathbb{V}_n = \mathbb{F}_2^n$ which is expressed as[5] :-

$$\mathbb{V}_n = \mathbb{F}_2^n = \{\mathbf{x} = (x_n, \ldots, x_1) : x_i \in \mathbb{F}_2, i = 1, \ldots, n\}.$$

Let $a = (a_1, \ldots, a_n)$ and $b = (b_1, \ldots, b_n)$ are two vectors in $\mathbb{V}_n$. Their scalar product is given by $a \cdot b = a_1 b_1 \oplus \cdots \oplus a_n b_n$, where multiplication and addition $\oplus$ are over $\mathbb{F}_2$.

A Boolean Function on n variables is a function f mapped from $\mathbb{F}_2^n$ to $\mathbb{F}_2$. The symbol B_n denotes the set of all Boolean functions on n variables. A boolean function can be represented by its truth table which is a $(0, 1)$ sequence and is depicted below[3] :-

$x_n \ldots x_1$	$f(x_1, \ldots, x_n)$
$0 \ldots 0$	$\star$
$\vdots$	$\vdots$
$1 \ldots 1$	$\star$

where in the first column there are all the vectors over $\mathbb{F}_2^n$ and in the second column the resultant value of the boolean function is there. $f(x_1 \ldots x_n)$ is the vector of value of length 2^n. Thus, there will be 2^{2^n} boolean functions[19].

The weight of x is the number of x_i's that are equal to 1. It is given by :-

$$wt(x) := \sum_{i=1}^{n} x_i$$

For a function $f \in B_n$, the weight is

$$wt(f) := \sum_{x \in \mathbb{F}_2^n} f(x)$$

An affine function is a function composed of linear function added with a constant. $l_{a,c} = a \cdot x \oplus c$ is the affine function, where $a \cdot x$ is the inner product which is given as follows:-

$$a \cdot x = a_1 x_1 \oplus \ldots \oplus a_n x_n$$

It can also be written as $\langle a, x \rangle$. So,

$$l_{a,c} = (a_1 x_1 \oplus \ldots \oplus a_n x_n) + c$$

Two boolean functions are affinely equivalent if there is a non singular matrix $(n \times n)$ A and a vector B of length n such that,

$$g(y) = f(Ay \oplus B), \forall y \in \mathbb{F}_2^n$$

Similarly, extended affine function is represented by :-

$$g(y) = f(Ay \oplus B) \oplus c \cdot y \oplus \lambda, \lambda \in \mathbb{F}_2, y \in \mathbb{F}_2^n$$

2.1 Hamming Distance

The Hamming distance between two binary vectors is defined as the number of coordinates in which the two vectors differ from each other. Let x and y be the two binary vectors. Their hamming distance, $d(x, y)$ is given by :- $d(x, y) = wt(x \oplus y)$, where $wt(z)$ denotes the hamming weight of a binary vector z which is defined as the number of non-zero coordinates in the vector z[19].
Example- $x = 19$ and $y = 22$. The binary form of $x = (10011)_2$ and $y = (10110)_2$. The hamming weight of these vector are :-

$$wt(x) = 3 \text{ and } wt(y) = 3.$$

The hamming distance between vectors x and y is :-

$$d(x, y) = wt(x \oplus y)$$
$$d(x, y) = wt(10011 \oplus 10110)$$
$$d(x, y) = 2$$

Let f and g are two boolean functions. The hamming distance between f and g is the count of the positions where their vector of values differ from each other. It is denoted as follows-

$$dist(f, g) = |\{x \in \mathbb{F}_2^n : f(x) \neq g(x)\}| \tag{2.1}$$

The Hamming distance between two boolean functions F and G can also be expressed as :-

$$d_H(F, G) = \#(F \neq G)$$

where $(F \neq G)$ specifies the number of positions where F and G value doesn't match. Further expanding the above equation we get,

$$\begin{aligned} d_H(F, G) &= \#\frac{1}{2}(F = G) - \#\frac{1}{2}(F = G) + \#\frac{1}{2}(F \neq G) + \#\frac{1}{2}(F \neq G) \\ &= \frac{1}{2}(\#(F = G) + \#(F \neq G)) - \frac{1}{2}(\#(F = G) - \#(F \neq G)) \\ &= 2^{n-1} - \frac{1}{2}\sum_{x \in \mathbb{F}_2^n} (-1)^{F(x)+G(x)} \end{aligned}$$

The nonlinearity of a function f, is defined as follows

$$N_f = \min_{\phi \in A_n} d(f, \phi)$$

where A_n denotes the class of all Affine functions on F_2^n.

2.2 Vectorial Boolean Functions

Consider two positive integers n and m. A function F maps from vector space $\mathbb{F}_2^n$, consisting of all binary vectors of length n, to vector space $\mathbb{F}_2^m$, is called a (n,m)-function. Given any (n,m)-function F such that

$$F : \mathbb{F}_2^n \to \mathbb{F}_2^m$$

we can define m boolean functions $f_1, f_2, ..., f_m \epsilon B_m$ such that

$$F(x) = (f_1(x), f_2(x), ..., f_m(x))$$

for all $x \in \mathbb{F}_2^n$. Each boolean function f_i is said to be the coordinate function of F. Generally (n,m)-function are called as multioutput boolean functions, vectorial boolean functions or S-boxes, where S stands for Substitution[4]. Example of vectorial boolean function,

$$F(x) = (f_4(x), f_3(x), f_2(x), f_1(x))$$

x_4	x_3	x_2	x_1	f_4	f_3	f_2	f_1
0	0	0	0	1	1	1	0
0	0	0	1	0	1	0	0
0	0	1	0	1	1	0	1
0	0	1	1	0	0	0	1
0	1	0	0	0	0	1	0
0	1	0	1	1	1	1	1
0	1	1	0	1	0	1	1
0	1	1	1	1	0	0	0
1	0	0	0	0	0	1	1
1	0	0	1	1	0	1	0
1	0	1	0	0	1	1	0
1	0	1	1	1	1	0	0
1	1	0	0	0	1	0	1
1	1	0	1	1	0	0	1
1	1	1	0	0	0	0	0
1	1	1	1	0	1	1	1

Chapter 3

Block Cipher

Block Cipher is an Encryption and Decryption method which works on a block of plaintext and produces a block of ciphertext of equal length. It is a function that maps n-bit of plaintext blocks to n-bit of ciphertext blocks using a k-bit key. Generarlly the block size is 64 or 128 bits.

Since the block cipher takes a block of n bits of plain text as input, there can be 2^n possibilities of different plain text blocks. In order to have both encryption and decryption, each plain text block must map to a unique ciphertext block. Such transformation is called reversible mappings. Since its a n-bit block so there can be 2^n plaintext blocks and 2^n ciphertext blocks so number of transformation is $2^n!$. This is the case of ideal block cipher but it has some drawbacks such as -

1. It is easy to develop if the block size n is small but it may be vulnerable to cryptanalysis by an attacker who knows the structure of algorithm.

2. If n is large, it is infeasible to develop a ideal block cipher as key length required is $n \times 2^n$, which is very large with respect to n.

Feistel develped a structure that alternates substitution and permutation, which are defined as follows.
Substitution - Each plaintext character or block of characters is uniquely substituted or replaced by ciphertext character or block of characters.
Permutation - Permutation means rearranging or ordering of elements. A sequence of plaintext elements is replaced by a permutation of that sequence which means that element is neither added nor deleted nor replaced in the sequence, instead the order of the elements in which they appear is changed [16].

3.1 Feistel Cipher Structure

The figure 3.1 is the cipher structure proposed by Feistel. A plaintext block is given as input to the cipher. It is then divided into two halves LE_0 and RE_0. These two halves of data are processed in n rounds and then combined to give the ciphertext block. Each round i takes LE_{i-1} and RE_{i-1} as inputs which are dervied from the previous round i-1. The subkey k_i which is derived from the key K is also fed as input to the round i.

The structure of each round is same. The round function F is applied to the RE_{i-1} and the exclusive-OR is taken of the output of F and the LE_{i-1}. Thus substitution is performed on left half of data. The two halves of data are interchanged which results in permutation [16].

The decryption process with a Feistel Cipher is same as encryption, only the subkeys are used in reverse order. The right hand side of the figure 3.1 shows the decryption process.

A block cipher that uses the Feistel cipher structure is DES(Data Encryption Standard). The DES algorithm takes 64 bit plaintext and 56 bit key as input and produce 64 bit ciphertext as output. There are total 16 rounds in encrytpion and decryption algorithm of DES. The encryption and decryption of DES are similar in operation, only the key is used in reverse order for decryption. It also uses eight S-Boxes each of which takes 6 bit input to produce 4 bit output.

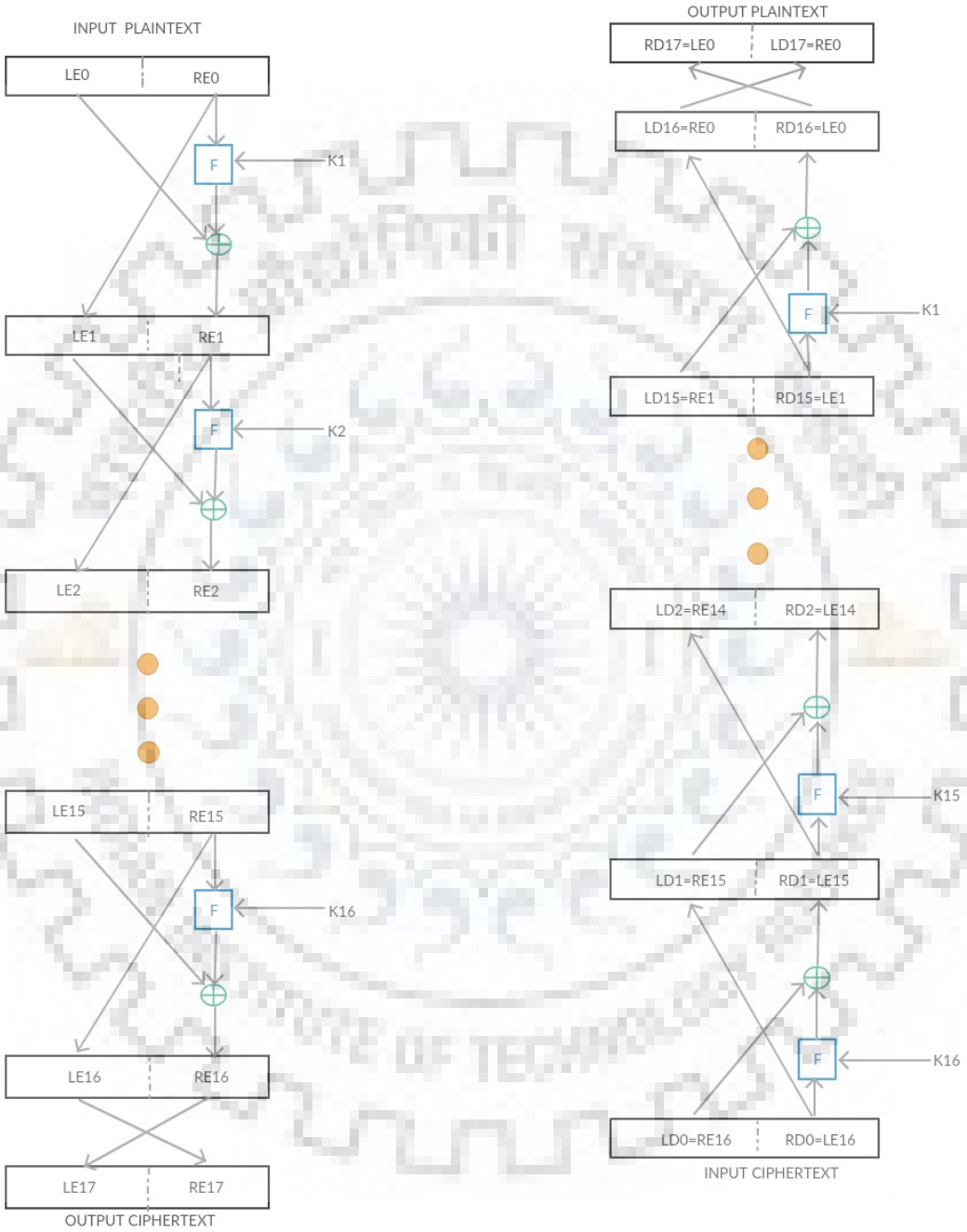


Figure 3.1: Feistel Encryption and Decryption.

3.2 Components of Modern Block Cipher

Block Ciphers are not designed as a single unit instead they are made of a combination of transposition units and substitution units which provide properties of diffusion and confusion. In diffusion, the redundancy in the statistical structure of plaintext is dissipated over the long-range statistics of ciphertext. In other words, we can say that each plaintext character must affect values of many ciphertext characters or each ciphertext character is affected by many plaintext charcter, thus a small change in plaintext wiil result in large change in ciphertext. This is acheived by permutation in which bits are transposed using transposition units known as P-Boxes. A P-Box can have n inputs and m outputs where either $n = m$, $n > m$ or $n < m$ [6, 16].

By Confusion, we mean that the relationship between the ciphertext and key should be as complex as possible so that no attacker is able to deduce the key. It is the process that changes the data drastically from input to output. This can be done by transforming the data through a non-linear table which can be created by the key. The substitution units known as S-Boxes are used to introduce confusion in the cipher. An S-Box can have n-bit input and m-bit output where n and m can be either equal or unequal. S-Box can be keyed or keyless, but most block cipher use keyless S-Boxes where the mapping of input to output is predetermined. For Example, DES uses a 6×4 S-Box.

3.3 S-Box

Earlier the block cipher were simple networks which combined substitution and permutation process to encrypt the data being transacted over an untrusted channel. In order to built strong ciphers, substitution is combined with transposition repeatedly. The strength of the cipher depends on how good is the substitution process. S-Box consists of non linear transformation which provides confusion property. So the ability of S-Box to distort the data decides how much secure is our data from attacks.

If the block cipher is linear, then linear cryptanalysis is possible using few plaintext-ciphertext pairs. In order to make block cipher complex, a non linear component is added. The nonlinearity in the block cipher is provided by the S-Boxes. The S-Boxes discussed in our report are the S-boxes used in PRESENT, SERPENT and Quasi Group ciphers.

3.3.1 Different S-Boxes used

S-Box are those vectorial boolean function that perform substitution to make the relationship between ciphertext and symmetric key more complex. The role of S-Box is to provide confusion. S-Boxes play a key role in all modern block cipher as it is the only non-linear part of the block cipher.

An S-Box takes n-bit input to give m-bit output. S-box consists of m boolean functions, each of which take n-bit input to generate a output.

We can write S-Box as a function F from $\mathbb{F}_2^n \to \mathbb{F}_2^m$, defined as follows :

$$F(x) = (f_1(x), f_2(x), \ldots, f_m(x))$$

where $x \in \mathbb{F}_2^n$ and $f_i : \mathbb{F}_2^n \to \mathbb{F}_2$, for all $i = 1, \ldots, m$. The function $f_i's$ are called Coordinate Function of F. Let us define inner product on $\mathbb{F}_2^n$ as

$$x \cdot y = (x_1, \ldots, x_n) \cdot (y_1, \ldots, y_n) = x_1y_1 + \cdots + x_ny_n$$

where all the sum are over $\mathbb{F}_2^n$.
Let $v \in \mathbb{F}_2^m$. So $v = (v_1, \ldots, v_m)$ when $v_i \in \mathbb{F}_2$ for all $i = 1, \ldots, m$.

$$\begin{aligned} v \cdot F(x) &= (v_1, \ldots, v_m) \cdot (f_1(x), f_2(x), \ldots, f_m(x)) \\ &= v_1f_1(x) + v_2f_2(x) + \ldots + v_mf_m(x) \end{aligned}$$

The Boolean Function $v \cdot F(x)$ for all $v \in \mathbb{F}_2^m$ are called Component Function of F. So all coordinate functions are component function but all component functions are not coordinate functions of a vector boolean function.
For example, if there are three coordinate functions f_1, f_2 and f_3, then its component functions will be : 0 function, f_1, f_2, f_3, $f_1 \oplus f_2$, $f_2 \oplus f_3$, $f_1 \oplus f_3$ and $f_1 \oplus f_2 \oplus f_3$. So in order to have a strong S-Box, its component functions have to be strong.

- **PRESENT** - PRESENT is a cipher based on Substitution-Permutation network. It consists of 31 rounds and each round comprises of XOR operation of the plaintext with the round key, a linear bitwise permutation and a non-linear substitution layer[2]. The permutation and substitution are performed using P-Box and S-Box respectively. The PRESENT cipher takes as input a block of 64 bits and a key of size 80 or 128 bits.

  The 4 bit S-Box is used in the PRESENT cipher. The S-Box is a function defined as :-

  $$S : \mathbb{F}_2^4 \to \mathbb{F}_2^4$$

  The S-Box is parallely applied 16 times to the input in each round. The S-Box used in PRESENT is as follows.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

- **SERPENT** - The SERPENT is a block cipher based on Substitution-Permutation network which operates on four 32 bit words that is a block of 128 bits and a key of size 128 bits. It consists of the following steps:
  - An initial permutation
  - There are 32 rounds in SERPENT cipher where each round consists of text exclusive-or'ed with the key, substitution performed by using 8 S-Boxes and a linear transformation.
  - A final permutation.

S-Box	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	3	8	F	1	A	6	5	B	E	D	4	2	7	0	9	C
1	F	C	2	7	9	0	5	A	1	B	E	8	6	D	3	4
2	8	6	7	9	3	C	A	F	D	1	E	4	0	B	5	2
3	0	F	B	8	C	9	6	3	D	1	2	4	A	7	5	E
4	1	F	8	3	C	0	B	6	2	5	4	A	9	E	7	D
5	F	5	2	B	4	A	9	C	0	3	E	8	D	6	7	1
6	7	2	C	5	8	4	6	B	E	9	1	F	D	3	A	0
7	1	D	F	0	E	8	2	B	7	4	C	A	9	3	5	6

Figure 3.2: SERPENT S-Box

The 32 rounds in SERPENT use 8 S-Boxes where each S-Box is used in 4 rounds and in each round, it is applied 32 times in parallel. The S-Box consists of a function that maps four input bits to four output bits. The S-Boxes used here are given in figure 3.2[1, 15]:

- **Quasigroup** - Consider $(Q, *)$ as a finite binary groupoid where * denotes one binary operation on non-empty set Q and $a, b \in Q$.

  Definition 3.3.1. A finite binary groupoid $(Q, *)$ is called a quasigroup if for all ordered pairs $a, b \in Q^2$ there exist unique solutions $x, y \in Q$ to the equations $x * a = b$ and $a * y = b$.

  Consider a finite set Q which is denoted by $Q^+ = \{a_1 a_2 \dots a_n | a_i \in Q\}$, a set of finite strings formed by elements of Q. Consider $(Q, *)$ be a quasigroup. Two functions are defined as follows for each $a \in Q$.

$$e_{a,*}, d_{a,*} : Q^+ \to Q^+$$

$$e_{a,*}(\alpha) = b_1 b_2 \dots b_n \Leftrightarrow b_1 = a * a_1, b_2 = b_1 * a_2, \dots, b_n = b_{n-1} * a_n$$

and

$$d_{a,*}(\alpha) = c_1 c_2 \dots c_n \Leftrightarrow c_1 = a * a_1, c_2 = a_1 * a_2, \dots, c_n = a_{n-1} * a_n$$

The functions defined above are called e- and d- transformation of Q^+ respectively based on the operation * with leader being a. These transformation can be represented graphically as [10]:

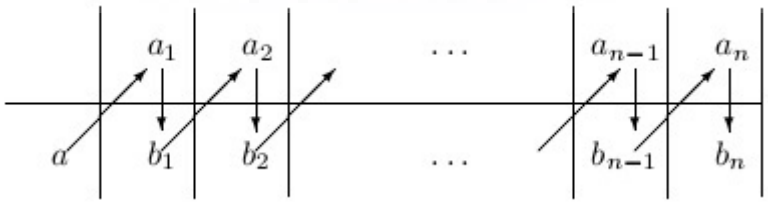


Figure 3.3: Graphical Representation of $e_{a,*}$ function

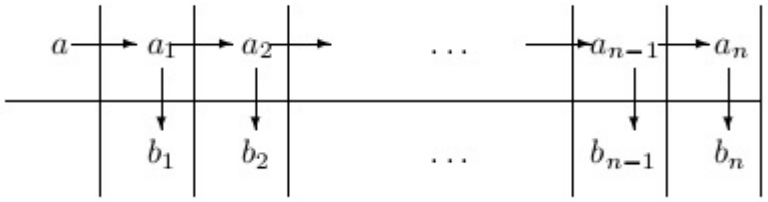


Figure 3.4: Graphical Representation of $d_{a,*}$ function

Construction of Quasi S-Box :- It has been proven that any inversion mapping in n dimension where n is even in Galois Field (2^n) should have algebraic degree less than n-1. Also a Good S-Box should have highest possible algebaric degree. So the 4×4 S-Boxes must have algebraic degree 3. The algorithm for constructing Q-S-Boxes is as follows [11]:

1. Take one quasigroup of order 4 from the class of non-linear.
2. Input the number of rounds.
3. Input the leaders. Number of leader is same as number of rounds.
4. Generate all possible input blocks of 4 bits in the lexicographic order.
5. Take input blocks one by one and for each of them do the following :
   (a) Apply e-transformation with leader a in the input block.
   (b) Reverse the result from above and apply e-transformation with other leader a again.
   (c) Continue this as many times as the number of rounds.
   (d) Save the 4bit result from the last round.
6. Concatenate all the saved result at the end to generate 16 bit permutation or 4×4 bit Q-S-Box.
7. Verify whether S-Box satisfies predetermined criteria.
   (a) If Q-S-Box satisfies the criteria, then it is a optimal S-Box
   (b) If not, go to step 3.
8. Analyze the optimal set of Q-S-boxes that are obtained.

The quasigroup S-Boxes are given in figure 3.5 :

S-Box	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	5	C	A	9	E	8	B	3	2	6	D	7	1	4	0	F
1	1	4	A	5	3	7	2	8	D	B	6	E	9	0	C	F
2	7	F	3	B	0	6	C	E	D	2	8	9	A	5	4	1
3	F	D	7	1	A	2	E	6	0	5	4	B	8	9	3	C
4	4	B	A	F	3	C	6	7	E	8	2	0	9	1	D	5
5	6	7	D	2	E	B	A	5	4	C	0	8	1	3	9	F
6	0	E	F	3	2	6	5	7	4	D	A	1	C	9	8	B

Figure 3.5: Quasi S-Boxes

Chapter 4

Walsh-Hadamard Transform

The Walsh-Hadamard Tranform[5, 19] of a funtion f on $\mathbb{F}_2^n$ is a map from $\mathbb{F}_2^n$ to Real numbers defined as

$$W_f(u) : \mathbb{F}_2^n \to \mathbb{R}$$
$$W_f(u) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) \oplus u \cdot x} \tag{4.1}$$

The numbers $W_f(u)$ are called Walsh-Hadamard coefficients of a boolean function f. For example, consider $f(x_1, x_2) = x_1 x_2$.
Using the equation 4.1, we find the Walsh-Hadamard coefficients.

$$\begin{aligned} W_f(00) &= (-1)^{f(00) \oplus \langle 00,00 \rangle} + (-1)^{f(01) \oplus \langle 00,01 \rangle} + (-1)^{f(10) \oplus \langle 00,10 \rangle} + (-1)^{f(11) \oplus \langle 00,11 \rangle} \\ &= (-1)^{0 \oplus \langle 00,00 \rangle} + (-1)^{0 \oplus \langle 00,01 \rangle} + (-1)^{0 \oplus \langle 00,10 \rangle} + (-1)^{1 \oplus \langle 00,11 \rangle} \\ &= (-1)^{0 \oplus 0} + (-1)^{0 \oplus 0} + (-1)^{0 \oplus 0} + (-1)^{1 \oplus 0} \\ &= (-1)^0 + (-1)^0 + (-1)^0 + (-1)^1 \\ &= 2 \end{aligned}$$

$$\begin{aligned} W_f(01) &= (-1)^{f(00) \oplus \langle 01,00 \rangle} + (-1)^{f(01) \oplus \langle 01,01 \rangle} + (-1)^{f(10) \oplus \langle 01,10 \rangle} + (-1)^{f(11) \oplus \langle 01,11 \rangle} \\ &= (-1)^{0 \oplus \langle 01,00 \rangle} + (-1)^{0 \oplus \langle 01,01 \rangle} + (-1)^{0 \oplus \langle 01,10 \rangle} + (-1)^{1 \oplus \langle 01,11 \rangle} \\ &= (-1)^{0 \oplus 0} + (-1)^{0 \oplus 1} + (-1)^{0 \oplus 0} + (-1)^{1 \oplus 1} \\ &= (-1)^0 + (-1)^1 + (-1)^0 + (-1)^0 \\ &= 2 \end{aligned}$$

$$\begin{aligned} W_f(10) &= (-1)^{f(00) \oplus \langle 10,00 \rangle} + (-1)^{f(01) \oplus \langle 10,01 \rangle >} + (-1)^{f(10) \oplus \langle 10,10 \rangle} + (-1)^{f(11) \oplus \langle 10,11 \rangle} \\ &= (-1)^{0 \oplus \langle 10,00 \rangle} + (-1)^{0 \oplus \langle 10,01 \rangle} + (-1)^{0 \oplus \langle 10,10 \rangle} + (-1)^{1 \oplus \langle 10,11 \rangle} \\ &= (-1)^{0 \oplus 0} + (-1)^{0 \oplus 0} + (-1)^{0 \oplus 1} + (-1)^{1 \oplus 1} \\ &= (-1)^0 + (-1)^0 + (-1)^1 + (-1)^0 \\ &= 2 \end{aligned}$$

$$
\begin{aligned}
W_f(11) &= (-1)^{f(00)\oplus\langle 11,00\rangle} + (-1)^{f(01)\oplus\langle 11,01\rangle} + (-1)^{f(10)\oplus\langle 11,10\rangle} + (-1)^{f(11)\oplus\langle 11,11\rangle} \\
&= (-1)^{0\oplus\langle 11,00\rangle} + (-1)^{0\oplus\langle 11,01\rangle} + (-1)^{0\oplus\langle 11,10\rangle} + (-1)^{1\oplus\langle 11,11\rangle} \\
&= (-1)^{0\oplus 0} + (-1)^{0\oplus 1} + (-1)^{0\oplus 1} + (-1)^{1\oplus 0} \\
&= (-1)^0 + (-1)^1 + (-1)^1 + (-1)^1 \\
&= -2
\end{aligned}
$$

From all the above 4 equations, we get a set of values $\{2, 2, 2, -2\}$. This set is the Walsh-Hadamard spectrum of *f*. So, the ordered multiset is

$$W_f = \{\, W_f(x) : x \in \mathbb{F}_2^n \}$$

where vector x takes value from $\mathbb{F}_2^n$ in lexicographical order, is the Walsh-Hadamard Spectrum of *f*.
If $u \in \mathbb{F}_2^n$ then following result is obtained.

$$\sum_{x\in\mathbb{F}_2^n} (-1)^{u.x} = \begin{cases} 0, & \text{if } u \neq 0 \\ 2^n, & \text{if } u = 0 \end{cases} \tag{4.2}$$

Consider a function $f : \mathbb{F}_2^n \to \mathbb{F}_2$, then its Walsh Transform will be

$$W_f(u) = \sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)+u\cdot x} \tag{4.3}$$

Now,

$$
\begin{aligned}
\sum_{u\in\mathbb{F}_2^n} W_f(u)(-1)^{z\cdot u} &= \sum_{u\in\mathbb{F}_2^n}\sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)+u\cdot x+z\cdot x} \\
&= \sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)} \sum_{u\in\mathbb{F}_2^n} (-1)^{u\cdot(x+z)} \qquad (4.4) \\
&= 2^n (-1)^{f(z)}
\end{aligned}
$$

In equation (4.4), assume $x = z$ so that $u \cdot (x + z) = 0$ and using result of equation (4.2).

$$(-1)^{f(z)} = 2^{-n} \sum_{u\in\mathbb{F}_2^n} W_f(u)(-1)^{z\cdot u} \tag{4.5}$$

The equation (4.5) gives the value of f(x) if the Walsh-Hadamard transform value is known.
Again consider the equation (4.3). Let a linear function $l_u(x)$ be written as

$$l_u(x) = u \cdot x = \sum_{i=1}^{n} u_i x_i \tag{4.6}$$

We can use equation (4.6) in equation (4.3).Thus, the Walsh Transform will be like

$$
\begin{aligned}
W_f(u) &= \sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)+l_u(x)} \\
&= \#(f = l_u) - \#(f \neq l_u)
\end{aligned} \tag{4.7}
$$

The above equation (4.7) gives the correlation between f and l_u. So we can say that the Walsh-Hadamard Transform at u of f is the correlation of f with l_u. Suppose f and g are two boolean functions in n variables, then the correlation between f and g is given by

$$C_{f,g}(0) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+g(x)}$$

CROSS-CORRELATION Again consider the above two functions, the cross correlation between f and g is given as follows

$$C_{f,g}(u) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+g(x+u)}$$

4.1 Parseval's Identity

Using equation (4.3)

$$W_f(u) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+u \cdot x}$$

$$\begin{aligned} W_f(u)^2 &= \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+u \cdot x} \sum_{y \in \mathbb{F}_2^n} (-1)^{f(y)+u \cdot y} \\ &= \sum_{x \in \mathbb{F}_2^n} \sum_{y \in \mathbb{F}_2^n} (-1)^{f(x)+f(y)+u \cdot (x+y)} \end{aligned}$$

$$\begin{aligned} \sum_{u \in \mathbb{F}_2^n} W_f(u)^2 (-1)^{z \cdot u} &= \sum_{u \in \mathbb{F}_2^n} \sum_{x \in \mathbb{F}_2^n} \sum_{y \in \mathbb{F}_2^n} (-1)^{f(x)+f(y)+u \cdot (x+y+z)} \\ &= \sum_{x \in \mathbb{F}_2^n} \sum_{y \in \mathbb{F}_2^n} (-1)^{f(x)+f(y)} \sum_{u \in \mathbb{F}_2^n} (-1)^{u \cdot (x+y+z)} \end{aligned} \tag{4.8}$$

To use the result of equation (4.2) in equation (4.8), assume that $(x + y + z) = 0$, which means $y = x + z$. Thus we get,

$$\sum_{u \in \mathbb{F}_2^n} W_f(u)^2 (-1)^{z \cdot u} = 2^n \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+f(x+z)}$$

Substitute $z = 0$ in above equation.

$$\begin{aligned} \sum_{u \in \mathbb{F}_2^n} W_f(u)^2 &= 2^n \sum_{x \in \mathbb{F}_2^n} (-1)^{2f(x)} \\ &= 2^n \sum_{x \in \mathbb{F}_2^n} 1 \\ &= 2^{2n} \end{aligned}$$

The Parseval's eqaution[5] is given by -

$$\sum_{u \in \mathbb{F}_2^n} W_f(u)^2 = 2^{2n} \tag{4.9}$$

If all the coefficients of Walsh Hadamard Transform are equal then the equation (4.9) becomes

$$W_f(u)^2 = 2^n$$

Thus we get the result as,

$$W_f(u) = \pm 2^{n/2} \tag{4.10}$$

If n is even in equation (4.10) then the function f is said to be the bent function. Bent Functions are those boolean functions on n number of variables where n is even,that are maximally distanced from the set of affine functions.

4.2 Hadamard Matrices

A Hadamard matrix H of order n is an $n \times n$ matrix of $\pm 1s$ such that

$$HH^t = nI_n \tag{4.11}$$

where H^t is the transpose of H and I_n is the $n \times n$ Identity matrix. The Hadamard matrices are given as[5]

$$H_0 = \begin{pmatrix}1\end{pmatrix} H_1 = \begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix} H_2 = \begin{pmatrix}1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\end{pmatrix}$$

$$H_n = \begin{pmatrix}H_{n-1} & H_{n-1}\\ H_{n-1} & -H_{n-1}\end{pmatrix}$$

The Walsh Hadamard Transform can be expresses in terms of hadamard matrix as follows :

$$W_f = H_n f$$

Solving the Walsh Transform given in equation (4.1) by assuming $u = u_1 u_2$,

$$\begin{aligned} W_f(u) &= \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+u\cdot x} \\ &= (-1)^{f(00)+(u_1u_2)\cdot(00)} + (-1)^{f(01)+(u_1u_2)\cdot(01)} + (-1)^{f(10)+(u_1u_2)\cdot(10)} + (-1)^{f(11)+(u_1u_2)\cdot(11)} \\ &= (-1)^0(-1)^{f(00)} + (-1)^{u_2}(-1)^{f(01)} + (-1)^{u_1}(-1)^{f(10)} + (-1)^{u_1+u_2}(-1)^{f(11)} \end{aligned}$$

$$\begin{aligned} W_f(00) &= (-1)^{f(00)} + (-1)^{f(01)} + (-1)^{f(10)} + (-1)^{f(11)} \\ W_f(01) &= (-1)^{f(00)} + (-1)(-1)^{f(01)} + (-1)^{f(10)} + (-1)(-1)^{f(11)} \\ W_f(10) &= (-1)^{f(00)} + (-1)^{f(01)} + (-1)(-1)^{f(10)} + (-1)(-1)^{f(11)} \\ W_f(11) &= (-1)^{f(00)} + (-1)(-1)^{f(01)} + (-1)(-1)^{f(10)} + (-1)^{f(11)} \end{aligned}$$

In matrix form, the above equations can be represented as :-

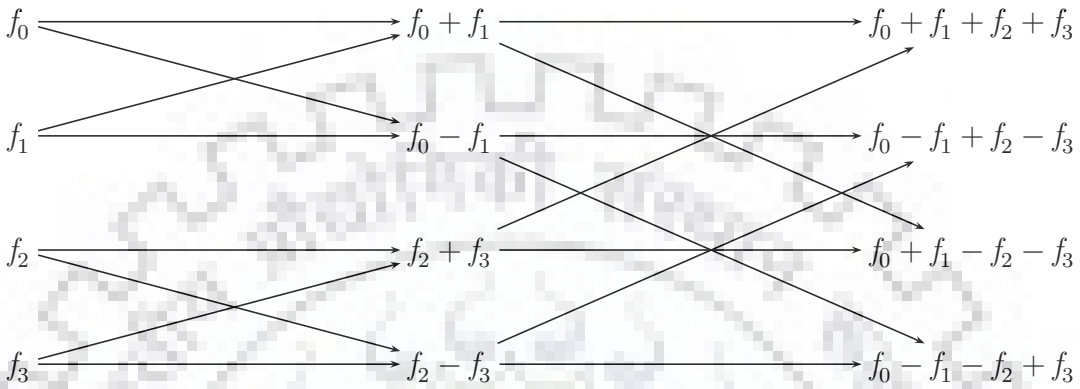
$$W_f = \begin{bmatrix}1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\end{bmatrix} \begin{bmatrix}(-1)^{f(00)}\\ (-1)^{f(01)}\\ (-1)^{f(10)}\\ (-1)^{f(11)}\end{bmatrix}$$

A boolean function, say f, will be a bent function if $|H_f(u)| = 1$. If the above matrix is solved using matrix multiplication method then it will take $O(2^{2n})$ time. There is another faster way of calculating the Walsh coefficients which is explained in the next section.

4.3 Fast Walsh Transform

Fast Walsh transform is the method of calculating the Walsh coefficients by performing $n2^n$ operations[5]. This method is depicted by the figure given below. In the figure following notations are used :

$f_0 = (-1)^{f(00)}$, $f_1 = (-1)^{f(01)}$, $f_2 = (-1)^{f(10)}$ and $f_3 = (-1)^{f(11)}$.



Chapter 5

Nega-Hadamarad Transform

$\mathbb{C}$ denotes the set of complex numbers. Let $z = a + b\iota$. $|z| = \sqrt{a^2 + b^2}$ gives the absolute value of z. The complex conjugate of z is given by $\bar{z} = a - \iota b$. Here $\iota^2 = -1$. The Nega-Hadamard transform of a function f at vector u is given as follows(as given in [17]) :-

$$N_f(u) = 2^{-n/2} \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) \oplus u \cdot x} \iota^{wt(x)}$$

The Nega-Hadamard can be represented as a matrix, shown below :-

$$N = \begin{pmatrix} 1 & \iota \\ 1 & -\iota \end{pmatrix}$$

Rewriting the Nega-Hadamard Transform (ignoring the constant),

$$N_f(u) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) \oplus u \cdot x} \iota^{wt(x)}$$

Assume vector $u = u_1 u_2$ and compute the transform.

$$\begin{aligned} N_f(u) &= (-1)^{f(00)+(u_1u_2)\cdot(00)} \iota^{wt(00)} + (-1)^{f(01)+(u_1u_2)\cdot(01)} \iota^{wt(01)} + (-1)^{f(10)+(u_1u_2)\cdot(10)} \iota^{wt(10)} + \\ &\quad (-1)^{f(11)+(u_1u_2)\cdot(11)} \iota^{wt(11)} \\ &= (-1)^{f(00)} \iota^0 + (-1)^{f(01)} (-1)^{u_2} \iota^1 + (-1)^{f(10)} (-1)^{u_1} \iota^1 + (-1)^{f(11)} (-1)^{(u_1+u_2)} \iota^2 \\ &= (-1)^{f(00)} + (-1)^{f(01)} (-1)^{u_2} \iota + (-1)^{f(10)} (-1)^{u_1} \iota + (-1)(-1)^{f(11)} (-1)^{(u_1+u_2)} \end{aligned}$$

$$\begin{aligned} N_f(00) &= (-1)^{f(00)} + (-1)^{f(01)} (-1)^0 \iota + (-1)^{f(10)} (-1)^0 \iota - (-1)^{f(11)} (-1)^0 \\ &= (-1)^{f(00)} + (-1)^{f(01)} \iota + (-1)^{f(10)} \iota - (-1)^{f(11)} \\ N_f(01) &= (-1)^{f(00)} + (-1)^{f(01)} (-1)^1 \iota + (-1)^{f(10)} (-1)^0 \iota - (-1)^{f(11)} (-1)^1 \\ &= (-1)^{f(00)} - (-1)^{f(01)} \iota + (-1)^{f(10)} \iota + (-1)^{f(11)} \\ N_f(10) &= (-1)^{f(00)} + (-1)^{f(01)} (-1)^0 \iota + (-1)^{f(10)} (-1)^1 \iota - (-1)^{f(11)} (-1)^1 \\ &= (-1)^{f(00)} + (-1)^{f(01)} \iota - (-1)^{f(10)} \iota + (-1)^{f(11)} \\ N_f(11) &= (-1)^{f(00)} + (-1)^{f(01)} (-1)^1 \iota + (-1)^{f(10)} (-1)^1 \iota - (-1)^{f(11)} (-1)^0 \\ &= (-1)^{f(00)} - (-1)^{f(01)} \iota - (-1)^{f(10)} \iota - (-1)^{f(11)} \end{aligned}$$

The above equations can be represented in the form of two matrices,where the multiplication of both the matrix will give Nega-Hadamard coefficients.

$$N_f = \begin{bmatrix} 1 & \iota & \iota & -1 \\ 1 & -\iota & \iota & 1 \\ 1 & \iota & -\iota & 1 \\ 1 & -\iota & -\iota & -1 \end{bmatrix} \begin{bmatrix} (-1)^{f(00)} \\ (-1)^{f(01)} \\ (-1)^{f(10)} \\ (-1)^{f(11)} \end{bmatrix}$$

The first matrix is the result of the tensor (or Kronecker) product of the Nega-hadamard matrix of order 2. If $|N_f(u)| = 1$ for a boolean function f then it is said to be a negabent function[8, 18]. The inverse of Nega-hadamard Transform is given as:

$$(-1)^{f(y)} = 2^{-n/2} \iota^{wt(y)} \sum_{y \in \mathbb{F}_2^n} N_f(u)(-1)^{u.y}$$

Chapter 6

HN Transform

In[7],the authors have explained the HN-transform and its application in quantum computation. The Hadamard kernel and Nega-Hadamard Kernel are denoted as follows:

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, N = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & \iota \\ 1 & -\iota \end{pmatrix}$$

These are the unitary transforms and the tensor product of H and N, n times in any permuted sequence results in 2^n sets which are denoted by set $\{H, N\}^n$.

$$\{H, N\}^n = \{K_n \otimes \ldots \otimes K_1 : K_i \in \{H, N\}, i \in 1 \ldots n\}$$

DEFINITION Let $f \in B_n$. Suppose $c = (c_n \ldots c_1) \in \mathbb{F}_2^n$ and $K^c \in \{H, N\}^n$ is such that $K^c = K_n \otimes \ldots \otimes K_1 = \bigotimes_{i=n}^{1} K_i$ where

$$K_i = \begin{cases} H, & \text{if } c_i = 0 \\ N, & \text{if } c_i = 1 \end{cases}$$

For $0 \leq j \leq 2^n - 1$, the following equation is defined,

$$K_f^c(u_j) = 2^{-n/2} \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) \oplus u_j . x} \iota^{wt(c * x)} \tag{6.1}$$

which refers to the HN-Transform of function f at u_j with respect to the combination K^c. For the entire set $\{H, N\}^n$, it gives values called HN-spectrum.

To compute HN spectra, $n2^{2n}$ operations are required. In the paper [7], **Fast *HN* Transform Algorithm** is introduced which reduces the complexity of HN spectra computation to $O(2^{2n})$. The proof of this complexity is explained in the paper is mentioned here- x,v and c are vectors of length n which can be divided into 2 parts- $n-1$ and 1.

$$f(x, y) = (y \oplus 1) f_1(x) \oplus y f_2(x)$$

For every $(v, u), (c_n, c) \in \mathbb{F}_2 \times \mathbb{F}_2^{n-1}$,

$$2^{-n/2} K_f^{(c_n, c)}(v, u) = \sum_{x \in \mathbb{F}_2^{n-1}} (-1)^{u \cdot x \oplus f_1(x)} \iota^{wt(c * x)} + (-1)^v \iota^{wt(c_n)} \sum_{x \in \mathbb{F}_2^{n-1}} (-1)^{u \cdot x \oplus f_2(x)} \iota^{wt(c * x)}$$

Let the time complexity to compute all the values of *HN*-spectrum for any boolean function on n variables be $T(n)$.
To compute

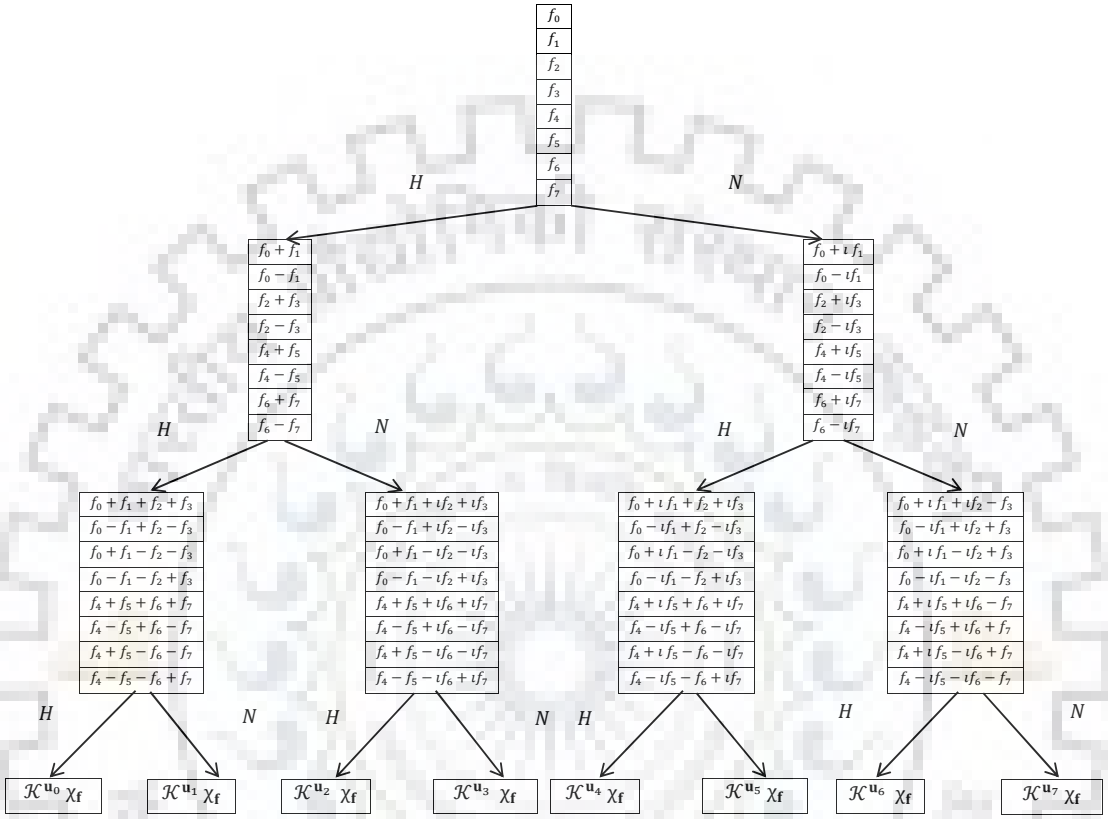
$$2^{-n/2}K^c(u) = \sum_{x\in\mathbb{F}_2^{n-1}} (-1)^{f_1(x)\oplus u\cdot x} \iota^{wt(c*x)}$$

$$2^{-n/2}K^c(u) = \sum_{x\in\mathbb{F}_2^{n-1}} (-1)^{f_2(x)\oplus u\cdot x} \iota^{wt(c*x)}$$

where $u, c \in \mathbb{F}_2^{n-1}$, $2T(n-1)$ time will be required.
Keeping $c \in \mathbb{F}_2^{n-1}$ fixed and varying $c_n \in \mathbb{F}_2$ and $(v, u) \in \mathbb{F}_2 \times \mathbb{F}_2^{n-1}$ then it requires $4.2^{n-1} = 2^{n+1}$ time. And if c is also varied then total time required will be $2^{n-1}.2^{n+1} = 2^{2n}$. Thus the total time to compute the *HN*-spectra is given by the recurrence relation as

$$T(n) = 2T(n-1) + 2^{2n}$$

which gives $T(n) = 2^{2n}$. The computation of *HN*-transform is represented by the tree.



$\chi_{\mathbf{f}} = (f_0, f_1, f_2, f_3, f_4, f_5, f_6, f_7)^T$. The vector $\mathbf{u}_j = (u_{j2}, u_{j1}, u_{j0})$ corresponds to the binary representation of $0 \leq j \leq 7$.

$\mathcal{K}^{\mathbf{u}_j}\chi_{\mathbf{f}} = K_2 \otimes K_1 \otimes K_0\,\chi_{\mathbf{f}}$ where $K_i = H$, if $u_{ji} = 0$ and $K_i = N$, if $u_{ji} = 1$, for all $0 \leq i \leq 2$. The normalizing factors of $\frac{1}{\sqrt{2}}$ and $\frac{1}{2}$ for the first and second levels, respectively, are not shown.

6.1 HN-Transform and Quadratic Functions

In the paper[7], relationship between HN-Transform and quadratic approximations of boolean functions has been discussed. Let $c \in \mathbb{F}_2^n$ be a vector. Then its (n-1)-dimensional orthogonal subspace is given as $c^\perp = \{x \in \mathbb{F}_2^n : c \cdot x = 0\}$. Consider a symmetric quadratic bent function $s \in B_n$. It is defined by $s(x) = \bigoplus_{i<j} x_i x_j$, for all $x \in \mathbb{F}_2^n$. For each $c \in \mathbb{F}_2^n$, function $s_c \epsilon B_n$ is defined by $s_c(x) = s(c * x)$. $s_c's$ are considered as quadratic symmetric functions on variables $x_i's$ having $c_i = 1$. Using $wt(c * x) = 2s_c(x) + (c \cdot x)(mod 4)$, we get from 6.1,

$$2^{\frac{n}{2}} K_f^c(u) = \sum_{x \in c^\perp} (-1)^{f(x) \oplus s_c(x) \oplus u \cdot x} + \iota \sum_{x \in \mathbb{F}_2^n \setminus c^\perp} (-1)^{f(x) \oplus s_c(x) \oplus u \cdot x}$$

Let $f \in B_n$ such that,

$$\Big| \sum_{x \in c^\perp} (-1)^{f(x) \oplus s_c(x) \oplus u \cdot x} \Big| = (-1)^{\epsilon_1(u,c)} \sum_{x \in c^\perp} (-1)^{f(x) \oplus s_c(x) \oplus u \cdot x}$$

$$\Big| \sum_{x \in \mathbb{F}_2^n \setminus c^\perp} (-1)^{f(x) \oplus s_c(x) \oplus u \cdot x} \Big| = (-1)^{\epsilon_2(u,c)} \sum_{x \in \mathbb{F}_2^n \setminus c^\perp} (-1)^{f(x) \oplus s_c(x) \oplus u \cdot x}$$

where $\epsilon_1(u,c), \epsilon_2(u,c) \in \mathbb{F}_2$ and $u, c \in \mathbb{F}_2^n$. Combining above two,

$$\begin{aligned} &\sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) \oplus s_c(x) \oplus \epsilon_1(u,c) \oplus (\epsilon_1(u,c) \oplus \epsilon_2(u,c)) c \cdot x \oplus u \cdot x} \\ &= \Big| \sum_{x \in c^\perp} (-1)^{f(x) \oplus s_c(x) \oplus u \cdot x} \Big| + \Big| \sum_{x \in \mathbb{F}_2^n \setminus c^\perp} (-1)^{f(x) \oplus s_c(x) \oplus u \cdot x} \Big| \\ &= |\Re(2^{\frac{n}{2}} K_f^c(u))| + |\Im(2^{\frac{n}{2}} K_f^c(u))| \end{aligned}$$

The Hamming distance between the functions f and $s_c \oplus \epsilon_1(u,c) \oplus (\epsilon_1(u,c) \oplus \epsilon_1(u,c)) c \cdot x \oplus u \cdot x$ is given by

$$2^{n-1} - \frac{1}{2}(|\Re(2^{\frac{n}{2}} K_f^c(u))| + |\Im(2^{\frac{n}{2}} K_f^c(u))|) \tag{6.2}$$

The spectra obtained by computing HN-spectra for any boolean function $f \in B_n$ is given as :

$$[|\Re(2^{\frac{n}{2}} K_f^c(u))| + |\Im(2^{\frac{n}{2}} K_f^c(u))| : u \in \mathbb{F}_2^n]$$

Thus, we can get following result

$$\max_{c \in \mathbb{F}_2^n} \max_{u \in \mathbb{F}_2^n} [|\Re(2^{\frac{n}{2}} K_f^c(u))| + |\Im(2^{\frac{n}{2}} K_f^c(u))| : u \in \mathbb{F}_2^n] \tag{6.3}$$

Thus, the equation 6.3 gives the best quadratic approximation possible for function f.

Chapter 7

Experiment with HN-Transform and Result

Here, the description of the approach used to get the quadratic approximation is explained. The S-Boxes of the PRESENT, SERPENT cipher and Quasigroup are used which have been defined in the previous sections. The PRESENT cipher uses only one S-Box, the SERPENT uses 8 S-Boxes and Quasigroup has 7 S-boxes. The following steps are performed :

1. The component functions of each S-Box is derived and stored in a text file. The S-Boxes of the above mentioned cipher are 4×4 map so there are 15 component functions for each S-Box.

2. The HN-Transform is applied to each S-Box, by iterating over its component functions by varying over vector c and u. The component functions are stored in text file so they are read as string. Each digit of the function is identified as real or imaginary depending on the value of $wt(c * x)$ which defines the hamming weight of a vector given as $c_1x_1, c_2x_2, \ldots, c_nx_n$.

3. A table is generated consisting of values of HN-spectra for each component function of S-Box at each value of c. Following figures show the HN-sprectra obtained for S-boxes of PRESENT, SERPENT and Quasigroup.

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	32	64	32	64	32	128	64	32	16	32	16	32	16	64	32
2	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40
3	64	64	40	72	40	72	32	64	40	72	32	64	32	64	40	72
4	64	32	40	40	40	40	32	32	32	64	40	72	40	72	32	64
5	64	32	40	40	40	40	32	32	32	32	40	40	40	40	32	32
6	64	32	32	32	32	64	32	32	40	40	40	40	40	72	40	40
7	64	32	32	32	32	64	32	32	40	40	40	40	40	72	40	40
8	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40
9	64	64	40	72	40	72	32	64	40	72	32	64	32	64	40	72
10	64	32	32	16	32	16	64	32	32	16	64	32	64	32	128	64
11	64	32	32	16	32	16	64	32	64	32	32	16	32	16	32	16
12	64	32	32	64	32	32	32	32	40	40	40	72	40	40	40	40
13	64	32	32	64	32	32	32	32	40	40	40	72	40	40	40	40
14	64	32	40	40	40	40	32	32	32	32	40	40	40	40	32	32
15	64	32	40	40	40	40	32	32	32	64	40	72	40	72	32	64

Figure 7.1: HN-Spectra of PRESENT S-Box

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	40	40	32	32	40	40	32	32	40	40	32	64	72	72	64
2	64	40	40	32	40	32	80	40	32	40	40	32	40	32	80	40
3	64	32	32	32	40	40	40	40	32	32	32	64	40	40	40	72
4	64	64	64	128	32	32	32	64	32	32	32	64	16	16	16	32
5	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
6	64	40	40	32	40	80	32	40	32	40	40	32	40	80	32	40
7	64	32	32	32	40	40	40	40	32	32	32	64	40	40	40	72
8	64	40	40	32	40	80	32	40	32	40	40	32	40	80	32	40
9	64	32	32	32	40	40	40	40	32	32	32	64	40	40	40	72
10	64	32	32	64	32	16	16	32	32	64	64	128	16	32	32	64
11	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
12	64	40	40	32	40	32	80	40	32	40	40	32	40	32	80	40
13	64	32	32	32	40	40	40	40	32	32	32	64	40	40	40	72
14	64	32	32	64	32	16	16	32	64	32	32	32	32	16	16	16
15	64	40	40	32	32	40	40	32	32	40	40	32	64	72	72	64

Figure 7.2: HN-Spectra of SERPENT S-Box-0

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	40	40	32	32	40	40	32	32	40	40	32	64	72	72	64
2	64	40	40	32	40	32	80	40	32	40	40	32	40	32	80	40
3	64	32	32	32	40	40	40	40	32	32	32	64	40	40	40	72
4	64	64	64	128	32	32	32	64	32	32	32	64	16	16	16	32
5	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
6	64	40	40	32	40	80	32	40	32	40	40	32	40	80	32	40
7	64	32	32	32	40	40	40	40	32	32	32	64	40	40	40	72
8	64	40	40	32	40	80	32	40	32	40	40	32	40	80	32	40
9	64	32	32	32	40	40	40	40	32	32	32	64	40	40	40	72
10	64	32	32	64	32	16	16	32	32	64	64	128	16	32	32	64
11	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
12	64	40	40	32	40	32	80	40	32	40	40	32	40	32	80	40
13	64	32	32	32	40	40	40	40	32	32	32	64	40	40	40	72
14	64	32	32	64	32	16	16	32	64	32	32	32	32	16	16	16
15	64	40	40	32	32	40	40	32	32	40	40	32	64	72	72	64

Figure 7.3: HN-Spectra of SERPENT S-Box-1

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	64	32	32	64	128	32	64	32	32	16	16	32	64	16	32
2	64	32	40	40	40	40	32	32	40	40	80	80	32	32	40	40
3	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40
4	64	32	40	40	40	40	80	80	32	32	40	40	40	40	80	80
5	64	32	40	40	40	40	32	32	32	32	40	40	40	40	32	32
6	64	32	32	64	32	32	32	32	40	40	40	72	40	40	40	40
7	64	32	32	64	32	32	32	32	40	40	40	72	40	40	40	40
8	64	32	64	32	32	32	32	32	40	40	72	40	40	40	40	40
9	64	32	64	32	32	32	32	32	40	40	72	40	40	40	40	40
10	64	32	40	40	40	40	80	80	32	32	40	40	40	40	80	80
11	64	32	40	40	40	40	32	32	32	32	40	40	40	40	32	32
12	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40
13	64	32	40	40	40	40	32	32	40	40	80	80	32	32	40	40
14	64	64	32	32	32	32	16	16	64	128	32	64	32	64	16	32
15	64	64	32	32	32	32	16	16	32	32	16	16	64	32	32	16

Figure 7.4: HN-Spectra of SERPENT S-Box-2

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	40	40	32	32	40	40	32	32	40	40	32	64	72	72	64
2	64	32	40	40	40	40	32	32	32	64	40	72	40	72	32	64
3	64	40	32	40	40	32	40	32	32	40	32	40	40	32	40	32
4	64	32	64	32	40	40	72	40	40	40	72	40	32	32	64	32
5	64	40	40	32	40	80	32	40	40	32	32	40	32	40	40	32
6	64	32	40	40	32	32	40	40	40	40	32	32	40	40	32	32
7	64	40	32	40	64	72	32	40	40	32	40	32	72	64	40	32
8	64	32	40	40	32	64	40	72	40	40	32	32	40	72	32	64
9	64	40	32	40	32	40	32	40	40	32	40	32	40	32	40	32
10	64	64	32	32	40	72	40	40	40	72	40	40	32	64	32	32
11	64	40	40	32	40	32	80	40	40	32	32	40	32	40	40	32
12	64	64	40	72	40	72	32	64	32	32	40	40	40	40	32	32
13	64	40	32	40	40	32	40	32	32	40	32	40	40	32	40	32
14	64	32	32	64	32	16	16	32	64	32	32	32	32	16	16	16
15	64	40	40	32	32	40	40	32	32	40	40	32	64	72	72	64

Figure 7.5: HN-Spectra of SERPENT S-Box-3

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	32	32	64	32	16	16	32	32	64	64	128	16	32	32	64
2	64	40	40	32	32	40	40	32	64	72	72	64	32	40	40	32
3	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
4	64	40	32	40	40	32	40	32	40	32	40	32	80	40	80	40
5	64	40	32	40	40	80	40	80	40	32	40	32	32	40	32	40
6	64	64	40	72	40	72	32	64	40	72	32	64	32	64	40	72
7	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40
8	64	32	64	32	40	40	72	40	32	32	32	32	40	40	40	40
9	64	32	64	32	40	40	72	40	32	32	32	32	40	40	40	40
10	64	40	40	32	40	32	32	40	32	40	40	32	40	32	32	40
11	64	40	40	32	40	32	32	40	32	40	40	32	40	32	32	40
12	64	40	32	40	32	40	32	40	40	32	40	32	40	32	40	32
13	64	40	32	40	32	40	64	72	40	32	40	32	40	32	72	64
14	64	32	40	40	32	32	40	40	40	40	32	32	40	40	32	32
15	64	32	40	40	64	32	72	40	40	40	32	32	72	40	64	32

Figure 7.6: HN-Spectra of SERPENT S-Box-4

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	32	32	64	32	16	16	32	32	64	64	128	16	32	32	64
2	64	32	32	32	40	40	40	40	64	32	32	32	72	40	40	40
3	64	32	32	32	40	40	40	40	64	32	32	32	72	40	40	40
4	64	40	40	32	40	80	32	40	32	40	40	32	40	80	32	40
5	64	40	40	32	40	32	80	40	32	40	40	32	40	32	80	40
6	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
7	64	40	40	32	32	40	40	32	32	40	40	32	64	72	72	64
8	64	64	40	72	32	32	40	40	40	72	32	64	40	40	32	32
9	64	32	40	40	32	32	40	40	40	40	32	32	40	40	32	32
10	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40
11	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40
12	64	40	64	72	40	32	72	64	40	32	72	64	32	40	64	72
13	64	40	32	40	40	32	40	32	40	32	40	32	32	40	32	40
14	64	40	32	40	32	40	32	40	40	32	40	32	40	32	40	32
15	64	40	32	40	64	72	32	40	40	32	40	32	72	64	40	32

Figure 7.7: HN-Spectra of SERPENT S-Box-5

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	40	32	40	40	80	40	80	40	32	40	32	32	40	32	40
2	64	64	32	32	32	32	16	16	64	128	32	64	32	64	16	32
3	64	40	32	40	40	32	40	32	40	32	40	32	80	40	80	40
4	64	40	32	40	40	32	40	32	40	32	40	32	80	40	80	40
5	64	32	32	16	32	16	64	32	32	64	16	32	16	32	32	16
6	64	40	32	40	40	80	40	80	40	32	40	32	32	40	32	40
7	64	32	32	16	32	16	16	16	32	64	16	32	16	32	16	32
8	64	40	64	72	32	40	32	40	40	32	72	64	40	32	40	32
9	64	32	32	32	40	40	40	40	32	32	64	32	40	40	72	40
10	64	40	32	40	32	40	32	40	40	32	40	32	40	32	40	32
11	64	32	32	32	40	40	40	40	32	32	64	32	40	40	72	40
12	64	32	32	64	40	40	40	72	32	32	32	32	40	40	40	40
13	64	40	64	72	32	40	32	40	40	32	72	64	40	32	40	32
14	64	32	32	64	40	40	40	72	32	32	32	32	40	40	40	40
15	64	40	32	40	32	40	32	40	40	32	40	32	40	32	40	32

Figure 7.8: HN-Spectra of SERPENT S-Box-6

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	40	40	80	32	40	40	80	32	40	40	80	32	40	40	80
2	64	32	40	40	40	40	80	80	32	32	40	40	40	40	80	80
3	64	40	32	40	40	32	40	32	32	40	32	40	40	32	40	32
4	64	40	32	40	40	32	40	32	40	80	40	80	32	40	32	40
5	64	32	40	40	40	40	32	32	40	40	32	32	80	80	40	40
6	64	40	40	32	32	40	40	32	40	32	32	40	40	32	32	40
7	64	32	32	32	32	64	32	32	40	40	40	40	40	72	40	40
8	64	64	32	32	32	32	32	32	40	72	40	40	40	40	40	40
9	64	40	40	32	32	40	40	32	40	32	32	40	40	32	32	40
10	64	32	40	40	40	40	80	80	40	40	32	32	32	32	40	40
11	64	40	32	40	40	32	40	32	40	32	40	32	32	40	32	40
12	64	40	32	40	40	80	40	80	32	40	32	40	40	80	40	80
13	64	32	40	40	40	40	32	32	64	32	72	40	72	40	64	32
14	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
15	64	32	32	16	64	32	32	16	32	16	64	32	32	16	32	16

Figure 7.9: HN-Spectra of SERPENT S-Box-7

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	64	32	32	32	32	16	16	32	32	16	16	64	32	32	16
2	64	32	40	40	32	64	40	72	32	32	40	40	32	32	40	40
3	64	32	40	40	32	64	40	72	32	32	40	40	32	32	40	40
4	64	40	40	80	40	32	32	40	40	32	32	40	32	40	40	32
5	64	40	40	32	40	32	32	40	40	32	32	40	32	40	40	80
6	64	40	32	40	40	32	40	32	40	32	40	32	32	40	32	40
7	64	40	32	40	40	32	40	32	40	32	40	32	32	40	32	40
8	64	32	40	40	32	64	40	72	32	32	40	40	32	32	40	40
9	64	32	40	40	32	64	40	72	32	32	40	40	32	32	40	40
10	64	32	32	16	32	64	16	32	64	32	32	16	32	32	16	16
11	64	32	32	16	64	32	32	16	32	64	16	32	32	32	16	16
12	64	40	32	40	40	32	40	32	40	32	40	32	32	40	32	40
13	64	40	64	72	40	32	72	64	40	32	72	64	32	40	64	72
14	64	40	40	32	40	32	80	40	40	32	32	40	32	40	40	32
15	64	40	40	32	40	32	32	40	40	32	80	40	32	40	40	32

Figure 7.10: HN-Spectra of Quasigroup S-Box-1

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	64	32	32	32	32	16	16	32	32	16	16	64	32	32	16
2	64	32	40	40	32	64	40	72	32	32	40	40	32	32	40	40
3	64	32	40	40	32	64	40	72	32	32	40	40	32	32	40	40
4	64	40	40	80	40	32	32	40	40	32	32	40	32	40	40	32
5	64	40	40	32	40	32	32	40	40	32	32	40	32	40	40	80
6	64	40	32	40	40	32	40	32	40	32	40	32	32	40	32	40
7	64	40	32	40	40	32	40	32	40	32	40	32	32	40	32	40
8	64	32	40	40	32	64	40	72	32	32	40	40	32	32	40	40
9	64	32	40	40	32	64	40	72	32	32	40	40	32	32	40	40
10	64	32	32	16	32	64	16	32	64	32	32	16	32	32	16	16
11	64	32	32	16	64	32	32	16	32	64	16	32	32	32	16	16
12	64	40	32	40	40	32	40	32	40	32	40	32	32	40	32	40
13	64	40	64	72	40	32	72	64	40	32	72	64	32	40	64	72
14	64	40	40	32	40	32	80	40	40	32	32	40	32	40	40	32
15	64	40	40	32	40	32	32	40	40	32	80	40	32	40	40	32

Figure 7.11: HN-Spectra of Quasigroup S-Box-2

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	40	32	40	32	40	32	40	64	72	32	40	32	40	32	40
2	64	40	40	32	40	80	32	40	32	40	40	32	40	80	32	40
3	64	32	40	40	40	40	32	32	32	64	40	72	40	72	32	64
4	64	64	40	72	40	72	32	64	40	72	32	64	32	64	40	72
5	144	72	72	64	72	64	64	72	72	64	64	72	64	72	72	64
6	144	72	80	40	80	40	80	40	72	64	40	32	40	32	40	32
7	64	32	32	64	32	32	32	32	40	40	40	72	40	40	40	40
8	64	32	40	40	40	40	32	32	32	32	40	40	40	40	32	32
9	64	40	40	32	40	32	32	40	32	40	40	32	40	32	32	40
10	64	40	32	40	32	40	32	40	64	72	32	40	32	40	32	40
11	64	32	64	32	32	16	32	16	32	16	32	16	64	32	32	16
12	64	64	32	32	32	32	32	32	40	72	40	40	40	40	40	40
13	64	40	64	72	32	40	32	40	40	32	72	64	40	32	40	32
14	64	40	40	32	40	32	32	40	40	32	32	40	80	40	40	32
15	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40

Figure 7.12: HN-Spectra of Quasigroup S-Box-3

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	40	32	40	32	40	32	40	64	72	32	40	32	40	32	40
2	64	40	40	32	40	80	32	40	32	40	40	32	40	80	32	40
3	64	32	40	40	40	40	32	32	32	64	40	72	40	72	32	64
4	64	64	40	72	40	72	32	64	40	72	32	64	32	64	40	72
5	144	72	72	64	72	64	64	72	72	64	64	72	64	72	72	64
6	144	72	80	40	80	40	80	40	72	64	40	32	40	32	40	32
7	64	32	32	64	32	32	32	32	40	40	40	72	40	40	40	40
8	64	32	40	40	40	40	32	32	32	32	40	40	40	40	32	32
9	64	40	40	32	40	32	32	40	32	40	40	32	40	32	32	40
10	64	40	32	40	32	40	32	40	64	72	32	40	32	40	32	40
11	64	32	64	32	32	16	32	16	32	16	32	16	64	32	32	16
12	64	64	32	32	32	32	32	32	40	72	40	40	40	40	40	40
13	64	40	64	72	32	40	32	40	40	32	72	64	40	32	40	32
14	64	40	40	32	40	32	32	40	40	32	32	40	80	40	40	32
15	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40

Figure 7.13: HN-Spectra of Quasigroup S-Box-4

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	40	32	40	32	40	32	40	64	72	32	40	32	40	32	40
2	64	40	40	32	40	80	32	40	32	40	40	32	40	80	32	40
3	64	32	40	40	40	40	32	32	32	64	40	72	40	72	32	64
4	64	64	40	72	40	72	32	64	40	72	32	64	32	64	40	72
5	144	72	72	64	72	64	64	72	72	64	64	72	64	72	72	64
6	144	72	80	40	80	40	80	40	72	64	40	32	40	32	40	32
7	64	32	32	64	32	32	32	32	40	40	40	72	40	40	40	40
8	64	32	40	40	40	40	32	32	32	32	40	40	40	40	32	32
9	64	40	40	32	40	32	32	40	32	40	40	32	40	32	32	40
10	64	40	32	40	32	40	32	40	64	72	32	40	32	40	32	40
11	64	32	64	32	32	16	32	16	32	16	32	16	64	32	32	16
12	64	64	32	32	32	32	32	32	40	72	40	40	40	40	40	40
13	64	40	64	72	32	40	32	40	40	32	72	64	40	32	40	32
14	64	40	40	32	40	32	32	40	40	32	32	40	80	40	40	32
15	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40

Figure 7.14: HN-Spectra of Quasigroup S-Box-5

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	40	32	40	32	40	32	40	64	72	32	40	32	40	32	40
2	64	40	40	32	40	80	32	40	32	40	40	32	40	80	32	40
3	64	32	40	40	40	40	32	32	32	64	40	72	40	72	32	64
4	64	64	40	72	40	72	32	64	40	72	32	64	32	64	40	72
5	144	72	72	64	72	64	64	72	72	64	64	72	64	72	72	64
6	144	72	80	40	80	40	80	40	72	64	40	32	40	32	40	32
7	64	32	32	64	32	32	32	32	40	40	40	72	40	40	40	40
8	64	32	40	40	40	40	32	32	32	32	40	40	40	40	32	32
9	64	40	40	32	40	32	32	40	32	40	40	32	40	32	32	40
10	64	40	32	40	32	40	32	40	64	72	32	40	32	40	32	40
11	64	32	64	32	32	16	32	16	32	16	32	16	64	32	32	16
12	64	64	32	32	32	32	32	32	40	72	40	40	40	40	40	40
13	64	40	64	72	32	40	32	40	40	32	72	64	40	32	40	32
14	64	40	40	32	40	32	32	40	40	32	32	40	80	40	40	32
15	64	32	40	40	40	40	32	32	40	40	32	32	32	32	40	40

Figure 7.15: HN-Spectra of Quasigroup S-Box-6

Function	c=0	c=1	c=2	c=3	c=4	c=5	c=6	c=7	c=8	c=9	c=10	c=11	c=12	c=13	c=14	c=15
1	64	32	32	64	32	16	16	32	64	32	32	32	32	16	16	16
2	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
3	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
4	64	64	40	72	32	32	40	40	40	72	32	64	40	40	32	32
5	64	32	40	40	32	32	40	40	40	40	80	80	40	40	80	80
6	64	40	64	72	32	40	32	40	40	32	72	64	40	32	40	32
7	64	40	32	40	32	40	32	40	40	80	40	80	40	80	40	80
8	64	32	32	16	64	32	32	16	32	16	16	16	32	16	16	32
9	64	32	32	16	32	16	16	16	32	16	16	16	64	32	32	32
10	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
11	64	40	40	32	32	40	40	32	32	40	40	32	32	40	40	32
12	64	64	40	72	32	32	40	40	40	72	32	64	40	40	32	32
13	64	32	40	40	32	32	40	40	40	40	80	80	40	40	80	80
14	64	40	64	72	32	40	32	40	40	32	72	64	40	32	40	32
15	64	40	32	40	32	40	32	40	40	80	40	80	40	80	40	80

Figure 7.16: HN-Spectra of Quasigroup S-Box-7

4. Next, the maximum value of HN-spectra for each component function of the S-Box is obtained and its corresponding real and imaginary value along with the vector c and u is also found which will be required for constructing the new function given by the following equation :

$$f'(x) = s_c(x) \oplus \epsilon_1(u, c) \oplus (\epsilon_1(u, c) \oplus \epsilon_2(u, c))c \cdot x \oplus u \cdot x$$

5. Then, the hamming distance is calculated between the component function of the S-Box whose maximum value of HN-spectra was used to get the new function f'(x) and the function f'(x) which is calculated using the above equation. The hamming distance is calculated in two ways - one, finding the hamming distance using equation 2.1 and second, using the HN-Transform to find the hamming distance according to equation 6.2.

We consider only those component functions for which the HN-Spectra value is maximum. For PRESENT S-Box, the maximum HN-Spectra value is 128. Some of the SERPENT S-Boxes have the maximum HN-Spectra value as 128 and others have 80 as their maximum value. The S-Boxes of Quasigroup have the maximum HN-spectra value as 144 and 80.

The algebraic degree of all such component functions of the S-Box is calculated using the Sage software. The algebraic degree of the component functions of the PRESENT S-Box is 2 which means they are a quadratic functions. The hamming distance between the component function and new function f'(x) is 0 which shows that both the functions are same. There cannot be a quadratic approximation for the component functions of PRESENT S-Box. The table 7.1 shows both the functions.

Similarly the component functions of S-boxes 0, 1, 2, 4, 5 and 6 of SERPENT cipher for which the maximum HN-spectra value is 128, also have the algebaric degree 2 which means the new function obtained is same as their component function so it cannot have quadratic approximation.

The component functions of S-Boxes 3 and 7 of SERPENT cipher, with maximum HN-sprectra value as 80, have algebraic degree 3. Their quadratic approximation is possible that is a function of degree 3 can be aprroximated by a function of degree 2. The new function f'(x) along with the component function and their hamming distances are given in the table 7.2.

Component Function of PRESENT S-Box	**Boolean Function**	**HD by HN Transform**	**HD by XOR**
$x_0 + x_1x_2 + x_2 + x_3$	$x_0 + x_1x_2 + x_3 + x_2$	0	0
$x_0+x_1x_2+x_1x_3+x_2x_3+1$	$x_0+x_1x_2+x_1x_3+x_2x_3+1$	0	0

Table 7.1: Quadratic Approximation of PRESENT S-Box

For Quasigroup S-Box, maximum value of HN-Spectra is 80 for some S-boxes and others have maximum value as 144. The component functions of both type of

Component Function of SERPENT S-Box	**Boolean Function**	**HD by HN Transform**	**HD by XOR**
$x_0x_1x_2+x_0x_1x_3+x_0x_1+x_0x_2x_3+x_0x_3+x_1x_2x_3+x_1x_2+x_1x_3+x_1+x_2x_3+x_2$	$x_1x_3 + x_4 + x_1 + 1$	2	2
$x_0x_1x_2+x_0x_1x_3+x_0x_1+x_0x_2x_3 + x_0 + x_1x_2x_3 + x_1x_2 + x_1 + x_2x_3 + x_2$	$x_1x_2 + x_1 + x_2 + x_0$	2	2
$x_0x_1 + x_0x_2x_3 + x_0x_3 + x_1x_2x_3 + x_1x_3 + x_2x_3 + x_2 + 1$	$x_0x_1 + x_2 + x_3$	2	2
$x_0x_1x_3+x_0x_1+x_0x_2x_3+x_0x_2+x_0x_3+x_1x_2+x_1+x_2 + x_3$	$x_1x_2 + x_0 + x_1 + x_2 + x_3$	2	2

Table 7.2: Quadratic Approximation of SERPENT S-Box

S-Boxes have algebraic degree 3. Quadratic approximation using HN-Transform is possible for the Quasi S-boxes having maximum HN spectra value 80 which is shown in the table 7.3 given below.

The Quasigroup S-Box with maximum HN-spectra value as 144 do not have quadratic approximation usinh HN-Transform. So we consider quadratic approximation for the component functions that have their second maximum value as 80. The table 7.4 shows the quadratic functions approximated for Quasi S-Boxes that have second maximum HN spectra value as 80.

Component Function of Quasi S-Box	**Boolean Function**	**HD by HN Transform**	**HD by XOR**
$x_0x_1x_2 + x_0x_1x_3 + x_0x_2x_3 + x_0x_2 + x_0x_3 + x_1x_2x_3 + x_1 + x_3 + 1$	$x_0x_1 + x_0 + x_1 + x_3 + 1$	2	2
$x_0x_1x_2 + x_0x_1x_3 + x_0x_2x_3 + x_0 + x_1x_2x_3 + x_2$	$x_0x_1 + x_0x_2 + x_0x_3 + x_1x_2 + x_1x_3 + x_2x_3 + x_1 + x_3 + 1$	2	2
$x_0x_1x_2 + x_0x_1x_3 + x_0x_2x_3 + x_0 + x_1x_2x_3 + x_2x_3 + x_3$	$x_0x_1 + x_0 + x_3$	2	2
$x_0x_1x_2 + x_0x_1x_3 + x_0x_2x_3 + x_0x_2 + x_0x_3 + x_1x_2x_3 + x_1 + x_2x_3 + x_3 + 1$	$x_0x_1 + x_0x_2 + x_0x_3 + x_1x_2 + x_1x_3 + x_2x_3 + x_3 + 1$	2	2
$x_0x_1x_2 + x_0x_2x_3 + x_0 + x_1x_3 + x_3$	$x_1x_3 + x_0 + x_3$	2	2
$x_0x_1x_2 + x_0x_1 + x_0x_2 + x_0x_3 + x_1x_2x_3 + x_1x_3 + x_1 + x_2x_3 + x_2 + x_3$	$x_0x_3 + x_0 + x_1 + x_2$	2	2

Table 7.3: Quadratic Approximation of Quasi S-Box

Component Function of Quasi S-Box	**Boolean Function**	**HD by HN Transform**	**HD by XOR**
$x_0x_1x_3+x_0x_2x_3+x_0x_2+x_0x_3+x_1x_2x_3+x_2+x_3+1$	$x_0x_2 + x_2 + x_3 + 1$	2	2
$x_0x_1x_2+x_1x_2x_3+x_1+x_3$	$x_1x_2 + x_1 + x_3$	2	2
$x_0x_1x_2 + x_0x_1 + x_0x_2 + x_0 + x_1x_2x_3 + x_1x_2 + x_1x_3 + x_2x_3 + x_2$	$x_2x_3 + x_2 + x_3$	2	2
$x_0x_1x_2+x_0x_1x_3+x_0x_1+x_0x_2x_3 + x_0 + x_1x_2x_3 + x_1 + x_2 + x_3 + 1$	$x_2x_3+x_2+x_3+x_0+x_1+1$	2	2
$x_0x_1x_3+x_0x_1+x_0x_2x_3+x_0 + x_1x_2x_3 + x_1x_2 + x_1x_3 + x_1 + x_2 + x_3$	$x_0x_2 + x_0 + x_2 + x_3$	2	2
$x_0x_1x_3+x_0x_1+x_0x_2x_3+x_0 + x_1x_2x_3 + x_1x_2 + x_1x_3 + x_1 + x_2 + x_3$	$x_0x_2 + x_0x_3 + x_2x_3 + 1$	2	2

Table 7.4: Quadratic Approximation of Quasi S-Box

Chapter 8

Conclusion

This report comprises of the description of the various transforms, their generalization and their use in aprroximating quadratic funcions for the various 4 bit S-Boxes. The Walsh-Hadamard transform discussed here classifies the boolean function into a new class called bent function. Similarly the Nega-Hadamard transform which is for the complex-valued function describe the criteria for a boolean function to be a negabent function. A new algorithm the for fast computation of the Walsh-hadamard coefficients is also described. The generalized form of Walsh and Nega-Hadamard transform is the HN-transform which is obtained by the recursive application of tensor product to both the Walsh and Nega-Hadamard transform. The complexity of computing the HN-spectra using the Fast HN-transform algorithm is also explained. The HN-Tranform is used to find the best quadratic approximation for the various S-boxes. The component functions of the various S-Boxes are dervied. The HN-Spectra for these functions is obtained and then for the maximum value of HN-spectra, the quadratic function is obtained. The hamming distance of both the functions is found out to know whether it is the best approximation or not. The Walsh-Hadamard transform has its application in data encryption and quantum computing which is further being explored by the researchers.

In our report, the HN-Transform is applied to 4 bit S-Boxes and quadratic approximation is obtained for them. This work can be further expanded by finding the quadratic approximation of 6×4 and 8 bit S-Boxes which are used in Data Encryption Standard and Advanced Encryption Standard respectively.

Bibliography

[1] Ross Anderson, Eli Biham, Lars Knudsen, and Haifa Technion. Serpent: A flexible block cipher with maximum assurance. In *The first AES candidate conference*, pages 589–606. Citeseer, 1998.

[2] Andrey Bogdanov, Lars R Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew JB Robshaw, Yannick Seurin, and Charlotte Vikkelsoe. Present: An ultra-lightweight block cipher. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 450–466. Springer, 2007.

[3] Claude Carlet. Boolean functions for cryptography and error correcting codes. *Boolean models and methods in mathematics, computer science, and engineering*, 2:257–397, 2010.

[4] Yves Crama and Peter L Hammer. *Boolean models and methods in mathematics, computer science, and engineering*, volume 2. Cambridge University Press, 2010.

[5] Thomas W Cusick and Pantelimon Stanica. *Cryptographic Boolean functions and applications*. Academic Press, 2017.

[6] Behrouz A Forouzan and Debdeep Mukhopadhyay. *Cryptography and Network Security (Sie)*. McGraw-Hill Education, 2011.

[7] Sugata Gangopadhyay, Subhamoy Maitra, Nishant Sinha, and Pantelimon Stanica. Quantum algorithms related to hn-transforms of boolean functions. In *Codes, Cryptology and Information Security: Second International Conference, C2SI 2017, Rabat, Morocco, April 10–12, 2017, Proceedings-In Honor of Claude Carlet*, volume 10194, page 314. Springer, 2017.

[8] Sugata Gangopadhyay, Enes Pasalic, and Pantelimon Stănică. A note on generalized bent criteria for boolean functions. *IEEE Transactions on Information Theory*, 59(5):3233–3236, 2013.

[9] Jonathan Katz, Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. *Handbook of applied cryptography*. CRC press, 1996.

[10] Smile Markovski. Design of crypto primitives based on quasigroups. *Quasigroups and Related Systems*, 23(1):41–90, 2015.

[11] Hristina Mihajloska and Danilo Gligoroski. Construction of optimal 4-bit s-boxes by quasigroups of order 4. In *The Sixth International Conference on Emerging Security Information, Systems and Technologies, SECURWARE*, 2012.

[12] Kamsiah Mohamed, Mohd Nazran Mohammed Pauzi, Fakariah Hani Hj Mohd Ali, Suriyani Ariffin, and Nurul Huda Nik Zulkipli. Study of s-box properties in block cipher. In *Computer, Communications, and Control Technology (I4CT), 2014 International Conference on*, pages 362–366. IEEE, 2014.

[13] Ritu Pahal and Vikas Kumar. Efficient implementation of aes. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(7):290–295, 2013.

[14] Matthew G Parker. Generalised s-box nonlinearity. *NESSIE Public Document-NES/DOC/UIB/WP5/020/A*, 2003.

[15] Bhupendra Singh, Lexy Alexander, and Sanjay Burman. On algebraic relations of serpent s-boxes. *IACR Cryptology ePrint Archive*, 2009:38, 2009.

[16] William Stallings. *Cryptography and network security: principles and practice.* Pearson Education India, 2003.

[17] Pantelimon Stănică, Sugata Gangopadhyay, Ankita Chaturvedi, Aditi Kar Gangopadhyay, and Subhamoy Maitra. Nega–hadamard transform, bent and negabent functions. In *International Conference on Sequences and Their Applications*, pages 359–372. Springer, 2010.

[18] Wei Su, Alexander Pott, and Xiaohu Tang. Characterization of negabent functions and construction of bent-negabent functions with maximum algebraic degree. *IEEE Transactions on Information Theory*, 59(6):3387–3395, 2013.

[19] Natalia Tokareva. *Bent functions: results and applications to cryptography.* Academic Press, 2015.