

ZEROABILITY PATTERNS OF MONOMIALS IN THE SIGN-REPRESENTATION OF BOOLEAN FUNCTIONS

A Thesis

by

Oytun Yapar

Submitted to the

Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Computer Science

Özyeğin University
June 2017

Copyright © 2017 by Oytun Yapar

ZEROABILITY PATTERNS OF MONOMIALS IN THE SIGN-REPRESENTATION OF BOOLEAN FUNCTIONS

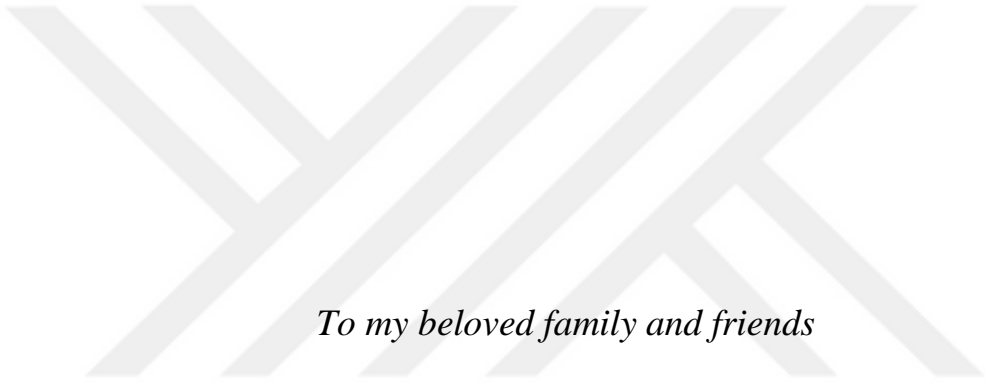
Approved by:

Associate Professor Erhan Öztöp, Advisor,
Department of Computer Science
Özyeğin University

Assistant Professor Melih Kandemir,
Department of Computer Science
Özyeğin University

Associate Professor Ali Fuat Alkaya,
Department of Computer Science and
Engineering
Marmara University

Date Approved: 24 May 2017



To my beloved family and friends

ABSTRACT

Boolean functions (BF) are one of the fundamental concepts in discrete mathematics. It is possible to represent any BF by a unique polynomial when one takes -1 as True and 1 as False. Coefficients of the polynomial representing the given BF can be found with Lagrange interpolation. When the exact interpolation criterion is replaced with the sign-match criterion, one can find infinitely many sign representing polynomials for a given truth table. The problem of finding a minimum number of monomial set that is sufficient to represent a BF is a difficult mathematical problem. This thesis aims to contribute to its solution by investigating the zeroability patterns of monomials. To this end, we asked which monomials must be in a minimum sign representing polynomial. This question drove us to make numerical investigations on the BFs in lower dimensions. For all 3- and 4-variable BFs, we found all the monomial subsets, whose elements can be zeroed and we introduced a graph representation indicating whether particular pairs of monomials could be absent from any sign representation. In addition to the numerical investigations, we have also proved that if a three-element monomial set S , could not be absent altogether from the sign representation of a BF, then there must be at least a two element subset of S which could not be absent in any sign representation of that BF. We expect these results will give support to the development of heuristic algorithms to construct close-to-minimum number of monomial sign representing polynomials for BFs.

ÖZETÇE

Boolean fonksiyonlar (BF) ayrık matematik alanındaki temel konulardan biridir. 1'i Yanlış ve -1'i Doğru olarak kabul edersek, bir BF'yi tek bir polinomla ifade edebiliriz. Verilen BF'in katsayıları Lagrange interpolasyonu ile bulunabilir. Ne zaman tam interpolasyon işaret eşleşme kriteri ile değiştirilirse, verilen bir gerçeklik tablosu için sonsuz tane işaret temsili polinomu bulunabilir. Bir BF'yi temsil etmek için yeterli, minimum sayıda terim içeren bir küme bulmak zor bir matematik problemidir. Bu tez bu problemin çözümüne, terimlerin BF'yi temsil ederken sıfırlanabilme düzenlerini araştırarak katkı sunmayı hedeflemektedir. Bu amaçla, hangi terimler minimum işaret temsili polinomda olmak zorundadır sorusunu sorduk. Bu soru bizi küçük boyutlarda numerik araştırmalar yapmaya itti. Tüm üç ve dört değişkenli BF'ler için, elemanları bir arada sıfırlanabilen tüm alt kümeleri bulduk ve hangi monomial çiftlerinin birlikte herhangi bir işaret temsilinden eksik olup olamayacağını belirten, bir graf tanımladık. Numerik araştırmalara ek olarak, üç elemanlı bir terim kümesi S , tüm elemanları bir arada bir BF'in işaret temsilinden çıkarılamıyorsa, S 'in iki elemanlı alt kümelerinden en az bir tanesinin bu BF'in işaret temsilinden çıkarılamaz olduğunu ispatladık. Bu sonuçların bize, minimum terim sayısına yakın sayıda terim bulunduran, BF'lerin işaret temsili polinomlarını bulmamızı sağlayacak buluşsal bir algoritma bulma konusunda destek olmasını bekliyoruz.

ACKNOWLEDGMENTS

Special thanks to my advisor Professor Erhan Öztop for his guidance, patience and knowledge. He was always constructive and positive to me. Thanks to his comments and remarks. Thanks to my family, they always supported me for finishing my thesis. Thanks to my friends, especially Ekin Yağmur Gönen for reviewing my thesis. Additionally subject of this thesis was really interesting and I would like to continue researching on this topic.




TABLE OF CONTENTS

ABSTRACT	iv
ÖZETÇE	v
ACKNOWLEDGMENTS	vi
LIST OF TABLES	x
LIST OF FIGURES	ix
I INTRODUCTION	1
1.1 Boolean Functions	1
1.1.1 Higher Order Neuron Representations of Boolean Functions ...	6
1.2 Minimum Number of Monomials Problem of Polynomial Boolean Functions	7
1.2.1 Equivalency of Boolean Functions	8
II PREVIOUS WORK	10
III ZEROABILITY PATTERNS OF MONOMIALS IN THE HIGHER ORDER NEURON REPRESENTATIONS OF BOOLEAN FUNCTIONS	13
3.1 Numerical Investigations on Monomial Zeroability.....	13
3.2 Pairwise Zeroability of Monomials From Sign Representations	18
3.3 Pairwise Zeroability and Incompatibility Graph	24
IV CONCLUSION	29
APPENDIX A	32
BIBLIOGRAPHY	34

LIST OF TABLES

1	Truth table of given f_3 in both Boolean domain and real numbers domain	1
2	Two dimensions Sylvester type Hadamard matrix	3
3	The function output vector $[-1 \ 1 \ -1 \ 1 \ -1 \ 1 \ -1 \ 1]$ can be mapped (with $t(b)=(1-b)/2$) to the binary number 10101010 that is 0xaa in hexadecimal notation.	9
4	Number of zeroable subsets for each function class is given for each subset cardinality (3 dimensions)	16
5	Number of zeroable subsets for each function class is given for each subset cardinality (4 dimensions)	16
6	Number of first time introduced non-zeroable subsets for each function class is given for each subset cardinality (3 dimensions).....	17
7	Number of first time introduced non-zeroable subsets for each function class is given for each subset cardinality (4 dimensions).....	18

LIST OF FIGURES

1	Higher-order neuron that represents Boolean function given in Table-1 and a biological neuron	6
2	Incompatibility graph of Class 0xaaff	26
3	Incompatibility graph of Class 0xaba5	26
4	Incompatibility graph of Class 0xaa55	27
5	Incompatibility graph of Class 0xab55	27
6	Incompatibility graph of Class 0xab12	28
7	Incompatibility graph of Class 0xac90	28
8	Incompatibility graph of Class 0xaa	32
9	Incompatibility graph of Class 0xab	32
10	Incompatibility graph of Class 0xac	33
11	Incompatibility graph of Class 0xbb55	33
12	Incompatibility graph of Class 0xaba4	33

CHAPTER I

INTRODUCTION

1.1 Boolean Functions

When -1 and +1 are used to represent True and False respectively, a Boolean function is identified by a real valued vector function $f: \{-1, 1\}^n \rightarrow \{-1, 1\}$. A motivating example will ease comprehension of this definition. Table 1 shows truth table of example function $f_3: \{-1, 1\}^3 \rightarrow \{-1, 1\}$.

Table 1: Truth table of given f_3 in both Boolean domain and real numbers domain

Decimal Value	Binary notation				Bipolar notation			
	x_3	x_2	x_1	$f_3(x_1, x_2, x_3)$	x_3	x_2	x_1	$f_3(x_1, x_2, x_3)$
0	0	0	0	1	+1	+1	+1	-1
1	0	0	1	0	+1	+1	-1	+1
2	0	1	0	0	+1	-1	+1	+1
3	0	1	1	1	+1	-1	-1	-1
4	1	0	0	0	-1	+1	+1	+1
5	1	0	1	0	-1	+1	-1	+1
6	1	1	0	1	-1	-1	+1	-1
7	1	1	1	0	-1	-1	-1	+1

Unique polynomial outputs this function could be obtained by polynomial interpolation. There could be at most 2^n terms in this polynomial (multiplication of all possible combinations of variables in the input vector e.g. $x_1x_2, x_1x_2x_3$).

Proposition 1: A Boolean function f is represented by a unique polynomial, terms of which are combinations of input variables multiplied with a coefficient.

$$f(x_1, x_2, \dots, x_n) = \sum_{i=1}^{2^n} a_i \prod_{k \in S_i} x_k \text{ where } S_i \subset \{1, 2, \dots, n\} \quad (1)$$

Proof: Application of Lagrange interpolation for a given output and dimension constructs function form defined in (1).

$$l_i(\mathbf{x}) = l_i(x_1, x_2, \dots, x_n) = f(\mathbf{x}_i) \frac{\prod_{k=1, k \neq i}^{2^n} (\mathbf{x} - \mathbf{x}_k)^T (\mathbf{x} - \mathbf{x}_k)}{\prod_{k=1, k \neq i}^{2^n} (\mathbf{x}_i - \mathbf{x}_k)^T (\mathbf{x}_i - \mathbf{x}_k)} \quad (2)$$

where \mathbf{x} represents vector of input variables, $\mathbf{x}_i \in \{1, -1\}^n$ represents the assigned input vector(assignment vector) and $\mathbf{x}_k \in \{1, -1\}^n$ represents all possible assignment vectors other than \mathbf{x}_i . Since $l_i(\mathbf{x}_k) = 0$ and $l_i(\mathbf{x}_i) = f(\mathbf{x}_i)$ the polynomial $q(\mathbf{x}) = \sum_{i=1}^{2^n} l_i(\mathbf{x})$ interpolates $f(\mathbf{x})$ at the assignment vectors. Noting that $\mathbf{x}_k^2 = 1$, expanding $q(\mathbf{x})$ results in given form (1) (Oztop 2009).

Lagrange interpolation calculated that polynomial $f(x_1, x_2, x_3) = (1 - x_1 + x_2 - x_1x_2 - x_3 + x_1x_3 - x_2x_3 - 3x_1x_2x_3)/4$ represents function given in the Table 1.

Remark 1: Lagrange interpolation also proves that polynomial functions could be in $f: \{-1, 1\}^n \rightarrow \mathbb{R}$ form.

Terms (monomials) of this polynomial form could be written as vectors and values of the vector elements depend on assignment vectors. For example if $x_1 = 1$ and $x_2 = -1$, x_1x_2 must be equal to -1 . If monomial vectors and assigned values to these vectors are ordered properly, so called Sylvester type Hadamard matrix \mathbf{D}_n (Siu, Roychowdhury et al. 1995) will be obtained, (Monomial order: $(1, x_1, x_2, x_1x_2, x_3, x_1x_3, x_2x_3, x_1x_2x_3, \dots, x_1x_2 \dots x_n)$, Assignment order: $(x_n, x_{n-1}, \dots, x_2, x_1) : (11\dots11, 11\dots1-1, 11\dots-11, \dots, -1-1\dots-1-1)$), where n is dimension. Simply \mathbf{D}_n represents all monomial values changing with binary counting input. Table-2 shows an example \mathbf{D}_n matrix for two dimensions (\mathbf{D}_2).

Table 2: Two dimensions Sylvester type Hadamard matrix

1	x_1	x_2	x_1x_2
1	1	1	1
1	-1	1	-1
1	1	-1	-1
1	-1	-1	1

\mathbf{D}_n has properties described below.

Lemma 1: \mathbf{D}_n is in the recursive form.

$$\mathbf{D}_1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \mathbf{D}_{n+1} = \begin{bmatrix} \mathbf{D}_n & \mathbf{D}_n \\ \mathbf{D}_n & -\mathbf{D}_n \end{bmatrix} \text{ for } n > 0 \quad (3)$$

Lemma 2: \mathbf{D}_n is symmetric (i. e. $(\mathbf{D}_n)^T = \mathbf{D}_n$).

Lemma 3: \mathbf{D}_n is orthogonal when scaled. $\mathbf{D}_n\mathbf{D}_n = 2^n\mathbf{I}$. This means inverse of \mathbf{D}_n is $(\mathbf{D}_n)^{-1} = 2^{-n}\mathbf{D}_n$.

With proposition 1 and the definition of \mathbf{D}_n , polynomial f could be defined as

$$\mathbf{f} = \mathbf{D}_n \mathbf{a} \quad (4)$$

where \mathbf{f} is vector representation of f and \mathbf{a} is vector of monomial coefficients (weights of monomials). Each element of \mathbf{a} corresponds a specific monomial. So, if elements of \mathbf{a} vector are found, automatically polynomial represents f is found. Equation that calculates \mathbf{a} can be constructed easily by using properties of \mathbf{D}_n , which is

$$\mathbf{a} = 2^{-n} \mathbf{D}_n \mathbf{f} \quad (5)$$

Definition 1 (Spectrum): Here $\mathbf{a} = (a_1, a_2, \dots, a_{2^n})^T$ and given equation (1) \mathbf{a} is called spectrum of f .

Remark 2: Spectrum \mathbf{a} represents given function f since it is coefficients of monomials. Noting $\mathbf{1} = \text{diag}(\mathbf{f}) \mathbf{D}_n \mathbf{a}$ ($\mathbf{f} = \text{diag}(\mathbf{f}) \mathbf{1}$ and $\text{diag}(\mathbf{f})^{-1} = \text{diag}(\mathbf{f})$), $2^n \mathbf{a}^T = \mathbf{1}^T \text{diag}(\mathbf{f}) \mathbf{D}_n$, so \mathbf{a}^T scaled by 2^n equals row sum of $\text{diag}(\mathbf{f}) \mathbf{D}_n$. Let's say scaled spectrum is $\mathbf{s} = 2^n \mathbf{a}^T$.

Remark 3: Scaled spectrum value of a monomial is s_x where $1 \leq x \leq 2^n$ and $-2^n \leq s_x \leq 2^n$

Remark 4: s_x indicates similarity between \mathbf{f} and vector of this monomial \mathbf{m}_x which is x^{th} column in \mathbf{D}_n . Let's say y^{th} element of \mathbf{f} (\mathbf{f}_y) and \mathbf{m}_x (\mathbf{m}_{xy}) agrees if $\mathbf{f}_y \mathbf{m}_{xy} = 1$ and disagrees if $\mathbf{f}_y \mathbf{m}_{xy} = -1$. Then for instance, if $s_x = -2^n$ all elements of \mathbf{f} and \mathbf{m}_x disagree and if $s_x = (2^n - 2)$, $2^n - 1$ elements of \mathbf{f} and \mathbf{m}_x agree.

Function form $f: \{-1, 1\}^n \rightarrow \mathbb{R}$ defined in remark 1 could be converted into basic Boolean function form $f: \{-1, 1\}^n \rightarrow \{-1, 1\}$. Noticed easily, only output parts

are different, if there is such a function $g: \mathbb{R} \rightarrow \{-1,1\}$, then conversion is done. Examine $signum(sign)$ function:

$$sign(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases} \quad (6)$$

Intended output conversion could be made by $sign$ function if $sign(0)$ remains unused.

This operation turns function definition (1) to:

$$f(x_1, x_2, \dots, x_n) = sign\left(\sum_{i=1}^{2^n} a_i \prod_{k \in S_i} x_k\right) \text{ where } S_i \subset \{1, 2, \dots, n\} \quad (7)$$

Definition 2 (Sign representation): A polynomial function p sign-represents n -variable Boolean function f iff $f(x_1, x_2, \dots, x_n) = sign(p(x_1, x_2, \dots, x_n))$. This definition could be written in vector domain as $\mathbf{f} = sign(\mathbf{D}_n \mathbf{a})$ where \mathbf{a} is the vector of monomial coefficients of p .

Definition 3 (Standard form): $\mathbf{f} = sign(\mathbf{D}_n \mathbf{a})$ notation is simply equivalent to $diag(\mathbf{f})\mathbf{k} = \mathbf{D}_n \mathbf{a}$ where $diag(\mathbf{f})\mathbf{k}$ vector notation of sign representing polynomial output. $\mathbf{k}^T = [k_1, k_2, \dots, k_{2^n}]^T > \mathbf{0}$ and as a result $diag(\mathbf{f})\mathbf{D}_n \mathbf{a} > \mathbf{0}$. This inequality system is defined as standard form.

Equation (8) determines all the solutions of \mathbf{a} for a given \mathbf{f} .

$$\mathbf{a}^T = \mathbf{k}^T diag(\mathbf{f})\mathbf{D}_n \quad (8)$$

Equation (8) indicates that there are infinitely many solutions of \mathbf{a} as there are infinitely many positive \mathbf{k} . In prior definition of Boolean functions (1), there is only one solution for a given \mathbf{f} .

1.1.1 Higher Order Neuron Representation of Boolean Functions

Function form in equation (7) is so called sigma-pi unit model of biological neurons or higher-order neurons. Biological neurons are simply composed of a number of dendrites, an axon and neuron body. Axons of neurons are connected to other neurons' dendrites in brain. Neuron fires (sends a signal with a specific voltage level) or does not fire upon reaching the cell body. Resemblances can be noticed immediately, if a neuron and sigma-pi unit are examined. Monomials are dendrites of a neuron, *sign* function is the policy that determines whether the neuron will fire or not (it is also called activation function) and the output is the action potential carried by the axon of the neuron to other neurons. Here coefficients of monomials correspond to the connection strengths (weights) of the dendrites. Figure-1 illustrates an example of a higher-order neuron that represents example Boolean function described in Table-1 and a biological neuron.

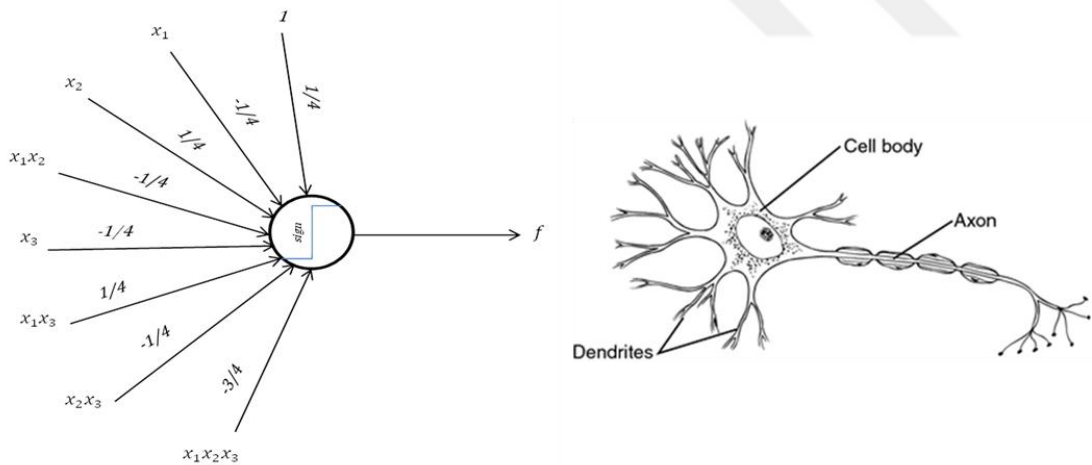


Figure 1: Higher-order neuron that represents Boolean function given in Table-1 and a biological neuron

1.2 Minimum Number of Monomials Problem of Polynomial Boolean Functions

There exists only one solution for the monomial coefficients vector of equation form (1); however, there could be many solutions for equation form (7). Then a tempting question arises: What is the minimum number of monomials that sign represents a given Boolean function (i.e. what is the maximum number of zeros in a monomial coefficients vector for a given Boolean function). Main subject of this dissertation is analyzing this problem and extracting results.

Briefly, we focus our attention on the infeasibility of the simultaneous absence of the set of monomials from any sign representation given a BF to represent. It is thought that the regularities observed then can be used to derive efficient algorithms for representing all or particular class of Boolean functions compactly with higher order neurons. There have been several studies for developing algorithms to find minimum monomial (low fan-in) solutions to given BFs (Ghosh and Shin 1992, Guler 2001, Oztop 2009). Mathematical results indicate that it is always possible to represent an n -variable BF with a higher order neuron that has at most 0.75×2^n input lines. In other words, at least 25% of the weights of higher order neuron can be zeroed; however, this bound is not tight, and thus algorithmic and theoretical improvements are needed (Oztop 2006).

Scrutiny is conducted on equivalent classes of Boolean functions. This way provides immense time efficiency without disturbing process of finding minimum monomial polynomial. Equivalent class concept is explained below.

1.2.1 Equivalency of Boolean Functions

The minimum number of monomials that will suffice to sign represent a Boolean function is not changed if the transformations 1 to 5(given below) are applied to a BF (Sezener and Oztop 2015).

1. Negation of input variables (e.g., $f(x_1, x_2, x_3) \rightarrow f(x_1, x_2, -x_3)$)
2. Permutation of input variables (e.g., $f(x_1, x_2, x_3) \rightarrow f(x_2, x_1, x_3)$)
3. Negation of the output (e.g., $f(x_1, x_2, x_3) \rightarrow -f(x_1, x_2, x_3)$)
4. XORing an input variable with other variables (e.g., $f(x_1, x_2, x_3) \rightarrow f(x_1, x_2 \oplus x_3 \oplus x_1, x_3)$)
5. XORing the function with input variables (e.g., $f(x_1, x_2, x_3) \rightarrow x_1 \oplus x_2 \oplus f(x_1, x_2, x_3)$)

These transformations were used in the spectral classification of BFs introduced by Edwards (1975). The first three of these transformations are usually known as NPN (negation-permutation-negation) transformations and their usage is more common. Equivalence classes over BFs could be created by applying any combination of these transformations. This means we could obtain any Boolean function from corresponding equivalence class of a BF by applying transformations 1-5. So a group of equivalent BFs could be represented by only one of them and if properties of the representing function are examined, we have idea about properties of all members in equivalent group. A natural labeling system for functions is used as illustrated in Table-3 by adopting a fixed ordering over the function arguments.

Table 3: The function output vector $[-1 \ 1 \ -1 \ 1 \ -1 \ 1 \ -1 \ 1]$ can be mapped (with $t(b)=(1-b)/2$) to the binary number 10101010 that is 0xaa in hexadecimal notation.

Arguments			Function
x_3	x_2	x_1	value
1	1	1	-1
1	1	-1	1
1	-1	1	-1
1	-1	-1	1
-1	1	1	-1
-1	1	-1	1
-1	-1	1	-1
-1	-1	-1	1



CHAPTER II

PREVIOUS WORK

There are many studies and researches on the minimum monomial sign representation problem. It has been investigated for years and important results are achieved. Studies depict boundaries on the problem and showed the possibility of a time efficient solution. However, even if the outcomes are milestones, still more research is required for finding the minimum monomial sign representation for a BF.

Guler (2001) introduces a new neural network model which is an expansion of higher order neuron representation. It is proposed that this model could learn higher order correlations and bypasses combinational explosion problem. Simulations were conducted on given training sets with this neural network model, and results are consistent with the proposal. This paper provides an algorithm for finding a sign representation of a given BF; however, it does not guarantee that the resulting sign representation has minimum number of monomials.

Oztop (2006) derived 3-quarters theorem which asserts any Boolean function could be represented with 0.75×2^n or less monomials where n is dimension. In other words a quarter of monomials could be absent from a sign representation. This theorem is remarkable, because it improved the upper bound for number of monomials that sign represents a given Boolean function for n dimensions. Also previous studies found upper bounds like $2^n - \sqrt{2^n} + 1$ (Gotsman 1989) and $2^n - 2^n/O(n)$ (O'Donnell and

Servedio 2003), but Oztop (2006) found a deterministic upper bound which is independent from n .

Oztop (2009) designed several algorithms which guarantee finding a sign representation of a given BF which has 0.75×2^n or less monomials. Constructed algorithms based on 3-quarters theorem. Instead of targeting time and space efficiency, this study provided a starting point for more efficient algorithms and improved theoretical results. Also this paper gives results about applications of 3-quarters theorem.

Amano (2010) found an approximate upper bound 0.617×2^n for number of sign representing monomials. However, provided upper bound is not valid for every BF, it is valid for almost every BF (a major group of BFs). This upper bound is found by iterating 3-quarters theorem.

Cazé, Humphries and Gutkin (2012) use monomials (dendrites) in CNF and DNF forms separately. Moreover they define two different transfer (activation) functions: spiking and saturation transfer functions. Linearly non-separable Boolean functions (lnBFs, a group of BFs) are tried to be implemented with these monomial forms and transfer functions. They found, lnBFs could be realized with monomials in DNF form by spiking transfer functions. However, this is not possible with monomials in DNF form by saturating transfer functions. CNF form monomials could be used with both spiking and saturating transfer functions for implementing lnBFs. Results indicate that if CNF architecture is used with saturating transfer functions, implemented lnBFs can require exponential number of monomials. However if CNF or DNF architectures are used with spiking transfer functions always require a linear number of monomials.

Sezener and Oztop (2015) researched lower dimensions for finding the exact upper bounds in these dimensions. Results found in this research improved the results found in Oztop (2006) for lower dimensions. Research proved the upper bound in one dimension is 1, in two dimensions 3, in three dimensions 4, in four dimensions (for the first time) 9 and in five dimensions (for the first time) 11. The paper proved that in six dimensions, upper bound must be smaller than 26. Paper introduced a heuristic algorithm for reaching these exact results. Also in research equivalent classes of Boolean functions are used. Thanks to equivalent classes, the space that must be researched reduces significantly and research time decreases dramatically.

All these outcomes of these researches are significant and give insight about the state and the structure of the problem. However, all approaches are required to be improved. Because still, there is not any algorithm that finds sharp upper bounds for all dimensions. Even Oztop (2006) provides a general upper bound for all dimensions, Sezener and Oztop (2015) proposes that the sharper upper bounds could be found if we examine different dimensions separately. Then for comprehending the problem completely, we decided to find out what limits the minimum number of monomials that suffice to sign represent a BF.

CHAPTER III

ZEROABILITY PATTERNS OF MONOMIALS IN THE HIGHER ORDER NEURON REPRESENTATION OF BOOLEAN FUNCTIONS

In this chapter we are interested in this type of questions: given a Boolean function \mathbf{f} , can there be a sign-representation which does not include any of the terms $(m_{x_1}, m_{x_2}, \dots, m_{x_r})$. Because our goal is minimizing number of terms, eliminating as much as possible monomials, this drives us constructing \mathbf{a} which include maximum number of zeros when complying with $\mathbf{k}^T = [k_1, k_2, \dots, k_{2^n}]^T > \mathbf{0}$. As every single element of \mathbf{a} is a coefficient of a specific monomial and we are trying to maximize zeros in coefficient vector, actually we are talking about ‘zeroability’ of monomials.

Definition 4 (Zeroability): We know $\mathbf{a}^T = \mathbf{k}^T \text{diag}(\mathbf{f}) \mathbf{D}_n$ is the coefficients of a sign-representing polynomial for \mathbf{f} and let's say a subset of \mathbf{a}^T ($a_{z_1}, a_{z_2}, \dots, a_{z_r}$) are coefficients of a set of monomials ($m_{z_1}, m_{z_2}, \dots, m_{z_r}$). If there is a positive vector \mathbf{k}^T which results in $a_{z_1} = a_{z_2} = \dots = a_{z_r} = 0$, then corresponding monomials of these coefficients are designated as zeroable.

3.1 Numerical Investigations on Monomial Zeroability

The zeroability of monomials depends on solvability of a set of linear equations with a positivity constraint. There are infinitely many solutions for \mathbf{a}^T as mentioned before, since $\mathbf{a}^T = \mathbf{k}^T \text{diag}(\mathbf{f}) \mathbf{D}_n = \mathbf{k}^T \mathbf{Q}$ and for any $\mathbf{k}^T = [k_1, k_2, \dots, k_{2^n}]^T > \mathbf{0}$ there

is a solution. If monomials $(m_{x_1}, m_{x_2}, \dots, m_{x_r})$ will be eliminated, $\mathbf{k}^T \mathbf{R} = \mathbf{0}$ must be satisfied, where \mathbf{R} is composed of $q_{x_1}, q_{x_2}, \dots, q_{x_r}$ which are corresponding columns of monomials in \mathbf{Q} , simply \mathbf{Q} is super-matrix of \mathbf{R} . So there are r equations with 2^n positive unknowns. If $\mathbf{k}^T \mathbf{R} = \mathbf{0}$ is not satisfied by any positive \mathbf{k} , then we concluded that sign representation of f without monomials $m_{z_1}, m_{z_2}, \dots, m_{z_r}$ does not exist. Otherwise (if there exists a positive \mathbf{k}) $\mathbf{a}^T = \mathbf{k}^T \mathbf{Q}$ will be the coefficient vector of the sign representing polynomial (weights of the higher order neuron that represents \mathbf{f}). Here we are trying to find maximum r that satisfies $\mathbf{k}^T \mathbf{R} = \mathbf{0}$. Exhaustive search over all column subsets of \mathbf{Q} (i.e. subsets of system of linear equations $\mathbf{0}^T = \mathbf{k}^T \mathbf{Q}$) could be performed easily on a given BF for achieving this task, however it is costly. This cost limits applicability of this exhaustive search to lower dimension, due to the combinatorial explosion of the number of subsets. Total time of process is decreased dramatically when only equivalence classes of BFs are used; however this way also becomes insufficient for dimensions higher than 6 when exhaustive search method is used. If results found by exercising the smaller subsets of the linear equation system $\mathbf{0}^T = \mathbf{k}^T \mathbf{Q}$ in lower dimensions could be inferred with potential behavior of overall linear equation system (for all dimensions), then we could have ideas about behavior of this problem. Hence all subsets of linear equations of all equivalent classes in three and four dimensions were examined and zeroability behaviors of monomials for all functions are obtained in these dimensions. For this we used the following exhaustive search algorithm.

Algorithm 1.

1. Input: \mathbf{f} :function ($2^n \times 1$ vector)

2. Output:

a. zeroable_counter

b. zeroable_index_list (number of zeroable subsets, and the list of index sets that achieve this)

3. Initialization:

a. Dimension of the problem: $n = \log_2(\text{length}(\mathbf{f}))$

b. Compute \mathbf{D}_n ($2^n \times 2^n$ Sylvester-type Hadamard matrix)

c. Potential coefficients of the system of linear equations: $\mathbf{Q} = \text{diag}(\mathbf{f})\mathbf{D}_n$

d. zeroable_counter[i]=0; for all $1 \leq i \leq 2^n$

e. zeroable_index_list[i] = {} for all $1 \leq i \leq 2^n$

4. for $r = 1$ to 2^n

a. for all r -column subset \mathbf{R} of \mathbf{Q} ($\mathbf{R} = [\mathbf{Q}(z_1), \mathbf{Q}(z_2), \dots, \mathbf{Q}(z_r)]$ for $\{z_1, \dots, z_r\} \subset 2^{\{1,2,\dots,n\}}$)

i. **if** $\mathbf{k}^T \mathbf{R} = 0$ feasible (check with Linear Programming)

ii. zeroable_counter(r) = zeroable_counter(r) + 1

iii. add $\{z_1, z_2, \dots, z_r\}$ to the zeroable_index_list(r) list

iv. **endif**

b. endfor

5. if zeroable_counter(r) == 0

break, since there cannot be anymore zeroable solution

6. end

7. endfor

8. return zeroable_counter, zeroable_index_list

This algorithm is run on all equivalent classes in only 3 and 4 dimensions since its complexity is $O(2^{2^n})$ where n is dimension. Results are given in Table-4 and Table-5.

3-variable Boolean functions: There are 3 equivalent classes of Boolean functions in three dimensions. Algorithm 1 is run on these equivalent classes and the number of zeroable subsets is found and shown in Table-4. Since equivalent classes represent all BFs, this table covers all the information about zeroability patterns in this dimension.

Table 4: Number of zeroable subsets for each function class is given for each subset cardinality (3 dimensions)

Function Label	1-sized subsets	2-sized subsets	3-sized subsets	4-sized subsets	5-sized subsets	6-sized subsets	7-sized subsets	8-sized subsets
0xaa	7	21	35	35	21	7	1	0
0xab	8	21	35	28	0	0	0	0
0xac	8	22	28	17	4	0	0	0
Total	8	28	56	70	56	28	8	1

4-variable Boolean functions: There are 8 equivalent classes of Boolean functions in four dimensions. Again Algorithm 1 is run on each of these classes for finding the number of zeroable subsets which is shown in Table-5. Again, this table covers all the information about zeroability patterns for all 4-variable BFs.

Table 5: Number of zeroable subsets for each function class is given for each subset cardinality (4 dimensions)

Function Label	1-sized subsets	2-sized subsets	3-sized subsets	4-sized subsets	5-sized subsets	6-sized subsets	7-sized subsets	8-sized subsets
0xaa55	15	105	455	1365	3003	5005	6435	6435
0xab55	16	105	455	1365	3003	5005	6435	6420
0xbb55	16	113	483	1414	2996	4690	5426	4573
0xaba5	16	117	521	1551	3156	4356	4236	3084
0xaa55	16	114	484	1375	2772	4092	4488	3663
0xaba4	16	119	546	1675	3388	4113	3490	2124
0xab12	16	120	560	1740	3492	4077	2910	1425
0xac90	16	120	560	1760	3648	4096	1600	0
Total	16	120	560	1820	4368	8008	11440	12870

Function Label	9-sized subsets	10-sized subsets	11-sized subsets	12-sized subsets	13-sized subsets	14-sized subsets	15-sized subsets	16-sized subsets
0xaa55	5005	3003	1365	455	105	15	1	0
0xab55	4900	2688	840	0	0	0	0	0
0xbb55	2724	1085	259	28	0	0	0	0
0xaba5	1684	672	144	0	0	0	0	0
0xaa55	2200	946	276	49	4	0	0	0
0xaba4	928	256	32	0	0	0	0	0
0xab12	400	61	6	0	0	0	0	0
0xac90	0	0	0	0	0	0	0	0
Total	11440	8008	4368	1820	560	120	16	1

Zeroable subsets of linear equations (elements of which are zeroable all together otherwise it is a non-zeroable subset) are exhausted after a subset size as seen in the Table-4 and Table-5. Ratio of non-zeroable subsets to all subsets grows with increasing subset size and it becomes %100 after a subset cardinality. Non-zeroability could be inherited from lower dimensions. This means, if a subset is non-zeroable it could include a smaller size non-zeroable subset. Otherwise non-zeroability is introduced first time. This means, elements of it become non-zeroable when they are tried to be solved all together only. This brings the question: How many new non-zeroable subsets are introduced for each subset size? Result of analysis (obtained by computer search) is given in Table 6 and Table 7 for 3 and 4 dimensions respectively.

Table 6: Number of first time introduced non-zeroable subsets for each function class is given for each subset cardinality (3 dimensions)

Function Label	1-sized subsets	2-sized subsets	3-sized subsets	4-sized subsets	5-sized subsets	6-sized subsets	7-sized subsets	8-sized subsets
0xaa	1	0	0	0	0	0	0	0
0xab	0	7	0	7	0	0	0	0
0xac	0	6	0	0	0	0	0	0
Total	8	28	56	70	56	28	8	1

Table 7: Number of first time introduced non-zeroable subsets for each function class is given for each subset cardinality (4 dimensions)

Function Label	1-sized subsets	2-sized subsets	3-sized subsets	4-sized subsets	5-sized subsets	6-sized subsets	7-sized subsets	8-sized subsets
0xaa55	1	0	0	0	0	0	0	0
0xab55	0	15	0	0	0	0	0	15
0xbb55	0	7	0	7	0	0	0	0
0xaba5	0	3	0	34	0	16	0	3
0xaaff	0	6	0	0	0	0	0	0
0xaba4	0	1	0	54	16	96	0	0
0xab12	0	0	0	80	0	120	0	0
0xac90	0	0	0	60	0	192	0	0
Total	16	120	560	1820	4368	8008	11440	12870

Function Label	9-sized subsets	10-sized subsets	11-sized subsets	12-sized subsets	13-sized subsets	14-sized subsets	15-sized subsets	16-sized subsets
0xaa55	0	0	0	0	0	0	0	0
0xab55	0	0	0	0	0	0	0	0
0xbb55	0	0	0	0	0	0	0	0
0xaba5	0	0	0	0	0	0	0	0
0xaaff	0	0	0	0	0	0	0	0
0xaba4	0	0	0	0	0	0	0	0
0xab12	0	0	0	0	0	0	0	0
0xac90	0	0	0	0	0	0	0	0
Total	11440	8008	4368	1820	560	120	16	1

If we comment on these tables, new non-zeroable subsets do not emerge for all subset cardinalities, only in some of them they appear.

3.2 Pairwise Zeroability of Monomials From Sign Representations

Sign representing polynomial or higher order neuron representation of a given BF \mathbf{f} could be found by selecting a \mathbf{k} all elements of which is positive ($\mathbf{k} > 0$), since $\mathbf{a}^T = \mathbf{k}^T \text{diag}(\mathbf{f}) \mathbf{D}_n$ where \mathbf{a} is the coefficients of the sign representing polynomial (or the weights of the higher order neuron). Fourier-Motzkin elimination is a procedure for

eliminating the variables of a given inequality system. If a set of monomials could not be eliminated together by applying FM elimination method, sign representing polynomial of a given BF \mathbf{f} does not exist without these monomials. Hence we could understand system feasibility by checking result of FM elimination method (Chandru 1993). Here our inequality system is $\text{diag}(\mathbf{f})\mathbf{D}_n\mathbf{a}^T = \mathbf{k}^T > 0 \rightarrow \text{diag}(\mathbf{f})\mathbf{D}_n\mathbf{a}^T > 0$. Simply \mathbf{a}^T includes the variables that we trying to eliminate (if a variable is eliminated inside of this vector corresponding monomial will be automatically eliminated) and rows of $\text{diag}(\mathbf{f})\mathbf{D}_n$ matrix includes coefficients of each inequality (There are 2^n inequalities where n is dimension). Let $\mathbf{Q} = \text{diag}(\mathbf{f})\mathbf{D}_n$. FM elimination method could be applied to the a selected columns of \mathbf{Q} which could be shown as $\mathbf{C} = \{c_{i_1}, c_{i_2}, \dots, c_{i_r}\}$. If a column is eliminable, it must include at least one positive and one negative number together. After elimination, all elements of this column become zero. Since elements of eliminated column are coefficients of corresponding variable in \mathbf{a}^T in the system of inequalities, this variable is deleted from all inequalities, which means this variable becomes ineffective. After selected set of columns is eliminated from \mathbf{Q} , let's say \mathbf{Q} is converted to the \mathbf{Q}_{FM} . \mathbf{Q}_{FM} can be easily converted to a sign-representation where $m_{i_1}, m_{i_2}, \dots, m_{i_r}$ are zero by taking the row sum of \mathbf{Q}_{FM} ($\mathbf{a}^T = \mathbf{1}^T\mathbf{Q}_{\text{FM}}$) (Oztop 2006). One way of finding the minimal sign representation of a given BF is finding a \mathbf{C} , includes maximum number of columns and all of these columns are eliminable by FM elimination. However searching over \mathbf{Q} for finding such \mathbf{C} have a high time complexity which is $O(2^{2^n})$ where n is dimension (number of variables). If this computational complexity could be overcome, it will be quite helpful for solving minimal term sign representation problem. Then we asked the question: Could \mathbf{C} s with smaller number of columns be used to construct \mathbf{C} s with larger number of columns? After this approach this result is obtained:

Theorem 1: Let $\mathbf{Q} = \text{diag}(\mathbf{f})\mathbf{D}_n$ for a given n -variable BF \mathbf{f} and let's say we are trying to eliminate any three monomials by applying FM elimination method to corresponding three columns from \mathbf{Q} . If these three columns cannot be zeroed with FM elimination, a pair of columns in these three columns, which could not be zeroed with FM elimination, must exist.

Proof: An exhaustive search is conducted on all possible unique $r \times 3$ matrices with $1 \leq r \leq 8$ to prove Theorem 1. The number of 8×3 sized matrices, with elements only 1 or -1 , is 2^{24} which is a quite large number. However, when duplicate rows are removed and the row order is ignored, there can be only 255 possible unique matrices. There are $2^3 = 8$ 3-bit patterns (1×3 sized vectors with elements only 1 or -1) and this number is the amount of non-empty subsets of all possible 3-bit patterns ($2^3 - 1$).

Assume now, we try to eliminate any 3 columns from $\mathbf{Q} = \text{diag}(\mathbf{f})\mathbf{D}_n$ by using FM elimination. Let's say these 3 columns construct \mathbf{Q}_0 which is a $2^n \times 3$ sub-matrix of \mathbf{Q} . Eliminability of these three columns and feasibility of $\mathbf{Q}_0\mathbf{a} > \mathbf{0}$ are equivalent (i.e. if these three columns are eliminable, $\mathbf{Q}_0\mathbf{a} > \mathbf{0}$ is feasible). Matrix \mathbf{R} will be obtained if we take only one row from duplicate rows in \mathbf{Q}_0 (taking unique rows from \mathbf{Q}_0 , apparently \mathbf{R} is sub-matrix of \mathbf{Q}_0) and the infeasibility of $\mathbf{Q}_0\mathbf{a} > \mathbf{0}$ is equivalent to the infeasibility of $\mathbf{R}\mathbf{a} > \mathbf{0}$. \mathbf{Q}_0 has at most 8 unique rows, so there is only 255 many such \mathbf{R} matrices mentioned earlier. Therefore only a small number of possible \mathbf{R} matrices must be checked, regardless of problem dimension n . Hence the given theorem will be proved completely, if the claim of the theorem is verified with all possible \mathbf{R} matrices. We do this by doing the search on a computer with the following algorithm:

Algorithm 2.

1. $claim = true$;
2. for all possible unique \mathbf{R} made up of ± 1 (of size $r \times 3$ where $1 \leq r \leq 8$)
do
 - a. Apply FM elimination to the columns of \mathbf{R}
 - b. if FM cannot eliminate all 3-columns
 - i. For each pair of columns from \mathbf{R} , apply FM elimination
 - ii. If any pair can be eliminated, a counterexample is found so set $claim = false$
 - c. endif
3. endfor
4. return $claim$

The execution of the code indeed does not find any counterexample (i.e. the claim is returned as true). Thus, the theorem is proven.

Corollary 1: Let's say there is a subset of monomials N with three elements for any n -variable BF. If all two elements subsets of N are zeroable, then N is zeroable.

Proof: Assume the contrary. N is not zeroable, Theorem 1 asserts there must be at least one, two elements subset which is not zeroable, a contradiction.

Corollary 2: If sign representation of a BF f does not exist without a group of monomials $M = \{m_1, m_2, m_3\}$, it means sign representation of this function could not be constructed without at least one of $\{m_1, m_2\}$ or $\{m_2, m_3\}$ or $\{m_1, m_3\}$.

Proof: Assume the contrary. Let's say sign representation of this BF f could be constructed with absence of either $\{m_1, m_2\}$ or $\{m_2, m_3\}$ or $\{m_1, m_3\}$. This means all two

elements subsets of M could be zeroed. If so, M is zeroable according to Corollary 1. Hence sign representation \mathbf{f} exists without M . A contradiction.

Whether this kind of relation exists for greater number of monomials is a question worth asking. For instance: If any 5 monomials could not be zeroed together for any BF, all the time are there any 4 monomials which could not be zeroed together already in this 5 monomials (5 monomials sets never introduce a new infeasibility)? However this is not true, there are counterexamples for infeasibility of 4 and 5 monomials. For example, the 4-variable Boolean function \mathbf{f} defined by $f(x_1, x_2, x_3, x_4) = (x_1 \text{ AND } x_2) \text{ XOR } (x_3 \text{ AND } x_4)$, which is a bent function and labeled as 0x111e in hexadecimal and (0001000100011110 in binary). This function could not be sign represented without monomials $M = \{x_1, x_2, x_1x_3, x_2x_3\}$. However, it is possible to construct sign representation of this function without any 3 elements subset of M (Zeroing 4 monomials creates a new infeasibility). Such sign representations are:

Sign representation 1 (x_1, x_2, x_1x_3 are absent):

$$689 - 689x_1x_2 + 689x_3 + 1056x_2x_3 - 689x_1x_2x_3 + 689x_4 + 977x_1x_4 + 977x_2x_4 - 689x_1x_2x_4 - 689x_3x_4 - 977x_1x_3x_4 - 977x_2x_3x_4 + 689x_1x_2x_3x_4$$

Sign representation 2 (x_1, x_2, x_2x_3 are absent):

$$689 - 689x_1x_2 + 689x_3 + 1056x_1x_3 - 689x_1x_2x_3 + 689x_4 + 977x_1x_4 + 977x_2x_4 - 689x_1x_2x_4 - 689x_3x_4 - 977x_1x_3x_4 - 977x_2x_3x_4 + 689x_1x_2x_3x_4$$

Sign representation 3 (x_1, x_1x_3, x_2x_3 are absent):

$$977 + 1056x_2 - 977x_1x_2 + 977x_3 - 977x_1x_2x_3 + 689x_4 + 689x_1x_4 + 689x_2x_4 - 689x_1x_2x_4 - 689x_3x_4 - 689x_1x_3x_4 - 689x_2x_3x_4 + 689x_1x_2x_3x_4$$

Sign representation 4(x_2, x_1x_3, x_2x_3 are absent):

$$689 + 1056x_1 - 689x_1x_2 + 689x_3 - 689x_1x_2x_3 + 977x_4 + 689x_1x_4 + 689x_2x_4 - 977x_1x_2x_4 - 977x_3x_4 - 689x_1x_3x_4 - 689x_2x_3x_4 + 977x_1x_2x_3x_4$$

Another example is a 4-variable BF labeled as 0xda2a in hexadecimal (1101101000101010 in binary). This function could not be sign represented without monomials $M = \{x_1, x_4, x_1x_4, x_2x_4, x_1x_2x_4\}$. But it is possible to construct sign representation of this function without any 4 elements subset of M (Zeroing 5 monomials creates a new infeasibility). Such sign representations are:

Sign representation 1(x_1, x_4, x_1x_4, x_2x_4 are absent):

$$16282 + 7472x_2 - 10298x_1x_2 + 9445x_3 + 42488x_1x_3 - 5706x_2x_3 - 757x_1x_2x_3 - 27876x_1x_2x_4 - 13290x_3x_4 + 19753x_1x_3x_4 - 28441x_2x_3x_4 - 23492x_1x_2x_3x_4$$

Sign representation 2($x_1, x_4, x_1x_4, x_1x_2x_4$ are absent):

$$16282 - 10298x_2 + 7472x_1x_2 + 9445x_3 + 42488x_1x_3 - 757x_2x_3 - 5706x_1x_2x_3 - 27876x_2x_4 - 13290x_3x_4 + 19753x_1x_3x_4 - 23492x_2x_3x_4 - 28441x_1x_2x_3x_4$$

Sign representation 3($x_1, x_4, x_2x_4, x_1x_2x_4$ are absent):

$$-7472 - 10298x_2 - 16282x_1x_2 + 5706x_3 + 42488x_1x_3 - 757x_2x_3 - 9445x_1x_2x_3 + 27876x_1x_4 - 13290x_3x_4 + 23492x_1x_3x_4 - 19753x_2x_3x_4 - 28441x_1x_2x_3x_4$$

Sign representation 4 ($x_1, x_1x_4, x_2x_4, x_1x_2x_4$ are absent):

$$- 7472 + 16282x_2 + 10298x_1x_2 + 5706x_3 + 42488x_1x_3 + 9445x_2x_3 + 757x_1x_2x_3 - 27876x_4 - 23492x_3x_4 + 13290x_1x_3x_4 - 19753x_2x_3x_4 - 28441x_1x_2x_3x_4$$

Sign representation 5 ($x_4, x_1x_4, x_2x_4, x_1x_2x_4$ are absent):

$$- 5400 - 1322102x_1 + 5400x_3 + 1322102x_1x_3 - 658351x_3x_4 + 658351x_1x_3x_4 - 663751x_2x_3x_4 - 663751x_1x_2x_3x_4$$

3.3 Pairwise Zeroability and Incompatibility Graph

Finding sign representation (higher order neuron representation) of a BF with possible minimum number monomials requires searching over all possible monomial subsets. If a method that uses pairwise zeroability of monomials, could be developed, it could be possible to decrease the steps to obtain a sign representation. So, a formal graph concept based on the notion of non-zeroability is defined.

Definition 5 (Incompatibility Graph): A graph is defined as $\mathbf{G}_f = (V, E)$ where V stands for vertices and E stands for edges. In an incompatibility graph of an n -variable BF f , each vertex is identified by a monomial of the form $m_i = \prod_{k \in S_i} x_k$ where $S_i \subset \{1, 2, \dots, n\}$, and there is an edge e_{ij} between two vertices v_i and v_j if only if m_i and m_j is not zeroable together.

With this definition we can give this simple Lemma.

Lemma 4: Given an n -variable Boolean function f , 3-vertex independent sets of \mathbf{G}_f are always zeroable.

Proof: Incompatibility graph is give information about pairwise zeroability. In a 3-vertex independent set, all elements are pairwise zeroable. So these three elements are zeroable together. Corollary 1 verifies this result.

The lemma above proposes that beginning search from independent sets will prevent complexity of searching process for zeroable monomials. Moreover this lemma suggests that dense subsets like cliques should be avoided when searching for zeroable monomial subsets. The problems of finding maximum independent set and maximum clique problems are computationally equivalent. Let's say S is a maximum independent set in graph G , in complementary graph of G , S is a maximum clique. There several efficient heuristic algorithms(e.g. Busygin, Butenko et al. 2002) for solving the problem of finding maximum independent sets (and so is finding maximum cliques) even it is np-hard. Briefly we suggest: Find the maximum independent set in an incompatibility graph and begin searching for zeroable monomial subsets from it. There are example graphs below. Consider the graph G_f given in Figure-2 which is incompatibility graph of 0xaaff. G_f has 16 elements, 12 of them constructs the maximum independent set in the graph, then 4 of them constructs a 4-clique, the monomials which construct this clique are $\{1, x_1, x_4, x_1x_4\}$ and this means these monomials cannot be eliminated together. All elements in the independent set could be zeroed and we could add only one of the elements in the clique to this zeroable subset to construct the largest zeroable monomial subset. This means the minimum sign representation of this class BFs could be constructed with any three element subset of unzeroable monomials subset (shown by the computer search).

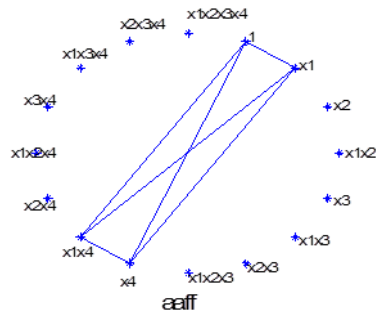


Figure 2: Incompatibility graph of Class 0xaaff

When the incompatibility graph of class 0xaba5 (in the Figure 3) is examined, it could be seen that the monomial x_1x_3 cannot be zeroed with three others which are $\{x_1, x_1x_4, x_1x_3x_4\}$ and they do not form a clique. It is found that all minimum sign representation of this class must include x_1x_3 and two other monomials from $\{x_1, x_1x_4, x_1x_3x_4\}$.

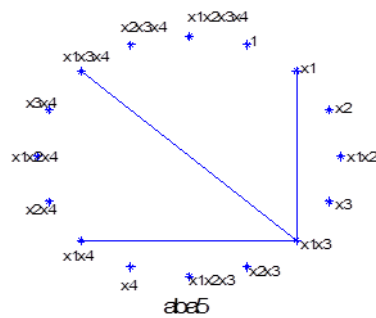


Figure 3: Incompatibility graph of Class 0xaba5

The incompatibility graphs of class 0xaa55 given in Figure 4 and 0xab55 given in Figure 5 are same. It is interesting since they are different classes of Boolean functions. 0xaa55 could be sign represented with one monomial only (monomial's itself); however minimum number of monomials that is required to sign represent of class 0xab55 is five.

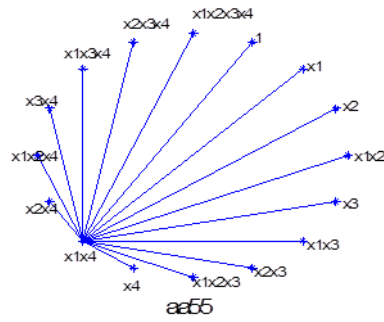


Figure 4: Incompatibility graph of Class 0xaa55

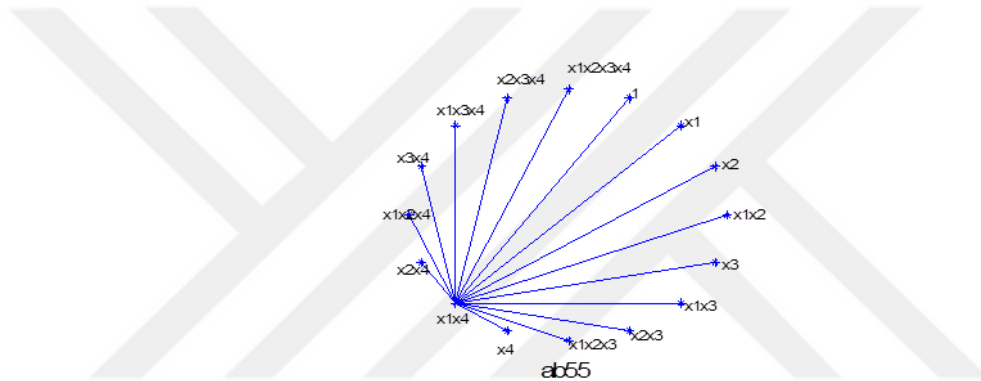


Figure 5: Incompatibility graph of Class 0xab55

There are also incompatibility graphs all elements of which are independent (i.e. there is not any edge between vertices). These graphs indicates all monomials of these functions are three-wise zeroable (from Corollary 1). Examples are graphs of class 0xab12 (Figure 6) and 0xac90 (Figure 7).

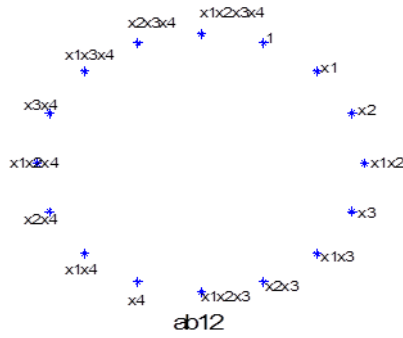


Figure 6: Incompatibility graph of Class 0xab12

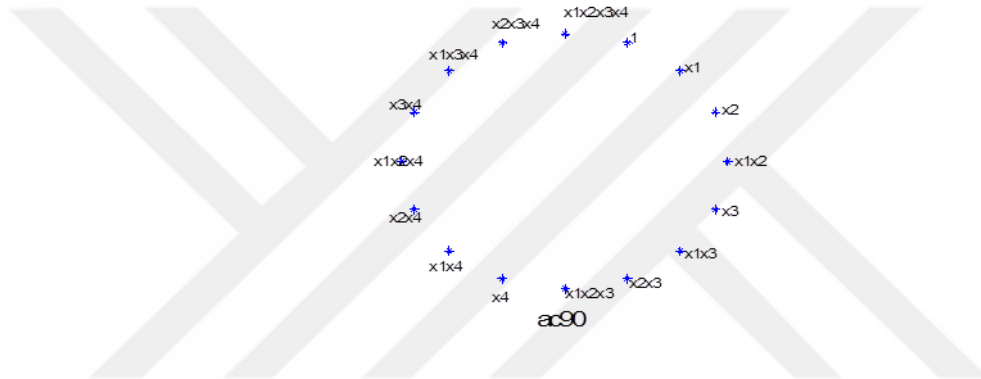


Figure 7: Incompatibility graph of Class 0xac90

These observations give clues about importance of incompatibility graphs. Incompatibility graphs could give patterns and hints about minimal sign representation of a BF. A final note on the incompatibility graph is that all the Boolean functions from a single equivalence class as defined via the transformations 1-5 of Section 1.2.1 have isomorphic incompatibility graphs. However, the reverse of this remark is not true.

CHAPTER IV

CONCLUSION

Boolean functions (BFs) have usages in neuroscience, cryptography, computer science and circuit design. For example sign representation of BFs could be used as a neuron model, BFs are essential in ALU (arithmetic logic unit) design and programming basics. Higher order neuron representations or sign representing polynomials with less number of monomials can have superiorities over the ones with more monomials. For instance sign representations with less number of monomials, may facilitate faster learning. However efforts spent for finding optimal polynomials could be too much time consuming, therefore using any sign representing polynomial for this purpose may be more beneficial (Guler 2001). The reason of this is the exponential explosion while solving minimum monomial polynomial problem since there is not any polynomial time algorithm for the solution. Considering this, the problem of representing BFs with possible minimum monomials is still waiting for an efficient heuristic algorithm to be found, which may widen the use of higher order neurons in neural networks and learning applications.

To this end we focused on this topic and tried to understand the structure of problem to make a contribution towards the development of a heuristic algorithm for finding compact sign-representations. First we examined which monomials could not be zeroable together for all equivalent classes in three and four dimensions. It is concluded that zeroable subsets vanish after a subset size and this determines minimum number of

monomials that is sufficient to represent a BF (If we zeroed maximum number of monomials possible the rest is the minimum number of monomials that suffice for writing sign representing polynomial). Let's say a subset of monomials is non-zeroable together. It means in this group there is a smaller group of monomials which are already non-zeroable (non-zeroability could be inherited) or this group of monomials are non-zeroable if and only if they are tried to be zeroed (new zeroable-subsets) together. In this sense, we searched for new non-zeroable monomial subsets and classified them with their subset sizes for each equivalent class. The purpose was to find patterns of zeroability or non-zeroability of monomials in lower dimensions and extracting general results from these. Then, it is noticed that three monomial subsets do not introduce new non-zeroabilities in three and four dimensions. After a research we proved that if three monomials could not be zeroed together, then it exists the case that at least one pair of monomials that cannot be zeroable together already in these three monomials, for any n -variable Boolean function. Same observation could not be made for bigger subset sizes and counter examples are given for subset size of four and five. Also pairwise non-zeroability is visualized with incompatibility graphs. All the functions from a single equivalence class must have isomorphic incompatibility graphs; however, the reverse is not true. We examined incompatibility graphs of equivalent classes and realized that some of them give important clues about minimum monomial sign representing polynomial. Incompatibility graphs should be extended to provide more clues on zeroability patterns. This way, they can be used to construct heuristic algorithms for obtaining close-to-minimum monomial sign representations.

In particular, the non-zeroability and zeroability relations of monomial subsets with cardinality larger than 2, should be investigated. Methods applied to BFs in three

and four dimensions could also be tried in higher dimensions if it is possible. However, these methods will take unaffordably long time after a dimension or subset size. For example, there are 601080390 16-monomials subsets for a 5-variable Boolean function, which means this many linear equations should be tested for zeroability. Each test takes 6 milliseconds on average with a 2.7 GHz CPU and 8 GB RAM PC, and the required overall time for all tests is 1000 hours for a 5-variable BF and subset size 16. Considering there are 48 equivalent classes in 5 dimensions (Sezener and Oztop 2015), it takes 2000 days to complete all zeroability tests. So, our brute-force algorithm must be improved for providing time affordability. As another option, cloud computing concept could be used to run our algorithm for higher dimensions. There are public providers for cloud computing services and cloud computing will dramatically decrease required time to run our algorithm. But after a dimension, even this method will become ineffective. Moreover, we already observed that new non-zeroable subsets do not emerge for all subset cardinalities, only in some of them they appear. This result could be used to improve our algorithm. For example if we don't check zeroability of subsets that include a smaller non-zeroable subset already (then obviously including subset is non-zeroable), there could be time improvements. The time spent for finding such subsets must be measured to verify this proposal, because this time could exceed the time spent for checking zeroability. However, if certain results and improvements are desired, the reasons behind zeroability of some monomials and non-zeroability of some of them should be investigated. While doing this, we should keep in mind that the values of monomial coefficients are dependent to linear equations which are not constructed randomly, but depend on the standard form (Equation (8)).

APPENDIX A

INCOMPATIBILITY GRAPHS OF REMAINING CLASSES

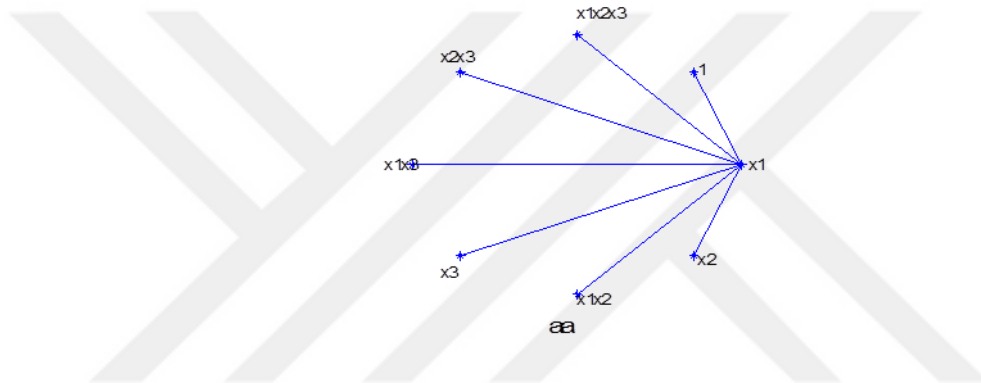


Figure 8: Incompatibility graph of Class 0xaa

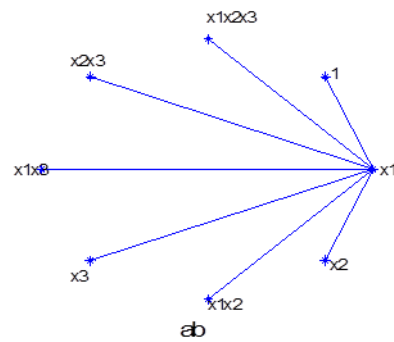


Figure 9: Incompatibility graph of Class 0xab

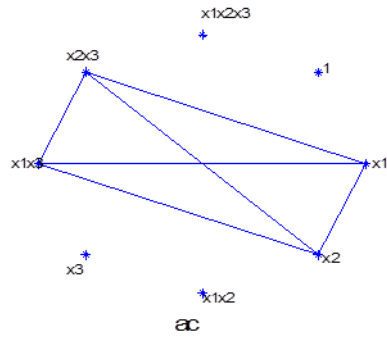


Figure 10: Incompatibility graph of Class 0xac

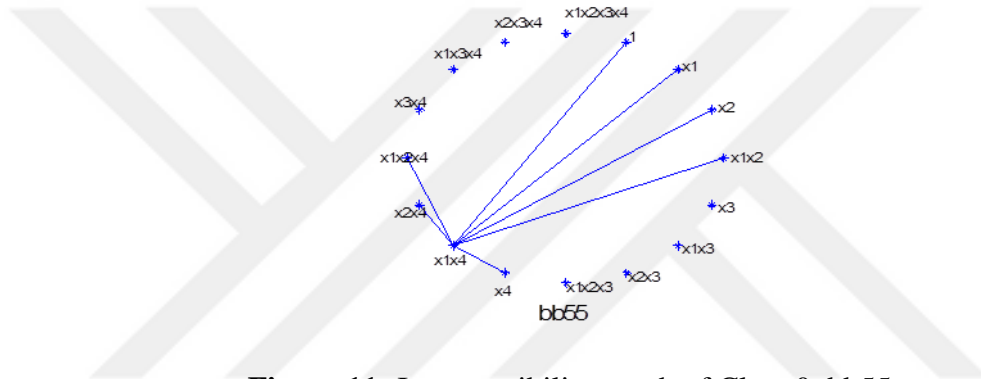


Figure 11: Incompatibility graph of Class 0xbb55

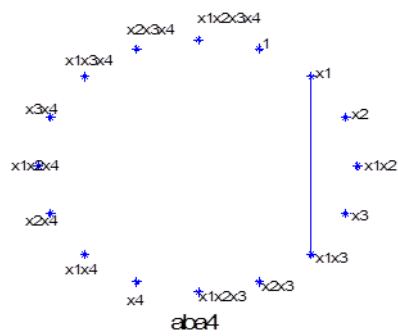


Figure 12: Incompatibility graph of Class 0xaba4

BIBLIOGRAPHY

- [1] Busygin, S., S. Butenko and P. M. Pardalos (2002). "A Heuristic for the Maximum Independent Set Problem Based on Optimization of a Quadratic Over a Sphere." Journal of Combinatorial Optimization **6**(3): 287-297.
- [2] Chandru, V. (1993). "Variable Elimination in Linear Constraints." The Computer Journal **36**(5): 463-470.
- [3] Ghosh, J. and Y. Shin (1992). "Efficient Higher Order Neural Networks for Classification and Function Approximation." International Journal of Neural Systems **3**(4): 323-350.
- [4] Guler, M. (2001). "A model with an intrinsic property of learning higher order correlations." Neural Networks **14**(4-5): 495-504.
- [5] Oztop, E. (2006). "An Upper Bound on the Minimum Number of Monomials Required to Separate Dichotomies of $\{-1, 1\}^n$." Neural Computation **18**(12): 3119-3138.
- [6] Oztop, E. (2009). "Sign-representation of Boolean functions using a small number of monomials." Neural Networks **22**(7): 938-948.
- [7] Sezener, C. E. and E. Oztop (2015). "Minimal Sign Representation of Boolean Functions: Algorithms and Exact Results for Low Dimensions." Neural Comput **27**(8): 1796-1823.
- [8] Siu, K. Y., V. Roychowdhury and T. Kailath (1995). Discrete Neural Computation. Englewood Cliffs, NJ, Prentice Hall.
- [9] Gotsman, C. (1989). On Boolean functions, polynomials and algebraic threshold functions. (Tech. Rep. TR-89-18). Tal Aviv: Department of Computer Science, Hebrew University.
- [10] O'Donnell, R., & Servedio, R. (2003). Extremal properties of polynomial threshold functions. In Eighteenth Annual Conference on Computational Complexity (pp. 3–12). Piscataway, NJ: IEEE Computer Society.
- [11] Amano, K. (2010). New upper bounds on the average PTF density of Boolean functions. In O. Cheong, K.-Y. Chwa, & K. Park (Eds.), Algorithms and computation (pp. 304–315). New York: Springer.
- [12] Cazé RD, Humphries M, Gutkin BS (2012) Spiking and saturating dendrites differentially expand single neuron computation capacity. Advances in neural information processing systems **25**: 1079–1087.