Fall 2014

# An Analysis of DNF Maximum Entropy

Belinda Hasanaj

# AN ANALYSIS OF DNF MAXIMUM ENTROPY

A Thesis

Submitted to the McAnulty College and Graduate School of Liberal Arts

Duquesne University

In partial fulfillment of the requirements for

the degree of Masters of Science in Computational Mathematics

By

Belinda Hasanaj

December 2014

**AN ANALYSIS OF DNF MAXIMUM ENTROY**

By

Belinda Hasanaj

Approved November 20, 2014

APPROVED________________________________________
Karl Wimmer, Ph.D., Assistant Professor
Department of Mathematics & Computer Science

APPROVED________________________________________
Jeffrey Jackson, Ph.D., Director of Graduate Studies
Department of Mathematics & Computer Science

APPROVED________________________________________
Abhay Gaur, Ph.D., Professor
Department of Mathematics & Computer Science

APPROVED________________________________________
James C. Swindal, Ph.D., Dean
McAnulty College and Graduate School of Liberal Arts

ABSTRACT

AN ANALYSIS OF DNF MAXIMUM ENTROPY

By

Belinda Hasanaj

December 2014

Thesis supervised by Karl Wimmer, Ph.D., Assistant Professor

This study focuses on the entropy of functions computed by monotone DNF formulas. Entropy, which is a measure of uncertainty, information, and choice, has been long studied in the field of mathematics and computer science. We will be considering spectral entropy and focus on the conjecture that for each fixed number of terms $t$, the maximum entropy of a function computed by a $t$-term DNF is achieved by a function computable by a read-once DNF. A Python program was written to first express the $t$-term DNF Boolean functions as multilinear polynomials and then to compute their spectral entropy. This was done for the cases $t = 1, 2, 3, 4$. Our results agree with the conjecture and show that the maximum entropy occurs for functions with a small number of literals.

## ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Dr. Karl Wimmer for all his support, and patience while teaching me these last two years. I have learned so much by working with him. Also, his ideas and guidance were crucial for the completion of the thesis. Additionally, I am grateful to Dr. Jeffrey Jackson and Dr. Abhay Gaur for their valuable suggestions and support.

Finally, I would like to thank my family, who have encouraged and supported me throughout my continuing education.

**TABLE OF CONTENTS**

# Chapter 1

# Introduction

Since first introduced by Fourier and Chebyshev, the approximation of functions by algebraic and trigonometric polynomials has been an important topic of mathematical studies. Also, it has provided powerful mathematical tools to other application areas like data representation, signal processing, numerical analysis, and solution of differential equations. The need for an approximation form, especially in theoretical computer science, arises from the idea that even though in principle one could calculate the function value for any given set of arguments, it is actually computationally expensive to do so. Therefore, rather than computing the function exactly, sometimes it is much easier to approximate it by a simpler function. The class of polynomials is possibly the simplest mathematical class of functions, since they require only multiplications and additions for their evaluation.

Consider two Boolean functions $f$ and $g$. We say that $f$ approximates $g$ if the fraction of inputs in $\{0,1\}^n$ on which f and g disagree is at most $\epsilon$: $Pr\{f(x) \neq g(x)\} \leq \epsilon$. All probabilities in this thesis are with respect to the uniform distribution unless otherwise specified. Many applications are encouraging for the study of the approximation degree as a complexity measure.

The study of the approximate degree of Boolean functions began first in 1969, presented by the work of Minsky and Papert [19], which proved that the parity function in $n$ variables can not be approximated by a polynomial of degree less than $n$. It is also used to solve problems in complexity theory and algorithm design. Recently, in computational learning theory, it has been shown to be useful in obtaining the best known algorithms for PAC (Probably Approximate Correct) learning DNF formulas, read-once formulas [1, 15, 24] and for agnostically learning disjunctions [12]. A great amount of work has been done in the last twenty years on developing algorithms that find polynomials $p$ such that: $E_{x\in\{0,1\}^n}[(p(x)-f(x))^2]\leq\epsilon$ for a given function $f$. The first progress was made by Linial *et al.* [16] , who proved that for any function computed by a $t$-term DNF formula $f$, there exists a polynomial $p:\{0,1\}^n\rightarrow\{0,1\}$ of degree $O(\log(t/\epsilon)^2)$ for which $E_{x\in\{0,1\}^n}[(p(x)-f(x))^2]\leq\epsilon$. In [16], it was shown that this approximation implies a quasipolynomial-time algorithm for the PAC-learning DNF formulas with respect to the uniform distribution. Furthermore, it was noticed that in fact this implies a stronger result, a sub-exponential time agnostic algorithm for learning disjunctions under any distribution [12].

In 1995, Mansour [18] proved that for any DNF formula with $t$ terms, there exists a polynomial $p$ with sparsity (number of nonzero coefficients of $p$) $t^{O(\log\log t\log(1/\epsilon))}$, that approximates $f$ to error $\epsilon$. This implied a nearly polynomial-time query algorithm for PAC learning DNF formulas under the uniform distribution. Then, he conjectured that this bound can be improved to $t^{O(log 1/\epsilon)}$ (known as Mansour Conjecture) [17], which means that most of the fourier coefficients are concentrated only on polynomial number of coefficients in $t$ for constant $\epsilon$. The importance of the conjecture relies on the fact that if it is true, it would imply a polynomial-time query algorithm for learning DNF formulas with respect to the uniform distribution. It was proven by Dr. Jackson [10] that such an algorithm exists, but this was done using the "Harmonic Sieve" algorithm and without proving the Mansour Conjecture. Also, another implication of the Mansour Conjecture is that the query algorithm of Gopalan *et al.* [7, 8] would agnostically learn DNF formulas under the uniform distribution, to within any constant accuracy, in polynomial time. Therefore, proving this result is a major open question in computational learning theory.

Major progress towards proving Mansour Conjecture was made in 2010, by Klivians *et al.* [14]. They showed that the conjecture is true for almost all DNF formulas and read-k DNF formulas.

Another important open problem related to the Mansour Conjecture is the Fourier Entropy-Influence Conjecture (FEI) made by Friedgut and Kalai in 1996 [6, 13]. This conjecture, which states that $H[f] \leq C \cdot Inf[f]$, relates two important measures of Boolean function complexity, the total influence and the spectral entropy. The conjecture was first initiated by a study of threshold phenomena in random graphs [6]. (In this introduction, we use many technical terms that we will formally define later in the thesis.)

Even though the interest shown in the FEI Conjecture over the past years has been large, the FEI conjecture has been proven to be true only for a few classes of Boolean functions. From direct calculations with the Fourier coefficient, it is easy to check that FEI Conjecture is true for the usual Boolean functions: AND, OR, Majority and Tribes. Also, the conjecture holds for the family of symmetric functions ($C = 12.04$), d-part symmetric functions ($C = 12.04 + \log_2 d$) and read-once DNF(C = 4.88) [5]. Beside those, there is a large but non-explicit family of functions that satisfies the Fourier Entropy Influence Conjecture [3]. It is already known that if $f$ is computable by a $t$ term DNF, then the total influence is bounded by $O(\log(t))$ [2]. For this reason, Mansours Conjecture is implied by the FEI Conjecture.

In this thesis, we study the maximum entropy of functions computed by $t-$term DNF formulas ($t = 1, 2, 3, 4$), and try to relate our results with the Fourier Entropy-Influence Conjecture and Mansour Conjecture. Entropy seems to be different than other complexity measures since it might be tricky to use with current techniques. A conjecture by Dr. Wimmer (unpublished) states that for each fixed number of terms $t$, the maximum entropy of a function computed by a $t$ term DNF formula is achieved by a function computable by a read-once DNF. The work done in this thesis mostly focuses on the above conjecture, which seems not to have been studied before.

The remainder of the thesis is laid out as follows. Chapter 2 presents background information on the field of analysis of Boolean functions. This chapter also includes definitions about entropy and DNF formulas. In the third chapter we consider the main theorems and conjectures to which our research is closely related. We conclude the thesis by discussing the results of the experiments performed, and different ideas for related future work.

# Chapter 2

# Background

## 2.1 Fourier Transform of Boolean Functions

Fourier transforms are often used in mathematics, computer science and engineering. When studying Boolean functions with $n$ variables, the Fourier transform is a linear map of the values of the function into the set of coefficients and it is considered over the Abelian group $\mathbb{Z}_2^n$. This is widely known as Fourier analysis over the Boolean cube, and has become one of the most important techniques for theoretical computer science and applied mathematics. By analysis of Boolean functions, we aim to obtain information and study the structural properties of Boolean functions using their Fourier expansion. For a wide overview in the area see the book [4].

We consider Boolean functions, which map length-n binary vectors, or strings, into a single binary value or bit. The bits can be represented as True and False, 1 and 0, -1 and 1 ( where -1 represents True). Mostly, throughout the thesis we will use the convention {-1, 1} and the Boolean function will be:

$$f : \{-1, 1\}^n \to \{-1, 1\}. \tag{2.1}$$

**Definition 2.1.1.** *For functions $f, g : \{-1,1\}^n \to \{-1,1\}$, the inner product is*

$$< f, g >= \frac{1}{2^n} \sum_{x \in \{-1,1\}^n} f(x)g(x) = E_{x \sim \{-1,1\}^n}[f(x)g(x)]$$

*where the sum $\sum f(x)g(x)$ is the correlation between $f, g$ and it is a measure of how often $f$ and $g$ agree.*

Let $\chi_S : \{-1,1\}^n \to \{-1,1\}$ be the parity function, which is defined as $\chi_S = \prod_{i \in S} x_i$.

**Theorem 2.1.2.** *The $2^n$ parity functions $\chi_S : \{-1,1\}^n \to \{-1,1\}$ form an orthonormal basis for the vector space $V$ of functions:*

$$< \chi_S, \chi_T >= \begin{cases} 1 & if S = T \\ 0 & if S \neq T \end{cases}$$

*Proof.* If $S = T$, $< \chi_S, \chi_S >= \frac{1}{2^n} \sum_{x \in \{-1,1\}^n} \chi_S(x)^2 = 1$.

If $S \neq T$, then

$$< \chi_S, \chi_T >= E_{x \sim \{-1,1\}^n}[\chi_S \cdot \chi_T] = E_{x \sim \{-1,1\}^n}[\chi_{S \triangle T}]$$

Since $S \neq T$, then $S \triangle T \neq 0$. Therefore $E_{x \sim \{-1,1\}^n}[\chi_{S \triangle T}] = 0$. Hence, the parity functions form an orthonormal basis. □

We can now define another function $\widehat{f} : \{-1,1\} \to \mathbb{R}$ by $\widehat{f}(S) = \langle f, \chi_S \rangle = E_{x \sim \{-1,1\}^n}[f(x)\chi_S(x)]$. These are known as the Fourier Coefficients, which measure the correlation between the function $f$ and a specific parity function $\chi_S$ under the uniform distribution. The function $\mathfrak{F} : f \mapsto \widehat{f}$ is called the Fourier Transform and is an invertible linear mapping of the values of the function onto a set of coefficients.

Let's now present the Fourier expansion theorem:

**Theorem 2.1.3.** *Every function $f : \{-1,1\}^n \to \mathbb{R}$ can be uniquely expressed as a multilinear polynomial*

$$f = \sum_S \widehat{f}(S)\chi_S$$

*The above expression represents the Fourier expansion of $f$, while the real numbers $\widehat{f}(S)$ are the Fourier coefficient of $f$ on $S$.*

Let $f : \{-1,1\}^n \to \{-1,1\}$ be a Boolean function. We will show through an example, how to express it in the multilinear polynomial form. We need to find a polynomial which interpolates the $2^n$ values that $f$ assigns to the points $\{-1,1\}^n \subset \mathbb{R}^n$. Define the indicator polynomial to be:

$$1_{\{a\}}(x) = (\frac{1+a_1x_1}{2})(\frac{1+a_2x_2}{2})\cdots(\frac{1+a_nx_n}{2}) \tag{2.2}$$

It will take value 1 when $x = a$ and value 0 when $x \in \{-1,1\}^n \setminus \{a\}$. Therefore, the polynomial representation of $f$ is

$$f(x) = \sum_{a\in\{-1,1\}^n} f(a)1_{\{a\}}(x) \tag{2.3}$$

For instance, consider the AND function on 2 variable. We know that $AND(+1,+1) = +1, AND(+1,-1) = +1, AND(-1,+1) = +1, AND(-1,-1) = -1$, therefore

$$\begin{aligned} AND(x_1,x_2) = (+1)\left(\frac{1+x_1}{2}\right)\left(\frac{1+x_2}{2}\right) + (+1)\left(\frac{1+x_1}{2}\right)\left(\frac{1-x_2}{2}\right) \\ + (+1)\left(\frac{1-x_1}{2}\right)\left(\frac{1+x_2}{2}\right) + (-1)\left(\frac{1-x_1}{2}\right)\left(\frac{1-x_2}{2}\right) \end{aligned}$$

After finishing the calculations, the AND function written as a multilinear polynomial is $AND(x_1,x_2) = \frac{1}{2} + \frac{1}{2}x_1 + \frac{1}{2}x_2 - \frac{1}{2}x_1x_2$.

One of the important properties of the Fourier transform is the following theorem.

**Theorem 2.1.4.** *(Plancherel Theorem) Given two functions* $f, g : \{-1,1\}^n \to \mathbb{R}$,

$$\langle f, g\rangle = E_{x\sim\{-1,1\}^n}[f(x)g(x)] = \sum_{S\subseteq[n]} \widehat{f}(S)\widehat{g}(S). \tag{2.4}$$

This theorem is saying that since the functions $\chi_S$ form an orthonormal basis, the inner product of 2 vectors equals the sum of the products of the corresponding coefficients.

In the special case when $f, g$ are the same function we get the Parseval Identity:

**Theorem 2.1.5.** *(Parseval Identity) For any* $f : \{-1,1\}^n \to \mathbb{R}$

$$\langle f, f\rangle = E_{x\sim\{-1,1\}^n}[f(x)^2] = \sum_{S\subseteq[n]} \widehat{f}(S)^2. \tag{2.5}$$

*In particular, if* $f : \{-1,1\}^n \to \{-1,1\}$ *is Boolean-valued then*

$$\sum_{S\subseteq[n]} \widehat{f}(S)^2 = 1 \tag{2.6}$$

The fact that the squared coefficients sum to 1, is important and it will be used later when defining the entropy.

We will now introduce two important learning models in computational learning theory. The Probably Approximate Correct (PAC) model [25] belongs to that class of learning models which is characterized by learning from examples.

**Definition 2.1.6.** *Let* $\mathcal{C}$ *be a class of Boolean functions* $f : \{-1,1\}^n \to \{-1,1\}$. *We say that* $\mathcal{C}$ *is PAC-learnable if there exists an algorithm* $\mathcal{L}$ *such that: for every* $f \in \mathcal{C}$, *any probability distribution* $\mathcal{D}$, *any* $\epsilon$ *such that* $0 \le \epsilon < 1/2$, *and any* $\delta$ *such that* $0 \le \delta < 1$, *the algorithm* $\mathcal{L}$ *will, with probability at least (1 -* $\delta$*), output a hypothesis* $h \in \mathcal{C}$ *such that* $error(h, f) \le \epsilon$, *in time that is polynomial in* $1/\epsilon$, $1/\delta$, $n$ *and size(c).*

In Agnostic learning model, no assumptions are made about the function that labels the examples, which means the learning model has no prior beliefs about the target function. The goal of agnostic learning algorithm for a concept class $\mathcal{C}$ is to produce a hypothesis $h \in \mathcal{C}$ whose error on the target concept is close to the best possible by a concept from $\mathcal{C}$.

**Definition 2.1.7.** *An algorithm $\mathcal{L}$ agnostically learns a class of Boolean functions $\mathcal{C}$ if for every $\epsilon > 0$, Boolean function $f$, and distribution $\mathcal{D}$ over $\{-1, 1\}^n$, $\mathcal{L}$ outputs with probability at least 1/2, a hypothesis $h \in \mathcal{C}$ such that $Pr[f \neq h] < \inf_{h \in \mathcal{C}} Pr[f \neq h] + \epsilon$.*

If a Boolean function is learnable in the Agnostic model, then it is also learnable in the PAC model.

## 2.2 DNF Formulas

One of the simplest ways to represent Boolean functions is by a disjunctive normal form (DNF) expression, which is a logical formula consisting of a disjunction of terms. For example, the expression $(x_1 \wedge x_2 \wedge x_3) \vee (x_2 \wedge x_4)$ is a DNF formula. Before that we formally define the DNF expressions, we need to define the AND and OR functions.

**Definition 2.2.1.** *The logical AND ( $\wedge$ ) indicates a conjunction between two statements. A conjunction is true if and only if both of the components are true.*
*The logical OR ( $\vee$ ) indicates a disjunction between two statements. A disjunction is false if and only if both of the components are false.*

The formal definition of DNF formula is presented below [4]:

**Definition 2.2.2.** *A Disjunctive Normal Form (DNF) formula over Boolean variables $x_1, \cdots, x_n$ is defined as the logical OR of terms $T_1 \vee \cdots \vee T_t$, where each of the terms $T_i$ is a logical AND of n literals. A literal is either a variable $x_i$ or its logical negation $\overline{x}_i$.*

We define the size of a DNF formula to be its number of terms and the width the maximal number of literals in any term. A function is monotone if it satisfies the monotonicity condition: for any $x, y \in \{-1,1\}^n$ if $x \leq y$ coordinate wise, then $f(x) \leq f(y)$. In other words: replacing -1 with 1 in $x$ will only increase the value of $f(x)$. In this thesis we focus on monotone disjunctive normal form formulas, which are DNF formulas with no negated literals in them. Monotone DNF formulas are one of the most widely used representations of monotone functions. A read-once DNF formula is a DNF formula in which the number of occurrences of each variable is at most one. We mostly focus on the read-once DNF formula.

A special case of DNF formulas are *Tribes*, which are read-once and monotone DNF. If we partition the $n$ variables into $n/w$ disjoint blocks, we get a tribe of width $w$. $Tribes_n$ is the OR of the $n/w$ AND's of the $w$ variables inside each block.

**Definition 2.2.3.** *The tribes function* $Tribes_{w,s} : \{-1,1\}^{sw} \to \{-1,1\}$ *is defined as*

$$Tribes_{w,s}(x_1, \cdots, x_{sw}) = (x_1 \wedge \cdots \wedge x_w) \vee \cdots \vee (x_{(s-1)w+1} \wedge \cdots x_{sw}) \tag{2.7}$$

*where* $x_i \in \{-1,1\}^w$

It is easy to see that this function will get the True value only if at least one of the blocks is unanimously True. Also, the following statement holds:

$$Pr_x[Tribes_{w,s}(x) = -1] = 1 - (1 - 2^{-w})^s \tag{2.8}$$

## 2.3 Entropy

Entropy, which is a measure of information, choice and uncertainty, was first introduced by Shannon in 1948 [22]. In his fundamental paper "A Mathematical Theory of Communication" he built the foundations for contemporary information and communication theory by developing a mathematical model for communication systems and a set of theoretical tools for analysing these systems. Given a discrete random variable $X$ with probability distribution $P_X$ the entropy is defined as:

$$H(X) = -\sum P_X(x) \log P_X(x)$$

In other words, it is the minimum number of bits required on average to describe the value $x$ of the random variable $X$.

Despite the fact that there are several ways to introduce the notion of entropy, we will define it in terms of its Fourier coefficients. This is possible, since from the Parseval Identity (2.1.5) we know that the squared coefficients sum to one, therefore we can treat them as a probability distribution on the subsets of $[n]$. Hence, for a Boolean function, the spectral entropy (called differently, the Fourier Entropy) is defined as:

$$\mathbb{H}(f) = \sum_{S \subseteq [n]} \widehat{f}^2(S) \log \frac{1}{\widehat{f}^2(S)} \tag{2.9}$$

It should be noted that the value of entropy ranges from 0 to the number of literals $n$ and it is larger when the spectrum of $f$ is in someway smeared out. The entropy is zero when the function $f$ is just a monomial and it is maximized if all of the Fourier coefficients have the same magnitude. There are cases when the entropy will be close to $n$ with high probability, for example in the case of randomly selected functions.

We now introduce some measures of complexity, like Sensitivity and Influence.

**Definition 2.3.1.** *Let $f$ be a Boolean function over $n$ variables.*

- *The sensitivity of a function $f : \{-1,1\}^n \to \{-1,1\}$ on input $x$ is the number of locations $i$ for which $f(x) \neq f(x^{\oplus i})$ ($s_x(f) = |\{i | f(x) \neq f(x^{\oplus i})\}|$).*
- *The average sensitivity of $f$ is the expected sensitivity of $f$ at a random assignment: $\overline{s}(f) = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} s_x(f) = E[s_x(f)]$.*

*Notation: $x^{\oplus i}$ denotes the input with the $i^{th}$ coordinate flipped.*

**Definition 2.3.2.** *For a Boolean function, the influence of the $i^{th}$ variable is the probability that flipping the $i^{th}$ coordinate changes the value of $f$.*

$$Inf_i = Pr[f(x) \neq f(x^{\oplus i})] \tag{2.10}$$

*The total influence of the function is the sum of the individual influences:*

$$I_f = \sum_{i=1}^{n} Inf_i$$

*which in the case of a monotone function will be $I_f = \sum_{i=1}^{n} \widehat{f}(i)$. Also, the average influence is defined as:*

$$\mathcal{E}[f] = avg_{i \in [n]}\{Inf_i[f]\}$$

*Which is the expected number of inputs bits that, when flipped, change the value of the function.*

By linearity of expectation, average sensitivity equals the total influence: $\overline{s}(f) = \sum_{i=1}^{n} Inf_i(f)$. Also, the average influence equals the expected fraction of sensitive bits in a randomly-selected input. Notice that for the AND function, every variable has a small influence of $2^{1-n}$.

# Chapter 3

# Thesis Work

## 3.1 Conjectures

The central research question of this study is related to the entropy of the functions computed by monotone DNF formulas. Over time, many studies have been done to calculate the entropy. We will be considering the Spectral Entropy of functions computed by $t-$ term DNF formulas. By fixing the number of terms, $(t)$, we consider each case, $(t = 1, 2, 3, 4)$, separately for the following questions: What is the maximum entropy for a function computed by a monotone DNF with a fixed number of terms? What is the form of the monotone DNF which seems to have the largest entropy? Does this occur over a function computed by a read-once DNF formula? How many literals does the DNF with the largest entropy contain? Before giving the detailed explanation of the experiments done in order to answer the above questions, we first state the main theorems and conjectures with which our research is closely related.

In 1998, Boppana was the first to prove that k-term DNF's have low average sensitivity [2] and that their influence is bounded by $\log(k)$.

**Theorem 3.1.1.** *Let $f : \{-1, 1\}^n \to \{-1, 1\}$ be computed by a k-term DNF formula. Then*

$I_f \leq O(\log k)$

Two of the most important open problems in the field of Fourier analysis of Boolean functions are the Fourier Entropy Influence Conjecture and Mansour's Conjecture, which are closely related to each other.

The initial motivation for Mansour's Conjecture was learning Boolean functions in polynomial time. Mansour conjectured that most of the Fourier coefficients are concentrated on only a polynomial number of coefficients.

**Conjecture 3.1.2.** (Mansour's Conjecture) *Let $f : \{-1,1\}^n \to \{-1,1\}$ be any function computable by a t-term DNF formula. Then there exists a polynomial $p : \{-1,1\}^n \to \mathbb{R}$ with $t^{O(\log 1/\epsilon)}$ terms such that $E[(f-p)^2] \leq \epsilon$.*

The Fourier Entropy Influence Conjecture was first posed in the setting of the random graphs and its motivation was to understand the influences under symmetry [6]. The conjecture is saying that the entropy is at most a constant times the influence of the function.

**Conjecture 3.1.3** (FEI Conjecture)**.** *For every Boolean function $f : \{-1,1\}^n \to \{-1,1\}$: $H(f) \leq C \cdot I(f)$*

$$\sum_{S \subseteq [n]} \widehat{f}(S)^2 \log_2 \frac{1}{\widehat{f}(S)^2} \leq C \cdot \sum_{S \subseteq [n]} \widehat{f}(S)^2 |S| \tag{3.1}$$

*for some universal constant $C > 0$.*

As a consequence of Theorem 3.1.1, Mansour's Conjecture will follow from the Fourier Entropy Influence Conjecture. This is considered one of the most important consequences of the FEI Conjecture, if it would be proven to be true.

Another interesting property is noticeable if we define the inequality 3.1 in terms of the function $f$ and not the Fourier Coefficients (think of the equation "without the hat"). In other words, consider the entropy of the squared values of the function by itself, in place of the squared values of the Fourier Coefficients of the function. In this case, it is true that $H(f) \leq 2 \cdot I(f)$, which is in fact the Logarithmic-Sobolev Inequality on $\{-1, 1\}^n$ [9].

The Mansour's Conjecture was recently proved to be true for read-once DNF [14]. We will formally state this theorem and give an outline of the proof.

**Theorem 3.1.4.** *Let $f$ be any read-once DNF formula with $t$ terms. Then there is a polynomial $p_{f,d}$ with $\|p_{f,d}\| = t^{O(\log 1/\epsilon)}$ and $E[(f - p_{f,d})^2] \leq \epsilon$ for all $\epsilon > 0$.*

The norm used in the above Theorem is the Fourier $l_1-$ norm (called differently the spectral norm) of $f$ and is defined to be $\| f \|_1 := \sum_S |\widehat{f}(S)|$.

The method used to prove Theorem 3.1.4 is polynomial interpolation. In other words, it is by constructing a polynomial $p$ in order to approximate a $t-$ term DNF.

Let $f = T_1 \vee \cdots \vee T_t$ be a DNF formula. Let $T_i(x) = 1$ if $x$ satisfies the term $T_i$, and 0 otherwise. Define $y_f : \{0, 1\}^n \to \{0, \cdots, t\}$ be a function that outputs the number of terms of $f$ satisfied by $x$, and let $P_d$ be the univariate polynomial which interpolates the values of $f$ on inputs $\{x : y_f(x) \leq d\}$.

It is shown than the $l_1$-norm of $p_{f,d}$ is polynomial in $t$ and exponential in $d$ and it will be a good approximation for any DNF formula $f$. Also, in the case of a read-once DNF formula, the probability that a term is satisfied is independent of the fact that any of the other terms are satisfied.

Our research is focused in the following conjecture, which to the best of our knowledge has not been studied before.

**Conjecture 3.1.5** (Wimmer, unpublished)**.** *Let $f$ be a function computed by a $t$-term DNF formula. For each fixed number of terms $t$, the maximum entropy of $f$ is achieved by a function computed by a read-once DNF formula.*

Intuitively, the conjecture asserts that for a fixed number of terms, the function which has the largest entropy can be expressed as a read-once DNF. This is because in a read-once DNF the literals appear only once, while in a non read-once DNF they appear more than once. Therefore, by the way spectral entropy is defined, we would expect some of the coefficients of a non read-once monotone DNF of the same literal to cancel out with each other. That will result in a decrease of the value of the entropy. We believe that for a fixed number of terms the maximum entropy should occur with a small number of literals, and eventually the maximum entropy decreases as the number of literals $n$ increases. Also, we think that the entropy is unbounded when the number of terms increases.

The consequences would be really interesting, if the above conjecture would be proven to be true. Since the Fourier Entropy-Influence Conjecture is true for read-once DNF [5], the entropy of a t-term DNF is $O(\log t)$ if the maximizer is a read-once DNF. Hence, the entropy of a t-term DNF being at most $O(\log t)$ implies Mansour's Conjecture. Therefore, the above conjecture would imply Mansour's Conjecture.

In the following section we explain the experiments that we run with the purpose of checking the value of the entropy. Throughout the thesis we will consider monotone DNF formulas and check the Conjecture 3.1.5 for the monotone case.

## 3.2 Experimental Results

To study the value of the entropy for a fixed number of terms, we wrote a program to generate a considerable number of monotone DNFs and calculated their spectral entropy. We considered the DNF by fixing the number of terms (t= 1, 2, 3, 4) and the number of literals ( $n$ ) and worked on each case separately.

### 3.2.1 1-term DNF and 2-term DNF formulas

First, we expressed the 1-term DNF, which in this case is an AND function, as a multilinear polynomial. This was done by using the symbolic library Sympy in Python [23]. Then, the coefficients were extracted and used to compute the spectral entropy. Let's see through an example how we compute the entropy. Let $x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5$ be a DNF formula. Using equation 2.3 we find its multilinear polynomial representation to be: $1/16 \cdot x_1 \cdot x_2 \cdot x_3 \cdot x_4 \cdot x_5 - 1/16 \cdot x_1 \cdot x_2 \cdot x_3 \cdot x_4 - 1/16 \cdot x_1 \cdot x_2 \cdot x_3 \cdot x_5 + 1/16 \cdot x_1 \cdot x_2 \cdot x_3 - 1/16 \cdot x_1 \cdot x_2 \cdot x_4 \cdot x_5 + 1/16 \cdot x_1 \cdot x_2 \cdot x_4 + 1/16 \cdot x_1 \cdot x_2 \cdot x_5 - 1/16 \cdot x_1 \cdot x_2 - 1/16 \cdot x_1 \cdot x_3 \cdot x_4 \cdot x_5 + 1/16 \cdot x_1 \cdot x_3 \cdot x_4 + 1/16 \cdot x_1 \cdot x_3 \cdot x_5 - 1/16 \cdot x_1 \cdot x_3 + 1/16 \cdot x_1 \cdot x_4 \cdot x_5 - 1/16 \cdot x_1 \cdot x_4 - 1/16 \cdot x_1 \cdot x_5 + 1/16 \cdot x_1 - 1/16 \cdot x_2 \cdot x_3 \cdot x_4 \cdot x_5 + 1/16 \cdot x_2 \cdot x_3 \cdot x_4 + 1/16 \cdot x_2 \cdot x_3 \cdot x_5 - 1/16 \cdot x_2 \cdot x_3 + 1/16 \cdot x_2 \cdot x_4 \cdot x_5 - 1/16 \cdot x_2 \cdot x_4 - 1/16 \cdot x_2 \cdot x_5 + 1/16 \cdot x_2 + 1/16 \cdot x_3 \cdot x_4 \cdot x_5 - 1/16 \cdot x_3 \cdot x_4 - 1/16 \cdot x_3 \cdot x_5 + 1/16 \cdot x_3 - 1/16 \cdot x_4 \cdot x_5 + 1/16 \cdot x_4 + 1/16 \cdot x_5 + 15/16$. Then, we extract the coefficients:

$$[1/16, -1/16, -1/16, 1/16, -1/16, 1/16, 1/16, -1/16, -1/16, 1/16, 1/16,$$
$$-1/16, 1/16, -1/16, -1/16, 1/16, -1/16, 1/16, 1/16, -1/16, 1/16, -1/16,$$
$$-1/16, 1/16, 1/16, -1/16, -1/16, 1/16, -1/16, 1/16, 1/16, 15/16]$$

and compute the spectral entropy, (2.9). In this case the value of entropy is: $H = 1.1324$. We ran experiments for 2 till 15 literals for the 1-term DNF and the results are shown in table 3.1. We noticed that after 3 literals, as the number of literals of the monotone DNF increases, the entropy decreases. Based on the results, we conjectured that the entropy for the 1-term DNF is maximized by the DNF with 3 literals $x_1 \wedge x_2 \wedge x_3$.

| **Number of literals** | **1-term DNF** | **Entropy** |
|---|---|---|
| 2 | $x_1 \wedge x_2$ | 2 |
| **3** | $x_1 \wedge x_2 \wedge x_3$ | **2.2169** |
| 4 | $x_1 \wedge x_2 \wedge x_3 \wedge x_4$ | 1.7012 |
| 5 | $x_1 \wedge \cdots \wedge x_5$ | 1.1324 |
| 6 | $x_1 \wedge \cdots \wedge x_6$ | 0.7012 |
| 7 | $x_1 \wedge \cdots \wedge x_7$ | 0.4161 |
| 8 | $x_1 \wedge \cdots \wedge x_8$ | 0.2402 |
| 9 | $x_1 \wedge \cdots \wedge x_9$ | 0.1360 |
| 10 | $x_1 \wedge \cdots \wedge x_{10}$ | 0.0759 |
| 11 | $x_1 \wedge \cdots \wedge x_{11}$ | 0.0419 |
| 12 | $x_1 \wedge \cdots \wedge x_{12}$ | 0.0229 |
| 13 | $x_1 \wedge \cdots \wedge x_{13}$ | 0.0124 |
| 14 | $x_1 \wedge \cdots \wedge x_{14}$ | 0.0066 |
| 15 | $x_1 \wedge \cdots \wedge x_{15}$ | 0.0035 |

Table 3.1: Entropy values for the 1 term DNF

A 2-term DNF is a disjunction of exactly two AND functions. So that we could check the value of entropy in this case we applied a brute force search throughout the search space of the 2-term DNF formula for $n$ fixed from 3 through 11. Then, following the same reasoning as the 1-term DNF, we computed the spectral entropy.

The number of DNF formulas checked for each number of literals is displayed in the table 3.2, along with the DNF which has the largest entropy for the specified number of literals. The results show that the 2-term monotone DNF with the largest entropy over the numbers of literals tested is: $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge x_6)$.

| n | \| Search Space \| | **2-term DNF** | **Maximum Entropy** |
|---|---|---|---|
| 3 | 3 | $(x_1 \wedge x_2) \vee (x_1 \wedge x_3)$ | 2.2169 |
| 4 | 5 | $(x_1 \wedge x_2) \vee (x_3 \wedge x_4)$ | 3.3253 |
| 5 | 8 | $(x_1 \wedge x_2 \wedge x_3) \vee (x_1 \wedge x_4 \wedge x_5)$ | 3.582 |
| **6** | 11 | $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge x_6)$ | **3.9314** |
| 7 | 15 | $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge x_6 \wedge x_7)$ | 3.5904 |
| 8 | 19 | $(x_1 \wedge x_2 \wedge x_3 \wedge x_4) \vee (x_5 \wedge x_6 \wedge x_7 \wedge x_8)$ | 3.2004 |
| 9 | 24 | $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge x_6 \wedge x_7 \wedge x_8 \wedge x_9)$ | 2.8013 |
| 10 | 29 | $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge x_6 \wedge x_7 \wedge x_8 \wedge x_9 \wedge x_{10})$ | 2.5663 |
| 11 | 35 | $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge x_6 \wedge x_7 \wedge x_8 \wedge x_9 \wedge x_{10} \wedge x_{11})$ | 2.4197 |

Table 3.2: 2-term DNF results

### 3.2.2 3-term DNF and 4-term DNF formulas

Since it would be hard to consider all the possible combinations of DNFs for the search space of the 3-term and 4-term DNF instead of an exhaustive search in these cases, we apply local search techniques to study the entropy. Local search methods are often used to tackle combinatorial search and optimization problems. They are based on the iterative exploration of a solution space; at each iteration, a local search algorithm steps from one solution to one of its "neighbors", or in other words, to solutions that are somehow close to the starting one. One of the most well-known local search techniques is Hill Climbing[21]. To describe the characteristics of it, we need to define these important concepts: the local move, the neighborhood space, the cost function, and the stopping criteria.

We define the neighbourhood $n(s)$ of a state $s$ to be all the states that are reachable in a single move from the state $s$, independent of the actual heuristic function used to choose which state to move to. A local move from state $s$ is a transition $s \rightarrow s'$, from $s$ to $s' \in n(s)$. When the algorithm makes a transition from one solution to another, it is said that the corresponding move has been accepted. The selection of moves is based on the values of the cost function. The cost function, which is associated to each state $s \in n(s)$, determines the "best" possible local move $s \to s'$ for the current state $s$ and it is used to drive the search toward good solutions for the optimization problem. Different stopping criteria can be used for Hill Climbing procedures. One is based on the total number of iterations: the search is stopped when a predetermined number of steps has been performed. Another stopping criteria can be the number of iterations without improving the cost function value of the best solution found so far. Also, the local search can terminate if the value of the cost function passes a certain threshold value.

A drawback of the Hill Climbing technique is that it is non-exhaustive in the sense that it is unable to guarantee detection of the global solution, but it searches non-systematically until the stopping criteria is met.

Considering our problem, we apply the Hill Climbing algorithm to iterate through the combinatorial space of solutions and give the local optimum for the 3-term and 4-term monotone DNF formula. We are optimizing the value of the entropy of a function computed by a DNF formula. In all the cases we start with a guess (more on this later) as the initial DNF while using a Hill Climbing algorithm, and select the local move that leads to the largest improvement of the current value of entropy. The Hill Climbing will terminate when no local move could improve any further the value of entropy. Upon termination, the search would have reached a local optimum of the entropy.

We need to define the neighbourhood space of the solution for the 3-term and 4-term DNF formulas. There are only finitely many types of variables in a DNF. By type of variable, we mean if the variable is contained in a specific term.

There are in total 7 distinct types of variables for the 3-term DNF:

- The variable is present in all the three terms.

- The variable is present in exactly two terms of the DNF formula. Here we distinguish

$\binom{3}{2} = 3$ subcases:

  - The variable is present in the first and second term.
  - The variable is present in the second and third term.
  - The variable is present in the first and third term.

- The variable is present in the exactly one term of the DNF. Here we distinguish $\binom{3}{1} = 3$ subcases:
  - The variable is present only in the first term.
  - The variable is present only in the second term.
  - The variable is present only in the third term.

For instance, the 3 term monotone DNF: $(x_1 \wedge x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_6) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_7)$ contains all the different types of variables. Following the same reasoning as for the 3-term case, for the 4 -term DNF formula there are 15 different types of variables:

- The variable is present in all the four terms.
- The variable is present in exactly three terms of the DNF formula. Here we distinguish $\binom{4}{3} = 4$ subcases.
- The variable is present in exactly two terms of the DNF formula. Here we distinguish $\binom{4}{2} = 6$ subcases.
- The variable is present in just one term of the DNF formula. We distinguish $\binom{4}{1} = 4$ subcases.

In our program we define the local move to be: flipping a single variable from one type to another. The neighborhood space will contain all the possible DNF that can be created by one local move from the initial DNF. We tried to choose as initial DNF, one that contains different types of variables. For each specific number of literals we chose a few different initial DNFs and we display all the details in the Appendix. Considering the amount of computer memory and time that the programs written for the 3 term and 4 term DNF require, we performed the experiments in 2 different servers with 12 GB and 64 GB respectively. Also, in order to save time the Parallel Python module [20] was used to parallelize the execution of the experiments.

The first 3 columns of tables 3.3 and 3.4 ( 3-term DNF and 4-term DNF case separately ) show the number of literals considered, the cardinality of the neighborhood space for each number of literals $n$ and the number of iterations that Hill Climbing performs until it finds the DNF with the largest value of entropy.

The DNF formulas that are outputted by the Hill Climbing ( have the largest entropy for a specific $n$ ) along with the value of spectral entropy are displayed at the last two columns of the respective tables.

| **n** | $\lvert n(s)\rvert$ | **Iterations** | **3-term DNF** | **Maximum Entropy** |
|---|---|---|---|---|
| 7 | 49 | 3 | $(x_1 \wedge x_3 \wedge x_5) \vee (x_2 \wedge x_4 \wedge x_6) \vee (x_3 \wedge x_4 \wedge x_7)$ | 4.7172 |
| 8 | 56 | 5 | $(x_3 \wedge x_4 \wedge x_6) \vee (x_1 \wedge x_5 \wedge x_7) \vee (x_2 \wedge x_5 \wedge x_8)$ | 4.9729 |
| **9** | 63 | 2 | $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge x_6) \vee (x_7 \wedge x_8 \wedge x_9)$ | **5.2551** |
| 10 | 70 | 5 | $(x_1 \wedge x_5 \wedge x_6) \vee (x_2 \wedge x_3 \wedge x_7 \wedge x_8) \vee (x_3 \wedge x_9 \wedge x_{10})$ | 5.0418 |
| 11 | 77 | 8 | $(x_2 \wedge x_6 \wedge x_7 \wedge x_8) \vee (x_1 \wedge x_5 \wedge x_{10} \wedge x_{11}) \vee (x_3 \wedge x_4 \wedge x_9)$ | 4.7986 |

Table 3.3: 3-term DNF results

| n | $\|n(s)\|$ | Iterations | 4-term DNF | Maximum Entropy |
|---|---|---|---|---|
| 7 | 105 | 5 | $(x_3 \wedge x_4 \wedge x_7) \vee (x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_6) \vee (x_1 \wedge x_4 \wedge x_7)$ | 5.0628 |
| 8 | 120 | 3 | $(x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_6) \vee (x_1 \wedge x_4 \wedge x_7) \vee (x_3 \wedge x_4 \wedge x_8)$ | 5.434 |
| 9 | 135 | 3 | $(x_2 \wedge x_3 \wedge x_6) \vee (x_1 \wedge x_5 \wedge x_7) \vee (x_4 \wedge x_5 \wedge x_8) \vee (x_3 \wedge x_4 \wedge x_9)$ | 5.6219 |
| 10 | 150 | 3 | $(x_3 \wedge x_5 \wedge x_6) \vee (x_2 \wedge x_7 \wedge x_8) \vee (x_3 \wedge x_4 \wedge x_9) \vee (x_1 \wedge x_4 \wedge x_{10})$ | 5.8467 |
| 11 | 165 | 3 | $(x_3 \wedge x_4 \wedge x_5) \vee (x_3 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_8 \wedge x_9) \vee (x_2 \wedge x_{10} \wedge x_{11})$ | 6.0564 |
| **12** | 180 | 4 | $(x_1 \wedge x_6 \wedge x_7) \vee (x_2 \wedge x_5 \wedge x_8) \vee (x_3 \wedge x_9 \wedge x_{10}) \vee (x_4 \wedge x_{11} \wedge x_{12})$ | **6.2878** |
| 13 | 195 | 5 | $(x_4 \wedge x_5 \wedge x_6 \wedge x_7) \vee (x_3 \wedge x_8 \wedge x_9) \vee (x_1 \wedge x_{10} \wedge x_{11}) \vee (x_2 \wedge x_{12} \wedge x_{13})$ | 6.1593 |

Table 3.4: 4-term DNF results

The results for the 3-term DNF show that the DNF with the largest entropy is $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge x_6) \vee (x_7 \wedge x_8 \wedge x_9)$ and for the 4-term DNF is $(x_1 \wedge x_6 \wedge x_7) \vee (x_2 \wedge x_5 \wedge x_8) \vee (x_3 \wedge x_9 \wedge x_{10}) \vee (x_4 \wedge x_{11} \wedge x_{12})$. Our results show that the value of entropy seems to decrease after a certain number of literals, hence this leads us to believe that the maximizer of the entropy for each term occurs on 3, 6, 9, 12 literals respectively for t=1, 2, 3, 4 terms. This also agrees with the Conjecture 3.1.5 for the monotone case, since all these DNFs are read-once. This increases our confidence in favor of the conjecture.

# Chapter 4

# Discussion

In this thesis, we present an analysis of the entropy of functions computed by a monotone DNF formula. We focused on the conjecture 3.1.5 and explored the question of whether the maximum entropy of a monotone DNF formula exists for a fixed number of terms. Even though the conjecture in general by itself seems hard to prove, from the experiments that we performed, it appears that the maximizer of the entropy exists for a fixed number of terms and it occurs on a small number of literals. Also, our results agree with the fact that the maximizer of the entropy is a read-once DNF.

The programs written with the intention of studying the entropy of t-term DNF formulas ($t = 1, 2, 3, 4$) can be further improved so the entropy of DNFs with higher terms can be checked. This way if a counterexample exists, the probability for detecting it would be higher. Also, another approach to the problem might be considering the entropy of Tribe functions or of functions that are close to read-once.

Since one of the limitations of this study is that the local search technique used does not guarantee finding the global optimum, we suggest that a more rigorous analysis should be done. The study should be extended to the non-monotone case of DNF formula.

# Bibliography


[1] Andris Ambainis, Andrew M. Childs, Ben W. Reichardt, Robert Špalek, Shengyu Zhang. Any AND-OR formula of size $N$ can be evaluated in time $N^{1/2+o(1)}$ on a quantum computer. *SIAM J. Comput.*, 39(6):2513-2530, 2010. Preliminary version in FOCS'07. [doi:10.1137/080712167] 654.

[2] Ravi B. Boppana. The average sensitivity of bounded-depth circuits. *Inf. Process. Lett.*, 63(5):257-261, 1997.

[3] Bireswar Das, Manjish Pal, and Vijay Visavaliya. The Entropy Influence Conjecture Revised,arXiv:1110.4301v2, 2011.

[4] Ryan O'Donnell. Analysis of boolean functions, 2012. http://analysisofbooleanfunctions.org/

[5] Ryan O'Donnell, John Wright, Yuan Zhou (2011). The Fourier Entropy-Influence Conjecture for certain classes of Boolean functions.

[6] Ehud Friedgut, and Gil Kalai. Every monotone graph property has a sharp threshold. Proceedings of the American Mathematical Society 124(10), 2993-3002, 1996.

[7] Parikshit Gopalan, Adam Kalai, and Adams R. Klivans. A query algorithm for agnostically learning DNF? In *21st Annual Conference on Learning Theory-COLT 2008, Helsinki, Finland, July 9-12, 2008*, pages 515-516. Omnipress, 2008.

[8] Parikshit Gopalan, Adam Kalai, and Adams R. Klivans. Agnostically learning decision trees. In *Proceesings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 527-536. ACM, 2008.

[9] Leonard Gross. Logarithmic Sobolev inequalities. *American Journal of Mathematics*, 97(4):1061-1083, 1975.

[10] Jeffrey Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55:414-440, 1997.

[11] Jeff Kahn, Gil Kalai, and Nathan Linial. The influence of variables on boolean functions. In *Proc. 29th IEEE Symp. on Foundations of Comp. Sci.*, pages 68-80, 1988.

[12] Adam Tauman Kalai, Adam R. Klivans, Yishay Mansour, Rocco A. Servedio. Agnostically learning halfspaces. *SIAM J. Comput.*, 37(6):1777-1805, 2008. Preliminary versoin in FOCS'05. [doi:10.1137/060649057] 654.

[13] Gil Kalai. The entropy/influence conjecture. Posted on Terence Tao's *What's new* blog, http://terrytao.wordpress.com/2007/08/16/gil-kalai-the-entropyinfluence-conjecture/, 2007.

[14] Adam Klivans, Homin Lee, and Andrew Wan. Mansours conjecture is true for random DNF formulas. In *Electronic Collocium on Computational Complexity TR10-023*, 2010.

[15] Adam R. Klivans, Rocco A. Servedio. Learning DNF in time $2^{\widetilde{O}(n^{1/3})}$. *J. Comput. System Sci.*, 68(2):303-318, 2004. Preliminary version in STOC'01. [doi:10.1016/j.jcss.2003.07.007] 654.

[16] Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform and learnability. *Journal of the ACM,* 40(3):607-620, 1993.

[17] Yishay Mansour. *Learning Boolean functions via the Fourier transform*, pages 391-424. Kluwer Academic Publishers, 1994.

[18] Yishay Mansour. An $O(n^{\log\log n})$ learning algorithm for DNF under the uniform distribution. *Journal of Computer and System Sciences*, 50:543-550, 1995. Prelim. ver. in *Proc. of COLT'92*.

[19] Marvin L. Minsky, Seymour A. Papert. Perceptions: An introduction to Computational Geometry. MIT Press, Cambridge, Mass., 1969. 653, 654.

[20] Parallel Python module. http://www.parallelpython.com/

[21] Stuart J. Russell, Peter Norvig. Artificial Intelligence, A Modern Approach, Prentice Hall, 2010.

[22] Claude E. Shannon. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27:379-423,623-656, 1948.

[23] Sympy library. http://sympy.org/en/index.html.

[24] Jun Tarui, Tatsuie Tsukiji. Learning DNF by approximating inclusion-exclusion formulae. In *Proc.* $14^{th}$ *IEEE Conf. on Computational Complexity(CCC'99)*, pp. 215-221. IEEE Comp. Soc. Press, 1999. [doi:10.1109/CCC.1999.766279] 654.

[25] L. G. Valiant. A Theory of the Learnable. Communications of the ACM 27(11):1134-1142, 1984.

# Appendices

# Appendix A

# Results

Here we display the results of all the experiments performed for the 3-term and 4-term DNFs. For a specific number of literals we chose a few different initial DNFs and then applied the Hill Climbing algorithm on each of them to find the DNF with the maximum entropy. The tables show the initial DNF along with their entropy, the number of iterations that the Hill Climbing performed and the DNF outputted with the respective entropy.

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_6) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_7)$ | $(H_{Init} = 3.6488)$ | 2 | $(x_1 \wedge x_2 \wedge x_3 \wedge x_5) \vee (x_2 \wedge x_4 \wedge x_6) \vee (x_3 \wedge x_4 \wedge x_7)$ | $(H_{Max} = 4.4453)$ |
| $(x_1 \wedge x_2 \wedge x_3) \vee (x_1 \wedge x_4 \wedge x_5) \vee (x_1 \wedge x_6 \wedge x_7)$ | $(H_{Init} = 4.1923)$ | 2 | $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5) \vee (x_6 \wedge x_7)$ | $(H_{Max} = 4.5504)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_6) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_7)$ | $(H_{Init} = 3.6488)$ | 3 | $(x_1 \wedge x_3 \wedge x_5) \vee (x_2 \wedge x_4 \wedge x_6) \vee (x_3 \wedge x_4 \wedge x_7)$ | $(H_{Max} = 4.7172)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_6) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_4 \wedge x_5)$ | $(H_{Init} = 3.1783)$ | 3 | $(x_3 \wedge x_4 \wedge x_6) \vee (x_3 \wedge x_5 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_4)$ | $(H_{Max} = 4.7172)$ |

Table A.1: Three term DNF with 7 literals

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_6) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_7) \vee (x_1 \wedge x_4 \wedge x_8)$ | $(H_{Init} = 3.5445)$ | 4 | $(x_4 \wedge x_5 \wedge x_6) \vee (x_2 \wedge x_3 \wedge x_7) \vee (x_1 \wedge x_4 \wedge x_8)$ | $(H_{Max} = 4.9729)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_6) \vee (x_1 \wedge x_2 \wedge x_5 \wedge x_7) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_8)$ | $(H_{Init} = 3.1298)$ | 5 | $(x_3 \wedge x_4 \wedge x_6) \vee (x_1 \wedge x_5 \wedge x_7) \vee (x_2 \wedge x_5 \wedge x_8)$ | $(H_{Max} = 4.9729)$ |
| $(x_1 \wedge x_2 \wedge x_4 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_8)$ | $(H_{Init} = 2.8425)$ | 4 | $(x_1 \wedge x_4 \wedge x_5) \vee (x_3 \wedge x_6 \wedge x_7) \vee (x_2 \wedge x_4 \wedge x_8)$ | $(H_{Max} = 4.9729)$ |

Table A.2: Three term DNF with 8 literals

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3) \vee (x_1 \wedge x_4 \wedge x_5 \wedge x_6) \vee (x_1 \wedge x_7 \wedge x_8 \wedge x_9)$ | $(H_{Init} = 4.2792)$ | 2 | $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge x_6) \vee (x_7 \wedge x_8 \wedge x_9)$ | $(H_{Max} = 5.2551)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_8) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_5 \wedge x_9)$ | $(H_{Init} = 2.6229)$ | 6 | $(x_1 \wedge x_6 \wedge x_7) \vee (x_2 \wedge x_3 \wedge x_8) \vee (x_4 \wedge x_5 \wedge x_9)$ | $(H_{Max} = 5.2551)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_6 \wedge x_8) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_5 \wedge x_6 \wedge x_9)$ | $(H_{Init} = 2.0590)$ | 7 | $(x_4 \wedge x_5 \wedge x_7) \vee (x_2 \wedge x_3 \wedge x_8) \vee (x_1 \wedge x_6 \wedge x_9)$ | $(H_{Max} = 5.2551)$ |
| $(x_1 \wedge x_2 \wedge x_4 \wedge x_5) \vee (x_1 \wedge x_3 \wedge x_6 \wedge x_7) \vee (x_2 \wedge x_3 \wedge x_8 \wedge x_9)$ | $(H_{Init} = 4.2835)$ | 4 | $(x_2 \wedge x_4 \wedge x_5) \vee (x_1 \wedge x_6 \wedge x_7) \vee (x_2 \wedge x_3 \wedge x_8 \wedge x_9)$ | $(H_{Max} = 5.2551)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_3 \wedge x_8 \wedge x_9)$ | $(H_{Init} = 3.6759)$ | 4 | $(x_2 \wedge x_4 \wedge x_5) \vee (x_1 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_3 \wedge x_8 \wedge x_9)$ | $(H_{Max} = 5.2551)$ |

Table A.3: Three term DNF with 9 literals

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_4 \wedge x_5 \wedge x_6) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_7 \wedge x_8) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_9 \wedge x_{10})$ | $(H_{Init} = 2.9429)$ | 5 | $(x_1 \wedge x_5 \wedge x_6) \vee (x_2 \wedge x_3 \wedge x_7 \wedge x_8) \vee (x_4 \wedge x_9 \wedge x_{10})$ | $(H_{Max} = 5.0418)$ |
| $(x_1 \wedge x_3 \wedge x_4 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_6 \wedge x_7 \wedge x_8) \vee (x_1 \wedge x_2 \wedge x_9 \wedge x_{10})$ | $(H_{Init} = 3.7781)$ | 3 | $(x_2 \wedge x_3 \wedge x_4 \wedge x_5) \vee (x_6 \wedge x_7 \wedge x_8) \vee (x_1 \wedge x_9 \wedge x_{10})$ | $(H_{Max} = 5.0418)$ |

Table A.4: Three term DNF with 10 literals

| **Initial DNF** | Entropy of Inital DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_5 \wedge x_6 \wedge x_7 \wedge x_8) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_9 \wedge x_{10}) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_{11})$ | $(H_{Init} = 2.8142)$ | 7 | $(x_5 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_9 \wedge x_{10}) \vee (x_3 \wedge x_4 \wedge x_8 \wedge x_{11})$ | $(H_{Max} = 4.7986)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_5 \wedge x_8 \wedge x_9) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{10} \wedge x_{11})$ | $(H_{Init} = 2.2900)$ | 8 | $(x_2 \wedge x_6 \wedge x_7 \wedge x_8) \vee (x_3 \wedge x_4 \wedge x_9) \vee (x_1 \wedge x_5 \wedge x_{10} \wedge x_{11})$ | $(H_{Max} = 4.7986)$ |

Table A.5: Three term DNF with 11 literals

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_6) \vee (x_1 \wedge x_7)$ | $(H_{Init} = 3.1078)$ | 5 | $(x_3 \wedge x_4 \wedge x_7) \vee (x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_6) \vee (x_1 \wedge x_4 \wedge x_7)$ | $(H_{Max} = 5.0628)$ |
| $(x_1 \wedge x_3 \wedge x_4) \vee (x_1 \wedge x_2 \wedge x_5) \vee (\wedge x_2 \wedge x_6) \vee (x_1 \wedge x_7)$ | $(H_{Init} = 4.0528)$ | 2 | $(x_3 \wedge x_4) \vee (x_2 \wedge x_5) \vee (x_2 \wedge x_6) \vee (x_1 \wedge x_7)$ | $(H_{Max} = 4.9103)$ |

Table A.6: Four term DNF with 7 literals

| **Initial DNF** | Entropy of Inital DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_6) \vee (x_1 \wedge x_4 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_4)$ | $(H_{Init} = 4.0344)$ | 3 | $(x_2 \wedge x_3 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_6) \vee (x_1 \wedge x_4 \wedge x_7) \vee (x_3 \wedge x_4 \wedge x_8)$ | $(H_{Max} = 5.4340)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_6) \vee (x_1 \wedge x_2 \wedge x_7) \vee (x_1 \wedge x_3 \wedge x_5 \wedge x_8) \vee (x_1 \wedge x_4 \wedge x_5)$ | $(H_{Init} = 4.1423)$ | 3 | $(x_2 \wedge x_3 \wedge x_6) \vee (x_1 \wedge x_2 \wedge x_7) \vee (x_3 \wedge x_5 \wedge x_8) \vee (x_1 \wedge x_4 \wedge x_5)$ | $(H_{Max} = 5.4340)$ |

Table A.7: Four term DNF with 8 literals

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_5 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_5 \wedge x_6 \wedge x_8) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_9) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_6)$ | $(H_{Init} = 3.1344)$ | 5 | $(x_2 \wedge x_5 \wedge x_7) \vee (x_5 \wedge x_6 \wedge x_8) \vee (x_1 \wedge x_4 \wedge x_9) \vee (x_2 \wedge x_3 \wedge x_6)$ | $(H_{Max} = 5.6086)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_6) \vee (x_1 \wedge x_2 \wedge x_5 \wedge x_7) \vee (x_1 \wedge x_4 \wedge x_5 \wedge x_8) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_9)$ | $(H_{Init} = 4.4437)$ | 3 | $(x_2 \wedge x_3 \wedge x_6) \vee (x_1 \wedge x_5 \wedge x_7) \vee (x_4 \wedge x_5 \wedge x_8) \vee (x_3 \wedge x_4 \wedge x_9)$ | $(H_{Max} = 5.6219)$ |

Table A.8: Four term DNF with 9 literals

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_5 \wedge x_8) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_6 \wedge x_9) \vee (x_1 \wedge x_3 \wedge x_6 \wedge x_{10})$ | $(H_{Init} = 3.0690)$ | 5 | $(x_1 \wedge x_5 \wedge x_7) \vee (x_3 \wedge x_5 \wedge x_8) \vee (x_2 \wedge x_6 \wedge x_9) \vee (x_4 \wedge x_6 \wedge x_{10})$ | $(H_{Max} = 5.8165)$ |
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_5 \wedge x_6) \vee (x_1 \wedge x_2 \wedge x_7 \wedge x_8) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_9) \vee (x_1 \wedge x_4 \wedge x_{10})$ | $(H_{Init} = 4.3374)$ | 3 | $(x_3 \wedge x_5 \wedge x_6) \vee (x_2 \wedge x_7 \wedge x_8) \vee (x_3 \wedge x_4 \wedge x_9) \vee (x_1 \wedge x_4 \wedge x_{10})$ | $(H_{Max} = 5.8467)$ |

Table A.9: Four term DNF with 10 literals

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_8 \wedge x_9) \vee (x_1 \wedge x_2 \wedge x_{10} \wedge x_{11})$ | $(H_{Init} = 4.2975)$ | 3 | $(x_3 \wedge x_4 \wedge x_5) \vee (x_3 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_8 \wedge x_9) \vee (x_2 \wedge x_{10} \wedge x_{11})$ | $(H_{Max} = 6.0564)$ |

Table A.10: Four term DNF with 11 literals

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_6 \wedge x_7) \vee (x_2 \wedge x_5 \wedge x_8) \vee (x_3 \wedge x_9 \wedge x_{10}) \vee (x_4 \wedge x_{11} \wedge x_{12})$ | $(H_{Init} = 5.5306)$ | 4 | $(x_1 \wedge x_6 \wedge x_7) \vee (x_2 \wedge x_5 \wedge x_8) \vee (x_3 \wedge x_9 \wedge x_{10}) \vee (x_4 \wedge x_{11} \wedge x_{12})$ | $(H_{Max} = 6.2879)$ |

Table A.11: Four term DNF with 12 literals

| **Initial DNF** | Entropy of Initial DNF | **Iterations** | **DNF with Maximum Entropy** | Maximum Entropy |
|---|---|---|---|---|
| $(x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_6 \wedge x_7) \vee (x_1 \wedge x_2 \wedge x_4 \wedge x_8 \wedge x_9) \vee (x_1 \wedge x_{10} \wedge x_{11}) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_{12} \wedge x_{13})$ | $(H_{Init} = 3.8751)$ | 5 | $(x_4 \wedge x_5 \wedge x_6 \wedge x_7) \vee (x_3 \wedge x_8 \wedge x_9) \vee (x_1 \wedge x_{10} \wedge x_{11}) \vee (x_2 \wedge x_{12} \wedge x_{13})$ | $(H_{Max} = 6.1593)$ |

Table A.12: Four term DNF with 13 literals