Fourier Analysis of Boolean Functions

Allen Philip J

2nd May 2014

Contents

1 Introduction

1.1 What are Boolean Functions

Boolean (or switching) function is a function of the form $f : \{0,1\}^n \to \{0,1\}$, where $\{0,1\}^n$is a Boolean domain in n dimentions, Consequently $\{0,1\}$represent logic $True$ or $False$. The output of a Boolean function is a Boolean value which is either $True$ or $False$,

It is possible to represent to every boolean function using n variables $x_1, x_2 \ldots, x_n$and two such formulas may be considered logically equivalent if and only if they express the same Boolean functions for all inputs in $\{0,1\}^n$.

1.2 Application of Boolean Functions

Boolean functions find their application in a broad variety of fields ranging from circuit design to social choice theory. A few of them have been briefly discussed below:

- Boolean functions have diverse applications in the field of cryptography. For example, boolean functions are the building blocks of symmetric cryptographic systems which are fundamental tools in the design of all types of digital security systems. A more detailed illustration of this facet of the boolean functions can be found in [2].
- Boolean functions in coding theory might be used to depict the indicator functions for the set of messages in a binary error-correcting code. As illustrated in [5], we can find a more elaborate analysis of boolean functions in the field of cryptography and error correcting codes.
- In cooperative game theory, monotone Boolean functions are called **simple games** (voting games); this notion is applied to solve problems in social choice theory as depicted in [7].
- Any digital circuit born out of a combination of a number of AND, OR, and NOT gates can be represented as a boolean function in n variables with a boolean output. An introduction to the field of cooperative boolean game theory can be found in [4].

Due to such deeprooted dependance of Boolean functions across so many diverse streams and fields, it becomes imperative to have a strong understanding of these boolean functions and find means to analyse them effectively.

1.3 Overview

We will start off with the fourier transformation and its application towards boolean functions and state and prove some important results pertaining to the same and try to gain an intuitive understanding as well. People familiar with the basics of Fourier analysis can move to sec. 2.2 where we will interpret the fourier expansion of boolean functions as a combination of parity functions.

Then we learn about one of the most important parameters of boolean functions, influences whose applications are broad and discussed in detail in sec. 3.1.1

Then we look at alternative means of representing Boolean functions which aim at interpreting and understanding the boolean functions more efficiently from an intuitive and mathematical perspective, with the two common special classes being Boolean functions as Decision Trees in sec. 4.2 and DNFs (Disjunctive Normal Forms) in sec. 4.3.

The next section deals with Spectral Learing techniques aimed at learning a boolean functions using just some of its significant fourier coeffecients within a certain error limit and extending the same to Decision Trees and DNFs over the course of which we'll be discussing about some extremely powerful and useful theorems, that is, the Goldreich Levin Theorem and Hastad's Switching Lemma which have revolutionzied the field of Boolean complexity theory. Sections until which serve as a survey and the references from which the information found in this survey are drawn are duly cited under the Bibliography.

Finally we discuss about Read-Once functions which are again another unique class of boolean functions which have not yet been explored entirely. We try to get some intuitive understanding on going about to bring about some technique to reduce the depth of ROFs to contant depth circuits and try to prove some results stating the same or proving otherwise.

2 Fourier Analysis of Boolean Functions

2.1 Fourier Transformation

We are all well aware of the textbook definition of Fourier transformation which is often described as taking a function in the "time-domain" and expressing it in the "frequency-domain". To make sense of this notion we will use vector spaces to motivate that the Fourier Transform infact represents some sort of a "change of basis" for a function and a more detailed explanation can be found for the same in [1].

2.1.1 Vectors and Bases

Consider a vector in $\mathbb{R}^2$and call it v. Let e_1, e_2represent the two standard basis vectors of $\mathbb{R}^2$.To write v in terms of them we need a way to somehow say "how much" of v is in the direction of e_1, e_2. Let's say we have a 'projection' operation which takes v and squishes it onto $e_1$and let's call it $proj_{e_1}v$ and similarly $proj_{e_2}v$. The magnitude of these projection quantities can be viewed as to what extent the vector v is close to the respective basis vector.

So we have a nice coordinate system representing our vector by projecting it onto the basis vectors. But suppose this particular set of basis vectors are not convenient for our problem and we'd like to look at it from a different persepective. We can always chose a new one and represent the vector by projecting them onto the new set of basis vectors as illustrated before.

2.1.2 The Fourier Transform

With the above mentioned premise we can proceed to the actual fourier transform and try to make some sense of it intuitively. A function f with it's coordinate representation can be thought of as the vector above with each x^{th}coordinate representing its distance from the origin. We can express it in terms of a basis using a list of n numbers, one for each x.

Now, the goal of the Fourier Transform is to decompose f into oscillations; which we can think of it as analogous to our previous goal of decomposing v into its components along certain basis vectors. If we can accomplish this for f, then we can write f as a sum of frequencies.[1]

2.2 Boolean Fourier Expansion

The Fourier Expansion of a Boolean function $f : \{-1,1\}^n \to \{-1,1\}^2$ is essentially the representation of the same as a real, multilinear polynomial meaning no variable $x_i$appears squared, cubed etc. Let us try to understand this with help of an example which can be found in [17].

Example. A min_2 function essentially evaluates the minumum of the two boolean values and is defined as follows:

$$min_2(+1,+1) = +1$$

$$min_2(-1,+1) = -1$$

$$min_2(+1,-1) = -1$$

$$min_2(-1,-1) = -1$$

Then min_2 can be expressed as a multilinear polynomial,

$$min_2(x_1,x_2) = -\frac{1}{2} + \frac{1}{2}x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_1x_2$$

which is the Fourier expansion of min_2.

2.2.1 Fourier Expansion for any Boolean Function

Theorem. *For any Boolean function we can say that its Fourier expansion is,*

$$f(x) = \sum_{S\subseteq[n]} \hat{f}(S)\chi_S(x)$$

Proof. There is an intuitive way in order to represent the Fourier expansion for any boolean function as a multilinear polynomial representation as illustrated in [17]. In order to achieve this, we define an **indicator polynomial** which for each point $a \in \{-1,1\}^n$

$$I_{\{a\}}(x) = \left(\frac{1+a_1x_1}{2}\right)\left(\frac{1+a_2x_2}{2}\right)\ldots\left(\frac{1+a_nx_n}{2}\right)$$

[1]For more details kindly refer to [1]

[2]Note: Here {-1,1} are analogous to the previous representation of {0,1}

which takes value 1 when $x = a$ and value 0 when $x \in \{-1, 1\}^n \setminus \{a\}$. Thus f has the polynomial representation

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

Multiplying through the indicator polynomial we end with the fourier expansion as

$$f(x) = \sum_{S \subseteq [n]} \hat{f}(S) x^S$$

where $\hat{f}(S)$ is the fourier coeffecient of f on S, and x^S is a multilinear polynomial in n, that is[3]

$$x^S = \prod_{i \in S} x_i = \chi_S(x)$$

Hence for any Boolean function we can say that its Fourier expansion is,

$$f(x) = \sum_{S \subseteq [n]} \hat{f}(S) \chi_S(x)$$

□

2.3 Parity Function as Orthonormal Basis

We had already discussed in sec. 2.1 that Fourier transformation can infact be thought of as "change of basis" of sorts. Over the course of this section we would like to show that the basis is infact that of parity functions and is an orthonormal basis using excerpts fom [15, 8].

In the case of the fourier expansion of boolean functions, observe the function characterised by $\chi_S(x)$. Take note that it is in fact a logical parity function or an exclusive-or (XOR) of the bits $(x_i)_{i \in S}$. And in case we stack out all the possible subsets of $[n]$, we can conclude that we can represent the function f as a linear combination of parity functions (over the reals).

$$f = \sum_{S \subseteq [n]} \hat{f}(S) \chi_S$$

More generally, we deduce that in fact that they are a linearly independent basis for the vector space formed by the boolean function which in turn shows the uniqueness of the fourier expansion.

[3]Note: $x^\emptyset = 1$ by convention

2.3.1 Inner Product

Definition 1. We define an inner product on a pair of functions $f, g : \{-1,1\}^n \to \mathbb{R}$ by

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

The inner product can be interpreted also as the point-wise distance between the functions f, g and hence if the two functions are equivalent the inner product would go to 1 while if the functions are orthogonal the inner product vanishes to 0. Therefore, the inner product can also be depicted using the notation $\|f\|_2 = \sqrt{< f, f >}$,and more generally,

$$\|f\|_p = \mathbf{E}[|f(x)|^p]^{\frac{1}{p}}$$

2.3.2 Orthonormality of the Basis

Going back to the basis of parity functions, the crucial fact underlying all analysis of boolean functions is that this is an orthonormal basis.

We know that for $x \in \{-1,1\}^n$ it holds that $\chi_S(x)\chi_T(x) = \chi_{S \triangle T}(x)$ where $S \triangle T$ represents the symmetric difference. The proof is as follows:

$$\chi_S(x)\chi_T(x) = \prod_{i \in S} x_i \prod_{i \in T} x_i = \prod_{i \in S \triangle T} x_i \prod_{i \in S \cap T} x_i^2 = \prod_{i \in S \triangle T} x_i = \chi_{S \triangle T}(x).$$

Also we know that if $S = \emptyset$ then $\mathbf{E}[\chi_S(x)] = \mathbf{E}[1] = 1$.[4] Otherwise,

$$\mathbf{E}\left[\prod_{\mathbf{i} \in \mathbf{S}} \mathbf{x_i}\right] = \prod_{\mathbf{i} \in \mathbf{S}} \mathbf{E}[\mathbf{x_i}] = 0$$

Using the above two results, the orthonormality of the basis functions follows suit.

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

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

2.3.3 Standard Fourier Results[5]

Theorem. ***Parseval's Theorem:*** *For any $f : \{-1,1\}^n \to \mathbb{R}$,*

$$< f, f >= \mathbf{E}_{x \sim \{-1,1\}^n}[f(x)^2] = \sum_{S \subseteq [n]} \hat{f}(S)^2$$

[4]Since the x_i are random and hence $(1/2)(+1) + (1/2)(-1) = 0$.

[5]For more fourier results and information pertaining to the results presented below, kindly refer to [14]

where $\mathbf{E}$ *is the expected value of* $f.f$ *using the uniform distribution on* $\{0,1\}^n$.

Proof. The parseval's identity can be proven as follows

$$E_x[f^2(x)] = E_x\left[\left(\sum_S \hat{f}(S)\chi_s(x)\right)\left(\sum_T \hat{f}(T)\chi_T(x)\right)\right]$$

$$= \sum_S \sum_T \hat{f}(S)\hat{f}(T)E_x\left[\chi_{S\cup T}(x)\right]$$

If $S \neq T$, then $E_x\left[\chi_{S\cup T}\right]$ is zero evident from the orthogonal property of boolean functions. Therefore the expression reduces to

$$< f, f >= \sum_{S\subseteq[n]} \hat{f}(S)^2$$

In particular if the function is boolean-valued then

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

□

Theorem. ***Plancheral's Theorem:*** *For any* $f, g : \{-1,1\}^n \to \mathbb{R}$,

$$< f, g >= \mathbf{E}_{x\sim\{-1,1\}^n}[f(x)g(x)] = \sum_{S\subseteq[n]} \hat{f}(S)\hat{g}(S)$$

Proof. A similar approach can be drawn as to the Parseval's identity except we have two different functions here and the sets should be split appropriately as follows:

$$< f, g >=< \hat{f_S}\chi_S, \hat{g_T}\chi_T >= \hat{f_S}\hat{g_T} < \chi_S, \chi_T >= \sum_{S\subseteq[n]} \hat{f}(S)\hat{g}(S)$$

□

Definition 2. Distance Function: For any $f, g : \{-1,1\}^n \to \{-1,1\}$, we define their (relative hamming) distance to be

$$dist(f,g) = \Pr_x[f(x) \neq g(x)],$$

the fraction of inputs on which they disagree.

3 Influences and Social Choice Theory

3.1 Boolean Functions and Social Choice Theory

As discussed earlier, we know that boolean functions play a dominant role in Social Choice Theory, which has been discussed in detail in [11], and one of the parameters of paramount interest to us is 'Influence'. Social Choice Theory is often widely discussed by economists, political scientists, mathematicians, and computer scientists addressing one fundamental question primarily which is how to best analyse or make sense of the aggregate opinions of many agents. This is quite significant especially in the scenario of citizens voting in an election, getting a consensus in a committee, or analysing scenarios requiring a collective decision in general.

Boolean functions in general have a very appealling and convenient interpretation with respect to social choice theory. For example, as boolean function $f : \{-1, 1\}^n \to \{-1, 1\}$can be thought of as a voting rule or a social choice function for an election with 2 candidates and n voters; and the votng rule could be as simple as a $Majority$ function to decide the winner of the election.

$$Maj_n : \{-1, 1\}^n \to \{-1, 1\}$$

here $Maj_n(x) = sgn(x_1 + x_2 + \ldots + x_n)$ for n odd.

3.1.1 Influences and Social Choice Theory

Having established the aforementioned interpretation of Boolean functions, we can begin addressing the burning questions of Social Choice Theory. So naturally we'd like to measure the $'power'$ or $'influence'$ of the i^{th} voter in order to see how stable/unstable the constituency is. In more mathematical terms, it can be defined as the probability that if the i^{th}voter were to differ the outcome of the election would change.

Definition 3. The $influence$ of coordinate i can be defined as:

$$\mathbf{Inf_i}[f] = Pr_{x\sim\{-1,1\}^n}\left[f(x) \neq f(x^{\oplus i})\right]$$

where $x^{\oplus i} = (x_1, x_2, \ldots x_{i-1}, -x_i, \ldots, x_n)$. For more information on influences of boolean functions refer [10].

3.1.2 Geometric Interpretation of Infuences

The Boolean function could also be represented as a *Hamming cube* [15]with is just the representation of a cube over n dimensions with the edges representing all the possible coordinates in $\{-1, 1\}^n$.

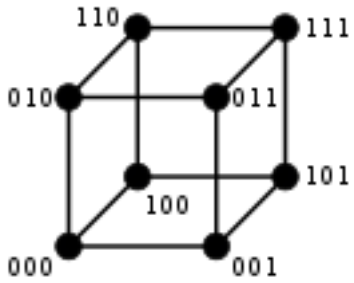

Figure 3.1: Hamming Cube in $\mathbb{R}^3$

Definition 4. The distance between any two points in the hamming cube which is called the hamming distance is defined as:

$$Dist(x, y) = \#\{i : x_i \neq y_i\}$$

Having established the Hamming Cube we can now try to understand the geometric interpretation of Influence. All the edges in the Hamming Cube which agree in all coordinates and differ only on the i^{th} coordinate are called $dimension - i$ edges and if any of those edges have $f(x) \neq f(y)$ then its called a boundary edge.

Take for example the Maj_3 function which is depicted in the figure below. The edges characterized by a grey colour correspond to function value of $+1$ and those with a white coloured edge have function value of -1. The boundary edges for the same have been highlighted in black. Intuitively, the influence can be defined as the fraction of $dimension - i$ edges in a Hamming Cube which are boundary edges.

3.2 Total Influence

Definition 5. Total Influence, as one might have guessed from the nomenclature, is the sum of the influences of a boolean function over all its n coordinates. In formal terms, the total influence of $f : \{-1, 1\}^n \to \{-1, 1\}$

$$\mathbf{I}[f] = \sum_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{Inf_i}[f]$$

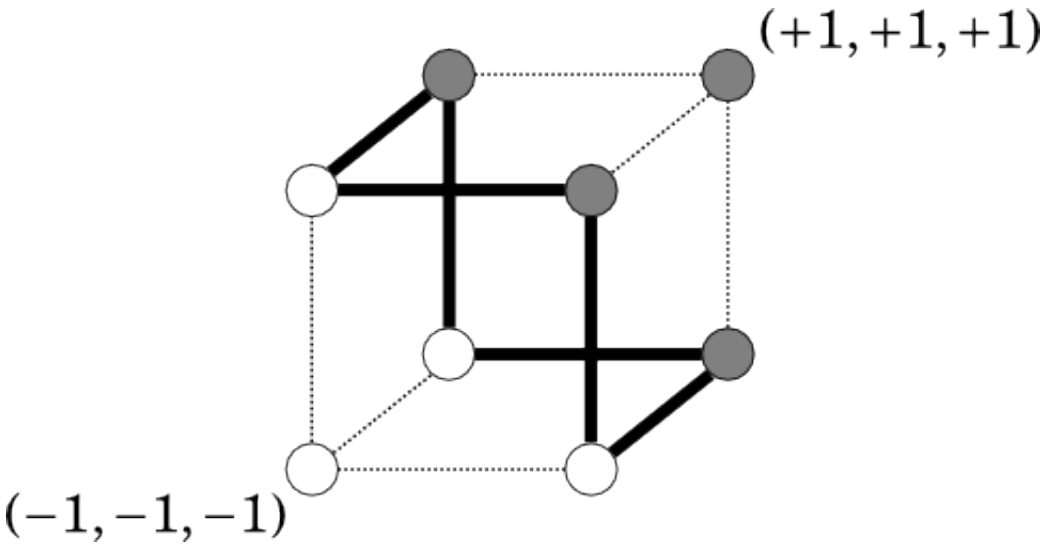

Figure 3.2: Maj_3 function

Total influence, from a social choice theory, can be intuitively understood as the succeptibility of the voting results (boolean function value) to fluctuate if any of the voters were to differ. From a coding theory stand point, if the boolean function representing a code transmitter have very high influence, if the reciever decodes the signal with an error its more likely to give a wrong function value.

4 Special Classes of Boolean Functions

4.1 Motivation behind Special Class of Boolean Functions

We know that a boolean function in general can be expressed as $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ which is a very generic and standard notation. It doesn't give us any other information about the boolean function other than the fact it takes values from $\{-1, 1\}^n$ and returns a real value. Morever, it is quite generic and exhaustively represents all the exisiting boolean functions. In general this can prove detrimental to further understand and analyse boolean functions in detail since some of the boolean functions in real-time applications have simpler forms and need not be analysed under this generic banner. Hence we need a more convenient representation wherein it increases our ease of function interpretation and analysis for these special class of boolean functions.

Any form of representation, though it may be limited to a specific class of boolean functions, if it can address efficiently some of the questions posed above or provide an eloquent platform to interpret or analyse the boolean functions, its worth consideratition. Another good representation would be if the model shows high correlation with prevelant pre-existing models which can be robustly used across a wide array of fields like graphs say.

Another guideline for building an a special representation for boolean functions is to incorporate the boolean logic in the function representation. The canonical representation is purely mathematical and has no concept boolean logic incorporated into it but we know that boolean functions can be expressed as a combination of logic gates. The options are virtually limitless and we explore different avenues thereby building a stronger understanding and grasp on boolean functions in general.

4.2 Decision Trees

A Decision Tree is a binary tree where each internal node is labeled with a variable, and each leaf is labeled with either 0 or 1. An assignment to the variables determines a unique path from the root to a leaf and at each internal node the children

correspond to the paths taken when the variable takes value 0 or 1. The value of the function at the assignment is the value at the leaf reached. A more mathematical representation of the boolean function would be:

$$f = \sum_{paths,P} f(P).\mathbf{I}_{C_p} \tag{4.1}$$

where $\mathbf{I}_{C_p}$ is indicator variable of all the variables in the path P of the decision tree. The *depth* of the decision tree is length of the longest path from the root to the

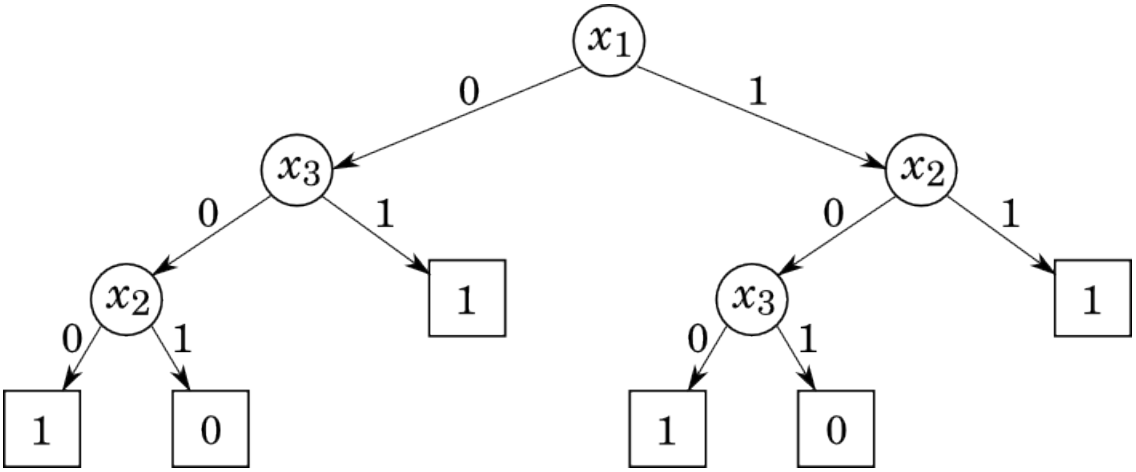

Figure 4.1: Decision Tree computing $Sort_3$

leaf and is denoted by $DT - depth(T)$. Similarly for a function f, we define depth $DT - depth(f)$ as the minumum depth of a decision tree that computes the function f. Note that every boolean function on n variables can be represented by a decision tree with at most 2^n nodes and depth at most n.

4.2.1 Important Results for DTs

We know that $deg(f)$, where f is a boolean function, is the number of variables in the largest term of its fourier expansion. Given that the decision tree T be of depth k, then

$$deg(f) \leq k \tag{4.2}$$

This follows from boolean expression as in eqn(4.1), where we select only variable in every level and since variables can't repeat the maximum degree of the function in such a representatio would be the $DT - depth(f)$.

4.3 DNFs and CNFs

In boolean logic, a disjunctive normal form (DNF) is a standardization of a logical formula which is a disjunction of conjunctive clauses; otherwise put, it is an OR

of ANDs also known as a Sum of products. Similarly, a formula is said to be in conjuctive normal form (CNF) if its in conjunction of clauses; that is its an AND of ORs. Besides being natural from a computational point of view, these representation classes are close to the limit of what complexity theorists can "understand" owing to the strong Fourier concentration properties they exhibit.

The size of the term in the number of variables in the term. The size of the DNF formula, however, is the number of terms it has and is denoted by $DNF_{size}(f)$. The width of the DNF formula is the size of the largest term in the formula and is dentoed by $DNF_{width}(f)$.

For example the DNF representation of the $Sort_3$ function, which take the value 1 if $x_1 \leq x_2 \leq x_3$ and 0 otherwise, would look like:

$$Sort_3(x_1, x_2, x_3) = (x_1 \wedge x_2) \vee (\bar{x_2} \wedge \bar{x_3}) \vee (\bar{x_1} \wedge x_3)$$

where $\wedge$ represents logical AND and $\vee$ represents logical OR. The size of the above DNF is 3 and its width is 2.

5 Spectral Learning

5.1 What do we mean by Spectral Learning

Our goal over the course of the upcoming sections would be to assess the "complexity" of a boolean functions it terms of our ability to "learn" the function. Intuitively, we can interpret learning a boolean function as an effective means of reconstructing the boolean, within a certain error limit, when given limited access to the actual boolean function. In a real time scenario, this could mean like a signal a transmitted using a boolean function generator and our goal at the transmitter side is to correctly estimate the source boolean function from the corrupt signal transmitted to the reciever side. The following section is built on the the intuition from [14] and a more detailed description can be found there.

This leads us to the question what factors are required to correctly estimate a boolean function. We learnt that a boolean function can be represented as $\sum_{S\subseteq[n]} \hat{f}(S)\chi_s$. Notice that if we can identify a collection $\mathcal{F}$ in $[n]$ that have very large fourier coeffecients we can effectively reconstruct the function as $\sum_{S\subseteq\mathcal{F}} \hat{f}(S)\chi_s$ wherein we ensure that the maximum amount of information in retained in this new function and the error arises due to the subsets of $[n] - \mathcal{F}$.

Let us take a example to cement our intuitive understanding of the same as illustrated in [14]. Let f be a boolean function such that for some unknown γ it holds that $\hat{f}(\gamma) = 0.99$ and no other information of the function f is available. The intuitive hypothesis would be $h(x) = \hat{f}(\gamma)\chi_\gamma = 0.99\chi_\gamma$. Now we'd like to estimate the squared error for the same. The error function would be $err_h(x) = |f(x) - h(x)|$. The expected square error would be,

$$E\left[(f-h)^2\right] = \left(\sum_{\beta\neq\gamma} \hat{f}(\beta)^2\right) + \left(f^2(\gamma) - f^2(\gamma)\right) = 1 - f^2(\gamma) = 0.0199$$

Knowing any more information of the fourier coefficients would further significantly reduce the expected square error. For example, if $f(\alpha) = 0.005$ then expected square error is,

$$E\left[(f-h)^2\right] = 1 - f^2(\alpha) - f^2(\gamma) = 0.019875$$

5.1.1 Learning Model

The learning model has a class of functions $\mathcal{F}$ which we wish to learn. Out of this class there is a specific function $f \in \mathcal{F}$ which is chosen as a target function. A learning algorithm has access to examples. An example is a pair $< x, f(x) >$ where x is an input and $f(x)$ is the value of the target function on the input x. After requesting a finite number of examples the learning algorithm outputs a hypothesis h. The error of a hypothesis h with respect to the function f, is defined to be $error(f,h) \triangleq Pr\left[f(x) \neq h(x)\right]$, where x is distributed uniformly over $\{0,1\}^n$.

We discuss two models for accessing the examples. In the uniform distribution model the algorithm has access to a random source of examples. Each time the algorithm requests an example a random input $x \in \{0,1\}^n$ is chosen uniformly and the example $< x, f(x) >$ is returned to the algorithm. In the membership queries model the algorithm can query the unknown function f on any input $x \in \{0,1\}^n$ and receive the example $< x, f(x) >$.

A randomized algorithm A learns a class of functions $\mathcal{F}$ if for every $f \in \mathcal{F}$ and the algorithm outputs an hypothesis h such that with probability at least $1-\delta$ the error is less than ϵ.The algorithm A learns in polynomial time if its running time is polynomial in n, $1/\epsilon$, and $\log 1/\delta$.

5.2 Small Fourier Spectrum

In the previous section we showed how to approcimate a single coeffecient. But in most cases the boolean function can only be approximated by taking into consideration a small number of fourier coeffecients. Broadly there are two ways in which they are commonly analysed which have the almost the same method of operation.

Definition 6. We say that the Fourier spectrum of $f : \{-1,1\}^n \to \mathbb{R}$ is $\epsilon-$concentrated on degree up to k if

$$\sum_{S \subseteq [n], |S| > k} \hat{f}(S)^2 \leq \epsilon$$

Similarly we say that the Fourier spectrum of $f : \{-1,1\}^n \to \mathbb{R}$ is $\epsilon-$concentrated on degree up to collection $\mathcal{F}$ if

$$\sum_{S \subseteq [n], |S| \notin k} \hat{f}(S)^2 \leq \epsilon$$

The fundamentals of the boolean learning theory are built on this sutle yet powerful premise. If we have estimated by some means that the fourier spectrum is concentrated in low degree or in a collection then the next question to be addressed is how do we construct our hypothesis, and what is the order of performance for that process.

5.3 Learning Theory

Before getting into the crux of Learning Theory which has been build upon [14], we must familiarize ourselves with the chernoff bound which will be used extensively through the subsequent sections.

5.3.1 Chernoff Bound

In probability theory, the Chernoff bound [6] gives exponentially decreasing bounds on tail distributions of sums of independent random variables. It is a sharper bound that the other well-known Markov's Inequality and Chebyshev Inequality.

Theorem. ***Chernoff Bound*** *[6]: Let $X_1, X_2, \ldots X_n$ be independent and identically distributed random variables such that, $X_i \in [-1, +1]$, $E[X_i] = p$ and $S_n = \sum_{i=1}^{n} X_i$. Then*

$$Pr\left[|\frac{S_n}{n} - p| \geq \lambda\right] \leq 2e^{-\lambda^2 n/2}$$

5.3.2 Approximating Single Coefficient

Consider the example taken in sec. 5.1 where we try to reconstruction the function from just one large fourier coefficient. But notice however that the hypothesis $h(x)$ is a function from $\{-1, 1\}^n$ to $\mathbb{R}$ and not to $\{-1, 1\}$. It makes sense thereby to construction another function $z : \{-1, 1\}^n \to \{-1, 1\}$ given by

$$z(x) = sgn(h(x))$$

where $sgn(x)$ is the sign function which takes the value -1 if $x < 0$ and $+1$ is $x > 0$.

We know that the fourier coefficient of a boolean function, in terms of the inner product, can b expressed as:

$$\hat{f}(S) =< f, \chi_S >= \mathbf{E}[f.\chi_s]$$

This expectation term can basically be estimated given random access to the function wherein we query the function and it returns a function value and the value of x corresponding to it, tha is, of the form $(x, f(x))$. Therefore the value from the i^{th} query could be considered as the value of the i^{th} random variable and over m iterations we can evaluate the Chernoff bound over some error limit of say λ. Here S_n would translate into the approximate value of fourier coefficient which is denoted by $\tilde{f}(\gamma)$. Doing so we would get,

$$Pr[|\hat{f}(\gamma) - \tilde{f}(\gamma)| \geq \lambda] \leq 2e^{-\lambda^2 m/2} \leq \delta$$

where we substitute the term on RHS as δ. Therefore we can say with high probability that $|\hat{f}(\gamma) - \tilde{f}(\gamma)| \leq \lambda$ and consequently $|\hat{f}(\gamma) - \tilde{f}(\gamma)|^2 \leq \lambda^2$ if appropriate values are taken for m, λ.

Re-evaluating the expected square error loss we get,

$$\mathbf{E}[(f-h)^2] = \mathbf{E}[(f - \tilde{f}(\gamma)\chi_\gamma)^2] = \sum_{\alpha \neq \gamma} \hat{f}(\alpha)^2 + (\hat{f}(\gamma) - \tilde{f}(\gamma))^2$$

$$\mathbf{E}[(f-h)^2] \leq 1 - f^2(\gamma) + \lambda^2$$

It should be noted that h is not the desired function but rather z. Essentially we should now show that

$$Pr[f(x) \neq sgn(h(x))] \leq \mathbf{E}[(f-h)^2]$$

This is can be quite easily proved through the following series of steps.

$$Pr[f(x) \neq sgn(h(x))] = 2^{-n} \sum_x \mathbf{I}[f(x) \neq sgn(h(x))]$$

where $\mathbf{I}$ is an indication function and equals $\#x : \{f(x) \neq sgn(h(x))\}$and

$$\mathbf{E}[(f-h)^2] = 2^{-n} \sum_x [f(x) - h(x)]^2$$

Taking two cases,

$$|f(x) - h(x)| > 1, f(x) \neq sgn(h(x))$$

$$|f(x) - h(x)| \geq 0, f(x) = sgn(h(x))$$

Therefore $Pr[f(x) \neq sgn(h(x))] \leq \mathbf{E}[(f-h)^2]$ and hence

$$\mathbf{E}[(f-z)^2] \leq 1 - f^2(\gamma) + \lambda^2$$

Here intuitively we can understand that λ^2 is the additional error term introduced due to estimating the fourier coefficent at γ since the error term originally had been $1 - f^2(\gamma)$.

5.3.3 Approximating Low Degree Boolean functions

As discussed earlier, most of the functions innnnnnn general can be approximated only by considering a small number of coefficients. In this sections, let's consider that the Fourier spectrum of $f : \{-1, 1\}^n \to \mathbb{R}$ is $\epsilon-$concentrated on degree up to k such that

$$\sum_{S \subseteq [n], |S| > k} \hat{f}(S)^2 \leq \epsilon$$

Sample m examples, $< x_i, f(x_i) >$. For each S, with $|S| \leq d$, compute $a_s = \frac{1}{m}\sum_{i=1}^{m} f(x_i)\chi_s(x_i)$. And the output function $z(x)$ is defined as

$$z(x) = \sum_{|S|\leq d} a_S \chi_s(s)$$

Learning theorem states that $\mathbf{E}[(f-z)^2] \leq \epsilon + \lambda$, which can be proved as follows.

First we claim that teh algorithm approximates each coefficient within λ with a high probability. That is,

$$Pr[|a_S - \tilde{f}(\gamma)| \geq \lambda] \leq 2e^{-\lambda^2 m/2} \leq \delta$$

The error in $z(x)$ is bounded by,

$$\mathbf{E}[(f-z)^2] \leq \epsilon + \sum_{|S|\leq d} (\hat{f}(S) - a_S)^2 \leq \epsilon + \sum_{|S|\leq d} \lambda^2 \leq \epsilon + n^d.\lambda^2$$

Let $n^d.\lambda^2 \leq \alpha \Rightarrow \lambda \leq \sqrt{\frac{\alpha}{n^d}}$. And hence proved that

$$\mathbf{E}[(f-z)^2] \leq \epsilon + \lambda$$

To find the lower bound on m, we know that the learning theorem holds with probability $1-\delta$, therefore,

$$2e^{-\lambda^2 m/2} \leq \delta$$

which holds true for

$$m \geq \frac{2n^d}{\alpha} ln\left(\frac{2n^d}{\delta}\right)$$

5.4 Goldreich Levin Theorem

Until now we discussed how to apply the learning theorem to obtain an approximate function within good error limits for a single coeffecient case and a low degree concentrated case. We could easily extend the same to the case where the fourier spectrum is restricted to a collection $\mathcal{F}$ which would be a far more powerful result and all the other results can be derived from it by making smart substitutions.

Now the question becomes how do we find this collection $\mathcal{F}$ wherein the bulk of the fourier spectrum is concentrated. To answer this question, the Goldreich Levin Theorem was formulated. Goldreich Levin theorem in some sense could be called the "opposite" of learning theory: cryptography. At the highest level, cryptography is concerned with constructing functions which are computationally easy to compute but computationally difficult to invert.

Before we venture into the Goldreich Levin statement and proof, which has been drawn from [16], we need to look at some pre-requisites which are crucial to understanding the same.

5.4.1 Restrictions

One of the most common operations on boolean functions $f : \{-1,1\}^n \to \mathbb{R}$ is restriction to subcubes. Suppose $[n]$ into two sets, J and $\bar{J} = [n] \setminus J$. If the input bits in $\bar{J}$ are fixed to constants, the result is a function $\{-1,1\}^J \to \mathbb{R}$.

Definition 7. Let $f : \{-1,1\}^n \to \mathbb{R}$ and let $(J, \bar{J})$ be a partion on $[n]$. Let $S \subseteq J$.Then,

$$\mathbf{F}_{S|\bar{J}} f(z) = \widehat{f_{J|z}}(S) \tag{5.1}$$

Notice that in the above definition the function on the LHS is a function of z while fixing S whereas the function on the right is a function of S while fixing z.

Proposition 8. *With the setting of the above definition, we have the fourier expansion as*

$$\mathbf{F}_{S|\bar{J}} f(z) = \sum_{T \subseteq \bar{J}} \hat{f}(S \cup T).z^T \tag{5.2}$$

i.e,

$$\widehat{\mathbf{F}_{S|\bar{J}} f}(T) = \hat{f}(S \cup T) \tag{5.3}$$

This can be proved quite easily as illustrated in detail in Ryan O'Donnell's page[1].

Corollary. *Let $f : \{-1,1\}^n \to \mathbb{R}$ and let $(J, \bar{J})$ be a partion on $[n]$. Let $S \subseteq J$ and $z \sim \{-1,1\}^{\bar{J}}$ is chosen uniformly at random. Then*

$$\mathbf{E}_z[\widehat{f_{J|z}}(S)] = \hat{f}(S) \tag{5.4}$$

$$\mathbf{E}_z[\widehat{f_{J|z}}(S)^2] = \sum_{T \subseteq \bar{J}} \hat{f}(S \cup T)^2 \tag{5.5}$$

Proof. The first statement can be immediately proved by taking $T = \emptyset$ in eqn(5.2). Coming to the second statement we need to apply the definition first, then the parseval's theorem and finally apply the proposition in eqn(5.2) as follows

$$\mathbf{E}_z[\widehat{f_{J|z}}(S)^2] = \mathbf{E}_{\mathbf{z}}[\mathbf{F}_{S|\bar{J}} f(z)^2] = < \mathbf{F}_{S|\bar{J}} f(z), \mathbf{F}_{S|\bar{J}} f(z) > = \sum_{T \subseteq \bar{J}} \widehat{\mathbf{F}_{S|\bar{J}} f}(T)^2 = \sum_{T \subseteq \bar{J}} \hat{f}(S \cup T)^2$$

□

[1] http://www.contrib.andrew.cmu.edu/~ryanod/?p=560#propfrestr-subcube-formula

5.4.2 Goldreich Levin Theorem

The Goldreich Levin Theorem[16] is a learning algorithm that the solves the problem: Given query access to a target function $f : F_2^n \to F_2$,where F represents the boolean logic domain and not real 0 and 1, the algorithm finds all of the **linear** functions with which f is at least slightly correlated. Equivalently, it is an algorithm which finds all of the noticeably large fourier coefficients of f.

Before we proceed we must define what we mean by "linear" with regard to boolean functions.

Definition 9. A function $f : F_2^n \to F_2$ is linear if either of the following equivalent conditions hold:

- $f(x + y) = f(x) + f(y)$ for all $x, y \in F_2^n$
- $f(x) = a.x$ for some $a \in F_2^n$ that is, $f(x) = \sum_{i \in S} x_i$ for some $S \subseteq [n]$

We can intuitively understand the linearity of a boolean function in terms of its proximity to any of its basis functions. If it has a very large fourier coefficient which means that its highly correlated with one of the parity functions. Essentially the Goldreich-Levin Theorem finds all the fourier coefficients or the parity function in the decreasing order of correlation.

Theorem. *Given query access to a target* $f : \{-1, 1\}^n \to \{-1, 1\}$ *as well as input* $0 < \tau \leq 1$*, there is a poly*$(n, 1/\tau)$*-time algorithm with high probability outputs a list* $L = \{U_1, U_2, \ldots, U_l\}$ *of subsets of* $[n]$ *such that:*

- $|\hat{f}(U)| \geq \tau \Rightarrow U \in L$
- $U \in L \Rightarrow |\hat{f}(U)| \geq \tau/2$

And by parseval's theorem we know that $|L| \leq 4/\tau^2$

Proof. The Goldreich Levin theorem essentially seeks the fourier coefficients $\hat{f}(U)$ with magnitude at least τ. Given any candidate $U \subseteq [n]$ we distinguish them as large, $|\hat{f}(U)| \geq \tau$ or small, $|\hat{f}(U)| \leq \tau/2$. The trouble is that there are 2^n potential candidates and going through all of them would make the algorithm exponential. Hence the Golreich Levin theorem employs a divide and conquer strategy where we measure measure the fourier weights of f on various collection of sets. □

Initially, all 2^n set U are implicitly put in a single "bucket". The algorithm essentially repeats the following loop:

- Select any bucket $\mathcal{B}$ containing 2^m sets, $m \geq 1$
- Split it into two buckets $\mathcal{B}_1, \mathcal{B}_2$ of 2^{m-1} sets each.
- "Weigh" each $\mathcal{B}_i$, $i = 1, 2$; that is, estimate $\sum_{U \in \mathcal{B}_i} \hat{f}(U)^2$

- Discard $\mathcal{B}_1$or $\mathcal{B}_2$if its weight estimate is at most $\tau^2/2$

The algorithm stops once all the buckets contain just 1 set and then it outputs the list of these sets. The buckes are indexed by an integer $0 \leq k \leq n$ and a subset $S \subseteq [k]$. The bucket $\mathcal{B}_{\mathrm{k,S}}$is defined by

$$\mathcal{B}_{\mathrm{k,S}} = \{S \cup T : T \subseteq \{k+1, k+2, \ldots, n\}\}$$

Note that the $|\mathcal{B}_{\mathrm{k,S}}| = 2^{n-k}$. The initial bucket is $\mathcal{B}_{0,\emptyset}$ and the algorithm always splits a bucket $\mathcal{B}_{\mathrm{k,S}}$ into two buckets $\mathcal{B}_{\mathrm{k+1,S}}$ and $\mathcal{B}_{\mathrm{k+1,S\cup\{k+1\}}}$. The final singleton buckets will be of the form $\mathcal{B}_{\mathrm{n,S}} = \{S\}$.

All that is left is to estimate $\sum_{U\in\mathcal{B}_i} \hat{f}(U)^2$ and subsequently apply Chernoff bound to obtain the approximate value of the fourier coeffients in the corresponding bucket. This is done as follows[2][3]

$$\sum_{U\in\mathcal{B}_i} \hat{f}(U)^2 = \sum_{U\in\mathcal{B}_{k,S}} \hat{f}(S \cup T : T \subseteq \{k+1, k+2, \ldots, n\})^2 = \mathbf{E}_{z\sim\{-1,1\}^{\bar{J}}}[\widehat{f_{J|z}}(S)^2]$$

$$\mathbf{E}_{z\sim\{-1,1\}^{\bar{J}}}[\widehat{f_{J|z}}(S)^2] = \mathbf{E}_{z\sim\{-1,1\}^{\bar{J}}}\left[\mathbf{E}_{y\sim\{-1,1\}^{J}}\left[f(y,z)\chi_S(y)\right]^2\right]$$

$$= \mathbf{E}_{z\sim\{-1,1\}^{\bar{J}}}\left[\mathbf{E}_{y,y'\sim\{-1,1\}^{J}}\left[f(y,z)\chi_S(y).f(y',z)\chi_S(y')\right]\right]$$

where y, y' are independent. A Chernoff bound can be applied which implies that $O(log(1/\delta)/\epsilon^2)$ samples are sufficient to estimate it with mean ϵ and confidence $1-\delta$.

Combining both this result with the fact that in the divide-and-conquer rule we must make at most $4n/\tau^2$ estimates which inturn implies that the algorithm runs for $O(log(1/\delta)/\epsilon^2, n/\tau^2)$. In general we can ensure that we get all the weightings to be accurate with high probability with the overall runtime of the algorithm being $poly(n, 1/\tau)$.

5.5 Learning Decision Trees

The approach would be same as in the previous cases. We would like to find when the fourier concentration of the function f is concentrated that is up to what degree are the bulk of the fourier spectrum concentrated. The beauty of the decision tree model is illustrated here[12].

Proposition. *Let $f : \mathbb{F}_2^n \to \{-1, 1\}$be computable by a decision tree T of size s and let $\epsilon \in (0, 1]$. Then the spectrum of f is ϵ- concentrated on degree up to $log(s/\epsilon)$.*

[2]As illustrated in http://www.contrib.andrew.cmu.edu/~ryanod/?p=589
[3]From eqn(5.5)

Proof. To prove this construct another decision tree H which is essentially obtained by truncating T at depth $log(s/\epsilon)$ and we add a leaf with value $+1$. Let the probability of reaching one of these leaves be ϵ, therefore $Pr[T \neq H] \leq \epsilon$. Since H is a decision tree we can write a term[4] for each leaf and we are guarenteed that exactly one term is true for each input. Each term for the decision tree H has at most t variables and all the coefficients of sets larger than size t is zero. Therefore, $\hat{H}(S) = 0$ for any S such that $|S| > log(s/\epsilon)$ which inturn implies $Pr[T \neq H] \leq \epsilon$ and hence the spectrum of f is ϵ- concentrated on degree up to $log(s/\epsilon)$. □

5.6 Leaning DNFs

DNFs are known to exhibit very strong fourier spectrum characteristics in general. In order to understand the dynamics behind the DNFs fourier spectrum we need to look into Random restricions which are pivotal to the study of not just DNFs but all boolean functions in general as eloquently put in [3].

5.6.1 Random Restrictions

In this section we describe the method of applying random restrictions and study some quintessential properties governing them. This is a very "Fourier-friendly" way of simplifying the boolean functions as we'll come to see below.

Definition 10. For $\delta \in [0, 1]$, we say that J is a δ-random subset of N if it is formed by including each element of N independently with probability δ. We define a δ-random restriction on $\{-1, 1\}^n$ to be a pair $(J|z)$, where first J is chosen to be a δ-random subset of $[n]$ is free if $i \in J$ and is fixed if $i \notin J$. An equivalent definition is that each coordinate i is (independently) free with probability δ and fixed to ± 1 with probability $(1 - \delta)/2$ each.

The properties exhibited by random restrictions are very much similar to the restrictions we discussed earlier in sec. 5.4.1 but only differ due to the fact that the set J is associated with a probability dependent on δ. For instance, in the case of normal restrictions we have $\mathbf{E}_z[\widehat{f_{J|z}}(S)] = \hat{f}(S)$ but in the case of random restrictions, since the set has a probability of $\delta^{|S|}$ associated with it, we get

$$\mathbf{E}_z[\widehat{f_{J|z}}(S)] = \mathbf{Pr}[S \subseteq J].\hat{f}(S) = \delta^{|S|}\hat{f}(S) \tag{5.6}$$

and similarly,

$$\mathbf{E}_z[\widehat{f_{J|z}}(S)^2] = \sum_{T \subseteq \bar{J}} \mathbf{Pr}[T \subseteq \bar{J}].\hat{f}(S \cup T)^2 = \sum_{U \subseteq [n]} \mathbf{Pr}[U \cap J = S].\hat{f}(U)^2 = \sum_{U \ni S} \delta^{|S|}(1-\delta)^{|U \setminus S|}\hat{f}(U)^2 \tag{5.7}$$

where U represents all the subsets of $[n]$ containing the set S.

[4] A term is basically a conjuction of all terms in the path from root to leaf

5.6.2 Hastad's Switching Lemma

Hastad's Switching Lemma is one of the most powerful tools in boolean complexity theory. The switching lemma essentially analyses the effect of random restrictions on DNFs. It goes on to say that under such random restrictions the DNFs seems to get simplified in surprising ways, that is, under random restriction with very high probability the DNF can reduce to a short Decision Tree therefore enabling us to draw strong strong fourier spectrum properties from it.

Theorem. ***Hastad's Switching Lemma[15, 13]*** *Let $f : \{-1,1\}^n \to \{-1,1\}$ be a width w DNF. Let $(J|z)$ be a δ-random restriction on it where $\delta \ll 1/w$, then*

$$\mathbf{Pr}_{\mathbf{z}}[DT - depth(f_{J|z}) \geq k] \leq (5\delta w)^k$$

Looking at the RHS and given that $\delta \ll 1/w$, we know that $\delta w \ll 1$ and hence consequently $(\delta w)^k \ll 1$ and very close to 0. Let's try to understand this statement intuitively. On applying a δ-random restriction one of the three following things can occur:

- First and by far the most likely, one of the literals in a term may be fixed to $False$ and we can delete it
- Otherwise all the literals in the term may be made $True$ in which the DNF reduces to a constant function
- The last possibility is when at least one of the term's literals is left free and the rest are fixed to $True$.

Notice that only in the third scenario that the DNF is not trivialized by the random restriction.

The proof for the Hastad's Switching Lemma is beyond the scope of this survey and a detailed analysis of the switching lemma can be found in the Switching Lemma Primer by Paul Beame[5].

Having established the Hastad's Switching Lemma we can now talk about the fourier spectrum of the DNF.

Lemma. ***[15]*** *Let $f : \{-1,1\}^n \to \{-1,1\}$ and let $(J|z)$ be a δ-random restriction, $\delta > 0$. Fix $k \in \mathbb{N}^+$and write $\epsilon = \mathbf{Pr}_{\mathbf{z}}[DT - depth(f_{J|z}) \geq k]$. Then the fourier spectrum of f is 3ϵ-concentrated on degree up to $3k/\delta$.*

Proof. The key observation to be made is that $DT - depth(f_{J|z}) < k$ implies that $deg(f_{J|z}) < k$ from sec. 4.2.1. This can be mathematically represented as

$$\mathbf{E}_{(J|z)}\left[\sum_{|S|\geq k} \widehat{f_{J|z}}(S)^2\right] \leq \epsilon$$

[5]homes.cs.washington.edu/~beame/papers/primer.ps

$$\mathbf{E}_{(J|z)}\left[\sum_{|S|\geq k}\widehat{f_{J|z}}(S)^2\right] = \sum_{|S|\geq k}\left[\mathbf{E}_{(J|z)}\widehat{f_{J|z}}(S)^2\right]$$

Using the result from eqn(5.7) we get,

$$= \sum_{|S|\geq k}\sum_{U\subseteq[n]}\mathbf{Pr}[U\cap J = S].\hat{f}(U)^2 = \sum_{U\subseteq[n]}\mathbf{Pr}_{(\mathbf{J}|\mathbf{z})}[|U\cap J|\geq k].\hat{f}(U)^2$$

The distribution of $|U\cap J|$ is Binomial$(|U|,\delta)$ the mean for which is $|U|.\delta$. When $|U|\geq 3k/\delta$ this random variable has mean at least $3k$, and a Chernoff bound shows $\mathbf{Pr}[|U\cap J|\geq k]\leq exp(-\frac{2}{3}k)\leq\frac{2}{3}$. Thus we get

$$\epsilon \geq \sum_{U\subseteq[n]}\mathbf{Pr}_{(\mathbf{J}|\mathbf{z})}[|U\cap J|\geq k].\hat{f}(U)^2 \geq \sum_{|U|\geq 3k/\delta}(1-2/3).\hat{f}(U)^2$$

and hence $\sum_{|U|\geq 3k/\delta}\hat{f}(U)^2\leq 3\epsilon$ as claimed.[6] □

[6]The proof follows suit from http://www.contrib.andrew.cmu.edu/~ryanod/?p=811

6 Read-Once Functions

6.1 What are Read-Once Functions?

Read-Once booleans functions are certainly one of the most interesting special families of Boolean functions. A function f can be called read-once if it can be represented by a Boolean expression using the operations of conjunction, disjunction and negation, in which every variable appears exactly once. We call such an expression a read-once expression for f. For example the function

$$f_0(a, b, c, d, e, f) = ab \vee ac \vee de \vee ef$$

is a read-once function since it can be factored into the expression

$$f_0 = a(b \vee c) \vee e(d \vee f)$$

which is a read-once expression. [1]

6.2 Representation of ROFs

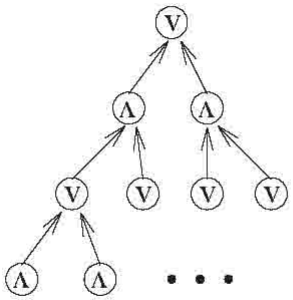

Figure 6.1: Canonical ROF representation

The above figure illustrates the canonical form of representing the ROFs. One of the the stark differences between DNFs and ROFs is that the depth is not restricted to 2 as in the case of DNFs but instead can extend up to $O(n)$.

[1] For more information on Read-Once Functions refer [9]

6.3 Analysing ROFs

We would like to study the effects of random restrictions on read-once functions and check if with high probability it reduces to a DT of small depth so that we can draw useful fourier spectrum inferences from the ROF function. Consider a Decision Tree of depth d and a read-once function of depth d. Let both the DT and the ROF be in canonical form. The number of variables be n say, for the decision tree. The maximum depth of the decision tree would be n and if it had been a DNF it will be 2 but in the case of the ROF it need not be bounded.

Therefore if we were to substitute variables in order to try and achieve a constant function, in the case of the decision tree if we were to substitute $O(depth)$ variables carefully chosen we can reduce it to a constant function. But in the case of a read-once function, in general, by substituting value for one variable we eliminate just the variable symmetrically next to it. This means that we should essentially make $O(n/2)$ substitutions which is quite a formidable task while employing a random restriction.

Now we should address two questions pertaining to the above statement.

1. If there were to exist a random restriction on the ROF such that it reduces to a decision tree of small depth with very high probability, does it imply a fourier concentration result for the ROF?
2. Is it possible if at all to obtain a random restriction which reduces the ROF to a short decision tree?

Essentially we would like to either prove or state otherwise the above propositions to further our understanding of ROFs.

6.4 Fourier Spectrum of ROF after reduction to short DT

As mentioned earlier, in this section we work under the assumption that we have a random restriction which reduces the ROF to a short DT with high probability. The statement can be formally drawn as:

Lemma. *Let $f : \{-1,1\}^n \to \{-1,1\}$ corresponding to a read-once function and let$(J|z)$ be a δ-random restriction, $\delta > 0$. For integer values of $k > 0$ we write $\epsilon = \mathbf{Pr}_{\mathbf{z}}[DT - depth(f_{J|z}) \geq k]$. We would like to see if we could get a stronger bound that the case for DNFs.*

$$\mathbf{E}_{(J|z)}\left[\sum_{|S|\geq k} \widehat{f_{J|z}}(S)^2\right] \leq \epsilon$$

$$\mathbf{E}_{(J|z)}\left[\sum_{|S|\geq k}\widehat{f_{J|z}}(S)^2\right] = \sum_{|S|\geq k}\left[\mathbf{E}_{(J|z)}\widehat{f_{J|z}}(S)^2\right]$$

Using the result from eqn(5.7) we get,

$$= \sum_{|S|\geq k}\sum_{U\subseteq[n]} \mathbf{Pr}[U\cap J = S].\hat{f}(U)^2 = \sum_{U\subseteq[n]} \mathbf{Pr}_{(\mathbf{J}|\mathbf{z})}[|U\cap J|\geq k].\hat{f}(U)^2$$

The distribution of $|U \cap J|$ is Binomial$(|U|, \delta)$ the mean for which is $|U|.\delta$. When $|U| \geq 3k/\delta$ this random variable has mean at least $3k$, and a Chernoff bound shows $\mathbf{Pr}[|U\cap J|\geq k] \leq exp(-\frac{2}{3}k) \leq \frac{2}{3}$. Thus we get

$$\epsilon \geq \sum_{U\subseteq[n]} \mathbf{Pr}_{(\mathbf{J}|\mathbf{z})}[|U\cap J|\geq k].\hat{f}(U)^2 \geq \sum_{|U|\geq 3k/\delta}(1-2/3).\hat{f}(U)^2$$

and hence $\sum_{|U|\geq 3k/\delta}\hat{f}(U)^2 \leq 3\epsilon$.[2] Notice that at no point we were able to factor any distinguishing property exhibited by ROFs and hence we were unable to get a stronger bound than one for DNFs. In fact we can go on to say that for any boolean function $f : \{-1,1\}^n \to \{-1,1\}$ if it can be reduced to a short decision tree by some random restriction with high probability then we can say that the fourier spectrum of f is 3ϵ-concentrated on degree up to $3k/\delta$.

6.5 ROF under Random Restrictions

A minterm can be viewed as a subtree of a ROF which is essentially constructed by considering both the children while encountering an AND gate and pick one of the children while encountering a OR gate. As illustrated in the figure below, it essentially represents an AND tree linked by OR.

Our goal is to show that under a random restriction the probability that minterms more than width k exist is very low or very high. For a ROF of the canonical form, we have the width of the minterm as $n/4$. Consider the example illustrated above wherein the minterm constructed has width 4 coherent with our previous proposition. Now if we want to evaluate the probability that under a δ-random restriction the ROF reduces to a depth 2 ROF, we need to show that at least one minterm with width 2 exists with high probability. And we can say that there will be $n/2$ minterms in all for the canonical representation hence in our case 8 [3]minterms in all. In the following we bound the probability that any given minter, reduces to a small width term.

Consider a read-once function of the canonical form with n variables and depth $log_2 n$. We use this canonical form as the limiting worst-case scenario and analyse

[2]The proof follows suit from http://www.contrib.andrew.cmu.edu/~ryanod/?p=811

[3]2*2*2 cases resulting from the choices at the OR gates

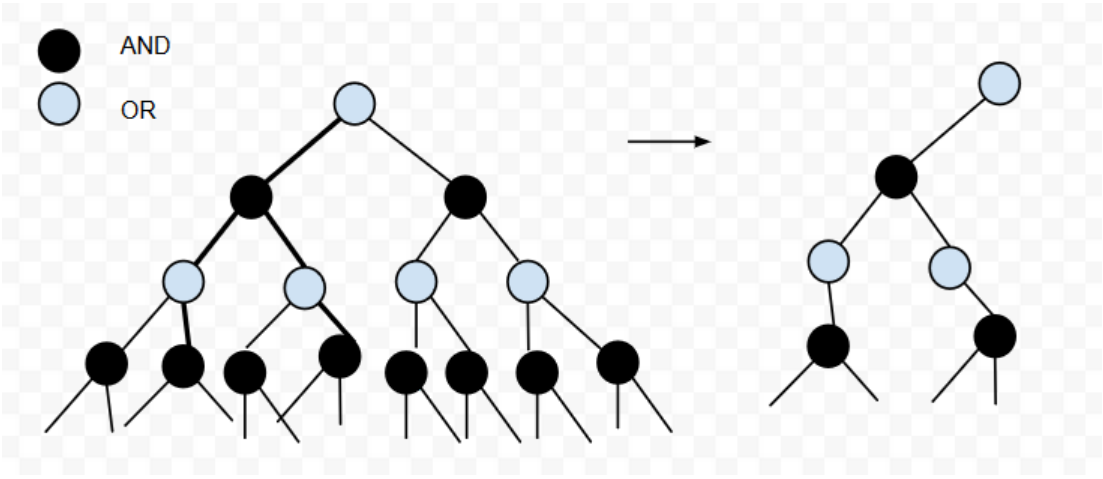

Figure 6.2: Minterm of a Read-Once Function

this. In doing this we intend to decrease the width of the minterm of this canonical ROF under a δ-random restriction by 2. The probabity associated with it assuming for even depth $d = log_2 n$ is obtained by setting one of the two variables in the AND nodes right above the leaf nodes. Since there are $n/4$ variables at depth d for the minterm subtree we should make $n/4$ substitutions and the probability associated will be

$$Pr[depth - 2] = [\delta.(1-\delta)/2 + \delta.(1-\delta)/2]^{n/8} = [\delta.(1-\delta)]^{n/8}$$

Now essentially this is for reducing the depth of the minterm by 2 say we intend to decrease the overall depth of the ROF to t through a δ-random restriction. Then we should union bound the above probability over $(n-t)/2$ iterations with n being constantly updated over every iteration.

$$Pr[depth - t] = \sum_{i=t+2}^{n} [\delta.(1-\delta)]^{n/8} \leq \frac{(n-t)}{2} [\delta.(1-\delta)]^{n/8} \leq \frac{(n-t)}{2} 2^{-n/4}$$

The second part of the inequality rises from the fact that the maximum value taken by $\delta.(1-\delta)$ is $1/4$ when $\delta = 1/2$. As we can observe from the above equation that the probability that the minterm reduce to depth t under a δ-random restriction is extremely small and we can say with high probability that a ROF does not reduce to a short decision tree under a random restriction.

Bibliography

[1] The fourier tranformations - but why? intuitive mathematics. Electronic Medium, June 2012.

[2] Carlisle M Adams. Symmetric cryptographic system for data encryption, April 23 1996. US Patent 5,511,123.

[3] Paul Beame. A switching lemma primer. Technical report, University of Washington, 1994.

[4] Wolfram Buttner and Helmut Simonis. Embedding boolean expressions into logic programming. *Journal of Symbolic Computation*, 4(2):191–205, 1987.

[5] Claude Carlet. Boolean functions for cryptography and error correcting codes. *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, 2:257, 2010.

[6] Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

[7] Paul E Dunne, Wiebe van der Hoek, Sarit Kraus, and Michael Wooldridge. Cooperative boolean games. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*, pages 1015–1022. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[8] Christophe Garban and Jeffrey E Steif. Noise sensitivity and percolation. *Preface vii Schramm-Loewner Evolution and other Conformally Invariant Objects 1 Vincent Beffara Noise Sensitivity and Percolation 49*, page 49, 2012.

[9] Martin Charles Golumbic, Aviad Mintz, and Udi Rotics. An improvement on the complexity of factoring read-once boolean functions. *Discrete Applied Mathematics*, 156(10):1633–1636, 2008.

[10] Jeff Kahn, Gil Kalai, and Nathan Linial. The influence of variables on boolean functions. In *Foundations of Computer Science, 1988., 29th Annual Symposium on*, pages 68–80. IEEE, 1988.

[11] Jerry S Kelly. *Social choice theory*. Springer, 1988.

[12] Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

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

[14] Yishay Mansour. Learning boolean functions via the fourier transform. In *Theoretical advances in neural computation and learning*, pages 391–424. Springer, 1994.

[15] Ryan O Donnell. Analysis of boolean functions. *Lecture Notes. Available online at http://www. cs. cmu. edu/odonnell/boolean-analysis*, 9:13–49, 2007.

[16] Ryan O Donnell. The goldreich-levin theorem. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 569–578. ACM, 2008.

[17] Ryan O Donnell. Some topics in analysis of boolean functions. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 569–578. ACM, 2008.