People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University M'Hamed BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering

Department of Electronics

Final Year Project Report Presented in Partial Fulfilment of the Requirements for the Degree of

MASTER

In **Electrical and Electronic Engineering**

Option: **Telecommunications**

Title:

Low Density Parity Check Coding

Presented by:

- **BRAHIMI Mohamed Amine**
- **YOUNSI Bilel**

Supervisor:

Dr. DAHIMENE Abdelhakim

Registration Number:……/2016

Acknowledgment

I would like to express my gratitude to my supervisor Dr. Abdelhakim Dahimene, whose teaching approach made me enjoy the mathematical theory of communication and eventually had my master graduation project under his supervision.

With all my love, I would like to thank my family and my dear fiancée for all their emotional support during the whole process.

I should not forget my friends; without whom, I would not be able to achieve anything.

Amine.

Acknowledgement

I wish to express my sincere appreciation to Dr. Abdelhakim Dahimene for his guidance and encouragement throughout this final year project.

I thank my family for their infinite patience and love. I must not forget Imen for here encouragement.

I highly appreciate the encouragement of my friends who fully supported me in the difficult moments and who shared with me amazing memories.

Abstract

Since their rediscovery by Mackay Neal in 1993 [11], LDPC codes have been an integral part of the coding field, and by now, they have been in many standards and are suggested for others. In terms of performance, many practical results have shown that those codes are actually very performant at rates that are within a fraction of capacity.

In this Master Project report, the general theory of LDPC codes, including construction, encoding and decoding procedures, is reviewed in the first chapters. In the simulation chapter, we consider the evaluation of error performance of short to medium length LDPC codes with BPSK modulation through the binary AGWN channel.

Keywords: BER, Bit-Flipping, Codeword, Coding theory, Decoder, Encoder, Graph, Message Passing, Parity-Check, Sum-Product, SNR.

Abbreviations

AGWN	**A**dditive **G**aussian **W**hite **N**oise
BCH	**B**ose Ray-**C**haudhuri **H**ocquenghem
BER	**B**it **E**rror **R**ate
BF	**B**it **F**lipping
BPSK	**B**inary **P**hase **S**hift **K**eying
BSC	**B**inary **S**ymmetric **C**hannel
GDBF	**G**radient **D**escent **B**it **F**lipping
LDPC	**L**ow **D**ensity **P**arity **C**heck
MP	**M**essage **P**assing
PEG	**P**rogressive **E**dge **G**rowth
QC	**Q**uasi-**C**yclic
SNR	**S**ignal to **N**oise **R**atio
SP	**S**um **P**roduct
SPA	**S**um **P**roduct **A**lgorithm
WBF	**W**eighted **B**it **F**lipping

List of figures

List of figures

List of tables

Abbreviations

AGWN	**A**dditive **G**aussian **W**hite **N**oise
BCH	**B**ose Ray-**C**haudhuri **H**ocquenghem
BER	**B**it **E**rror **R**ate
BF	**B**it **F**lipping
BPSK	**B**inary **P**hase **S**hift **K**eying
BSC	**B**inary **S**ymmetric **C**hannel
GDBF	**G**radient **D**escent **B**it **F**lipping
LDPC	**L**ow **D**ensity **P**arity **C**heck
MP	**M**essage **P**assing
PEG	**P**rogressive **E**dge **G**rowth
QC	**Q**uasi-**C**yclic
SNR	**S**ignal to **N**oise **R**atio
SP	**S**um **P**roduct
SPA	**S**um **P**roduct **A**lgorithm
WBF	**W**eighted **B**it **F**lipping

Chapter 1: Introduction

1. Development of information theory

Communication theory deals with the transmission of information from one point (information Source) to another (destination) through a transmission medium which called "the channel". A very generic communication system is shown in figure 1.1.

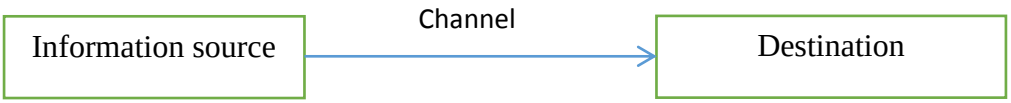


Figure 1.1: A basic communication system

When information is transmitted, it is subjected to error that may result from random noise, link failure, interference… etc., the main quest of information theory is how we can transmit Information at the highest rate possible with error being as small as possible.

In 1948, Claude Shannon introduced a limit in the transmission rate which he called "the capacity"; and he proved that as long as we are not transmitting at a rate higher than this quantity, we can minimize the error in the received message to a level that is as small as we want by using specific coding schemes [1].

However, Shannon did not provide a way or a specific algorithm to attain this and from that time on, coding theorists started inventing and optimizing codes that conform with Shannon's limit and reach the desired performance.

In essence, Coding is the process of adding redundancy to the message so that if an error took place in the channel it can be recovered [2], an example of that, is the following (5,1) repetition code.

Consider that we want to transmit message 110101, the coded message will then take following form:

111111111100000111110000011111

We transmit 5 bits for every message bit, so if the destination receives:

00111101010000100111010011110

Using the assumption, there is more correct bits than wrong ones, the decoder can easily recover the original codeword, omits the redundancy and produces the original message. However, if there are more erroneous bits than correct ones, we will have a decoding error and we say that the error correcting capabilities of the code has been reached.

2. Motivation for Coding

To avoid any possible ambiguity, the term coding is used to refer to "channel coding" which is used before transmission. Another type of coding which is "source coding" is not considered here.

Ideally speaking, the bit error for a given modulation scheme is uniquely governed by signal-bit energy per the noise power density ratio $\frac{E_b}{N_0}$ [3]. Mathematically speaking, the probability of error is given by:

$$P(e) = a \operatorname{erfc}\left(b.\sqrt{\frac{E_b}{N_0}}\right) \tag{1.1}$$

With $\operatorname{erfc}(x)$ being the complementary error function and a, b real valued constants that depend on the modulation scheme being applied.

The complementary error function is a decreasing function so by increasing $\frac{E_b}{N_0}$, either by increasing the transmission energy or using specific type of channels that have smaller N_0 we can minimize the probability of error in the transmitted message.

Ideally speaking, this method is totally valid. However, increasing the transmission energy is equivalent to high costs; besides, not all modulation schemes are power efficient [4], so changing the type of modulation used can increase the requirements on signal energy.

A second point is that the noise density N_0 is not a deterministic parameter to quantify exactly, we can only predict a range for it with some probability of occurrence.

By using coding, we reduce the requirement on $\frac{E_b}{N_0}$, which means less power signal and less hardware cost. Note that the power of the signal is not only a matter of cost, another benefit from reducing the transmission power is for health considerations, for example without coding, cell phones would have higher power to transmit the same message we transmit using today's phones which may affect brain and body in general [5].

3. The search for good codes

Since the time Shannon has published his paper, codes continued to evolve .in the years 1950-1960 many famous codes have been invented such as hamming [6] and BCH [7] codes.

Those codes are said to be block codes since the coding mechanism was done block by block. In the year 1962, Robert Gallager invented a type of block codes called low density parity-check codes [8]. However, they were soon forgotten due to their encoding and decoding complexity at a time where computational power was not that great compared to now.

the next decade (1970's) revolutionized the world of coding by introducing convolutional codes [9] which served as a new way of thinking in coding theory. In 1993 Berrou and Galavieux introduced a new type of codes called turbo codes [10] which were fast and more performant that soon were integrated in many applications. In 1997, Mackey and Neal published a paper on a type of codes that rely on a sparse parity check matrix [11], soon after the publication, the scientific community recognized that the MN codes are nothing but a rediscovery of the original LDPC codes proposed by Gallager in 1963 in his PhD thesis, it was shown that these codes were very performant at rates close to Capacity and it was not long that they got the attention to be used in application. This work deals with LDPC encoding and decoding.

4. Motivation

The adoption of LDPC codes for error correcting in high rate data communications have been increasing in recent years such that LDPC along with Turbo codes are the focus of many standards [12] [20] and the research to optimize them is going too fast.

LDPC codes are generally very lengthy and for complexity reasons the decoders are not Optimal but iterative with a maximum number of possible iterations. In our case, we are motivated to study the performance of LDPC codes under many constraints in the AWGN Channel.

5. Outline of the project

Along with the introduction, the rest of the thesis is organized as follows:

- Chapter 2 provides a general overview on the mathematical concepts that we need to tackle this subject, with a focus on the structure of finite fields and the preliminaries of Graph Theory.

- Chapter 3 starts by introducing Shannon's theorem and the motivation behind Coding theory, then a formal definition of block codes, from which we focus on linear and cyclic codes, is given. This chapter ends with a general description of LDPC codes.

- In Chapter 4, A formal treatment of the subject is given, including the definitions, code representations and construction methods.

- Chapter 5 focuses on the encoding process of LDPC codes from the generator matrix as

well as linear encoding from the H matrix, In This Chapter, we assume we have constructed the parity check matrix of the code.

- Chapter 6 deals with the decoding side of LDPC codes. Many types of decoders that are used in practice are presented.

- Chapter 7 is a simulation of the performance of a set of LDPC codes through the AWGN Channel under different conditions.

- In the Last Chapter, a summary of what has been done throughout the report is given with conclusions about each chapter and some threads of future work are suggested.

1. Introduction

Channel coding is considered as the most important result of information theory, thanks to it, reliable data transmission over unreliable channels has been made possible.
This later paradoxical fact was shown to be true by Claude Shannon himself [1]. In this chapter, the main theory and concepts of channel coding are presented.

2. The channel

The channel is the set of media that separate the transmitter from the receiver. Channels come in a variety of forms radio, wire, optical …, However, all channels have one thing in common, the signal undergoes degradation from the transmitter to the receiver [3].

Another view to look at the channel, is to consider it as a logical connection over a physical circuit [18].and this definition is very important in the analysis of communication systems. To define the channel from information theory point of view, we need to define some concepts, let us first consider the following diagram:

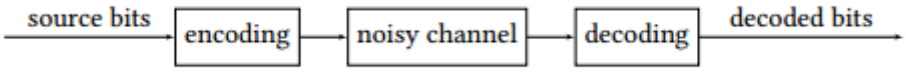


Figure 2.1: a general communication system with channel coding

- Source: the part of transmission system responsible of generating information.
- Alphabet: the set of symbols from which information sequences are created.
- Encoder: this block maps the information sequence into a codeword.
- Decoder: the work of this block is to do the inverse mapping done by the encoder, it guesses the correct codeword and recovers the original message.

Definition

Given an input and output alphabets I and O respectively, the channel the is mapping from I to O [14].

Note that the decoder and encoder are usually viewed as a part of the channel.

3. The Channel Capacity

The capacity is the upper bound on the rate at which information can be reliably transmitted. For a memoryless Channel the capacity is given by [2]:

$$C = \max_p I(x, y) \tag{2.1}$$

Where $I(x, y)$ is the mutual information and p is the symbol distribution of the input of the Channel.

4. Noise in communication Channels

Noise refers to random and unpredictable signals produced by natural processes, either internal or external to the system [17]. Noise cannot be totally avoided; it can only be limited by using low noise equipment, protecting the channel from external distribution or by providing suitable signals transmission power …etc.

In theory, noise is modeled as a stochastic process to help understand its behavior and to construct systems that are less affected by it. The following types of noise are very common in practice:

- AGWN: additive Gaussian white noise, this noise is caused by random motion of electrons due to thermal energy, it's presented in every communication channel.
- Shot noise: shot noise occurs whenever charged particles cross a potential barrier.

5. Noisy Channels

When the message is transmitted through the noisy channel, the destination does not receive the original message but a distorted one (at least from signal level point of view).

In digital communication, the message is composed of 1's and 0's, the decoder chose one if the voltage of the bit is higher than a predetermined value and zero otherwise, noise affects

this by changing signal levels, which leads to a corrupt message; when this happens we say an error in the transmitted message has happened. One way to correct this error is by using channel coding. Practically, all channels are noisy and according to the type of noise the channel is modelled. Two important Channel models are:

5.1 The Binary Symmetric Channel (BSC)

The BSC is a channel in which the input and output alphabets are the same [14]. More specifically, $I = O = \{0,1\}$;along with a parameter p called the cross-over probability that denotes the reliability of the channel in transmitting correctly the information. In other words each bit is independent from the other bits and has a probability p to be flipped and $1 - p$ to be received without error. The term 'symmetric' refers to the fact that the input and output ports are exchangeable. Figure 3.2 Shows the input-output diagram of the BSC Channel.

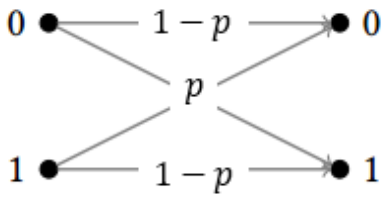


Figure 2.2: The input-output diagram of the BSC channel

If $H(p)$ is the entropy of the source then, the capacity of the BSC is:

$$C = 1 - H(p) \tag{2.2}$$

.

5.2 The binary additive white Gaussian noise (AWGN) Channel

Given an input Alphabet I= {0,1}, the binary AWGN maps I into R. The term "additive" refers to the way this mapping done [17]. Given an input message x, the output of the AWGN channel is $Y = x + e$ Where e is a Gaussian random variable with zero mean an variance σ^2.

When dealing with the AWGN the concept of SNR or signal-to-noise ratio is very important and is defined to be the ratio of the energy per transmitted symbol Es over the noise variance

$$SNR = \frac{E_s}{\sigma^2} = \frac{2E_s}{N_o} \tag{2.3}$$

Another quantity of great importance is the energy per bit which is

$$E_b = \frac{Es}{r} \tag{2.4}$$

This yields to the definition of Signal energy per bit to noise ratio

$$\frac{Eb}{N_o} = \frac{Es}{2r\sigma^2} \tag{2.5}$$

Note that those quantities are usually used in dB

Using those definitions, The AWGN Capacity is given by:

$$C = Wlog(1 + SNR) \tag{2.6}$$

With W being the channel bandwidth.

6. Channel coding and Shannon's theorem

Channel coding refers to the process of adding redundant bits to the transmitted messages so that if an error happened in the channel, the original message can be recovered using the added redundancy as Shannon proved in his paper [1].

6.1 Shannon's theorem

Definition

Consider a channel with capacity C ,let R any positive number with $R < C$, There exists always a sequence Cn of r-ary codes with a rate R such that the probability of error tends to 0 as n tends to infinity[2].

Note that The code rate is defined as

$$R = \frac{log_r M}{n} \tag{2.7}$$

With M being the number of valid codewords which is equal to the number of message sequences.

For simplicity, we let k= $log_r M$, with k being the number of bits per information sequence and we write

$$R = \frac{k}{n} \tag{2.8}$$

7. Block codes:

The type of codes in which we are interested, do belong to a family of codes called block codes.

a. Definition:

Let the finite set $S = \{s_1, s_2, \dots, s_r\}$ be a code alphabet and let S^n be the set of n-tuples over S. A nonempty set C$\subseteq S^n$ is called an r-ary block code[14].

Remarks:

- Usually S is a finite field, more specifically, an extension of the binary field.
- If S is a field then S^n is the nth extention of S or it is the vector space of dimension n over S.
- The n-tuples of C are called codewords

There are codes that belong to this family and are very important in practice which are linear and cyclic codes. LDPC codes which are the main topic of this project are linear block codes.

8. Linear block codes

A code $L \subseteq S^n$ is a linear code if and only if L is a subspace of S^n. The term linear comes From the fact that the sum of any two codeword is a valid codeword. The fact that linear bock codes are linear mappings gives us the ability to represent them by matrices [14].

8.1 Generator matrix

Let L be an (n, k) linear block code, a $k \times n$ matrix G whose row vectors form a basis for L is called the generator matrix for L [14]. In other words, given the information space S^k we can write

$$L = \{ c = i.G | i\epsilon S^k \}. \tag{2.9}$$

Remarks:

- If the codeword is in the form of $c = \left(i , x_p \right)$ wherex_p is the redundant (parity-check) vector, we say the coding is systematic and the corresponding generator matrix will take the form, $G = [I_k \ P]$ with I_k being the kxk identity matrix and P a $k\, x(n-k)$ matrix.

To illustrate this, let us consider the generator matrix of the (7, 4) hamming code:

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{2.10}$$

In this code, the information vector belongs to the vector space V= $\{0,1\}^4$ which is the fourth extension of the binary field. To show how encoding is done let us randomly choose an information vector and find the corresponding codeword.

Suppose i= 0101, then

$$c = [0\ 1\ 0\ 1]\begin{bmatrix}1 & 0 & 0 & 0 & 1 & 1 & 1\\ 0 & 1 & 0 & 0 & 0 & 1 & 1\\ 0 & 0 & 1 & 0 & 1 & 0 & 1\\ 0 & 0 & 0 & 1 & 1 & 1 & 0\end{bmatrix} = [0\ 1\ 1\ 0\ 1\ 1\ 0] \tag{2.11}$$

The generator matrix is used to encode the information bits into codewords. Another matrix that is used to check if the received word is indeed the transmitted one, is called the parity check matrix.

8.2 Parity-check matrix

Given an (n, k) linear block code L, we call the $(n-k) \times n$ matrix H whose null space is C [14] , the parity check matrix and we write

$$C = \{c \in S^n |\ c.H^T = 0^T\} \tag{2.12}$$

The following matrix is the parity check matrix for the hamming code (7,4)

$$H = \begin{bmatrix}1 & 0 & 1 & 1 & 1 & 0 & 0\\ 1 & 1 & 0 & 1 & 0 & 1 & 0\\ 1 & 1 & 1 & 0 & 0 & 0 & 1\end{bmatrix} \tag{2.13}$$

The H matrix in this example is in systematic form, strictly speaking, a parity-check matrix H is said to be in systematic form if

$$H = \left[\ P^T\ I_{n\_k}\ \right] \tag{2.14}$$

Remarks:

- The linear system of equations that results from $c.H^T$, are called parity check equations.

- A codeword is valid if the linear system is homogenous .i.e. $c.H^T = 0$
- Error is modeled as a vector such that $e = \{ e \epsilon S^n \ | e \notin C \}$
- The receiver receives the vector r such that $r = c + e$
- The vector $S = r.H^T$ is called the syndrome vector.
- the syndrome vector may be further simplified.

$$S = X.H^T = C.H^T + e.H^T = e.H^T \tag{2.16}$$

- The generator and parity check matrices are orthogonal.

There are types of decoders that use the syndrome vectors to decide on the valid code word and are usually called syndrome decoders. However, not all decoders use the syndrome to decide on the codeword, there some of them that make use of the parity check equations directly like flip decoders as we are going to see later in LDPC decoding.

9. Cyclic codes

Definition:

A linear code c is called cyclic if and only if the following statement is valid [2]

$$[(c_0, c_1, c_{,2}, \dots, c_{n-1}) \epsilon C \rightarrow (c_{n-1}, c_0, \dots, c_{n-2}) \epsilon C] \tag{2.17}$$

9.1 Polynomial representation of cyclic codes

If we represent the codewords as polynomials in $F_2[x]$, then a code L is cyclic if and only if L is an ideal in $F_{2_{/X^n-1}}[x]$ [2].

In other words:

- if c is an ideal in $F_{2_{/X^n-1}}[x]$

Then $$c = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n+1} \tag{2.18}$$

- For any codeword $c(x)$, $x.c(x)$ is also a codeword.
- The codeword $c' = x^i.c$ is c shifted by $i\ mod(n)$ to the right.

9.2 The generator polynomial of a cyclic code

Definition

Let C be a cyclic code of length n, There exists a unique monic polynomial of minimum degree in C called the generator polynomial denoted $g(x)$ and this polynomial generates C[19].

Properties of the generator polynomial

- $g\ (x)|x^n - 1$
- $Degree(g)\ + Dim(C) = n$
- $g_0 \neq 0$

Given

$$g(x) = x^r + g_{r-1}x^{r-1} + g_{r-2}x^{r-2} + g_0 \tag{2.19}$$

The generator matrix G can be constructed by shift-cycling the vector

$$(\ g_0, g_1, \dots, g_{r-2}\ , g_{r-1}, 1, 0_{1x(n-r+1)}) \tag{2.20}$$

$$G = \begin{bmatrix} g_0 & g_1 & g_2 & \cdots & 1 & 0 & 0 & \dots & 0 \\ 0 & g_0 & g_1 & g_2 & \dots & 1 & 0 & \dots & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & \cdots & 1 & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \dots & 0 & g_0 & g_1 & g_2 & \dots & 1 \end{bmatrix} \tag{2.21}$$

Given an information polynomial

$$i(x) = i_0 + i_1x + i_2x^2 + \cdots + i_{k-1}x^{k-1} \tag{2.22}$$

The Corresponding code word is easily obtained by a simple multiplication

$$c(x) = i(x)g(x) \tag{2.23}$$

9.3 The check polynomial of a cyclic code

The check polynomial $h(x)$ is a polynomial that satisfies the following[19]:

$$\begin{cases} g(x)h(x) = x^n - 1 & \text{over } F_2[x] \\ g(x)h(x) = 0 & \text{over } F_{2_{/x^n-1}}[x] \end{cases} \quad (2.24)$$

Since

$$c(x) = i(x)g(x) \quad (2.25)$$

Then

$$c(x)h(x) = i(x)g(x)h(x) = 0 \quad (2.26)$$

Which means that the check polynomial $h(x)$, is equivalent to the H matrix in matrix representation.

The following polynomials are also important:

- The error polynomial $e(x)$ defined as:

$$e(x) = r(x) - c(x) \quad (2.27)$$

With $r(x)$ being the received vector.

- The syndrome polynomial $s(x)$ defined as:

$$s(x) = r(x)h(x) = e(x)h(x) \quad (2.28)$$

The syndrome and error vectors are used for the same reasons as their counterparts in Matrix representations.

10.LDPC Codes

LDPC Codes are linear block codes with the constraint on H to be sparse. The sparsity of H helps in reducing the complexity of the encoding and decoding process since the zero entries of the parity check matrix are neglected in multiplication and addition. Despite the fact that they have been discovered 53 years ago, it is only recently that the they got the attention they deserve.

11.Summary

Information is sent from a transmitter to a receiver through a channel. Practically, this latter is always noisy which may distort the message.

One way to solve this problem is to use channel coding, which is the process by which the transmitted information can be recovered from an output of a noisy Channel by adding extra information in the form of check bits.

Codes come in different types, for our case the focus was on linear and cyclic block codes, due to their importance to our work. The chapter ends by a qualitative presentation of LDPC codes that serves as an introduction to a quantitative treatment of the subject in the next chapter.

1. Introduction

Low density parity check codes are forward error correction linear block codes that were first proposed by Gallager in his PhD thesis [8] in1962 at MIT but they remained ignored and deemed inefficient because of their computational demands that were impractical at that time.

As technology advanced, the need for near Shannon limit error performance was highly required, Mackay and Neal [11] rediscovered the original Gallager‘s LDPC codes and showed that they perform well at rates very close to the capacity and from that time on more modifications and enhancements were made .

Recently LDPC codes have drawn much attention and they are used in many standardization activities such as: IEEE 802.11n and 802.16e, ETSI DVBs and T6nSynC [20]. In this chapter, the basic terminology and concepts concerning LDPC are presented.

2. Low Density Parity Check codes

A Low Density Parity Check code is a class of linear block codes with a parity check matrix H that have the quality of being sparse i.e. H has only a very few non-zero entries. Formally H is sparse if it has O(n) non-zero elements [21].

Aside from the fact that LDPC codes’ parity check matrix is sparse, LDPC themselves are no different than any other block codes. Indeed, all the algorithms developed for LDPC can be used with other linear block codes if they can be represented by a sparse H.

We will see later in the thesis that the biggest difference between LDPC codes and other block codes is how they are decoded.

3. LDPC representation

3.1 matrix representation

We have already mentioned that LDPC codes are nothing but linear block codes with a

Sparse H, so LDPC codes are represented by a parity check matrix H from which the decoding (as well as the encoding) takes place. The H matrix provides us with the needed parity check equations whose results form the entries of the syndrome vector that is used in the decoding.

3.2 Tanner graph representation

Aside from this matrix form, LDPC codes are often represented in graphical form using Tanner graphs, this type of representation is so useful specially in message-passing decoding algorithms as we are going to see in the upcoming chapters.

Definition

A Tanner [22] graph is a bipartite graph i.e. it is composed of two sets of vertices: n vertices for the codeword bits called (bit nodes) and m ones for the parity check equations called check codes. Note that bit nodes are called sometimes message or variable nodes and check nodes are termed constrained nodes.

The graph is drawn such that an edge joins a bit node to a check node if and only if that bit is included in the corresponding parity check equation.

The following graph of figure 3.1 Corresponds to the parity check matrix of the (7,4) hamming code.

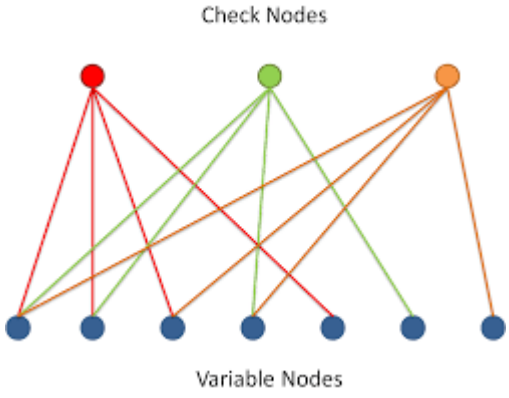


Figure 3.1: Tanner Graph of the (7, 4) hamming code.

Note that for systematic codes, we can have the message-bit and the parity bit separated. From the above definition we can clearly note that the number of edges is equal to the number of ones in the PCM. The following terminology is important in graph theory.

- **The cycle**: a sequence of connected vertices which starts and ends at the same vertex in the graph and which contain other vertices no more than one.
- **The length of the cycle**: the number of edges contained in the cycle.
- **The girth**: the size of the smallest cycle.
- **The degree** number of edges.

4. Construction of LDPC codes

The construction of LDPC codes is equivalent to the creation of a matrix H with only a Few entries are made ones and the rest zeros, depending on the way the matrix is created an LDPC code might be regular or irregular.

4.1 Regular LDPC codes

An LDPC code is said to be regular if the row weight W_r and the Column weight W_C are constant throughout all the rows and columns of the parity check matrix [8]. Another way to define regularity, is that each code bit is contained in a fixed number of parity check equations and each parity check equation contains a fixed number of code bits.

A consequence of definition is that

$$N = m.W_r = n.W_r \tag{3.1}$$

Where N is the total number of ones in H.

An important parameter when talking about codes is the code rate

$$R = \frac{k}{n} = \frac{n-m}{n} = 1 - \frac{m}{n} = 1 - \frac{Wc}{Wr} \tag{3.2}$$

The definition of regular LDPC codes can be also formulated in terms of Tanner graph.

A regular LDPC code is the one in which the number of edges emanating from each check node is the same in all the check nodes, we call this number the degree of the check node (Wc). The same is said about the bit nodes.

4.2 Irregular LDPC codes

An irregular LDPC code is an LDPC code that is not regular. In other words, an LDPC code is said to be irregular, if at least one of the two, the degree of the check node W_r or the degree of the bit node W_C are not constant throughout all the nodes.

The fact that an LDPC code is irregular, makes the equation (4.1) useless since at least one of Wc or Wr is not constant. To go round this problem, a new set of descriptors [34] is introduced, the bit node and check node distributions, defined respectively as:

$$\lambda(x) = \sum_{j=1}^{Wrmax} \lambda j.\, x^{J-1} \tag{3.3}$$

$$P(x) = \sum_{i=1}^{Wcmax} pj.\, x^{i-1} \tag{3.4}$$

- Where $Wrmax$ and $Wcmax$ are the maximum bit nodes and check nodes degrees.
- λj is the fraction of all edges emanating from variable nodes of degree j.
- Pj is the fraction of all edges emanating from check nodes of weight i.

Note that an irregular code is totally specified by the two degree distributions and the dimension of the parity check matrix.
Based on equation (4.3) and equation (4.4),

$$\sum_J \lambda j = Pj = 1 \tag{3.5}$$

The code rate of this type of LDPC codes is then given:

$$R = 1 - \frac{\sum_i {Pi}/{i}}{\sum_j {\lambda j}/{j)}} = 1 - \frac{\int_0^1 p(x).dx}{\int_0^1 \lambda(x).dx} \tag{3.6}$$

Since we have formally defined the two types of codes, let take a look at the methods used for their construction.

4.3 GALAGER's LDPC codes

This type of LDPC codes are defined by a parity check matrix that is banded, the algorithm for Construction is given as follows [8]:

1. Determine the dimension of your parity check matrix
2. Set the desired weights Wc and Wr and make sure that M/Wc is integer
3. Divide it into sub matrices each having M/Wc rows
4. In the first set of rows let $1 \leq i \leq \frac{M}{Wc}$ (the ith row has nonzero entries in the $[(i\text{ -1}).\, Wr\text{ +1}]^{th}$ column up to $i.\, Wr$ th column.
5. The other sub-matrices are created by permuting the columns of the first sub- matrix.

Since this codes are not totally random, they are called pseudo-random. Note that when we talk about LDPC codes, we usually talk about the ensemble since setting Wc and Wr will lead to many possibilities of LDPC codes all having the same degree. In literature We call them the (Wc, Wr) ensemble of LDPC codes.

4.4 MacKay and NEAL regular LDPC codes

MacKay and Neal [11] have proposed a way to construct (almost) regular LDPC codes. Unlike Gallagher's, the MN LDPC codes are constructed columnwise. In this method, H is created by generating weight Wc Columns and making sure to obtain an (almost) uniform row weight.

The algorithm of construction is described as follows:

1. Start adding wc columns from left to right with the non-zero entries in each column are chosen randomly.
2. If at any point some rows are still unfulfilled (Wr is not uniform), it can be corrected in the next columns to be added, if not possible the process can be back tracked so that Wr is as uniform as possible.

The method is usually designed with short cycles (4-cycles) being avoided by check in Each pair of columns in H to make sure they don't overlap in two places. The reason 4-cycles are avoided is because they will degrade the performance of iterative decoding Algorithms.

4.5 Random construction

Up to now, the codes that were generated are considered pseudo-random, "random" because we can't know a priori the LDPC code we are going to get at the end of the construction process and the prefix "pseudo" refers to the fact that it's not purely random.

The construction of random codes [23] involves the following steps:

1. We set the number of variables and check nodes, we call them the "sockets "in the sense that the edges will enter them later.
2. A random interleaver is generated and the connection between the two sockets is made.
3. We check that multiple edges between nodes are non-existent.

4.6 Progressive Edge Growth Algorithm

PEG [24] is a graph construction algorithm which proceeds by placing edges of the graph one by one with the constraint that no cycle is created or that cycles are created with the maximum length possible, and it focuses on maximizing the girth locally without considering the final girth of the graph.

The algorithm works such that, for each bit node b_i, ω_c-check nodes are connected to it, and if the graph is constructed so that x-cycles avoided then the check nodes chosen to join the bit node are the ones with the least degree at a distance $\frac{x}{2}$ or more edges away.

More specifically, the algorithm follows the following steps:

1. Start with a variable node, we call it the "root" and attach to it all the check nodes with edges connecting to it to form the first level of the global tree.
2. In the second level of the tree, we attach all the variable nodes of the first level excluding the root variable node.
3. The algorithm proceeds and if a node already exists in the tree it is excluded from the new level till one of the two outcomes take place,
   - The expansion at a given level is done and no check nodes are available.
   - On the completion of an expansion level, all the check nodes are already contained on the graph.

If the graph ends with the first outcome, then it will be acyclic. However, the second case produces cycles, but they will be cycles of the greatest length possible.

4.7 Structured LDPC codes

4.7.1 Quasi-Cyclic LDPC codes

Quasi-cyclic LDPC codes come with different versions, but the general structure is that H is made of several sub-matrices that are either the zero matrix or right-cyclic shifts of the identity matrix by a given amount [25].

The structured MxN QC-LDPC matrix is constructed as follows:

$$H = \begin{bmatrix} H_{11} & H_{12} & \dots & H_{1N'} \\ H_{21} & H_{22} & \dots & H_{2N'} \\ \vdots & \vdots & \ddots & \vdots \\ H_{M'1} & H_{M'2} & \dots & H_{M'N'} \end{bmatrix} \tag{3.6}$$

Note that if the sub matrices are of size UxV then $M = M'U$ and $N = N'V$

QC-LDPC codes are very useful in practice for many considerations, like reduced storage requirements and low-complexity encoder and decoder implementations.

Note that QC construction method is usually combined with the PEG [36].

4.7.2 Repeat Accumulate LDPC codes

Repeat Accumulate LDPC codes are LDPC codes with a parity check matrix of the form:

$$H = [H_1 \quad H_2] \tag{3.7}$$

With H_2 being simply the dual-diagonal matrix

$$H_2 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 0 & 1 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 1 & 0 \\ 0 & 0 & \dots & 1 & 1 \end{bmatrix} \tag{3.8}$$

H_1 in the other side is constructed column by column using a repetition code that is the output of a sequential encoder and depending on the type of this encoder it may be regular [20] or irregular [21].

5. Encoding

Definition:

Encoding deals with the creation of the codeword set that corresponds to the constructed parity check matrix, two common methods of encoding LDPC codes are:

5.1 Encoding using systematic generator matrix

We have already defined the generator matrix for linear block codes in the previous chapter. Since LDPC are themselves linear block codes, one way to perform encoding is to exploit the relation between the H and the G.

The algorithm of encoding in this method is as follows:

- Construct your sparse $m * n$ parity check matrix H using the construction methods stated in the previous chapter.
- Using elementary row operation defined over GF (2), we bring H into systematic form

$$H = [P^T I_N] \tag{3.9}$$

If row operations are not enough, one can also use column permutations to ensure systematic form. However, the corresponding codewords will be permuted versions of the original ones .

One way to keep track of the permutation, is to apply the inverse permutation before the codewords are transmitted.

- G can be obtained from H as

$$G = [I_K P] \tag{3.10}$$

- Using G and a set of information vectors of size k, the set of valid codewords is as follows:

$$C = \{c \, \epsilon \{0,1\}^{1*n} | \, c = i.G, \;\; i \epsilon I\} \tag{3.11}$$

Notes

For LDPC codes, H is a sparse matrix as we have seen earlier; it would be favorable to obtain a sparse G as well. However, in most cases this is not the case so as n gets bigger, this type of encoding becomes impractical due to the time complexity of the encoding algorithm which it's clearly $O(n^2)$.

When we compare algorithms for performance, we prefer those which have linear time complexity $O(n)$, what we mean by linear time complexity, is that the number of operations performed by the algorithms grows linearly with the code length.

Richardson and Urbanke have created such an algorithm for encoding which we going to investigate now.

5.2 RU encoding algorithm

In their work [21], Richardson and Urbanke have provided a way to encode LDPC codes in (almost) linear time, they went ahead and shown that if an ensemble has a degree distribution for which we can encode in linear time, the performance of such an ensemble will be good under message passing decoding.

The idea of the algorithm is as follows:

To encode in linear time, you need a sparse matrix but G is usually not sparse and have a quadratic complexity. So if one can use the sparse PCM H to encode directly, we can encode in linear time and that what they have done exactly.

The RU encoding method has three steps:

1. Given an $m * n$ sparse matrix, we use only row and column permutations to make our matrix takes the form

$$H = \begin{pmatrix} A & B & T \\ C & D & E \end{pmatrix} \tag{3.12}$$

With the sub matrices being:

A: $m - g * k$ D= $g * g$

B: $m - g * g$ E= $g * m - g$ (3.13)

C: $g * k$ T= $m - g * m - g$

Notes

- The sparsity of H is not affected at this stage since we have just rearranged the entries.

In this form T is a triangular matrix and we say that the matrix H is an approximate lower triangular form.

- g is called the gap of the parity check matrix H

Note that if the gap is zero the encoding is done simply by back substitution.

2. After that form is obtained, we multiply H by the following matrix:

$$\begin{pmatrix} I_{m-g} & O \\ -ET^{-1} & I_g \end{pmatrix} \quad (3.14)$$

This yields,

$$\begin{pmatrix} A & B & T \\ -ET^{-1}A + c & -ET^{-1}B + D & 0 \end{pmatrix} \quad (3.15)$$

Now put the code word in the following systematic form

$$x = (x_S \; x_{p1} \; x_{p2}) \quad (3.16)$$

- x_s: the information vector of length s.
- x_{p1}: parity vector of Length g.
- x_{p2}: parity vector of Length m-g.

Since we multiply by an invertible matrix the code remains equivalent to the original one.

Hence,

$$\begin{pmatrix} I_{m-g} & 0 \\ -ET^{-1} & I_g \end{pmatrix}.H.x^T = 0 \tag{3.17}$$

The resulting linear system of equation is:

$$A_{xs}{}^T + B_{p1}{}^T + T_{xp2}{}^T = 0^T \tag{3.18}$$

$$(-ET^{-1}A + C)x_s{}^T + (-ET^{-1}B + D)x^T{}_{p1} = 0^T \tag{3.19}$$

$$\text{Let} \qquad \Phi = -ET^{-1}B + D \tag{3.20}$$

IF Φ is invertible then

$$x_{p1}{}^T = -\Phi^{-1}(-ET^{-1}A + C)x_s{}^T \tag{3.21}$$

$$Tx_{p2}{}^T = -Ax_s{}^T - Bx_{p1}{}^T \tag{3.22}$$

3. From equation (1) we obtain x_{p1}.
4. Plugging in the second and by back substitution (no need to calculate T^{-1}) x_{p2} is obtained.

We have done the steps under the assumption that Φ invertible, if it is not, we permute columns from the left of parity check matrix into Φ so to make Φ invertible However if this is not possible, it means H is not a full rank matrix. Practically, the possibility of this happening decreases as of the block length increases [21].

We have said that it is a (almost) linear time encoding. To verify this, one has to evaluate the complexity of the computation involved in the process.

To arrive to the linear encoding, we need to be smart in making the calculations for example in (5.13), the following matrix multiplication:

$$-\Phi^{-1}(ET^{-1}A + C) \text{ gives } O\big(g * (n-m)\big) \tag{3.23}$$

However, if we follow the steps as given in the tables [21] to follow, we will end up by a total complexity of $O(n + g^2)$

Table 3.1 steps involved calculating x_{p1}

Operation	Comment	Complexity
$Ax_s{}^T$	Multiplication by sparse matrix	$O(n)$
$T^{-1}[Ax_s{}^T]$	$T^{-1}[Ax_s{}^T]$=y^T=>$[Ax_s{}^T]$=Ty^T	$O(n)$
$-E[T^{-1}Ax_s{}^T]$	Multiplication by sparse matrix.	$O(n)$
$Cx_s{}^T$	Multiplication by sparse matrix.	$O(n)$
$\left[-ET^{-1}Ax_s{}^T\right] + [Cx_s{}^T]$	addition	$O(n)$
$-\varphi^{-1}[-ET^{-1}Ax_s{}^{T^T} + Cx_s{}^T]$	Multiplication by dense $g * g$ matrix.	O(g^2)

Table 3.2 Steps involved in calculating x_{p2}

Operation	Comment	
$Ax_s{}^T$	Multiplication by sparse matrix	$O(n)$
B$[x_{p1}{}^T]$	Multiplication by sparse matrix	$O(n)$
$[Ax_s{}^T] + [\mathrm{B}x_{p1}{}^T]$	Addition	$O(n)$
$-T^{-1}\left[Ax_s{}^T + \mathrm{B}x_{p1}{}^T\right]$	$y^T = -T^{-1}\left[Ax_s{}^T + \mathrm{B}x_{p1}{}^T\right]$ $Ty^T = Ax_s{}^T + \mathrm{B}x_{p1}{}^T]$	$O(n)$

This method of encoding yields a time complexity as long as g is small and does not exceed$\sqrt{n}$.

6. Decoding

Decoding is the inverse operation of encoding, there are many types of decoding from which we mention the following.

6.1ML decoding

Like any other block codes, LDPC codes can be decoded using maximum likelihood decoding [14]. Given a received vector, the decoder will compare this vector to each one of the valid code word and chose the most one as being transmitted.

$$C_t = \min w(C_i + r) \tag{3.24}$$

w refers to the hamming weight, which is defined as the number of non-zero entries in the codeword. In this method, we have an exponential number of codewords to compare with, which means exponential time complexity. This algorithm is not common in practice.

6.2Bit- Flipping decoder

In this type of decoding, if the parity check equations for a given received codeword are not satisfied, one or more bits of the codeword are flipped according to some predetermined order and criteria so that we arrive to a valid codeword (valid but not necessarily correct). In the flipping decoders the main questions to address are:

- How to decide a bit should be flipped.
- In which order bits are considered.

There are many bit flipping algorithms from which we present the following:

6.3 Gallagher's Bit-Flipping decoder

Gallager's Bit Flipping decoder [8] was introduced in 1962, and it is the simplest of all LDPC BF decoders.

The algorithm for the decoder is as follows:

a. Let K be a constant.
b. Count the number of unsatisfied parity check equations in which the first bit is involved ("first" is according to the decoder and not to the codeword). lf the number is greater or equal K flip the bit.
c. We proceed through the other bits until the maximum number of iterations is attained or all the parity check equations are satisfied.

6.4 Weighted Bit-Flipping decoder (WBF)

WBF starts by finding the most reliable message node included in each individual check. Reliability of a message is based in its soft value [28].

The algorithm proceeds as follow:

a. The received codeword r is multiplied by H^T to get the syndrome vector.

$$S = (S_1, S_2, \dots, S_r) \tag{3.25}$$

b. To find the lowest magnitude of all messages node included in the m^{th}check.

$$Y_m{}^{min} = \min y(n) | n\epsilon\, N(m) \tag{3.26}$$

c. Calculate E_n such that

$$E_n = \sum_{m\epsilon M(n)} (2S_m - 1).Y_m{}^{min} \tag{3.27}$$

d. Flip the bit in r that has the highest E_n

Note that E_n is used to decide if the bit at position n in the m^{th} check should be flipped or not. This kind of weighted Bit-Flipping decoders uses only the information available in the check node to make decision. An improved version of the WBF decoder makes use of the message node information [29]. This time, the error term takes the following value

$$E_n = \sum (2S_m - 1).Y^{min}{}_m - \alpha |Y_n| \quad , m \epsilon\, M(n) \tag{3.28}$$

We can write,

$$E_{iWBF} = E_{WBF} - \alpha \,.\, |Y_n| \quad (3.29)$$

This equation means that if $|Y_n|$ is high enough, it demonstrates some confidence that it is might be true.

α>0 is a parameter chosen separately and found experimentally for best error performance.

6.5 Boot-Strapped Weighted Bit-Flipping decoder

The Boot Strapped Weighted Bit-Flipping decoder [30] is just a modification of the WBF by introducing a condition to evaluate nodes if reliable or not.

Let δ be a positive parameter to denote the threshold of reliability. Using this parameter, we have the following constraints:

- The soft values y_i for which y_i< δ are deemed unreliable and the rest are reliable.
- A check node is reliable if all its neighbors are reliable and unreliable otherwise.

After the classification is made, the unreliable nodes are then re-initialized as:

$$y_i \triangleq y_i + \sum_{j\epsilon N(m) j\neq i} \min |y_j| \,.\, \prod_{j\epsilon N(m) j\neq i} Sgn(y_j) \quad (3.30)$$

With $Sgn(x)$ is the $Signum$ function.

Here, we are using the information stored in the other variable during the check m. we will recognize later that this step in nothing but one iteration in the sum-product decoder.

6.6 Gradient-Descent Bit Flipping decoders

The Gradient Descent Bit-Flipping algorithm [31] tries to maximize the correlation between the received word and the decoded vector.

In this type of decoders, we assume the alphabet is $\{-1,1\}$, and we define an objective function (x) :

$$f(x) = \sum_{i=1}^{n} x_i y_i + \sum_{a=1}^{m} \prod_{i \in N_{(a)}} x_i \quad (3.31)$$

The first part of the function is the correlation of the current decoded codeword and the soft channel values y_i; the second term is used to account for the satisfied parity checks.

if all of them are satisfied, it takes the value m and if none of them is satisfied it takes –m.

Our goal is to maximize f or minimize –f, the algorithm then proceeds as follow:

a. Find the partial derivative of f with respect to x_i

$$\frac{\partial}{\partial x_i} f_{(x)} = y_i + \sum_{a \in N_i} \prod_{j \in N_{(a) \setminus i}} x_i \quad (3.32)$$

b. Use first order approximation of f in the x_i-coordinate

$$f_{(x_1, \ldots x_i + s, \ldots x_n)} = f(x) + s \frac{d}{dx_i} f(x) \quad (3.33)$$

- The step 's' in chosen such that

$$s \frac{d}{dx_i} f(x) > 0 \quad (3.34)$$

c. Define

$$E_i = x_i \frac{\partial}{\partial x_i} f(x) = x_i . y_i + \sum_{a \in N(i)} \prod_{j \in N(a)} x_j \quad (3.35)$$

Flip the bit that has smallest E_i .

Note that one can define

$$E_i = -x_i \frac{\partial}{\partial x_i} f_{(x)} \quad (3.36)$$

d. Flip the bit with the greatest E_i.

6.7Message passing decoding

Message passing decoders [2] [32] [33] are similar to Bit-Flipping decoders in the sense that both of them are iterative. However, this time the values sent to each node of the Tanner graph are probabilistic.

In literature, message passing decoders come with many names, sometimes they are called “Belief-propagation” decoders that stems from the fact that messages are probabilities or the “sum-product” decoders to refer to the iterations of the algorithm that are nothing but sums of products of beliefs.

The MP algorithm in best explained in Tanner graph, more specifically, given a Tanner graph with $\boldsymbol{m}$ chock nodes and $\boldsymbol{n}$ variable nodes, we look at those nodes as being processors that produce beliefs or messages that propagate along the edges of graph.

Those beliefs get updated till a valid codeword is obtained or the maximum number of iterations is reached.

Before stating the different steps of the algorithm, we introduce some notations.

Given a Tanner graph with m check nodes and n variable nodes:

$\boldsymbol{N(m)}$: the set of variables nodes in the graph connected to a check node $\boldsymbol{m}$ with

$$n = 1,2, \dots, N$$

$\boldsymbol{M(n)}$**:** the set of check nodes of the graph connected to a variable node **n** with

$$m = 1,2, \dots, M$$

$\boldsymbol{N(m)/n}$**:** the neighbors of check node $\boldsymbol{m}$ excluding $\boldsymbol{n}$**.**

$\boldsymbol{M(n)/m}$**:** the neighbors of variable node $\boldsymbol{n}$ excluding $\boldsymbol{m}$**.**

The probabilities involved in the algorithm are defined as follows:

$$f_{mn}^{x} = p(x_n = x) \text{ With } f_n^0 + f_n^1 = 1 \tag{3.37}$$

e. q_{mn}^{x} = Probability that a variable node $\boldsymbol{n}$ equals $\boldsymbol{x}$ given that messages are sent from all nodes except for $\boldsymbol{m}$**.**

f. r_{mn}^{x}= Probability that the $\boldsymbol{m}^{th}$ parity check equation is satisfied with the $\boldsymbol{n}^{th}$ variable node being $\boldsymbol{x}$.

Note that q_{mn}^{x} and r_{mn}^{x} are called extrinsic information since they exclude the node to which information is sent. The algorithm works by updating them at each iteration till a valid code word is obtained or the iteration process ends.

Usually, the algorithm is implemented in the log-domain; however; we are going to present first the probability-domain version for the sake of completeness.

6.8 The probability-domain MP algorithm

a. Initialization

$$\forall m \in \{1,2 \ldots . M\} \text{ and } \forall n \in \{1,2 \ldots . N\} \quad (3.38)$$
$$(q_{mn}^0 = f_n^0 \ \Lambda \ q_{mn}^1 = f_n^1)$$

b. Iterative step

- Let $\delta q_{mn} = q_{mn}^0 - q_{mn}^1$, exclude nodes that have no edges connecting them
- Let $$\delta r_{mn} = \prod_{n' \in N_{\frac{m}{n}}} \delta r_{mn'} \quad (3.39)$$
- Update check-to-bit

$$r_{mn}^x = \frac{1}{2}(1 + (-1)\, \delta q_{mn} . \delta r_{mn}) \quad (3.40)$$

- Update bit-to-check

$$q_{mn}^x = \alpha_{mn}\, f_n \prod_{m' \in N_{(m)/n}} \delta r_{m'n} \quad (3.41)$$

- Update the a posteriori probability q_n^x using

$$q_n^x = \alpha_n\, f_n^x \prod_{m' \in N_{\frac{m}{n}}} \delta r_{m'n} \quad (3.42)$$

- Note that α_{mn} and α_n are constraint factors to make sure that

$$q_{mn}^0 + q_{mn}^1 = 1 \quad \text{and} \quad q_n^0 + q_n^1 = 1 \quad (3.43)$$

c. Decision

$$\begin{cases} \hat{x}_n = 1 & \text{if} \quad q_n^1 > 0.5 \\ \hat{x}_n = 0 & \text{if} \quad q_n^0 > 0.5 \end{cases} \quad (3.44)$$

- Compute $H^T \hat{x}$
- If $H^T . \hat{x} = 0$ stop and forward $\hat{x}$ as the valid codeword. Else, if the maximum number of iterations is reached.

- stop and state that the maximum of possible iterations has been reached and forward $\hat{x}$ as the code word Else return to 1 .

6.9 The log-domain SPA

Usually the SPA is used in the log domain which works by replacing the probabilities with log-likelihood ratios (LLR) The LLR is defined as follows:

$$L(x) = \log \frac{P_r(x=0)}{p_r(x=1)} \tag{3.45}$$

The log here is assumed to be natural, if it is not the case, a scaling parameter is introduced.

To keep track of the previous notation let:

$$L(n) = \log \frac{f_n^0}{f_n^1} \; ; \; \lambda_{mn} = \log \frac{q_{mn}^0}{q_{mn}^1} \; ; \; \lambda_n = \log \frac{q_n^0}{q_n^1} \; ; \; \Lambda_{mn} = \log \frac{r_{mn}^0}{r_{mn}^1} \tag{3.46}$$

Using the expressions of r_{mn}^0 and r_{mn}^1 , we can write :

$$\Lambda_{mn} = \log \frac{1 + \prod_{n' \in N_{(m)/n}} (q_{mn'}^0 - q_{mn'}^1)}{1 - \prod_{n' \in N_{(m)/n}} (q_{mn'}^0 - q_{mn'}^1)} \tag{3.47}$$

$$\Lambda_{mn} = 2 \tanh^{-1} \left(\prod_{n' \in N_{(m)/n}} (q_{mn'}^0 - q_{mn'}^1) \right) \tag{3.48}$$

This may be further simplified to give:

$$\Lambda_{mn} = 2 \tanh^{-1} \left(\prod_{n' \in N_{(m)/n}} \tanh \frac{\lambda_{mn'}}{2} \right) \tag{3.49}$$

We have also

$$\lambda_{mn} = \log \frac{q_{mn}^0}{q_{mn}^1} \tag{3.50}$$

$$\lambda_{mn} = \log \frac{f_n^0}{f_n^1} + \sum_{m' \in N_{(n)/m}} \log \frac{r_{m'n}^0}{r_{m'n}^1} \tag{3.51}$$

Hence,

$$\lambda_{mn} = L_n + \sum_{m \in N_{(n)}} \Lambda_{mn} \tag{3.52}$$

Using these parameters, the algorithm is defined as follows:

a. Initialization

- For all variable nodes $\boldsymbol{n}$ and check nodes $\boldsymbol{m}$ find:

$$\lambda_{mn}=L_n \tag{3.53}$$

L_n Depends on the channel being used for transmission

b. Iteration –step

- check-to-bit update

$$\Lambda_{mn}= 2\tan \mathrm{h}^{-1}\left(\prod_{\mathrm{m}\in N_{(n)/m}} \Lambda_{m\prime n}\right) \tag{3.54}$$

- bit-to-check update

$$\lambda_{mn}=L_n+\sum_{\mathrm{m}\prime\in N_{(n)/m}} \Lambda_{m\prime n} \tag{3.55}$$

$$\lambda_{mn}=L_n+\sum_{\mathrm{m}\in N_{(n)}} \Lambda_{mn} \tag{3.56}$$

c. Decision

$$\begin{cases}\hat{x}_n = 1 & \text{if} \quad \lambda_n > 0\\ \hat{x}_n = 0 & \text{if} \quad \lambda_n < 0\end{cases} \tag{3.57}$$

If $H^T.\hat{x}$ =0 or the maximum number of iterations or the maximum number of iterations is reached, the program stops. Otherwise it returns to 2.

Notes

In the fourth chapter, we have said that 4-cycles are not desirable and codes are usually optimized to have as minimum 4-cycles as possible because they exhaust the iteration process.

Summary

LDPC codes are linear block codes with a sparse parity check matrix, and depending on the way the row and column weights are distributed, an LDPC code might be regular or irregular.

Aside from the matrix representation, LDPC codes are usually described using Tanner graphs. The reason behind this is to be able to employ data structures algorithms to perform iterations that are based on graph theory.

LDPC codes are constructed using many methods some of which are matrix based like Gallager and Mackay-Neal methods or graph based like the progressive edge growth algorithm. After the code is constructed, the next step is to design the encoder which may be done

using many methods. One of them is by using the G-matrix. however, the G matrix is not necessarily sparse which leads to complex encoding procedure. Richardson and Urbanke have developed a linear time encoding method that uses the H matrix directly without computing the generator matrix. In the next chapter we will look at the opposite i.e. decoding LDPC codes.

Decoding LDPC codes is performed using Bit-flipping and Message passing decoders, with the last being preferred, since it is based on the belief or probability of each node.

Despite their optimality from the error-correction's point of view, ML decoders are not used in practice due to their exponential complexity, so the choice of the decoders is a tradeoff between error-performance and time-complexity.

The next chapter is a simulation of some LDPC codes under different constraints.

1. Introduction

In this chapter, a set of different LDPC codes in terms of length and construction, were chosen to simulate their performance through the binary AGWN channel with BPSK modulation under many constraints. An important issue to point out to, is that very long codes are excluded due to their considerably long execution time in the available machines.

The code was written in MATLAB 2011b and 2014b and two computers with intel i3 and i5 processors respectively, were used to run the simulation.

2. Procedure of Simulation

The parity check matrices are constructed and used to encode the message sequences to form a set of codewords to be transmitted. The coded bits are then, modulated using BPSK modulation and sent through a binary Additive Gaussian White Noise (AWGN) channel.

A BPSK modulator receives the noised vector, demodulates it, and passes it to the decoder that makes decision about the correct codeword and produces the message.

The average BER is calculated for many values of SNR and used as a criterion for evaluation.

The following table contains the LDPC codes used throughout our simulation.

Table 4.1: LDPC codes used in the simulation

H	H1	H2	H3	H4	H5	H6	H7
m	128	128	128	128	512	2048	3072
n	256	256	256	256	1024	4096	4096
Rate	½	½	½	½	½	½	¾
Rank	128	128	128	128	512	2048	3072
Method of construction	PEG	Gallagher's Method	Mackay Neal	QC-PEG	PEG	PEG	PEG
Girth	16	4	6	16	16	16	16
Encoding Procedure	G-encoding	G-encoding	G-encoding	G-encoding	G-encoding	G-encoding	G-encoding

The simulation was divided into many stages where different aspects of the codes were investigated and BER versus SNR graphs were sketched.

For the sake of clarity, before going through the simulation, some definitions are reviewed as follows:

The girth of the graph: The length of the largest cycle.

The Bit-Error-Rate (BER): The number of bits in error divided by the total number of transmitted bits.

3. The Simulation results

3.1 Performance with respect to the code length:

In this step, the goal was to evaluate the performance of the code with respect to the code length; so, we had set the following criteria:

- All the codes are of the same rate, which was set to be ½
- The construction method is PEG for all the codes
- The received codewords are decoded using the sum-product decoder

From table, H1, H5, H6 were chosen for this step

The data used for encoding and decoding is shown in the following table:

Table 4.2: the decoding data

Number of sent messages	1000
The '0' probabilities for the SP decoder	$\begin{pmatrix}0.8\\0.2\end{pmatrix}$
The '1' probabilities for the SP decoder	$\begin{pmatrix}0.2\\0.8\end{pmatrix}$
Maximum number of iteration	10

The choice of the number of messages was made 1000 to obtain a more practical average BER values. However, the iterations were set to be 10 because of the processing powers of our machines.

The probabilities are chosen according to real scenarios. The simulation was done by generating the messages randomly encoded, BPSK-modulated, transmitted and then demodulated and decoded. The BER was calculated for each sample from the information set for a given SNR value, and then, summed up and averaged. The results for this step are shown in the following figure 7.1:

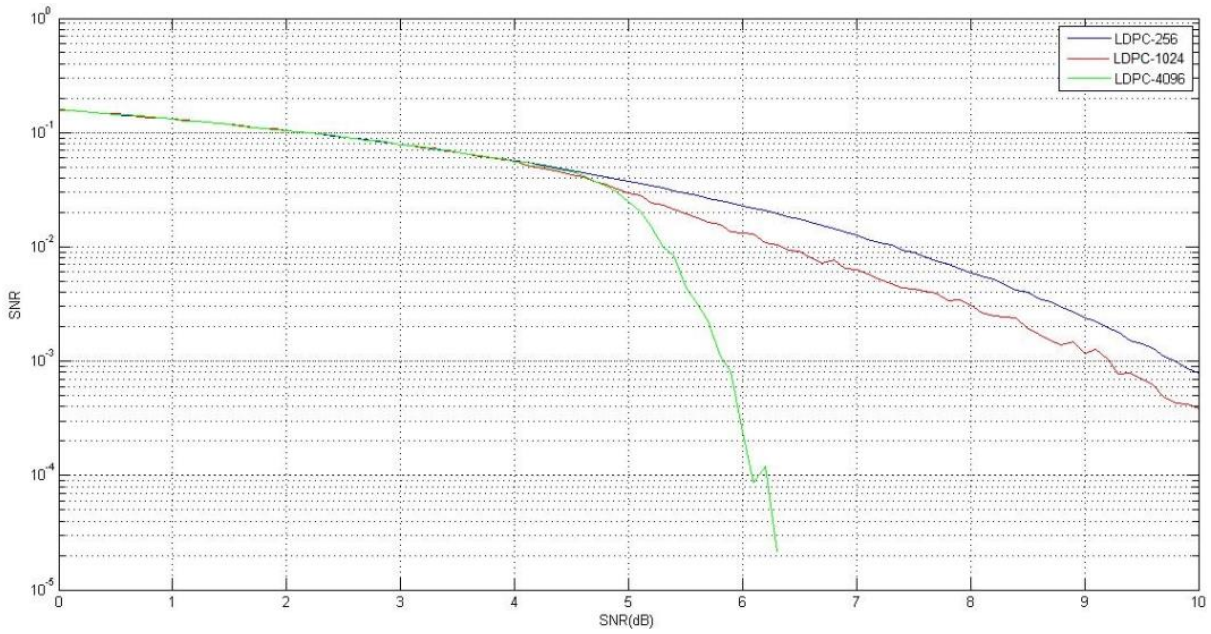


Figure4.1: BER *Vs* SNR for different code lengths

Discussion:

The three codes behave similarly at lower SNR values, their BER performance is very close for example at SNR=0 BER is 0.1589, 0.1587, and 0.1578 for the 256-,1024- and 4098- length codes r**e**spectively.

As SNR goes higher we notice that the longer the code the better error performance. To achieve a BER< 10^-3, extra values of at least 4 dB for the 256-length LDPC code and 3.4 for the 1024 code are required.

The same remark can be said about Shannon limit (0.18 for rate ½ BPSK) , At BER 10 3 The 4098 code is within 5.7 dB from Shannon's limit, compared to 9.2 and 9.7 dB for the 1024- and 256-length code. Practically, we are still far from Shannon's limit performance that LDPC codes are said to achieve ,this is because of the small number of iterations and the fact that the codes used here are at most of medium length compared to codes used in Standards. However, from the graph a clear tendency towards Shannon's limit as the code length gets higher is clearly shown.

3.2 Performance evaluation with respect to the construction method

In this step we tried to investigate the impact of the construction method on the error performance of the code. The LDPC code chosen for this simulation is the ½-rate 256-length LDPC code. The code is constructed 4 times using:

- Gallagher's Method
- Mackay-Neal method
- Progressive edge growth algorithm
- Q-C combined with the progressive edge growth algorithm

The result of this construction is H1,H2,H3 and H4 shown in table7.1 After the construction, H1 and H2 had some of their bits flipped (10 for H1 and 13 for H2) the reason was to ensure full rank property. The encoding and decoding data was the same as the ones in the previous experiment.

The result are shown in figure 7.2:

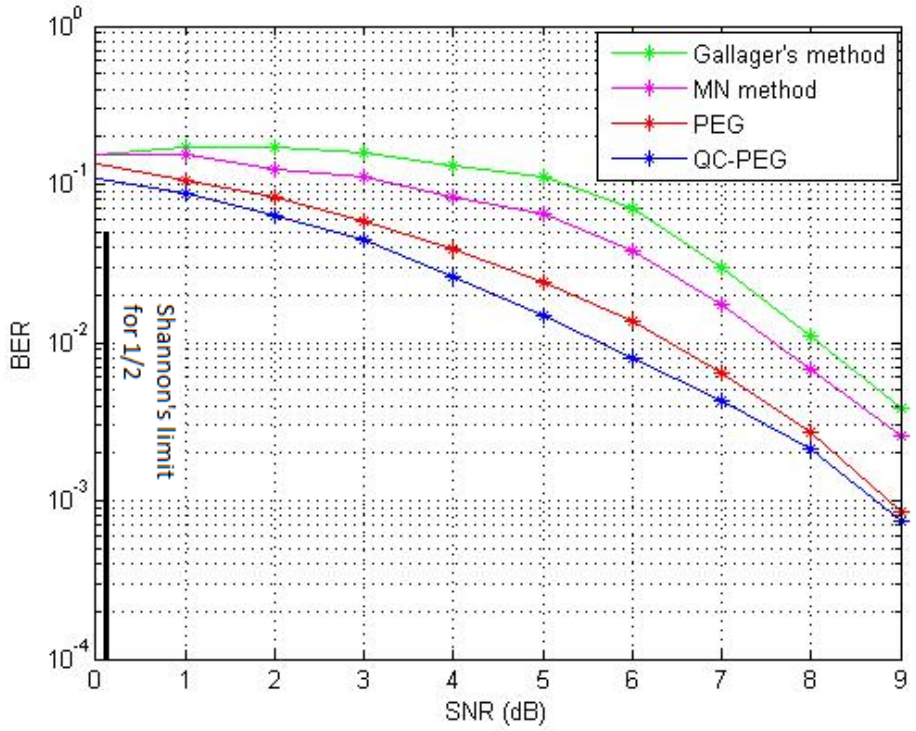


Figure4.2: the performance of the rate ½ 256-LDPC code under many construction methods

Discussion :

At lower SNR values (~0) the 4 methods are close in terms of performance. However at medium SNR a value of 0.5 Db to 1 DB is maintained between each pair of Gallager-MN and PEG-QCPEG with about a range of 1to 2 Db between the two pairs.

The performance is then can be said to be in favor of the structured LDPC codes. This can be a result of the more constraints these algorithms place on the construction of the code like the largest girth possible for the PEG and the avoidance of 4-cycles in the MN method.

3.3 performance with respect to the Decoder type

Up to now , we have been using the SP decoder as the standard decoder. In this part of the simulation we try to investigate other type of decoders and compare their performance with the SP decoder.

The code chosen for this simulation is the ½ 4098-length LDPC code whose PCM is H6 Because the code is so long (for our processing machines), we are going to limit the simulation to the following set of decoders:

- SP decoder
- Gallagher's BF decoder
- Gradient Descent Decoder

The other data kept unchanged.

The following graph shows the result of this step in figure 7.3:

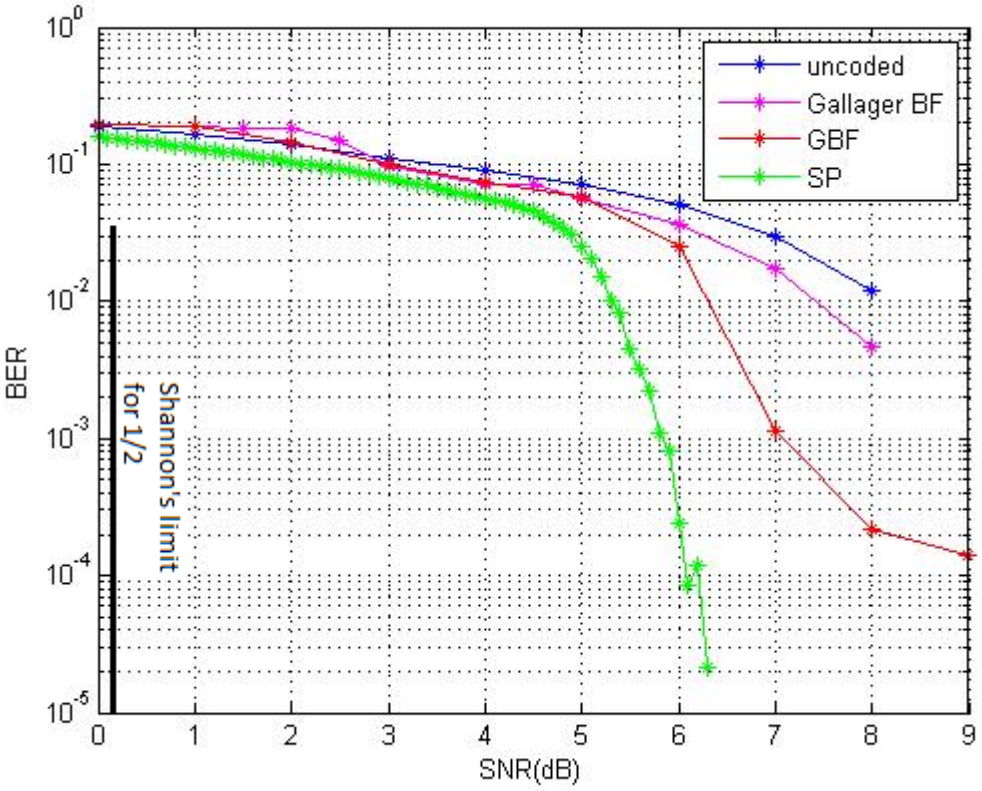


Figure 4.3: performance with respect to decoder type

Discussion:

The graph shows that the decoders are ordered according to their error performance as follows

1. The SP decoder
2. GBF decoder
3. Gallagher's BF decoder
4. The uncoded case

Gallagher's bit flipping decoder behaves better than the uncoded with some points recorded to be even worse, however much less than the GDBF and SP decoders. Some points at lower SNR values are even worse than the uncoded for Gallagher's BF decoder however as SNR goes higher it surpasses it and provides a slightly better performance.

The GDBF decoder behaves better than Gallagher's BF decoder but not as the SP decoder.

To illustrate this consider the case of BER= 10^-2 The Gallaher's BF is better by about 0.5 dB and worse by about 1 and 2 dBs than the GDBF and the SP respectively.

Another important point from the graph is the rate of change of the SNR-BER graph , A decoder can be evaluated also according to this feature. In other words the more steeper is the rate of change of the decoder the better the performance with respect to SNR.

3.4 Performance with respect to the rate

In this part, we focus on the effect of the rate on The performance of the code

For better performance we have chosen the longest LDPC codes we have constructed: the ½ 4098-length and the ¾ 4098-length code. The decoder used is the SP decoder and the other data is kept unchanged.

The results are shown in the following figure 7.4:

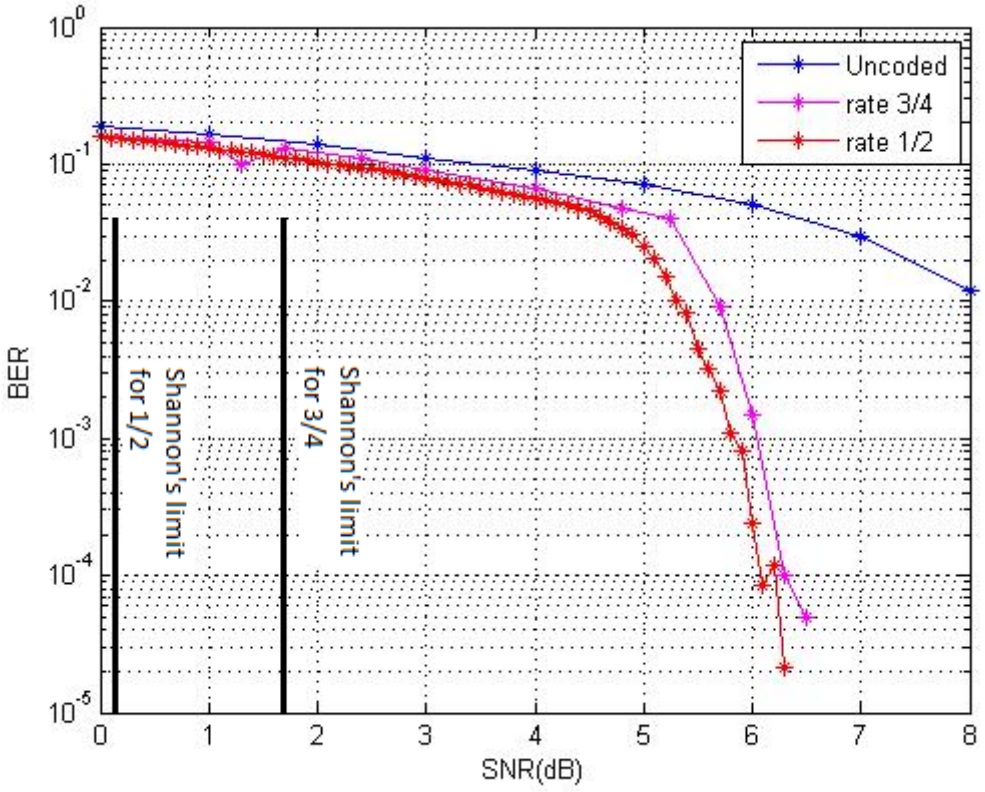


Figure 4.4: performance with respect to the rate

At lower SNR (around 0) values the coded and uncoded behaves similarly, this can be a result from the fact that coding beyond Shannon's limit is meaningless.

At higher SNR values the LDPC codes with rates ½ and ¾ behave much better than the uncoded BPSK signal with the first code being the more performant.

For example at a BER of 10^-2 , an extra 2.8 and 0.5 dB are required for the uncoded BPSK and the ¾ rate LDPC code to achieve the same performance as the rate ½ code.

This results shows that the smaller the rate the more performant the code which verifies Shannon's noisy coding theorem.

4. Summary:

The performance of LDPC codes is governed by many factors. In this chapter the effects of length, construction method, decoder type and code rate were investigated.

The results obtained for the code length and rate can be seen as a verification of Shannon's noisy coding theorem. In the construction method, we have seen that the structured and graph-based methods are more performant since they avoid 4-cycles and are usually designed with a large girth.

In the decoder experiment, the probabilistic decoders were the best compared to the BF decoders because the nature of the algorithm hat does not lose belief about the bits. In the other side, we notice that the more careful we are about flipping a bit the better the performance of the BF decoder.

Conclusion

A linear block code with a sparse parity check matrix is called an LDPC code. This latter was discovered by Robert Gallager in 1962 [8] and rediscovered by Mackay Neal [11] in 1997 that showed that those codes are actually very performant at SNR levels very close to Shannon's limit.

Based on the distribution of the 1s in the parity check matrix, an LDPC code is either regular or irregular, and it can be further sub-categorized based on the construction method used to create the parity check matrix. In the simulation chapter we have shown that the construction method affects the performance of the code, and a value of more than 3 dB difference was noticed at some BER levels between the classical Gallager's method and the QC-PEG, which shows that graph and structured codes are more performant.

Most of the LDPC codes used in practice are very long, in the order of 10000 and more, and therefore the G-encoding and the ML decoding are not very attractive due to their high time complexity. For the encoder, Richardson and Urbanke have created an encoding algorithm that uses the H matrix directly without computing the G, to encode the messages which produce an almost linear complexity. The decoder in the other side, is either a BF decoder that guesses on the valid codeword by flipping the bits according to a set of criteria, or a message passing decoder that uses the belief or probabilities to decode the received vector correctly. Both types of decoders are based on iterations. However, with the same number of iterations, we have found that Belief Propagation methods are more performant.

The results obtained in this work, are realized using short and medium size LDPC codes with the BPSK modulation scheme and the AGWN channel. We suggest to the reader that is interested in this work to expand this results to longer codes while varying the modulation techniques, from which we recommend FSK and OFDM, and the transmission channels to cover the fading channels Rayleigh and Racian.

Another point is about the error, the error considered in our simulation was assumed to be independent with the other errors that may have occurred, so we recommend the consideration of errors produced by shot noise in the evaluation of LDPC codes.

1. Overview on Algebraic Structures

The first concept that we will be using extensively (without even paying attention to, sometimes) is that of Galois fields. However, we need to define first, groups and rings so that a concise and structured definition of the field is obtained.

1.1 Groups:

A group, denoted as $(G,*)$, is a set G closed under a binary operation * such that the Following axioms are satisfied[13]:

a. Associativity:

$$\forall\ a,b,c\ \epsilon\ G\ ,\ (a*b)*c=a*(b*c) \tag{A.1}$$

b. Identity element:

Given $x\ \epsilon\ G$, there is a unique element $e\epsilon\ G$ such that

$$e*x=x*e=x \tag{A.2}$$

c. Inverse element:

Given $x\epsilon G$, There is a unique element $a\epsilon\ G$ such that

$$a*x=x*a=e \tag{A.3}$$

Remark:

- If a group $(G,*)$ is commutative, i.e. $\forall\ a,b\ \epsilon G\ , a*b=b*a$,we call it an abelian group.

1.2 Rings

a. Definition

A ring $(R,+,.)$ is a set R with two binary operations, addition and multiplication, with The following axioms satisfied [13]:

- (R, +) is an abelian group.
- Multiplication is associative.

- Multiplication distributes over addition.

A ring is said to be a commutative ring when the multiplication operation is commutative

b. Characteristic of a ring

Let R be a ring, for all $a \epsilon R$, the smallest positive integer for which $na = 0$ is called the characteristic of R and denoted as $Char\ (R)$ [14] .If no such integer exists we say R has Characteristic of 0.

c. Ideals

Given a Ring R, A subset I of R is called an ideal [14] if:

- $\forall\ a, b \epsilon I \rightarrow$ a − b ϵI (A.4)
- $(\forall\ a \epsilon R \wedge \forall\ i \epsilon I)$, ai $\epsilon I\ and$ ia ϵI (A.5)

Now that Groups and rings are defined, we proceed to the most important algebraic structure to us, which is the concept of fields.

1.3 Fields

A Field is a commutative ring whose nonzero elements constitute a group under multiplication [15].

a. Extension Fields

Let F and G be fields under the same operations with $F \subseteq G$, we say that F is a Subfield of G and G is an extension of F. G can Also be viewed as a vector space over F and If $dim(G)$ is finite we say that G is a finite extension of F [15].

b. Finite Fields:

A finite field F is a field with $char\ (F) = p, p \neq 0$ being a prime integer ,that is a finite extension of Zp (the ring of integers modulo p) [15]. Those fields are called also prime fields or Galois fields and are denoted $GF(p^n)$ with n refers to the extension.

Remarks:

- The number of elements of a field is called the order of that field, if F is finite then

$$|F| = p^n \quad \text{(A.6)}$$

- For a given $GF(p)$ and for all $a \epsilon G$ the powers of a are also elements of the field since $GF(p)$ is closed under multiplication.
- The smallest positive integer n such that a $a^n = 1$ is called the order of a.
- The type of finite fields that is of interest to us, are the extensions of the binary field $(p = 2)$.

c. Binary Fields

The field $GF(2)$ whose elements are in the set $\{0,1\}$, is called the binary field. and it is the field that we need in our work , the addition and multiplication tables corresponding to this field, are shown in tables .

Table A.1: Binary addition

+	0	1
0	0	1
1	1	0

TableA.2: Binarymultiplication

*	0	1
0	0	0
1	0	1

d. Polynomials over the binary field

A Polynomial$f(x)$ over $GF\ (2)$ is of the form

$$f(x) = F_0 + F_1 x + F_2 x^2 + \cdots + F_n x^n \quad \text{(A.7)}$$

Where $f_i\ \epsilon \{0,1\}$.

The addition and multiplication of polynomials are done modulo 2.

2. Overview on Graph Theory

Graph theory is used extensively in LDPC coding in both processes, encoding and decoding and it is the basis for all iterative decoders.

2.1 Graphs:

A graph G is a mapping from the set V into the set $E \subseteq V^2$, i.e. $G = (v, E)$[16].

The elements of V are the vertices (or nodes) of the graph and the elements of E are its edges.

A graph is usually pictured by drawing a dot for each vertex and a line for each edge.

An example of a graph is in the figure 2.1.

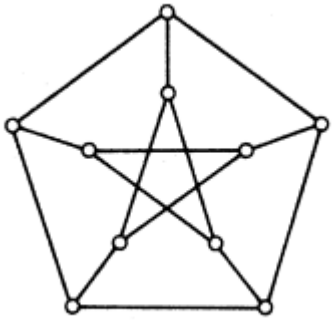


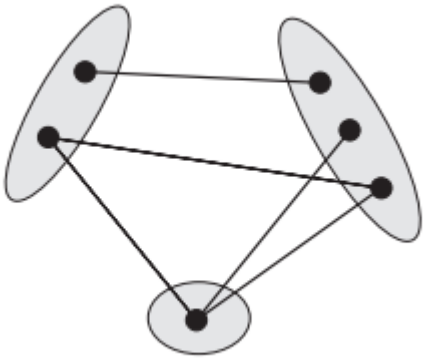
Figure A.1: a graph with 10 vertices and 15 edges

Remark:

The number of vertices in a graph is its order.

2.2 R-partite graphs:

Let $r \geq 2$ be an integer, a graph $G = (V, E)$ is said to be r-partite if V is partitioned into r Classes so that every edge has its ends in different classes [16]. The case important to us is the 2-partite graph that is usually called bipartite graph. An example of this type of graphs in the figure 2.2.



FigureA.2 A 3-partite graph

Remark :

In an r-partite graph an edge cannot has both ends in one class.

2.3 Trees

A tree is a connected graph that contains no cycles [16], the following statements are satisfied for the tree:

- There is a unique path that connects every two vertices of the graph.
- If the tree has n vertices then it must have n-1 edges.
- The tree is called sometimes an acyclic graph.

Figure 2.3 shows an example of a tree.

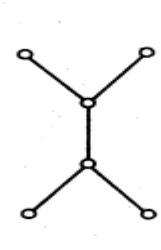


Figure A.3: A tree with 6 vertices and 5 vertices

References

[1] C.E. Shannon, "A Mathematical Theory of Communication," Bell System Technical, journal, July 1948.

[2] Alain Glavieux. "Channel Coding in communication networks, from theory to turbo codes", ISTE Ltd, pp. 19-62, 2007.

[3] Rodger R E. Ziemer, William H. Tranter, " Principles of communications: systems, modulation and noise", John Wiley & Sons Inc, 7th edition, pp. 466-467, 2015.

[4] Rodger R E. Ziemer ,Rojer L. Peterson, "introduction to digital communication" prentice hall Inc, 2nd Edition, pp. 241, 2001.

[5] Bhargavi K, KE Balachandrudu, Nageswar P.,"Mobile Phone Radiation Effects on Human Health", International Journal of Computational Engineering Research,Vol.03, Issue 4, pp. 196-203, April 2013.

[6] Hamming R.W., "Error detection and correction code", bellsys Tech, bell telephone laboratories, J.29:147-60, Murray Hill, 1950.

[7] R.C. Bose and D.K. Ray-Chaudhuri, " On A Class of Error Correcting Binary Group Codes", at Information and Control 3, pp. 68-79, 1960.

[8] Gallagher, R.G., "Low-density parity check codes," Information theory, IRE transaction on, Vol.8, No.1, pp.21-28, 1962.

[9] P. Elias, "Coding for noisy channels," IRE Convention Record, Part 4, pp. 37–46, 1955.

[10] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limimt error correcting coding and decoding: Turbo-codes," in Proc. IEEE Int. Conf. Commun. (ICC), (Geneva, Switzerland), pp. 1064 – 1070 vol.2, May 1993.

[11] Mackay, D. G. C., Neal. R. M.," Near Shannon limit performance of low density parity check codes," Electronics Letter, Vol.33, No.6, pp.457-458, Mar. 1997.

[12] IEEE 802.11n, "Structured LDPC Codes as an Advanced Coding Scheme for 802.11n", IEEE Doc. 802.11-04/884r0, August 2004.

[13] David S.Dummit, Richard M.Foote., " Abstract Algebra", Jhon Wiley & Sons Inc., p16-223, 2004.

[14] Steven Roman, "Coding and information theory", Springer Verlag New York Inc, pp119-427. 1992

[15]Pierre Antoine Grillet, "Abstract Algebra", Springer Science, pp155-192, 2nd Edition 2007.

[16]Reinhard Diestel, " Graph Theory", Springer Verlag New York Inc., pp 2-15, 2000.

[17] A. Bruce Carlson Paul B. Crilly, "Communication systems: An introduction to Signals and Noise in Electrical Communication", McGraw-Hill, pp.3-424, 5th Edition 2010.

[18] Ray Horak, "Webster's New World Telecom Dictionary", Wiley Publishing, Inc., p94, 2008.

[19]J. H. van Lint, " Introduction to Coding Theory", Springer-Verlag New York pp.83, 1999.

[20] T. Lestable and E. Zimmermann, "LDPC Options for Next Generation Wireless Systems" ,Wireless world Research forum

[21] Richardson, T. J. and Urbanke, R. L, "Ecient encoding of low-density parity-check codes", IEEE Transactions on Information Theory, pp.638–656, 2001.

[22] Tanner, R. M.," A recursive approach to low complexity codes," Information theory, IEEE transaction on, Vol.27, No.5, pp.533-547, Sep.1981.

[23] MacKay, D. J C, "Good error-correcting codes based on very sparse matrices», Information Theory, IEEE Transactions on, Vol.45, No.2, pp.399-431, Mar. 1999.

[24] Hu, X. Y., Eleftheriou, E., Arnold, D. M., "Regular and irregular progressive edge growth tanner graphs", Information Theory, IEEE Transactions on, Vol.51, No.1, pp.386-398, Jan. 2005

[25] Fossorier, M.P.C., "Quasicyclic low-density parity-check codes from circulant permutation matrices", Information Theory, IEEE Transactions on , Vol.50, Vo.8, pp.1788-1793, Aug. 2004.

[26] Divsalar, D., Jin, H., McEliece, R. J., "Coding theorems for turbo-like codes", Proc. 36th Allerton Conf. on Communication, Control, and Computing, 1998.

[27] Jin, H., Khandekar, A., McEliece, R. J., "Irregular repeat accumulate codes", Proc. 2nd Int. Symp. Turbo Codes and Related Topics, pp.1-8, 2000.

[28] Kou, Y., Lin, S., and Fossorier, M, "Low-density parity-check codes based on finite geometries", a rediscovery and new results. IEEE Transactions on Information Theory, pp.2711–2736, 2001.

[29] Chinna Nabu.J, S.Parathyusha, V.Usha Sree, " Hard Decision And Soft Decision Decoding Algorithm Of LDPC And Comparison Of LDPC With Turbo Codes, RS Codes And BCH Codes", Proceedings of 09th IRF International Conference Bengaluru, India, ISBN. PP. 978-93-84-209-40-7, 27th July-2014.

[30] Nouh, A. and Banihashemi, A. " Bootstrap Decoding Of Low-Density Parity Check Codes", IEEE Communications Letters, pp.391–393, 2002.

[31] Wadayama, T., Nakamura, K., Yagita, M., Funahashi, Y., Usami, S., and Takumi, "Gradient descent bit -ipping algorithms for decoding LDPC codes". arXiv:0711.0261 [cs, math]. arXiv: 0711.0261, 2007.

[32] Wiberg, N., Leoliger, H.-A, Kotter, R., "Codes and iterative decoding on general graphs'', Information theory proceeding (ISIT), 1995 IEEE International symposium on, Sep. 1995

[33] McEliece, R. J., MacKay, D. J. C., and Cheng.J, "Turbo decoding as an instance of pearl's "belief propagation" algorithm. IEEE Journal on Selected, Areas in communications, Vol.16, No.2, Feb1998.

[34] Luby, M. G., Mitzenmacher, M., Shokrollahi, M. A., Spielman, D. A., Stemann,

V., " Practical loss-resilient codes", Proc. 29th Annual ACM Symp. Theory of Computing, pp.150159, 1997

[35] T. K. Moon, "Error Correction Coding: Mathematical Methods and Algorithms,"

Wiley-Blackwell, July 1, 2005

[36] Liqun Huang Yuliang Wang, Ping Gong, "An improved Construction Method Of QC-LDPC Codes Based On The PEG Algorithm", Third Pacific-Asia on Circuits and Communications Systems (PACCS), july 2011