Pure and Applied Mathematics Journal
2016; 5(6): 220-231
http://www.sciencepublishinggroup.com/j/pamj
doi: 10.11648/j.pamj.20160506.17
ISSN: 2326-9790 (Print); ISSN: 2326-9812 (Online)

Error Detection and Correction Using Hamming and Cyclic Codes in a Communication Channel

Irene Ndanu John[1, *], Peter Waweru Kamaku[2], Dishon Kahuthu Macharia[1], Nicholas Muthama Mutua[1]

[1]Mathematics and Informatics Department, Taita Taveta University, Voi, Kenya

[2]Pure and Applied Mathematics Department, Jomo Kenyatta University of Agriculture and Technology, JKUAT, Nairobi, Kenya

Email address:

irenejohn02@gmail.com (I. N. John), wkamaku@jkuat.ac.ke (P. W. Kamaku), mach@ttu.ac.ke (D. K. Macharia), nicholasmuthama@ttu.ac.ke (N. M. Mutua)

[*]Corresponding author

To cite this article:

Irene Ndanu John, Peter Waweru Kamaku, Dishon Kahuthu Macharia, Nicholas Muthama Mutua. Error Detection and Correction Using Hamming and Cyclic Codes in a Communication Channel. *Pure and Applied Mathematics Journal.* Vol. 5, No. 6, 2016, pp. 220-231. doi: 10.11648/j.pamj.20160506.17

Received: December 1, 2016; **Accepted**: December 28, 2016; **Published**: January 20, 2017

Abstract: This paper provides an overview of two types of linear block codes: Hamming and cyclic codes. We have generated, encoded and decoded these codes as well as schemes and/or algorithms of error-detecting and error-correcting of these codes. We have managed to detect and correct errors in a communication channel using error detection and correction schemes of hamming and cyclic codes.

Keywords: Linear Blocks, Hamming, Cyclic, Error-Detecting, Error-Correcting

1. Introduction

Coding theory is concerned with the transmission of data across noisy channels and the recovery of corrupted messages, Altaian [1]. It has found widespread applications in electrical engineering, digital communication, Mathematics and computer science. While the problems in coding theory often arise from engineering applications, it is fascinating to note the crucial role played by mathematics in the development of the field.

The importance of algebra in coding theory is a commonly acknowledged fact, with many deep mathematical results being used in elegant ways in the advancement of coding theory; therefore coding theory appeals not just to engineers and computer scientists, but also to mathematicians and hence, coding theory is sometimes called algebraic coding theory, Doran [3].

An algebraic techniques involving finite fields, group theory, polynomial algebra as well as linear algebra deal with the design of error-correcting codes for the reliable transmission of information across noisy channels.

Usually, coding is divided into two parts:

a Source coding:
- Source encoding
- Source decoding

b Channel coding:
- Channel encoding
- Channel decoding

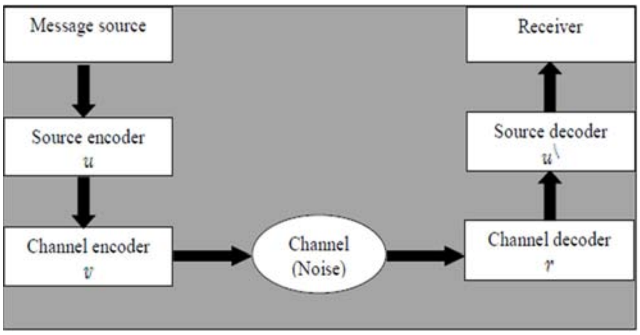

Figure 1. *Model of a Data Transmission System.*

Source encoding involves changing the message source to a suitable code say u to be transmitted through the channel. Channel encoding deals with the source encoded message u , by introducing some extra data bits that will be used in detecting and/or even correcting the transmitted message, Hall [4]. Thus the result of the source encoding is a code word, say v . Likewise, channel decoding and source decoding are applied on the destination side to decode the received code word r as correctly as possible.

Figure 1 represents a model of a data transmission system.

For example: Consider a message source of four fruit words to be transmitted: apple, banana, cherry and grape. The source encoder encodes these words into the following binary data $(u_1u_2u_3u_4)$:

Apple → $u_1 = (0, 0)$, banana → $u_2 = (0, 1)$,

Cherry → $u_3 = (1, 0)$, Grape → $u_4 = (1, 1)$.

Suppose the message ‘apple’ is to be transmitted over a noisy channel. The bits $u_1 = (0,0)$ will be transmitted instead. Suppose an error of one bit occurred during the transmission and the code (0, 1) is received instead as seen in the following figure. The receiver may not realize that the message was corrupted and the received message will be decoded into ‘banana’.

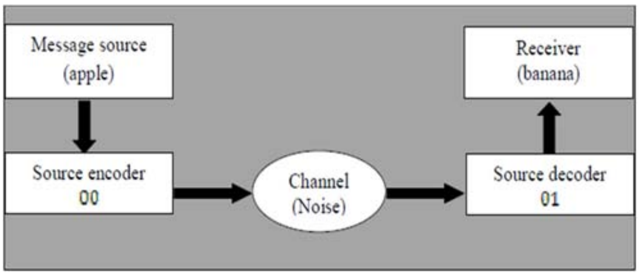

Figure 2. *A communication error occurred.*

With channel coding, this error may be detected (and even corrected) by introducing a redundancy bit as follows $(v_1v_2v_3v_4)$:

$$\begin{aligned}(00) &\to v_1 = (000),\\ (01) &\to v_2 = (011),\\ (10) &\to v_3 = (101),\\ (11) &\to v_4 = (110).\end{aligned}$$

The newly encoded message ‘apple’ is now (000). Suppose this message was transmitted and an error of one bit only occurred. The receiver may get one of the following: (100), (010) or (001). In this way, we can detect the error, as none of (100), (010) or (001) is among our encoded messages.

Note that the above channel encoding scheme does not allow us to correct errors. For instance, if (100) is received, then we do not know whether (100) comes from (000), (101) or (110). However, if more three redundancy bits are introduced instead of one bit, we will be able to correct errors. For instance, we can design the following channel coding scheme:

$$\begin{aligned}(00) &\to (00000),\\ (01) &\to (01111),\\ (10) &\to (10110),\\ (11) &\to (11001)\end{aligned}$$

Again if the message (00000) was transmitted over a noisy channel and that there is only one error introduced, then the received word must be one of the following five: (10000), (01000), (00100), (00010) or (00001). Since only one error occurred and since each of these five codes differs from (00000) by only one bit, and from the other three correct codes (01111), (10110) and (11001) by at least two bits, then the receiver will decode the received message into (00000) and, hence, the received message will be correctly decoded into ‘apple’.

Algebraic coding theory is basically divided into two major types of codes: Linear block codes and Convolutional codes, Blahut [2].

In this paper we present some encoding and decoding schemes as well as some used error detection/correction coding techniques using linear block codes only. We discuss only two types of linear block codes: Hamming and cyclic codes.

1.1. Problem Statement

In any environment, noise, electromagnetic radiations and any other forms of disturbances affect communication leading to corrupted messages, errors in the received messages or even to an extent of the message not being received at all.

1.2. Objectives of the Study

1.2.1. General Objective

The main objective of this study was to provide an overview of two types of linear block codes: Hamming and cyclic codes and study schemes and/or algorithms of error detection and correction of these codes.

1.2.2. Specific Objectives

To generate, encode and decode hamming and cyclic codes.

To detect and correct errors using hamming and cyclic codes.

1.3. Justification of Study

Transmission of data across noisy channel and other forms of interference affect communication resulting to misdirected messages. Hence, we need the recovery of these corrupted messages. Considering the present concern with privacy and secrecy, and the prospect that such problems will increase significantly as communication services and data repositories grow, importance is thus attached to finding means of detecting and correcting any error that occur. Thus the need for a code that fully guarantees security in the sense that whenever two or more persons send or receive this code then data integrity and authentication are guaranteed. In this paper we present some encoding and decoding schemes as well as error detection/correction coding techniques using linear block codes only.

This study is applicable in:

- Deep space communication.
- Satellite communication.
- Data transmission.
- Data storage.
- Mobile communication.
- File transfer.
- Digital audio/video transmission.

1.4. Null Hypothesis

Codes for non-binary input channels such as the dual – code [1] useful on multiple frequency shift channels, and on practical implementations of high rate codes cannot work on binary channels.

2. Literature Review

The history of data-transmission codes began in 1948 with the publication of a famous paper by Claude Shannon. Shannon showed that associated with any communication channel or storage channel is a number C (measured in bits per second), called the capacity of the channel, which has the following significance: Whenever the information transmission rate R (in bits per second) required of a communication or storage system is less than C then, by using a data-transmission code, it is possible to design a communication system for the channel whose probability of output error is as small as desired. Shannon, however, did not tell us how to find suitable codes; his contribution was to prove that they exist and to define their role. Throughout the 1950s, much effort was devoted to finding explicit constructions for classes of codes. The first block codes were introduced in 1950 when Hamming described a class of single-error-correcting block codes and he published what is now known as Hamming code, which remains in use in many applications today.

In 1957, among the first codes used practically were the cyclic codes which were generated using shift registers. It was quickly noticed by Prange that the cyclic codes have a rich algebraic structure, the first indication that algebra would be a valuable tool in code design.

In the 1960s, the major advances came in 1960 when Hocquenghem and Bose and Ray-Chaudhuri found a large class of multiple-error-correcting codes (the BCH codes). The discovery of BCH codes led to a search for practical methods of designing the hardware or software to implement the encoder and decoder. In the same year independently, Reed, Solomon and Arimoto found a related class of codes for non-binary channels. Concatenated codes were introduced by Forney (1966), later Justesen used the idea of a concatenated code to devise a completely constructive class of long block codes with good performance.

During the 1970s, these two avenues of research began to draw together in some ways and to diverge further in others. Meanwhile, Goppa (1970) defined a class of codes that is sure to contain good codes, though without saying how to identify the good ones.

The 1980s saw encoders and decoders appear frequently in newly designed digital communication systems and digital storage systems, Hamming [5].

The 1990s witnesses an evaluation of all groups in informatics at the universities in Norway. The evaluation was performed by a group of internationally recognized experts. The committee observed that the period 1988-92, had the largest number of papers (27) published in internationally refereed journals among all the informatics groups in Norway. In the period 1995-1997 the goal of finding explicit codes which reach the limits predicted by Shannon's original work has been achieved. The constructions require techniques from a surprisingly wide range of pure mathematics: linear algebra, the theory of fields and algebraic geometry all play a vital role, Han [6]. Not only has coding theory helped to solve problems of vital importance in the world outside mathematics, it also has enriched other branches of mathematics, with new problems as well as new solutions, Kolman [7]. In 1998 Alamouti described a space-time code.

In 2000 Aji, McEliece and others synthesize several decoding algorithms using message passing ideas. In the period 2002-2006 many books and papers are introduce such as Algebraic soft-Decision Decoding of Reed- Solomon Codes by Koetter R., and Error Control Coding: Fundamentals and Applications by Lin and Costello and Error Correction Coding by Moon T. in 2005.

During this decade, development of algorithms for hard-decision decoding of large nonbinary block codes defined on algebraic curves, Kabatiansky [8]. Decoders for the codes known as hermitian codes are now available and these codes may soon appear in commercial products. At the same time, the roots of the subject are growing even deeper into the rich soil of mathematics.

Doumen (2003), researched on the aims of cryptography in providing secure transmission of messages in the sense that two or more persons can communicate in a way that guarantees confidentiality, data integrity and authentication.

Sebastia (2003) studied on the Block error correcting codes. He found that the minimum distance decoder maximizes the likelihood of correcting errors if all the transmission symbols have the same probability of being altered by the channel noise. He also noted that if a code has a minimum distance d, then d(C) - 1 is the highest integer with the property that the code detects d(C) - 1 errors.

Todd [11], studied on the Error control coding. He showed that if a communication channel introduces fewer error than the minimum distance errors, d(C), then these can be detected and if d(C)- 1 errors are introduced, then error detection is guaranteed (see also [8]). He also noted that the probability of error detection depends only on the error introduced by the communication channel and that the decoder will make an error if more than half of the received bit strings are in error [9].

In 2009, Nyaga [10] studied the Cyclic ISBN-10 to improve the conventional ISBN-10. They designed a code that would detect and correct multiple errors without many conditions attached for error correction and found out that the code could correct as many errors as the code could detect. The method involves trial and error calculation and thus it needs to be improved on and simplified to speed up the process.

Asma & Ramanjaneyulu [12] studied the implementation of Convolution Encoder and Adaptive Viterbi Decoder for Error Correction. Egwali Annie and Akwukwuma [13] investigated Performance Evaluation of AN-VE: An Error Detection and Correction Code. Vikas Gupta and Chanderkant Verma [14] examined Error Detection and Correction: Viterbi Mechanism. Error Detecting and Error Correcting Codes were examined by Chauhan *et al* [15].

3. Methodology

This section sets the methodology of the research by discussing the Linear Block codes.

3.1. Basic concepts of Block Codes

The data of output of the source encoder are represented by sequence of binary digits, zeros or ones. In block coding this sequence is segmented into message blocks $u = (u_0 u_1 ... u_{k-1})$ consisting of k digits each.

There are a total of 2^k distinct messages. The channel encoder, according to certain rules, transforms each input message into a word $v = (v_0 v_1 ... v_{n-1})$ with $n \geq k$.

3.2. Basic Properties of a Linear Block Code

- The zero word (00…0), is always a codeword.
- If c is a codeword, then (-c) is also a codeword.
- A linear code is invariant under translation by a codeword. That is, if c is a codeword in linear code C, then C + c = C.
- The dimension k of the linear code C(n, k) is the dimension of C as a subspace of V_n over GF(2), i.e. $\dim(C) = k$.

3.3. Encoding Scheme

If $u = (u_0 u_1 ... u_{k-1})$ is the message to be encoded, then the corresponding codeword v can be given as follows: v = u. G

3.4. Error Detection, Error Correction & Decoding Schemes

A fundamental concept in secure communication of data is the ability to detect and correct the errors caused by the channel. In this chapter, we will introduce the general schemes/methods of linear codes decoding.

Channel Model / Binary Symmetric Channel

The channel is the medium over which the information is conveyed.

Examples of channels are telephone lines, internet cables and phone channels, etc. These are channels in which information is conveyed between two distinct places or between two distinct times, for example, by writing information onto a computer disk, then retrieving it at later time.

Now, for purposes of analysis, channels are frequently characterized by mathematical models, which (it is hoped) are sufficiently accurate to be representative of the attributes of the actual channel.

In this paper we restrict our work on a particularly simple and practically important channel model, called the binary symmetric channel (BSC), and defined as follows:

Definition 1: A binary symmetric channel (BSC) is a memoryless channel which has channel alphabet {0, 1} and channel probabilities.

p(1 received |0 send) = p(0 received | 1 sent) $= p < \frac{1}{2}$,

(after = there should not be space)

p(0 received | 0 send) = p(1 received | 1 sent) = 1 – p.

Figure 3, shows a BSC with crossover probability p.

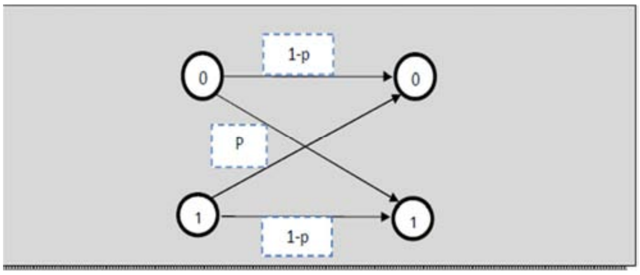

Figure 3. Binary Symmetric Channel.

3.5. General Methods of Decoding Linear Codes Over BSC

In a communication channel we assume a code word $v = (v_0 ... v_{n-1})$ is transmitted and suppose $r = (r_0 ... r_{n-1})$ is received at the output of the channel. If r is a valid codeword, we may conclude that there is no error in v. Otherwise, we know that some errors have occurred and we need to find the correct codeword that was sent by using any of the following general methods of linear codes decoding:

- Maximum likelihood decoding,
- Nearest neighbor/Minimum distance decoding
- Syndrome decoding
- Standard array
- Syndrome decoding using truth table

These methods for finding the most likely codeword sent are known as decoding methods.

3.5.1. Maximum Likelihood Decoding

Suppose the codewords $\{v_0, v_1, ..., v_{2^k - 1}\}$ form the linear block code C(n, k) and suppose a BSC with crossover probability $p < \frac{1}{2}$ is used.

Let a word $r = (r_0 r_1 ... r_{n-1})$ of length n be received when a codeword $v_r = (v_{r_0} v_{r_1} ... v_{r_{n-1}}) \in C$ is sent. Then, the maximum likelihood decoding (MLD) will conclude that v_r is the most likely codeword transmitted if v_r maximizes the forward channel probabilities.

3.5.2. Nearest Neighbor Decoding/Minimum Distance Decoding

Important parameters of linear block codes called the hamming distance and hamming weight are introduced as well as the minimum distance decoding.

3.5.3. The Minimum Distance Decoding

Suppose the codewords $\{v_0, v_1, ..., v_{2^k-1}\}$ from a code C(n, k) are being sent over a BSC. If a word r is received, the nearest neighbor decoding or (minimum distance decoding) will decode r to the codeword v_r that is the closest one to the received word r. Such procedures can be realized by an exhaustive search on the set of codewords which consists of comparing the received word with all codewords and choosing of the closest codeword.

3.5.4. Syndrome & Error Detection

Consider an (n, k) linear code C. Let $v = (v_0 v_1 ... v_{n-1})$ be a codeword that was transmitted over a noisy channel (BSC). Let $r = (r_0 r_1 ... r_{n-1})$ be the received vector at the output of the channel. Because of the channel noise, r may be different from v. Hence, the vector sum $e = r + v = (e_0 e_1 ... e_{n-1})$ is an n-tuple where $e_i = 1$, and $r_i \neq v_i$ for $i = 0,1,...,n-1$. This n-tuple is called an error vector or (error pattern). The 1s in e are the transmission errors that the code is able to correct.

3.5.5. Syndrome & Error Correction

The syndrome s of a received vector r = v + e depends only on the error pattern e, and not on the transmitted codeword v.

3.5.6. Error-Detecting& Error-Correcting Capabilities of Block Codes

Error-Detecting Capabilities of Block Codes

Let *m* be a positive integer. A code C is u error detecting if, whenever a codeword incurs at least one and at most u errors, the resulting word is not a codeword.

A code is exactly u error detecting if it is *m* error detecting but not $(m+1)$ error detecting.

Error-Correcting Capabilities of Block Codes

If a block code with minimum distance $d_{\min}$ is used for random-error correction, one would like to know how many errors that the code is able to correct.

A block code with minimum distance $d_{\min}$ guarantees correcting all the error patterns of $t = \left[\frac{d_{\min}-1}{2}\right]$ or minimal errors. The parameter *t* is called the random-error-correcting capability of the code.

3.5.7. Syndrome Decoding

We will discuss a scheme for decoding linear block codes that uses a one-to-one correspondence between a coset leader and a syndrome. So we can form a decoding table, which is much simpler to use than a standard array. The table consists of 2^{n-k} coset leaders (the correctable error patterns) and their corresponding syndromes.

So the exhaustive search algorithm on the set of 2^{n-k} syndromes of correctable error patterns can be realized if we have a decoding table, in which syndromes correspond to coset leaders.

4. Binary Hamming Codes

4.1. Construction of Binary Hamming Codes

Hamming codes are the first important class of linear error-correcting codes named after its inventor, Hamming [1] who asserted by proper encoding of information, errors induced by a noisy channel or storage medium can be reduced to any desired level without sacrificing the rate of information transmission or storage. We discuss the binary Hamming codes with their shortened and extended versions that are defined over GF (2). These Hamming codes have been widely used for error control in digital communication and data storage. They have interesting properties that make encoding and decoding operations easier.

In this section we introduce Hamming codes as linear block codes that are capable of correcting any single error over the span of the code block length.

Let $r \in Z$, the hamming code of order r is a code generated when you take a parity check matrix H and $r \times (2^r - 1)$ matrix with columns that are all the $(2^r - 1)$ non-zero bit strings of length r in any order such that the last r columns form an identity matrix.

Remark 1:

Interchanging the order of the columns lead to an equivalent code.

Example 1:

Find codewords in hamming code C of order.

1 b) 2 c) 3

Solution

r = 1

$$1 \times (2^1 - 1) matrix$$
$$\Rightarrow 1 \times 1$$
$$\Rightarrow (1)$$
$$G = 1$$
$$G = (I / A)$$
$$H = (A^t / I)$$
$$(1)(1) = 1$$
$$(1)(0) = 0$$
$$\Rightarrow C = \{0,1\}$$

r = 2

$$2 \times (2^2 - 1) matrix$$
$$\Rightarrow 2 \times 3$$
$$H = \begin{pmatrix} 110 \\ 101 \end{pmatrix} \Rightarrow (A^t / I_2)$$
$$A^t = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$
$$\therefore A = (11)$$
$$G = (I / A)$$
$$= (111)$$

Perform linear combinations of rows of G.

C = {111, 000}

Remark 2:

Let $r=1$. Then the G is a 1×1 matrix. Since $G=(I/A)$ and $H=(A^t/I)$, we conclude that $G=(1)$ and hence $C=\{0,1\}$.

Since all the codewords are linear combinations of the rows of G, this code has only two codewords.

$$r = 3$$

$$3\times(2^3-1) matrix$$

$$H\Rightarrow 3\times 7 matrix$$

$$H=\begin{pmatrix}0111100\\1011010\\1101001\end{pmatrix}$$

$$H=(A^t/I_3)$$

$$A^t=\begin{pmatrix}0111\\1011\\1101\end{pmatrix}$$

.

$$A=\begin{pmatrix}011\\101\\110\\111\end{pmatrix}$$

$$G=(I/A)$$

$$=\begin{pmatrix}1000011\\0100101\\0010110\\0001111\end{pmatrix}$$

Perform linear combination of rows of G.

= {1000011, 0100101, 0010110, 0001111, 1100110, 1010101, 1001100, 0110011, 0101100, 0011001, 1110000, 1101001, 0111100, 1111111, 0000000 and 0101010}

Suppose the linear code $C(n,k)$ has a $(n-k)\times n$ matrix H as the parity check matrix and that the syndrome of the received word r is given by $S^T=H.r^T$. Then the decoder must attempt to find a minimum weight e which solves the equation $S^T=H.e^T$.

Write $e=(e_0,e_1,...,e_{n-1})$ and $H=(h_0h_1,...,h_{n-1})$, where $e_i\in GF(2)\forall i=0,1,...,n-1$ and each h_i is an $(n-k)$ dimensional column vector over $GF(2)$, then

$$S^T=[h_0h_2...h_{n-1}]\begin{pmatrix}e_0\\e_1\\.\\.\\e_{n-1}\end{pmatrix}=\sum_{i=0}^{n-1}e_ih_i .$$

In other words, the syndrome may be interpreted as the vector sum of those columns of the matrix corresponding to the positions of the errors.

Now, consider all error words of weight one are to have distinct syndromes, and then it is evidently necessary and sufficient that all columns of the matrix must be distinct.

For if $w(e)=1$ say $e_i=1$ then $S_i^T=h_i$ if $e_j=1$ then $S_j^T=h_j$ now, if $S_i^T=S_j^T$ then $h_i\neq h_j$ for $i\neq j$.

In other words, the parity-check matrix H of this code consists of all the nonzero $(n-k)$-tuples as its columns. Thus, there are $n=2^{(n-k)}-1$ possible columns.

The code resulting from above is called a Binary Hamming code of length $n=2^m-1$ and $k=2^m-1-m$ where $m=n-k$.

Definition 2:

For any integer $m>1$ there is a Hamming code, Ham (m), of length 2^m-1 with m parity bits and 2^m-1-m information bits.

Using a binary $m\times n$ parity check matrix whose columns are all of the m - dimensional binary vectors different from zero, the Hamming code is defined as follows:

Ham $(m)=\{v=(v_0v_1...v_{n-1})\in V_n : H.v^T=0\}$

Table 1. *(n,k) Parameters for Some Hamming Codes.*

M	Hamming Code
3	(7, 4)
4	(15, 11)
5	(31, 26)
6	(63, 57)
7	(127, 120)

Theorem 1: The minimum distance of a Hamming code is at least 3.

Proof:

If Ham (m) contained a codeword v of weight 1, then v would have 1 in the i^{th} position and zero in all other positions.

Since $Hv^T=0=h_i$, then i^{th} column of H must be zero. This is a contradiction of the definition of H. So Ham (m) has a minimum weight of at least 2.

If Ham (m) contained a codeword v of weight 2, then v would have 1 in the i^{th} and j^{th} positions and zero in all other positions. Again, since $Hv^T=0=h_i+h_j$, then h_i & h_j are not distinct. This is a contradiction.

So Ham (m) has a minimum weight of at least 3.

Then $W_{\min}\geq 3$. Since $d_{\min}=W_{\min}$ in linear codes, then $d_{\min}\geq 3$, therefore the minimum distance of Hamming code is at least 3.

Theorem 2: The minimum distance of a Hamming code is exactly 3.

Proof:

Let $C(n,k)$ be a Hamming code with parity-check matrix $H_{m\times n}$. Let us express the parity-check matrix H in the following form:

$H = [h_1,..,h_i,..,h_j,..,h_{2^m-1}]$, where each h_i represents the i^{th} column of H. Since the columns of H are nonzero and distinct, no two columns add to zero. It follows that the minimum distance of a Hamming code is at least 3. Since H consists of all the nonzero m-tuples as its columns, the vector sum of any two columns, say h_i and h_j, must also be a column in H, say h_S i.e. $h_i + h_j = h_s$. Thus, $h_i + h_j = h_s = 0$ (In modulo 2-addition)

It follows from that the minimum distance of a Hamming code is exactly 3.

Corollary 1: The Hamming code is capable of correcting all the error patterns of weight one and is capable of detecting all 2 or fewer errors.

Proof:

$t = \left[\frac{d_{\min}-1}{2}\right] = \left[\frac{3-1}{2}\right] = 1$. So the Hamming code is capable of correcting all the error patterns of weight one.

And $d_{\min} - 1 = 3 - 1 = 2$. Thus it also has the capability of detecting all 2 or fewer errors.

Result

For any positive integer $m > 1$, there exists a Hamming code with the following parameters:

Code length: $n = 2^m - 1$

Number of information symbols: $k = 2^m - m - 1$

Number of parity-check symbols: $n - k = m$

Random-error-correcting capability: $t = 1(d_{\min} = 3)$.

4.2. The Generator and the Parity Check Matrices of Binary Hamming Codes Ham (m)

GENERATOR MATRICES

When we use PCB we encode a message $x_1x_2...x_k$ as $x_1x_2...x_kx_{k+1}$ where $x_{k+1} = \sum_{i=1}^{k} x_i(\text{mod}\,2)$.

To generalize this notion we add more than one check bit and encode the message $x_1x_2...x_k$ as $x_1x_2...x_kx_{k+1}...x_n$. Where the last n – k bits are PCB's obtained from the k bits in the message.

The PCB are $x_{k+1}x_{k+2}...x_n$ specified as follows:

i. Consider the k bit message $x_1x_2...x_k$ as a 1 x k matrix X.

ii. Let G be a k x n matrix that begins with I_k. I.e. k x k identity matrix. Hence $G = (I_k / A)$ where A is a k x (n - k) matrix. G is called a generator matrix.

iii. We encode the message X as E(X) =XG doing arithmetic mod 2.

Example 3:

a Consider encoding by adding the PCB to a 3 bit message, where

$$G = \begin{pmatrix} 1001 \\ 0101 \\ 0011 \end{pmatrix}.$$

(I.e.) The column of 1's is added to I_3.

$$G = (I_3 / A). \text{ Where } A = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

b Consider the encoding using triple repetition for 3 bit messages as follows.

$$G = (I_3 \mid I_3 \mid I_3).$$

$$G = \begin{pmatrix} 100100100 \\ 010010010 \\ 001001001 \end{pmatrix}.$$

c Let

$$G = \begin{pmatrix} 100111 \\ 010110 \\ 001101 \end{pmatrix}.$$

$$G = (I_3 \mid A). \text{ Where}$$

$$A = \begin{pmatrix} 111 \\ 110 \\ 101 \end{pmatrix}.$$

What codewords does G generate?

Solution:

E(X) = XG

$$f : B^3 \to B^6$$
$$B^3 = \{000,001,010,100,011,101,110,111\}.$$

$$(000)G = 000000$$
$$(001)G = 001101$$
$$(010)G = 010110$$
$$(100)G = 100111$$
$$(011)G = 011011$$
$$(101)G = 101010$$
$$(110)G = 110001$$
$$(111)G = 111100$$

$$G = \{000000,001101,010110,100111,011011,101010,110001,111100\}$$

Remark 3:

a The codewords in a binary code generated by the generator matrix G can be obtained by performing all possible linear combinations of the rows of G working mod 2.

$$\text{I.e. } G = \begin{pmatrix} 100111 \\ 010110 \\ 001101 \end{pmatrix}.$$

Rows = 100111, 010110, 001101

Adding any = 110001, 011011, 101010, 111100, 000000.

The binary codes formed using the generator matrix have the closure property. They are therefore linear codes. I.e. consider y_1 and y_2 codewords generated by G.

$$\begin{aligned} &I.e. y_1 = x_1G, i.e.. E(x_1) \\ &y_2 = x_2G, i.e.. E(x_2) \\ &E(x_1+x_2) = (x_1+x_2)G \end{aligned} .$$

$$\begin{aligned} &= x_1G + x_2G \\ &= y_1 + y_2 \end{aligned} .$$

Parity Check Matrices

A simple way to detect errors is by use of parity check bit (PCB).

A parity bit, or check bit is a bit added to the end of a string of binary code that indicates whether the number of bits in the string with the value one is even or odd. Parity bits are used as the simplest forms of error detecting code. There are two types of parity bits: even parity bit and odd parity bit. An odd number of bits (including the parity bit) are transmitted incorrectly; the parity bit will be incorrect, thus indicating that a parity error occurred in the transmission. Because of its simplicity, parity is used in many hardware applications where an operation can be repeated in case of difficulty, or where simply detecting the error is helpful. In serial data transmission, a common format is 7 data bit, an even parity bit, and one or two stop bits. This format neatly accommodates all the 7-bit ASCII characters in a convenient 8-bit byte.

If a bit string contains an even number of 1s we put 0 at the end.

If it contains an odd number of 1s we put a 1 at the end.

Our aim is to ensure an even number of 1s in any codeword.

I.e. message $x_1x_2...x_n$.

Encoded as $x_1x_2...x_nx_{n+1}$.

Where $x_{n+1} = x_1 + x_2 + ... + x_n$.

A single error in communication will therefore be noted since it will change the parity.

Example 4:

Message: 101

Encoded as 1010

Suppose sent: 101

Received: 111 (check 111(odd) → error)

Example 5:

Message: 10101

Encoded as 101011

Suppose sent: 101011

Received: 111111(check (even number of 1s) → no error), but there is an unnoticed error.

Remark 4:

We notice that, when an even number of errors occur, it is not noticed.

Suppose a PCB is added to a bit string during transmission, what would you conclude on the following received messages.

a 101011101 – It contains an even number of 1s. Hence it is either a valid codeword or contains an even number of errors.

b 11110010111001 – It contains an odd number of 1s hence it cannot be a valid codeword and must therefore contain an odd number of errors.

Consider the generator matrix $G = \begin{pmatrix} 100111 \\ 010110 \\ 001101 \end{pmatrix}$ the bit string $x_1x_2x_3$ is encoded as $x_1x_2x_3x_4x_5x_6$ where:

$$\begin{aligned} x_4 &= x_1 + x_2 + x_3 \\ x_5 &= x_1 + x_2 \\ x_6 &= x_1 + x_3 \end{aligned} .$$

$$\text{I.e.} \left.\begin{aligned} x_1 + x_2 + x_3 + x_4 &= 0 \\ x_1 + x_2 + x_5 &= 0 \\ x_1 + x_3 + x_6 &= 0 \end{aligned}\right\} \text{– parity check equations.}$$

I.e.

$$\begin{pmatrix} 111100 \\ 110010 \\ 101001 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

$H[E(x)^t] = 0$. Where $E(x)^t$ is the transpose of E(x) and the parity check matrix is

$$\begin{aligned} H &= \begin{pmatrix} 111100 \\ 110010 \\ 101001 \end{pmatrix}. \\ &= (A^t / I_3) \end{aligned}$$

In general H $= (A^t / I_{n-k})$.

Remark 5:

- Relationship between generator matrix and parity check matrix. Suppose G is a k x n matrix (i.e.) G $= (I_k / A)$. Hence A is a k x (n - k) matrix. We associate to G the parity check matrix H where: H $= (A^t / I_{n-k})$. x is then a codeword iff G = (I_k / A), H = (A^t / I_{n-k}).
- From a generator matrix, we can find the associated parity check matrix and vice versa.

4.3. Syndrome & Error Detection/Correction

The parity check matrix is used to detect errors.

Any received bit string y that does not satisfy the equation,

$Hy^t = 0$ is not a valid codeword, that is, it is in error.

When the columns of the parity check matrix are distinct and are all non-zero, H can be used to correct the errors.

Suppose x is sent and y received in error then y = x + e, e being the error string.

If e = 0 then no error.

In general the error string e has a 1 in the position where y differs from x and 0 in all other places.

Example 6:

$$x = 110010$$

$$y = 100010$$

$$\Rightarrow y = x + e$$
$$Hence, e = 010000 \cdot$$

Remark:

$$H[y^t] = H[x+e]^t .$$

$$= Hx^t + He^t$$
$$= He^t \cdot$$

$Hy^t = He^t = c_j$. Where c_j is the jth column of H.

Assuming no more than one error exists, we can find the codeword x that was send by simply computing Hy^t. If Hy^t = 0, then no error and y is the sent codeword. Otherwise the j^{th} bit is in error and should be changed to produce x.

Example 7:

$$\text{Let } G = \begin{pmatrix} 100111 \\ 010110 \\ 001101 \end{pmatrix} .$$

Obtain H.

Determine the codeword sent given the received:

y = 001111

y = 010001

Assuming no more than one error.

Solution

$$G = (I_3 / A) \cdot$$

$$\text{Where A} = Hy^t .$$

$$H = (A^t / I_3) \Rightarrow H = \begin{pmatrix} 111100 \\ 110010 \\ 101001 \end{pmatrix}$$

(i) Hy^t.

$$\begin{pmatrix} 111100 \\ 110010 \\ 101001 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = c_j \cdot$$

Check H $\Rightarrow c_j$ is the 5th column. Hence the 5th bit string received is in error.

$y = 0011\underline{1}1$.

Hence x = 001101

(ii) Hy^t.

$$\begin{pmatrix} 111100 \\ 110010 \\ 101001 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = c_j .$$

Check H $\Rightarrow c_j$ is the 1st column. Hence the 1st bit string received is in error.

$$y = \underline{0}10001 \cdot$$

Hence x = 110001.

4.4. Cyclic Codes

Cyclic codes form an important subclass of linear block codes and were first studied by Prange in 1957. These codes are popular for two main reasons: first, they are very effective for error detection/correction and second, they possess many algebraic properties that simplify the encoding and the decoding implementations.

A code C is said to be cyclic if:

i. C is a linear code.

ii. Whenever a right or left shift is performed on any codeword it yields another codeword.

i.e. whenever $a_0a_1...a_n \in C \;\; then \;\; a_1a_2...a_na_0 \in C$.

Remark 6:

$a_1a_2...a_na_0$ is the first cyclic shift.

Example 8:

$$C = \{000, 110, 101, 011\}$$

Hence C is cyclic.

$$000 \to 000 \to 000$$
$$110 \to 101 \to 011 \to 110 \cdot$$

Cyclic codes are useful in:

i. Shift registers.

ii. On the theoretical side, cyclic codes can be investigated by means of algebraic theory of rings and polynomials.

Description of Cyclic Codes

If the components of an n –tuple $v = (v_0v_1...v_{n-1})$ are cyclically shifted one place to the right, we obtain another n-tuple, $v^{(1)} = (v_{n-1}v_0...v_{n-2})$ which is called a cyclic shift of v.

Clearly, the cyclic shift of v is obtained by moving the right most digit v_{n-1} of v to the left most digit and moving every other digit $v_0, v_1, ...v_{n-2}$ one position to the right.

Shifting the components of v cyclically, i places to the right, the resultant n-tuple would be

$$v^{(i)} = (v_{n-i}v_{n-i-1}...v_{n-1}v_0.v_1...v_{n-i-1})$$

Remark 9: Cyclically shifting v i-places to the right is equivalent to cyclically shifting $v(n-i)$-places to the left.

Definition 3: An (n,k) linear code C is called cyclic if any cyclic shift of a codeword in C is also a codeword in C, i.e. whenever $(v_0v_1...v_{n-1}) \in C$, then so is $(v_{n-1}v_0...v_{n-2})$.

Example: Consider the following (7, 4) linear code C;

$$C = \{(0000000),(1101000),(0110100),(1011100),(0011010),(1110010),(0101110),\\ (1000110),(0001101),(1100101),(0111001),(1010001),(0010111),(1111111),\\ (0100011),(1001011)\}$$

One can easily check that the cyclic shift of a codeword in C is also a codeword in C. For instance, let $v = (1101000) \in C$, then $v^{(1)} = (0110100) \in C$:

Hence, the code C is a cyclic.

Correspondence between bit string and polynomials over Z_2.

The key to algebraic treatment of cyclic code is the correspondence between the word $a = a_0a_1a_2...a_{n-1}$ is V^n *or* B^n and polynomial

$$a(x) = a_0 + a_1x + a_2x^2 + ... + a_{n-1}x^{n-1} in \quad \mathrm{Z}_2[x]$$

In this correspondence the first cyclic shift of a codeword $\hat{a}$ is represented by the polynomial

$$\hat{a}(x) = a_{n-1}x^0 + a_0x + a_1x^2 + ... + a_{n-2}x^{n-1}.$$

i.e.

Table 2. *First Cyclic Shift*

	x^0	x^1	x^2	$x^3....x^{n-1}$
$a(x)$	a_0	a_1	a_2	a_3 .... a_{n-1}
$\hat{a}(x)$	a_{n-1}	a_0	a_1	a_2 ... a_{n-2}

Consider:

$$\begin{aligned}
&xa(x) = x(a_0 + a_1x + a_2x^2 + ... + a_{n-1}x^{n-1})\\
&\textit{Consider}\\
&xa(x) - a_{n-1}x^{n-1}\\
&= x(a_0 + a_1x + ... + a_{n-1}x^{n-1}) - a_{n-1}(x^n - 1)\\
&\Rightarrow a_0x + a_1x^2 + ... + a_{n-1}x^n - a_{n-1}x^n + a_{n-1} = \hat{a}(x)\\
&\hat{a}(x) = xa(x) - a_{n-1}(x^n - 1)\\
&= xa(x)\bmod(x^n - 1)
\end{aligned}$$

Working with polynomials help us to perform operations on cyclic codes for better understanding.

We denote the ring of polynomials modulo $x^n - 1$ by $V^n[x]$ with coefficients in Z_2.

The addition and multiplication of polynomials modulo $x^n - 1$ can be regarded as addition and multiplication of equivalence classes of polynomials.

The equivalence classes form a ring, and iff F(x) is reducible we get a field.

Example 9:

$$f(x) = 1 + x^2 \quad in \quad V^3[x].$$

Solution: Elements P(x) in $V^3[x]$ are;

$$0, 1 + x + x^2, 1, x, x^2, 1 + x, 1 + x^2, x + x^2.$$

$$\begin{aligned}
&< 1 + x^2 >= \{0, 1 + x^2, 1 + x, x + x^2\}\\
&0 \rightarrow 000\\
&1 + x^2 \rightarrow 101\\
&1 + x \rightarrow 110\\
&x + x^2 \rightarrow 011
\end{aligned}$$

C = {000, 101, 110, 011}

4.4.1. Shift-Register Encoders for Cyclic Codes

In this section we present circuits for performing the encoding operation by presenting circuits for computing polynomial multiplication and division.

Hence, we shall show that every cyclic code can be encoded with a simple finite-state machine called a shift-register encoder.

To define the shift register we want to by the following definition;

Definition 4: A D flip-flop is a one-bit memory storage in the field $GF(2)$.

Figure 4. *Flip-Flop.*

External clock: Not pictured in our simplified circuit diagrams, but an important part of them, which generates a timing signal ("tick") every t_0 seconds.

When the clock ticks, the content of each flip-flop is shifted out of the flip-flop in the direction of the arrow, through the circuit to the next flip-flop.

The signal then stops until the next tick.

Adder: The symbol of adder has two inputs and one output, which is computed as the sum of the inputs (modulo 2-addition

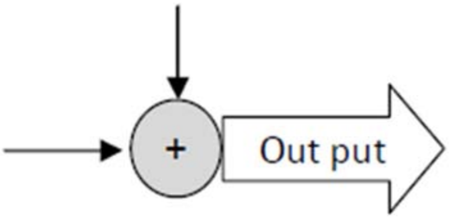

Figure 5. *Adder.*

Multiplication: The symbol of multiplication has one input and one output, where the output is the multiplication of the input and the number g_i which is stored in this symbol (either 1 or 0), where 0 represented by no connection and 1 by a connection.

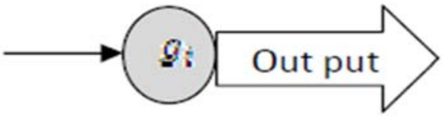

Figure 6. *Multiplication.*

Definition: A shift-register is a chain of $(n-k)$ D flip-flops connected to each other, where the output from one flip-flop becomes the input of the next flip-flop.

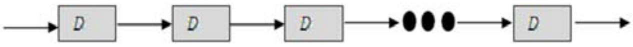

Figure 7. *Shift Register.*

All the flip-flops are driven by a common clock, and all are set or reset simultaneously.

4.4.2. Cyclic Codes Decoding

Decoding of cyclic codes consists of the same three steps as for decoding linear codes:

a Syndrome computation.
b Association of the syndrome to an error pattern.
c Error correction.

For any linear code, we can form a standard array, or we can use the reduced standard array using syndromes. For cyclic codes it is possible to exploit the cyclic structure of the code to decrease the memory requirements.

First we must determine if the received word r is a codeword in C or not using a Theorem which states that an $r(x) \in C$ if and only if

$$r(x)h(x) \equiv 0 \bmod x^n + 1 \Rightarrow (x^n+1) divided\ r(x)h(x).$$

If $r(x) \notin C$ we determine the closest codeword in $C(n,k)$ using the syndrome of as r(x) follows:

Since every valid received code polynomial $r(x)$ must be a multiple of the generator polynomial $g(x)$ of C, then when we divide $r(x)$ by $g(x)$ the remainder is zero exactly when $r(x)$ is a codeword, i.e.

$$r(x) = a(x)g(x) + 0$$

Thus we can employ the division algorithm to obtain a syndrome as follows:

$$r(x) = a(x)g(x) + s(x)$$

where $a(x)$ is the quotient and $s(x)$ is the remainder polynomial having degree less than the degree of $g(x)$:

$$s(x) = s_0 + s_1 x + ... + s_{n-k-1} x^{n-k-1}$$

Thus, to compute the syndrome we can use a circuit.

5. Applications, Conclusion and Recommendation

5.1. Applications

This study is applicable in:

- Deep space communication.
- Satellite communication.
- Data transmission.
- Data storage.
- Mobile communication.
- File transfer.
- Digital audio/video transmission.

5.2. Conclusion

Decoding can be accomplished in the following manner:

i. If $s(r) = 0$, then we assume that no error occurred.
ii. If $s(r) \neq 0$ and it contains odd number of 1's, we assume that a single error occurred. The error pattern of a single error that corresponds to s is added to the received word for error correction.
iii. If $s(r) \neq 0$ and it contains even number of 1's, an uncorrectable error pattern has been detected.

5.3. Recommendation

Hamming code corrects only the error patterns of single error and is capable of detecting all 2 or fewer errors hence finding a way to correct more than one error and detect more than two errors would be effective.

All error detection, correction controlling mechanisms has been studied. Hamming code is most efficient error correction mechanism in long distance communication. Interesting area of future research is the study of how the presence of caches would affect the correlation in the data input to the ECC memory, and whether there is any systematic pattern there that can be exploited by the optimization algorithms

Acknowledgements

The authors wish to thank Taita Taveta University for the support given towards the completion of this research. Special thanks goes to the Department of Mathematics and Informatics.

References

[1] Attarian Ad. Algebraic Coding Theory. 2006. 12P.

[2] Blahut R. Algebraic Codes for Data Transmission. United Kingdom: Cambridge University Press; 2003. 482p.

[3] Doran R. Encyclopedia of Mathematics and its Applications. 2nd ed. Cambridge University Press; 2002. 205.

[4] Hall J. Notes on Coding Theory. United State America: Michigan State University. 2003. 10P.

[5] Hamming R. Error Detecting and Error Correcting Codes. Bell Syst. Tech. J., 29. 1950; 147-160.

[6] Han Y. Introduction to Binary Linear Block Codes. National Taipei University. Taiwan. 97P.

[7] Kolman B. Introductory Linear Algebra: with Applications. 3rd ed. United States of America: Prentice Hall; 1997. 608P.

[8] Kabatiansky G. Error Correcting Coding and Security for Data Networks. John Wiley & Sons, Ltd; 2005. 278p.

[9] Lemmermeyer F. Error Correcting Codes. 2005. 100P.

[10] Nyaga, L. and Cecilia, M. (2008). Increasing error detection and correction efficiency in the ISBN. Discovery and Innovation, 20: 3–4.

[11] Todd, K. M. (2005). Error Correction Coding: Mathematical Methods and Algorithms. John Wiley & Sons Inc.

[12] Asma & Ramanjaneyulu [2015]: Implementation of Convolution Encoder and Adaptive Viterbi Decoder for Error Correction, International Journal of Emerging Engineering Research and Technology.

[13] Egwali Annie O. and Akwukwuma V. V. N. (2013): Performance Evaluation of AN-VE: An Error Detection and Correction Code, African Journal of Computing & ICT.

[14] Vikas Gupta, Chanderkant Verma (2012): Error Detection and Correction: Viterbi Mechanism, International Journal of Computer Science and Communication Engineering.

[15] Neha Chauhan, Pooja Yadav, Preeti Kumari (2014): Error Detecting and Error Correcting Codes, International Journal of Innovative Research in Technology.