1-1-1984

On the error-correcting capability and decoding of cyclic codes of composite length.

Homayoun Shahri

ON THE ERROR-CORRECTING CAPABILITY AND DECODING OF CYCLIC CODES OF COMPOSITE LENGTH

by

Homayoun Shahri

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

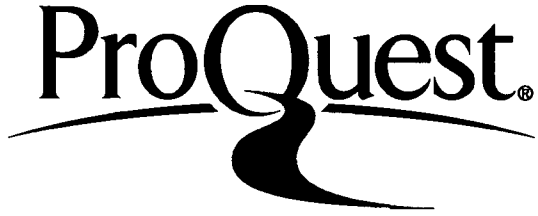
Master of Science

in

Electrical Engineering

Lehigh University

1984



This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science.

Sept. 7, 1984
Date

Professor in Charge

Chairman of Department

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I wish to take this time to express my deep gratitude and appreciation to my professor and thesis advisor, Dr. K.K. Tzeng, for the help and guidance he gave me in my studies of coding theory.

I would also like to gratefully acknowledge the support extended to me by the department of Computer Science and Electrical Engineering of Lehigh University, during my studies for the completion of the Degree of Master of Science.

Finally, I wish to thank Valerie J. VanBilliard for typing the manuscript.

ABSTRACT

In this thesis we studied the error-correcting capability of a subclass of cyclic codes of composite length $n=n_1 \times n_2 \times \ldots$ such that n_1, n_2, ... are pairwise relatively prime. This study led us to a new decoding procedure for this class of codes. The algorithm is based on the contractions of a binary cyclic code $\underline{C}$ of composite length. When applied to a certain subclass of codes of composite lengths for correcting burst errors, we found that this decoding procedure is optimal asymtotically, and gives considerably good results in many other cases. A study of the minimum distance of this class of codes is also conducted. And is shown that this decoding procedure may also be utilized for correcting random errors. This decoding technique is very easy to apply to certain codes of composite lengths. An efficient decoding algorithm for this class of codes is also presented. This work concludes by giving some suggestions on this subject for further investigations.

I. INTRODUCTION

In recent years, the demand for efficient and reliable digital data communication and storage systems, has been increasing. The recent breakthrough in developing large-scale integrated circuits and data transmission and storage networks has caused an even greater increase in demand for more reliable and high transmission rate channels.

A good design of these systems must incorporate both computer and communication technology. A major task for the designer is the control of error so that the data may be reproduced reliably.

When information is transmitted, noise might cause the received data to be slightly different from the original data. As Shannon showed in 1948, the noise need not cause any decrease in reliability. However, the noise does introduce some limiting capacity on the throughput rate of the channel, although that limit is typically above the throughput rate at which real systems operate. Error-correcting codes enable a system to achieve a high degree of reliability despite the presence of noise. In addition to the data bits that one wishes to transmit,

one also must transmit some additional redundant check bits, to combat the additive noise. Even though the noise causes some errors in both the transmitted data bits and the transmitted check bits, there is usually still enough information available to the receiver to allow the decoder to correct all the errors unless the noise is extremely severe.

There are two important class of codes in use today, block codes and convolutional codes. The encoder for a block code divides the information into messages of k-bits each, written as a binary k-tupple $U = (u_1, u_2, \ldots, u_k)$ called a message. Then the encoder transforms this vector into a binary n-tupple $V = (v_1, v_2, \ldots, v_n)$ which is called a code word. Thus there are a total of 2^k messages and 2^k possible code words.

The set of 2^k code words form an (n, k) block code. (In this work we are only concerned with block codes. For more information on both block and convolutional codes, please refer to [2]).

There are two types of errors that can occur in a channel. If the noise affects each transmitted symbol independently, then the errors occur randomly and so we

have random errors. Codes for correcting random errors are called random error correcting codes. Examples of random-error-correcting channels are: deep space channels and many satellite channels. Most line of sight transmission facilities are as well affected by random errors.

When the error does not affect all bits at random but occurs in cluster or bursts, then the channels are called burst-error-channels, and codes developed for correcting burst errors are burst error correcting codes. Examples of burst-error-channels are: impulsive switching noise and crosstalk, errors caused by signal fading due to multipath transmission.

And finally there are channels that have a combination of burst and random errors. These are called compound channels and codes devised for correcting error on these channels are called burst-and random-error-correcting codes. (In this work we are concerned primarily with burst errors, although occasionally we will also consider random errors)

Cyclic codes form an important subclass of linear codes. These codes are attractive for two reasons:

First, encoding and decoding can be implemented easily by employing shift registers with feed back connections (or linear sequential circuits), and second, because they have considerable inherent algebraic structure, it is possible to find various practical methods for decoding them.

"Cyclic codes were first studied in 1957 by Prange. Since then, progress in the study of cyclic codes for both random-error-correction and burst-error-correction has been spurred by many algebraic coding theorists." [8]

In our effort to study the bounds on the minimum distance of cyclic codes, we came across the most recent work by Wilson and Van Lint on this subject. In which they improve all previously known bounds on minimum distance of cyclic codes namely BCH bound, Hartmann-Tzeng bound and Roos' bound. Based on part of Wilson and Van Lint's recent research, we developed a new decoding algorithm for correcting random or burst error. This decoding algorithm applies to a subclass of cyclic codes with composite length $n=n_1 \times n_2 \times \ldots$, where n_1, n_2, ... are relatively prime. The decoding is based on a set of very simple decision makings and can be carried out very fast. When applied to codes for correcting bursts, this

algorithm is assymtotically opitmal, and is efficient in many other cases.

Cyclic codes of composite length may be mapped onto codes with shorter lengths ('contractions'), in which case these shorter codes can be used to decode errors. This is the basic idea that is used in devising this new decoding procedure.

In Chapter II, we will consider some preliminaries which prove to be essential to understanding our main result and the discussions following it.

In Chapter III we will describe our main result and several examples will be presented as to how this procedure can be applied. At the end of the chapter we discuss some further results which deal with the minimum distance of this class of codes and their burst and random error correcting capability.

In Chapter IV, we will present the decoding algorithm and will consider a few examples.

Chapter V presents the conslusions drawn from this investigation and suggests some areas of further research.

II. PRELIMINARIES

In this chapter we will present some background material for decoding of random-error and/or burst-error correcting codes. A review of cyclic codes and their burst-error correcting capability will conclude the chapter.

Throughout this thesis work, we use the standard notations from coding theory. A cyclic code C of length n over the field GF(q) with m as multiplicative order of q modulo n, is generated by a polynomial g(x) over GF(q), which divides x^n-1. The minimum distance of C is denoted by d. In the case of binary codes, the distance between two code words is the number of places they differ, and hence the minimum distance is the minimum of the distances between all code vectors contained in C.

If α is a primitive n'th root of unity in an extension field $GF(q^m)$ of GF(q) then g(x) is a product of polynomials $m_i(x)$ where $m_i(x)$ denotes the minimal polynomial of α^i (i.e., the polynomial with smallest degree such that α^i is a zero of $m_i(x)$) over GF(q). It will be convenient to use the following terminology. If

$A = \{\alpha^{i_1}, \alpha^{i_2}, \ldots, \alpha^{i_\ell}\}$ is a set of n'th roots of unity such that:

$c(x) \in C \Leftrightarrow \forall \zeta A;[c(\zeta) = 0]$. Where C is the set of all code polynomials in this case. Then we shall say that A is a defining set for C.

Let C be a binary (n, k) linear code. Let v_1, v_2, ..., v_{2^k} be the code vectors of C. Regardless of which code vector is transmitted on a noisy channel, the received vector r can be any of the 2^n n-tuples over GF(2). Any decoding algorithm is a way of partitioning the 2^n possible received vectors into 2^k disjoint subsets S_1, S_2, ..., S_2k, such that there is a one to one correspondance to a code vector v_i. If the received vector r is found in the subset S_i, r is decoded into v_i. Correct decoding is possible if and only if the received vector r is in the subset S_i which corresponds to the actual code vector transmitted over the noisy channel.

A scheme to partition the 2^n possible received vectors into 2^k disjoint subsets satisfying the condition that each subset contains one and only one code vector is described below.

This partition is dependent on the linear structure of the code. First, the 2^k code vectors of C are written

in a row with the all zero code vector $v_1 = (0, 0, \ldots, 0)$ as the first left most) element. From the remaining $2^n - 2^k$ n-tuples, an n-tuple e_2 is selected and written under the zero vector v_1. Next, the second row is formed by adding e_2 to each vector v_1, in the first row and writing the sum $e_2 + v_1$ under v_1. When the second row is completed, the procedure continues until all 2^n n-tuples are partitioned. Each partition is called a coset. Thus we have an array of rows and columns shown below. This is called the standard array of the given linear code C. [8]

$$
\begin{array}{cccc}
v_1=0 & v_2 & v_i & v_{2^k} \\
e_2 & e_2+v_2 & e_2+v_i & e_2+v_{2^k} \\
\vdots & \vdots & \vdots & \vdots \\
e_\ell & e_\ell+v_2 & e_\ell+v_i & e_\ell+v_{2^k} \\
\vdots & \vdots & \vdots & \vdots \\
e_{2^{n-k}} & e_{2^{n-k}}+v_2 & e_{2^{n-k}}+v_i & e_{2^{n-k}}+v_{2^k}
\end{array}
$$

Theorem 1.1. [8], (a version of Lagrange's theorem). No two n-tuples in the same row of a standard array are identical. Every n-tupee appears in one and only one row.

Proof." The first part of the theorem follows from the fact that all the code vectors of C are disjoint. Suppose that two n-tuples in the ℓ'th row are identical, say $e_\ell+v_i=e_\ell+v_j$ with $i \neq j$. This means that $v_i=v_j$, which is impossible. Therefore, no two n-tuples in the same row are identical.

It follows from the construction rule of the standard array that every n-tuple appears at least once. Now, suppose that an n-tuple appears in both ℓ'th row and the m'th row with $\ell < m$. Then this n-tuple must be equal to $e_\ell+v_i$ for some i and equal to e_m+v_j for some j. As a result, $e_\ell+v_i=e_m+v_j$. From this equality we obtain $e_m=e_\ell+(v_i+v_j)$. Since v_i and v_j are code vectors in C, v_i+v_j is also a code vector in C, say v_s. Then $e_m=e_\ell+v_s$. This implies that the n-tuple e_m is in the ℓ'th row of the array, which contradicts the construction rule of the array that e_m the first element of the m'th row, should be unused in any previous row. Therefore, no n-tuple can appear in more than one row of the array, concludes the proof of the second part of the theorem." [8]

□

As was discussed in chapter I there are communication channels which are affected by disturbances that cause transmission errors to cluster into bursts. For example, on telephone lines, a stroke of lightning or a human-made electrical distrubance frequently affects many adjacent transmitted digits. On magnetic storage systems, magnetic tape defects may last up to several miles and cause clusters of errors. Therefore, it is desirable to design codes specifically for correcting burst errors. Codes of this kind are called burst-error-correcting codes.

Cyclic codes are effective not only for burst error detection, but they are also very effective for burst-error correction. Many effective cyclic codes for correcting burst-errors have been discovered for the past 20 years. Cyclic codes for single-burst-error correction were first studied by Abramson,[3] In an effort to generalize Abramson's results, Fire discovered a large class of burst-error-correcting cyclic codes. Fire codes can be decoded with very simple circuitry. Besides the Fire codes. Many other effective burst-error-correcting cyclic codes have been constructed and analyzed both analytically and with the aid of a computer.

Such as Burton codes, product codes, interleavel codes, etc." [8]

A burst of length ℓ is defined as a vector whose non-zero components are confined to ℓ consecutive digit positions; the first and last of which are non-zero. For example e=(0000 0110100000) is a burst of length 6. A linear code that is capable of correcting all error burst of length $\underline{\ell}$ or less but not all error bursts of length $\ell+1$ is called an ℓ burst-error-correcting code, or the code is said to have burst-error-correcting capability ℓ. It is clear that for give code length n and burst-error-correcting capability ℓ, we must construct an (n,k) code with as small a redundancy (n-k) as possible; next, we will prove a number of theorems which prove to be helpful in presenting our main result.

Theorem 2.1. A linear block code that has no burst of length ℓ or less as a code word must have at least ℓ parity-check symbols. For a proof refer to [2]

Theorem 2.2. For detecting all burst error of length ℓ or less with a linear block code of length n, ℓ parity-check symbols are necessary and sufficient. For a proof refer to [2]

Theorem 2.3. (Reiger, 1960, [2]) In order to correct

all burst errors of length b or less, a linear block code must have at least 2b parity-check symbols. In order to correct all bursts of length b or less and simultaneously detect all burst of length $\ell \geq b$ or less, the code must have at least $b+\ell$ parity-check symbols.

Proof. "Any vector that has the form of a burst of length 2b or less can be written as the difference of two bursts of length b or less (except in the degenerate case of a burst consisting of a single non-zero element). Since, in order to correct all burst of length b or less these must be in different cosets, their difference cannot be a code word. The first part of the theorem then follows from theorem (2.1).

Similarly, every burst of length $b+\ell$ or less can be written as the difference of a burst of length ℓ or less and a burst of length b or less. If the code is simultaneously to correct bursts of length b or less and to detect all bursts of length ℓ, the burst of length b and the burst of length ℓ must be in different cosets, and their sum must not be a code word, this theorem then follows from theorem 2.1." [2] □

For a given n and k, theorem 2.3 implies that the burst-error-correcting capability of an (n,k) code is at

at most $(n-k)/2$, that is, $\frac{n-k}{2}$. This is an upper bound on the burst-error-correcting capability of an (n,k) code and is called the Reiger bound. Codes that meet the Reiger bound are said to be optimal. The ratio

$$z = \frac{2\ell}{n-k}$$

is used as a measure of the burst correcting efficiency of a code. An optimal code has burst correcting efficiency equal to 1.

III. A STUDY OF CYCLIC CODES OF COMPOSITE LENGTH VIA CONTRACTIONS

In this chapter we will discuss a new decoding method for cyclic codes of composite lengths and we shall show that for correcting bursts this new method meets the Reiger bound asymtotically in many cases and proves to be efficient in other cases.

All the codes that we will consider in this section are binary cyclic codes with composite lengths $n=n_1$, n_2, ... where n_1, n_2, ... are relatively prime.

Let C_n be a cyclic code over the field GF(q) and generated by g(x), and let m be the multiplicative order of q modulo n. Let $n=n_1.n_2$. Next we define a homomorphism with operation addition over the field GF(2) from C_n to C_{n_1}(cyclic code of length n_1), which is a mapping of the codes in C_n onto the codes in C_{n1}. This homomorphism is called contraction, and C_{n1} is a contraction of C_n. Now we are ready to prove a theorem (due to Wilson and Van Lint [5]) which is the basis of our work.

Theorem 3.1. Let C_n be a binary cyclic code of length $n= n_o$ for which the defining set contains $\alpha^{\ell_{i_1}}, \alpha^{\ell_{i_2}}, \ldots \alpha^{\ell_{i_k}}$. ($\alpha$ is a primitive n'th root of

unity). Let C_{n_0} be a cyclic code of length n_0 (a "contraction" of C) with defining set:

$\{\zeta^{i_1}, \zeta^{i_2}, \ldots \zeta^{i_k}\}$, $(\zeta = \alpha^{\ell})$ then if

$$b_j = C_j + C_{j+n_0} + C_{j=2n_0} + \ldots + C_{j=(\ell-1)n_0};$$

$$c \in C_n \quad b = (b_0, \ldots, b_{n_{0-1}}) \in C_{n_0}$$

Proof. Note that α is a primitive n'th root of unity and therefore $\zeta = \alpha^{\ell}$ is a primitive n_0'th root of unity.

let $c = (c_0, c_1, \ldots, c_{n-1}) \in C$

$$\sum_{j=0}^{n-1} c_j (\alpha^{\ell i \nu})^j = 0 \qquad \forall \qquad 1 \leqslant \nu \leqslant k$$

This follows because $\alpha^{\ell i \nu} \quad \forall \quad 1 \leqslant \nu \leqslant k$ are the roots of the generator polynomial. Therefore they must satisfy the code polynomial.

And since $\sum_{j=0}^{n-1} c_j (\alpha^{\ell i \nu})^j = 0$, we can write

$$\sum_{j=0}^{n-1} c_j (\alpha^{\ell i \nu})^j = \sum_{j=0}^{n_0-1} c_j (\alpha^{\ell i \nu})^j + \sum_{j=n_0}^{2n_0-1} c_j (\alpha^{\ell i \nu})^j +$$

$$\ldots + \sum_{j=(\ell-1)n_0}^{n-1} c_j (\alpha^{\ell i} \nu)^j = 0$$

let $b_j = c_j + c_{j+n_0} + c_{j+2n_0} + \ldots + c_{j+(\ell-1)n_0}$

$$\sum_{j=0}^{n-1} c_j (\alpha^{\ell i} \nu)^j = \sum_{j=0}^{n_0-1} b_j (\alpha^{\ell i} \nu)^j = 0$$

This follows because $\ell n_0 = n = 0 \bmod n$. Then $(\ell i_\nu \; xj)$ repeats a cycle, that is, α^ℓ has order n_0 and hence, $b = (b_0, b_1, \ldots, b_{n_{0-1}}) \in C_{n_0}$. Therefore b is a code word in C_{n0}. □

Example 3.1. Let C be a binary cyclic code of length $n = n_1 \times n_2$ such that $(n_1, n_2) = 1$. (i.e., n_1 and n_2 are relatively prime). Suppose that the defining set of C contains $(\alpha^{n_1}, \alpha^{n_2})$ where α is a primitive n'th root of unity. This implies that α^{n_1} is a primitive n_2'th root of unity and has order n_2, hence it can generate a code of length n_2. Thus C can be contracted to a code of length n_2. For the same reason C can also be contracted to a code of length n_1, where its defining set contains α^{n_2}.

The purpose of this example was to show that by

selecting properly the defining set of the code C we can contract it to one or both of its factors. In this work however, we are primarily concerned with simultaneous contractions, the reason being that we do not gain very much by looking at only one contraction alone. Although this can be used for error detection.

We will make the term simultaneous contraction more clear by considering an example.

Example 3.2. Let C_n be a binary cyclic code of length n = 15 and defining set:

$$S = \left\{ \alpha^i \;\middle|\; i = 3,5 \right\}.$$

Notice that α^3 is primitive 5'th root of unity and has order 5, and α^5 is a primitive 3'rd root of unity and hence has order 3.

Consider a code word $c \in C_n$

$$c = (c_0, c_1, c_2, \ldots, c_{14})$$

the contractions are formed as follows:

I. CONTRACTION OF C_{15} to C_5

$$C_0 + C_5 + C_{10} = b_0$$

$$C_1 + C_6 + C_{11} = b_1$$

$$C_2 + C_7 + C_{12} = b_2$$

$$C_3 + C_8 + C_{13} = b_3$$

$$C_4 + C_9 + C_{14} = b_4$$

II. CONTRACTION OF C_{15} to C_3

$$C_0 + C_3 + C_6 + C_9 + C_{12} = b'_0$$

$$C_1 + C_4 + C_7 + C_{10} + C_{13} = b'_1$$

$$C_2 + C_5 + C_8 + C_{11} + C_{14} = b'_2$$

As was shown in theorem 3.1. $b = (b_0, b_1, \ldots, b_4)$ is a code word in C_5 and $b' = (b'_0, b'_1, b'_2)$ is a code word in C_3.

The important point to note is the following: Each row of table I has at most one element in common with each row of table II i.e., they are orthogonal on one element. Of course this property holds as long as n can be factored into two relatively prime factors.

While studying the properties of simultaneous contractions, we thought that since each row of tables I and II are orthogonal on one element, we might be able to correct and/or detect errors. As it turns out and will be shown later, this procedure can best deal with bursts of error.

At this point there are two questions that must be answered before proceeding any further.

(1) How is the code word c related to its contractions? (i.e., if c is an odd or even weight code word, what is the parity of its contractions?)

(2) What should the error correcting capability of the two contractions be, to ensure maximum efficiency? The first question may be answered by proving the follow-lemma.

Lemma 3.2. A code word $c \in C_n$, (where C_n is a binary cyclic code of length $n = n_1 \times n_2$) with odd parity contracts to a code word b C_n, with odd parity, and a code word $c \in C_n$ with even parity contracts to a code word b C_{n1} with even parity.

Proof. Consider code word $c \in C_n$

$$c = (c_0, c_1, \ldots, c_{n-1})$$

and consider the contraction of c to b with length n_1

$$\begin{array}{l} c_0 + c_{n_1} + c_{2n_1} + \ldots + c_{n=n_1} = b_0 \\ \vdots \qquad\qquad\qquad\qquad\qquad \vdots \\ c_1 \qquad\qquad\qquad + c_{n-n_1+1} = b_1 \\ \vdots \qquad\qquad\qquad\qquad\quad \vdots \\ c_2 \qquad\qquad\qquad\qquad\quad \vdots \\ \vdots \qquad\qquad\qquad\qquad\quad \vdots \\ c_{n_1-1} + c_{2n_1-1} + \ldots + c_{n-n_1-1} + c_{n-1} = b_{n_1-1} \end{array}$$

Case I. c has odd parity, i.e., $[c_0+c_1+\ldots+c_{n-1})=1]$

$$(c_0+c_{n_1}+\ldots+c_{n-n_1}) + (c_1+c_{(n_1+1)}+\ldots+c_{(n-n_1+1)} +$$

$$\ldots + (c_{n_1-1}=c_{2n_1-1}+\ldots+c_{n-1}) = 1$$

$$\text{or } b_0+b_1+\ldots+b_{n_1-1} = 1$$

This follows because $c_0+c_{n_1}+\ldots+c_{n-n_1}=b_0$ is the first row of the contraction table and therefore as shown

$(b_0+b_1+\ldots+b_{n_1-1}) = 1$ which concludes the proof of case I.

Case II. can be proven similarly and hence the proof is omitted. □

Therefore lemma 3.2. shows that the even weight codes in c_n are contracted to even weight codes in c_{n_1} and odd weight codes in c_n are mapped onto odd weight codes in c_{n_1}. A corollary from lemma 3.2. is the following: In the case of two simultaneous contractions, the odd weight codes in c_n are contracted to odd weight codes in c_{n_1} and simultaneously odd weight codes in c_n are mapped onto odd weight codes in c_{n_2}. Of course the result holds for even weight codes.

We are still conducting further research to answer the second question, namely, how many contractions are needed, or how the length of the contractions should be chosen to ensure maximum efficiency. Although of all the cases that we studied, contracting the code C to two trivial (repetition) codes leads to very simple decoding and ensures optimality in asymtotic sense, in many cases. Where as contracting to two or more non-trivial codes

results in more complex decoding and at the same time does not guarantee maximum efficiency for correcting bursts.

It is possible to show that by selecting the defining set S of a binary cyclic code C of length $n=n_1 \times n_2$ properly, we can always contract it to two trivial codes of lengths n_1 and n_2.

We are now ready to discuss the decoding procedure for single bursts. It will be much easier to present the algorithm by way of an example.

Example 3.3. Let C_n be a binary cyclic code of length n=15. Let the defining set of C_n be $S = \{\alpha^3, \alpha^5\}$, where α is a primitive 15'th root of unity, α^3 is a primitive 5'th root of unity and has order 5. Its minimal polynomial has degree 4 and therefore the contraction of C_n to C_{n_1} ($n_1 = 5$) is a trivial code. α^5 is a primitive 3'rd root of unity and its minimal polynomial has degree 2. Hence the contraction of C_n to C_{n_2};($n_2 = 3$) is also a trivial code. Let us consider the two tables as in ex.3.2

I. CONTRACTION OF C_{15} to C_5

$$C_0 + C_5 + C_{10} = b_0$$

$$C_1 + C_6 + C_{11} = b_1$$

$$C_2 + C_7 + C_{12} = b_2$$

$$C_3 + C_8 + C_{13} = b_3$$

$$C_4 + C_9 + C_{14} = b_4$$

II. CONTRACTION OF C_{15} to C_3

$$C_0 + C_3 + C_6 + C_9 + C_{12} = b'_0$$

$$C_1 + C_4 + C_7 + C_{10} + C_{13} = b'_1$$

$$C_2 + C_5 + C_8 + C_{11} + C_{14} = b'_2$$

Suppose there is a single burst of length 1 and furthermore assume that C has odd parity. Lemma 3.2 implies that $b = b_0, \ldots, b_4$ and $b' = (b'_0, b'_1, b'_2)$ must both be all 1's vectors. Let us say that C_6 has been received incorrectly. Then the vectors b and b' are:

$$b = (10111)$$

$$b' = (011)$$

Since both b and b' must be all 1's vectors we know that there is an incorrect bit in the second row of table I and the first row of table II but the two rows mentioned above are orthogonal on c_6. Hence the error is identified and can be corrected.

Now suppose there is a burst of length 2. Say c_2 and c_3 are received incorrectly, in which case b and b' are

$$b = (11001)$$

$$b' = (010).$$

From this we can see that the burst is contained in third and fourth row of table I, and first and third row of table II. We can form the following array:

row (3) of I and row (1) of II are orthogonal on C_{12}

row (3) of I and row (3) of II are orthogonal of C_2

row (4) of I and row (1) of II are orthogonal of C_3

row (4) or I and row (3) of II are orthogonal of C_8

Assuming that the burst error correcting capability of this code is less than or equal to 3. Then the bits in error must be consecutive and therefore c_2 and c_3 must be the errors. The decoding is exactly the same if we have a burst of length 3. Such that, the burst has the

for (101). But the problem arises as soon a solid burst of length 3 has corrupted the received code vector. Say c_0, c_1, c_2 are corrupted bits, then b and b' could be:

b = (00011)

b' = (000) if $c \in C_n$ has odd parity

The burst is contained in row one, two and three of table I and rows one, two and three of table II. But there are three possible error patterns

(C_0, C_1, C_2) or

(C_5, C_6, C_7) or

(C_{10}, C_{11}, C_{12}).

Hence considering any of these error patterns as the error pattern and changing the corresponding bits in the code word would satisfy the two contractions of C_n. Hence an incorrect decoding might result. The interested reader must bear in mind that the problem arises as soon as we have a solid burst of length equal to the length of the shorter contraction code. (In this case 3).

Lemma 3.3. A solid burst of length equal to the length of the shorter contraction of C_n (that is C_{n_2}) cannot be decoded.

Proof. Let us assume that the burst has length (n_2) then its polynomial representation will be:

$$e(x) = x^{n_2-1} + x^{n_2-2} + \ldots + x + 1.$$

This follows because the code is cyclic and the corresponding error vector is

$$\bar{e} = (\underbrace{000\ldots00}_{n-n_2}\underbrace{111\ldots11}_{n_2})$$

and furthermore all such errors can be represented by:

$$(x^{n_1})^p \cdot e(x);\ o \leqslant p \leqslant n_2$$

Let us consider a polynomial consisting of two of these bursts. Say,

$v(x) = e(x) + x^{n_1} e(x)$, that is we have:

$$v(x) = e(x) + x^{n_1} e(x) = e(x)\cdot[x^{n_1} + 1]$$

$$v(x) = (\underbrace{x^{n_2-1} + x^{n_2-2} + \ldots}_{g_2(x)} + x+1)\cdot(x^{n_1} - 1) \text{ and}$$

$$(x^{n_1-1} + x^{n_1-2} + \ldots + x + 1) = g_1(x) \text{ but}$$

$g_1(x)$ and $g_2(x)$ are the generator polynomials of the two trivial codes which are the contractions of C_n. This

implies that if we receive a code vector with an error pattern e(x), and furthermore, we decode it incorrectly to $(x^{n_1})^p$; $1 \leqslant p \leqslant n_2$ the syndrome will be zero, and decoder cannot know that an incorrect decoding has occured.□

Therefore this is a limitation on this decoding algorithm. For this particular example that is the (15,9) BCH code, (n-k) is 6. Therefore to achieve the Reiger bound this code must correct a burst of length 3. However all the bursts of length 3 can be corrected except the solid burst of length 3.

Next we decided to try an alternative way which is to transform the solid burst to a double error and use the random error correcting capability of C_n to locate the burst.

The procedure is as follows:

Let e be a solid burst of length n_2. Therefore we can represent the burst as a polynomial of degree (n_2-1) as shown in the previous example. That is

$$e(x) = x^{n_2-1} + x^{n_2-2} + \ldots + x + 1$$

Suppose: the received vector r = v+e that is, the code vector v has been corrupted by e and r has been received. In polynomial form we have $r(x) = v(x) + e(x)$

let us multiply r(x) by (x+1), which gives

$$(x+1)\cdot r(x) = (x+1)\, v(x) + (x+1)\, e(x)$$
$$= (x+1)\, v(x) + (x^{n_2}+1) = (x+1)\, v(x) + e'(x)$$

thus the solid bursts is transformed into a burst of length (n_2+1) such that only its beginning and end bits are 1. Therefore theoretically one can correct this burst using the random error correcting capability of the code, and this information can be used to locate the burst, and hence the proper decoding follows. However we found that if the defining set of these codes (i.e., cyclic codes with length $n = n_1 \times n_2$ such that $(n_1, n_2) = 1$) is formed by selecting only the roots that are necessary to form the contractions, the minimum distance of the code will be 4. Therefore the code is not capable of correcting all combinations of double errors and in particular for the cases that there is a tie we have:

$e(x)$ is a solid burst of length n_2

$(x^{n_1})^p \cdot e(x)$ is another solid burst of same length that would result in the same syndrome, where $0 \leqslant P \leqslant n_2$

let us consider a combination of two of these bursts

that are transformed using the procedure that was presented. w.l.o.g. we have:

$$e''(x) = (x+1)\cdot e(x) + (x+1)\cdot (x^{n_1})^P \cdot e(x)$$
$$= (x^{n_2}+1) + (x^{n_1})^P \cdot (x^{n_2}+1) = (x^{n_2}+1)\cdot((x^{n_1})^P+1)$$

which is divisable by $g(x) = g_1(x) \cdot g_2(x)$ where $g_1(x) = x^{n_2-1} + x^{n_2-2} + \ldots + x + 1$ and $g_2(x) = x^{n_1-1} + x^{n_1-2} + \ldots + x + 1$ are the generator polynomial of the two contractions of c_n of lengths n_2 and n_1 respectively, and $g(x)$ is the generator polynomial of c_n. Hence $e''(x)$ is a code word polynomial therefore if an incorrect decoding results the syndrome will be zero. Hence the double errors of such form cannot be corrected.

Let us define the following, S_{n_1} and S_{n_2} are the defining sets for the two contractions of C_n with defining set S_n, such that $S_{n_1} \cup S_{n_2} = S_n$ and $S_{n_1} \cap S_{n_2} = \phi$ and the two contractions are trivial codes. Then the burst error correcting capability of this class of codes is one less than the length of the shorter contraction of C_n, and furthermore all bursts of length equal to n_2 can be corrected except the solid burst of such length. Hence to achieve the best efficiency on this class of codes, n_1 and n_2 must be as close as

possible.

If $n_1 = n_2+2$ then to meet the Reiger bound the code of length $n = n_1 \cdot n_2$ must correct a burst of length up to n_2. However our procedure limits this capability to (n_2-1) therefore this decoding procedure meets the Reiger upper bound asymtotically. The interested reader must also bear in mind that only one combination of a burst of length n_2 cannot be corrected. (i.e., the solid burst of length n_2).

However sometimes it is possible to achieve the Reiger bound, by adding one extra root to the defining set of C_n. The idea is to increase the minimum distance, so that we can transform the solid burst into a double error and use the random error correcting capability of the code to correct it. One example that we found is the (15,9) code with the defining set $S_n = \{\alpha^3, \alpha^5\}$, by adding α^1 to the defining set the minimum distance increases to 8, so the code c'_n with defining set $S'_n = \{\alpha^1, \alpha^3, \alpha^5\}$ is capable of correcting any combinations of three random errors. C'_n is a (15,5) code, hence according to Reiger bound it must be capable of correcting a burst of length 5. Let us consider the (15,5) code.

Example 3.4. Let c'_n be cyclic code of length 15 with defining set $S'_n = (\alpha^1, \alpha^3, \alpha^5)$, and two contraction tables as in example 3.3. We have already considered a burst of length 3. Notice that a solid burst of length 3 can be transformed to a double error and hence can be corrected.

I. CONTRACTION OF C_{15} to C_5

$$C_0 + C_5 + C_{10} = b_0$$
$$C_1 + C_6 + C_{11} = b_1$$
$$C_2 + C_7 + C_{12} = b_2$$
$$C_3 + C_8 + C_{13} = b_3$$
$$C_4 + C_9 + C_{14} = b_4$$

II. CONTRACTION OF C_{15} to C_3

$$C_0 + C_3 + C_6 + C_9 + C_{12} = b'_0$$
$$C_1 + C_4 + C_7 + C_{10} + C_{13} = b'_1$$
$$C_2 + C_5 + C_8 + C_{11} + C_{14} = b'_2$$

Let us consider a solid burst of length 4 say, the solid burst corrupts (c_0, c_1, c_2, c_3) then if the code had odd weigth before being corrupted, we would have

b = (00001) and b' = (100). In this case an incorrect decoding might result because from the tables it seems that the error is contained in row 5 of table I and row 1 of table II so one might pick c_9 as the possible candidate, however according to a well known theorem considered in chapter II a burst of length (n-k) cannot be a code word, that is in this case a burst of length 10 cannot be a code word, therefore if c_9 is picked as the corrupted bit, the syndrome will not be zero. Hence the decoder will know that the error is contained in rows 1 through 4 of table I hence a solid burst, and to decode. Either one can transform the burst or simply compare the two tables as discussed in example 3.3. However one can easily show that all the bursts of length up to 5 can be corrected, and the procedure involves checking the syndrome. However for this particular code we can meet the Reiger bound using error trapping decoding, etc. of all the other codes that we considered, the (15,5) code was the only one which we were successful in reaching the Reiger theoretical upper bound. In contracting the (15,5) code our intension was to increase the minimum distance of the code, which was done by adding more roots to its defining set.

Next we considered the code of length 45 that contracts to 9 and to 5, based on this we formed the (45,33) code for which Reiger bound guarantees that a burst of length 12 cannot be a code word. The defining set of this code is $S_n=\{\alpha^5,\alpha^9,\alpha^{15}\}$, we added α^3,α^{21} to S_n, hence the new defining set is $S'_n=\{\alpha^3,\alpha^5,\alpha^9,\alpha^{15},\alpha^{21}\}$ (where α is a primitive root of unity). This did not increase the minimum distance of the code (d=4) so, the code is only capable of correcting one random error as guaranteed by the Hamming bound. However, this is a (45,25) code, and the roots are chosen such that this code can be contracted to length 3, 5, 9 and 15. For this code Reiger bound guarantees a burst of length 20 is not a code word, therefore to meet the Reiger bound this code must be capable of correcting any burst of length 10.

We now consider an example which shows the burst-error-correcting capability of this code.

Example 3.5. Let c_n be a cyclic code of length 45, for this code if only the contractions of length 5 and 9 are considered, our procedure guarantees that any burst of length 4 can be corrected, and in this case this code is only (4/6 = .67) 67% efficient. However when

α^3 and α^{21} are added to the defining set of this code we have also the contraction of c_n of length 15 and that of length 3. Let us consider the tables (labeled as I, II, III and IV):

I.	3, 9, 15, 21 ⟶	15	Code of length 15
II.	5, 15 ⟶	9	Code of length 9
III.	9 ⟶	5	Code of length 5
IV.	15 ⟶	3	Code of length 3

TABLE I: CODE OF LENGTH 15

0	15	30
1	16	31
2	17	32
3	18	33
4	19	34
5	20	35
6	21	36
7	22	37
8	23	38
9	24	39
10	25	40
11	26	41
12	27	42
13	28	43
14	29	44

TABLE II: CODE OF LENGTH 9

0	9	18	27	36
1	10	19	28	37
2	11	20	29	38
3	12	21	30	39
4	13	22	31	40
5	14	23	32	41
6	15	24	33	42
7	16	25	34	43
8	17	26	35	44

TABLE III: CODE OF LENGTH 5

0	5	10	15	20	25	30	35	40
1	6	11	16	21	26	31	36	41
2	7	12	17	22	27	32	37	42
3	8	13	18	23	28	33	38	43
4	9	14	19	24	29	34	39	44

TABLE IV: CODE OF LENGTH 3

0	3	6	9	12	15	18	21	24	27	30	33	36	39	42
1	4	7	10	13	16	19	22	25	28	31	34	47	40	43
2	5	8	11	14	17	20	23	26	29	32	35	48	41	44

Using tables II and III we can correct a burst of length 4. If tables I and IV are also used it is very easy to see that a burst of length 8 can be corrected. But

suppose a code vector is corrupted by a solid burst of length 9, say that the burst covers c_0 through c_8. This burst is not distinguishable from another solid burst covering c_{15} through c_{23} or the burst covering c_{30} through c_{38}. In polynomial form we have:

$$e_1(x) + e_2(x) = x^8 + x^7 + \ldots + 1 + x^{23} + x^{22} + \ldots + x^{15}$$

$$(x+1)\cdot e_1(x) + e_2(x) = x^9 + 1 + x^{24} + x^{15} = (x^9 + 1) + x^{15}(x^9 + 1)$$

$$= (x^9 + 1)\cdot(x^{15} + 1)$$

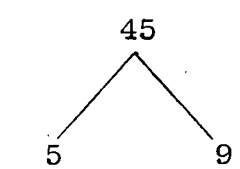
however $\mathrm{LCM}\left\{m_3(x).m_9(x).m_{15}(x).m_{21}(x)\right\} \mid (x^{15}+1)$ and

$\mathrm{LCM}\left\{m_5(x).m_{15}(x)\right\} \mid (x^9+1)$

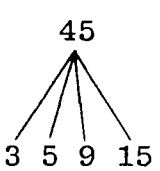
and this implies that the sum of two such bursts is a code word, and hence, they result in the same syndrome, thus such bursts cannot be corrected. Please note that all bursts of length 9 can be corrected except a solid burst of such length. To conclude the example we must mention that by adding α^3 and α^{21} to the defining set c_n and hence forming the contractions to length 21 and 3. The efficiency of this code jumped to (8/10 = .80) 80% an increase of 13 percent. Therefore sometimes we can

gain more by looking at several contractions in parallel. So far in all our examples we considered parallel contractions. However sometimes it is possible to consider sequential contractions or parallel contractions in conjunction with sequential contractions.

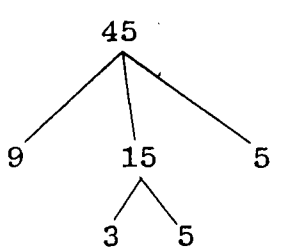
Next we tried to apply this to cyclic code of length 45, we selected the defining set of this code such that it contracts to trivial codes of lengths 5 and 9. Next we add α^3 to the defining set, and since α^3 has order 15 it can generate a code of length 15. If only α^3 is in the defining set, it forms a (15,11) code. However in conjunction with α^{15} and α^9 which have order 3 and 5 respectively, the set $\alpha^3, \alpha^9, \alpha^{15}$ generates a (15,5) code which was considered in one of the examples and it was shown that this code is capable of correcting a burst of length 5. This will enable the decoder to break the tie when a solid burst of length 5 occurs.



two parallel contractions



four parallel contractions



sequential and parallel contraction together

All the codes that we have considered so far have length $n = (n_1 \cdot n_2)$, where $(n_1, n_2) = 1$. In many cases however, it is possible to form 3 or more contractions such that all are relatively prime to each other.

Example 3.6. Consider the binary cyclic code c_n of length $n = 105 = n_1 \cdot n_2 \cdot n_3$, where $(n_1, n_2) = 1$, $(n_1, n_3) = 1$ and $(n_2, n_3) = 1$, hence by selecting the defining set properly, we can form the contractions of c_n to c_{n_1}, c_{n_2} and c_{n_3}, where $n_1 = 7$, $n_2 = 5$ and $n_3 = 3$ respectively. To meet the Reiger bound this code must be capable of correcting a burst of length 6.

Let us consider the contraction tables of this code. Note that the elements in each row represent the positions of a code word of length 105 in c_n.

(3, 5, 7) n = 105

I. (3, 1) CODE CONTRACTION

0 3 6 9 12 15 18 21 24 27 30 33 36 39 42
45 48 51 54 57 60 63 66 69 72 75 78
81 84 87 90 93 96 99 102

1 4 7 10 13 16 19 22 25 28 31 34 37 40
43 46 49 52 55 58 61 64 67 70 73 76
79 82 85 88 91 94 97 100 103

2 5 8 11 14 17 20 23 26 29 32 35 38 41
44 47 50 53 56 59 62 65 68 71 74 77
80 83 86 89 92 95 98 101 104

II. (5, 1) CODE CONTRACTION

0 5 10 15 20 25 30 35 40 45 50 55 60 65
70 75 80 85 90 95 100

1 6 11 16 21 26 31 36 41 46 51 56 61 66
71 76 81 86 91 96 101

2 7 12 17 22 27 32 37 42 47 52 57 62 67
72 77 82 87 92 97 102

3 8 13 18 23 28 33 38 43 48 53 58 63 68
73 78 83 88 93 98 103

4 9 14 19 24 29 34 39 44 49 54 59 64 69
74 79 84 89 94 99 104

III. (7, 1) CODE CONTRACTION

0	7	14	21	28	35	42	49	56	63	70	77	84	91
	98												
1	8	15	22	29	36	43	50	57	64	71	78	85	92
	99												
2	9	16	23	30	37	44	51	58	65	72	79	86	93
	100												
3	10	17	24	31	38	45	52	59	66	73	80	87	94
	101												
4	11	18	25	32	39	46	53	60	67	74	81	88	95
	102												
5	12	19	26	33	40	47	54	61	68	75	82	89	96
	103												
6	13	20	27	34	41	48	55	62	69	77	83	90	97
	104												

and the roots of the generator polynomial are $S = \{\alpha^{35}, \alpha^{21}, \alpha^{15}, \alpha^{45}\}$, where α is a primitive 105'th root of unity. Any 3 rows that belong to different tables are orthogonal on one element and hence we have a total of $\binom{3}{1}\binom{5}{1}\binom{7}{1} = 105$ possible combinations thus we can use this property for error correction.

Since any combination of three rows of different tables are orthogonal on one element then obviously any single burst of length one can be corrected. Now,

let us assume that a burst of length two occurs, say that bits 0 and 1 are erased, by refering to the table, one can easily see that the two bits in rows one and two of tables I, II and III are bit zero and bit one. The reader should note that we are considering a burst of length two. However, let us assume that a solid burst of length 3 occurs say that it covers bits 0, 1, and 2. In this case table I would not give us any information and the bits in common between the first three rows of tables I, II and III are (0, 1, 2), (35, 36, 37) and (70, 71, 72), hence the decoder fails. In polynomial representation we have:

$$\ell_1(x) = x^2 + x + 1, \quad \ell_2(x) = x^{37} + x^{36} + x^{35}$$

$$\ell_1(x) + \ell_2(x) = x^{35}(x^2 + x + 1) + (x^2 + x + 1)$$

$$= (x^2 + x + 1)(x^{35} + 1)$$

$$g(x) = (x^2 + x + 1)(x^4 + x^3 + x^2 + x + 1)(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1)$$

and since $g(x) \mid (\ell_1(x) + \ell_2(x))$ these two error patterns have the same syndrome, and hence the decoding of this error pattern is not possible. However, all other bursts of length three can be corrected.

For this code we were not able to find any set of

consecutive roots and hence BCH bound, says that min. dist. is 2, HT and Roos bounds also did not show anything more.

As to what the actual minimum distance is, we do not know. However this method is very simple for random error correction. Especially, since it does not involve any kind arithmatic over finite fields.

One may also contract a code of length 105 to trivial codes of length 35, 21 or 15. If these are simultaneous contractions the defining set contains $\{\alpha^7, \alpha^{21}, \alpha^{35}, \alpha^{49}\}$ which generate the code of length 15 and $\{\alpha^5, \alpha^{15}, \alpha^{25}, \alpha^{35}, \alpha^{45}\}$ that generate the code of length 21 and finally $\{\alpha^3, \alpha^9, \alpha^{25}, \alpha^{21}; \alpha^{45}\}$ which generate the code of length 35. The number of parity checks equals 68. Hence to meet the Reiger bound, decoder must correct bursts of up to length 34. Note that Lemma 3.2. also applies to three or more simultaneous contractions. The contraction tables for this code are shown below:

15 ⟶ 7, 21, 35, 49	length 15 code
21 ⟶ 2, 15, 25, 35, 45	length 21 code
35 ⟶ 3, 9, 15, 21, 45	length 35 code

where the numbers in the bracket represent powers of

in an extension field.

(15, 1) CODE

0	15	30	45	60	75	90
1	16	31	46	61	76	91
2	17	32	47	62	77	92
3	18	33	48	63	78	93
4	19	34	49	64	79	94
5	20	35	50	65	80	95
6	21	36	51	66	81	96
7	22	37	52	67	82	97
8	23	38	53	68	83	98
9	24	39	54	69	84	99
10	25	40	55	70	85	100
11	26	41	56	71	86	101
12	27	42	57	72	87	102
13	28	43	58	73	88	103
14	29	44	59	74	89	104

(21, 1) CODE

0	21	42	63	84
1	22	43	64	85
2	23	44	65	86
3	24	45	66	87
4	25	46	67	88
5	26	47	68	89
6	27	48	69	90
7	28	49	70	91
8	29	50	71	92
9	30	51	72	93
10	31	52	73	94
11	32	53	74	95
12	33	54	75	96
13	34	55	76	97
14	35	56	77	98
15	36	57	78	99
16	37	58	79	100
17	38	59	80	101
18	39	60	81	102
19	40	61	82	103
20	41	62	83	104

(35, 1) CODE

0	35	70
1	36	71
2	37	7
3	38	73
4	39	74
5	40	75
6	41	76
7	42	77
8	43	78
9	44	79
10	45	80
11	46	81
12	47	82
13	48	83
14	49	84
15	50	85
16	51	86
17	52	87
18	53	88
19	54	89
20	55	90
21	56	91
22	57	92
23	58	93
24	59	94
25	60	95
26	61	96
27	62	97
28	63	98
29	64	99
30	65	100
31	66	101
32	67	102
33	68	103
34	69	104

If one does not wish to use the decoding procedure for correcting bursts, instead correcting random errors are

intended. The following considerations are to be taken into account.

1) This algorithm best handles single random error or a single burst of length one.

2) For correcting a single burst of length one there is no need to contract the original code that has composite lengths c_n to trivial codes. It is necessary and sufficient if the contractions of c_n are capable of correcting a single error.

To clarify this let us consider an example.

Example 3.7. Let C be a binary cyclic code of length n = 21. If the defining set of this code is $S = \{\alpha^3, \alpha^7, \alpha^9\}$, we can contract it to two trivial codes of length 3 and 7 in which case it has minimum distance four, as will be shown later. Therefore the random error correcting capability of this code is one. However if the defining set of this code does not contain α^9, then the contraction of c to length 7 will not be a trivial code, rather it is a (7,4) Hamming code which has minimum distance 3 which is the minimum distance of the code C. Which is capable of correcting any single burst of length one.

Example 3.8. Let C be a binary cyclic code of length n = 35, let the defining set of this code be $S = \{\alpha^5, \alpha^7, \alpha^{15}\}$, in which case the code C can be contracted to trivial codes of length 5 and 7. But if α^{15} is omitted from the defining set, the code of length 7 will not be trivial, rather it is a (7, 4) hamming code with minimum distance 3, which is also the minimum distance of C.

Example 3.9. Let c be a binary cyclic code of length n = 63. Let the defining set of this code be $S = \{\alpha^7, \alpha^9, \alpha^{21}, \alpha^{27}\}$. Thus code C can be contracted to two trivial codes of length 7 and 9. The minimum distance of the code is 4. Now suppose that α^{21} and α^{29} are omitted from the defining set of C. In which case $S = \{\alpha^7, \alpha^9\}$ and thus C contracts to a (7,4) hamming code and a (9, 3) code, which both have minimum distance 3. The new minimum distance of C is also 3. Therefore C can correct any single burst of length one.

ON THE MINIMUM DISTANCE OF CYCLIC CODES OF LENGTH $n=n_1.n_2$

In this section we are going to prove that if c_n is a cyclic code of length $n = n_1 \cdot n_2$ and defining set $S_n = S_{n_1} \bigcup S_{n_2}$ and $S_{n_1} \bigcap S_{n_2} = \emptyset$ and the two contractions are trivial codes; S_{n_1} and S_{n_2} are the defining sets for the two contractions of c_n of lengths n_1 and n_2 respectively. Then c_n has minimum distance equal to 4. Before presenting the formal proof, we are going to state some theorems without proof. However these results prove to be both informative and helpful in the derivation of our proof, the interested reader could refer to [2], [5], [6], [7] for formal proofs.

Theorem 4.1. (BCH Bound) If a defining set A for a cyclic code contains a consecutive set of length $\delta-1$, then $d_A \leqslant \delta$.

Theorem 4.2. (Hartmann-Tzeng Bound). If $A = \{\beta^{i_1}, \beta^{i_2}, \ldots, \beta^{i_\ell}\}$ is a defining set for a cyclic code and if β is a primitive n'th root of unity such that A contains the consecutive sets $\{\beta^{i+ja}, \beta^{i+1+ja}, \ldots, \beta^{i+\delta-2+ja}\}$, $0 \leqslant j \leqslant S$, and if $(\delta, n) = 1$ and $(a, n) = 1$ then $d_A \geqslant \delta + S$.

Theorem 4.3. (Roos bound). If A is a defining set for a cyclic code with minimum distance d_A and if B is a set of n'th roots of unity such that $|\overline{B}| \leqslant |B| + d_A - 2$, then the code with defining set AB has minimum distance $d \geqslant |B| + d_A - 1$ where $\overline{B}$ is a consecutive set containing B.

Theorem 4.4. (Wilson and VanLint) Let $n = \ell \cdot n_0$ Let d be the minimum distance of the binary cyclic code C of length n for which the defining set contains $\{\alpha^{\ell i_1}, \alpha^{\ell i_2}, \ldots, \alpha^{\ell i_k}\}$ (where α is a primitive n'th root of unity). Let d_0 be the minimum distance of the cyclic code C of length n_0 (contraction of c) with defining set $\{\xi^{i_1}, \xi^{i_2}, \ldots, \xi^{i_k}\}$, ($\xi = \alpha^{\ell}$) then d is even or $d \geq d_0$.

Proof. In theorem 3.1. we showed that $b = (b_0, b_1, \ldots, b_{n_0-1}) \in C$ where $b_j = c_j + c_{j+n_0} + c_{j+2n_0} + \ldots + c_{j+(\ell-1)n_0}$; $c \in C$ therefore, if $b_j = 0$ for $0 \leqslant j \leqslant n_0 - 1$ then c has even weight. Otherwise at least d_0 of the coordinates c_j are 1. □

To translate this result to be useful in our pro-

blem, consider the following: Since we are contracting the code c of length $n = n_1 \cdot n_2$ to two trivial codes c_{n_1} and c_{n_2} of lengths n_1 and n_2 respectively the minimum distance of c_{n_2} is n_2, we also showed that the odd weight vectors contract to all one vector of length n_1 and all one vector of length n_2 hence in the simultaneous contraction the minimum distance of the odd weight vectors is at least n_2 (suppose that $n_2 > n_1$). We also showed that the even weight vectors contract to all zero vectors of length n_1 and n_2. Therefore the minimum distance of the even weight codes is even and hence, the minimum distance of c_n is even or $d_{c_n} \geq n_2$. (Assuming $n_2 > n_1$).

If S_{n_1} and S_{n_2} are the defining sets that generate the two contractions of c_n and if $\alpha^{\ell+ic}$ is a set of consecutive roots in the defining set of c_n, for $0 \leq i \leq d_0 - 2$ and $(n, c) = 1$ then we have $d \geq d_0$, however since $(n_1, n_2) = 1$, d_0 must equal three otherwise $(n, c) \neq 1$ and hence BCH does not apply. Therefore according

to BCH bound we have $d \geqslant 3$, but theorem 4.4. implies that either $d \geqslant n_2$ of d is even therefore $d \geqslant 4$, and in fact d = 4 for all the cases that we considered.

□

IV. A DECODING ALGORITHM FOR CYCLIC CODES OF LENGTH $n=n_1.n_2.n_3$

In the previous sections we very briefly discussed the decoding algorithm for this class of binary cyclic codes of composite lengths. However we mentioned that this decoding algorithm has the advantage of being very simple, and furthermore that no arithmetic has to be carried out over finite field.

The algorithm can best be presented by considering an example.

Example 4.1. Let C be a binary cyclic code of length $n = 15 = (n_1 \cdot n_2) = (5 \times 3)$, let the defining set for this code be $S = \{\alpha^3, \alpha^5\}$, where α is a primitive 15'th root of unity. The contraction tables for this code are:

TABLE I

0	5	10
1	6	11
2	7	12
3	8	13
4	9	14

TABLE II

0	3	6	9	12
1	4	7	10	13
2	5	8	11	14

where the numbers correspond to digits of a code in C. Consider the following permutation of the elements of the rows of table II:

TABLE III

0	3	6	9	12
10	13	1	4	7
5	8	11	14	2

This permutation in fact combines the two tables into one, because the rows of table III are the rows of table II, and the columns of III are the rows of table I.

Furthermore, consider any column between any two consecutive element in any column (including the wrap around), there is a jump of 10 mod 15; for instance consider the third column, which is $(6 \quad 1 \quad 11)^T$ then we have:

$$1 = 6 + 10 \mod (15)$$
$$11 = 1 + 10 \mod (15)$$
$$6 = 11 + 10 \mod (15)$$

Also between consecutive elements of any row there is a jump of 3. Therefore if the decoder is intelligent, there is no need to save these tables.

At this point the decoding is only as complex as the

old multiplication tables that primary school students use to learn how to multiply.

Let us assume that an error changes the parity of the second row and third column . But the second row and third column meet on position 1 and hence that is the error.

Now let us assume that the error changes the parity of the first two rows and colums one and three. These two rows and columns meet on the following positions. (0, 1, 6, 10) but since the burst error correcting capability of this code is 2 plus all the non-solid bursts of length 3. Decoder knows that bits zero and one are the errors.

Let us consider another example to make the algorithm more clear.

Example 4.2. Let c be a binary cyclic code of length n = 35 with defining set S = $\{\alpha^5, \alpha^7, \alpha^{15}\}$, the contraction tables are:

TABLE I

0	7	14	21	28
1	8	15	22	29
2	9	16	23	30
3	10	17	24	31
4	11	18	25	32
5	12	19	26	33
6	13	20	27	34

TABLE II

0	5	10	15	20	25	30
1	6	11	16	21	26	31
2	7	12	17	22	27	32
3	8	13	18	23	28	33
4	9	14	19	24	29	34

to combine the two tables we must shift row two of II by 4 and row 3 of II by 8 and so on, then we have:

TABLE III

0	5	10	15	20	25	30
21	26	31	1	6	11	16
7	12	17	22	27	32	2
28	33	3	8	13	18	23
14	19	24	29	34	4	9

In this case the jumps between the consecutive elements of any column is 21. We will now show how the jumps between elements of rows and columns can be used for decoding the burst. Assume that a burst of length one occurs and

changes the parity of say third row and fourth column. The decoder labels the rows and columns differently, the first row is labeled as row zero, and the first column is labeled as column zero. Decoder also knows that row zero and column zero meet on position zero. Thus decoder sees that the parity of row two and column 3 has changed. Since the jump in columns is 21 then row two and column three must meet on $15 + (2 \times 21) \bmod (35) = (15 + 42)_{35} =$

$$(57)_{35} = 22$$

which is indeed the correct position. Therefore if the decoder has some intelligence it need not save the table in memory. However for codes of considerably short length, it might be advantageous to use a table look up procedure for decoding.

Now let us consider a burst of length 4 and sssume that the burst corrupts bits (0, 1, 3) hence the parity of row zero, row one and row three changes, along with that the parity of columns zero, two and three. Using table, look up or taking advantage of the jumps, we find out that the burst is hidden in the following sequence of the positions:

0, 21, 28, 10, 31, 3, 15, 1, 8

At this point decoder must sort the sequences which results in:

0, 1, 3, 8, 10, 15, 21, 28, 31

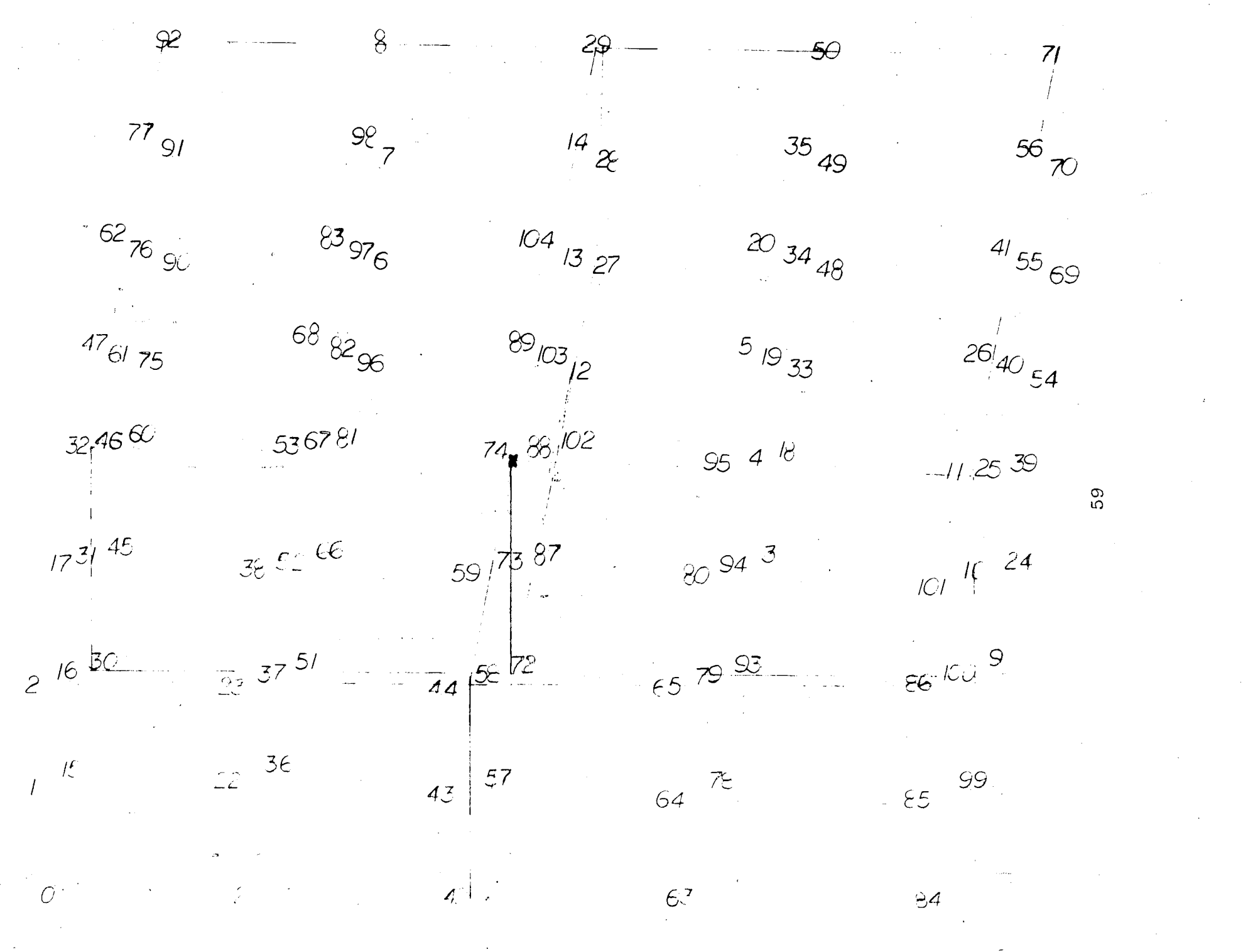
decoder knows that three bits are in error as can be seen from the total number of columns that changed parity, and furthermore the burst error capability of this code is 4. Hence a burst of larger length is not decoded as the error pattern. Thus, decoder must find a sequence of at most four consecutive numbers in the above sequence which represent the positions in a code word in C. The only consecutive subset of the above sequence and cardinality less than 4 is $(0, 1, 3)$ which is in fact the correct error pattern.

The speed of the decoder is only limited by the search method used. Hence by implementing an efficient search algorithm in software or hardware, the decoding time and speed increases substantially.

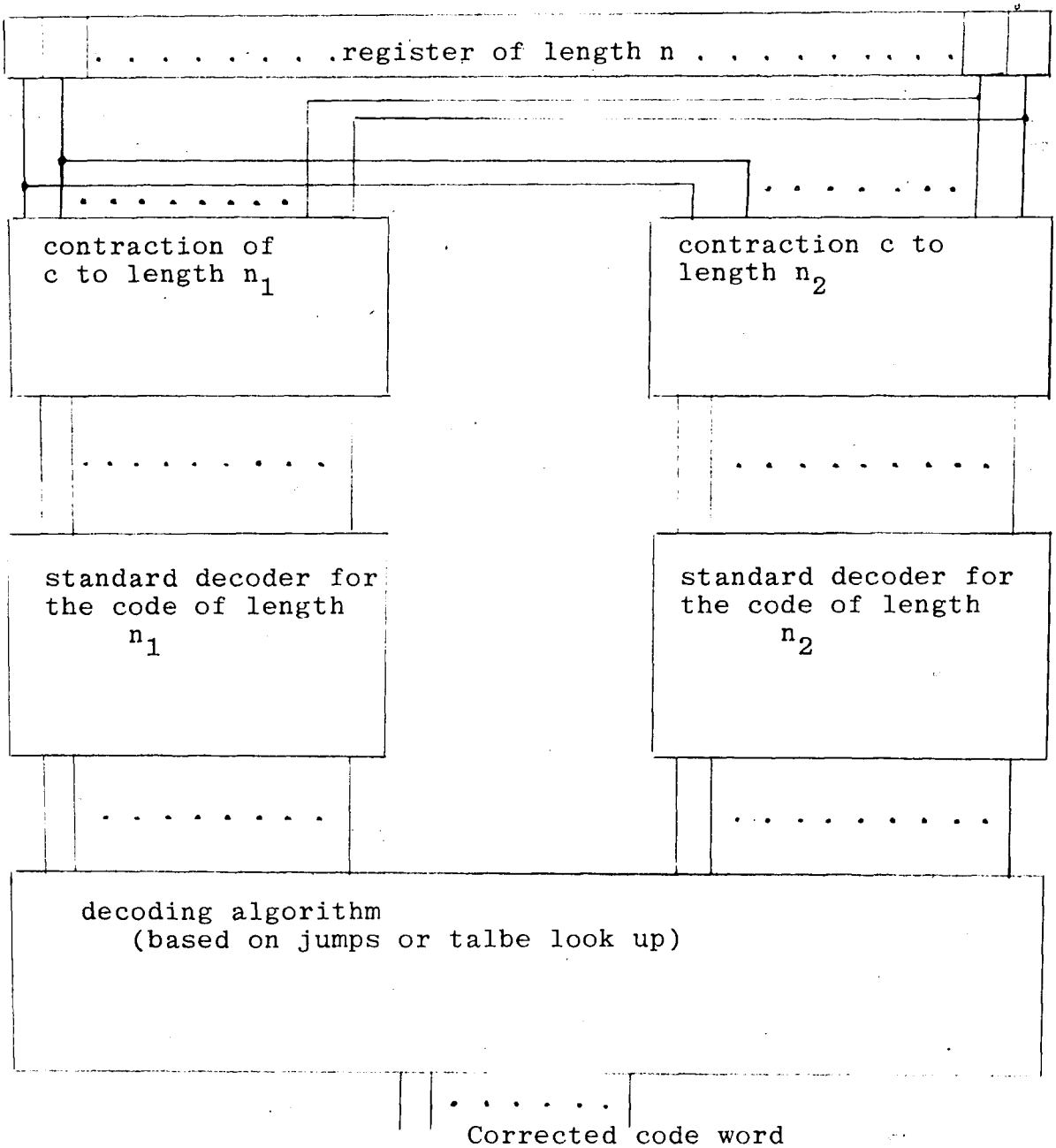
For three or more contractions, a similar procedure can be devised, for instance for the code of length 105 which contracts to trivial codes of length 3, 5, and 7, the contraction tables can be combined in a cube form. However, in the cube the error bits is recognized as the

intersection of three planes, which meet on one point, a position in this case.Therefore in the three contraction tables, each row is represented as a plane in the cube. Hence if the parity of some row changes, it will cause a change in the parity of a plane.

Furthermore, since each combination of 3 rows are orthogonal on one element, this implies that any three non-parallel planes intersect on one code position, thus the decoding can be carried out easily. Please refer to the contraction tables in example 3.6. and compare with the cube shown below, and notice how the rows of contraction tables are arranged as planes in the cube.


92
8
29
50
71
77 91
98 7
14 28
35 49
56 70
62 76 90
83 97 6
104 13 27
20 34 48
41 55 69
47 61 75
68 82 96
89 103 12
5 19 33
26 40 54
32 46 60
53 67 81
74 88 102
95 4 18
11 25 39
59
17 31 45
38 52 66
59 73 87
80 94 3
101 10 24
2 16 30
23 37 51
44 58 72
65 79 93
86 100 9
1 15
22 36
43 57
64 78
85 99
0
4
63
84

A block diagram for the decoder is shown below:



V. CONCLUSION

In this thesis, we considered the random and burst error correcting capability of a certain class of cyclic block codes.

The main work was based on our study of a subclass of cyclic codes, namely, the cyclic codes of composite length $n = n_1 \times n_2 \times \ldots$. Such that n_1, n_2, ... are pair-wise relatively prime. Based on simultaneous contractions of a code of composite length to its factors, we developed a new decoding technique. Furthermore, it was shown in this thesis that for correcting bursts, this technique is efficient because it meets the Reiger theoretical upper bound asymtotically, when applied to a certain subclass of codes of composite length. And gives considerably good results in other cases. We also considered the application of this decoding algorithm to random error-correction, and showed that the algorithm can easily be implemented in software and/or hardware.

In this work we mainly considered two simultaneous contractions of a code C. And occasionally we considered three contractions. Further research on this subject, in

our opinion must concentrate on error-correcting capability of codes that contract to several shorter codes. How to increase the efficiency of these codes, and what are the possible limitations of this procedure when applied to codes with multiple contractions.

REFERENCES

[1] E.R. Berlekamp, Algebraic Coding Theory, New York: McGraw-Hill, 1968.

[2] W.W. Peterson and E.J. Weldon, Jr., Error-Correcting Codes, 2d edition., Cambridge, Mass.: MIT Press, 1972.

[3] N. Abramson, "A Class of Symmetric Codes for Non-independent Errors," IRE Trans. Inf. Theory IT-4(4): 150-157, December, 1959.

[4] H.O. Burton, Some Asymtotically Optimal Burst-Correcting Codes and Their Relation to Single-Error-Correcting Reed-Solomon Codes, Bell Laboratories, 1968.

[5] J.H. VanLint and R.M. Wilson, "On the Minimum Distance of Cyclic Codes," pre-published paper.

[6] C. Roos, "A New Lower Bound for the Minimum Distance of a Cyclic Codes," IEEE Trans. Inform. Theory, IT-29, pp.330-332, May, 1983.

[7] C.R.P. Hartmann and K.K. Tzeng, "Generalizations of the BCH bound," Inform. Contr. 30:489-498, 1972.

[8] S. Lin and D.J. Costello, Error Control Coding: Fundamentals and Applications, Prentice-Hall, 1983.

[9] F.J. McWilliams and N.J.A. Sloan, The Theory of Error-Correcting Codes, Amsterdam: North-Holland, 1977.

[10] V.C. DaRocha, "Efficient Burst-Correcting Cyclic Codes," Electronics Letters, 19(2), 20 January, 1983.

VITA

Homayoun Shahri was born in Iran on November 13, 1960. He graduated from Alborz High School in 1977 in Iran. He attended Hofstra University from 1978 until 1980 and the State University of New York at Stony Brook from 1980 until 1982 from which he received with honors the B.E. degree in Electrical Engineering. Since 1982, he has attended Lehigh University, where he has been a Teaching Assistant in the department of Computer Science and Electrical Engineering. Homayoun Shahri is a member of IEEE.