THRESHOLD ANALYSIS WITH FAULT-TOLERANT OPERATIONS FOR NONBINARY QUANTUM ERROR CORRECTING CODES

A Thesis

by

APARNA KANUNGO

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2005

Major Subject: Computer Science

THRESHOLD ANALYSIS WITH FAULT-TOLERANT OPERATIONS FOR NONBINARY QUANTUM ERROR CORRECTING CODES

A Thesis

by

APARNA KANUNGO

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Andreas Klappenecker
Committee Members,	Rabi Mahapatra
	Suhail Zubairy
Head of Department,	Valerie Taylor

August 2005

Major Subject: Computer Science

ABSTRACT

Threshold Analysis with Fault-Tolerant Operations for Nonbinary Quantum Error Correcting Codes. (August 2005)

Aparna Kanungo, B.E., Anna University

Chair of Advisory Committee: Dr. Andreas Klappenecker

Quantum error correcting codes have been introduced to encode the data bits in extra redundant bits in order to accommodate errors and correct them. However, due to the delicate nature of the quantum states or faulty gate operations, there is a possibility of catastrophic spread of errors which might render the error correction techniques ineffective. Hence, in this thesis we concentrate on how various operations can be carried out fault-tolerantly so that the errors are not propagated in the same block. We prove universal fault-tolerance for nonbinary CSS codes. This thesis is focussed only on nonbinary quantum codes and all the results pertain to nonbinary codes.

Efficient error detection and correction techniques using fault-tolerant techniques can help as long as we ensure that the gate error probability is below a certain threshold. The calculation of this threshold is therefore important to see if quantum computations are realizable or not, even with fault-tolerant operations. We derive an expression to compute the gate error threshold for nonbinary quantum codes and test this result for different classes of codes, to get codes with best threshold results.

To My Family

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to Dr.Andreas Klappenecker for guiding me and motivating me to give my best. Without his encouragement and support, this work would not have been possible. I would like to thank my committee members, Dr. Rabi Mahapatra and Dr. Suhail Zubairy, for their willing assistance. I would also like to take this oppurtunity to thank Avanti Ulhas Ketkar, Santosh Kumar, Salah A Aly and Pradeep Kiran Sarvepalli for providing a lot of useful tips and feedback on my work. I also thank all those who took time to proofread my thesis and give valuable suggestions.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION TO NONBINARY QUANTUM ERROR CORRECTING CODES

A. Background

Modern day computers have grown from valve technology to VLSI logic and will soon reach a point where the processor performance can no longer be increased by size reduction. However, a school of thought was developed where the rules of quantum mechanics are used to the advantage of a new kind of computing called quantum computing. Quantum computer can basically perform all the operations that a classical computer can, and also is in some sense capable of carrying out computations in parallel. This leads to the synthesis of quantum algorithms which provide exponential speedup. However, quantum computing has its disadvantages too. The technology to implement a quantum computer is currently beyond reach, and we have to develop robust systems that are capable of detecting and correcting the errors introduced during computation.

Quantum computers have tremendous potential. However, in order to harness this potential to the maximum, we must ensure that the computations performed are error free. Quantum states are very delicate since they are usually a superposition of multiple classical states, and any measurement or interaction with the environment leads to the decoherence of the quantum state. Hence quantum error correcting codes have been introduced to protect the quantum states through redundancy addition. A lot of work has already been done on binary quantum error correcting codes. This thesis is concentrated on the nonbinary quantum error correcting codes and all the

---

The journal model is *International Journal of Foundations of Computer Science*

concepts discussed will pertain to the nonbinary codes. In order to understand the error correcting techniques, we start with the discussion of some basic notions.

1. Notations and Definitions

The unit of information in nonbinary quantum systems is known as a quantum digit or *qudit*. Hence, for a q-dimensional system, the quantum states can be represented through normalized vectors in a complex Hilbert space $\mathcal{H} = \mathbf{C^q}$. We choose a fixed orthonormal basis $\mathcal{B}$ of $\mathcal{H}$ and denote its elements by $|x\rangle$. When x is an element of a q-ary field $\mathcal{F}_q$, that is, $\mathcal{B} = \{|x\rangle \quad | \quad |x\rangle \in \mathcal{F}_q\}$ where q is a prime power, a general state of a qudit can be given by the linear combination of of these orthonormal basis states, that is,

$$|\psi\rangle = \sum_{i=0}^{q-1} \alpha_i |i\rangle, \qquad \text{where } \alpha_i \in \mathbf{C} \text{ and } \sum_{i=0}^{q-1} |\alpha_i|^2 = 1.$$

If we use a system with n qudits we obtain a quantum register whose canonical basis states are given by the tensor product of the basis states of these n single qudits. The computational basis of $\mathcal{C}^{q^n}$ is given by

$$|x_1\rangle \otimes |x_2\rangle \otimes \ldots \otimes |x_n\rangle = |x_1\rangle |x_2\rangle \ldots |x_n\rangle = |x_1, x_2, \ldots, x_n\rangle$$

where $x_k \in \mathcal{F}_q$, $1 \leq k \leq n$. Sometimes we abbreviate $|x_1\rangle |x_2\rangle \ldots |x_n\rangle$ by $|x\rangle$ where x is some vector in $\mathcal{F}_q^n$

A general state of a quantum register of length n is a normalized vector is given by

$$|\psi\rangle = \sum_{x \in \mathcal{F}_q^n} \alpha_x |x\rangle, \qquad \text{where } \alpha_x \in \mathbf{C} \text{ and } \sum_{x \in \mathcal{F}_q^n} |\alpha_x|^2 = 1.$$

2. Error Correcting Codes

Nonbinary quantum error correcting codes are defined over a finite field $\mathcal{F}_q$ of q elements. The number q is necessarily a power of a prime p, that is, $q = p^m$ for some integer $m \geq 1$. The trace function tr from $\mathcal{F}_{q^m}$ to $\mathcal{F}_q$ is defined by

$$tr(x) = \sum_{k=0}^{m-1} x^{q^k} \tag{1.1}$$

We have,

$$\begin{aligned} tr(x+y) &= tr(x) + tr(y), \\ tr(\alpha x) &= \alpha tr(x), \end{aligned}$$

for all $x, y \in \mathcal{F}_{p^m}, \alpha \in \mathcal{F}_p$. A code C is self-orthogonal for $*$, a sesquilinear form, if for any $a, b \in C$,

$$a * b = 0. \tag{1.2}$$

There are several notions of sesquilinear forms such as the standard euclidian form, the trace symplectic form, the hermitian form, and the trace alternating form. The dual of a code C can be defined as follows

$$C^{\perp} = \{\mathbf{v} : \mathbf{v} * \mathbf{a} = \mathbf{0} \text{ for } \forall \mathbf{a} \in \mathbf{C}\}. \tag{1.3}$$

3. Stabilizer Codes

Quantum error correction is based on the idea of encoding k qudits in n qudits so as to map $\mathbf{C^{q^k}}$ into $\mathbf{q^k}$ dimensional subspace of $\mathbf{C^{q^n}}$.

The error correcting properties of the code depend on the subspace rather than the mapping.These subspaces are chosen in such way that an error translates one subspace to another subspace perpendicular to it. It is this property which helps in

error detection.

A stabilizer code $\mathbf{Q}$ is a non-zero subspace of $\mathbf{C^{q^n}}$ that satisfies

$$\mathbf{Q} = \bigcap_{\mathbf{E} \in \mathbf{S}} \{\mathbf{v} \in \mathbf{C^{q^n}} | \mathbf{E}\mathbf{v} = \mathbf{v}\} \tag{1.4}$$

for some subgroup $\mathbf{S}$ of $\mathbf{G_n}$. The coding space is described in terms of the error operators rather the codewords itself, that is, the code space is composed of those states which are fixed by the error operators in the stabilizer. In other words, the elements in the stabilizer do not have any effect on the codewords, that is, $E|\psi\rangle = |\psi\rangle$ for $|\psi\rangle \in S$. For example, if the quantum system is in the state $|\psi\rangle = \alpha|000\rangle + \beta|111\rangle$, then it remains in the same state even after applying $Z \otimes Z \otimes I$ or $Z \otimes I \otimes Z$. Hence the operators $Z \otimes Z \otimes I$ and $Z \otimes I \otimes Z$ are said to stabilize the state $|\psi\rangle$, that is, $|\psi\rangle = Z \otimes Z \otimes I|\psi\rangle = Z \otimes I \otimes Z|\psi\rangle$ If an error operator E is not a multiple of an element in the stabilizer S, then

$$EM = \lambda\ ME$$

for some element M of the stabilizer. If $\lambda \neq 1$ then we can detect the error E.

The set of operators that commute with all the operators in the stabilizer constitute the centralizer, $C(S) = \{A \mid AM = MA \text{ for all } M \in S\}$ of the stabilizer code. The elements in the centralizer transform one codeword into another. The operations which are valid for any stabilizer code should take one codeword to another, and hence must be within the centralizer of the stabilizer code.

B. Motivation

A lot of work has been done on binary quantum error correcting codes. However, the nonbinary equivalent still remains to be explored. Chapter II has been devised to understand the basics of nonbinary error correction techniques and the operations involved on these nonbinary codes. Because of the easy error propagation in quantum circuits, we not only have to carry out the operations with accuracy, but we also have to ensure that the error correction techniques do not induce more errors into the circuit in addition to the existing ones. Such a thing would have a multiplicative effect and completely nullify the effect of error correction. The propagation of errors is prevented by reducing the interaction between the *'qudits'* in the same block, that is, by carrying out the computations bitwise in a fault-tolerant way. The conditions and requirements to be satisfied for allowing such fault-tolerance are explained in chapter III. In spite of the fault-tolerant operations, the gate errors or the general noise may lead to change of states during computations. Hence, we need to find the minimum threshold for the success of a computation. The threshold theorem says that as long as the error probability of each gate is below a certain threshold, the quantum error correction will be successful. The details of threshold calculation and analysis is presented in chapter IV. Comparison of various codes to find the most efficient code with minimum threshold is given in chapter V.

CHAPTER II

QUDIT GATES

Non-binary quantum codes are a good alternative to the binary counterparts because of the existence of a better bound entanglement and our ability to construct better codes. In this chapter, we study the most common gates that can be used on nonbinary quantum codes and the basics of encoded operations.

A. Elementary Gates

Some of the most commonly used operations can be described through the following quantum gates. Let q be a power of a prime p, $q = p^m$ with $m \geq 1$. Let ω denote the p-th root of unity. If $\alpha \in \mathcal{F}_{\mathbf{q}}$, then the trace function $tr(\alpha)$ is defined as $\mathrm{tr}(\alpha) := \sum_{i=0}^{m-1} \alpha^{p^i} \in \mathcal{F}_{\mathbf{p}}$. Based on these, the most common elementary operations that can be performed on the non-binary codewords can be defined as follows

$$X_\alpha := \sum_{x \in \mathcal{F}_q} |x + \alpha\rangle \langle x| \quad for \quad \alpha \in \mathcal{F}_q \tag{2.1}$$

X_α operation represents the addition of a fixed element $\alpha \in \mathcal{F}_q$. In other words, it finds the addition modulo q in the field $\mathcal{F}_q$. In the binary case X corresponds to the Pauli σ_x operator. A generalization of the Pauli σ_z operator is given by

$$Z_\beta := \sum_{z \in \mathcal{F}_q} \omega^{\mathrm{tr}(\beta z)} |z\rangle \langle z| \text{ for } \beta \in \mathcal{F}_q. \tag{2.2}$$

We also introduce the operator M_γ given by

$$M_\gamma := \sum_{y \in \mathcal{F}_q} |\gamma y\rangle \langle y| \quad for \quad \gamma \in \mathcal{F}_q \setminus \{0\}. \tag{2.3}$$

Applying M_γ is equivalent to multiplying by a fixed element $\gamma \neq 0$.

We have a two-qudit gate to swap the values of the two digits involved in the gate operation. This can be shown as

$$SW_{x,y} := \sum_{x,y \in \mathcal{F}_q} |x\rangle |y\rangle \langle y| \langle x| . \tag{2.4}$$

Another useful gate operation for the non-binary codes is Fourier transform

$$\mathrm{FT} := \frac{1}{\sqrt{q}} \sum_{x,z \in \mathcal{F}_q} \omega^{\mathrm{tr}(xz)} |z\rangle \langle x| \tag{2.5}$$

This represents the quantum Fourier transform which transforms the state $|0\rangle$ to a superposition of all the basis states with equal amplitudes, i.e.,

$$\mathrm{FT} |0\rangle = \frac{1}{\sqrt{q}} \sum_{\alpha \in \mathcal{F}_q} |\alpha\rangle .$$

Fourier transform is non-binary equivalent of Hadamard transform.

Controlled-addition operation is another two-qudit gate, where the the first one acts as the control and the second one acts as the target for addition of the two elements. This operation is equivalent to the controlled-not operation in the binary quantum codes.

$$\mathrm{C-X}^{(1,2)} := \sum_{x,y \in \mathcal{F}_q} |x\rangle_1 |x+y\rangle_2 \langle x|_1 \langle y|_2 \tag{2.6}$$

Double-controlled-addition operation is a three-qudit operator which adds the product of the first and second qudits to the third one based on the first two qudits acting as the control bits. This is analogous to the Toffoli gate in the binary quantum codes and can be used for universal reversible gate over $\mathcal{F}_q$

$$\mathrm{C-C-X}^{(1,2,3)} := \sum_{a,x,b \in \mathcal{F}_q} |a\rangle_1 |x\rangle_2 |ax+b\rangle_3 \langle a|_1 \langle x|_2 \langle b|_3 \tag{2.7}$$

B. Relevance to Nonbinary Quantum Stabilizer Codes

The error basis of a qudit system can be given by the following set of unitary operators

$$\mathcal{E} = \{\mathbf{X}_\alpha \mathbf{Z}_\beta \colon \alpha, \beta \in \mathcal{F}_\mathbf{q}\}. \tag{2.8}$$

These q^2 operators form an orthogonal basis with respect to the inner product $\langle A, B\rangle = \mathrm{tr}(A^\dagger B)$. They also generate an error group $\mathcal{G}$ of size pq^2 whose elements can be uniquely written as $\omega^\gamma X_\alpha Z_\beta$ where $\gamma \in \{0, \ldots, p-1\}$ and $\alpha, \beta \in \mathcal{F}_q$. The commutation relation between any two elements of this group is given by

$$(X_\alpha Z_\beta)(X_{\alpha'} Z_{\beta'}) = \omega^{\mathrm{tr}(\alpha'\beta - \alpha\beta')}(X_{\alpha'} Z_{\beta'})(X_\alpha Z_\beta). \tag{2.9}$$

For the n qudit system, the error basis is just an n-fold tensor product given by $\mathcal{E}^{\otimes n}$ and $\mathcal{G}_\mathbf{n} := \mathcal{G}^{\otimes \mathbf{n}}$. Any element E of the error group $\mathcal{G}_\mathbf{n}$ can uniquely be written as

$$E = \omega^\gamma (X_{\alpha_1} Z_{\beta_1}) \otimes (X_{\alpha_2} Z_{\beta_2}) \otimes \ldots \otimes (X_{\alpha_n} Z_{\beta_n}) =: \omega^\gamma X_\alpha Z_\beta,$$

where $\gamma \in \{0, \ldots, p-1\}$ and $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n), \beta = (\beta_1, \beta_2, \ldots, \beta_n) \in \mathcal{F}_q^n$. The weight of an element $X_\alpha Z_\beta$ is equal to the number of non-zero components of α_i and β_i. From the commutation relation 2.9, we get, for $(\alpha, \beta), (\alpha', \beta') \in \mathcal{F}_q^n \times \mathcal{F}_q^n$

$$(X_\alpha Z_\beta)(X_{\alpha'} Z_{\beta'}) = \omega^{(\alpha,\beta)*(\alpha',\beta')}(X_{\alpha'} Z_{\beta'})(X_\alpha Z_\beta)$$

where the inner product $*$ is defined by

$$(\alpha, \beta) * (\alpha', \beta') := \sum_{i=1}^{n} \mathrm{tr}(\alpha_i' \beta_i - \alpha_i \beta_i'). \tag{2.10}$$

Nonbinary stabilizer codes are based on the idea that there exists an Abelian subgroup $\mathcal{S}$ such that its intersection with the center of $\mathcal{G}_\mathbf{n}$ is trivial. The stabilizer code $\mathcal{C}$ is defined as the common eigenspace of the operators in $\mathcal{S}$.

C. Encoding and Decoding Stabilizer Codes

1. Encoding Basic Operations

Encoding a stabilizer code refers to the process of storing k qudits of information in n qudits where $n > k$. However, to carry out various operations it would be unwise to decode, apply the gate operation and encode again. For this reason, we aim to find encoded operations that would be equivalent to applying the original operations on the unencoded bits.

If $\{g_1, g_2, \ldots, g_{n-k}\}$ where $g_i = \omega^{\gamma_i} X_{\alpha_i} Z_{\beta_i}$ with $\gamma_i \in \{0, \ldots, p-1\}$ and $(\alpha_i, \beta_i) \in \mathcal{F}_q^n \times \mathcal{F}_q^n$ be a minimal set of generators for $\mathcal{S}$, then the stabilizer matrix corresponding to the stabilizer code $\mathcal{C}$ is a generator matrix of a (classical) linear code $C \subseteq \mathcal{F}_q^n \times \mathcal{F}_q^n$. This matrix can be represented in the form

$$\left(\begin{array}{c|c} \alpha_1 & \beta_1 \\ \alpha_2 & \beta_2 \\ \vdots & \vdots \\ \alpha_{n-k} & \beta_{n-k} \end{array}\right) \in \mathcal{F}_q^{(n-k)\times 2n} \tag{2.11}$$

The generators of the stabilizer are represented by the the rows of the stabilizer matrix, that is, each $\omega^{\gamma_i} X_{\alpha_i} Z_{\beta_i}$ is represented by the row $(\alpha_i|\beta_i)$. We can perform a set of elementary row and column operations on a matrix to convert it to an equivalent form. (2.11) when converted into an equivalent form still represents the stabilizer of the code since, interchanging the rows does not change the generator elements and interchanging the columns just interchanges the positions of the qudits in the matrix. The other elementary operations are also valid as long as the resultant element is part

of the group generated by these generators elements. Hence, after a set of row and column operations (Gaussian reductions), the matrix can be reduced to the standard form as illustrated in [14]. Adding one row of the generator matrix with another row is equivalent to multiplying the two generators corresponding to the two rows and belongs to the group. Multiplying a row by a constant in $\mathcal{F}_q$ gives an element of the group as long as we consider an $\mathcal{F}_q$-linear code. Hence, we can multiply the rows with $\alpha \in \mathcal{F}_q$ such that the final addition of the rows when carried out in modulo q arithmetic gives us the required 0 or 1 at appropriate places. Some of the reduction steps can be shown as follows:

$$\begin{array}{r} \\ r\{ \\ n-k-r\{ \end{array} \left(\begin{array}{cc|cc} \overbrace{I}^{r} & \overbrace{A}^{n-r} & \overbrace{B}^{r} & \overbrace{C}^{n-r} \\ 0 & 0 & D & E \end{array} \right). \tag{2.12}$$

Now carrying out similar gaussian reductions on the Z part, the matrix can be further reduced to the form

$$\begin{array}{r} \\ r\{ \\ n-k-r-s\{ \\ s\{ \end{array} \left(\begin{array}{ccc|ccc} \overbrace{I}^{r} & \overbrace{A_1}^{n-k-r-s} & \overbrace{A_2}^{k+s} & \overbrace{B}^{r} & \overbrace{C_1}^{n-k-r-s} & \overbrace{C_2}^{k+s} \\ 0 & 0 & 0 & D_1 & I & E_2 \\ 0 & 0 & 0 & D_2 & 0 & 0 \end{array} \right). \tag{2.13}$$

Since we require the first r generators to commute with the last s generators, we conclude that $s = 0$. Hence, we can always bring any code to the following standard form.

$$\begin{array}{r} \\ r\{ \\ n-k-r\{ \end{array} \left(\begin{array}{ccc|ccc} \overbrace{I}^{r} & \overbrace{A_1}^{n-k-r} & \overbrace{A_2}^{k} & \overbrace{B}^{r} & \overbrace{C_1}^{n-k-r} & \overbrace{C_2}^{k} \\ 0 & 0 & 0 & D & I & E \end{array} \right) \tag{2.14}$$

The generators of the stabilizer of a particular code, expressed in the standard form can be used to derive the $\overline{X_\alpha}$ and $\overline{Z_\alpha}$ operators in the following way - we assume that the k data bits form the last part of the n encoded bits, with the first n-k bits fixed as zeros. In this case, the X operator applied to each of the k data qudits result in an I in the X part of the $\overline{X_\alpha}$ operator and 0 in the Z part of the $\overline{X_\alpha}$ operator. The procedure for derivation of the general formula for $\overline{X}$ and $\overline{Z}$ can be explained with the following example.

Example : A general formula formula for $\overline{X}$ and $\overline{Z}$ for the binary case ($q = 2$), can be derived in a systematic way from the stabilizer matrix (represented in the standard form as shown in 2.14). If we consider an encoded operation to be of the form $[u_1, u_2, u_3|v_1, v_2, v_3]$, we can set $u_1 = I$ and $v_3 = 0$ when this encoded operation corresponds to $\overline{X}$. Using the fact that $\overline{X}$ commutes with the elements of the stabilizer, we get

$$D.u_1^T + u_2^T + E = 0$$

In order to satisfy this equation, we take $u_1 = 0$ and $u_2 = E^T$. Similarly checking the commutation of this encoded operation with the first part of the stabilizer matrix 2.14, we get

$$I.v_1^T + A_1.v_2^T + B.u_1^T + C = 0$$
$$\Rightarrow v_1^T + A_1.v_2^T + C = 0$$

We can substitute $v_1 = C^T$ and $v_2 = 0$ to satisfy the above equation. Hence the $\overline{X}$ operator can be given as $[0E^T I|C^T 00]$. The $\overline{Z}$ operator can be derived in a similar

fashion using the commutativity property with the generators of the stabilizer, as $[000|A_2^T 0 I]$.

Lemma 1. *$\overline{X_\alpha}$ and $\overline{Z_\alpha}$ operations can be carried out for any stabilizer code*

Proof. Any $\mathcal{F}_q$-linear stabilizer of any code can be transformed into the standard form by a series of gaussian eliminations carried out on both rows and columns. Since the $\overline{X_\alpha}$ and $\overline{Z_\alpha}$ are directly obtained from the standard form of the stabilizer, they can be defined for any stabilizer. The verification that these operators actually carry out the logical X and logical Z operations on the encoded bits can be concluded from the method of construction of these operators. That is, the existence of identity in the X sector of the last k bits indicates that X is applied to the k^{th} bit and this operator commutes with the generators in the stabilizer, giving the encoded X operator. The same argument follows for the $\overline{Z_\alpha}$ operator. □

2. Other Encoded Operations

Apart from the basic $\overline{X_\alpha}$ and $\overline{Z_\alpha}$ operators, we are interested in finding other encoded operators in the centralizer which commute with the stabilizer and hence can be used during error correction. However, these operators can not be directly represented in terms of X and Z in the form of a matrix. Hence the standard form of the stabilizer cannot be used directly for the derivation of these operators. To derive these operators we have to use other properties of these operators. For example, if we want to find the encoded Fourier transformation operation, we need to satisfy the property $\overline{FT_\alpha}\ \overline{X_\alpha}\ \overline{FT_\alpha} = \overline{Z_\alpha}$ since $FT_\alpha\ X_\alpha\ FT_\alpha = Z_\alpha$. Such properties help us derive a fault-tolerant implementation of encoded Fourier transformation operation since we test its effect on input gates present at each qudit and alter the stabilizer. An encoded operation thus derived is considered a valid one if it leads to no change in the stabilizer

or just a permutation which is equivalent to the original stabilizer. A more systematic way to get such encoded operation is explained in the following chapter.

CHAPTER III

FAULT-TOLERANT QUANTUM ERROR CORRECTION

A. Introduction to Fault-Tolerant Computation

Quantum error correction has stirred great hopes in developing a scalable quantum computer for real. However, just the existence of good and efficient quantum codes is not enough. Due to the delicate nature of quantum states, there is a possibility of catastrophic spread of the existing errors which might render the error correction techniques ineffective. For instance, during a controlled addition operation, the phase error might be propagated from the target bit to the control qudit leading to two errors from one existing error. Hence, it is necessary to perform these error correction procedures in a fault-tolerant manner. A device that works effectively even when its elementary components are imperfect is said to be *fault-tolerant* [15].

1. Transversal Operations

Transversal operations are operations in which the gate acting on the original set of qudits is implemented digit-wise on each of the encoded qudits. For instance, a single qudit operation that has a transversal implementation is realised by single qudit operation on each of the qudits. The advantage of transversal computations is that the errors can propagate between different encoded blocks instead of the same block and hence can be corrected more easily. For example, bitwise controlled-not is a fault-tolerant operation because it acts from one block to another leading to the possibility of spread of errors only between corresponding qudits in different blocks. This is illustrated by an example in Figure 1 which shows the controlled-not operation implemented transversally on a $[[5, 1, 3]]_2$ code. However, not all bitwise operations

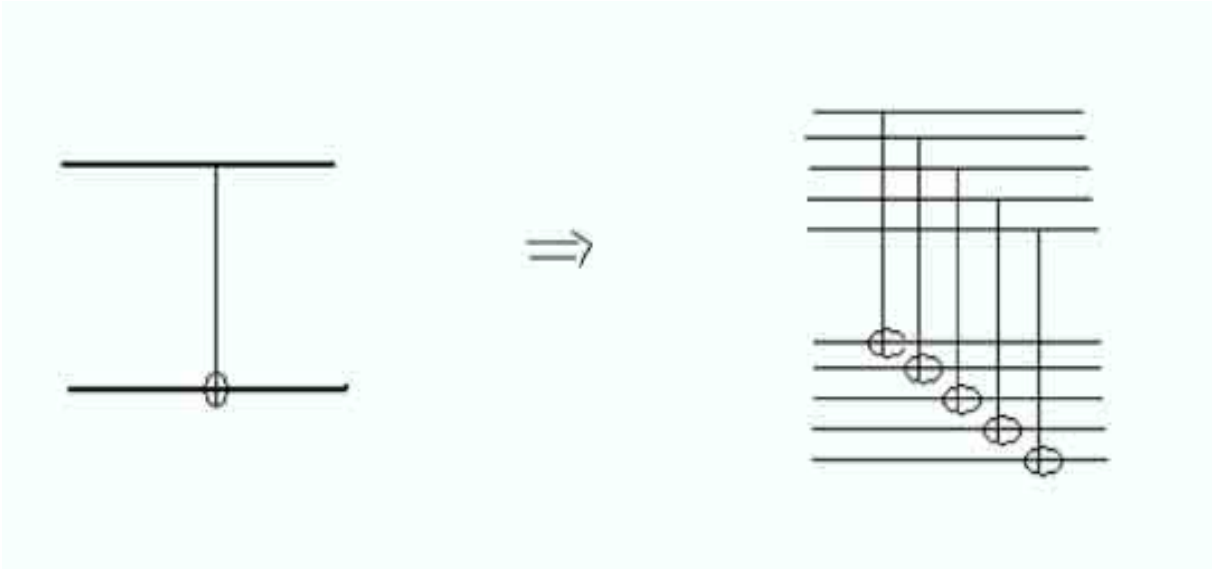

Fig. 1. CNOT implemented transversally on $[[5, 1, 3]]_2$ code

map one codeword to another. Therefore, it is necessary that the bitwise operations leave the stabilizer unchanged or just rearrange the code so that the operation takes a codeword to another codeword.

2. Conjugation of Operators

Quantum circuits consist of many gate operations. Ideally we would like to perform these operations on the encoded bits without having to decode them. But we have to decide the operations that can be performed on these codewords. A valid operation should leave the stabilizer unchanged or just rearrange the code so that the operation takes codewords to other valid codewords. The operators X_α, Y_α and Z_α can be performed bitwise on the encoded state because they commute with the stabilizer and hence leave the code space unaltered. However an arbitrary unitary transformation U, may alter the code space in which case it is not considered to be a valid operation. In order to perform an operation on the codewords, it is necessary to study the effect

of these operators on the elements of stabilizer, S, and the centralizer, C(S). A unitary transformation U is considered a valid operation if the following equation is satisfied

$$U|\psi\rangle = UM|\psi\rangle = (UMU^{\dagger})U|\psi\rangle \tag{3.1}$$

where $|\psi\rangle$ is the input state, and M is an element of the stabilizer. By applying the operator U to $|\psi\rangle$, we transform the operator M into $UMU^{\dagger}$. Hence, for $|\psi\rangle$ to remain a codeword U$|\psi\rangle$ must still be in the code space and $UMU^{\dagger}$ must fix all the codewords $|\psi\rangle$. Hence, to check whether a particular operation U can be carried out fault-tolerantly or not, we need to check that we get an element of the stabilizer when we conjugate the gate operation with X_{α} and Z_{α} operators.

B. Calderbank-Shor-Steane (CSS) Codes

Calderbank-Shor-Steane (CSS) code [14] is a class of code that is derived from two classical linear codes $C_1[n, k_1]$ and $C_2[n, k_2]$ such that $C_2 \subset C_1$. The quantum code thus generated from these two classical codes is represented as $CSS(C_1/C_2)$ (this is read as CSS of C_1 over C_2) which is an $[[n, k_1 - k_2]]$ code encoding $k_1 - k_2$ qudits into n qudits. However, for the X sector to be exactly same as the Z sector, the two classical codes have to be identical, and the quantum code must be derived from the classical code that contains its dual. A punctured self-dual CSS code satisfies these properties.

Example: Consider the Hamming $[7, 4, 3]_2$ code, C. The dual $C^{\perp}$ of this code is a $[7, 3]$ code such that $C^{\perp} \subseteq C$. Through the above CSS code construction, we get $[[n, 2k - n]] = [[7, 1]]$ quantum code that can correct errors on a single logical qubit. The stabilizer of the $[[7, 1, 3]]_2$ quantum code is given in Table I.

Table I. The Seven-Qubit CSS Code

M_1	σ_x	σ_x	σ_x	σ_x	I	I	I
M_2	σ_x	σ_x	I	I	σ_x	σ_x	I
M_3	σ_x	I	σ_x	I	σ_x	I	σ_x
M_4	σ_z	σ_z	σ_z	σ_z	I	I	I
M_5	σ_z	σ_z	I	I	σ_z	σ_z	I
M_6	σ_z	I	σ_z	I	σ_z	I	σ_z
$\overline{X}$	I	I	I	I	σ_x	σ_x	σ_x
$\overline{Z}$	I	I	I	I	σ_z	σ_z	σ_z

C. Fault-Tolerant Operations on CSS codes

1. Conjugation of Fourier Transform Operation

The action of the Fourier Transform operation on X_α is given as follows

$$\begin{aligned}
\mathrm{FT}^{-1} X_\alpha \mathrm{FT} &= \frac{1}{\sqrt{q}} \sum_{i,j\in\mathbf{F}_q} \omega^{-\operatorname{tr}(ij)} |i\rangle \langle j| \sum_{x\in\mathbf{F}_q} |x+\alpha\rangle \langle x| \frac{1}{\sqrt{q}} \sum_{k,l\in\mathbf{F}_q} \omega^{\operatorname{tr}(kl)} |k\rangle \langle l| \\
&= \frac{1}{q} \sum_{i,l\in\mathbf{F}_q} \sum_{x\in\mathbf{F}_q} \omega^{-\operatorname{tr}(i(x+\alpha))} \omega^{\operatorname{tr}(xl)} |i\rangle \langle l| \\
&= \frac{1}{q} \sum_{i,l\in\mathbf{F}_q} \omega^{\operatorname{tr}(-i\alpha)} \sum_{x\in\mathbf{F}_q} \omega^{\operatorname{tr}(x(l-i))} |i\rangle \langle l| \\
&= \sum_{i\in\mathbf{F}_q} \omega^{\operatorname{tr}(-i\alpha)} |i\rangle \langle i| = Z_{-\alpha}
\end{aligned}$$

Similarly, its action on Z_β can be given by

$$\begin{aligned}
\mathrm{FT}^{-1} Z_\beta \mathrm{FT} &= \frac{1}{\sqrt{q}} \sum_{i,j \in \mathbf{F}_q} \omega^{-\operatorname{tr}(ij)} |i\rangle \langle j| \sum_{z \in \mathbf{F}_q} \omega^{\operatorname{tr}(\beta z)} |z\rangle \langle z| \frac{1}{\sqrt{q}} \sum_{k,l \in \mathbf{F}_q} \omega^{\operatorname{tr}(kl)} |k\rangle \langle l| \\
&= \frac{1}{q} \sum_{i,l \in \mathbf{F}_q} \sum_{j \in \mathbf{F}_q}^{d-1} \omega^{-\operatorname{tr}(ij)} \omega^{\operatorname{tr}(\beta j)} \omega^{\operatorname{tr}(jl)} |i\rangle \langle l| \\
&= \frac{1}{q} \sum_{i,l \in \mathbf{F}_q} \sum_{j \in \mathbf{F}_q} \omega^{\operatorname{tr}((l-i+\beta)j)} |i\rangle \langle l| \\
&= \sum_{l \in \mathbf{F}_q} |l+\beta\rangle \langle l| = X_\beta.
\end{aligned}$$

Lemma 2. *Fourier Transformation can be performed fault-tolerantly on CSS codes which contain their dual.*

Proof. The CSS codes which contain their dual have the X and Z elements occurring exactly at the same position [14]. Hence, if X_α belongs to the stabilizer, the Z_α will also belong to the stabilizer. Likewise if Z_β belongs to the stabilizer, then X_α will also belong to the stabilizer. Another useful fact that can be used is that, since the stabilizer forms a group, the inverse of every element α, i.e. $-\alpha$ will also be in the group. Hence, $X_{-\alpha}$ and $Z_{-\alpha}$ is also an element of the stabilizer. Using these results, we can conclude that application of the fourier transformation operation switches the X_α to $Z_{-\alpha}$ and Z_β to X_β. Since these new elements $Z_{-\alpha}$ and X_β also belong to the stabilizer, we find that the stabilizer remains unchanged after the application of the Fourier transformation operation. Hence we conclude that Fourier transformation can be performed fault-tolerantly on nonbinary CSS codes which contain their own dual. □

2. Conjugation of Multiplication Operation

The effect of M_γ on the elements of the stabilizer can be summarized as follows

$$
\begin{aligned}
M_\gamma^{-1} X_\alpha Z_\beta M_\gamma &= \sum_{y\in\mathbf{F}_q} |y\rangle\langle\gamma y| \sum_{x\in\mathbf{F}_q} |x+\alpha\rangle\langle x| \sum_{z\in\mathbf{F}_q} \omega^{\mathrm{tr}(\beta z)} |z\rangle\langle z| \sum_{v\in\mathbf{F}_q} |\gamma v\rangle\langle v| \\
&= \sum_{y\in\mathbf{F}_q} |y\rangle\langle\gamma y| \sum_{x\in\mathbf{F}_q} |x+\alpha\rangle\langle x| \sum_{v\in\mathbf{F}_q} \omega^{\mathrm{tr}(\beta\gamma v)} |\gamma v\rangle\langle v| \\
&= \sum_{y\in\mathbf{F}_q} \left|\gamma^{-1}y\right\rangle\langle y| \sum_{v\in\mathbf{F}_q} \omega^{\mathrm{tr}(\beta\gamma v)} |\gamma v+\alpha\rangle\langle v| \\
&= \sum_{v\in\mathbf{F}_q} \omega^{\mathrm{tr}(\beta\gamma v)} \left|v+\gamma^{-1}\alpha\right\rangle\langle v| \\
&= \sum_{x\in\mathbf{F}_q} \left|x+\gamma^{-1}\alpha\right\rangle\langle x| \sum_{z\in\mathbf{F}_q} \omega^{\mathrm{tr}(\beta\gamma z)} |z\rangle\langle z| \\
&= X_{\gamma^{-1}\alpha} Z_{\gamma\beta}.
\end{aligned}
$$

Hence, M_γ acts on (α,β) as $\overline{M}_\gamma := \begin{pmatrix} \gamma^{-1} & 0 \\ 0 & \gamma \end{pmatrix}$.

Lemma 3. *Multiplication operation, M_γ, can be performed fault-tolerantly on any CSS code.*

Proof. M_γ, the multiplication operation, has the effect of switching X_α with $X_{\gamma^{-1}\alpha}$ and Z_β with $Z_{\gamma\beta}$. Since X_α and Z_β are elements of the stabilizer, $X_{\gamma^{-1}\alpha}$ and $Z_{\gamma\beta}$ also belong to the stabilizer, since multiplication with a constant phase throughout a since element of the stabilizer does not change the stabilizer. Hence, Multiplication operation can also be performed fault-tolerantly on any CSS code. □

3. Conjugation of Controlled-Addition Operation

The $\mathrm{C}-\mathrm{X}^{(1,2)}$ operation involves two blocks, hence, the transformations for both the blocks combined must be considered. The action of $\mathrm{C}-\mathrm{X}^{(1,2)}$ can be tested on the basic operations of the type $X_{\alpha_1}\otimes Z_{\beta_2}$ and $X_{\alpha_1}\otimes Z_{\beta_2}$. Similarly, the results of the $\mathrm{C}-\mathrm{X}^{(1,2)}$ operation on all basic operations can be summarized as follows. The amplitudes are copied forward and phases are copied backward.

$$(\mathrm{C}-\mathrm{X}^{(1,2)})^{-1}(X_{\alpha_1}\otimes Z_{\beta_2})\mathrm{C}-\mathrm{X}^{(1,2)}$$

$$\begin{aligned}
&= (\mathrm{C}-\mathrm{X}^{(1,2)})^{-1}\sum_{v,w\in\mathbf{F}_q}\omega^{\mathrm{tr}(\beta_2 w)}\left|v+\alpha_1\right\rangle_1\left|w\right\rangle_2\left\langle v\right|_1\left\langle w\right|_2\sum_{x,y\in\mathbf{F}_q}\left|x\right\rangle_1\left|x+y\right\rangle_2\left\langle x\right|_1\left\langle y\right|_2\\
&= (\mathrm{C}-\mathrm{X}^{(1,2)})^{-1}\sum_{x,y\in\mathbf{F}_q}\omega^{\mathrm{tr}(\beta_2(x+y))}\left|x+\alpha_1\right\rangle_1\left|x+y\right\rangle_2\left\langle x\right|_1\left\langle y\right|_2\\
&= \sum_{v,w\in\mathbf{F}_q}\left|v\right\rangle_1\left|w\right\rangle_2\left\langle v\right|_1\left\langle v+w\right|_2\sum_{x,y\in\mathbf{F}_q}\omega^{\mathrm{tr}(\beta_2(x+y))}\left|x+\alpha_1\right\rangle_1\left|x+y\right\rangle_2\left\langle x\right|_1\left\langle y\right|_2\\
&= \sum_{x,y\in\mathbf{F}_q}\omega^{\mathrm{tr}(\beta_2 x)}\omega^{\mathrm{tr}(\beta_2 y)}\left|x+\alpha_1\right\rangle_1\left|y-\alpha_1\right\rangle_2\left\langle x\right|_1\left\langle y\right|_2\\
&= (X_{\alpha_1}Z_{\beta_2})\otimes(X_{-\alpha_1}Z_{\beta_2}).
\end{aligned}$$

Similarly,

$$(\mathrm{C}-\mathrm{X}^{(1,2)})^{-1}(Z_{\beta_1}\otimes X_{\alpha_2})\mathrm{C}-\mathrm{X}^{(1,2)} = Z_{\beta_1}\otimes X_{\alpha_2}$$

Lemma 4. $\mathrm{C}-\mathrm{X}^{(1,2)}$ *operation can be performed fault-tolerantly on any CSS code.*

Proof. Multi-qubit operations like CNOT can be performed transversally using two blocks. To show this operation can be done for any CSS code, we see its effect on $M\otimes I$ and $I\otimes M$, where M is either X or Z. Since conjugating $Z_{\beta_1}\otimes X_{\alpha_2}$ with $\mathrm{C}-\mathrm{X}^{(1,2)}$ does not produce any change in the stabilizer,$\mathrm{C}-\mathrm{X}^{(1,2)}$ is a valid operation. Also, since $(X_{-\alpha_1}Z_{\beta_2})$ is also an element of the stabilizer, conjugating $X_{\alpha_1}\otimes Z_{\beta_2}$ produces an element of S x S. Hence $\mathrm{C}-\mathrm{X}^{(1,2)}$ is a valid transversal operation since it retains the same stabilizer. □

Lemma 5. *Swap operation can be performed fault-tolerantly on any encoded code.*

Proof. The swap function operates on two qudits and functions to swap the values of the qudits on which it operates. As such this operation is not fault-tolerant, but

when applied between blocks, it switches the values along with the errors. □

4. Universal Operation

The Fourier Transform (FT), Multiplication operation(M_γ) and Controlled addition operation ($\mathrm{C}-\mathrm{X}^{(1,2)}$) are necessary to produce all the operations within the centralizer. However, this is not enough. To make this set universal, we must be able to perform a nonbinary analog of Toffoli gate [17], which is the $\mathrm{C}-\mathrm{C}-\mathrm{X}^{(1,2,3)}$ operation. The action of this operation can be summarized as

$$|a\rangle\,|b\rangle\,|c\rangle \quad -^{\mathrm{C}-\mathrm{C}-\mathrm{X}^{(1,2,3)}} \rightarrow \quad |a\rangle\,|b\rangle\,|c+ab\rangle. \tag{3.2}$$

Lemma 6. *A set of gates $\mathcal{G}$ on $k>1$ qudits in a q-ary system is said to be universal if $\mathcal{G}\cup\{e^{i2\pi\theta}I\}_{real\ \theta}$ generates a dense subset in $U(q^k)$.[2]*

The set of gates with $FT, M, \mathrm{C}-\mathrm{X}^{(1,2)}, \mathrm{C}-\mathrm{C}-\mathrm{X}^{(1,2,3)}$ gives us the universal get of gates. This result has been proposed by Peter W. Shor in the paper "Fault-Tolerant Quantum Computation" for the binary case. Dorit Aharonov and Ben-Or extended this result to the nonbinary case in the paper [2]. Although, this is a well known result used in many of the papers, a formal proof of this is hard to find. Aharonov and Ben Or have given a proof for this statement in the paper [2]. We use some of the lemmas that are already proved in this paper, and use them to prove the universality property. We do not provide the proof of these lemmas here since they can be found in [2], but use them directly in our proof.

Theorem 7. *Addition of the double-controlled-addition operation $\mathrm{C}-\mathrm{C}-\mathrm{X}^{(1,2,3)}$ to the set FT, M, $\mathrm{C}-\mathrm{X}^{(1,2)}$ makes the set universal for quantum computation. i.e. $\mathcal{G}=FT, M, \mathrm{C}-\mathrm{X}^{(1,2)}, \mathrm{C}-\mathrm{C}-\mathrm{X}^{(1,2,3)}$ is a universal set.*

Proof. To prove this statement, we use many know results and lemmas from the

paper [2]. We start out by generating Q_i where Q_i, $0 \leq i < p$, denotes one qudit diagonal matrix which multiplies $|i\rangle$ by w, and applies identity of other basis states. It can be proved that Q_i and Q_i^{-1} are in the subgroup generated by $\mathcal{G}$ on three qudits [2, *Lemma* 5]. Q_i can be used to generate X_i and Y_i, which operate as identity on $S_i^{\perp}$, where S_i is given by $S_i = span\{|i\rangle, \sum_{x \in F_q, x \neq i} w^{ix}|x\rangle\}$ [2, *Lemma* 6]. When $w \neq \pm 1$, or $p \neq 2$, X_i' and Y_i' do not commute since $X_i'Y_i' - Y_i'X_i' \neq 0$. We also have, for $p > 3$, the eigenvalues of X_i and Y_i confined to S_i are not integer roots of unity [2, *Lemma* 7]. Since the eigenvalues are not roots of unity, we can get a dense subset of unitary group of all subspaces of S_i, from the statement that two non-commuting matrices U_1, U_2 in $SU(2)$ having their eigenvalues which are not integer roots of unity can generate a dense subgroup in $SU(2)$ [2, *Lemma* 2]. If G_A, G_B be dense subsets of $U(A)$,$U(B)$ respectively where A, B be two non orthogonal subspaces of C^n, then the subgroup generated by $G_A \cup G_B$ is dense in $U(A \oplus B)$ [2, *Lemma* 3]. Using this result and performing induction on i, we can generate a dense subgroup of $U(C^q)$, which represents all operations on one qudit. Since the set of gates consisting of all one-qudit gates $U(C^q)$ and all classical two-qudit gates generates all unitary matrices on two qudits, $U(C^{q^2})$ [2, *Lemma* 4], we can guarantee the universality on two qudits. The set with Hadamard, Phase, Controlled Not and Toffoli gate (binary equivalent of the gates in $\mathcal{G}$) generates a dense subgroup in the group of special unitary matrices operating on m qudits, $U(p^m)$ [2, *Theorem* 7].This ensures that these matrices can be used to construct all matrices on three qudits, $U(p^3)$. □

5. Fault-Tolerant Implementation of $\mathrm{C-C-X}^{(1,2,3)}$ Operation

We can show that the $\mathrm{C-C-X}^{(1,2,3)}$ *operation can be performed fault-tolerantly if each of the above operations can be carried out in a fault-tolerant manner.* This can be done using an additional ancilla bit as follows [6]:

The fault-tolerant implementation of doubled-controlled addition is done by generalizing the Toffoli gate construction given in Shor's Paper [17]. The double-controlled addition gate is given by

$$|a\rangle|b\rangle|c\rangle \rightarrow |a\rangle|b\rangle|c+ab\rangle \ . \tag{3.3}$$

In addition to the data bits we also need to use three extra ancilla bits in the following state:

$$|A\rangle = \sum_{a,b} |a\rangle|b\rangle|ab\rangle \ . \tag{3.4}$$

This state is a $+1$ eigenstate of the three operators

$$O_1 = (X \otimes I \otimes I)\, \mathrm{C-X}^{(2,3)} \ , \tag{3.5}$$

$$O_2 = (I \otimes X \otimes I)\, \mathrm{C-X}^{(1,3)} \ , \tag{3.6}$$

$$O_3 = (I \otimes I \otimes Z)\, P_{(1,2)^{-1}} \ . \tag{3.7}$$

$\mathrm{C-X}^{(i,j)}$ is controlled-addition performed with the ith qudit as control and the jth qudit as target. $P_{(i,j)}$ is the Phase gate

$$\mathrm{P}|a\rangle|b\rangle = \omega^{ab}|a\rangle|b\rangle \tag{3.8}$$

performed on the ith and jth qudits. This state can be constructed through Shor's method where we start with $|000\rangle$ and apply fourier transform on this state to get

$$\sum_{\alpha=0}^{q-1} |A_\alpha\rangle = \sum_{a,b,c} |a\rangle|b\rangle|c\rangle \tag{3.9}$$

By measuring O_3 for this state, we can collapse the state into once of the states $|A_\alpha\rangle$

where

$$|A_\alpha\rangle = \sum_{a,b} |a\rangle|b\rangle|ab+\alpha\rangle$$

and each $|A_\alpha\rangle$ is related to $|A\rangle$ by

$$|A_\alpha\rangle = (I \otimes I \otimes X_\alpha)\,|A\rangle$$

Application of $X_{-\alpha}$ yields the state $|A\rangle$. To construct the double-controlled addition, we assemble the ancilla qudits in the state $|A\rangle$ and data qudits below the ancilla qudits, and apply inverse controlled-addition from first and second ancilla qudit to first and second data qudit respectively, and controlled-addition operation from third data qudit to third ancilla qudit. The last three data qudits are measured in the bases Z, Z and X respectively and appropriate corrections are applied. This leaves the data in the first three ancilla qudits and and the last three data qudits have the result of the controlled-addition operation.

Since each of the operations involved in the construction of double-controlled addition can be carried out transversally, we have a fault-tolerant implementation of double-controlled addition.

CHAPTER IV

THRESHOLD ANALYSIS WITH FAULT-TOLERANT QUANTUM COMPUTATION

A. Background

Large scale quantum computations will be realistic if we ensure that the the process of error correction itself does not introduce new errors in the quantum circuits. Efficient error detection and correction techniques using fault-tolerant techniques can help in this as long as we ensure that the error probability is below a certain threshold. The calculation of this threshold is therefore important to see if quantum computations are realizable or not. The use of certain codes however may bring down this minimum threshold value. Hence, we study the basics of how threshold is calculated and compare various codes to see which one is the most efficient code.

1. Assumptions

In order to estimate the reliability of a quantum computation, we make some assumptions about the noise model present.

- The noise model taken into consideration assumes independent stochastic errors among qudits so that the probability of a correct computation can be assumed as sum of classical error probabilities.

- We assume that the error in one qudit is not related to the errors in others, that is, we have a random error model in a depolarizing channel. Hence if p_1 is the probability of one error and p_2 is the probability of another error, then the errors are uncorrelated, and the total probability of both the errors occurring can be given as $p_1 p_2$.

- We also assume that the noise level per qudit is independent of the total number of qudits or gates in the system.

Given a quantum circuit, there are various type of errors possible. However, the errors to be taken into consideration for the threshold calculation are *gate errors* and *storage errors*. A quantum computation consists of many gate operations. Hence, improper functioning of the gates can cause an error in the quantum states. The error probability increases with the increase in the number of gates, especially two qudit gates. Even when all the gates function perfectly, the surrounding noise may be such that it may cause a change in the quantum state. Either a digit may get flipped, or the phases might change. Hence, the storage errors depend on the number of timesteps involved in any computation. Hence, parallelism of operations is preferred to reduce the probability of storage errors.

Hence the threshold calculation takes four parameters into consideration [18]:

- The gate error probability, γ, and the storage error probability per qudit per timestep, ϵ, are the two parameters of noise level.

- Scale up refers to the increased number of qudits needed in the quantum computation after encoding. If there are K logical qudits and the number of actual physical qudits are N, then N/K refers to the scale up involved in the computation. This is the overhead which has to be tolerated for achieving better threshold results.

- Slowdown refers to the increased number of operations after encoding is adopted, that is, the number of gates per computational step. If K logical qudits use Q computational steps, and the number of elementary gates used for N physical qudits is T, then the slow down is given by T/Q.

- The amount of parallelism refers to the ability of performing multiple gate operations simultaneously. This feature is important for reducing the storage errors.

The analysis of which quantum error correcting code is most efficient is based on the above four factors. Optimizing all the four parameters is not possible, but we can select the code based on the requirements of the computation.

2. Syndrome Calculation

Error detection and correction for a quantum code is done through syndrome calculation. The procedure adopted is similar to the one suggested by Andrew Steane in [18]. The step by step illustration is given as follows:

- The n digit blocks to be corrected are referred to as b. For each block b, an ancilla register, a_x is prepared which consists of n qudits and another set of n qudits are used for verification. The subscript x signifies that the ancilla block is used for syndrome calculation for bit errors. Similarly, we use another n-qudit block, a_z, for phase error correction.

- The ancilla state is a cat state represented as

$$|cat\rangle = \sum_{a=0}^{q-1} |a^n\rangle \tag{4.1}$$

  This state is a superposition of all strings of equal qudits. This state is prepared by starting with an all zero state and using a Fourier transform on the first qudit, then taking controlled- addition from first qudit to all other qudits. Since this is not a fault-tolerant operation, we have to first verify that the cat state thus created did not have any errors. This can be done by checking if the parity of

the qudits in the state thus created is zero mod q. If the state was found to be in error, we discard the state and start with the cat state preparation from the beginning. However, if the state was verified to be correct, we take the Fourier transform on each of the qudit which gives us the desired state

$$\sum_{j, j\cdot\vec{1}=0 \ mod \ q} |j\rangle$$

which is a superposition of all states j which satisfy $\sum_k j_k = 0 \ mod \ q$.

This process of cat state preparation can be clearly explained with the help of the following Figure 2.

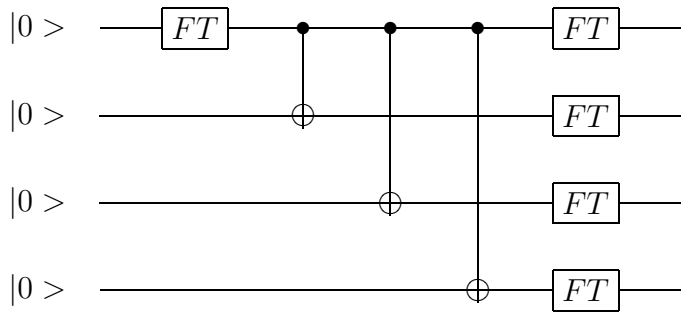

Fig. 2. Preparation of the cat state

- When we want to compute the i'th bit of the syndrome fault-tolerantly, we simply have to take the inner product of the i'th row of the parity check matrix with the codeword. This can be carried out by applying controlled-addition operations from the block we are correcting to the cat state only on those coordinates having a non-zero value in the i'th row of the parity check matrix, that is, to compute the inner product for the i'th row of the parity check matrix, $h_{i,l}$, with a vector a_l, we need to sum $\sum_l h_{i,l} a_l$. We thus add $h_{i,l} a_l$ to the l'th coordinate of the ancillary state. If the parity of the ancilla state after such operation is non-zero, then we conclude that i'th bit of the syndrome is in

error.

- The advantage of using such a cat state for syndrome calculation is that there is no entanglement between the qudits in block b and the qudits in the ancilla. The controlled-additions are carried out digit-wise from the block b to the qudits in the ancilla block a_x as can be seen in the above step. Hence, we cannot propagate more errors into block b than those existing already, from the ancilla block. [17].

- The syndrome calculation method is repeated r number of times where $r = t+1$ number of times to ensure a high probability of correct syndrome calculation.

- Phase checks are carried out in the similar manner, except that the entire process is repeated in the conjugate basis.

3. Error Model

The threshold analysis can be well understood from the error model described in this section. The error correcting code can correct at most t errors. Hence n digits in block b can be in error. However, we have to ensure that the ancillary block does not introduce or propagate more than t errors into b. The errors due to single gate failure is accounted as an error in the corresponding bit, and there is no propagation involved in this case. However, when the controlled-addition operations are involved, there is interaction between the digits in different blocks which can lead to propagation of errors too. For this, we have to assume that the syndromes for amplitude errors and phase errors are calculated alternately, so that the phase errors introduced by the error correction procedure can be corrected in the next phase and vice versa. However, this also requires that the first stage of syndrome calculation is more accurate than the later stages. Hence, the error correction will fail when either there are more than t

errors in the block b already, or the gate errors lead to more than t errors in the block b.

If p is the probability that an error occurs in one qudit, then the probability of an errorless computation is $1 - p$. The error basis for a $q - ary$ system can be given by q^2 operators out of which one is the identity. Since the identity operator does not change anything, each of the other $q^2 - 1$ operators cause the state of the qudit to be changed, hence, each of these operations have $\frac{p}{q^2-1}$ probability of an error. The bit errors become sign errors in the conjugated basis, and hence have the same probability in the conjugate computational basis too.

When two qudits are involved, each of the qudits can undergo q^2 errors giving totally q^4 combinations. Out of these, $I \otimes I$ represents an error free case and other $q^4 - 1$ cases occur with equal probability of $\frac{p}{q^4-1}$. We assume that one of the qudit among the two, is already in error. The probability that this error is propagated to the second qudit depends on whether the correct qudit is the target or the control in the two-qudit gate operation. The errors in the correct qudit may be induced due to the error in the other qudit controlling it, or due to the faulty gate operation. Hence, when the controlling qudit is in error, there are $q^2 - 1$ opportunities of inducing error in the other qudit or a faulty gate can induce $q^2 - 1$ error operators again. Since we are finding the error probability on the same qudit in various situations, we sum the error opportunities to give a probability of $\frac{2q^2-2}{q^4-1}$ of error in the second qudit during a two qudit operation when one is already in error. This expression can be further simplified as $\frac{2}{q^2+1}$.

B. Analysis

The assumptions and error model explained in the previous sections can be used to estimate the threshold for gate error. We use the steps described above to compute the number of gates used and timesteps required in syndrome calculation, which are the key factors in the threshold calculation. If m is the number of rows in the generator matrix of classical code giving $|0_E\rangle$, and w the average weight of each row, then the total number of gates and timesteps required for error correction can be calculated in the following way: We need 1 step for initial fourier transform while preparing the ancilla a_x, $m(w-1)$ steps for the controlled-addition operation between the digits in a_x, 1 step again for fourier transform for rotation, $m(w+1)$ controlled-addition operations between a_x and verification ancilla block, and finally $m+1$ steps for measurement operations. A further $2n+1$ steps required for measurement of the two ancilla blocks. Hence, the total timesteps required is :

$$1 + m(w-1) + 1 + m(w+1) + m + 1 + 2n + 1 = m(2w + 1/2) + 2n + 4 \qquad (4.2)$$

For CSS codes, the generator divides equally into X part and Z part. Hence, $m = (n-k)/2$. So this yields $(n-k)(w+1/2) + 2n + 4$ timesteps for each digit in the codeword. There are totally n digits in the codeword. And the syndrome calculation repetition is done $r = t+1$ times. Hence, the total number of storage errors are given as

$$S_s = n((n-k)(w+1/2) + 2n + 4)r \qquad (4.3)$$

when the syndrome is generated in series, that is one after the other. Else

$$S_p = n((n-k)(w+1/2) + 4 + 2nr) \qquad (4.4)$$

when syndrome is generated in parallel.

The gate errors are directly dependant on the number of gates and the number of interactions between the ancilla blocks and the block b. There are n opportunities for controlled-addition operation between first ancilla a_x and block b, giving rn opportunities (since syndrome calculation is repeated r times). The interaction between second ancilla a_x and block b gives further rn opportunities. The gates errors which were undetected in b were due to the controlled-addition and fourier transform operations applied in the last stage giving $2rn$ opportunities again. Finally n correction gates applied from the ancilla bits to the block b gives n more gate error possibilities. Hence the total gate error opportunities is given as

$$g = rn + rn + 2rn + n = n(4r + 1) \tag{4.5}$$

An $[[n, k, d]]_q$ code can correct at most t errors. Hence, with this code, the probability that more errors (either bit errors or sign errors) accumulate than can be corrected when there are zero errors in the block b and we allow t errors to propagate from the ancilla block a_x is given as

$$P \simeq 2 \sum_{i=t+1}^{g} \begin{pmatrix} g \\ i \end{pmatrix} \left(\frac{2}{q^2+1}\gamma + \frac{s}{g}\frac{2}{q^2+1}\epsilon \right)^i \tag{4.6}$$

This expression basically means that when we have an $[[n, k, d]]_q$ code which t error correcting capabilities, then the presence of $t + 1$, $t + 2$, $\ldots$, g errors lead to the failure of error correction procedure. Hence we have the summation of all of these error probabilities. Also, the presence of say $t+1$ errors can be in any of the g gates. So we get to choose $t+1$ of these g values and assuming independent error probability between different qudits, we have a product of the error (both gate and storage errors) of each of these $t + 1$ errors.

For serial syndrome repetition, we take $n\epsilon = \gamma/2$ and for parallel syndrome

repetition, we have $n\epsilon = 2\gamma$ [18]. For a computation with Q steps and K logical qudits to be successful, we require that $P < \eta/KQ$, where η is the average number of computational steps per correction of the whole computer. Factorizing a 430 digit number using Shor's algorithm requires approximately $K \simeq 5 \times 430 = 2150$ encoded qubits and approximately 10^9 double controlled-addition operations [4] which requires 20 fault-tolerant operations. Hence, $Q = 20 \times 10^9$ which gives $P < 2 \times 10^{-14}\eta$. Using different nonbinary codes may require different number of encoded operations and computations. However, to compare the threshold with various codes, we would rather consider a generic application requiring the same number of computations and encoded operations as that illustrated by Shor. Hence, we would compare the threshold got from various codes for $P < 2 \times 10^{-14}\eta$. we Hence, we can solve equation (4.6) to find the maximum value of γ such that this expression is satisfied.

Also, this procedure of error correction produces extra overheads such as the scale up of the bits (i.e. extra bits required for error correction) and slowdown (given by the number of gates per computational step) which can be given by

$$\frac{N}{K} = \begin{cases} n + 2(n+1) & \text{ancilla syndromes calculated in a serial manner} \\ n + 2r(n+1) & \text{ancilla syndromes calculated parallelly} \end{cases} \quad (4.7)$$

$$\frac{T}{Q} = \left((n-1)w + 5n\right) 2rK/\eta. \quad (4.8)$$

As the code size increases, the scale up and slowdown overheads increase with it. Hence, the threshold analysis should be carried out on various codes to strike a balance between the threshold results and overheads incurred.

CHAPTER V

THRESHOLD RESULTS

Fault-Tolerant computation aims at prevention of spread of errors which causes the quantum error correction to fail. However, fault-tolerance is achieved by adding redundant qudits and extra gates, which also increases the probability of error. Hence, we require the quantum circuits to perform the computation with an arbitrary accuracy, as long as the component gates have the error probability below a certain threshold. It is important to find the range of this threshold estimate, in order to check the feasibility of computations. Using the threshold analysis done in Chapter 4, we can test various classes of codes to derive the threshold value. In this chapter, we will first review the various classes of codes that we can use, and find the most efficient ones through this threshold analysis.

A. Codes Families

1. Quantum q-ary Hamming Code

The $q-ary$ Hamming code is an example of a doubly even CSS codes, where each column of the parity check matrix H consists of one non-zero vector from each vector subspace of dimension 1 of ${F_q}^r$ denoted by Ham(r,q). Ham(r,q) is a single error correcting code with constant distance 3, and it can be given as:

$$H = [[(q^m - 1)/(q-1), (q^m - 1)/(q-1) - 2m, 3]]_q \tag{5.1}$$

The parity check matrix of this Hamming code has m rows and n columns. This can be got into the standard form where the first part of the matrix is the identity.

$$m\{ \left(\overbrace{I}^{m} \middle| \overbrace{A}^{n-m} \right) \tag{5.2}$$

The identity matrix on the left ensures that there is only one non-zero element in that portion of each row. The other part indicated as A on the right can have both zero or non-zero entries. As the size of the matrix increases, this portion becomes more and more dense. Hence, for worse case, we can take all the entries in this matrix to be non-zero which gives us the average weight of each row to be $n-m+1$. This is the value w required in the calculation of the probability of successful error correction given in (4.6). Lower the value of w, higher is the probability or error correction, since number of gates during syndrome calculation i.e. no. of interaction between qudits is directly proportional to w.

Thus, equation (4.6) is used to find the success rate of an application which requires $1/KQ = 2 \times 10^{-14}$ for both serial and parallel syndrome calculation. The results of this analysis are presented in the Table II.

2. BCH Codes

Bose-Chaudhuri-Hocquenghem codes (BCH codes) [9] is an important class of error correcting codes constructed from a generating polynomial g(X), a monic polynomial over F_q of smallest degree that has $\delta - 1$ numbers $\omega^b, \omega^{b+1}, \ldots, \omega^{b+\delta-2}$ among its zeroes. Therefore,

$$g(X) = lcm\{m_b(X), m_{b+1}(X), \ldots, m_{b+\delta-2}(X)\} \tag{5.3}$$

where $m_i(X)$ is the minimal polynomial of ω^i, and we require $b \geq 0$ and $\delta \geq 1$.

Self-orthogonal classical BCH codes can be used to construct quantum BCH

Table II. Threshold Analysis for Quantum $q-ary$ Hamming Codes

$[[n,k,d]]_q$ codes	Units	γ_p	ϵ_p	γ_s	ϵ_s
$[[7,1,3]]_2$	10^{-9}	1.86	0.53	3.15	0.22
$[[4,0,3]]_3$	10^{-9}	7.03	3.51	1.17	0.14
$[[13,7,3]]_3$	10^{-9}	1.94	0.29	3.30	0.12
$[[5,1,3]]_4$	10^{-9}	9.57	3.82	1.59	0.15
$[[21,15,3]]_4$	10^{-9}	2.01	0.19	3.43	0.08
$[[6,2,3]]_5$	10^{-8}	1.22	0.40	2.03	0.16
$[[31,25,3]]_5$	10^{-9}	2.07	0.13	3.53	0.05
$[[7,3,3]]_6$	10^{-8}	1.44	0.41	2.48	0.17
$[[43,37,3]]_6$	10^{-7}	4.59	0.21	6.80	0.07

codes. Let $C = [n, k, d]$ be such a classical BCH code used in the construction of the quantum BCH code. If Z_c denotes the zero set of C over field F_{q^2}, then the generator polynomial of C is given by

$$g(X) = \prod_{z \in Z_c} (X - \alpha^z)$$

The generator polynomial of the dual code is given by

$$h(X) = \prod_{z \in Z_n \setminus Z_c} (X - \alpha^{-qz})$$

Quantum BCH codes can also be constructed from the extension field $E = F_{q^{2m}}$ which is the extension of the base field F_{q^2}. BCH codes can be constructed by expanding over the hermitian self-dual basis denoted by $\mathbf{E/F}$ [10]. Given a vector x in $\mathbf{E^n}$, we denote by $\mathbf{x_E}$ the expansion of x with respect to the basis E, i.e.

$$x = \left(\sum_{b \in E} x_{bk} b \right)_{k \in N}$$

We can follow the similar procedure to construct $\mathbf{D_E}$ given by $\mathbf{D_E} = \{\mathbf{x_E} | \mathbf{x} \in \mathbf{D}\}$ where $\mathbf{D}$ is an additive code of length n over $\mathbf{E}$.

A few simple constructions of BCH codes are summarized in the paper [1] which can be used to derive the quantum BCH codes used in our analysis. Restating the two formulae given in this paper:

Construction Method 1: [1]

If q is a power of a prime, and m and δ are integers such that $m \geq 2$ and $2 \leq d \leq q^{\lceil m/2 \rceil} - 1 - (q-2)$ [m odd], then there exists a quantum code with parameters

$$[[q^m - 1, q^m - 1 - 2m\lceil (d-1)(1-1/q) \rceil, \geq d]]_q$$

Table III. List of Quantum BCH Codes

$[[n,k,d]]_q$ codes	Details
$[[8,6,2]]_3, [[10,2,3]]_3, [[10,6,2]]_3$ $[[13,7,3]]_3, [[7,1,3]]_4, [[9,1,3]]_4$ $[[9,3,2]]_4, [[9,7,2]]_4, [[17,9,3]]_4$	Codes taken from [9]
$[[80,72,4]]_3, [[80,60,8]]_3, [[255,243,4]]_4$ $[[255,211,15]]_4, [[624,612,4]]_5, [[624,592,23]]_5$	Construction Method 1
$[[26,14,3]]_3, [[26,2,7]]_3, [[63,51,3]]_4$ $[[63,15,13]]_4, [[124,112,3]]_5, [[124,28,21]]_5$	Construction Method 2

that is pure up to δ.

Construction Method 2: [1]

If q is a power of a prime, and m is a positive integer and $2 \leq d \leq q^m - 1$ then there exists a quantum code with parameters

$$[[q^{2m} - 1, q^{2m} - 1 - 2m\lceil (d-1)(1-1/q)\rceil, \geq d]]_q$$

that is pure up to δ.2

Using these construction methods, we derive the following codes in Table III. The threshold results on some of these codes is summarized in Table IV.

3. Generalized Reed-Muller Codes

Tadao Kasami in [8] has generalized the Reed-Muller codes to get primitive Reed-Muller codes. These codes have been used to construct a series of stabilizer codes

Table IV. Threshold Analysis for Quantum BCH Codes

$[[n,k,d]]_q$ codes	Units	γ_p	ϵ_p	γ_s	ϵ_s
$[[10,2,3]]_3$	10^{-9}	2.40	0.48	4.11	0.20
$[[16,6,4]]_3$	10^{-9}	1.34	0.16	2.33	0.07
$[[40,26,5]]_3$	10^{-8}	9.94	0.49	14.3	0.17
$[[7,1,3]]_4$	10^{-9}	6.34	1.81	10.7	0.76
$[[9,1,3]]_4$	10^{-9}	4.59	1.02	7.85	0.43
$[[17,9,3]]_4$	10^{-9}	2.28	0.26	3.94	0.11
$[[25,3,5]]_4$	10^{-7}	2.54	0.20	3.64	0.07
$[[26,14,3]]_3$	10^{-10}	7.35	0.56	12.9	0.24
$[[26,2,7]]_3$	10^{-6}	2.18	0.16	2.70	0.05
$[[63,51,3]]_4$	10^{-10}	4.84	0.15	8.60	0.06
$[[63,15,13]]_4$	10^{-5}	3.84	0.12	3.16	0.02
$[[80,60,8]]_4$	10^{-6}	1.08	0.02	1.31	0.008

from generalized Reed-Muller codes using the CSS code construction.

Let $(P_1, \ldots, P_n)$ denote an enumeration of all points in F_q^m with $n = q^m$. We denote by $L_m(v)$ the subspace of $F_q[x_1, \ldots, x_m]$ that is generated by polynomials of degree v or less. Let v denote an integer in the range $0 \leq v < m(q-1)$. The classical generalized Reed-Muller code $R_q(v, m)$ of order v is defined as

$$R_q(v, m) = \{(f(P_1), \ldots, f(P_n)) | f \in L_m(v)\} \tag{5.4}$$

The dimension $k(v)$ of the code $R_q(v, m)$ equals

$$k = \sum_{j=0}^{m} (-1)^j \binom{m}{j} \binom{m+v-jq}{v-jq} \tag{5.5}$$

The minimum distance of this code is given by

$$d = (R+1)q^Q \tag{5.6}$$

where $m(q-1) - v = (q-1)Q + R$ and $0 \leq R < q-1$. The dual code of $R_q(v, m)$ is again a generalized Reed-Muller code given by

$$R_q(v, m)^\perp = R(v^\perp, m) \qquad with \;\; v^\perp = m(q-1) - 1 - v. \tag{5.7}$$

This classical generalized Reed-Muller codes can be used to derive quantum reed-Muller codes through CSS construction i.e. if $C_1 = [n, k_1, d_1]_q$ and $C_2 = [n, k_2, d_2]_q$ be linear codes over F_q with $C_1 \subseteq C_2$. Furthermore, let $d = \text{MinWt}\{(C2 \setminus C1) \cup (C_1^\perp \setminus C_2^\perp)\}$ if $C_1 \subset C_2$ and $d = \text{MinWt}\{(C1) \cup (C_1^\perp)\}$ if $C_1 = C_2$. Then there exists an $[[n, k_2 - k_1, d]]_q$ quantum code [16].

Kasami gives examples of some of the generalized Reed-Muller codes in his paper [8]. We use these codes to tabulate the threshold values. The result of the runs on these codes is tabulated in Table V.

Table V. Threshold Analysis for Quantum GRM Codes

$[[n,k,d]]_q$ codes	Units	γ_s	ϵ_s	γ_p	ϵ_p
$[[7,4,3]]_2$	10^{-9}	2.09	0.59	3.47	0.24
$[[15,11,3]]_2$	10^{-10}	9.41	1.25	1.57	0.05
$[[15,5,7]]_2$	10^{-6}	2.40	0.32	3.19	0.10
$[[31,26,3]]_2$	10^{-10}	4.22	0.27	7.13	0.11
$[[8,3,5]]_3$	10^{-7}	7.03	1.75	1.04	0.06
$[[26,17,5]]_3$	10^{-7}	1.81	0.13	2.66	0.05
$[[26,10,8]]_3$	10^{-6}	2.35	0.18	2.98	0.05
$[[15,10,3]]_4$	10^{-9}	2.97	0.39	5.01	0.16
$[[15,3,11]]_4$	10^{-4}	1.19	0.15	1.39	0.04
$[[63,54,3]]_4$	10^{-10}	5.55	0.17	9.69	0.07
$[[63,44,7]]_4$	10^{-6}	1.42	0.04	1.73	0.01
$[[24,19,3]]_5$	10^{-9}	2.83	0.23	4.79	0.09
$[[24,15,4]]_5$	10^{-9}	2.30	0.19	4.00	0.08
$[[24,10,9]]_5$	10^{-5}	3.57	0.29	4.21	0.08
$[[24,6,14]]_5$	10^{-4}	2.24	0.18	2.30	0.04
$[[48,43,3]]_7$	10^{-9}	2.72	0.11	4.60	0.04
$[[48,39,4]]_7$	10^{-9}	2.15	0.08	3.76	0.03
$[[48,34,5]]_7$	10^{-7}	4.09	0.17	5.88	0.06
$[[48,28,6]]_7$	10^{-7}	3.51	0.02	5.007	0.05
$[[63,58,3]]_8$	10^{-9}	2.70	0.08	4.56	0.03
$[[63,54,4]]_8$	10^{-9}	2.12	0.06	3.70	0.02
$[[63,49,5]]_8$	10^{-7}	3.99	0.12	5.73	0.04
$[[63,36,7]]_8$	10^{-7}	3.99	0.12	55.4	0.43

B. Interpretation of Results

Equation 4.6 was solved in mathematica, substituting various $[[n, k, d]]_q$ codes. However, since the order of the polynomial was large and increasing with the code size, the solution set consisted of many positive and negative real and imaginary values. The tables above have been tabulated based on the positive real solution having the worst precision (the negative numbers were disregarded since probability cannot be negative). To study the variance of this threshold with various factors such as size of the code, distance of the code, the number of nonbinary levels etc, we plot the gate error probabilities against each of these factors. The relation of the threshold value with the type of code considered can be determined from the graphs plotted.

First we try to find the relation between the threshold value and the number of alphabets in the nonbinary system. Hence, we plot a graph with γ vs q for the Hamming codes. We find that as we move to higher nonbinary levels, higher is the threshold estimate. In other words, the nonbinary codes with higher q values can tolerate more gate errors than those with smaller q levels. Hence, it is more advantageous to use nonbinary codes over binary error correcting codes. This relation is illustrated through Figure 3

We can also see the correlation between the size of the code used and the threshold results. This is done by keeping a constant value of q, and taking bigger codes for a particular value of q. If we consider bigger codes having the same error correcting capability, then the threshold decreases. This can be illustrated from Figure 4.

The variance of the threshold with respect the codes having increasing error correcting capability can be shown in Figure 5. In this graph, we consider codes with a particular value of n for a particular value of k and varying value of d. In this case too, we find improving threshold results with codes correcting having better error

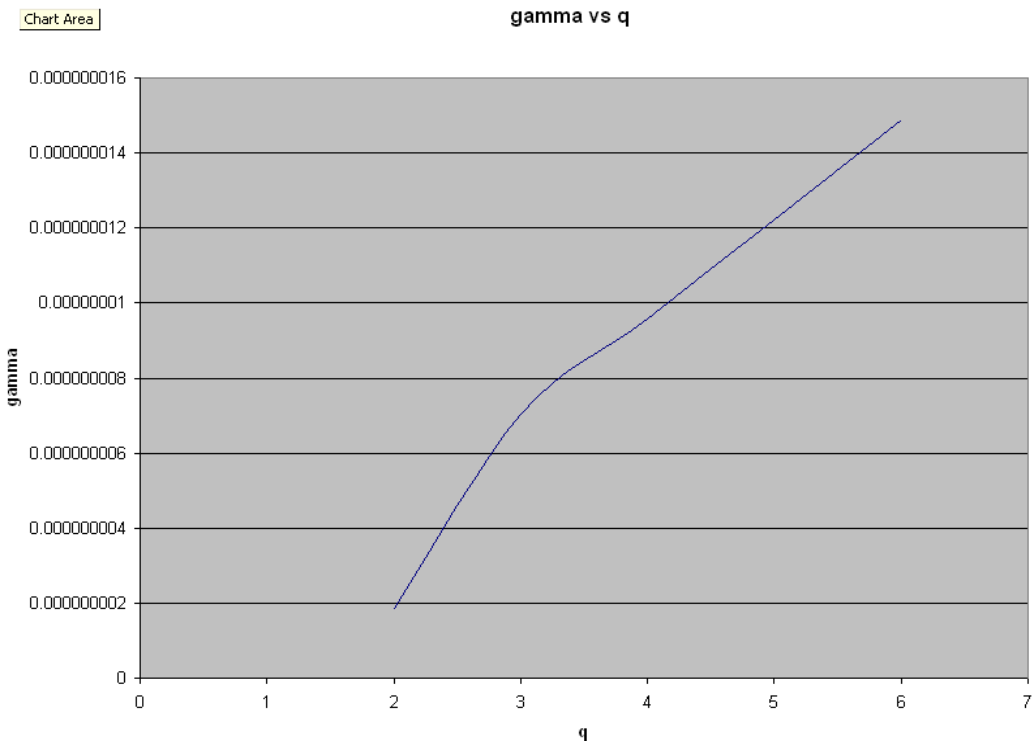

Fig. 3. Variance of γ with q

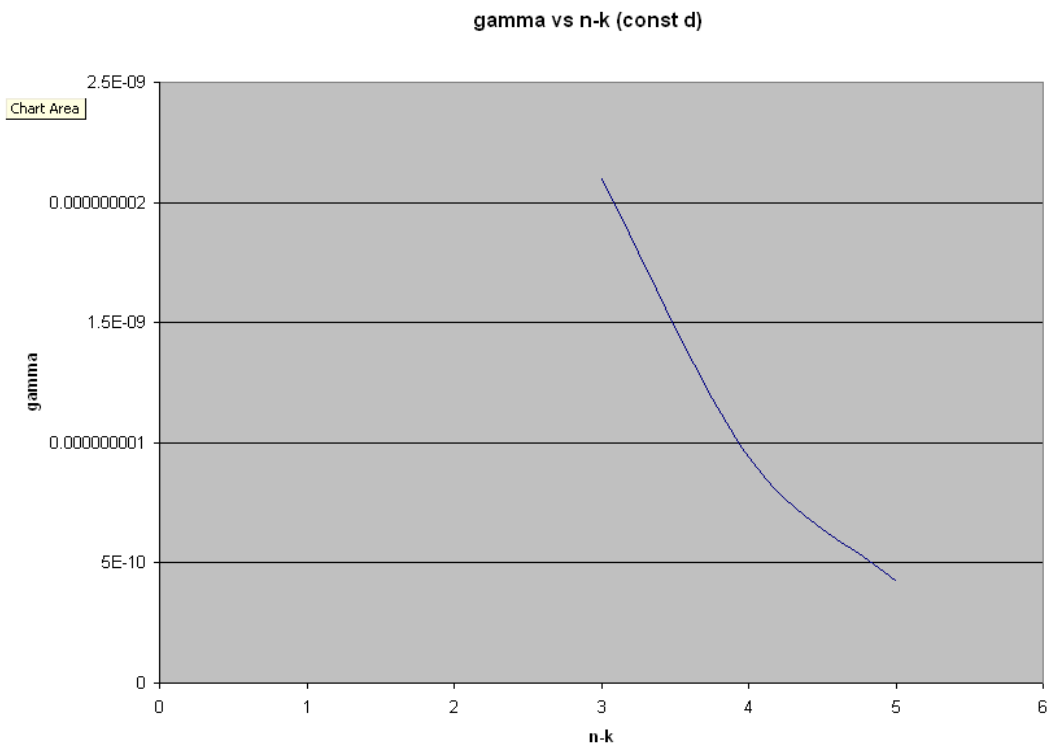

Fig. 4. Variance of γ with $n-k$ for codes with constant d

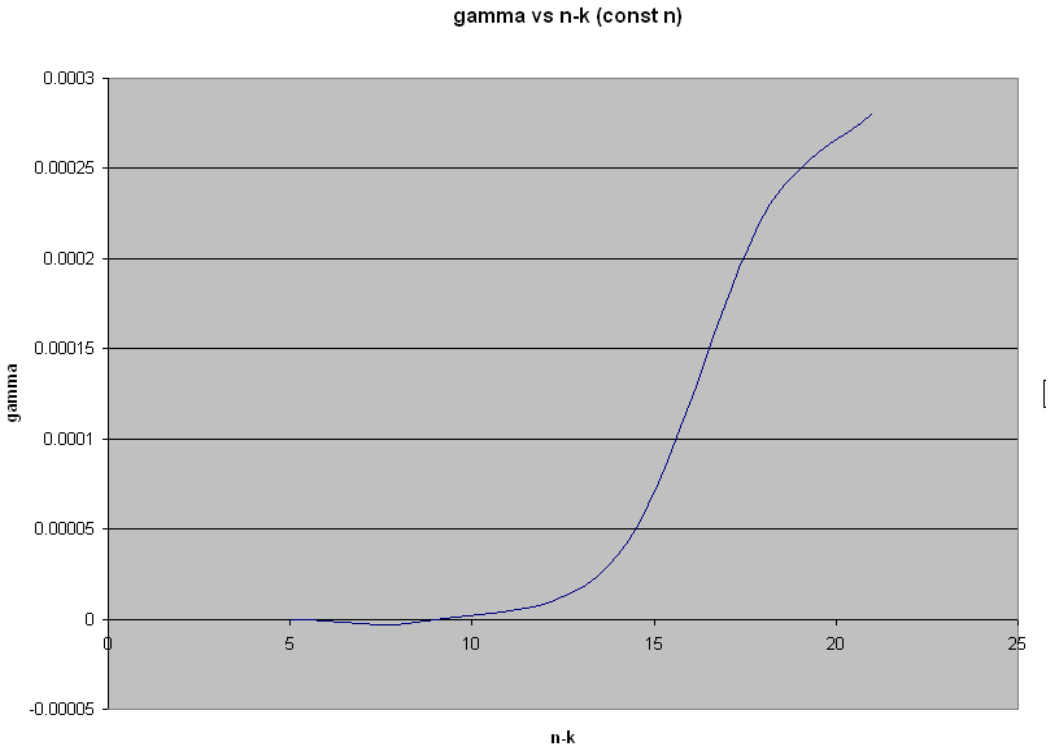

Fig. 5. Variance of γ with $n-k$ for codes with constant n

correction capability.

From Figures 4 and 5, we can conclude that it is more advantageous to use bigger nonbinary codes with higher error correcting capability, to have a higher estimate for gate error threshold.

CHAPTER VI

FUTURE WORK

Chapter III gives us a set of fault-tolerant encoded operations that can be used for universal computation only on CSS codes. However, there are a lot of different operations that can be used to generate the universal set. Hence, we can try to find different classes of codes which support different set of universal transversal operations, other than those mentioned in chapter III. This might lead us to better error correction techniques based on the classes of codes considered, which in turn, might give us a better threshold result.

Chapter IV gives the threshold analysis based on the assumption that all the errors can be classified as either gate errors or storage errors. We also assume that errors occur in the block b, only after propagation of $t+1$ or more errors. In other words, there was no error in this block b from the encoding process. This might not be the case. There might be errors in the codeword which might be due to the encoding gates or due to the noise level in the surrounding atmosphere. Hence, if the channel and noise properties are better known, then we can prepare a more accurate error model. We also assume the standard form of encoding for the various nonbinary codes. However, this might not be the best way of encoding requiring minimum number of gates. Hence, we can also try to find other efficient encoding techniques that require minimum gate operations.

Also, we have performed the threshold analysis for various classes of codes in chapter V. The comparison of different classes is difficult when represented in the nonbinary form, because we do not have an accurate estimate for the amount of encoding done at various levels, and their relation to the threshold. The actual computations on a quantum computation would also probably be in terms of binary

values. Hence, in order to have an accurate comparison between various classes of codes, we need to bring everything in terms of the binary codes. This can be done in two ways

- We can try to reduce the nonbinary codes to equivalent binary codes and compute the threshold.
- Another way to have a fair comparison is to represent the nonbinary gate operations in terms of the binary gates.

The introduction of these changes might lead to a better interpretation and evaluation of the threshold results. One might also consider different approaches for threshold evaluation such as those considered in [12].

CHAPTER VII

CONCLUSION

This thesis covers the area of fault-tolerance for non-binary quantum error correcting codes. We have given a universal set of fault-tolerant operations valid for CSS codes. This ensures that the already existing errors do not propagate while computations are performed. When this is the case, the major source of errors can be the error correcting circuit itself, where the errors can propagate from the syndrome ancilla bits to the data bits, hence corrupting the information bits. Fault-Tolerance is a big leap in the quantum world and is needed to make computations feasible. Even with fault-tolerant operations, there is a probability of failure of the computations because of the error introduction and propagation from the gate interaction between the data qudits and the detection and correction circuit. Hence, we have also performed the threshold analysis to check the probability of success of computations. This analysis summarizes how the gate failure affects the reliability of the computations and fixes an upper bound of the threshold for the gate error probability. However, this threshold is calculated for a benchmark application (the factorization of a 430 digit number) and depends on the parameters of the code used. So we compute the threshold for various families of codes and compare these values to find which one is more efficient.

The analysis performed on threshold suggests that the upper bound on allowable gate error improves as the number of nonbinary levels increases. It is also directly proportional to the size of the code i.e. bigger the code, higher is the value of threshold we get. For a particular value of encoding done (i.e particular n value in an $[[n, k, d]]_q$ code), better gate error probability bounds are achieved for codes with higher error correcting capability (i.e bigger d value) than those encoding more number of information bits, k. Hence, we need to use bigger nonbinary codes with higher error

correcting capability to get a higher value of gate error threshold.

REFERENCES

[1] S. ALY, A. KLAPPENECKER, P.K. SARVEPALLI, *On the Dimension, Minimum Distance, and Duals of Primitive BCH Codes*, quant-ph/0501126.

[2] D. ARAHANOV, *Fault-Tolerant quantum computation with constant error rate*, SIAM Journal of Computation. See also LANL preprint quantum/9906129.

[3] A. ASHIKMIN, E. KNILL, *Nonbinary quantum stabilizer codes*, IEEE Trans. Inf. Theory, 47 (2001), 3065-3072.

[4] D. BECKMAN, A. CHARI, S. DEVABHAKTUNI, J. PRESKILL, *Efficient networks for quantum factoring*, Phys. Rev. A, 54 (1996), 1034-1063.

[5] R. CLEVE, D. GOTTESMAN, *Efficient computations of encodings for quantum error correction*, Phys. Rev., A, 56 (1997), 76-82.

[6] D. GOTTESMAN, *Fault-Tolerant quantum computation with higher-dimensional systems*, Chaos Solitons Fractals, 10 (1999), 1749-1758, quant-ph/9802007.

[7] M. GRASSL, M. ROETTELER, T. BETH, *Efficient quantum circuits for non-qubit quantum error-correcting codes*, International Journal of Foundations of Computer Science (IJFCS), 14(5)(2003), 757-775.

[8] T. KASAMI, S. LIN, W. PETERSON, *New generalizations of the Reed-Muller Codes part I: Primitive codes*, IEEE Transactions on Information Theory, 14(2)(1968), 189-199.

[9] A.U. KETKAR, *Code Constructions and Code Families For Nonbinary Quantum Stabilizer Codes*, M.S. Thesis, Department of Computer Science, Texas A&M University, College Station, 2004.

[10] A.U. KETKAR, A. KLAPPENECKER, S. KUMAR, P.K. SARVEPPALI, *Nonbinary Stabilizer Codes Over Finite Fields*, Department of Computer Science, Texas A&M University, College Station, 2004.

[11] A. KLAPPENECKER, *A Short Introduction to Stabilizer Codes*, Department of Computer Science, Texas A&M University, College Station, 2003.

[12] E. KNILL, *Fault-Tolerant Postselected Quantum Computation: Threshold Analysis*, quant-ph/0404104.

[13] R. MATSUMOTO, T. UYEMATSU, *Constructing quantum error-correcting codes for p^m-state systems from classical error-correcting codes*, IEICE Trans. Fundamentals, E83-A(10)(2000), pp. 1878-1883.

[14] M. NEILSON, I. CHUANG, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, 2000.

[15] J. PRESKILL, *Fault-Tolerant Quantum Computation*, quant-ph/9712048.

[16] P.K. SARVEPALLI, A. KLAPPENECKER, *Nonbinary Quantum Reed-Muller Codes*, quant-ph/0502001.

[17] P. SHOR, *Fault-tolerant quantum computation*, Presented at the 37th Symposium on Foundations of Computing, IEEE Computer Society Press, 1996, pp. 56-65.

[18] A.M. STEANE, *Space, time, parallelism and noise requirements for reliable quantum computing*, Fortsch. Phys. 46 (1998), pp. 443-458.

[19] A.M. STEANE, *Efficient fault-tolerant quantum computing*, quant-ph/9809054.

[20] A.M. STEANE, *Overhead and noise threshold of fault-tolerant quantum error correction*, quant-ph/020711.

APPENDIX A

IMPLEMENTATION DETAILS

(**********)

(* Threshold calculation for Various codes *)

(**********)

(* Input: n,k,d,q,w values *)

(* Output: The various values for γ *)

(* Benchmark for computation is factorization of a 130 digit number. *)

(* For a computation involving Q sets and K logical qudits to be *)

(* successful, we require $P < \eta/KQ$. *)

(* The program calculates the gate error threshold required to perform *)

(* computations with K=2150 encoded qudits and $Q = 2 \times 10^{10}$ *)

(* computational steps. The program is repeated to test the performance *)

(* of serial syndrome calculation with parallel syndrome calculation. *)

(* We assume $n\epsilon = \gamma/2$ for serial syndrome repetition and *)

(* $n\epsilon = 2\gamma$ for parallel syndrome calculation. *)

(**********)

(* We input the values n,k,d based on the [[n,k,d]] code *)

n=7

k=1

d=3

(* Based on the distance d, the number of errors that can be corrected is given as *)

t=(d-1)/2

(* r refers to the number of times the syndrome calculation is repeated *)

$r = t + 1$

(* w refers to the average number of non-zero elements in each row of the parity check matrix. The number of controlled-addition operation and hence the possibility of error propagation depends on w *)

$w = 4$

(* s is the number of storage errors. When syndrome calculation is done serially, $s = n((n-k)(w+0.5) + (2n) + 3)r$. And when syndrome calculation is done parallely, $s = n((n-k)(w+0.5) + 3 + 2nr)$. *)

$s = n((n-k)(w+0.5) + (2n) + 3)r$

(* q is the number of levels in the non-binary system. It is usually taken as a prime power. *)

$q = 2$

(* g refers to the gate error opportunities *)

$g = n(4r+1)$

(* $1/KQ$ is taken as $2(10^{(-14)})$ which is sufficiently long computation *)

$p = 2(10^{(-14)})$

(* e refers to the value of ϵ which refers to the probability of storage error per qudit per timestep. $\epsilon = \gamma/(2n)$ for serial syndrome calculation and $\epsilon = 2\gamma/n$ for parallel syndrome calculation. *)

$e = gam/(2n)$

(* Probability that error is introduced in the second qudit during a 2-qudit interaction when one qudit is in error *)

frac= $(2)/(q^{(2)} + 1)$

(* Probability that an error cannot be corrected is given by the expression in equation (4.6) *)

$r = 2$Sum[((Factorial[g]/($Factorial[i]Factorial[g-i]$))(((frac)$gam$)+((frac)$(s/g)e$))^i), $\{i, t+1, g\}$]

(* Computation is successful when $P < \eta/KQ$. Hence, we solve for $P = \eta/KQ$ and take the maximum positive value of γ to be the gate error threshold. *)

Solve[$p == r$]

VITA

Name : Aparna Kanungo

Educational Background : B.E. Computer Science - Anna University

Permanent Address : A-3 Income Tax Colony, Peddar Road, Mumbai - 400026, India