Noisy Quantum Computation

Thesis submitted for the degree
Doctor of Philosophy
by

Dorit Aharonov

Thesis submitted to the Senate of the Hebrew University, July 1999

This work was carried out under the supervision of
Professors Michael Ben-Or and Avi Wigderson.

Acknowledgments

I am most grateful to Michael Ben-Or who introduced me to this beautiful subject. It was with Michael that I first experienced the fascinating process of a new scientific idea coming to life; I learned a lot from him about kindness, and about setting the mind free on its way to seek for ideas. Noam Nisan showed me how to look at things honestly, clearly and brightly, in a way which slightly resembles a child. It was great fun to have uncountably many enthusiastic brain storms with Avi Wigderson, on infinitely many subjects. I learned a lot from Avi, about everything.

I wish to thank my friends in the theory lab, Amnon Ta-Shma who inspired me with his joy in seeking the truth, Roy Armoni who encouraged me in the beginning of my work on this thesis, Ziv Bar-Yoseph, Eli Ben-Sasson, Ido Bergman, Yonatan Bilu, Shlomo Churi, Eyal Gofer, Boaz Lesloy, Avner Magen, Ahuva Mualem, Dana Pe'er and Ronen Shealtiel; together they created a joyful atmosphere which made it impossible to study.

It is a special pleasure to thank all my colleagues in quantum computation and surrounding topics. It will be impossible to mention all the people that answered my endless questions, shared their knowledge with me, and helped me in one way or another. Just a few of them are Charlie Bennett, Ofer Biham, Gilles Brassard, Ike Chuang, David DiVincenzo, Arthur Ekert, Daniel Gottesman, Powel Horodecki, Peter Hoyer, Alesha Kitaev, Daniel Lidar, Nati Lineal, Michael Nielsen, Asher Peres, Yuval Peres, Sandu Popesco, Ran Raz, Eric Rains, Muli Safra, Peter Shor, John Smolin, Bob Solovay, Chaim Sompolinsky, Barbara Terhal, Lidror Troyanski, Boris Tsirelson and Umesh Vazirani.

I want to thank my companion, Ehud Friedgut, for the joy and happiness that we share together. Most of all, I thank my close family: my twin brother Niv, my sister Einat, and my parents, Dov and Tsipora Aharonov. The love and freedom that they gave me will be with me for life.

Contents

Summary

In the last few years, theoretical study of quantum systems serving as computational devices has achieved tremendous progress. We now have strong theoretical evidence that quantum computers, if built, might be used as a dramatically powerful computational tool, capable of performing tasks which seem intractable for classical computers. A major issue of interest in the area of quantum computation is that of understanding the effect of noise on the power of quantum computation. This question cannot be separated from that of quantum complexity, because any realistic model will inevitably be subject to such inaccuracies. In this thesis, I investigate the computational power of quantum computers subjected to noise.

The thesis begins with an overview on the area of quantum computation, in chapter I. I survey the origins and history of the area, and the major developments in the field. This is put in the context of basic notions in computational complexity theory and quantum mechanics.

In chapter II I set up the formal ground for the rest of the thesis, and describe the model of noisy quantum circuits, where I will use quantum circuits with mixed states. This model is equivalent to the standard model of quantum circuits with unitary operations, but its advantage is that unlike the standard model, it allows implementing noise in a natural way. This chapter discusses the problems and incompleteness in the standard model, and shows how to solve these issues in the model of quantum circuits with mixed states. It also includes definitions and analysis of metrics on the quantum states and quantum operations, which are useful when dealing with questions of errors in the computation. Finally, we implement noise into the model, and the important parameter $0 \leq \eta \leq 1$, called the "error rate" in the system, is given its meaning. The rest of the thesis deals with the two extreme cases on the noise scale: strong noise and weak noise.

We start with the case of strong noise in chapter III. The main lesson from this chapter is that when the noise exceeds a certain threshold, quantum computers no longer have advantage over classical computers. To show this, I describe a classical simulation of noisy quantum circuits, and analyze its cost, using branching processes and percolation techniques. An interesting aspect of the simulation is that its cost undergoes a phase transition from exponential to polynomial at a critical value of the strength of the noise. We thus find that when the noise is stronger than a certain threshold, there exists an efficient classical simulation of the quantum computer. Apart from the implications to the computational power of quantum systems, the phase transition seems to have interesting implications to the long standing question in physics regarding the transition from quantum to classical behavior. This will be discussed in the conclusions.

Chapter IV deals with the other regime of noise, the case of weak noise. Here I describe the most important result in the thesis, the "threshold theorem", which asserts that if the noise in the quantum circuit is smaller than the threshold error rate $\eta_c \simeq 10^{-6}$, then the effect of noise can be corrected, and the noisy quantum circuit is equivalent in computational power to an ideal quantum computer. The cost is polylogarithmic in space and time. The proof of the threshold result involves several ingredients. The basic notion is that of quantum error correcting codes; I explain what they are and define a new class of such codes, called polynomial codes. Next, I

show how to apply a universal set of gates fault tolerantly on encoded states. Geometrical and group theoretical arguments are used to prove the universality of the sets of gates. The key theorem in this chapter asserts that applying computation on encoded states recursively achieves fault tolerance against constant error rate, with probabilistic error. Some more effort is required to generalize this result to allow general noise, computation with any universal set of gates, and computation with only nearest neighbor gates. This chapter can be read as a self contained proof for fault tolerant universal quantum computation.

I tried to put all results in their context, asking what the implications to other issues in computer science and physics are. In the end of this thesis, in chapter V, I make these connections explicit, discussing the possible implications of these results to fundamental physical questions, such as the transition from quantum to classical physics. I also highlight the important open questions related to the questions discussed here.

Most of the results in this thesis were published elsewhere. The results and definitions of chapter II were first published in a joint paper with Alesha Kitaev and Noam Nisan [10]. Most of the results of chapter III were published in a joint paper with Michael Ben-Or [9], where the results concerning one dimensional circuits and percolation are still being written[13]. The threshold result in chapter IV was first published in a joint paper with Michael Ben-Or[8], where all the other results in this chapter will appear in the journal version of the same paper[8]. Both the introduction and the conclusions are slight modifications of the first and last chapters of a review paper I wrote[7].

Chapter I

Quantum Computation- An Overview

Since ancient times, humanity has been seeking tools to help us perform tasks which involve calculations. Such are computing the area of a land, computing the stresses on rods in bridges, or finding the shortest route from one place to another. A common feature of all these tasks is their structure:

Input ——> **Computation** ——> **Output**

The computation part of the process is inevitably performed by a dynamical physical system, evolving in time. In this sense, the question of what can be computed, is intermingled with the physical question of which systems can be physically realized. If one wants to perform a certain computation task, one should seek the appropriate physical system, such that the evolution in time of the system corresponds to the desired computation process. If such a system is initialized according to the input, its final state will correspond to the desired output.

A very nice such example was invented by Gaudí, a great Spanish architect, who lived around the turn of the century. His design of the holy family church, (*la sagrada familia*) in Barcelona is a masterpiece of art, and is still in the process of building, after almost a hundred years. The church resembles a sand palace, with a tremendous complexity of delicate thin but tall towers and arcs. Since the plan of the church was so complicated, towers and arcs emerging from unexpected places, leaning on other arcs and towers, it is practically impossible to solve the set of equations which corresponds to the requirement of equilibrium in this complex. Instead of solving this impossible task, Gaudí thought of the following ingenious idea: For each arc he desired in his complex, he took a rope, of length proportional to the length of the arc. He tied the edges of one rope to the middle of some other rope, or where the arcs were supposed to lean on each other. Then he just tied the edges of the ropes corresponding to the lowest arcs, to the ceiling. All the computation was instantaneously done by gravity! The set of arcs arranged itself such that the whole complex is in equilibrium, but upside down. Everything was there, the angles between the different arcs, the radii of the arcs. Putting a mirror under the whole thing, he could simply see the design of the whole church! [112].

Many examples of analog computers exist, which were invented to solve one complicated task. Such are the differential analyzer invented by Lord Kelvin in 1870[133], which uses friction, wheels, and pressure to draw the solution of an input differential equations. The spaghetti sort is another example, and there are many more[212].

Are these systems "computers"? We do not want to construct and build a completely different machine for each task that we have to compute. We would rather have a general purpose machine, which is "universal". A mathematical model for a "universal" computer was defined long before the invention of computers and is called the Turing machine[206]. Let me describe this model briefly. A Turing machine consists of an infinite tape, a head that reads and writes on the tape, a machine with finitely many possible states, and a transition function δ. Given what the head reads at time t, and the machine's state at time t, δ determines what the head will write, to which direction it will move one step and what will be the new machine's state at time $t+1$. The Turing machine model seems to capture the entire concept of computability, according to the following thesis[70]:

> **Church Turing Thesis:** A Turing machine can compute any function computable by a reasonable physical device

What does "reasonable physical device" mean? This thesis is a physical statement, and as such it cannot be proven. But one knows a physically unreasonable device when one sees it. Up till now there are no candidates for counterexamples to this thesis (but see Ref. [113]). All physical systems, (including quantum systems), seem to have a simulation by a Turing Machine.

It is an astonishing fact that there are families of functions which cannot be computed. In fact, most of the functions cannot be computed. There are trivial reasons for this: There are more functions than there are ways to compute them. The reason for this is that the set of Turing machines is countable, whereas the set of *families* of functions is not. In spite of the simplicity of this argument (which can be formalized using the *diagonal argument*) this observation came as a complete surprise in the 1930's when it was first discovered. The subject of computability of functions is a cornerstone in computational complexity. However, in the theory of computation, we are interested not only in the question of which functions can be computed, but mainly in the *cost* of computing these functions. The cost, or *computational complexity*, is measured naturally by the physical resources invested in order to solve the problem, such as time, space, energy, etc. A fundamental question in computation complexity is how the cost function behaves as a function of the input size, n, and in particular whether it is exponential or polynomial in n. In computer science problems which can only be solved in exponential cost are regarded intractable, and any of the readers who has ever tried to perform an exponentially slow simulation will appreciate this characterization. The class of tractable problems constitutes of those problems which have polynomial solutions.

It is worthwhile to reconsider what it means to *solve* a problem. One of the most important conceptual breakthroughs in modern mathematics was the understanding[180] that sometimes it is advantageous to relax the requirements that a solution be always correct, and allow some (negligible) probability for an error. This gave rise to much more rapid solutions to different problems, which make use of random coin flips, such as the Miller-Rabin randomized algorithm to test whether an integer is prime or not[82]. Here is a simple example of the advantage of probabilistic algorithms:

> we have access to a database of N bits, and we are told that they are either all equal, ("constant") or half are 0 and half are 1 ("balanced"). We are asked to distinguish between the two cases.

A deterministic algorithm will have to observe $N/2+1$ bits in order to always give a correct answer. To solve this problem probabilistically, toss a random i between 1 to N, observe the i'th bit, and repeat this experiment k times. If two different bits are found, the answer is "balanced", and if all bits are equal, the answer is "constant". Of course, there is a chance that we are wrong when declaring "constant", but this chance can be made arbitrarily small. The probability for an error equals the chance of tossing a fair coin k times and getting always 0, and it decreases exponentially with k. For example, in order for the error probability to be less than 10^{-10}, $k=100$ suffices. In general, for any desired confidence, a constant k will do. This is a very helpful shortcut if N is very large. Hence, if we allow negligible probability of error, we can do much better!

The class of tractable problems is now considered as those problems solvable with a negligible probability for error in polynomial time. These solutions will be computed by a probabilistic Turing machine, which is defined exactly as a deterministic Turing machine, except that the transition function can change the configuration in one of several possible ways, randomly. The modern Church thesis refines the Church thesis and asserts that the probabilistic Turing machine captures the entire concept of computational complexity:

> **The modern Church thesis**: A probabilistic Turing machine can simulate any reasonable physical device in polynomial cost.

It is worthwhile considering a few models which might seem to contradict this thesis at first sight. One such model is the DNA computer which enables a solution of NP-complete problems in polynomial time[6, 154]. However, the cost of the solution is exponential because the number of molecules in the system grows exponentially with the size of the computation. Vergis et al[212] suggested a machine which seems to be able to solve instantaneously an NP-complete problem using a construction of rods and balls, which is designed such that the structure moves according to the solution to the problem. A careful consideration[195] reveals that though we tend to think of rigid rods as transferring the motion instantaneously, there will be a time delay in the rods, which will accumulate and cause an exponential overall delay. Shamir[186] showed how to factorize an integer in polynomial time *and* space, but using another physical resource exponentially, namely precision. In fact, J. Simon showed that the class of problems called Polynomial space, which are harder than NP-complete problems, can be solved with polynomial cost in time and space[193], but with exponential precision. Hence all these suggestions for computational models do not provide counterexamples for the modern Church thesis, since they require exponential physical resources. However, note that all the suggestions mentioned above rely on classical physics.

In the early 80's Benioff[31, 32] and Feynman[103] started to discuss the question of whether computation can be done in the scale of quantum physics. In classical computers, the elementary information unit is a *bit*, i.e. a value which is either 0 or 1. The quantum analogue of a bit would be a two state particle, called a quantum bit or a **qubit**. A two state quantum system is described by a unit vector in the Hilbert space C^2, where C are the complex numbers. One of the two states will be denoted by $|0\rangle$, and corresponds to the vector $(1,0)$. The other state, which is orthogonal to the first one, will be denoted by $|1\rangle=(0,1)$. These two states constitute an orthogonal basis to the Hilbert space. To build a computer, we need to compose a large number of these two state particles. When n such qubits are composed to one system, their Hilbert space is the tensor product of n spaces: $C^2\otimes C^2\otimes\cdots\otimes C^2$. To understand this space better, it is best to think of it as the space spanned by its basis. As the natural basis for this space, we take the basis consisting of 2^n vectors, which is sometimes called the computational basis:

$$\begin{aligned}&|0\rangle \otimes |0\rangle \otimes \cdots \otimes |0\rangle \\ &|0\rangle \otimes |0\rangle \otimes \cdots \otimes |1\rangle \\ &\quad\vdots \\ &|1\rangle \otimes |1\rangle \otimes \cdots \otimes |1\rangle.\end{aligned} \tag{1}$$

Naturally classical strings of bits will correspond to quantum states:

$$i_1 i_2 ... i_n \longleftrightarrow |i_1\rangle \otimes |i_2\rangle \otimes \cdots \otimes |i_n\rangle \equiv |i_1 .... i_n\rangle \tag{2}$$

How can one perform computation using qubits? Suppose, e.g., that we want to compute the function $f : i_1 i_2 ... i_n \longmapsto f(i_1, ... i_n)$, from n bits to n bits. We would like the system to evolve according to the time evolution operator U:

$$|i_1 i_2 ... i_n\rangle \longmapsto U|i_1 i_2 ... i_n\rangle = |f(i_1, ... i_n)\rangle. \tag{3}$$

We therefore have to find the Hamiltonian $\mathcal{H}$ which generates this evolution according to Schrödinger's equation: $i\hbar \frac{d}{dt}|\Psi(t)\rangle = \mathcal{H}|\Psi(t)\rangle$. This means that we have to solve for $\mathcal{H}$ given the desired U:

$$|\Psi_f\rangle = \exp\left(-\frac{i}{\hbar}\int \mathcal{H} dt\right)|\Psi_0\rangle = U|\Psi_0\rangle \tag{4}$$

A solution for $\mathcal{H}$ always exists, as long as the linear operator U is unitary. It is important to pay attention to the unitarity restriction. Note that U will be unitary only if f is one-to-one, or reversible. Hence, reversible classical function can be implemented by a physical Hamiltonian. Researchers investigated the question of reversible classical functions in connection with completely different problems, e.g. the problem of whether computation can be done without generating heat (which is inevitable in irreversible operations) and as a solution to the "maxwell demon" paradox[149, 34, 35, 134]. It turns out that any classical function can be represented as a reversible function[150, 33] on a few more bits, and the computation of f can be made reversible without losing much in efficiency. Moreover, if f can be computed classically by polynomially many elementary reversible steps, the corresponding U is also decomposable into a sequence of polynomially many elementary unitary operations. We see that quantum systems can imitate all computations which can be done by classical systems, and do not lose much in efficiency.

Quantum computation is interesting not because it can imitate classical computation, but because it can probably do much more. In a seminal paper[102], Feynman pointed out the fact that quantum systems of n particles seem exponentially hard to simulate by classical devices. In other words, quantum systems do not seem to obey the modern Church thesis, i.e. they do not seem to be polynomially equivalent to classical systems! If quantum systems are hard to simulate, then quantum systems, harnessed as computational devices, might be dramatically more powerful than other computational devices.

Where can the "quantumness" of the particles be used? When I described how quantum systems imitate classical computation, the quantum particles were either in the state $|0\rangle$ or $|1\rangle$. However, quantum theory asserts that a quantum system, like Schrödinger's cat, need not be in one of the basis states $|0\rangle$ and $|1\rangle$, but can also be in a *linear superposition* of those. Such a superposition can be written as:

$$c_0|0\rangle + c_1|1\rangle \tag{5}$$

where c_0, c_1 are complex numbers and $|c_0|^2 + |c_1|^2 = 1$. The wave function, or superposition, of n such quantum bits, can be in a superposition of all of the 2^n possible basis states! Consider for example the following state of 3 particles, known as the GHZ state[120]:

$$\frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) \tag{6}$$

What is the superposition describing the first qubit? The answer is that there is no such superposition. Each one of the 3 qubits does not have a state of its own; the state of the system is not a tensor product of the states of each particle, but is some superposition which describes quantum correlations between these particles. Such particles are said to be quantumly *entangled*. The Einstein Podolski Rosen paradox[98], and Bell inequalities[29, 30, 76, 120], correspond to this puzzling quantum feature by which a quantum particle does not have a state of its own. Because of the entanglement or quantum correlations between the n quantum particles, the state of the system cannot be specified by simply describing the state of each of the n particles. Instead, the state of n quantum bits is a complicated superposition of all 2^n basis states, so 2^n complex coefficients are needed in order to describe it. This exponentiality of the Hilbert space is a crucial ingredient in quantum computation. To gain more understanding of the advantages of the exponentiality of the space, consider the following superposition of n quantum bits.

$$\frac{1}{\sqrt{2^n}} \sum_{i_1, i_2, \ldots, i_n = 0}^{1} |i_1, i_2, \ldots, i_n\rangle \tag{7}$$

This is a uniform superposition of all possible basis states of n qubits. If we now apply the unitary operation which computes f, from equation 3, to this state, we will get, simply from linearity of quantum mechanics:

$$\frac{1}{\sqrt{2^n}} \sum_{i_1, i_2, \ldots, i_n = 0}^{1} |i_1, i_2, \ldots, i_n\rangle \longmapsto \frac{1}{\sqrt{2^n}} \sum_{i_1, i_2, \ldots, i_n = 0}^{1} |f(i_1, i_2, \ldots, i_n)\rangle. \tag{8}$$

Applying U once computes f simultaneously on all the 2^n possible inputs i, which is an enormous power of parallelism!

It is tempting to think that exponential parallelism immediately implies exponential computational power, but this is not the case. In fact, classical computations can be viewed as having exponential parallelism as well. The problem lies in the question of how to extract the exponential information out of the system. In quantum computation, in order to extract information one has to *observe* the system. The measurement process causes the famous *collapse of the wave function*. In a nutshell, this means that after the measurement the state is projected to only one of the exponentially many possible states, so that the exponential amount of information which has been computed is completely lost! In order to gain advantage of exponential parallelism, one needs to combine it with another quantum feature, known as interference. Interference allows the exponentially many computations done in parallel to cancel each other, just like destructive interference of waves or light. The goal is to arrange the cancellation such that only the computations which we are interested in remain, and all the rest cancel out. The combination of exponential parallelism and interference is what makes quantum computation powerful, and plays an important role in quantum algorithms.

A quantum algorithm is a sequence of elementary unitary steps, which manipulate the initial quantum state $|i\rangle$ (for an input i) such that a measurement of the final state of the system yields the correct output. The first quantum algorithm which combines interference and exponentiality to solve a problem faster than classical computers, was discovered by Deutsch and Jozsa[89]. This algorithm addresses the problem we have encountered before in connection with probabilistic algorithms: Distinguish between "constant" and "balanced" databases. The quantum algorithm solves this problem *exactly*, in polynomial cost. As we have seen, classical computers cannot do this, and must release the restriction of exactness. Deutsch and Jozsa made use of the most powerful tool in quantum algorithms, the *Fourier transform*, which indeed manifests interference and exponentiality. Simon's algorithm[194] uses similar techniques, and was the seed for the most important quantum algorithm known today: Shor's algorithm.

Shor's algorithm (1994) is a polynomial quantum algorithm for factoring integers, and for finding the logarithm over a finite field[189]. For both problems, the best known classical algorithms are exponential. However, there is no proof that classical efficient algorithms do not exist. Shor's result is regarded as extremely important both theoretically and practically, mainly due to the fact that the assumption that factorization is hard lies in the heart of the RSA cryptographic system [182, 82]. A cryptosystem is supposed to be a secure way to transform information such that an eavesdropper will not be able to learn in reasonable time significant information about the message sent. The RSA cryptosystem is used very heavily: The CIA uses it, the security embedded into Netscape and the Explorer Web browsers is based on RSA, banks use RSA for internal security as well as securing external connections. However, RSA can be cracked by any one who has an efficient algorithm for factoring. It is therefore understandable why the publication of the factorization algorithm caused a rush of excitement all over the world.

It is important that the quantum computation power does not rely on unreasonable precision but a polynomial amount of precision in the computational elements is enough[42]. This means that the new model requires physically reasonable resources, in terms of time, space, and precision, but yet it is (possibly) exponentially stronger than the ordinary model of probabilistic Turing machine. As such, it is the only model which really threatens the modern Church thesis.

There are a few major developing directions of research in the area of quantum computation. In 1995 Grover[122] discovered an algorithm which searches an unsorted database of N items and finds a specific item in $\sqrt{N}$ time steps. This result is surprising, because intuitively, one cannot search the database without going through all the items. Grover's solution is quadratically better than any possible classical algorithms, and was followed by numerous extensions and applications[49, 123, 124, 96, 53, 54], all achieving polynomial advantage over classical algorithms. A promising new branch in quantum complexity theory is the study of a class of problems which is the quantum analog of the complexity class NP[139]. Another interesting direction in quantum computation is concerned with quantum computers simulating efficiently other physical systems such as many body Fermi systems[222, 2, 216, 47]. This direction pursues the original suggestion by Feynman[102], who noticed that quantum systems are hard to simulate by classical devices. An important direction of investigation is the search for a different, perhaps stronger, quantum computation model. For example, consider the introduction of slight non-linearities into quantum mechanics. This is completely hypothetical, as all experiments verify the linearity of quantum mechanics. However, such slight non linearities would imply extremely strong quantum algorithms[3]. A very interesting quantum computation model which is based on anyons, and uses non-local features of quantum mechanics, was suggested by Kitaev[138]. A possibly much stronger model, based on quantum field theory, was sketched recently by Freedman, but it has not been rigorously defined yet[107]. One other direction is oracle results in quantum complexity. This direction compares quantum complexity power and classical complexity power when the two models are allowed to have access to an oracle, i.e. a black box which can compute a certain (possibly difficult) function in one step [42, 40, 44, 45]. In fact, the result of Bernstein and Vazirani[42] from 1993 demonstrating a superpolynomial gap between quantum and classical computational complexity with an access to a certain oracle initialized the sequence of results leading to the Shor's algorithm. An important recent result[27] in quantum complexity shows that quantum computers have no more than polynomial advantage in terms of number of accesses to the inputs. As of now, we are very far from understanding the computational power of quantum systems. In particular, it is not known whether quantum systems can efficiently solve NP complete problems or not.

Quantum information theory, a subject which is intermingled with quantum computation, provides a bunch of quantum magic tricks, which might be used to construct more powerful quantum algorithms. Probably the first "quantum pearl" that one encounters in quantum mechanics is the Einstein Podolsky Rosen paradox, which, as is best explained by Bell's inequalities, establishes the existence of correlations between quantum particles, which are stronger than any classical model can explain. Another "quantum pearl" which builds on quantum entanglement, is teleportation[38]. This is an amazing quantum recipe which enables two parties (Alice and Bob) which are far apart, to transfer an unknown quantum state of a particle in Alice's hands onto a particle in Bob's hand, without sending the actual particle. This can be done if Alice and Bob share a pair of particles which interacted in the past and therefore are quantumly entangled. Such quantum effects already serve as ingredients in different computation and communication tasks. Entanglement can be used, for example, in order to gain advantage in communication. If two parties, Alice and Bob, want to communicate, they can save bits of communication if they share entangled pairs of qubits[77, 57, 78, 18]. Teleportation can be viewed as a quantum computation[55], and beautiful connections were drawn[39] between teleportation and quantum algorithms which are used to correct quantum noise. All these are uses of quantum effects in quantum computation. However, I believe that the full potential of quantum mechanics in the context of complexity and algorithmic problems is yet to be revealed.

Despite the impressive progress in quantum computation, a menacing question still remained. Quantum information is extremely fragile, due to inevitable interactions between the system and its environment. These interactions cause the system to lose part of its quantum nature, a process called *decoherence*[202, 224]. In addition, quantum elementary operations (called *gates*) will inevitably suffer from inaccuracies. Will physical realizations of the model of quantum computation still be as powerful as the ideal model? In classical computation, it was already shown by von-Neumann[167] how to compute when the elements of the computation are faulty, using redundant information. Indeed, nowadays error corrections are seldom used in computers because of extremely high reliability of the elements, but quantum elements are much more fragile, and it is almost certain that quantum error corrections will be necessary in future quantum computers. It was shown that if the errors are not corrected during quantum computation, they soon accumulate and ruin the entire computation[65, 66, 21, 163]. Hence, a method to correct the effect of quantum noise is necessary. Physicists were pessimistic about the question of whether such a correction method exists[148, 207]. The reason is that quantum information in general cannot be cloned[92, 219, 24], and so the information cannot be simply protected by redundancy, as is done classically. Another problem is that in contrast to the discreteness of digital computers, a quantum system can be in a superposition of eigenstates with continuous coefficients. Since the range of allowed coefficients is continuous, it seems impossible to distinguish between bona fide information and information which has been contaminated.

As opposed to the physical intuition, it turns out that clever techniques enable quantum information to be protected. The conceptual breakthrough in quantum error corrections was the understanding that quantum errors, which are continuous, can be viewed as a discrete process in which one out of four quantum operations occurs. Moreover, these errors can be viewed as classical errors, called bit flips, and quantum errors, called phase flips. Bit flip errors can be corrected using classical error correction techniques. Fortunately, phase flips transform to bit flips, using the familiar Fourier transform. This understanding allowed using classical error correction codes techniques in the quantum setting. Shor was the first to present a scheme that reduces the affect of noise and inaccuracies, building on the discretization of errors[190]. As in classical error correcting codes, quantum states of k qubits are *encoded* on states of more qubits. Spreading the state of a few qubits on more qubits, allows correction of the information, if part of it has been contaminated. These ideas were extended [59, 198] to show that a quantum state of k qubits can be encoded on n qubits, such that if the n qubits are sent through a noisy channel, the original state of the k qubits can be recovered. k/n tends asymptotically to a constant *transmission rate* which is non zero. This is analogous to Shannon's result from noisy classical communication[188]. Many different examples of quantum error correcting codes followed[199, 147, 67, 144, 181, 151], and a group theoretical framework for most quantum codes was established[61, 60, 116].

Performing quantum computation fault tolerantly is more complicated than simply protecting quantum information which is sent through a noisy quantum channel. Naturally, to protect the information we would compute on encoded states. There are two problems with noisy computation on encoded states. The first is that the error correction is done with faulty gates, which cause errors themselves[23]. We should be careful that the error correction does not cause more harm than it helps. The second problem is that when computing on encoded states, qubits interact with each other through the gates, and this way errors can *propagate* through the gates, from one qubit to another. The error can spread in this way to the entire set of qubits very quickly. In order to deal with these problems, the idea is to perform computation and error correction in a *distributed manner*, such that each qubit can effect only a small number of other qubits. Kitaev[137] showed how to perform the computation of error correction with faulty gates. Shor[191] discovered how to perform a general computation in the presence of noise, but only under the restriction that the noise decreases (slowly) with the size of the computation. Shor's assumption is not physically reasonable. What seems to be reasonable to assume is that the devices we will use in the laboratory have some finite precision, independent of the size of the computation. In a paper by myself and Ben-Or[8] we show how to compute fault tolerantly in the presence of constant noise rate. The idea is to apply a concatenation of Shor's scheme. We encode the state once, and then encode the encoded state, and so on for for several levels. This idea was discovered independently by other groups as well [140, 141, 117, 8, 138, 177]. Using this technique, it is possible to perform resilient quantum computation for polynomially many time steps, if the noise is smaller than a certain *threshold.* This result is called the "threshold result", and is fully described in chapter IV. Decoherence and imprecision are therefore no longer considered insurmountable obstacles to realizing a quantum computation.

In accord with these theoretical optimistic results, attempts at implementations of quantum circuits are now being carried out all over the world. However, the progress in this direction is much slower than the impressive pace in which theoretical quantum computation has progressed. The reason is that handling quantum systems experimentally is extremely difficult. Entanglement is a necessary ingredient in quantum computers, but experimentally, it is a fragile property which is difficult to create and preserve. So far, entangled pairs of photons were created successfully[146, 203], and entanglement features such as violation of Bell inequalities were demonstrated [15, 16]. Even entangled pairs of atoms were created[126]. However quantum computation seems to be advantageous only when macroscopically many particles are entangled[131, 9], a task which seems impossible as of now. Promising experimental developments come from the closely related subject of quantum cryptography[56, 38, 52]. Quantum communication was successfully tested[129, 161]. Bouwmeester et.al have recently reported on experimental realization of quantum teleportation[48]. Suggestions for implementations of quantum computation [71, 83, 114, 156, 94, 130, 41, 159, 130, 174, 178, 200] include quantum dots, cold trapped ions and nuclear magnetic resonance, and some of these suggestions were already implemented [165, 205, 161, 114, 84]. Unfortunately, these implementations were so far limited to three qubits. With three qubits it is possible to perform partial error correction, and successful implementation of error correction of phases using NMR was reported[85, 68]. Using nuclear magnetic resonance techniques, a quantum algorithm was implemented which achieves proven advantage over classical algorithms[69]. It should be noted, however, that all these suggestions for implementation suffer from severe problems. Some of them do not allow parallel operations, which are necessary for error resilience, as we will see in chapter III. In all the above systems controlling more than 10 qubits looks like a dream. Never the less, these experimental successes encourage our hope that the ambitious task of realizing quantum computation might be possible.

The exciting developments in quantum computation give rise to deep new open questions in both the fields of computer science and physics. In particular, computational complexity questions shed new light on old questions in fundamental quantum physics such as the transition from quantum to classical physics, and the measurement process. Let me delay the discussion of these interesting topics to the end of the thesis, and go on to investigate different aspects concerning the topic of the thesis: Noisy quantum computation.

Chapter II

The Model of Noisy Quantum Circuits

1 Summary

In this chapter I set up the formal ground for the rest of the thesis, and define the model of noisy quantum circuits with mixed states.

Let us first recall the definition of standard quantum circuits, due to Deutsch [88] and Yao[221]. The device operates on n quantum-bits ("qubits"). Each one of the 2^n possible Boolean configurations $i \in \{0,1\}^n$ of these bits denotes a *basic state* $|i\rangle$. The state of the computation at any point in time is a *superposition* of basic states $\sum_{i\in\{0,1\}^n} c_i|i\rangle$, where the c_i's are complex numbers satisfying $\sum_i |c_i|^2 = 1$ (i.e. $||c|| = ||c||_2 = 1$). These superpositions are called *pure states*. Each computational operation is (1) *local:* i.e. involves only a constant number of qubits (2) *unitary:* i.e. maintains $||c|| = 1$. At the end of the computation a *measurement* of one qubit is made, which returns a Boolean value "true" with probability $\sum_{i,|i\rangle\in M} |c_i|^2$, where M is a subspace of $\mathcal{C}^{(\{0,1\}^n)}$ specified by the measurement. The state changes (in a well defined way) according to the outcome of this measurement, and becomes a probability distribution over pure states.

Let us consider the model described above. During the computation, the operations must be unitary, and the state must be a pure state. But at the end of the computation, a non-unitary operation, a measurement, is applied, and the state becomes a probabilisty distribution over pure states, namely a "mixed state". So we find out that in quantum physics operations might also be non-unitary, and states are not nessecarily pure states. Restricting the model to unitary gates and pure states seems arbitrary. It is natural to ask: what is the most general model that captures quantum physics? It turns out that the most general quantum state is a probability distribution over vectors in the Hilbert space, i.e. a "mixed state". The description of the state of the circuit by a probability distribution over pure states is not unique, and physicists use an alternative unique description, namely density matrices, which provides many conceptual and practical advantages. I describe here the model of a quantum circuit which is allowed to be in a mixed state, and which is allowed to use any quantum operation as a gate, not necessarily unitary.

The first result in this chapter is a simple corollary of known results in physics[127]:

Theorem 1 *The model of quantum circuits with mixed states is polynomially equivalent in computational power to the standard unitary model.*

There are several key issues, which appear to be inconvenient, difficult, or impossible to deal with inside the standard, unitary model. In the non-unitary model of quantum circuits with mixed states, these problems essencialy disappear.

- **Measurements in the middle of computation**: it has been often remarked, and implicitly used (e.g. Shor's algorithm), that quantum computations may allow measurements in the middle of the computation. However, the state of the computation after a measurement is a mixed state, which is not allowed in the standard model. This problem no longer exists in our model.
- **Noise and Decoherence:** Noise and decoherence are key obstacles in implementing quantum computers devices. Recent results show that theoretically, fault tolerant computation exists[191, 8, 140, 135]. Still, in the task of realizing quantum computers, it is likely that more theoretical work will be needed in this direction. A key problem in this interface between quantum physics and quantum computation models is the fact that quantum noise, and in particular, decoherence, are non-unitary operations that cause a pure state to become a mixed state. Incorporating quantum noise, which was impossible in the standard unitary model, is naturally done in our model.

The main technical result in this chapter is a soulution of one more problem in the unitary model, namely the **subroutine problem**. A cornerstone of computer science is the notion of using subroutines (or oracles, or reductions): once we are able to perform an operation A within our model, we should be able to use A as a "black box" in further computations. The general and natural function that a quantum computer outputs is a *probabilistic function:* for an input X, the output is distributed according to a distribution, D_X, which depends on the input. When using such a probabilistic function as a subroutine, the state is affected in a non-unitary manner. Therefore, using subroutines in their full generality was not previously dealt with, the reason being that in the language of pure states, this simply can not be done. This is an incompleteness of the standard model, because computatble functions cannot be used as "black boxes".

The special case of using deterministic subroutines was defined in[40], and it was shown that using determinstic subroutines does not strengthen the model. As to probabilistic subroutines, these were implicitly used in quantum algorithms,(e.g. Shor's algorithm,) but always on a classical input, for this case can be easily understood. A conceptual difficulty lies in the combination of applying *probabilistic* subroutines to quantum *superpositions.* It is not clear what the natural definition should be. Here, we are able to give a natural definition which generalizes both the case of deterministic subroutines on superpositions, and the case of probabilistic subroutines on classical inputs. We prove that using general subroutines does not strengthen the model. Let us define FQP to be the set of probabilistic fuctions computed by uniform quantum circuits with polynomial size and depth. The result can be stated as:

Theorem 2 $FQP^{FQP} = FQP$.

Hopefully, the fact that probabilistic functions can be used as subroutines might prove to be useful in constructing quantum algorithms.

As in any model of computation, we would like to understand how certain amount of noise affects the correctness of the computation. In order to deal with such questions, one needs appropriate tools: metrics on the state of the system and the gates. In [10] such metrics are suggested and discussed. For probabilistic functions, we use a natural metric,relying on total variation distances. As a metric on density matrices, we propose to use the *trace metric,* induced by the *trace norm*: $\|H\| = \sum_i |\lambda_i|$, where λ_i are the eigenvalues of H. We show that it is an appropriate metric since it quantifies the *measurable distance between two quantum states.* We also define the diamond metric on quantum gates which has very nice properties. These metrics allows one to keep track of errors in the noisy circuit. We establish the fact that errors in subroutines and gates contribute at most an additive error to the function computed by the circuit.

Given the above model, it is possible to implement any general noise processes and define noisy quantum circuits. I will give a definition of a very general noise model, where the main restriction is that the noise is local, meaning that there are no correlations between errors in different locations in the circuit. This model includes probabilistic errors, decoherence, systematic inaccuracies in gates, and more. This will complete the definition of the model of noisy quantum circuits.

2 Some Useful Physics Background

The model of Quantum computers is based on the rules of quantum mechanics, and good references are Cohen-Tanudji[80], Sakurai [184], and Peres[175].

2.1 Pure States

A quantum physical system in a *pure state* is described by a unit vector in a Hilbert space, i.e a vector space with an inner product. In the *Dirac* notation a pure state is denoted by $|\alpha\rangle$. The physical system which corresponds to a quantum circuit consists of n quantum two-state particles, and the Hilbert space of such a system is $\mathcal{H}_2^n = \mathcal{C}^{\{0,1\}^n}$ i.e. a 2^n dimensional complex vector space. $\mathcal{H}_2^n$ is viewed as a tensor product of n Hilbert spaces of one two-state particle: $\mathcal{H}_2^n = \mathcal{H}_2 \otimes \mathcal{H}_2 \otimes \ldots \otimes \mathcal{H}_2$. The k'th copy of $\mathcal{H}_2$ will be referred to as the k'th **qubit**. We choose a special basis for $\mathcal{H}_2^n$, which is called the computational basis. It consists of the 2^n orthogonal states: $|i\rangle, 0 \le i < 2^n$, where i is in binary representation. $|i\rangle$ can be seen as a tensor product of states in $\mathcal{H}_2$: $|i\rangle = |i_1\rangle|i_2\rangle....|i_n\rangle = |i_1 i_2..i_n\rangle$, where each i_j gets 0 or 1. Such a state, $|i\rangle$, corresponds to the j'th particle being in the state $|i_j\rangle$. A pure state $|\alpha\rangle \in \mathcal{H}_2^n$ is generally a *superposition* of the basis states: $|\alpha\rangle = \sum_{i=1}^{2^n} c_i|i\rangle$, with $\sum_{i=1}^{2^n} |c_i|^2 = 1$. A vector in $\mathcal{H}_2^n$, $v_\alpha = (c_1, c_2, ..., c_{2^n})$, written in the computational basis representation, with $\sum_{i=1}^{2^n} |c_i|^2 = 1$, corresponds to the pure state: $|\alpha\rangle = \sum_{i=1}^{2^n} c_i|i\rangle$. $v_\alpha^\dagger$, the transposed-complex conjugate of v_α, is denoted $\langle\alpha|$. The inner product between $|\alpha\rangle$ and $|\beta\rangle$ is denoted $\langle\alpha|\beta\rangle = (v_\alpha^\dagger, v_\beta)$. The matrix $v_\alpha^\dagger v_\beta$ is denoted as $|\alpha\rangle\langle\beta|$.

2.2 Mixed States

In general, a quantum system is not in a pure state. This may be attributed to the fact that we have only partial knowledge about the system, or that the system is not isolated from the rest of the universe. We say that the system is in a *mixed state*, and assign with the system a probability distribution, or *mixture* of pure states, denoted by $\{\alpha\} = \{p_k, |\alpha_k\rangle\}$. This means that the system is with probability p_k in the pure state $|\alpha_k\rangle$. This description is not unique, as different mixtures might represent the same physical system. As an alternative description, physicists use the notion of **density matrices**, which is an equivalent description but has many advantages. A density matrix ρ on $\mathcal{H}_2^n$ is an hermitian (i.e. $\rho = \rho^\dagger$) semi positive definite matrix of dimension $2^n \times 2^n$ with trace $\mathrm{Tr}(\rho) = 1$. A pure state $|\alpha\rangle = \sum_i c_i|i\rangle$ is represented by the density matrix: $\rho_{|\alpha\rangle} = |\alpha\rangle\langle\alpha|$, i.e. $\rho_{|\alpha\rangle}(i,j) = c_i c_j^*$. (By definition, $\rho(i,j) = \langle i|\rho|j\rangle$). A mixture $\{\alpha\} = \{p_l, |\alpha_l\rangle\}$, is associated with the density matrix $\rho_{\{\alpha\}} = \sum_l p_l|\alpha_l\rangle\langle\alpha_l|$. This association is not one-to-one, but it is **onto** the density matrices, because any density matrix describes the mixture of it's eigenvectors, with the probabilities being the corresponding eigenvalues. Note that diagonal density matrices correspond to probability distributions over classical states. Density matrices are linear operators on their Hilbert spaces.

2.3 Operations on Quantum States

Transformations of density matrices are linear operators on operators (sometimes called *super-operators*). Any physically allowed super-operator $T : \mathbf{L}(\mathcal{N}) \to \mathbf{L}(\mathcal{M})$ sends density matrices to density matrices. This is equivalent to say that T is positive and trace-preserving. (A super-operator is called *positive* if it sends positive semi-definite Hermitian matrices to positive semi-definite Hermitian matrices). However, this is not enough for a super-operator to be physically allowed. The positivity must remain if we extend the spaces $\mathcal{N}$ and $\mathcal{M}$ by adding more qubits. That is, the super-operator $T \otimes \mathrm{I}_{\mathcal{F}}$ must be positive, where $\mathrm{I}_{\mathcal{F}}$ is the identity super-operator on an arbitrary finite-dimensional Hilbert space $\mathcal{F}$. Such T is called a *completely positive map*. Hence physically allowed quantum operations are linear trace preserving completely positive maps. Clearly, linear operations on mixed states preserve the probabilistic interpretation of the mixture, because

$$T \circ \rho = T \circ (\sum_l p_l|\alpha_l\rangle\langle\alpha_l|) = \sum_l p_l T \circ (|\alpha_l\rangle\langle\alpha_l|). \tag{1}$$

A very important example of a quantum super-operator is the super-operator corresponding to the standard unitary transformation on a Hilbert space $\mathcal{N}$, $|\alpha\rangle \mapsto U|\alpha\rangle$, which sends a quantum state $\rho = |\alpha\rangle\langle\alpha|$ to the state $U\rho U^\dagger$. Another very important example is a *unitary embedding* $V : \mathcal{N} \to \mathcal{M}$. This defines the super-operator $T : \rho \mapsto V\rho V^\dagger$. A unitary embedding naturally appears when we add a blank qubit to the system, $V_0 : |\xi\rangle \mapsto |\xi\rangle \otimes |0\rangle : \mathbf{C}^{2^n} \to \mathbf{C}^{2^{n+1}}$. Another important super-operator is discarding a set of qubits. A density matrix of n qubits can be reduced to a subset, A, of $m < n$ qubits. We say that the rest of the system, represented by the Hilbert space $\mathcal{F} = \mathbf{C}^{2^{n-m}}$, is *traced out*, and denote the new matrix by $\rho|_A = \mathrm{Tr}_{\mathcal{F}}\,\rho$, where $\mathrm{Tr}_{\mathcal{F}} : \mathbf{L}(\mathcal{N} \otimes \mathcal{F}) \to \mathbf{L}(\mathcal{N})$ is defined by: $\rho|_A(i,j) = \sum_{k=1}^{2^{n-m}} \rho(ik, jk)$, and k runs over a basis for $\mathcal{F}$. In words, it means averaging over $\mathcal{F}$, and is sometimes called a partial trace. If the state of the qubits which are traced out is in tensor product with the state of the other qubits, then discarding the qubits means simply erasing their state. However, if the state of the discarded qubits is not in tensor product with the rest, the reduced density matrix is always a mixed state. Any quantum operation which does not operate on $\mathcal{F}$ commutes with this tracing out.

The following lemma provides the link between super-operators and standard unitary operations. It turns out that any super-operator from n to m qubits can be constructed from adding $n + m$ blank qubits to the system, applying a unitary matrix on the $2n+m$ qubits and then tracing out $2n$ qubits.

Lemma 1 *(modification of Choi (1970),Hellwig and Kraus(1975), and Schumacher (1996)[127]): The following conditions are equivalent:*

1. *A super-operator $T : \mathbf{L}(\mathcal{N}) \to \mathbf{L}(\mathcal{M})$ is trace-preserving and completely positive.*
2. *There is a Hilbert space $\mathcal{F}$ with $\dim(\mathcal{F}) \leq \dim(\mathcal{N})\dim(\mathcal{M})$, and a unitary embedding $V : \mathcal{N} \to \mathcal{N} \otimes \mathcal{F}$ such that $T\rho = \mathrm{Tr}_{\mathcal{F}}(V\rho V^{\dagger})\ \forall \rho \in \mathbf{L}(\mathcal{N})$.*

Super-operators can be extended to operate on larger spaces by taking tensor product with the identity operator: $T : \mathbf{L}(\mathcal{N}) \to \mathbf{L}(\mathcal{M})$ will be extended to $T \otimes I_{\mathcal{R}} : \mathbf{L}(\mathcal{N} \otimes \mathcal{R}) \to \mathbf{L}(\mathcal{M} \otimes \mathcal{R})$. Usually, we will be interested in those super-operators that are extensions of super-operators on spaces with small dimensionality. These will correspond to local gates later on.

In order to describe the operation of super-operators on density matrices $\in \mathbf{L}(\mathcal{N})$ is suffices, from linearity, to specify what happens to a basis which spans the density matrices: $\{|i\rangle\langle j|\}_{i,j}$ where $|i\rangle, \langle j|$ run over all basis vectors of $\mathcal{N}$. (Any density matrix can be written as $\rho = \sum_{i,j} \rho_{i,j} |i\rangle\langle j|$.) For example the unitary operation $|i\rangle \longmapsto |v_i\rangle$ corresponds to the super-operator specified by $|i\rangle\langle j| \longmapsto |v_i\rangle\langle v_j|$. If U is extended to $\in \mathbf{L}(\mathcal{N} \otimes \mathcal{M})$ $|i,k\rangle\langle j,l| = |i\rangle\langle j| \otimes |k\rangle\langle l| \longmapsto (|v_i\rangle\langle v_j|) \otimes (|k\rangle\langle l|)$.

2.4 Measurements

A quantum system can be **measured**, or observed. Let us consider a set of positive semi-definite Hermitian operators $\{P_m\}$, such that $\sum_m P_m = I$. The measurement is a process which yields a probabilistic classical output. For a given density matrix ρ, the output is m with probability $\Pr(m) = \mathrm{Tr}(P_m \rho)$.

We will use only *projection* measurements. Namely, we assume that P_m are orthogonal projections onto mutually orthogonal subspaces S_m which span the whole space $\mathcal{N} = \mathbf{C}^{2^n}$, i.e. $\mathcal{N} = \bigoplus_m S_m$. A more particular type of measurement which we will be using a lot is a basic measurement of r qubits. In this case, P_m (with $1 \leq i \leq 2^r$) are the projection on the subspace S_m, which is the subspace spanned by basic vectors on which the measured qubits have the values corresponding to the string m: $S_m = \mathrm{span}\{\,|m,j\rangle,\ j = 1,\ldots,2^{n-r}\}$. This process corresponds to measuring the value of r qubits, in the basic basis, where here, for simplicity, we considered measuring the first r qubits.

The classical result of a measurement, m, can be represented by the density matrix $|m\rangle\langle m|$ in an appropriate Hilbert space $\mathcal{M}$. The state of the quantum system after a projection measurement is also defined; it is equal to $\Pr(m)^{-1} P_m \rho P_m$. (It has the same meaning as a conditional probability distribution). Thus, the projection measurement can be described by a super-operator T which maps quantum states on the space $\mathcal{N}$ to quantum states on the space $\mathcal{N} \otimes \mathcal{M}$, the result being diagonal with respect to the second variable:

$$T\rho \;=\; \sum_m (P_m \rho P_m) \otimes \Big(|m\rangle\langle m|\Big) \tag{2}$$

3 Quantum Circuits with Mixed States

We define here a model of quantum circuits, using density matrices. This enables us to apply general non-unitary gates. The circuit is defined to compute probabilistic functions, which are a generalization of Boolean functions computed by a standard quantum circuit. A quantum gate is defined to be the most general quantum operation:

Definition 1 *A* **quantum gate***, g, of order (k, l) is a trace preserving, completely positive, linear map from density matrices on k qubits to density matrices on l qubits. Its action on the density matrices is denoted as follows: $\rho \longmapsto g \circ \rho$. (The "$\circ$" sign is used for clarity and could be omitted).*

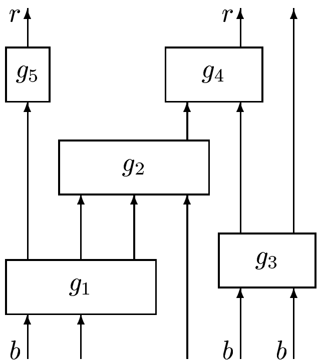
The unitary gate, U, of the standard model is a special case of a quantum gate. The corresponding super-operator is $U \cdot U^\dagger$. Using our "$\circ$" notation, we can denote it simply by U with no danger of confusion. Thus, $U \circ \rho = U\rho U^\dagger$.

A measurement is also a special case of a quantum gate — the probabilistic projection onto a set of mutually orthogonal subspaces. Besides changing the state of the qubits, it produces a classical probabilistic result. (As shown above, both results can be described by a joint density matrix, but it is better to take the advantage of the second result being classical).

We now define a quantum circuit:

Definition 2 *Let $\mathcal{G}$ be a family of quantum gates.* **A Quantum circuit that uses gates from** $\mathcal{G}$ *is a directed acyclic graph. Each node v in the graph is labeled by a gate $g_v \in \mathcal{G}$ of order (k_v, l_v). The in-degree and out-degree of v are equal k_v and l_v, respectively. An arbitrary subset of the inputs are labeled "blank". An arbitrary subset of the outputs are labeled "result".*

Here is a schematic example of such a circuit. We use "b", "r" for "blank", "result".



The circuit operates on a density matrix as follows:

Definition 3 final density matrix*: Let Q be a quantum circuit. Choose a topological sort for Q: $g_t \ldots g_1$. Where g_j are the gates used in the circuit. The final density matrix for an initial density matrix ρ is $Q \circ \rho = g_t \circ \ldots \circ g_2 \circ g_1 \circ \rho$.*

Usually there exists more than one topological sort for a given circuit Q. Yet $Q \circ \rho$ is well defined and operations of the gates in a quantum circuit determined by two different topological sorts are equivalent:

Lemma 2 *$g_t \circ ... \circ g_2 \circ g_1 \circ \rho = g_{\sigma(t)} \circ ... \circ g_{\sigma(2)} \circ g_{\sigma(1)} \circ \rho$, where σ is a permutation, If the two orderings are two topological sorts of the same quantum circuit.*

The reason for this is that two gates that operate on different qubits commute. The proof of lemma 2 easily follows from:

Lemma 3 *Let g_1, g_2 be two quantum gates operating on different qubits. Then $\forall\ \rho$, $g_1 \circ g_2 \circ \rho = g_2 \circ g_1 \circ \rho$.*

Proof: To extend the gates to operate on the whole set of n qubits, we tensor with the identity. For simplicity, let us assume that g_1 operates on the first k_1 qubits (the Hilbert space $\mathcal{N}$), and g_2 operates on some of the rest of the qubits (the Hilbert space $\mathcal{M}$). We only need to show that $(g_1 \otimes \mathrm{I}_{\mathcal{M}}) \circ (\mathrm{I}_{\mathcal{N}} \otimes g_2) = g_1 \otimes g_2$ and $(\mathrm{I}_{\mathcal{N}} \otimes g_2) \circ (g_1 \otimes \mathrm{I}_{\mathcal{M}}) = g_1 \otimes g_2$. These are simply particular cases of the identity

$$(X_1 \otimes X_2)\,(Y_1 \otimes Y_2) \;=\; (X_1 Y_1) \otimes (X_2 Y_2) \tag{3}$$

where X_1, Y_1 and X_2, Y_2 act on arbitrary linear two spaces. (The "$\circ$" signs are omitted here). $\Box$

Now we are ready to define the function that the circuit computes. The circuit produces a probability distribution over strings of r bits, which depends on it's input string. The probability distribution is computed out of the final density matrix, and is the same probability distribution as one would get over the strings of outcomes, if at the end of the computation we apply a basic measurement of all the "result" qubits.

Definition 4 Computed function: *Let Q be a quantum circuit, with n inputs and r "result" outputs. The probabilistic function that Q computes, $f_Q = f$: $\{0,1\}^n \longmapsto [0,1]^{\{0,1\}^r}$, is defined as follows: For input i, the probability for output j is $f_{i,j} = \langle j| \left(Q \circ |i\rangle\langle i|\right)\Big|_A |j\rangle$, where A is the set of the "result" outputs.*

4 Equivalence to the Standard Model

General non-unitary gates can be replaced by unitary gates, as is shown by the following lemma:

Lemma 4 *Let g be a quantum gate of order (n,m). There exists a unitary quantum gate U_g on $2n+m$ qubits which satisfies: For any ρ, $g \circ \rho = \left(U_g \circ \left(\rho \otimes |0^{n+m}\rangle\langle 0^{n+m}|\right)\right)\Big|_A$, where A is the set of the first m qubits.*

Proof: This lemma follows from lemma 1. By definition 1, the gate g is a trace-preserving completely positive super-operator $\mathbf{L}(\mathcal{N}) \to \mathbf{L}(\mathcal{M})$, where $\mathcal{N} = \mathbf{C}^{2^n}$ and $\mathcal{M} = \mathbf{C}^{2^m}$. By lemma 1, it has a representation of the form $g = \mathrm{Tr}_{\mathcal{F}}(V \cdot V^\dagger)$, where $\mathcal{F} = \mathbf{C}^{2^{n+m}}$, and $V : \mathcal{N} \to \mathcal{M} \otimes \mathcal{F}$ is a unitary embedding. Let us choose an orthonormal basis $\{|\eta_{i,j}\rangle,\ 1 \le i \le 2^n,\ 1 \le j \le 2^{n+m}\}$ in $\mathcal{M} \otimes \mathcal{F}$, such that $|\eta_{i,0^{n+m}}\rangle = V|i\rangle$ for any i (the other basis vectors are arbitrary). There is a unique unitary operator U which sends each vector $|i,j\rangle$ of the basic basis to the vector $|\eta_{i,j}\rangle$ of the new basis. It is obvious that $V = UV_0$, where $V_0 : |\xi\rangle \mapsto |\xi\rangle \otimes |0^{n+m}\rangle$. Hence $g = \mathrm{Tr}_{\mathcal{F}}(UV_0 \cdot V_0^\dagger U^\dagger)$. This is what we need. $\Box$

It follows that a circuit with general gates can be simulated by a circuit with unitary gates efficiently:

Theorem 1: *The model of quantum circuits with mixed states is polynomially equivalent in computational power to the standard model.*

5 Using Probabilistic Subroutines

A (probabilistic) subroutine is a function, $f : \{0,1\}^m \longmapsto R^{\{0,1\}^p}$, which outputs j with distribution which depends on the input i. A quantum circuit that uses subroutines is a circuit in which a node of fan-in=fan-out=$m+p$ may be associated instead of a quantum gate, a probabilistic function $f : \{0,1\}^m \longmapsto R^{\{0,1\}^p}$. Our definition of the way this "subroutine gate", denoted by g_f, effects the density matrix is by operating all possible deterministic functions, in the standard way, where each deterministic function is applied with the induced probability from the probabilistic function:

Definition 5 operation of a subroutine gate:

$$g_f \circ \rho = \sum_d \Pr(d) U_d \rho U_d^\dagger \tag{4}$$

Where the sum is over all deterministic functions $d : \{0,1\}^m \longmapsto \{0,1\}^p$, U_d *is defined by* $U_d|i,0\rangle = |i,d(i)\rangle$ *and the induced probability for* d *is:*

$$\Pr(d) = \prod_i \mathrm{Prob}(i \mapsto d(i)) = \prod_i f_{i,d(i)}. \tag{5}$$

Note that this definition generalizes both the case of deterministic subroutines on superpositions, and the case of probabilistic subroutines on classical inputs. The sum contains $2^{p^{2^m}}$ summands. It turns out that the same operation on density matrices can be written in a much more compact form.

Lemma 5

$$g_f \circ (|i_1,0\rangle\langle i_2,0|) = \begin{cases} \sum_j f_{ij} |i,j\rangle\langle i,j| & \text{if } i_1 = i_2 = i \\ \sum_{j_1,j_2} f_{i_1 j_1} f_{i_2 j_2} |i_1,j_1\rangle\langle i_2,j_2| & \text{if } i_1 \neq i_2 \end{cases} \tag{6}$$

Proof:

$$\sum_d \Pr(d) U_d |i_1,0\rangle\langle i_2,0| U_d^\dagger = \sum_d \Pr(d) |i_1,d(i_1)\rangle\langle i_2,d(i_2)| = \tag{7}$$
$$\sum_{j_1,j_2} \Big(\sum_{d, d(i_1)=j_1, d(i_2)=j_2} \Pr(d) \Big) \; |i_1,d(i_1)\rangle\langle i_2,d(i_2)|.$$

We now compute the term in the brackets. If $i_1 = i_2$, it becomes:

$$\begin{array}{l}\delta(j_1,j_2)\sum_{d,d(i_1)=j_1}\prod_i \mathrm{Prob}(i\mapsto d(i)) = \\ \delta(j_1,j_2)f_{i_1,j_1}\sum_{j_i,i\neq i_1}\prod_i f_{i,j_i} = \delta(j_1,j_2)f_{i_1,j_1}.\end{array} \tag{8}$$

If $i_1 \neq i_2$, then the same computation yields

$$\begin{array}{l}\sum_{d,d(i_1)=j_1,d(i_2)=j_2}\prod_i \mathrm{Prob}(i\mapsto d(i)) = \\ f_{i_1,j_1}f_{i_2,j_2}\sum_{j_i,i\neq i_1,i\neq i_2}\prod_i f_{i,j_i} = f_{i_1,j_1}f_{i_2,j_2}. \quad \Box\end{array} \tag{9}$$

Before we state our main theorem, here are some notations: For a family of unitary gates $\mathcal{G}$ we denote by $\mathcal{U}_{\mathcal{G}}$ the set of unitary gates corresponding to gates from $\mathcal{G}$, according to lemma 4. We denote by $\mathcal{U}_{\mathcal{G}}^{\dagger}$ the set of daggered unitary gates corresponding to gates from $\mathcal{G}$. C is a special unitary gate on two qubits, the *controlled not gate.* It satisfies $C|00\rangle = |00\rangle, C|10\rangle = |11\rangle$, and thus serves as a copying gate.

Theorem 2: $FQP^{FQP} = FQP$: *Let $\mathcal{G}$ be a set of gates, $\mathcal{S}$ a set of probabilistic functions from m to r bits, computable by quantum circuits using no more than k gates from $\mathcal{G}$. Let Q be a quantum circuit, which uses n gates from $\mathcal{G}$, and l subroutines from $\mathcal{S}$. There exists a quantum circuit $\tilde{Q}$ which uses no more then $n + l(O(k) + O(r) + O(m))$ gates from $\mathcal{U}_{\mathcal{G}} \cup \mathcal{U}_{\mathcal{G}}^{\dagger} \cup \mathcal{C}$ and computes f_Q.*

Proof: We now show that a subroutine $s \in S$, for $s : \{0,1\}^m \longmapsto R^{\{0,1\}^r}$, can be replaced by $O(k) + O(r) + O(m)$ gates from $\mathcal{U}_{\mathcal{G}} \cup \mathcal{U}_{\mathcal{G}}^{\dagger} \cup \mathcal{C}$. The idea is to apply Q_s, read the result by copying it to extra r qubits, and undo the subroutine. Up till now, this is just following the line of the proof for deterministic subroutines[40]. However, this is not enough when dealing with probabilistic functions. The reason, intuitively, is that in probabilistic subroutines there is more than one possible output for one input, so the state of the bits that are used to copy the output, is not in tensor product with that of the input and output bits, even if the input was classical. Hence, undoing the subroutine does not take the input bits back to their original input state. The reader is urged to try and see for herself why more effort is needed. We proceed by the following operations: We add $m+1$ blank qubits to the circuit. The last bit will be a *garbage controll* bit. First, we check if there is garbage left, i.e. if the string written on the qubits other than the main qubits is different from zero, and if so, we change the *garbage control* bit to $|1\rangle$. Then, conditioned that the garbage control bit is one, we copy the *input m* bit string of the subroutine to m ancilla bits. If there is no garbage, we leave the ancilla bits to be blank. Then we trace out, or discard, the garbage and the $m+1$ ancilla bits. This procedure results with the same operation as the subroutine gate, g_s, and we have used $O(k) + O(r) + O(m)$ gates.

Let us agree on some notation before continuing: Let the subroutine s be computed by the circuit Q_s, which uses only unitary gates from $\mathcal{U}_{\mathcal{G}}$, using theorem 4. Thus the operation of Q_s is unitary, and is described by the unitary matrix $U_s = U$. So the final density matrix of Q_s is a density matrix of a pure state, which can be written as $U|i,0\rangle$ for an input i, and r blank qubits. We can write:

$$U|i,x\rangle \;=\; |i\rangle \otimes U_i|x\rangle \tag{10}$$

$$U_i|0\rangle \;=\; \sum_j |j\rangle \otimes |\psi_{ij}\rangle \qquad \text{where} \quad \langle\psi_{ij}|\psi_{ij}\rangle = f_{ij} \tag{11}$$

Let us track the procedure step by step. We will do that by seeing what happens to a matrix of the form $|i_1\rangle\langle i_2|$. From linearity, this will be enough.

$$U|i,0\rangle \;=\; \sum_j |i\rangle \otimes |j,\psi_{ij}\rangle \tag{12}$$

When j is copied, the above expression becomes

$$\sum_j |i,j\rangle \otimes |j,\psi_{ij}\rangle \tag{13}$$

The first two registers will be referred to as the *qubits*, and the last two registers will be discarded later. Then $U^{-1} = U^\dagger$ is applied which yields

$$|\xi_i\rangle \;=\; \sum_j |i,j\rangle \otimes U_i^\dagger|j,\psi_{ij}\rangle \tag{14}$$

Let us represent $|\xi_i\rangle$ as $|\eta_i\rangle + |\nu_i\rangle$, where

$$|\eta_i\rangle \;=\; \sum_j \langle 0|U_i^\dagger|j,\psi_{ij}\rangle\,|i,j\rangle \otimes |0\rangle \tag{15}$$

corresponds to the possibility of having no garbage, and $|\nu_i\rangle = |\xi_i\rangle - |\eta_i\rangle$ is orthogonal to $|\eta_i\rangle$. Note that $\langle 0|U_i^\dagger|j,\psi_{ij}\rangle \;=\; \langle\psi_{ij}|\psi_{ij}\rangle \;=\; f_{ij}$, so

$$|\eta_i\rangle \;=\; f_{ij}\,|i,j\rangle \otimes |0\rangle \tag{16}$$

We now add the step of computing the state of the control garbage qubit, and conditioned on that coping the input. The overall procedure can be represented as follows

$$|i\rangle \;\mapsto\; |\eta_i, no, 0\rangle + |\nu_i, yes, i\rangle \tag{17}$$

where no and yes are states of the controlled garbage bit. As long as the garbage and the ancilla bits are discarded, i.e. the reduced density matrix on the original set of qubits (denote it by Q) is taken, we have:

$$\begin{gathered}|i_1\rangle\langle i_2| \;\mapsto \\ (|\eta_{i_1}, no, 0\rangle + |\nu_{i_1}, yes, i\rangle)(\langle\eta_{i_2}, no, 0| + \langle\nu_{i_2}, yes, i|)|_Q = \\ (|\eta_{i_1}\rangle\langle\eta_{i_2}|)|_Q + (|\nu_{i_1}\rangle\langle\nu_{i_2}|)|_Q\delta_{i,j}\end{gathered} \tag{18}$$

For $i_1 \neq i_2$, we have:

$$= \sum_{j_1,j_2} f_{i_1j_1} f_{i_2j_2}\,|i_1,j_1\rangle\langle i_2,j_2| \tag{19}$$

For $i_1 = i_2$, we have to go few steps back in our calculations. Recall that the vectors $|\eta_i\rangle$ and $|\nu_i\rangle$ were defined in such a way that $(|\eta_i\rangle\langle\nu_i|)|_Q = 0$ (because $|\eta_i\rangle$ corresponds to no garbage whereas $|\nu_i\rangle$ corresponds to non-null garbage). Hence

$$(|\eta_i\rangle\langle\eta_i| + |\nu_i\rangle\langle\nu_i|)|_Q = (|\xi_i\rangle\langle\xi_i|)|_Q = \sum_{j,j'}\langle j',\psi_{ij'}|U_iU_i^\dagger|j,\psi_{ij}\rangle|i,j\rangle\langle i,j'| = \sum_j f_{ij}|i,j\rangle\langle i,j|. \tag{20}$$

Thus we have the desired transformation. $\Box$

6 Metrics for Noisy Quantum Circuits

In the theory of quantum computation (as in the real life) operators, quantum states, etc. are defined with some precision. Thus, we need to define certain metrics on the corresponding spaces. We will find a natural metric (more specifically, a norm) for each class of objects we deal with: pure and mixed states, unitary and arbitrary gates. After proving some basic properties of these norms, we will show, in a very general form, that error accumulation in quantum computation is at most additive (see Theorem 3 below).

6.1 The Natural Distance Between Probabilistic Functions

We need a measure for the accuracy of the function computed. The natural norm to use is the ℓ_1-norm, called the total variation distance (t.v.d.) between probability distributions. We use t.v.d. to define a metric on probabilistic functions.

Definition 6 *Let f, g be probabilistic functions. For input i, f_i, g_i are probability distributions. The total variation distance between f_i, g_i is $|f_i - g_i| \stackrel{\text{def}}{=} \sum_j |f_{i,j} - g_{i,j}|$ and $\|f - g\| \stackrel{\text{def}}{=} max_i |f_i - g_i|$.*

6.2 The Trace Metric on Density Matrices

Precision of a vector $|\xi\rangle \in \mathcal{N}$ (where $\mathcal{N}$ is a Hilbert space) is characterized by the natural (Euclidean) norm $\||\xi\rangle\| = \sqrt{\langle\xi|\xi\rangle}$. Since we have passed to density matrices, we need a metric on general quantum states. There are two natural norms on the space of linear operators on $\mathcal{N}$: the usual operator norm,

$$\|A\| \;=\; \sup_{|\xi\rangle\neq 0} \frac{\|A|\xi\rangle\|}{\||\xi\rangle\|} \;=\; \text{largest eigenvalue of } \sqrt{A^\dagger A} \tag{21}$$

and the dual norm called the *trace norm*,

$$\|A\|_1 \;=\; \sup_{B\neq 0} \frac{|\operatorname{Tr} AB|}{\|B\|} \;=\; \operatorname{Tr}\sqrt{A^\dagger A} \tag{22}$$

The norms $\|\cdot\|$ and $\|\cdot\|_1$ are well behaved. Specifically, if $A \in \mathbf{L}(\mathcal{N})$ and $B \in \mathbf{L}(\mathcal{M})$ then

Lemma 6

$$\begin{aligned} \|A\otimes B\| &= \|A\|\,\|B\|, \\ \|A\otimes B\|_1 &= \|A\|_1\,\|B\|_1 \\ \|AB\| &\leq \|A\|\,\|B\| \\ \|AB\|_1,\ \|BA\|_1 &\leq \|B\|\,\|A\|_1 \\ |\operatorname{Tr} A| &\leq \|A\|_1 \end{aligned} \tag{23}$$

(Proof is trivial).

There are many good reasons to use the trace norm as the norm on density matrices (though the operator norm will be very useful in proofs.) First, two pure states $|\xi\rangle, |\eta\rangle$ which are close in the Euclidean norm are close also in the trace norm:

$$\Big\| |\xi\rangle\langle\xi| - |\eta\rangle\langle\eta| \Big\|_1 \;=\; 2\sqrt{1 - |\langle\xi|\eta\rangle|^2} \;\leq\; 2\Big\| |\xi\rangle - |\eta\rangle \Big\| \tag{24}$$

The important feature of the trace metric is that it captures the *measurable* distance between different density matrices. It turns out that the trace distance between two density matrices equals the following quantity. For each observable O, a density matrix ρ induces a probability distribution, p^O_ρ, over $i's$. The trace distance between two density matrices is the maximal t.v.d between the two probability distributions, taken over all possible observables.

Lemma 7 $\|\rho_1 - \rho_2\|_1 = max_O \left\{ |p^O_{\rho_1} - p^O_{\rho_2}| \right\}$.

Proof: Let $\mathcal{N} = \bigoplus_j S_j$, where the subspaces S_j are mutually orthogonal. Let P_j be the orthogonal projection onto S_j. Then, for any pair of mixed states ρ_1 and ρ_2, $\sum_j \Big| \mathrm{Tr}(P_j\rho_1) - \mathrm{Tr}(P_j\rho_2) \Big| \leq \|\rho_1 - \rho_2\|_1$. To see this, present the left hand side of this inequality as $\mathrm{Tr}((\rho_1 - \rho_2)B)$, where $B = \sum_j \pm P_j$. It is obvious that $\|B\| = 1$. Then use lemma 6. To see that the trace distance can be achieved by some measurement, let O project on the eigenvectors of $\rho_1 - \rho_2$. $\Box$

6.3 The Diamond Metric on Quantum Gates

The natural norm on the space of super-operators is

$$\|T\|_1 \;=\; \sup_{X\neq 0} \frac{\|TX\|_1}{\|X\|_1} \tag{25}$$

Unfortunately, this norm is not stable with respect to tensoring with the identity. Counterexample: $T : |i\rangle\langle j| \mapsto |j\rangle\langle i| \;\; (i,j = 0,1)$. It is clear that $\|T\|_1 \leq 1$. However $\|T \otimes I_{\mathcal{B}}\|_1 \geq 2$. (Apply the super-operator $T \otimes I_{\mathcal{B}}$ to the operator $X = \sum_{i,j} |i,i\rangle\langle j,j|$). For this reason, we have to define another norm on super-operators

Definition 7 *Let* $T : \mathbf{L}(\mathcal{N}) \to \mathbf{L}(\mathcal{M})$ *and* $A, B \in \mathbf{L}(\mathcal{N}, \mathcal{M} \otimes \mathcal{F})$*, where* $\mathcal{F}$ *is an arbitrary Hilbert space of dimensionality* $\geq (\dim \mathcal{N})(\dim \mathcal{M})$.

$$\|T\|_\diamond \;=\; \inf\Big\{ \|A\|\,\|B\| \;:\; \mathrm{Tr}_{\mathcal{F}}(A \cdot B^\dagger) = T \Big\} \tag{26}$$

This definition seems very complicated. However it is worthwhile using this norm because it satisfies very nice properties, and provides powerful tools for proofs regarding quantum errors. Here are some properties which are satisfied by the diamond norm. The first property is that the diamond norm is the stabilized version of the "naive" norm $\|\cdot\|_1$. The proof of this is complicated and non-trivial. It implies also that $\|\cdot\|_\diamond$ is a norm.

Lemma 8

1. $\|T\|_\diamond \;=\; \|T \otimes I_{\mathcal{G}}\|_1 \;\geq\; \|T\|_1,\ where, \dim \mathcal{G} \geq \dim \mathcal{N}.$
2. $\|T\rho\|_1 \leq \; \|T\|_\diamond \, \|\rho\|_1$
3. $\|TR\|_\diamond \;\leq\; \|T\|_\diamond \, \|R\|_\diamond$
4. $\|T \otimes R\|_\diamond \;=\; \|T\|_\diamond \, \|R\|_\diamond$
5. The norm of any physically allowed super-operator T is equal to 1.

6. If $\|V\| \le 1$ and $\|W\| \le 1$ then $\|V \cdot V^\dagger - W \cdot W^\dagger\|_\diamond \le 2\|V - W\|$.

Proof of 8.1: It is easy to see that $\|T \otimes I_\mathcal{G}\|_1 \le \|T \otimes I_\mathcal{G}\|_\diamond \le \|T\|_\diamond$. The inequality $\|T\|_\diamond \le \|T \otimes I_\mathcal{G}\|_1$ is not so obvious. W.l.o.g. $\|T\|_\diamond = 1$. We are to prove that $|T \otimes I_\mathcal{G}\|_1 \ge 1$.

We will use the following notation. $\mathbf{D}(\mathcal{K})$ denotes the set of density matrices on $\mathcal{K}$, whereas $\mathbf{H}(\mathcal{F})$ is the set of Hermitian operators. We can impose the restriction $\|A\| = \|B\| \le 2$ without changing the infimum in the definition 7. Due to compactness, the infimum is achieved at some A and B. W.l.o.g. $\|A\| = \|B\| = 1$. The quantity $\|A\|\,\|B\|$ is minimal with respect to infinitesimal variations of the scalar product $\delta\langle\cdot|\cdot\rangle = \langle\cdot|Z|\cdot\rangle$ on the space $\mathcal{F}$. (Here Z is a infinitely small Hermitian operator on $\mathcal{F}$). When computing the variations $\delta\|A\|$ and $\delta\|B\|$, we can restrict A and B to the subspaces $\mathcal{K} = \mathrm{Ker}(A^\dagger A - 1_\mathcal{N})$ and $\mathcal{L} = \mathrm{Ker}(B^\dagger B - 1_\mathcal{N})$. Clearly,

$$
\begin{aligned}
\delta\|A\| &= \max_{|\xi\rangle\in\mathcal{K},\ \||\xi\rangle\|=1} \langle\xi|A^\dagger(1_\mathcal{M}\otimes Z)A|\xi\rangle &&= \max_{X\in E} \mathrm{Tr}(XZ) \\
\delta\|B\| &= \max_{|\eta\rangle\in\mathcal{L},\ \||\xi\rangle\|=1} -\langle\eta|B^\dagger(1_\mathcal{M}\otimes Z)B|\eta\rangle &&= \max_{Y\in F} -\mathrm{Tr}(YZ)
\end{aligned}
\tag{27}
$$

where

$$
\begin{aligned}
E &= \left\{\mathrm{Tr}_\mathcal{M}(A\rho A^\dagger) : \rho \in \mathbf{D}(\mathcal{K})\right\} \\
F &= \left\{\mathrm{Tr}_\mathcal{M}(B\gamma B^\dagger) : \gamma \in \mathbf{D}(\mathcal{L})\right\}
\end{aligned}
\tag{28}
$$

Thus, for any $Z \in \mathbf{H}(\mathcal{F})$

$$
\delta\left(\|A\|\,\|B\|\right) = \max_{X\in E,\ Y\in F} \left(\mathrm{Tr}(XZ) - \mathrm{Tr}(YZ)\right) \ \ge\ 0 \tag{29}
$$

This means that the sets $E, F \subseteq \mathbf{H}(\mathcal{F})$ can not be separated by a hyper-plane. As E and F are convex and compact, $E \cap F \ne \emptyset$. Let $\mathrm{Tr}_\mathcal{M}(A\rho A^\dagger) = \mathrm{Tr}_\mathcal{M}(B\gamma B^\dagger) \in E \cap F$, where $\rho \in \mathbf{D}(\mathcal{K})$, $\gamma \in \mathbf{D}(\mathcal{L})$. Let us represent ρ and γ in the form $\rho = \mathrm{Tr}_\mathcal{G}\big(|\xi\rangle\langle\xi|\big)$, $\gamma = \mathrm{Tr}_\mathcal{G}\big(|\eta\rangle\langle\eta|\big)$, where $|\xi\rangle, |\eta\rangle \in \mathcal{N}\otimes\mathcal{G}$ are unit vectors. Put $X = |\xi\rangle\langle\eta|$. Then $\|(T \otimes I_\mathcal{G})X\|_1 = \|X\|_1 = 1$.

Proof of 8.2,8.3: follows from the relation to the norm $\|\cdot\|_1$ and the definition of the latter.

Proof of 8.4: To prove first direction, $\|T \otimes R\|_\diamond \ \le\ \|T\|_\diamond\,\|R\|_\diamond$ follows from the definition 7, whereas the inverse inequality follows from 1.

Proof of 8.5: W. l. o. g. $\|T\|_\diamond = \|T\|_1$ (since we can tensor T with the identity). Let $T = \mathrm{Tr}_\mathcal{F}(V \cdot V^\dagger) : \mathbf{L}(\mathcal{N}) \to \mathbf{L}(\mathcal{M})$. Let us define the dual super-operator $R : \mathbf{L}(\mathcal{M}) \to \mathbf{L}(\mathcal{N})$ with the property: $\mathrm{Tr}(Y(TX)) = \mathrm{Tr}((RY)X)$ for every $X \in \mathbf{L}(\mathcal{N})$ and $Y \in \mathbf{L}(\mathcal{M})$. It is obvious that $RY = V^\dagger(Y \otimes I_\mathcal{F})V$ and

$$
\|T\|_1 = \sup_{Y\ne 0} \frac{\|RY\|}{\|Y\|} \tag{30}
$$

As $\|V\| \le 1$, the inequality $\|T\|_1 \le 1$ follows immediately. To prove the inverse inequality, take the identity operator for Y.

Proof of 8.6: To prove this we need lemma 9 from the next section. (For completeness this property appears here). In the definitions of the lemma put $T_1 = V \cdot I,\ T_1' = W \cdot I,\ T_2 = I \cdot V^\dagger,\ T_2' = I \cdot W^\dagger$. $\square$

The distance between unitary super-operators $V \cdot V^\dagger$,$W \cdot W^\dagger$, can be calculated explicitly, and has a geometrical interpretation. Denote by d the distance between 0 and the polygon (in the complex plane) whose vertices are the eigenvalues of $VW^\dagger$. Then

$$\|V \cdot V^\dagger - W \cdot W^\dagger\|_\diamondsuit = \max_{\rho \in \mathbf{D}(\mathcal{N})} \left\|V\rho V^\dagger - W\rho W^\dagger\right\|_1 = 2\sqrt{1-d^2}. \tag{31}$$

(The proof is left to the reader).

7 Bounding the Overall Error

By definition, the error of a quantum gate is measured by the $\diamondsuit$-norm. The accumulation of errors is bounded by the following lemma:

Lemma 9 *Let T_1, T_2 and T_1', T_2' be super-operators with norm ≤ 1, such that $\|T_j' - T_j\|_\diamondsuit \leq \epsilon_j$ $(j = 1, 2)$. Then $\|T_2'T_1' - T_2T_1\|_\diamondsuit \leq \epsilon_1 + \epsilon_2$.*

Proof: Write $T_2'T_1' - T_2T_1 = T_2'(T_1' - T_1) + (T_2' - T_2)T_1$, and use lemma 8.5 $\square$

For subroutines, the natural error measure is different. Fortunately, there is a linear upper bound for the $\diamondsuit$-norm error of a subroutine:

Lemma 10 *Let f and f' be two probabilistic subroutines, such that $\|f - f'\| \leq \epsilon$. Then $\|g_{f'} - g_f\|_\diamondsuit \leq 5\epsilon$. (The super-operator g_f is described in section 5, definition 5).*

Proof: Let $\mathcal{N}$ be the space of the inputs $|i\rangle$, $\mathcal{M}$ the space of the outputs $|i,j\rangle$. Define the following objects (and their primed versions)

$$A \ :\ \mathcal{N} \to \mathcal{M} \ :\ |i\rangle \mapsto \textstyle\sum_j f_{ij}|i,j\rangle \tag{32}$$

$$B \ :\ \mathcal{N} \to \mathcal{M} \otimes \mathcal{N} \ :\ |i\rangle \mapsto \textstyle\sum_j f_{ij}|i,j,i\rangle \tag{33}$$

$$\rho_i = \sum_j f_{ij}|i,j\rangle\langle i,j| \tag{34}$$

$$P_i \ :\ \mathbf{L}(\mathcal{N}) \to \mathbf{C} \ :\ |i_1\rangle\langle i_2| \mapsto \delta_{i_1 i_2} \tag{35}$$

Then $g_f = A \cdot A^\dagger - \mathrm{Tr}_{\mathcal{N}}(B \cdot B^\dagger) + \sum_i \rho_i P_i$ (the same for $g_{f'}$). Clearly, $\|A' - A\| \leq \epsilon$, $\|B' - B\| \leq \epsilon$, and the norm of each operator A, B, A', B' does not exceed 1. It remains to show that $\|T\|_\diamondsuit \leq \epsilon$, where $T = \sum_i (\rho_i' - \rho_i)P_i$.

Note that $\|\rho_i' - \rho_i\|_1 \leq \epsilon$ for each i. Hence $\|(T \otimes I_{\mathcal{G}})X\|_1 \leq \epsilon \sum_i \|Y_i\|_1$, where $\mathcal{G}$ is an arbitrary Hilbert space, $X \in \mathbf{L}(\mathcal{N} \otimes \mathcal{G})$, and $Y_i = (P_i \otimes I_{\mathcal{G}})X$. On the other hand,

$$\sum_i \|Y_i\|_1 = \left\|\sum_i |i\rangle\langle i| \otimes Y_i\right\|_1 = \|(P \otimes I_{\mathcal{G}})X\|_1 \leq \|P\|_\diamondsuit \|X\|_1 = \|X\|_1 \tag{36}$$

where $P: |i_1\rangle\langle i_2| \mapsto \delta_{i_1 i_2}|i_1\rangle\langle i_2|$ is a physically realizable super-operator. $\Box$

Due to lemmas 9 and 7, an ϵ error generated somewhere in the circuit can not contribute more than ϵ error to the computed function. This proves the following theorem:

Theorem 3 *Let Q be a quantum circuit which uses L probabilistic subroutines and gates, each with at most ϵ error. The function that Q computes has at most $O(L\epsilon)$ error.* $\Box$

8 Quantum Circuits With General Noise

As any physical system, a quantum system is subjected to noise. The process of noise is dynamic and depends on time. In order to incorporate faults into quantum circuits, we consider leveled circuits, in which the quantum gates are applied at discrete times. Note that we permit that qubits are input and output at different time steps to the circuit, unlike Yao[221]. The density matrix of the circuit develops by applying alternately computation steps and noise steps. After the t'th time step, an error operator $\mathcal{E}(t)$ operates on the density matrix $\rho(t)$:

$$\rho(t) \longmapsto \mathcal{E}(\rho(t)) \tag{37}$$

Of course this model of noise is too general for specific possible physical realizations of a quantum computer. Different physical implementations will probably exhibit noise which can be described by super-operators $\mathcal{E}$ of a restricted type. In this thesis I will adopt two very important assumptions.

- Locality: No correlations between noise operators on different qubits, except through the gates.
- The Markovian assumption: The environment is renewed each time step, and hence no correlations between the environments at different time steps.

These are very common assumptions in the literature, but researchers have also started to investigate noise with long range correlations, such as "collective decoherence"[152], a fascinating subject which is now being studied. If the typical wavelength in the environment is much longer than the typical distance between two qubits in the quantum computer, than the collective noise process is likely to hold. Physically, the locality and the Markovian assumptions hold when the typical wavelength in the environment is much shorter than the typical distance between two qubits in the quantum computer, which means that each qubit interacts with its own environment.

Based on the locality and the Markovian assumptions, we can define sites, of a small number of qubits at a certain time, in which the faults can occur. These sites will be called the *locations* of the circuit.

Definition 8 *A set $(q_1, q_2, ..q_l, t)$ is a location in the quantum circuit Q if the qubits $q_1, ...q_l$ participated in the same gate in Q, in time step t, and no other qubit participated in that gate. For this matter, the absence of a gate is viewed as the identity gate on one qubit, so if a qubit q did not participate in any gate at time t, then (q, t) is a location in Q as well.*

Faults occur in between time steps, in the *locations* of the circuit. The faults are thus confined to a small set of qubits, at a specific time step. The super-operator $\mathcal{E}(t)$ describing the noise operator will be of the following form. If n is the number of locations in the system at time t, then $\mathcal{E}(t)$ can be described by a tensor product of n super-operators, each operating on one qubit.

$$\mathcal{E}(t) = \mathcal{E}_{A_{1,t}}(t) \otimes \mathcal{E}_{A_{2,t}}(t) \otimes \cdots \otimes \mathcal{E}_{A_{l_t,t}}(t), \tag{38}$$

where $A_{i,t}$ runs over all possible locations at time t. For each location,

$$||\mathcal{E}_{A_{i,t}}(t) - I|| \leq \eta \tag{39}$$

The exact properties of the norm on super-operators are not used in this thesis. We merely need the following properties:

1. $||T\rho||_1 \leq ||T||||\rho||_1$
2. $||TR|| \leq \ ||T||||R||$
3. $||T \otimes R|| = \ ||T||||R||$
4. The norm of any physically allowed super-operator T is equal to 1.

A reasonable norm on super-operators should satisfy these properties. One example is the diamond norm defined in section 6.3.

Definition 9 *η is called the "error rate" of the circuit.*

The noisy circuit is said to compute a function f with confidence ϵ if for any choice of the noise operators $\mathcal{E}'$, the output probabilistic function is within ϵ total variation distance to f.

In most of the thesis we will not use this very general model, but mainly use a probabilistic noise model, in which the noise operators in the above tensor product are of specific form described below.

8.1 Quantum Circuits with probabilistic Errors.

Stochastic errors is the most common error model in the quantum error correction literature. The reason is perhaps that this is the easiest model to understand and the closest to our classical intuition. This model can be described as follows: Each time step, each location with probability η is faulty, and with probability $1-\eta$ remains as it is. An arbitrary super-operator $\mathcal{E}'_{A_t}(t)$ corresponding to the fault is applied on the faulty location $A_t = (q_1, ..q_l, t)$ after time step t. In equation 38, this corresponds to the restriction that for each location,

$$\mathcal{E}_{A_{i,t}} = (1-\eta)\mathcal{I} + \eta\mathcal{E}'_{A_{i,t}} \tag{40}$$

This satisfies the conditions of local noise 38,39 with error rate 2η. This model generalizes the classical stochastic bit flips model, in which each bit is flipped with independent probability η. In the quantum case, the super-operator $\mathcal{E}'_{A_{i,t}}$ can be much more general than simply a bit flip.

8.2 How to Implement Specific Noise Processes in This Noise Model

Almost all noise processes which are dealt with in the literature are special cases of the general noise model. It includes:

- Independent probabilistic errors,
- depolarization,
- Stochastic collapses
- amplitude and phase damping,
- decoherence,
- and systematic inaccuracies in gates.

We have seen in the previous subsection how probabilistic errors are implemented in the general noise model. A very commonly used stochastic noise process is depolarization, in which $\mathcal{E}'$ replaces all qubits in the location by the density matrix of a qubit in a random state (this density matrix is simply half the identity). Another stochastic noise model is the stochastic collapses, in which each qubit with probability η is measured in some basis. $\mathcal{E}'$ is then the operator which applies this measurement. This model is equivalent to a certain process of decoherence in which the off diagonal terms of the density matrix of one qubit decay in between two time steps by a factor of $(1-\eta)$[172]. Decoherence[65], amplitude damping and phase damping are all special cases of interactions between the qubits and the environments, and as such they can be described in the general noise model by definition, as long as the process is local. Systematic inaccuracies in a given gate are described by applying the same noise operator on all locations where that gate operated.

This completes the definition of noisy quantum circuits. We proceed to discuss noisy quantum circuits with strong noise.

Chapter III

Strong Noise

1 Summary

In this chapter I prove upper bounds on the computational power of very noisy quantum circuits. For upper bounds, it suffices to restrict the noise process. I will present a classical randomized simulation of a quantum circuit subjected to stochastic collapses or to depolarization. The simulation will turn out to be efficient if the error rate η is larger than a certain threshold. This will show that quantum circuits with strong noise are not stronger in computational power than classical randomized computers.

The idea of the simulation is simple. The algorithm gets as an input the description of the quantum circuit, the input string of n bits for the circuit and the error rate η. It then initializes a description of the density matrix of the quantum circuit, and updates it according the development in time of the circuit. The problem of course, is that in general the density matrix is exponential in size. Trying to be efficient, the simulation presents the density matrix as a tensor product of smaller density matrices, where each density matrix describes a set of qubits. Each such set is completely disentangled from the other sets of qubits, and we will call each set a "cluster". The size of the description will be exponential in the sizes of the clusters, and so we will try to make the clusters as small as possible. For example, the initial density matrix can be written as a tensor product of n density matrices, each describing one qubit. The initial configuration of clusters is therefore n clusters each of size one. To imitate a computation step, the algorithm computes the operation of the gate on the relevant density matrix. If the gate operates on qubits from different clusters, then the algorithm joins the two clusters to one bigger cluster, by taking the tensor product of the relevant density matrices. Here is how the simulation imitates the noise step, i.e. the stochastic measurements. For each qubit the algorithm chooses randomly, with bias η, whether the qubit has collapsed at that time step. If the answer is yes, it computes the probability p for it to collapse on $|1\rangle$ versus $|0\rangle$. This is a simple computation given the relevant density matrix. The algorithm then tosses another coin with the newly computed bias p. This determines whether the qubit has collapsed on $|1\rangle$ or on $|0\rangle$. After a qubit collapses to one of it's basic states it is no longer entangled to the other qubits, and can be described by it's own density matrix. The new, smaller density matrix of the rest of the qubits in the cluster can be easily computed. The density matrix of that qubit's cluster is now replaced by two density matrices: One describing the new state of the measured qubit, and one describing the new state of the rest of the qubits which were in the same cluster. After this is done for all the qubits, the simulation continues to the next computation step in the quantum circuit, and to the next noise step, and so on and so forth until the final time step was simulated. Then the algorithm imitates a measurement of the relevant qubits, and outputs the result. Clearly, the simulation outputs the correct output. The idea is that since the noise is stochastic, the classical simulation chooses one of the possibilities with the correct probability, and continues the simulation. (A similar idea regarding a simulation of classical probabilistic machines was presented in [42].) The question is now whether this simulation is efficient.

Before I explain the results, let us gain some intuition. If the circuit had not been noisy, very soon all the clusters would become one giant cluster of n qubits. What makes the simulation more efficient is the stochastic collapses, which will separate a collapsed qubit from its cluster, thereby decreasing the size of the cluster by one, and creating another cluster of size one. One can view the noisy quantum computation as a struggle between two forces: The computation, which tries to use the exponential dimension of the Hilbert space, by entangling as many qubits as it can, and the stochastic collapses, which destroy this entanglement constantly. From this point of view, a sequential computer obviously cannot win the battle - each time step it applies only one local operation, but $\theta(n)$ qubits collapse. The picture in circuits with general parallelism is different, since both forces that compete have linear power: $\theta(n)$ gates can be applied each time step, and a percentage of the n qubits collapses each time step. Since the competition here is even, the question which force wins in this struggle is delicate and requires careful consideration. We will find out that there is a critical error rate, η_c, at which the two forces are equal. For noise stronger than η_c, the collapses manage to disentangle most of the qubits: In this regime, the simulation will be polynomial, and the quantum circuit has no advantage over classical computation.

I can now state the results. In our analysis we assume that the gates are of fan-in not bigger than two[93], but the results can be extended to any constant fan-in.

- **Limited Parallelism**. When the quantum circuit applies only one operation per time step, as is the case for a quantum Turing Machine, or when each time step at most $O(\log(n))$ participate in an operation, the simulation is always efficient, as long as $\eta > 0$.
- **General circuits** with $\eta > 0.97$, can be simulated efficiently. To show this we used branching processes techniques.
- **Sparse circuits**, which apply two-qubit gates rarely, can be simulated efficiently even for much weaker noise. If each qubit participates in a many-qubit gate only once every ν time steps, the circuit loses its quantum advantage already at $\eta > 1 - (0.03)^{\frac{1}{\nu}}$.
- **A Random Quantum circuit** can be simulated efficiently already for $\eta > 0.63$. This was not proved, but shown numerically, using computer experiments.
- **One dimensional Quantum circuits** where gates operate only on nearest neighbor qubits, can be simulated efficiently already for $\eta > 0.5$.

The most important result here is that for general quantum circuits. The lower bound of 0.97 is of course not a reasonable physical number, to even start quantum computation with. Note however, that even with this strong noise, it is not clear that quantum computers are not stronger than classical computers, since a noisy quantum computer with stochastic collapses can simulate with zero error ideal classical computation- measurements do not effect a classical computation performed by the quantum circuit. The number 0.97 comes out so large in the analysis because it is good for any quantum circuit. Most circuits are simulated efficiently with much weaker noise, as we see by the numerical results for random circuits. However, it is important that this bound, 0.97, is bounded away from 1, so that there is a non-trivial regime in which noisy quantum computation is not stronger than classical computation.

A very interesting feature of our simulation is that it cost turns abruptly from exponential to polynomial as the noise increases above a certain critical point, i.e. the cost exhibits a phase transition at a critical error rate, η_c. This point, η_c, depends on the specific quantum circuit we are simulating. In the case of one dimensional quantum array, which operates gates only on nearest neighbors, I was able to prove the existence of this phase transition analytically and to find the value of the critical error rate, $\eta = 0.5$. For this I use a reduction to standard percolation methods. The phase transition occurs also for the case of a random quantum circuit, but in a different critical point: $\eta \approx 0.63$. This is shown using computer experiments. This phase transition seems to be connected to a very important phenomenon in physics, namely the transition from quantum to classical physics. I will discuss the possible implications at the end of the thesis.

All of these results except for the discussion of one dimensional circuits were presented in a paper by myself and Michael Ben-Or[9]. The discussion of the one-dimensional case can be found in [13].

2 A Classical Simulation Of Noisy Quantum Circuits

We define an algorithm $S(Q, i, \eta)$ which gets as an input a description of a quantum circuit, Q, an input string i and an error rate η. The algorithm outputs a string exactly with the same probability distribution that Q would output for an input i, when subjected to measurements in the computational basis, with probability η per qubit, per time step. It should be noted that the simulation can be trivially modified to work when the quantum circuit is subjected to single qubit collapses in any basis, but for convenience we will use the computational basis. The simulation can also be easily modified to work for quantum circuits subjected to depolarization, and all the results still hold. S operates as follows:

Input:

Description of the circuit Q, the noise strength η and the input string i. Let us assume that the number of initial qubits in Q is n, and that the circuit uses poly(n) local gates.

Simulation:

- S initializes the configuration $C(0)$, to be a list of n density matrices of one qubit, $|0\rangle\langle 0|$ or $|1\rangle\langle 1|$ that correspond to the n bits in i.

- For $t = 1$ to T, S does:

a) To simulate the t'th computation step: Let $\mathcal{G}$ be a gate that operates at time t. A computes the tensor product ρ of all the density matrices in $C(t)$ which describe qubits on which $\mathcal{G}$ operates. Then S computes $\mathcal{G} \circ \rho$. Then S erases all the matrices which contributed to ρ from $C(t)$ and adds the matrix $\mathcal{G} \circ \rho$ to the configuration. This is done for all the gates of the $t'th$ time step to give $C^*(t)$.

b) To simulate a collapse step: For each qubit S tosses a coin with bias η, to decide whether the qubit collapses at time t. If the answer is 1, then S computes the probability p for the qubit to collapse on $|1\rangle$, from the relevant density matrix ρ in $C^*(t)$. Then S tosses another coin with bias p, to decide whether the qubit have collapsed on $|1\rangle$ or $|0\rangle$. Given the result, say $|1\rangle$, the algorithm computes the new density matrix of the rest of the qubits described by ρ, i.e it computes the projection of ρ on the space where the collapsed qubit is $|1\rangle$, and then traces out the collapsed qubit. The algorithm adds to $C^*(t)$ this new density matrix, and another 2×2 density matrix which describes the state of the collapsed qubit. This is done for all the qubits, to give $C(t+1)$.

Output: To output the results of the computation, S imitates basic measurements of the result qubits, on the relevant matrices in $C^*(T)$, just like in the collapse step, and outputs the outcome.

Clearly The algorithm clearly outputs exactly the same distribution on the outputs as the quantum circuit would. (This is known as a Monte-Carlo simulation). It is convenient to change the simulation slightly, and make it a Las Vegas simulation: we fix a time bound, and change S such that after $T_b = poly(n)$ time steps the simulation stops, regardless of the state of the computation, and outputs an arbitrary number. This guarantees that the simulation is efficient, but of-course now we have introduced an error. We will have to show that the error generated due to long runs is small. We will do this by showing, in the different cases, that the probability for the simulation to rum for more than T_b time steps is polynomially small. This will guarantee that the probability distribution which the simulation outputs is within polynomially small total variation distance from the correct output. It should be noted that the important parameter here is the size of the individual density matrices in the configurations $C(t), C^*(t)$. Recall that the set of qubits described by each one of the density matrices is called a cluster. As long as the sizes of the clusters are $O(log(n))$, the simulation is polynomial in n, since the number of entries in each density matrix in $C(t)$ will be polynomial. Therefore, in order to show that the error is small, it suffices to consider the development in time of the sizes of clusters, during the simulation, and show that the probability that any cluster exceeds $O(log(n))$ during the simulation is polynomially small.

3 Limited Parallelism

I will start with the case of limited parallelism, since in this case the analysis is much simpler than in the general case. We consider the case in which the circuit is allowed to operate on $O(log(n))$ qubits at the most each time step. All the other qubits are "resting". As was described in the introduction to this chapter, the "battle" is lost before it even begins: there is no way such a limited computation power can compare with the parallel power of collapses, which cause a percentage of the qubits to be disentangled each time step. The purpose of this section is to show that the clusters will therefore stay small even with very weak noise, as long as the noise is non-zero.

To analyze the cost of the simulation, We divide the qubits each time step to two sets: The *individual qubits*, that in the configuration of S at time t, are described by their own 2×2 density matrix, and the rest of the qubits. We will refer to the union of the non-individual qubits as a "super-cluster", even though they might be divided to smaller clusters. Our purpose is to show that the size of the super-cluster is small enough, with high probability.

Denote by $K(t)$, $K^*(t)$ the size of the super-cluster at time t in the configuration $C(t), C^*(t)$ in the simulation. Let us denote by P the parallelism in the circuit, i.e. the number of qubits which are active in one time step. I will now show that the probability distribution for $K(t)$, $K^*(t)$ decays exponentially starting from sizes which are of order of P, for any non-zero error rate, $\eta > 0$.

Lemma 11 *Let Q be a circuit which operates on no more than P qubits each step, with $\eta > 0$. Consider $S(Q,i,\eta)$. There exist constants $a > 4$, and $b > 0$ such that*

$$Pr\left(K(t) \geq bP + j\right) \leq \frac{1}{a^j}, \tag{1}$$

$$Pr\left(K^*(t) \geq (b+1)P + j\right) \leq \frac{1}{a^j}. \tag{2}$$

Proof: We use induction on t. For $t = 0$, $K(0) = 0$, $K^*(0) \leq P$. For the induction step, assume $Pr(K(t) \geq bP + j) \leq \frac{1}{a^j}$ and $Pr(K^*(t) \geq bP + j + P) \leq \frac{1}{a^j}$ for t. To analyze $Pr(K(t+1))$, We apply collapses on $K^*(t)$. The probability to reduce the number of non-individual qubits is binomial so:

$$Pr(K(t+1) \geq bP + i) = \sum_{j=0}^{\infty} Pr(K(t+1) = bP + i + j) \leq \tag{3}$$

$$\sum_{j,l=0}^{\infty} \frac{1}{a^{i+j+l-P}} \begin{pmatrix} bP + i + j + l \\ l \end{pmatrix} \eta^l (1-\eta)^{bP+i+j} = \tag{4}$$

$$\sum_{j=0}^{\infty} \frac{(1-\eta)^{bP+i+j}}{a^{i+j-P}} \left(\sum_{l=0}^{\infty} \begin{pmatrix} bP + i + j + l \\ l \end{pmatrix} \left(\frac{\eta}{a}\right)^l \right) = \tag{5}$$

$$\frac{a^{P+1}}{a-1} \left(\frac{(1-\eta)a}{a-\eta}\right)^{bP} \left(\frac{1-\eta}{a-\eta}\right)^i. \tag{6}$$

We want this to be smaller than $\frac{1}{a^i}$, for any i. This implies two demands: (1) $\frac{1-\eta}{a-\eta} \leq \frac{1}{a}$, which is achieved for $a \geq 1$. (2) The coefficient of the geometrical sequence must be smaller then 1. There exists b that fixes that since $\frac{(1-\eta)a}{a-\eta}$ is strictly smaller than 1, because $\eta > 0$.

For $Pr(K^*(t+1))$ we have that $K^*(t+1) \leq K(t+1) + P$. Then $Pr(K^*(t+1) \geq bP + i + P) \leq Pr(K(t+1) \geq bP + i) \leq \frac{1}{a^i}$.$\Box$

The fact that the probability distribution over the size of the super-cluster decays exponentially, implies that the probability for the super-cluster to be much larger is very small. This will mean that the simulation is rarely stopped by the bound of T_b time steps, and so the error is small:

Theorem 4 *Let Q be a circuit which operates on no more than $P = O(log(n))$ qubits each step, with $\eta > 0$, for $T = poly(n)$ time steps. Then the error in $S(Q,i,\eta)$ is polynomially small.*

Proof: By lemma 11, the probability for the super-cluster to be of size O(log(n)) throughout the simulation is polynomially small. $\Box$

This proves that quantum circuits with limited parallelism are not stronger than classical randomized computation devices, even with very weak noise, as long as $\eta > 0$. It should be noted that this result holds in particular for the model of quantum Turing machines, which are inherently limited in parallelism. However, I will not explain this here since I haven't defined a quantum Turing machine.

4 General Circuits

For general circuits, the analysis using the super-cluster fails, since basically a large fraction of the qubits are non individual. A more refined analysis is required here. We intend to show by induction on t that the probability for any cluster to be large decays exponentially. The difficulty here is to bound the change in the clusters' sizes during the computation step. If we could bound this, so that the clusters do not grow too much during the computation step, and if the noise is strong enough to shrink them back, then the induction step will work. In the following we analyze separately the change in the distribution of sizes of clusters, during the computation step, and during the noise step. Then we combine the two to show that the cost of the simulation is polynomial for strong enough noise: $\eta > 0.97$.

4.1 Clusters' Growth in a computation step

We define the size of the cluster at time t for each qubit:

Definition 10 Cluster's size *We define $K_q(t)$,$K_q^*(t)$ as the size of the cluster of qubit q in $C(t), C^*(t)$, respectively.*

Assuming an exponential decaying bound on the sizes of clusters before the computation step, we can give an exponential decaying bound on the sizes of clusters after the computation step. To do that we observe a connection to a branching process. Let the clusters in the configuration at time t, $C(t)$, be $A, B, ...$ Let M be the next computational step. The question is what is the cluster's size of a qubit $a \in A$ after M is applied. Recall that we assume gates of fan-in two at the most. If a gate in M operates on two qubits then they are called *mates*. We associate with a a *tree* of which a is the *root*. a's sons in the tree are the qubits in A. Suppose a_1 is a son of a and a mate of $b \in B$. Then all members in B are the sons of a_1, and so on. In each stage we ignore qubits that already appear in the tree. The distribution on sizes of clusters corresponds to a distribution on the number of sons in a branching process. The distribution on the sizes of clusters after the computation corresponds to the sizes of trees in the branching process. If a has no mate, $K_a^*(t)$, which is the size of a's cluster after M is applied, equals the size of a's tree. If a has a mate, c, then $K_a^*(t)$ is the sum of sizes of a's tree and of c's tree, where again duplicated qubits are ignored. A schematic example of this branching process is demonstrated in the following figure:

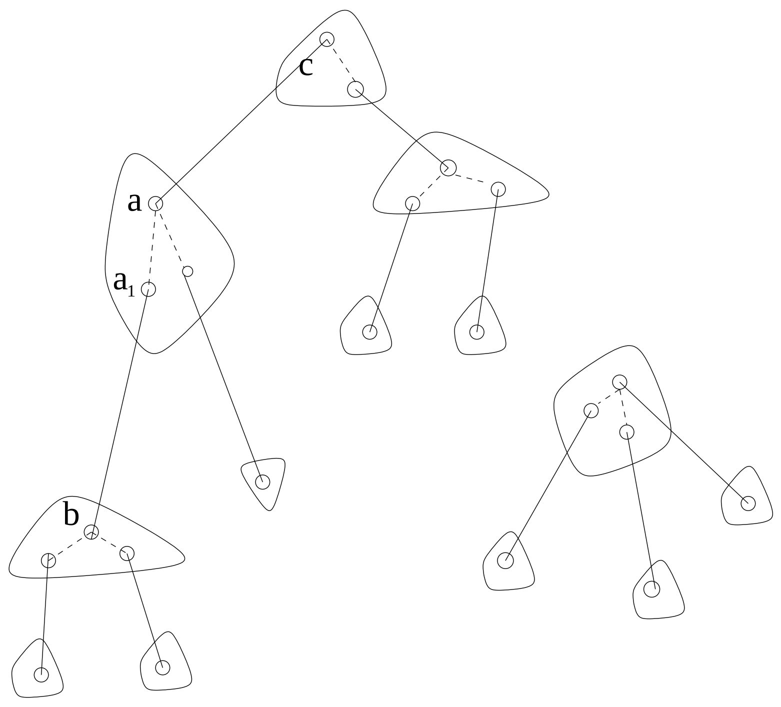


Figure III.1: In this figure, the small circles denote qubits. The large circles describe the configuration of the clusters before the t'th computation step. The edges correspond to gates applied at time t. The dashed lines connect a qubit and its sons, in the corresponding branching process.

We first consider a simplified version of the above process where (1) All gates in M are of fan-in 2, and (2) Gates in M operate only on qubits from different clusters. With these simplifications, the process that generates a tree, defined above, turns out to be a simple branching process, for which we can compute the probability distribution over the total size of two trees, given the distribution on the number of sons.

Lemma 12 *Consider a branching process, where z denotes the number of sons. Let T_a, T_b, be the sizes of two independent trees generated by the branching process. Let $L = 2T_a + 2T_b - 2$. Then if $a > 8$,*

$$Pr(z = j) \leq \frac{1}{a^j} \quad \Rightarrow \quad Pr(L = i) \leq a(\sqrt{2} - 1)^2 (\sqrt{\frac{8}{a}})^i. \tag{7}$$

Proof: For the standard proofs in the following discussion, consult [14]. Let the generating functions[1] of z, T, L be G_z, G_T, G_L. In the following formulas, the first[14] connects the generating functions of the size of a tree T to the number of sons Z By definition of L we have. The second we have by definition of L:

$$G_{L+2}(x) = (G_T(x^2))^2 \ , \ G_T(x) = xG_z(G_T(x)) \tag{8}$$

We don't know the generating function for the variable z since we only have a bound on the distribution. Let us refer to the exponential decaying bound as a pseudo distribution, although it is not normalized to one, and define "pseudo generating functions", (denoted by a tilde), such that $\tilde{G}_z(x)$ will "generate" the exponential decay $\frac{1}{a^i}$:

[1]The generating function $G_A(x)$ of positive integer random variable A, is the expectation of x^A: $E(x^A) = \sum_{i=0}^{\infty} Pr(A = i)x^i$

$\tilde{G}_z(x) = \sum_{i=0}^{\infty} \frac{1}{a^i} x^i = \frac{a}{a-x}$, and $\tilde{G}_T(x)$, $\tilde{G}_L(x)$ will satisfy equations 8: $\tilde{G}_T(x) = x\tilde{G}_z(\tilde{G}_T(x))$, $\tilde{G}_{L+2}(x) = (\tilde{G}_T(x^2))^2$. The coefficients z_i, t_i, l_i in the analytic expansion of $G_Z(x)$ $G_T(x)$, and $G_{L+2}(x)$ are smaller then the corresponding coefficients $\tilde{z}_i, \tilde{t}_i, \tilde{l}_i$ of the "pseudo generating function," when the expansions exist. To prove this, note that the assumptions of the lemma imply $z_i \leq \tilde{z}_i$. Writing equation 8 using analytic expansions, one can show by induction that $t_i \leq \tilde{t}_i$. Using this and writing equation 8 in analytic expansion, we have $l_i \leq \tilde{l}_i$.

We intend to give an upper bound on $\tilde{l}_i$, since this will imply an upper bound on $l_{i+2} = Pr(L = i)$ and therefore on $Pr(L \geq i)$. Given $\tilde{G}_z(x)$, we can solve $\tilde{G}_T(x) = x\tilde{G}_z(\tilde{G}_T(x))$ for $\tilde{G}_T(x)$, which gives a quadratic equation. We choose the "minus" solution of the quadratic equation, which corresponds to the correct choice in the case of $G_T(x)$. It gives $\tilde{G}_T(\frac{a}{8}) = \frac{(\sqrt{2}-1)a}{\sqrt{8}}$, which implies

$$\frac{(\sqrt{2}-1)^2 a^2}{8} = (\tilde{G}_T(\frac{a}{8}))^2 = \tilde{G}_{L+2}(\sqrt{\frac{a}{8}}). \tag{9}$$

$\tilde{G}_T(x)$ is analytic for $|x| < \frac{a}{4}$, so $\tilde{G}_{L+2}(y)$ is analytic at $y = \sqrt{\frac{a}{8}}$, and we can write: $\frac{(\sqrt{2}-1)^2 a^2}{8} = \sum_{i=0}^{\infty} \tilde{l}_i (\sqrt{\frac{a}{8}})^i$, which gives $\tilde{l}_i \leq (\sqrt{2}-1)^2 \frac{a^2}{8} \left(\sqrt{\frac{8}{a}}\right)^i$ that implies the desired result. $\Box$

We use the connection to the branching process defined above, to show an upper bound on the clusters' size after the computation step, given the upper bound before the computation step.

Lemma 13 *Let $a > 8$, and $C = (\sqrt{2}-1)^2 \frac{a^{\frac{3}{2}}}{\sqrt{a}-\sqrt{8}}$.*

$$Pr(k_q(t) \geq j) \leq \frac{1}{a^{j-1}} \quad \Rightarrow \quad Pr(K_q^*(t) \geq i) \leq C(\sqrt{\frac{8}{a}})^i. \tag{10}$$

Proof: Compare our process to the simplified branching process, where identifying $k_q(t) - 1$ with the variable z and $k_q^*(t)$ with L. Our process differs in two aspects: It might connect mates inside a cluster, and a qubit might not have a mate at all, i.e might not have a chance to give birth to children. The only effect of these fact can be to reduce the final cluster. $\Box$

4.2 The Noise Step

We now show that with strong enough noise, the distribution over sizes of clusters is "pulled" by the stochastic collapses back below the original exponential decaying bound:

Lemma 14 *$\exists a > 8$ such that $\forall$ $\eta > 0.97$, if $C = (\sqrt{2}-1)^2 \frac{a^{\frac{3}{2}}}{\sqrt{a}-\sqrt{8}}$ then:*

$$Pr(K^*(t) \geq i) \leq C(\sqrt{\frac{8}{a}})^i \quad \Rightarrow \quad Pr(k(t+1) \geq l) \leq \frac{1}{a^{l-1}}. \tag{11}$$

Proof: We have to calculate how a distribution over sizes of clusters changes because of the stochastic collapses. The probability to reduce the number of qubits in a cluster is binomial so:

$$Pr(k(t+1) \geq l) = \sum_{r=0}^{\infty} Pr(k(t+1) = l + r) = \tag{12}$$

$$\sum_{r=0}^{\infty} \sum_{p=0}^{\infty} Pr(K^*(t) = l + r + p) \begin{pmatrix} l+r+p \\ p \end{pmatrix} \eta^p (1-\eta)^{l+r} \leq \tag{13}$$

$$\sum_{p=0}^{\infty} Pr(K^*(t) \geq l+p) \sum_{r=0}^{\infty} \begin{pmatrix} l+r+p \\ p \end{pmatrix} \eta^p (1-\eta)^{l+r} \leq \tag{14}$$

$$\leq C(\sqrt{\frac{8}{a}})^l \sum_{r=0}^{\infty} (1-\eta)^{r+l} \sum_{p}^{\infty} (\eta\sqrt{\frac{8}{a}})^p \begin{pmatrix} l+r+p \\ p \end{pmatrix} = \tag{15}$$

$$= C(\sqrt{\frac{8}{a}})^l \sum_{r=0}^{\infty} \frac{(1-\eta)^{r+l}}{(1-\eta\sqrt{\frac{8}{a}})^{l+r+1}} = \tag{16}$$

$$= (\sqrt{2}-1)^2 \frac{a^{\frac{5}{2}}}{\eta(\sqrt{a}-\sqrt{8})^2} \left(\frac{\sqrt{8}(1-\eta)}{\sqrt{a}-\eta\sqrt{8}} \right)^l . \tag{17}$$

We want that for all l $Pr(k(t+1) \geq l) \leq a(\frac{1}{a})^l$, so we demand that the factor of the geometrical sequence is smaller than $\frac{1}{a}$ and that the first term is smaller then a. The requirements are satisfied with $a = 30$ and $\eta \geq 0.97$. $\Box$

4.3 Combining All the Pieces of the Proof Together

The above two subsections show that if $\eta > 0.97$, then during the noise step the clusters' sizes shrink in an amount which compensates for their expansion during the computation step. The exponential decaying bound on the clusters' size holds, therefore, in all time steps, by induction.

Theorem 5 *Let Q be a quantum circuit with $\eta > 0.97$, which operates for $T = poly(n)$ time steps. Then the error in $S(Q, i, \eta)$ is polynomially small.*

proof: We show that under the conditions of the theorem, the sizes of q's clusters in the simulation S satisfy for some $a > 8$ $Pr(k_q(t) \geq j) \leq \frac{1}{a^{j-1}}$ and $Pr(K_q^*(t) \geq i) \leq C(\sqrt{\frac{8}{a}})^i$ for $C = (\sqrt{2}-1)^2 \frac{a^{\frac{3}{2}}}{\sqrt{a}-\sqrt{8}}$. This is done by induction on t. The basis of the induction is trivial, as $K_q(0) = 1, K_q^*(0) = 1$ always. The induction step is done in two stages, by lemmas 13 and 14. This shows that the distribution over sizes of clusters is bounded by a decaying exponent. The probability for any cluster to be larger than $O(log(n))$ during the entire simulation is polynomially small. $\Box$

5 Sparse Circuits

A quantum circuit that uses single qubit gates often, is simulated efficiently already for weak noise, since single qubit gates do not participate in the struggle against dis-entanglement; They do not entangle qubits.

Theorem 6 *Let Q be a quantum circuit in which each qubit participates in a many-qubit-gate at most once every ν steps, which operates for $T = poly(n)$ time steps, with*

$$\eta > 1 - 0.03^{\frac{1}{\nu}}. \tag{18}$$

Then the error in $S(Q, i, \eta)$ is polynomially small.

Proof: One can treat every ν time steps as a one time step, in the branching process analysis of the previous section, with an effective noise parameter which is simply the probability for a collapse during this "extended" time step. This probability is exactly $\eta' = 1 - (1-\eta)^{\nu}$. $\Box$

This theorem shows that in order for a quantum computation to be resilient against errors, it needs to apply frequently many-qubit-gates, which generate entanglement.

6 Random circuits

The term "random circuit" means a circuit the *wires* of which are connected randomly. For quantum circuits with two-bit gates this means choosing a random matching between all the qubits. We have performed computer experiments to investigate how the clusters sizes depend on the error rate. To study the random algorithm case, we started with n clusters of size one, and each time step chose a random matching between qubits. A match represented a gate on two qubits, so their clusters were joined. To imitate collapses, we randomly and independently separated a qubit from it's cluster with probability η, each time step. For $\eta \simeq 0.63$, all the clusters remained smaller than $log(n)$.

Let us turn briefly to discuss the other side of the noise scale. and ask what is the expected cost for weak noise. In our computer simulations, for $\eta \simeq 0.63$, a "giant" cluster appeared, with linearly many qubits. This means that the density matrix describing it is exponential, and the simulation S stops being efficient. Analytically we can show that the cost of the simulation S is exponential for $\eta < 0.5$.

Lemma 15 *For $\eta < 0.5$ and Q a random quantum circuit, the simulation $S(Q, i, \eta)$ is exponential, except for polynomially small probability.*

Proof: We prove that when the noise is $\eta < 0.5$ the expected size of the a cluster of any qubit is linear in n. The expected size of a cluster of αn qubits after one computation step is larger than $(\alpha + \alpha(1-\alpha))n = (2-\alpha)\alpha n$ since the probability for qubits in the cluster to be matched outside the cluster is $(1-\alpha)$, and these matching add at least one qubit to the cluster. The expected size of this cluster after a collapse step is this size multiplied by η. After t time steps, the expected size of the cluster is its original size multiplied by $(2-\alpha)^t\eta^t$. For $\eta > 0.5$ this is bigger than 1 for small α, and the cluster will grow until it is of size $(2-\frac{1}{\eta})n$, which is linear in n. The rate of growth of the cluster is exponential, because as long as α is smaller than $2 - \frac{1+\epsilon}{\eta}$, the factor is bigger than $1+\epsilon$. $\Box$

7 One Dimensional Quantum Circuit

In this section we will be interested in the case of a one dimensional array of qubits which operate only on nearest neighbors. In this case, the analysis of the dynamics can be made accurate due to a nice analogy between the dynamics of the clusters in the simulation, and a static percolation process. The reduction to the static percolation case in two dimensions allows me to use standard percolation techniques, by which I show a phase transition at the critical value $\eta = 0.5$. In the supercritical phase, entanglement in the system is only local, the clusters are small, and the simulation is efficient. In the sub-critical phase, the simulation is shown to be exponential, due to the existence of a giant, i.e. linear in size, cluster.

The circuit which I will study will be a circuit of the following type:

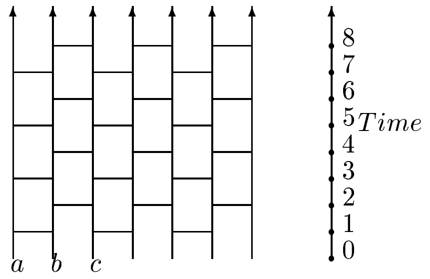


Figure III.2: A one dimensional quantum circuit. Horizontal lines indicate gates operating on the two qubits which the line connects. Qubits interact alternately with qubits to their left or to their right.

It turns out that the dynamics of the clusters in the simulation of a one dimensional circuit are intimately connected with a percolation process on the *quantum circuit* itself.

7.1 The Corresponding Percolation Process

The percolation process on this circuit is defined as follows: For each time step, each vertical edge is erased with independent probability η. This corresponds to the collapse of the qubit in between two time steps. This collapse can be interpreted as an event that disconnects between the qubit's past and future. Connected components in the circuit will correspond to entangled sets of qubits, or to clusters.

Let me now establish this fact more rigorously. There is a natural one to one and measure preserving mapping between events in the probability space of the cluster model and that of the percolation model: A possible list of which qubits collapsed at which time step during the simulation, or what might be called a **fault path**, corresponds to a certain list of vertical lines which were erased in the percolation process, and the probabilities for both events to occur is the same. I will compare an event in the percolation process to the corresponding event in the simulation, and show that there is a one to one correspondence between connected components in the percolation process, and clusters of the simulation.

Lemma 16 Correspondence Lemma: *Two qubits a and b are in the same cluster at time t, in the cluster model, iff (a,t) and (b,t) are connected in the percolation model.*

Proof: Let us prove this by induction on t. It will be useful to define $B(t)$ as the percolation on a strip of the infinite lattice which contains n qubits from time $t = 0$ to t. The base of the induction, $t = 0$ this is true by definition. Assume it is true for t. Let a and b be two qubits entangled at time $t+1$ in the percolation model, so that there is a path between them, using edges with time not larger than $t+1$. This path can be viewed as a concatenation of paths, which are alternately confined to B_{t+1}/B_t and to B_t. Let the qubits where the transitions happen be denoted $X1, ...X_{2k}$. We claim that all these qubits are in the same cluster at time $t+1$

in the cluster model. This is true because by the induction assumption, X_{2i+1} and X_{2i+2} for all $0 \leq i \leq k$, are in the same cluster at time t. None of these qubits collapsed between time steps t and $t+1$ by the assumption that the transition from B_t to B_{t+1} or vice-versa happened along these qubits. So at time $t+1$ the qubit X_{2i+1} is in the same cluster as X_{2i}. Now in the percolation model, there are paths confined to B_{t+1}/B_t connecting the qubits X_{2i} and X_{2i+1}, this means that in the cluster model all these clusters are joined to one in time step $t+1$. But $a(b)$ is in the same cluster as $X_1(X_{2k})$ at time $t+1$, and this completes one side of the proof.

Now suppose a and b are in the same cluster at time $t+1$. From this we know that there exists a set of qubits $X_1, ... X_{2k}$ such that the following conditions are satisfied

1. At time t X_{2i+1} is in a different cluster than X_{2i+2} and X_{2i} is in the same cluster as X_{2i+1}.
2. At time $t+1$ unitary matrices are applied on qubits X_{2i+1} and X_{2i+2}.
3. X_i did not collapse between the time steps t and $t+1$.
4. (A unitary matrix is applied on a and X_1 at $t+1$)

   or

   (X_1 is in the same cluster as a at time t and a did not collapse between t and $t+1$)
5. (A unitary matrix is applied on b and X_{2k} at $t+1$)

   or

   (X_{2k} is in the same cluster as b at time t and b did not collapse between t and $t+1$)

The claim is that there is a path connecting a and b in B_{t+1}. We build this path as a concatenated path from a to X_1 to X_2... to b, where the paths between X_{2i} and X_{2i+1} exist due to the induction assumption, and the paths between X_{2i+1} and X_{2i+2} exist because of conditions 2 and 3. The edges of the path (From a to X_1 and from X_{2k} to b exist because of conditions 4 and 5. □

We can now investigate the percolation model and then translate our findings to the cluster model. Let us first release the restriction to a percolation on a square of size $n \times T$ and consider the infinite (in four directions) case. We show the existence of a phase transition in the probability for an infinite cluster, and this will imply a phase transition in the entangled cluster in the finite case.

7.2 Background for Percolation in Transversal Invariant Lattices

In the study of phase transitions and percolation on infinite lattices, usually one uses the parameter p which indicates the probability for an edge to exist, as opposed to being erased. Hence we can think of p as equal to $1-\eta$. Two quantities are standardly defined: the first quantity, p_c, is the smallest probability p in which the point 0 belongs to an infinite cluster with positive probability:

$$p_c = inf\{p \mid Pr_p(|C(0)| = \infty) > 0\} \tag{19}$$

where $C(0)$ is the connected component of 0. The second quantity is the smallest p in which the expected size of $0'$s cluster is infinite:

$$\chi(p) = E_p(|C(0)|) \tag{20}$$

$$\pi_c = inf\{p \mid \chi(p) = \infty\}. \tag{21}$$

Clearly, $\pi_c \leq p_c$. However, in transversal invariant lattices, they are equal. There is no intermediate phase in which the expected size of $0'$s cluster is infinite, while the probability for this cluster to be infinite equals 0.

Theorem 7 *[164, 1, 62] For translation invariant lattices,*

$$\pi_c = p_c. \tag{22}$$

Below π_c, the probability for a node to be connected to a far away point decays exponentially in the distance between the points.

Theorem 8 *[62] For transversal invariant lattices, $p < \pi_c$, the probability $\tau(x, y : p)$ for x to be connected to y, decays exponentially with the distance:*

$$\tau(x, y : p) \leq exp(-\frac{|x - y|}{\xi(p)}) \tag{23}$$

with $\xi(p) < \infty$.

For $d = 2$, the supercritical phase is characterized by the fact that the probability for crossing a square of size n from left to right at p-percolation, $S_n(p)$, tends to 1 exponentially fast in n.

Theorem 9 *[62] In $d = 2, p > p_c$ then*

$$liminf_{n \mapsto \infty} S_n(p) = 1. \tag{24}$$

We will also need a lemma due to Russo, Seymour and Welsh (The RSW lemma) which essentially asserts that if the probability of crossing a square is sufficiently large, than the probability $R_L(p)$ of crossing a $2L \times L$ rectangle the hard way is also reasonably large.

Lemma 17 *(Russo,Seymour, Welsh)* **The RSW Lemma:***[179, 183, 62] There is a continuous increasing function $F : [0, 1] \mapsto [0, 1]$ such that $F(0) = 0, F(1) = 1$ and*

$$R_L(p) \geq F(S_L(p)). \tag{25}$$

The two theorems together imply that in transversal invariant lattices, there are two phases. One in which the probability to cross large squares tends to 1. In the other phase the probability for connectedness decays exponentially.

7.3 $p_c = 0.5$

The case of the one dimensional quantum computer is easy to deal with analytically, because its infinite version is self dual. Consider the dual lattice of the infinite lattice in lattice in figure III.2. It is defined to be the Z^2 lattice shifted by half a square so as to get $z^2 + (0.5, 0.5)$. An edge in the dual lattice exists iff the crossing edge in the original lattice is absent. Note the self duality: The dual lattice is equivalent to the original one, with η replaced by $1 - \eta$, and the axis t and x exchanged. From self duality it is a standard derivation to show that the critical p is 0.5, and this is what we will do next.

Theorem 10 *In the percolation process defined above*

$$P_c = \frac{1}{2}. \tag{26}$$

Proof: Consider an $n \times (n+1)$ box in the lattice. A crossing from left to right in the original lattice occurs iff there is no crossing from up to down in the dual box.

$$Pr_p(left \longleftrightarrow right) = 1 - Pr_p^d(up \longleftrightarrow down). \tag{27}$$

However, from self duality we have:

$$Pr_p^d(up \longleftrightarrow down) = Pr_{1-p}(left \longleftrightarrow right). \tag{28}$$

For $p = \frac{1}{2}$, we have

$$2Pr_{\frac{1}{2}}(left \longleftrightarrow right) = 1, \tag{29}$$

so the probability for a crossing from the right to the left of the box at $p = 0.5$ is 0.5:

$$Pr_{\frac{1}{2}}(left \longleftrightarrow right) = \frac{1}{2}. \tag{30}$$

Assuming now that $p_c > 0.5$, then for $p = 0.5$ we are in the sub-critical phase. From lemma 8, we know that the probability for two sides to be connected decay exponentially:

$$\frac{1}{2} = Pr_{\frac{1}{2}}(left \longleftrightarrow right) \leq (n+1)^2 exp(\frac{-n}{\xi(\frac{1}{2})}), \tag{31}$$

Which is a contradiction for large n.

Hence we have shown $p_c \leq 0.5$. Assuming that $p_c < 0.5$, there is a regime $(p_c, 1 - p_c)$ in which, according to lemma 9, with non zero probability there is a path crossing a square in the original lattice, from right to left, simultaneously with a path in the dual lattice crossing the dual square from up to down. Such two crossings cannot occur simultaneously because of a corollary of the Jordan theorem, which leads to a contradiction. □

7.4 Phase Transition in Clusters' Size

The phase transition in the infinite percolation model corresponds to a phase transition in the cluster model. We define an order parameter which is the expected size of $K_1(t)$, the cluster containing, say, the first qubit, divided by n, in the limit of large n.

$$\zeta_p = lim_{n \mapsto \infty} E(|K_1(t)|)/n. \tag{32}$$

This order parameter seems to depend on t, the number of time steps which the simulation is running, but in fact this dependence is very weak. I show that ζ_p exhibits a second order phase transition at $p = \frac{1}{2}$. In the sub-critical phase of the percolation model, the probability to be connected decays exponentially with the distance, by lemma 8. By the correspondence lemma 16, we have that the probability for two particles to be connected decays exponentially with the distance between them. Hence

$$E(|K_1(t)|) \leq \sum_i exp(\frac{-i}{\xi(p)}) = const. \tag{33}$$

Dividing by n and taking the limit $n \mapsto \infty$ to give ζ_p, we have that for $p < p_c$, $\zeta_p = 0$.

For the supercritical percolation phase, we show that $\zeta_p > 0$. To see this, we show that the probability for two particles to be in the same cluster at time t is larger than a constant $c > 0$, if $t = \Omega(n)$. This will show that the expected size of $K_1(t)$ at time $t = \Omega(n)$ is cn, so $\zeta_p > 0$ To show that the probability for two qubits to be connected is larger than a constant, we use the rescaling lemma[4], which asserts that the probability $R_L(p)$ for crossing an $L \times 2L$ rectangle from left to right at $p-$percolation approaches 1 exponentially in L.

Lemma 18 *(Aizenman,Chayes,Chayes,Frohlich,Russo)[4] Let $c = \frac{1}{16}$ and $\lambda \in (0,1)$. In $d = 2$,*

$$R_L(p) \geq 1 - c\lambda \longmapsto R_{2^k L}(p) \geq 1 - c\lambda^{2^k} \tag{34}$$

Now consider the following figure:

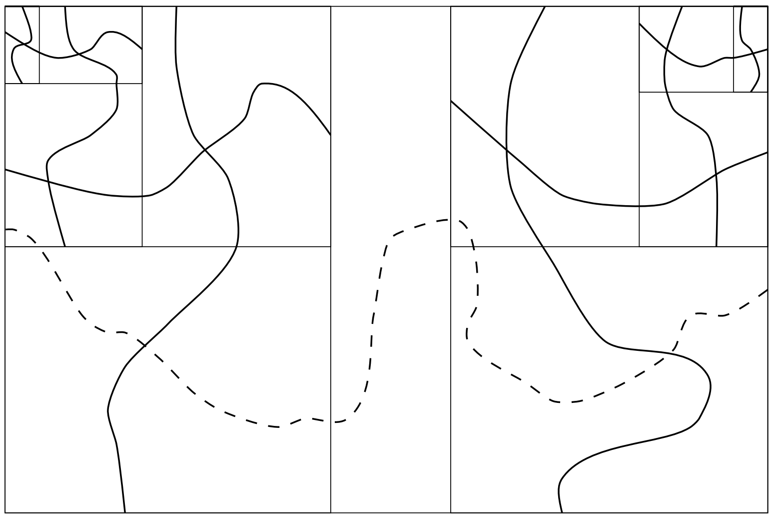


Figure III.3: The path between two qubits at time t.

This figure demonstrates a path from the leftmost qubit at time t to the rightmost qubit at time t. This path is a concatenation of many segments. We start with constructing a path to a qubit at time $t - n$. This is done using paths each connecting the two edges of an $L \times 2L$ rectangle, such that two such paths necessarily cross each other. The first rectangle, i.e. the upper-left small one, is of size $L_0 \times 2L_0$, and the rectangles are getting bigger and bigger, so as to get away from t and get to $t-n$. Each rectangle is twice as big as the former one. The idea is that by the rescaling lemma 18 the paths exist with constant probability. Therefore the size of the smallest rectangle is chosen so that the hypothesis of the rescaling lemma is satisfied. A similar path is constructed for the rightmost qubit. Then a path (in dashed line) connecting the edges of the large rectangle is used. This path connects between the two paths, so that a complete path between the two qubits at time t is achieved. I now prove that indeed the probability for the existence of such a path is bigger than a constant:

Lemma 19 *For $t > n$, and $p > p_c$, there exists a constant $const(p)$ independent of n such that*

$$Pr(a(t) \longmapsto b(t)) \geq const(p). \tag{35}$$

Proof: The events of the paths connecting the edges of the different rectangles are positively correlated. By the FKG inequality[62], this means that the probability for all the paths to exist is larger than the product of the probabilities for each event to occur. Therefore the probability for the path from a to b can be bounded from below:

$$Pr(a(t) \longmapsto b(t)) \geq p^{2L_0} \left(\Pi_{k=1}^{log(n/L_0)} R_{2^k L_0}\right)^2 R_{n/2} \tag{36}$$

We want to use the rescaling lemma, so let us find L_0 which satisfies the hypothesis of the rescaling lemma, i.e $R_{L_0}(p) \geq 1 - \frac{1}{16}\lambda$ for $\lambda \in (0,1)$. Such L_0 exists because of theorem 9 and the RSW lemma (lemma 17)x. We therefore have that

$$Pr(a(t) \longmapsto b(t)) \geq p^{2L_0} \left(\Pi_{k=1}^{log(n/L_0)} (1 - c\lambda^{2^k})\right)^2 (1 - c\lambda^{n/L_0}) \geq p^{2L_0} \left(\Pi_{k=1}^{\infty} (1 - c\lambda^{2^k})\right)^2 > 0. \Box \tag{37}$$

Corollary 1 *For $p > p_c$, $\zeta_p > 0$. For $p < p_c$, $\zeta_p = 0$.*

7.5 The Simulation Makes Small Error for $\eta > 0.5$

We can now investigate the error in the simulation.

Theorem 11 *Let Q be a one dimensional quantum circuit with $\eta > 0.5$, which operates for $T = poly(n)$ time steps. Then the error in $S(Q, i, \eta)$ is polynomially small.*

Proof: We will show that except for polynomially small probability, all the clusters will be of size $O(\log(n))$, during the entire simulation. The correspondence lemma (16) implies that the probability for two qubits to be in the same cluster at time t is equal to the probability they are connected in the percolation model at time t. By lemma 8, the probability for two qubits to be connected decays exponentially in the distance between them, which means that the probability for any qubit to be connected to $\Omega(\log(n))$ qubits is polynomially small. $\Box$

If we use the Monte-Carlo version of the simulation, and do not stop after polynomially many steps, then we can also analyze the case of $\eta < \frac{1}{2}$, and show that the cost of the simulation is exponential with constant probability in this noise regime. This is true since by lemma 19, the expected size of the cluster is linear in n, and since the size of the cluster is at most n, this means that with constant probability the size of the cluster is linear, and the simulation is exponential.

The results can also be extended to higher dimensions, but the critical point in η will increase.

Chapter IV

Correcting Weak Noise

1 Summary

In this chapter I discuss the other side of the scale of noise. I show that quantum computation can be made robust against errors and inaccuracies, when the error rate, η, is smaller than a constant threshold, η_c. This improves on a result by Shor[191] who showed how to perform fault tolerant quantum computation when the probability for an error in a qubit or a gate, η, decays with the size of the computation polylogarithmically, an assumption which is physically unreasonable.

The cost is polylogarithmic in space and time. The result holds for the general noise model, which includes probabilistic errors, decoherence, amplitude damping, depolarization, and systematic inaccuracies in the gates. Fault tolerant computation can be performed with any universal set of gates. The result also holds for quantum particles with $p > 2$ states, namely qupits, and is also generalized to one dimensional quantum computers with only nearest neighbor interactions. No measurements, or classical operations, are required during the quantum computation.

The threshold result was first presented in a paper with Michael Ben-Or[8]. In that paper, we use Calderbank-Shor Steane (CSS) quantum error correcting codes, generalized to qupits, and introduce a new class of CSS codes over F_p, called polynomial codes. It is shown how to apply a universal set of gates fault tolerantly on states encoded by general CSS codes, based on modifications of Shor's procedures, and on states encoded by polynomial codes, where the procedures for polynomial codes have a simple and systematic structure based on the algebraic properties of the code. Geometrical and group theoretical arguments are used to prove the universality of the two sets of gates. Our key theorem asserts that applying computation on encoded states recursively achieves fault tolerance against constant error rate. The generalization to general noise models is done using the framework of quantum circuits with density matrices. Finally, we calculate the threshold to be $\eta_c \simeq 10^{-6}$, in the best case.

I provide new and significantly simpler proofs for most of the known results which are used in the threshold result. For example, I give a simple proof that it suffices to correct bit and phase flips, I significantly simplify Calderbank and Shor's original proof of the correctness of CSS codes. I also give a simple proof of the fact that two-qubit gates are universal. This chapter thus provides a self contained and complete proof for universal fault tolerant quantum computation.

2 Outline

In the last chapter I have shown that quantum computation subjected to strong noise lose their advantage over classical computation devices. However, we should expect that if quantum computers are ever built, they would be made as error free as possible. The question that should be asked is whether there exists a non zero error rate in which noisy quantum computation maintains its computational power. The question of correcting noise cannot be separated from the complexity questions. Without error corrections, the effect of noise will accumulate and ruin the entire computation[207, 65, 172, 163, 21], and hence the computation must be protected. For classical circuits, von Neumann has shown already in 1956 that the computation can be made robust to noise[167]. However, the similar question for quantum systems is much more complicated. Even the simpler question of protecting quantum information is harder than the classical analogue because one must also protect the quantum correlations between the qubits. It was argued by scientists that due to the continuity of quantum states, and to the fact that quantum states cannot be cloned[219], it will be impossible to protect quantum information from noise[207, 148]. Despite these pessimistic beliefs, Shor discovered a scheme to reduce the effect of decoherence[190]. Immediately after that, Calderbank and Shor[59] and Steane[198] showed that good quantum error correcting codes exist, a result which was followed by many explicit examples of quantum codes (e.g.[147, 198]). A theory for quantum error correcting codes was developed[142], and a group theoretical framework for almost all codes was found [60, 61, 115].

The existence of quantum error correcting codes, however, does not imply the existence of noise resistant quantum computation, since due to computation the faults propagate. One must be able to compute without allowing the errors to propagate too much, while error corrections should be made in the presence of noise. Recently Shor[191] showed how to use quantum codes in order to perform fault tolerant quantum computation in the presence of probabilistic errors, when the *error rate*, or the fault probability each time step, per qubit or gate, is polylogarithmically small. This assumption is physically unrealistic. In the paper [10] by Aharonov and Ben-Or we improve this result and show how to perform fault tolerant quantum computation in the presence of constant error rate, as long as the error rate η, is smaller than some constant threshold, η_0. The result holds for a very general noise model, which includes, besides probabilistic errors, also decoherence, amplitude and phase damping, depolarization, and systematic inaccuracies in the gates. Fault tolerance can be achieved using any universal set of gates. The result also holds when working with quantum particles of more than two states, namely "qupits", instead of qubits. Our scheme can be generalized to work also in the case in which the quantum computer is a one dimensional array of qubits or qupits, with only nearest neighbor interactions. No measurements, or classical operations, are required during the quantum computation. The cost is polylogarithmic in the depth and size of the quantum circuit. The main assumption on the noise is that of locality, i.e. the noise process in different gates and qubits is independent in time and space. Such assumptions are made also in the classical scenario, and are likely to hold in physical realizations of quantum computers. Thus, the results described in this chapter settle the question of quantum computation in the presence of local noise.

Before we explain how to make quantum circuits fault tolerant, let us discuss how to protect quantum information against noise using quantum error correcting codes. As in classical linear block codes, a quantum error correcting code encodes the state of each qubit on a *block* of, say, m qubits. The encoded state is sometimes called the *logical* state. The code is said to correct d errors if the logical state is recoverable given that not more than d errors occurred in each block. The difference from classical codes is that quantum superpositions should be recoverable as well, so one should be able to protect quantum coherence, and not only basis states. Calderbank and Shor[59] and Steane [198] were the first to construct such quantum codes, which are now called CSS codes. The basic ingredients are two observations. First, the most general fault, or quantum operation, on a qubit can be described as a linear combination of four simple operations: The identity, i.e. no error at all, a bit flip ($|0\rangle \leftrightarrow |1\rangle$), a phase flip ($|0\rangle \longmapsto |0\rangle, |1\rangle \longmapsto -|1\rangle$) or both a bit flip and a phase flip. We give here a simple proof of the fact, first proven by Bennett *at. al.*[39], that it suffices to correct only these four basic errors. In order to correct bit flips, one can use analogues of classical error correcting codes. To correct phase flips, another observation comes in handy: A phase flip is actually a bit flip in the Fourier transformed basis, and hence, one can correct bit flips in the original basis, and then correct bit flips in the Fourier transformed basis, which translates to correcting phase flips in the original basis. In this chapter I give an alternative proof for CSS codes, which generalized to qupits, and is significantly simpler than the original proof of Calderbank and Shor[59].

We define a new class of quantum codes, which are called *polynomial codes.* The idea is based on a theorem by Schumacher and Nielsen[185] which asserts that a quantum state can be corrected only if no information about the state has leaked to the environment through the noise process. More precisely, if the reduced density matrix on any t qubits does not depend on the logical qubit, then there exists a unitary operation which recovers the original state even if the environment interacted with t qubits, i.e. t errors occurred. This is reminiscent of the situation in classical secret sharing schemes: We should divide the "secret", i.e. the logical qubit, among many parties (i.e. physical qupits) such that no t parties share any information about the secret. This analogy between secret sharing and quantum codes suggests that secret sharing schemes might prove useful in quantum coding theory. Here, we adopt the scheme suggested by Shamir[187], who suggested to use random polynomials, evaluated at different points in a field of p elements, as a way to divide a secret among a few parties. A random polynomial of degree d is chosen, and then each party gets the evaluation of the polynomial at a different point in the field F_p. The secret is the value of the polynomial at 0. To adopt this scheme to the quantum setting, we simply replace the random polynomial by a superposition of all polynomials, to get a quantum code. It turns out that we get a special case of the CSS codes over F_p. Polynomial codes are useful from many aspects. First, they have a very nice algebraic structure, which allows to manipulate them easily. This allows us to apply fault tolerant operations on states encoded by polynomial codes in a simple and systematic way, as we will see later. Polynomial codes might also have nice applications to other areas in quantum information theory, as was demonstrated recently by Gottesman *et al*[79] who used polynomial codes for quantum secret sharing. We will use both polynomial codes and general CSS codes in our fault tolerant scheme.

In order to protect a quantum computation against faults, one can try to compute not on the states themselves, but on states encoded by quantum error correcting codes. Each gate in the original computation will be replaced by a "procedure", which applies the encoded gate on the encoded state. Naturally, in order to prevent accumulation of errors, error corrections should be applied frequently. Unfortunately, computing on encoded quantum states does not automatically provide protection against faults, even if error corrections are applied after every procedure. The problem lies in the fact that during the computation of the procedure, faults can *propagate* to "correct" qubits. This can happen if a gate operates on a damaged qubit and some "correct" qubits- in general, this can cause all the qubits that participate in the gate to be damaged. The procedures should thus be designed carefully, in such a way that a fault during the operation of the procedure can only effect a small number of qubits in each block. We refer to such procedures as fault tolerant procedures. It turns out that many gates can be applied *bitwise*, meaning that applying the gate on each qubit separately, achieves the desired gate on the encoded state. Unfortunately, not all gates can be applied fault tolerantly in such a simple way, and sometimes a very complicated procedure is needed. We are actually looking for a pair consisting of a quantum code which can correct d errors, and a corresponding *universal* set of gates, such that their procedures, with respect to the code, allow one fault to spread to at most d qubits. Since the error corrections, encoding, and decoding procedures are also subjected to faults, we need them to be fault tolerant too.

In [191], Shor introduced a universal set of gates, which we denote by $\mathcal{G}_1$, and showed how to apply the gates fault tolerantly on encoded states. It turns out that almost all of the gates in $\mathcal{G}_1$ can be applied bitwise. The complicated procedures are the fault tolerant error correction and encoding, and the Toffoli gate. Shor used noiseless measurements and classical computation extensively in these procedures. Here, we modify Shor's procedures so that they can be performed within the framework of our model, i.e. all operations are quantum operations, subjected to noise. Basically, we follow Shor's constructions, with some additional tricks. I remark here that an effort is made to show that fault tolerance can be achieved without using measurements and classical operations conditioned on the results of these measurements. This fact is desirable because of two reasons. One is purely theoretical: One would like to know that classical operations are not essential, and that the quantum model is complete in the sense that it can be made fault tolerant within itself. The other reason is practical: In some suggestions for physical realizations of quantum computers, such as the NMR computer, it might be very hard to incorporate measurements and classical computations during the quantum computation. Thus, we require that all procedures are done using only quantum computation.

For polynomial codes, we introduce a new universal set of gates which we denote by $\mathcal{G}_2$, and show how to perform these gates fault tolerantly on states encoded with polynomial codes. Here is where the advantage of polynomial codes come into play. Unlike the construction of the fault tolerant Toffoli gate for CSS codes, (and other fault tolerant constructions) which is a long and complicated sequence of tricks, the fault tolerant procedures for polynomial codes all have a common relatively simple structure, which uses the algebraic properties of the polynomials. To perform a gate in $\mathcal{G}_2$, we apply two steps. First the gate is applied pit-wise. This always achieves the correct operation of the gate, but sometimes the final state is encoded by polynomial codes with degree which is twice the original degree d. The second step is therefore a degree reduction, which can be applied using interpolation techniques, as the quantum analogue of the degree reductions used by Ben-Or, Goldvasser and Wigderson in [46] for fault tolerant distributed computation with classical polynomial codes. Thus, the fault tolerant procedures for polynomial codes have a simple systematic structure.

Next, we prove the universality of the two sets of gates which we use, $\mathcal{G}_1$ for the CSS codes and $\mathcal{G}_2$ for the polynomial codes. Universality of a set of gates means that the subgroup generated by the set of gates is dense in the group of unitary operations on n qubits, $U(2^n)$, (for qupits the group is $U(p^n)$). Kitaev[135] and Solovay[196] showed, by a beautiful proof which uses Lie algebra and Lie groups, that universality of a set $\mathcal{G}$ guarantees that any quantum computation can be approximated by a circuit which uses $\mathcal{G}$ with only polylogarithmic cost. Our proof of universality of $\mathcal{G}_1$ is a simple reduction to a set of gates, which was shown to be universal by Kitaev[135]. The proof that $\mathcal{G}_2$ is universal is more involved. The idea is first to generate, using gates from $\mathcal{G}_2$, matrices with eigenvalues which are not integer roots of unity, a fact which is proved using field theoretical arguments. This enables us to show that certain $U(2)$ subgroups are contained in the subgroup generated by $\mathcal{G}_2$, where these $U(2)$ operate on non orthogonal subspaces, which together span the entire space. To complete the proof, we prove a geometrical lemma which asserts that if A and B are non orthogonal then $U(A) \cup U(B)$ generate the whole group $U(A \oplus B)$. Thus we have that both sets of gates which are used in our scheme are universal.

We proceed to explain how a quantum code accompanied with fault tolerant procedures for a universal set of gates can be used to enhance the reliability of a quantum circuit. We start with our original quantum circuit, M_0. The circuit M_1 which will compute on encoded states is defined as a simulation of M_0. A qubit in M_0 transforms to a block of m qubits in M_1, and each gate in M_0 transforms to a fault tolerant procedure in M_1 applied on the corresponding blocks. Note that procedures might take more than one time step to perform. Thus, a time step in M_0 is mapped to a time interval in M_1, which is called a *working period*. In order to prevent accumulation of errors, at the end of each working period an error correction is applied on each block. The working period now consists of two stages: a computation stage and a correction stage. The idea is therefore to apply alternately computation stages and correction stages, hoping that during the computation stage the damage that accumulated is still small enough so that the corrections are still able to correct it.

We now want to show that the reliability of the simulating circuit is larger than that of the original circuit. In the original circuit, the occurrence of one fault may cause the failure of the computation. In contrast, the simulating circuit can tolerate a number of errors, say k, in each procedure, since they are immediately corrected in the following error correcting stage. The effective error rate of M_1 is thus the probability for more than k faults in a procedure. This effective error rate will be smaller than the actual error rate η, if η is smaller than a certain threshold, η_c, which depends on the size of the procedures, and on the number of errors which the error correcting code can correct. If $\eta < \eta_c$, M_1 will be more reliable than M_0. For fault tolerance, we need the effective error rate to be polynomially small, to ensure that with constant probability no effective error occurred. In [191] Shor uses quantum CSS codes which encode one qubit on polylog(n) to achieve fault tolerance with the error rate η being polylogarithmically small.

The main result of the thesis, the "threshold result", improves Shor's result and shows that quantum computation can be robust even in the presence of constant error rate. Such an improvement from a constant error rate η to polynomially small effective error s rate is impossible to achieve in the above method. Instead, reliability in the presence of constant error rate is achieved by applying the above scheme recursively. The idea is simple: as long as the error rate is below the threshold, the effective error rate of the simulating circuit can be decreased by simulating it again, and so on for several levels, say r, to get the final circuit M_r. It will suffice that each level of simulation improves only slightly the effective error rate, since the improvement is exponential in the number of levels. Such a small improvement can be achieved when using a code of constant block size.

The final circuit, M_r, has an hierarchical structure. Each qubit in the original circuit transforms to a block of qubits in the next level, and they in their turn transform to a block of blocks in the second simulation and so on. A gate in the original circuit transforms to a procedure in the next level, which transforms to a larger procedure containing smaller procedures in the next level and so on. The final circuit computes in all the levels:

The largest procedures, computing on the largest (highest level) blocks, correspond to operations on qubits in the original circuit. The smaller procedures, operating on smaller blocks, correspond to computation in lower levels. Note, that each level simulates the error corrections in the previous level, and adds error corrections in the current level. The final circuit, thus, includes error corrections of all the levels, where during the computation of error corrections of larger blocks smaller blocks of lower levels are being corrected. The lower the level, the more often error corrections of this level are applied, which is in correspondence with the fact that smaller blocks are more likely to be quickly damaged.

The analysis of the scheme turns out to be quite complicated. To do this, we generalizes the works of Tsirelson [132] and Gác's [108] to the quantum case. First, we distinguish between two notions: The error in the state of the qubits, i.e. the set of qubits which have errors, and the actual faults which took place during the computation, namely the fault path, which is a list of points in time and place where faults had occurred (in a specific run of the computation.) We then define the notion of sparse errors and sparse fault paths. Naturally, due to the hierarchical structure of this scheme, these notions are defined recursively. A block in the lowest level is said to be close to it's correct state if it does not have too many errors, and a higher level block is "close" to it's correct state if it does not contain too many blocks of the previous level which are far from their correct state. If all the blocks of the highest level are close to being correct, we say that the set of errors is *sparse*. Which set of faults does not cause the state to be too far from correct in the above metric? The answer is recursive too: A computation of the lowest level procedure is said to be undamaged if not too many faults occurred in it. Computation of higher level procedures are not damaged if they do not contain too many lower level procedures which are damaged. A fault path will be called *sparse*, if the computations of all the highest level procedures are not damaged. The proof of the threshold result is done by showing that the probability for "bad" faults, i.e. the probability for the set of faults not to be sparse decays exponentially with the number of levels, r. Thus, bad faults are negligible, if the error rate is below the threshold. This is the easy part. The more complicated task is to show that the "good" faults are indeed good: i.e. if the set of faults is sparse enough, then the set of errors is also kept sparse until the end of the computation. This part is done using induction on the number of levels r, and is quite involved.

So far, we have dealt only with probabilistic faults. Actually, the circuit generated by the above recursive scheme is robust also against general local noise. We prove this by writing the noise operator on each qubit as the identity plus a small error term. Expanding the error terms in powers of η, we get a sum of terms, each corresponding to a different fault path. The threshold result for general noise is again proved by dividing the faults to good and bad parts. First, we show that the bad part is negligible i.e. the norm of the sum of all terms which correspond to non-sparse fault paths is small. The proof that the good faults are indeed good, i.e. that the error in the terms corresponding to sparse fault paths is sparse, is based on the proof for probabilistic errors, together with some linearity considerations.

So far, what we have shown is that fault tolerant computation can be achieved using one of the two sets of gates, $\mathcal{G}_1, \mathcal{G}_2$. In fact, fault tolerant computation can be performed using any universal set of gates. This last generalization is shown by first designing a circuit which uses gates from $\mathcal{G}_1$ or $\mathcal{G}_2$, and then approximating each gate in the final circuit, up to constant accuracy, by a constant number of gates from the desired set of gates which we want to work with. This completes the proof of the threshold result in full generality.

The threshold result is shown to hold also for one dimensional quantum circuits. These are systems of n

qubits arranged in a line, where gates are applied only on nearest neighbor qubits, and all qubits are present from the beginning. In this model, one cannot insert a new qubit between two existing qubits, in order to preserve the geometry of the arrangements. However, the non unitary operation which initializes a qubit in the middle of the computation is allowed. To apply fault tolerant procedures in this case, we use swap gates in order to bring far away qubits together. We show that these swap gates do not cause the error to propagate too much.

Finally, it is left to estimate the exact threshold in all the above cases. We give a general simple formula to calculate the threshold in the error rate, given the fault tolerant procedures. This formula depends only on two parameters of these procedures: the size of the largest procedure used in our scheme, and the number of errors which the code corrects. The size of the procedures depend strongly on many variables, such as the geometry of the system, the exact universal set of gates which is used, and whether noiseless classical computation can be used during the computation. We give here an estimation of the threshold in one case, in which no geometry is imposed on the system, error free classical operations and measurements can be performed during the computation, and the codes which are used are polynomial codes. The threshold which is achieved in this case is $\simeq 10^{-6}$. We did not attempt to optimize the threshold, and probably many improvements can be made which can save a lot of gates in our suggested procedures.

A substantial part of the results in this chapter were published in [8]. The threshold result was independently described also by Knill, Laflamme and Zurek[140], and by Kitaev[137, 135]. However, this was done assuming that classical computation can be done instantaneously, even on n bits, and assumption which makes the analysis a lot easier. These constructions were done using different quantum correcting codes, and required measurements in the middle of the quantum computation. Kitaev has shown another, very elegant way to achieve fault tolerance in a different model of quantum computation, which works with anyons[138]. After the results were first published, it was shown by Gottesman[116] that fault tolerant procedures for the universal set of gates introduced by Shor can be constructed for any stabilizer code, using basically the same technique as presented by Shor for the CSS codes, a result which was generalized to qupits[118]. Fault tolerant quantum computation in low dimensions is also discussed by Gottesman[119]. Non binary codes were defined independently also by Chuang[67] and Knill[144].

3 Quantum Error Correcting Codes

Here we recall the definition of quantum codes, and of Calderbank-Shor-Steane (CSS) codes[59, 198]. These codes were first presented for qubits, that is over the field F_2. We give here a proof for the fact that CSS codes are quantum codes, which is significantly simpler than the original proof[59], and generalize it to F_p for any $p > 2$. On our way, we use the fact that it suffices to correct phase flips and bit flips, in order to correct general errors. This was first proved by Bennett *et. al.*[39], using a beautiful argument which involved the notion of group symmetry and a quantum operation called "twirl". We give here a simple proof of this theorem, which uses merely linearity of quantum operators, and is based on an idea by Steane[198].

We then define a new class of quantum codes, which we call polynomial quantum codes. These are a special case of CSS codes over F_p. These codes are the quantum analogue of polynomial codes defined by Shamir[186] and are useful for fault tolerance by analogy to Ben-Or *et. al.*[46] who used random polynomials for distributed fault tolerant classical computation. This section assumes basic knowledge in classical error correcting codes, which can be found in [153].

3.1 Quantum Codes

Definition 11 *A $[k,m]$-quantum code is a subspace of dimension k in the Hilbert space of m qubits.*

A word in the code is any vector in the subspace, or more generally any density matrix ρ supported on the subspace. We say that the word had d errors in qubits $q_1,..q_d$ if a general physical operator on these d qubits, $\mathcal{E}(q_1,..q_d)$ was applied on ρ. A procedure $\mathcal{R}$, i.e. a sequence of gates, is said to correct an error $\mathcal{E}(q_1,..q_d)$ on ρ if

$$\mathcal{R}\circ\mathcal{E}(q_1,..q_d)\circ\rho=\rho \tag{1}$$

Definition 12 *Let C be a $[k,m]$ quantum code. Let $\{|\alpha_i\rangle\}_{i=1}^k$ be a basis for this quantum code. C is said to correct d errors if there exists a sequence of quantum gates $\mathcal{R}$, called an error correction procedure, which corrects any d errors on any d qubits, on any word in the code, i.e for any i,j and $\mathcal{E}(q_1,..q_d)$*

$$\mathcal{R}\circ\mathcal{E}(q_1,..q_d)\circ|\alpha_i\rangle\langle\alpha_j|=|\alpha_i\rangle\langle\alpha_j| \tag{2}$$

Note that during the operation of $\mathcal{R}$, it is allowed to use extra qubits, but at the end all ancillary qubits are discarded, and the original state should be recovered.

3.2 From General Errors to Bit Flips and Phase Flips

A crucial and basic fact[39] in quantum error corrections is that it suffices to correct two basic errors, called bit flips and phase flips, in a small number of qubits, in order to correct a general error on these qubits. Our proof of this fact uses merely linearity of quantum operators. Let us define basic errors as those in which one of the Pauli unitary operators occur:

$$\mathcal{I}=\begin{pmatrix}1&0\\0&1\end{pmatrix},\sigma_x=\begin{pmatrix}0&1\\1&0\end{pmatrix},\sigma_y=\begin{pmatrix}0&1\\-1&0\end{pmatrix},\sigma_z=\begin{pmatrix}1&0\\0&-1\end{pmatrix}. \tag{3}$$

σ_x is called a bit flip, since it takes $|0\rangle$ to $|1\rangle$ and $|1\rangle$ to $|0\rangle$. σ_z is called a phase flip, as it multiplies $|1\rangle$ by a minus sign, while leaving $|0\rangle$ as is. σ_y is a combination of the two, since $\sigma_y=\sigma_z\sigma_x$. A basic error on d qubits is obtained by taking tensor products of d of the above Pauli matrices. More formally, consider all possible strings e of length d in the alphabet I,x,y,z. The set of 4^d matrices:

$$\sigma_e=\sigma_{e_d}\otimes\cdots\sigma_{e_1},\quad e\in\{1,x,y,z\}^d \tag{4}$$

is a basis for operators on the Hilbert space of d qubits. Before we show that it suffices to correct basic errors, we need two definitions:

Definition 13 *We say that $\mathcal{R}$ corrects basic errors of length d in C if for any two basis states of the code, $|\alpha\rangle$ and $|\alpha'\rangle$, and any basic error of length d, σ_e we have*

$$\mathcal{R} \circ \sigma_e|\alpha\rangle\langle\alpha'|\sigma_e^\dagger = |\alpha\rangle\langle\alpha'| \tag{5}$$

Definition 14 *We say that $\mathcal{R}$ detects basic errors of length d in C if for any two basis states of the code, $|\alpha\rangle$ and $|\alpha'\rangle$, and any two different basic error of length d, $\sigma_e \neq \sigma_{e'}$,*

$$\mathcal{R} \circ \sigma_e|\alpha\rangle\langle\alpha'|\sigma_{e'}^\dagger = 0 \tag{6}$$

The reason for the fact that such $\mathcal{R}$ is said to detect errors is that indeed, if at the end of the procedure $\mathcal{R}$ the error e is written down on an ancilla, $\mathcal{R} \circ \sigma_e|\alpha\rangle = |\alpha\rangle \otimes |e\rangle$, then for two different errors the two ancilla states are orthogonal, and thus when discarding the ancilla qubits we get zero. In the remaining of the chapter, I will only use error correcting procedures which detect the errors. We can now prove the following:

Theorem 12 *Let C be a quantum code, and let $\mathcal{R}$ correct and detect any basic error of length d in C. Then R corrects any general error on any d qubits in C.*

Proof: We recall that any noise operator $\mathcal{L}$ from d to d' qubits can be written as a combination of three operators: adding $d + d'$ blank qubits, applying a unitary transformation on the $2d + d'$ and discarding $2d$ qubits. The last operator which discards qubits will be denoted as $\mathcal{T}$. Combining the first two operations, we have in the most general case the following operation on d qubits. For all $0 \leq i \leq 2^d - 1$,

$$|i\rangle \longmapsto |0^{d+d'}\rangle|i\rangle \longmapsto \sum_{j=0}^{2^d-1} |A_i^j\rangle \otimes |j\rangle \tag{7}$$

where all the coefficients are put into the vectors $|A_i^j\rangle$ which are not unit vectors. We now use the fact that the basic errors,σ_e, on d qubits form an orthonormal basis with respect to the inner product $(V,U) = \frac{1}{2^d}tr(VU^\dagger)$ for $2^d \times 2^d$ matrices. We define A, a vector of matrices, or a matrix with entries which are vectors, by $A_{i,j} = |A_i^j\rangle$. For any basic error vector e of length d, define $|A_e\rangle = (A, \sigma_e) = \frac{1}{2}tr(A \cdot \sigma_e^\dagger)$, where A should be thought of as a vector of matrices, and thus $A\sigma_e^\dagger$ is also a vector of matrices, the trace of which is a vector. Due to the fact that the basic errors are orthonormal, we have that $\sum_j |A_i^j\rangle \otimes |j\rangle = \sum_e |A_e\rangle \otimes \sigma_e|i\rangle$, or equivalently the two first steps of the noise operator can be written as:

$$\mathcal{L} = \sum_{e,e'} |A_e\rangle\langle A_{e'}| \otimes \sigma_e \cdot \sigma_{e'}^\dagger \tag{8}$$

Now, $\mathcal{R}$ will indeed correct $\mathcal{E} = \mathcal{T} \circ \mathcal{L}$, by the following:

$$\mathcal{R} \circ \mathcal{E} \circ |\alpha\rangle\langle\alpha'| = \mathcal{R} \circ \mathcal{T} \circ \sum_{e,e'} |A_e\rangle\langle A_{e'}| \otimes \sigma_e|\alpha\rangle\langle\alpha'|\sigma_{e'}^\dagger = (\mathcal{T} \circ \sum_{e} |A_e\rangle\langle A_e|) \otimes |\alpha\rangle\langle\alpha'| = |\alpha\rangle\langle\alpha'| \tag{9}$$

where in the second equality we have used the fact that $\mathcal{T}$ commutes with $\mathcal{R}$ because they operate on different qubits, and that $\mathcal{R}$ corrects and detects errors. In the last equality we used the fact that $\mathcal{T} \circ \sum_e |A_e\rangle\langle A_e| = c$ does not depend on $|\alpha\rangle, |\alpha'\rangle$. Choosing $|\alpha\rangle = |\alpha'\rangle$, $\mathrm{Tr}(\mathcal{R} \circ \mathcal{E} \circ |\alpha\rangle\langle\alpha|) = \mathrm{Tr}(c|\alpha\rangle\langle\alpha|) = c$, but on the other hand $\mathcal{R} \circ \mathcal{E}$ is trace preserving so $c = 1$. □

3.3 Calderbank-Shor-Steane Codes

We give here the definition of CSS codes, which is slightly modified from the definition in [59]. A linear code of length m and dimension k is a subspace of dimension k in F_2^m, where F_2^m is the m dimensional vector space over the field of F_2 of two elements. Let C_1, C_2 be linear code such that $\{0\} \subset C_2 \subset C_1 \subset F_2^m$, and let us define a quantum code by taking the superpositions of all words in a coset of C_2 in C_1 to be one basis word in the code. We have:

$$\forall a \in C_1/C_2 : |S_a\rangle = \frac{1}{\sqrt{2^{dim(C_2)}}} \sum_{w \in C_2} |w + a\rangle. \tag{10}$$

Note that $|S_a\rangle$ is well defined and does not depend on the representative of the coset since if $(a1 - a2) \in C_2$ then $|S_{a1}\rangle = |S_{a2}\rangle$. Also, for different cosets the vectors are orthogonal, because if $(a1-a2) \notin C_2$, then $\langle S_{a1}|S_{a2}\rangle = 0$. Thus, this defines a basis for a subspace of dimension $2^{dim(C_1)-dim(C_2)}$, which is our quantum code. Note that the support of $|S_a\rangle$ are words in the code C_1. We will see that bit flips can be corrected using classical error correction techniques for the code C_1. Before we discuss how to correct phase flips, let us define a very important quantum gate on one qubit, called the Hadamard gate, or the Fourier transform over F_2:

$$H = H^{-1} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \tag{11}$$

Observe that

$$H\sigma_z H^{-1} = \sigma_x \tag{12}$$

meaning that a phase flip transforms to a bit flip in the Fourier transform basis. Applying the Hadamard gate on each qubit in $|S_a\rangle$, we get the state:

$$|C_a\rangle = H \otimes H \otimes \cdots \otimes H|S_a\rangle = \frac{1}{\sqrt{2^{m+dim(C_2)}}} \sum_{b=0}^{2^m-1} \sum_{w \in C_2} (-1)^{(w+a)\cdot b}|b\rangle = \frac{1}{\sqrt{2^{m-dim(C_2)}}} \sum_{b \in C_2^\perp} (-1)^{a\cdot b}|b\rangle \tag{13}$$

which is a superposition of words in $C_2^\perp$, and so to correct phase flips, one transforms to the Fourier basis and corrects bit flips in the code $C_2^\perp$. These ideas lead to the following theorem by Calderbank and Shor[59]. We give here a simple proof of this theorem, based on theorem 12.

Theorem 13 *Let C_1 and $C_2^\perp$ be linear codes over F_2, of length m, such that $0 \subset C_2 \subset C_1 \subset F_2^m$, and such that $C_2^\perp, C_1$ can correct t errors. Then the subspace spanned by $|S_a\rangle$ for all $a \in C_1/C_2$ is a $[2^{dim(c_1)-dim(C_2)}, m]$ quantum code which can correct t errors. The error correction procedure, $\mathcal{R}$, is constructed by correcting pit flips with respect to C_1 in the S-basis, rotating to the C-basis by applying Fourier transform bit-wise, correcting with respect to $C_2^\perp$ and rotating back to the S-basis.*

Proof: We define the procedure $\mathcal{R}_{C_1}$ to be a unitary embedding of m qubits to $2m$ qubits, by

$$\mathcal{R}_{C_1}|i\rangle = |w\rangle \otimes |e\rangle \tag{14}$$

for each $i \in F_2^m$, where $w \in C_1$ is a string of minimal distance to i, and $e \in \{0,1\}^m$ satisfies $w + e = i$. Since this is a one to one transformation, it is a unitary embedding, and is a possible quantum operator. Let $e_b \in \{0,1\}^m$

have at most t $1'$s. Let $\mathcal{E}_b$ be an error operator which is a tensor product of bit flips (σ_x) where e_b is one and identity on the other coordinates. Then for any $|\alpha\rangle, |\alpha'\rangle$ supported on C_1 we have

$$R_{C_1} \circ \mathcal{E}_b \circ |\alpha\rangle\langle\alpha'| = |\alpha\rangle\langle\alpha'| \otimes |e_b\rangle\langle e_b| \tag{15}$$

$\mathcal{R}_{C_2^\perp}$ is defined similarly:

$$\mathcal{R}_{C_2^\perp}|j\rangle = |w\rangle \otimes |e\rangle \tag{16}$$

where $w \in C_2^\perp$ is a string of minimal distance to j, and $e \in \{0,1\}^m$ satisfies $w + e = j$. Let $e_f \in \{0,1\}^m$ have at most t $1'$s (e_f for phase flips.). Let $\mathcal{E}_f$ be an error operator which is a tensor product of phase flips (σ_z) where e_f is one and identity on the other coordinates. Then for any $|\beta\rangle, |\beta'\rangle$ supported on $C_2^\perp$ we have

$$R_{C_2^\perp} \circ \mathcal{E}_f \circ |\beta\rangle\langle\beta'| = |\beta\rangle\langle\beta'| \otimes |e_f\rangle\langle e_f| \tag{17}$$

Denote by $\mathcal{H}$ the operator of applying H on every qubit: $\mathcal{H} = H \otimes H \otimes \cdots \otimes H$. We claim that the operator

$$\mathcal{R} = \mathcal{T}_f \circ \mathcal{T}_b \circ \mathcal{H} \circ \mathcal{R}_{C_2^\perp} \circ \mathcal{H} \circ \mathcal{R}_{C_1} \tag{18}$$

is the desired error correcting procedures. $\mathcal{T}_f, \mathcal{T}_b$ are the operators discarding the qubits added for $\mathcal{R}_{C_2^\perp}, \mathcal{R}_{C_1}$, respectively.

By theorem 12, it is enough to show that this procedure corrects and detects d basic errors. To show that it corrects d basic errors, write the error vector e in two parts, e_b and e_f as follows: e_b will be 1 in the coordinates where σ_x or σ_y occurred, and 0 elsewhere. e_f will be 1 in the coordinates where σ_z or σ_y occurred, and 0 elsewhere. We can therefore write the error operator $\mathcal{E}$ as a product of two operators, $\mathcal{E} = \mathcal{E}_b\mathcal{E}_f$. Now, using, in the following order, equation 18, the fact that $\mathcal{R}_{C_1}$ commutes with $\mathcal{E}_f$, equation 15,12 and 17, we have the desired result:

$$\begin{aligned}
\mathcal{R} \circ \mathcal{E} \circ |\alpha\rangle\langle\alpha'| \quad = \quad & \mathcal{T}_f \circ \mathcal{T}_b \circ \mathcal{H} \circ \mathcal{R}_{C_2^\perp} \circ \mathcal{H} \circ \mathcal{E}_f \circ \mathcal{R}_{C_1} \circ \mathcal{E}_b \circ |\alpha\rangle\langle\alpha'| = \\
& \mathcal{T}_f \circ \mathcal{T}_b \circ \mathcal{H} \circ \mathcal{R}_{C_2^\perp} \circ \mathcal{H} \circ \mathcal{E}_f \circ |\alpha\rangle\langle\alpha'| \otimes |e_b\rangle\langle e_b| = \\
& \mathcal{T}_f \circ \mathcal{H} \circ \mathcal{R}_{C_2^\perp} \circ \mathcal{H} \circ \mathcal{E}_f \circ |\alpha\rangle\langle\alpha'| = \\
& \mathcal{T}_f \circ \mathcal{H} \circ \mathcal{R}_{C_2^\perp} \circ (\mathcal{H} \circ \mathcal{E}_f \circ \mathcal{H}) \circ (\mathcal{H} \circ |\alpha\rangle\langle\alpha'|) = \\
& \mathcal{T}_f \circ \mathcal{H} \circ (\mathcal{H} \circ |\alpha\rangle\langle\alpha'|) \otimes |e_f\rangle\langle e_f| = |\alpha\rangle\langle\alpha'|.
\end{aligned} \tag{19}$$

It is left to show that $\mathcal{R}$ also detects d basic errors. This follows from the fact that if $e \neq e'$, then either $e_f \neq e'_f$ or $e_b \neq e'_b$. Now reducing two orthogonal vectors gives zero, i.e. if $e_b \neq e'_b$ $\mathcal{T}_b \circ |e_b\rangle\langle e'_b| = 0$ and if $e_f \neq e'_f$ $\mathcal{T}_f \circ |e_f\rangle\langle e'_f| = 0$. $\Box$.

3.4 CSS codes over F_p

The theory of quantum error corrections can be generalized to quantum computers which are composed of quantum particles of $p > 2$ states, called qupits. To generalize the notion of bit flips and phase flips to qupits define the following two matrices:

- B : $\quad B|a\rangle = |(a+1)\text{mod p}\rangle$

- $P:\quad P|a\rangle = w^a|a\rangle$

where $w = e^{\frac{2\pi i}{p}}$. We will consider combinations of powers of these matrices, i.e. the p^2 matrices,

$$B^cP^{c'}, \quad \forall c, c' \in F_p. \tag{20}$$

This set can be easily seen to be an orthonormal basis for the set of $p \times p$ complex matrices, with the inner product $(U,V) = \frac{1}{p}tr(UB^\dagger)$. Like in the case of qubits, errors of type B transform to errors of type P and vice versa, via a Fourier transform, which is defined to be

$$W : |a\rangle \longmapsto \frac{1}{\sqrt{p}}\sum_{b\in F} w^{ab}|b\rangle. \tag{21}$$

And it can be easily checked that

$$\forall c \in F_p, \quad WP^cW^{-1} = B^c \quad , \quad WB^cW^{-1}P^c \tag{22}$$

We can therefore define CSS codes over F_p. The statements and proofs of theorems 12 and 13 are generalized to F_p using the above definition of bit flips and phase flips, where F_2^m is replaced by F_p^m and Hadamard gate is replaced by the Fourier transform W over F_p everywhere.

3.5 Polynomial Quantum Codes

We define polynomial codes over F_p. Set d to be the degree of the polynomials we are going to use, and set m to be the length of the code. $p > m$ will be the number of elements in the field F_p we will be working with. Set $\alpha_1, ...\alpha_m$ to be m distinct non zero elements of the field F_p. Define the linear codes:

$$\begin{aligned} C_1 &= \{(f(\alpha_1)\cdots f(\alpha_m))|f(x) \in F_p(x), deg(f(x)) \leq d\} \subset F_p^m \\ C_2 &= \{(f(\alpha_1)\cdots f(\alpha_m))|f(x) \in F_p(x), deg(f(x)) \leq d, f(0) = 0\} \subset C_1 \end{aligned} \tag{23}$$

$F_p(x)$ is the field of polynomials with coefficients in F_p. We can now define the quantum code:

$$\forall a \in C_1/C_2, \quad |S_a\rangle = \frac{1}{\sqrt{p^d}}\sum_{f\in F_p, deg(f)\leq d, f(0)=a} |f(\alpha_1)\cdots f(\alpha_m)\rangle = \frac{1}{\sqrt{p^d}}\sum_{w\in C_2}|w + \vec{a}\rangle. \tag{24}$$

Clearly, C_2 has p different cosets in C_1, and so $C_1^\perp$ has p disjoint cosets in $C_2^\perp$. Thus the dimension of the code is p, and the code encodes exactly one qupit. Note that this code is a special case of CSS codes, which will be used to prove the following theorem:

Theorem 14 *A polynomial code of degree d with length m over F_p is a $[p,m]$ quantum code which corrects* $\min\{\lfloor\frac{m-d-1}{2}\rfloor, \lfloor\frac{d}{2}\rfloor\}$ *errors.*

Proof: Two different words in C_1 agree on at most d coordinates, and thus C_1 is a linear code of distance $m-d$. It can thus correct and detect $\lfloor(m-d-1)/2\rfloor$ errors. $C_2^\perp$ is a linear code of minimal distance $\geq d+1$. This is true since the projection on any d coordinates of the code C contains all possible strings of length d, and therefore the only vector of length d orthogonal to all the vectors is the 0 vector. Thus, $C_2^\perp$ corrects and detects $\lfloor d/2\rfloor$ errors. Theorem 14 follows from theorem 13. $\square$

4 Computing on States Encoded by CSS codes

CSS codes will be used to perform computations fault tolerantly: The idea is to compute on quantum states encoded by CSS codes. Each gate is replaced by a procedure which imitates the operation of the gate on the encoded states. In order for the computation to be fault tolerant, the procedures have to be designed in such a way so that a small number of errors during the procedure cannot propagate to too many errors at the end of the procedure, before error correction can be applied. The basic definitions are given next. We then show how to perform fault tolerant operations on states encoded by CSS codes. We follow Shor's constructions very closely, with some additional tricks and modifications mainly in the construction of the ancilla state required for the Toffoli gate, and in the decoding and error correction procedures. These modifications are done in order to avoid measurements and classical operations during the computation. In fact, this is the only difference between the results presented in this section, and the results derived by Shor in [].

4.1 Fault Tolerant Procedures on Encoded States- General Definitions

Say we have a unitary gate g which was applied on the state $|\alpha\rangle$ in the original circuit. We now want to apply a sequence of gates, or a "procedure", $P(g)$, on the state encoding $|\alpha\rangle$, such that $P(g)$ will take the encoded $|\alpha\rangle$ to the state encoding $g|\alpha\rangle$.

Definition 15 *A sequence of gates $P(g)$ is said to encode a gate g for the code C if for any superposition $|\alpha\rangle$, $P(g)|S_{|\alpha\rangle}\rangle = |S_{g|\alpha\rangle}\rangle$.*

We will want $P(g)$ to be such that an error occurring at one gate or qubit during the procedure will not affect too many qubits at the end of the procedure, so that the error can be corrected. A location $(q_1, ..q_l, t)$ effects a qubit q' at time $t' > t$ if there is a path in the circuit from $(q_1, ..q_l, t)$ to (q', t').

Definition 16 *The "spread" of a procedure is the maximal number of qubits in one block in the output of the procedure, which are effected by one location in this procedure.*

If we use only procedures with small spread, the error corrections will still be able to correct the damage using the undamaged qubits, provided that not too many errors happened during the procedure.

The notion of reduced density matrices is useful here: At the end of a fault tolerant procedure which operates on a state $|\alpha\rangle$, the result will be "correct" on all qubits except those effected by the fault. This means that the reduced density matrix on all the qubits except those which where effected, is the same as it would have been if no fault occurred at all.

4.2 Questions of Ancilla Qubits

In some of the procedures, we will use ancilla qubits, as extra working space. At the end of the procedure these qubits will be discarded, in order to get exactly the state we need. As was explained in subsection 2.3, we will always discard qubits which are in tensor product with the rest of the system, so the operation of discarding means simply erasing their state, and the resulting state is a pure state. This is necessary if we want to operate unitary operations on the encoded states. We will describe a procedure by specifying what it does to basic states of the code. It is easy to see that if for any input basis state, the state of the ancilla qubits at the end of the procedure are in a tensor product with the rest of the qubits, and their state does not depend on the input basis state, i.e.

$$|S_a\rangle \longmapsto |S_{g(a)}\rangle \otimes |A\rangle \tag{25}$$

where A is independent of a, then for any input superposition for the procedure the ancilla qubits will be in tensor product with the rest of the qubits, and thus they can be discarded simply by erasing them.

This requirement of independence of the ancilla qubits on the basis state can be released in two cases. An encoding procedure, is the procedure which takes a block of input bits, 0^m to the state encoding 0, $|S_0\rangle$, and likewise for 1. For the encoding procedure, we know that the procedure always gets as an input a basis state. Thus, we release the above requirement, and demand only that the ancilla state is in a tensor product with the computational qubits when the input to the procedure is a basis state. (The ancilla might therefore depend on the input, and indeed it will, but we are allowed do discard it anyway.) The requirement of tensor product with the ancilla can also be released in the decoding procedure, since the state will not be used any more, and we should only check that it gives the correct answer when measured.

It suffices to check the independence of the ancilla in the case of no fault in the procedure. The fact that the procedures are fault tolerant guarantees that even if a fault occurred, the reduced density matrix of the non-effected qubits is just like in the case in which no fault occurred at all.

4.3 Some Restrictions on the CSS Codes

In the following, we will put some restrictions on the CSS codes which we will use. This is done in order to be able to apply several gates fault tolerantly bitwise, as will be seen shortly. We require that C_1 is a punctured doubly even self dual code, and that $C_2 = C_1^\perp$. A punctured code is a code which is obtained by deleting one coordinate from a code C', and a punctured self dual code means that we require in addition that C' is self dual, i.e. $C' = C'^\perp$. We also require that C' is a doubly even code, i.e. the weight of each word in the code is divisible by 4. To see that $C_2 = C_1^\perp \subset C_1$, as in the definitions of CSS codes, observe that if $v \perp C_1$, then $v0 \perp C'$ so $v0 \in C'^\perp = C'$, so $v \in C_1$. We will denote $C_1 = C$, and $C_2 = C^\perp$.

We now claim that there are only two cosets of $C^\perp$ in C. If the length of C is m, then $dim(C'^\perp) = dim(C') = (m+1)/2$. Hence $dim(C) = (m+1)/2$ as well, since $|C| = |C'|$, because no two words in C' are mapped to the same word in C by the punctuation. Hence, $dim(C^\perp) = m - (m+1)/2 = (m-1)/2$ and so $dim(C) - dim(C^\perp) = 1$. Observe that C includes the all one vector: $\vec{1} \in C$. This is true since $1^{m+1} \in C'^\perp$, because C' is even, and since C' is self dual, $1^{m+1} \in C'$. Hence $1^m \in C$. Observe also that the length m must be odd due to the above considerations. This implies that $\vec{1} \notin C^\perp$. The two code words in our quantum code can thus be written as

$$\begin{aligned} |S_{\vec{0}}\rangle &= \sum_{w \in C^\perp} |w\rangle \\ |S_{\vec{1}}\rangle &= \sum_{w \in C^\perp} |w + \vec{1}\rangle \end{aligned} \tag{26}$$

$|S_{\vec{0}}\rangle$,$|S_{\vec{1}}\rangle$, can be thought of as encoding $|0\rangle$, $|1\rangle$ respectively. We will make use of the fact that $\vec{a}\cdot\vec{b} \bmod 2 =$ ab, for $a, b \in 0, 1$, and that $\vec{a} + \vec{b} = \overrightarrow{a+b}$. This fact allows us to shift easily between operations on vectors, and operations on the bits they represent. We will therefore usually omit the vectors in the notations of $|S_{\vec{0}}\rangle$ and $|S_{\vec{1}}\rangle$, unless there is ambiguity.

4.4 The Set of Gates for CSS codes

We work with the following set of gates, which we denote by $\mathcal{G}_1$:

1. Not: $|a\rangle \longmapsto |1-a\rangle$,
2. Controlled not: $|a, b\rangle \longmapsto |a, a+b\rangle$,
3. Phase rotation: $|a\rangle \longmapsto i^a|a\rangle$,
4. Controlled Phase rotation: $|a\rangle|b\rangle \longmapsto (-1)^{ab}|a\rangle|b\rangle$,
5. Hadamard: $|a\rangle \longmapsto \frac{1}{\sqrt{2}}\sum_b(-1)^{ab}|b\rangle$,
6. Toffoli gate: $|a, b, c\rangle \longmapsto |a, b, c+ab\rangle$,
7. Swap $|a\rangle|b\rangle \longmapsto |b\rangle|a\rangle$,
8. Adding a qubit in the state $|0\rangle$,
9. Discarding a qubit.

where all the addition and multiplication are in F_2(i.e. mod 2). We will show later on that this set of gates is universal. We remark here, that $\mathcal{G}_1$ is by no means a minimal universal set of gates, but the fault tolerant procedures become simpler and shorter if we have a larger repertoire of fault tolerant gates that we can use. The following theorem shows how to perform gates from $\mathcal{G}_1$ on encoded states fault tolerantly.

Theorem 15 *There exists fault tolerant procedures which simulate the operations of all the gates from $\mathcal{G}_1$, on states encoded by punctured self dual doubly even CSS codes such that one error in a qubit or a gate effects at most four qubits in each block at the end of the procedure. There exist such procedures also for encoding, decoding and error correction. Moreover, all these procedures use only gates from $\mathcal{G}_1$, and in particular do not use measurements.*

To prove the theorem, we first show this for the gates which can be applied bit-wise, then for the encoding, decoding and error correction procedures, and then for the rest of the gates.

4.5 Bitwise Fault Tolerant Procedures

Let g be a gate on k qubits. A bitwise procedure of this gate is defined by labeling the qubits in each one of k blocks from 1 to m, and then applying the gate m times, each time on all qubits with the same label. Obviously, an error in this procedure can effect only one qubit. All the gates in G_1 can be applied bitwise, except the Toffoli gate and the gate which adds a blank qubit. These are the NOT, Controlled not, Phase rotation, Controlled Phase rotation, Hadamard, Swap and Discarding a qubit. This is trivial for the SWAP gate, and the gate which discards a qubit. Simple calculations show this also for the other gates. In the following we omit overall normalization factors, since all vectors are known to be unit vectors. We also set $a, b \in C/C^{\perp}$.

$$|S_a\rangle = \sum_{w \in C^{\perp}} |a_1 + w_1\rangle \otimes ...|a_m + w_m\rangle \longmapsto \sum_{w \in C^{\perp}} |a_1 + 1 + w_1\rangle \otimes ...|a_m + 1 + w_m\rangle = |S_{a+\vec{1}}\rangle \tag{27}$$

For CNOT,

$$\begin{aligned} |S_a\rangle|S_b\rangle \quad = \quad & \sum_{w \in C^{\perp}} |a_1 + w_1\rangle \otimes ...|a_m + w_m\rangle \sum_{w' \in C^{\perp}} |b_1 + w'_1\rangle \otimes ...|b_m + w'_m\rangle \longmapsto \\ & \sum_{w \in C^{\perp}} |a_1 + w_1\rangle \otimes ...|a_m + w_m\rangle \sum_{w' \in C^{\perp}} |a_1 + b_1 + w_1 + w'_1\rangle \otimes ...|a_m + b_m + w_m + w'_m\rangle \\ & = |S_a\rangle|S_{a+b}\rangle \end{aligned} \tag{28}$$

where the last equality follows from the fact that $C^{\perp}$ is a linear subspace, and therefore summing over $w + w'$ for a fixed w in the code is equivalent to summing over w'. For the Phase gate, apply the gate $|a\rangle \longmapsto i^a|a\rangle$ three times on each coordinate. This gives

$$|S_a\rangle = \sum_{w \in C^{\perp}} |a_1 + w_1\rangle \otimes ...|a_m + w_m\rangle \longmapsto \sum_{w \in C^{\perp}} i^{3(\sum_k a_k + w_k)}|a_1 + w_1\rangle \otimes ...|a_m + w_m\rangle. \tag{29}$$

This is the desired result, because C is a punctured doubly even self dual code, and it is easy to check that all words in $C^{\perp}$ have weight which is divisible by 4, and all words in C but not in $C^{\perp}$ have weight which is 3 mod 4.

For the encoded controlled phase gate,

$$|S_{\vec{a}}\rangle|S_{\vec{b}}\rangle = \sum_{w, w' \in C^{\perp}} |\vec{a} + w\rangle|\vec{b} + w'\rangle \longmapsto \sum_{w, w' \in C^{\perp}} (-1)^{(\vec{a}+w)\cdot(\vec{b}+w')}|\vec{a} + w\rangle|\vec{b} + w'\rangle \tag{30}$$

Now, $(\vec{a} + w) \cdot (\vec{b} + w') = \vec{a} \cdot \vec{b}$ mod 2. This is true since $\vec{a} \in C$ and $w' \in C^{\perp}$ so $\vec{a} \cdot w' = 0$ mod 2, and likewise $w \cdot \vec{b} = w \cdot w' = 0$ mod 2. Moreover, $\vec{a} \cdot \vec{b}$ mod 2 is equal to ab, and so the final state is indeed the desired state. Finally, for the Hadamard gate,

$$\begin{aligned} |S_a\rangle \quad &= \quad \sum_{w \in C^{\perp}} |a_1 + w_1\rangle \otimes ...|a_m + w_m\rangle \longmapsto \sum_{x \in F_2^m} \sum_{w \in C^{\perp}} (-1)^{(a+w)\cdot x}|x\rangle = \\ &= \quad \sum_{x \in C} (-1)^{a \cdot x}|x\rangle = \sum_{b \in C/C^{\perp}} \sum_{w \in C^{\perp}} (-1)^{a\cdot(b+w)}|b + w\rangle = \sum_b (-1)^{a \cdot b}|S_b\rangle. \end{aligned} \tag{31}$$

4.6 Fault Tolerant Error Correction Procedure for CSS Codes

So far, we have described how to apply those procedures which can be obtained by bitwise operations. Before we continue to the Toffoli gate and the gate which adds a blank qubit, we first show how to apply encoding, decoding and error correction fault tolerantly. These procedures will be used later in the remaining of the gates.

We now construct the error correction procedure. It is composed from detecting and correcting bit-flips, using classical error correction techniques for the code C, rotating by Hadamard gate to the $C-$basis, and correcting bit-flips using again classical error correction techniques. Finally we rotate back by applying Hadamard bit-wise.

We now describe how to correct bit-flips. For this, we recall that for linear classical codes over F_2, one can define the *parity check matrix* H of the code C. The kernel of H is exactly the words in C. If an error occurred in a word w, the new word can be written $w+e$, where e is called the error vector. $H(w+e)=He=s$ is called the syndrome of the error, and given the syndrome s, one can find the error vector of minimal weight which gives s. The error correction is therefore done by first finding the syndrome, and then deducing the error vector from it.

We now want to compute the $j'th$ bit of the syndrome fault tolerantly. This is simply the inner product of the j'th row of H with the word. If out computation was noiseless, we could do the following: we apply CNOT from the block we are correcting to one blank qubit, only on the coordinates which are 1 in the j'th row of the parity check matrix of the code. This will compute the inner product of the word we are correcting with the j'th row of the parity check matrix, and the extra blank qubit will thus contain the $j'th$ bit of the syndrome. However, this is not a fault tolerant procedure, since an error in one of the CNOT gates causes an error in the extra qubit, which propagates through the other CNOT gates to other qubits in the block.

To avoid this problem, we will use an ancillary state on l qubits:

$$\frac{1}{2^{\frac{(l-1)}{2}}}\sum_{b;b\cdot\vec{1}}|b\rangle \tag{32}$$

where l is the number of $1'$s in the in the $j'th$ raw of H, the parity check matrix of $C^{\perp}$. This state is actually the superposition of all states with an even number of 1's. It can be easily constructed by starting with $|0^l\rangle$, applying Hadamard on the first qubit and then CNOT from this qubit to all the other $l-1$ qubits, to get the "cat state" on l qubits,

$$|cat_l\rangle=\frac{1}{\sqrt{2}}(|0^l\rangle+|1^l\rangle). \tag{33}$$

and then rotating each qubit of the cat state to get the desired state.

We would now like to apply CNOT bitwise from the block we are initializing to the cat state only on the coordinates which are 1 in the j'th row of the parity check matrix. Then we apply a classical computation, using only gates from $\mathcal{G}_1$, on the ancilla state, to find out the parity of the strings in the ancilla state, and write this parity on another qubit, initialized in the state $|0\rangle$. The resulting state will be the j'th syndrome bit.

The only problem in this scheme is that the generation of the cat state is not done fault tolerantly, and thus one error can cause the entire cat state to be ruined; The CNOT gates will then propagate this error to many qubits in the block. The solution to this problem is to verify that the cat state is indeed a superposition of the two states $|0^l\rangle$ and $|1^l\rangle$, before continuing. The relative phase between the two states can still be mistaken, but this will not cause any error to propagate to the state by the CNOT gates, but only cause the syndrome to be mistaken. To see this, suppose we have instead of the cat state, the state $c_0|0\rangle+c_1|1\rangle$. This state transforms after the Hadamard gates to the state:

$$c_0\sum_i|i\rangle+c_1\sum_i(-1)^{i\cdot\vec{1}}|i\rangle \tag{34}$$

Now, consider applying CNOT's on all the 1 coordinates in the parity check matrix from a correct encoded state to $\sum_i |i\rangle$. This is simply the identity operator:

$$(c_0|S_0\rangle + c_1|S_1\rangle)\sum_i |i\rangle \longmapsto (c_0|S_0\rangle + c_1|S_1\rangle)\sum_i |i\rangle \tag{35}$$

and the same for the other term $\sum_i (-1)^{i\cdot\vec{1}}|i\rangle$. This means that in spite of the fact that the cat state is not correct, this does not cause any error in the original state.

This verification was done by Shor[191] using measurements and assuming noiseless classical computation, by measuring the XOR of pairs of the qubits. If we want to avoid measurements, we proceed as follows. We now want to compute whether the bits in the cat state are all equal. For this, note that checking whether two bits are equal, and writing the result on an extra blank qubit, can be easily done by a small circuit which uses Toffoli and NOT gates. We denote this circuit by S, and also denote the qubits in the cat state by $1,...l$. We add $l-1$ extra blank qubits, and apply the circuit S first from each even pair of qubits (e.g. the pair of qubits $(1,2),(3,4)...$), to one of the blank qubits; Then apply S from each odd pair of qubits (e.g. the pairs $(2,3),(4,5)...$) to one of the remaining blank qubits. We get $l-1$ qubits which are all 1, if no error occurred, indicating that all the qubits are equal. We then apply a classical circuit on all these qubits, which checks whether they are all 1, and write the result on an extra blank qubit, which is our check bit, and indicates that all the bits in the cat state are equal. We now want to use the cat state, but condition all the operations on the fact that the check bit is indeed 1. However, if we do this, an error in the check bit, might propagate to all qubits in the state we are trying to correct. Hence, to keep the procedure fault tolerant, we construct m different check bits, one for each qubit in the state we are correcting. This is done using $m(l-1)$ blank qubits, and applying all the operations above, where each operation is repeated m times. to m different target qubits. Thus, we can verify that the cat state is of the form $c_0|0\rangle + c_1|1\rangle$, fault tolerantly. We can now condition all the operations done in the syndrome measurement involving the i'th qubit, on the i'th check bit.

From the syndrome s we can compute the error vector e which is the vector of minimal weight which gives $He = s$. This can be done by classical operations, and need not be done fault tolerantly. Applying a CNOT from the result bit e_i to the i'th bit will correct a bit flip in the i'th bit. Now, we do not correct all qubits according to the error vector computed from one copy of the syndrome, since an error during the computation of the error vector might result in the entire error vector being mistaken, and the procedure will not be fault tolerant. Instead, we compute the syndrome independently m times, and from the i'th copy of the syndrome we compute e_i, from which we apply a CNOT to the i'th bit of the state which we are correcting. The fact that for each bit we compute the syndrome independently ensures us that the procedure is fault tolerant.

To apply error correction, we first apply error corrections of bit flips, according to the code C. We then apply a Hadamard gate to transform to the C-basis, and correct bit flips again, according to the code C. Then, we apply a Hadamard gate again, to get back to the S-basis.

In all the procedures we have described so far, one fault causes at most one error in each block. Unfortunately, the propagation of errors in the error correction procedure is much worth. One fault in one of the circuits S, checking whether two qubits in the cat state are the same, might cause four qubits in this cat state to be contaminated, where their check bits might still indicate a green light for following procedures. Hence, during the error corrections, each one of the four errors can propagate to the one qubit which is conditioned upon it. Apart from this bottle neck, all other errors have exactly one spread.

4.7 Fault Tolerant Encoding Procedure for CSS codes

An encoding procedure takes

$$\begin{aligned} |0^m\rangle &\longmapsto |0^m\rangle|S_0\rangle \\ |1^m\rangle &\longmapsto |1^m\rangle|S_1\rangle \end{aligned} \tag{36}$$

To construct this procedure fault tolerantly, it is enough to generate a state $|S_0\rangle$ fault tolerantly. This suffices since given a string of bits, $|0^m\rangle$ or $|1^m\rangle$, we first generate $|S_0\rangle$ and then apply a controlled not bitwise from the given input string to the encoded state $|S_0\rangle$.

To generate the state $|S_0\rangle$, start with m blank qubits, $|0^m\rangle$, then apply an error correction procedure with respect to a code which consists of one word: $|S_0\rangle$ alone. This is done by first applying Hadamard gates bitwise on $|0^m\rangle$, and then correcting bit flips according to the code $C^\perp$. To see that we indeed get the state $|S_0\rangle$, note that after the first rotations we have $\sum_{i=0}^{2^m-1}|i\rangle$. The corrections will then take this state to a uniform distribution over all the basic states in $C^\perp$, due to the linearity of the code.

4.8 Error Correction which Projects any Word into the Code

We would like to insert here one important modification, which is not necessary for the fault tolerant error correction, but will become crucial when applying the error corrections in the recursive scheme. This is the requirement that the error correction takes any word to some word in the code, regardless of the number of faults. Roughly, this is done by checking whether too many errors occurred, and if so replacing the entire block by another block which is initialized in the state $|S_0\rangle$. However, we should be careful to keep the procedure fault tolerant. We do this in the following way: Before starting the correction procedure, we generate another state $|S_0\rangle$ on ancilla qubits, as in the encoding procedure. When computing from the i'th copy of a syndrome whether the i'th qubit is wrong, also compute whether the number of faults according to the syndrome is smaller than d, and write the answer on another qubit. The CNOT which checks if the i'th bit is wrong and if so applies NOT on the $i'th$ qubit, is replaced by a Toffoli gate which also checks if the number of faults is smaller than d. We then swap the i'th qubit with the i'th qubit of the state $|S_0\rangle$, conditioned that the number of faults is indeed larger that d. To achieve a swap between the second and third qubit conditioned on the first qubit, apply three Toffoli gates one after the other, where the target of the first gate is the third qubit, the target of the second gate is the second qubit and the target of the third gate is again the third qubit, i.e.

$$Controlled(a)SWAP(b,c) = T(a,b,c)T(a,c,b)T(a,b,c) \tag{37}$$

which is easy to check.

To see that this procedure indeed takes any word to some word in the code, observe that this is true if no error occurs in the procedure itself, and since the procedure is fault tolerant, the final state will differ from a word in the code only in the qubits effected by an error.

4.9 A Fault Tolerant Decoding Procedure for CSS codes

A decoding procedure applies

$$|S_a\rangle \longmapsto |A_a\rangle|\vec{a}\rangle \tag{38}$$

where the state $|A_a\rangle$ is an ancillary state which depends on a. (We can discard this state at the end of the procedure; we will see that whenever the decoding procedure is applied, the state encodes a well defined logical bit a.) To apply this transformation, we compute a independently m times from the state $|S_a\rangle$. To do this, we add m^2 blank qubits, and copy each qubit from $|S_a\rangle$ m times to m different blank qubits, using m CNOT gates. We get m "copies" of $|S_a\rangle$. (These of course are not really copies of $|S_a\rangle$, since they are entangled. However, each word in the classical code is copied m times.) On each copy of the word we apply the quantum analog of the classical computation that computes what is the logical bit that the word encodes. The answer, which is a if no error occurred, is written on another blank qubit. For this computation we use Toffoli, CNOT and NOT gates. We might need some extra blank qubits as working space. The computation of a is done is the shortest way possible, regardless of whether it is fault tolerant; An error in this computation can effect only the one copy of a which it computes. A fault in the first stage of copying the qubits m times can only effect one qubit in each of the copies, and if the number of errors in S_a plus number of faults in the first stage is smaller than the number of errors correctable by the code, these faults have no effect. One fault in the second stage of the procedure, during the computation of one of the a's, can effect only the correctness of that a.

4.10 Toffoli Gate

To apply the Toffoli gate, we roughly follow Shor's scheme, where we construct an ancillary state and use it to obtain the Toffoli gate. The main difference from Shor's scheme is in the construction of the ancillary state, which is not completely straightforward if one wants to avoid using measurements. The following construction involves many details, and has no underlying structure, as far as we can see. As we will see in the next section, the Toffoli gate on polynomial codes can be applied in a much easier and shorter way. However, for completeness, we also give the details of the Toffoli procedure in the case of CSS codes.

Construction of the Ancilla state $|A\rangle$

We would now like to show how to construct the state

$$|A\rangle = \frac{1}{2}|S_0S_0S_0\rangle + |S_0S_1S_0\rangle + |S_1S_0S_0\rangle + |S_1S_1S_1\rangle. \tag{39}$$

This will be done using the help of the encoded cat states. We will need the definition of the state

$$|B\rangle = \frac{1}{2}|S_0S_0S_1\rangle + |S_0S_1S_1\rangle + |S_1S_0S_1\rangle + |S_1S_1S_0\rangle. \tag{40}$$

which is easily convertible to $|A\rangle$ by applying an encoded NOT on the third block.

The idea is that

$$\frac{1}{\sqrt{2}}(|A\rangle + |B\rangle) \tag{41}$$

is actually very easy to construct, because it is equal to

$$\frac{1}{2\sqrt{2}}(|S_0\rangle + |S_1\rangle)(|S_0\rangle + |S_1\rangle)(|S_0\rangle + |S_1\rangle) \tag{42}$$

which can be constructed easily from 3 block of m blank qubits by applying an encoding procedure and then an encoded Hadamard gate on each block. In order to convert this state to $|A\rangle$, we use encoded cat states. An encoded cat state,

$$|S_{cat}\rangle = \frac{1}{\sqrt{2}}(|S_0\rangle^m + |S_1\rangle^m) \tag{43}$$

is also easy to construct fault tolerantly, in the following way: Generate m $|S_0\rangle$ states, $|S_0\rangle^m$, fault tolerantly, using the encoding procedure on m blocks each containing m blank qubits. Then, apply an encoded Hadamard gate on the first block, and then copy this block to all the other blocks by applying encoded CNOT from the first block to all other blocks. This results in an encoded cat state. We now apply fault tolerant error corrections on each block.

Now, consider the transformation

$$\begin{aligned} |S_0\rangle^m|A\rangle &\longmapsto |S_0\rangle^m|A\rangle \\ |S_1\rangle^m|A\rangle &\longmapsto |S_1\rangle^m|A\rangle \\ |S_0\rangle^m|B\rangle &\longmapsto |S_0\rangle^m|B\rangle \\ |S_1\rangle^m|B\rangle &\longmapsto -|S_1\rangle^m|B\rangle \end{aligned} \tag{44}$$

If we can apply this transformation fault tolerantly, we could start with the state

$$\frac{1}{2}(|S_0\rangle^m + |S_1\rangle^m)(|A\rangle + |B\rangle) \tag{45}$$

and apply the transformation 44 to get

$$\frac{1}{2}(|S_0\rangle^m + |S_1\rangle^m)|A\rangle + \frac{1}{2}(|S_0\rangle^m - |S_1\rangle^m)|B\rangle \tag{46}$$

We might then be able to compute fault tolerantly in which of the cat states $\frac{1}{2}(|S_0\rangle^m \pm |S_1\rangle^m)$ the ancilla is, and apply a NOT on the third block of $|B\rangle$, conditioned that the result is the second state $\frac{1}{2}(|S_0\rangle^m - |S_1\rangle^m)$. Let us refer to this process as a "measurement" of the cat state, though no measurement will be involved.

The transformation 44 can be done by applying

$$|S_a\rangle|b\rangle|c\rangle|d\rangle \longmapsto (-1)^{a(bc+d)}|S_a\rangle|b\rangle|c\rangle|d\rangle \tag{47}$$

on the i'th block in the encoded cat state and the i'th bit in each of the three blocks of $|A\rangle + |B\rangle$. Note that transformation 47 need not be fault tolerant, and it might ruin the entire block $|S_a\rangle$. Applying transformation 47 for $1 \le i \le m$ it is easy to check that we get

$$|S_a\rangle^m|S_b\rangle|S_c\rangle|S_d\rangle \longmapsto (-1)^{a(bc+d)}|S_a\rangle^m|S_b\rangle|S_c\rangle|S_d\rangle \tag{48}$$

which exactly achieves transformation 44.

We now run into the problem of how to measure the encoded cat state. To solve this problem, we use three encoded cat states, instead of one, and repeat everything that we did on the first encoded cat state also for these states. The resulting state is

$$\frac{1}{2}(|S_0\rangle^m + |S_1\rangle^m)^3|A\rangle + \frac{1}{2}(|S_0\rangle^m - |S_1\rangle^m)^3|B\rangle \tag{49}$$

Now it becomes possible to measure the triple cat state, using majority. This can be done in several ways. One way is to apply encoded Hadamard gates on all blocks of the encoded cat states, which takes the encoded states into superpositions of encoded states, with certain parity:

$$\frac{1}{\sqrt{2}}(|S_0\rangle^m + |S_1\rangle^m) \longmapsto \frac{1}{\sqrt{2^{m-1}}}\sum_{i=0,i\cdot\vec{1}=0}^{2^m}|S_{i_1}\rangle|S_{i_2}\rangle...|S_{i_m}\rangle, \tag{50}$$
$$\frac{1}{\sqrt{2}}(|S_0\rangle^m - |S_1\rangle^m) \longmapsto \frac{1}{\sqrt{2^{m-1}}}\sum_{i=0,i\cdot\vec{1}=1}^{2^m}|S_{i_1}\rangle|S_{i_2}\rangle...|S_{i_m}\rangle$$

To compute the parity fault tolerantly, we compute from each block the bit it represents, decoding it fault tolerantly. Then independently compute the parity of the m i'th bits in the m blocks. The parity can be computed using Toffoli, CNOT and NOT gates. We now have m parity bits, for each encoded cat state. We compare them bitwise by applying on the three i'th parity bits and an ancilla bit a majority vote

$$|a,b,c,d\rangle \longmapsto |a,b,c,d+maj(a,b,c)\rangle \tag{51}$$

which can be constructed from Toffoli, CNOT and NOT gates, since they are universal for classical computations. Now, apply CNOT bitwise from the resulting majority votes of the parity bits to the qubits in the third block of $|A\rangle$.

It is left to see that one error in this procedure effects only one qubit in $|A\rangle$. We will consider errors in different stages of the procedure. The construction of an encodes cat state id fault tolerant since it is composed of the fault tolerant encoding procedure, and bitwise CNOT's. The construction of the state $\frac{1}{\sqrt{2}}(|A\rangle + |B\rangle)$ is fault tolerant because we only used the fault tolerant encoding procedure and the fault tolerant encoded Hadamard gate. A problem arises in transformation 44. During this transformation, an error can cause the entire block $|S_a\rangle$ to be ruined. Hence, an error during this transformation can cause a whole block in one of the cat states in the state (49) to be effected, together with one qubit in each one of the last 3 blocks. However, one such block can ruin the parity bits of only one encoded cat state, and as long as the other two are still fine, the majority vote will still give the correct parity. An error in the parity computations or in the majority vote cannot effect more than one qubit in $|A\rangle$, since they are done bitwise. This concludes the fault tolerant construction of the state $|A\rangle$.

Note, that the above procedure is fault tolerant if one error occurred, but if we are unlucky and two errors occurred during transformation (49) when applied on two different encoded states, this can effect the parity bits of two encoded cat states, and thus the majority vote will fail here. In order to tolerate more than one error in a procedure, we will need to use more encoded cat states. Using $2k+1$ encoded cat states, we can still say that one error effects only four qubits in each block, as long as the number of errors is less than k.

Construction of Toffoli Gate Given $|A\rangle$

The construction of the Toffoli gate given $|A\rangle$ follows Shor's scheme almost exactly, with minor changes due to the fact that measurements are replaced by CNOT gates to additional blank qubit. Also, classical conditioning on the results of the measurements are replaced by classical unitary gates (i.e. permutation matrices) on the computation qubits and extra blank qubits. Here is a short description of how this is done: We try to generate the transformation

$$|\alpha\rangle = |S_a\rangle|S_b\rangle|S_c\rangle \longmapsto |S_a\rangle|S_b\rangle|S_{c+ab}\rangle \tag{52}$$

For this, we will first generate "half" a Toffoli gate:

$$|S_a\rangle|S_b\rangle|S_0\rangle \longmapsto |S_a S_b S_{ab}\rangle \tag{53}$$

A Toffoli gate can be generated from transformation (53) in the following way. We start with the three blocks on which we want to apply Toffoli, $|S_a\rangle|S_b\rangle|S_c\rangle$. Then we generate $|S_0\rangle$ on an extra block, using the encoding procedure. We then apply transformation (53) on the first two blocks and the newly generated $|S_0\rangle$. Then, we apply an encoded CNOT from our original third block $|S_c\rangle$ to the new block. We finally apply an encoded Hadamard on the original third block. This gives the overall transformation:

$$|S_a\rangle|S_b\rangle|S_c\rangle|S_0\rangle \longmapsto \frac{1}{\sqrt{2}}|S_a S_b S_{c+ab}\rangle(|S_0\rangle + (-1)^c|S_1\rangle) \tag{54}$$

Note that if the fourth block was not there, we are done, because the operation on the first three blocks is exactly Toffoli. We next decode the fourth block, and apply some operations conditioned on the qubits containing the results of the decoding. Note that if the decoding is $\vec{0}$, there is no phase to correct, since the gate which has been performed is exactly the Toffoli gate. Hence, we would like to apply the operation

$$|S_a\rangle|S_b\rangle|S_c\rangle \longmapsto (-1)^{ab+c}|S_a\rangle|S_b\rangle|S_c\rangle \tag{55}$$

on the first three blocks, conditioned that the decoded qubits are 1. This can be applied bitwise, in the following way. First, apply bitwise a controlled phase shift on $|S_c\rangle$ and the decoded qubits. This will give the factor $(-1)^c$. To apply $|S_a\rangle|S_b\rangle \longmapsto (-1)^{ab}|S_a\rangle|S_b\rangle$ conditioned on the decoded qubits, apply bitwise the operation: $|a\rangle|b\rangle|d\rangle \longmapsto (-1)^{abd}|a\rangle|b\rangle|d\rangle$ where $|d\rangle$ is a decoded qubit. This operation can be achieved by adding a blank qubit, $|a\rangle|b\rangle|0\rangle|d\rangle$ applying a Toffoli gate on the first three qubits, followed by a controlled shift on the last two qubits, and then by a Toffoli gate again on the first three qubits.

It is left to show how to construct transformation (53). This is done by generating the ancilla state $|A\rangle$ as before, where one of the blocks of $|A\rangle$ is the third block. and apply encoded CNOT from the first block of $|A\rangle$ to the first block of $|\alpha\rangle$ and from the second block of $|A\rangle$ to the second block of $|\alpha\rangle$. This achieves the transformation:

$$|S_a\rangle|S_b\rangle|A\rangle \longmapsto \frac{1}{2}(|S_0 S_0 S_a S_b S_{ab}\rangle + |S_0 S_1 S_a S_{\bar{b}} S_{a\bar{b}}\rangle + |S_1 S_0 S_{\bar{a}} S_b S_{\bar{a}b}\rangle + |S_1 S_1 S_{\bar{a}} S_{\bar{b}} S_{\bar{a}\bar{b}}\rangle) \tag{56}$$

We note that the last three blocks have strong connection to the Toffoli gate. More precisely, we note that projecting the above states on the subspace where the first two blocks is $|S_0\rangle|S_0\rangle$, the gate which was achieved is exactly the Toffoli gate. The projection on the subspace in which the two blocks are $|S_0\rangle|S_1\rangle$ is not a Toffoli gate, but can be converted to a Toffoli gate by applying a NOT on the second block and a CNOT from the first block to the second block. Projecting to the other two possible subspaces, we again get a Toffoli gate by simple corrections: for $|S_1\rangle|S_0\rangle$ we should apply NOT on first block and CNOT from second to third block, while for $|S_1\rangle|S_1\rangle$ the gate can be corrected to Toffoli by applying a CNOT from the second block to the third, then a CNOT from the first to the third and then a NOT on the last black.

We now have to make these adjustments, conditioned on the state of the first two blocks, without measurements. This is done by decoding the first two blocks, and applying the adjustments conditioned on the state of the decoded qubits, bitwise, meaning that we apply the following operation on five qubits:

$$\begin{aligned}
|0,0\rangle|a,b,c\rangle &\longmapsto |0,0\rangle|a,b,c\rangle \\
|0,1\rangle|a,b,c\rangle &\longmapsto |0,1\rangle NOT(2)CNOT(1,2)|a,b,c\rangle \\
|1,0\rangle|a,b,c\rangle &\longmapsto |1,0\rangle NOT(1)CNOT(2,3)|a,b,c\rangle \\
|1,1\rangle|a,b,c\rangle &\longmapsto |1,1\rangle NOT(3)CNOT(1,3)CNOT(2,3)|a,b,c\rangle
\end{aligned} \tag{57}$$

This operation is reversible, and is classical. Therefore it can be constructed out of the classical gates which are in the set $\mathcal{G}_1$. It is easy to check that indeed it achieves the correct transformation.

5 Computing on States Encoded by Polynomial Codes

In the last section it was shown how to implement fault tolerantly the set of gates $\mathcal{G}_1$. It turned out that all the gates were very easy to implement, except for the last gate: The Toffoli gate. This turned out to involve many tricks, and seems not to have an underlying theoretical structure. In this section we show that when transforming to qupits, and using the polynomial codes, one can use the algebraic properties of these codes to implement the gates fault tolerantly in a manner which seems logical and easy to explain. The error correction, decoding and encoding procedures will be similar to those used for the CSS codes.

We define a set of gates $\mathcal{G}_2$, which will be shown to be universal for qupits in the next section. We show how to apply all gates from $\mathcal{G}_2$ fault tolerantly on states encoded by polynomial codes. It turns out that all these procedures can be applied using the same basic underlying idea. This idea is that when applying the gates pitwise, we always get the desired result, except for one problem: Instead of getting the final state as a superposition of polynomials with degree d, we sometimes end up with the correct logical pit, encoded with polynomials with degree $2d$ instead of d. To get back into the code which uses polynomials of degree d, one has to apply degree reduction. We use techniques which were introduced by Ben-Or *et al* [46] for fault tolerant classical distributed computation, on a secret which is shared by several parties. In order to be able to use these degree reduction techniques, it is required that after the degree has increased to $2d$, the state is still a polynomial code, so that errors can still be corrected. For this, we will work with codes of length $m = 3d + 1$.

5.1 The Set of Gates for Polynomial Codes

We work with the following set of gates, which we denote by $\mathcal{G}_2$:

1. Generalized NOT: $\forall\ c \in F$, $|a\rangle \longmapsto |a+c\rangle$,
2. Generalized CNOT: $|a,b\rangle \longmapsto |a,a+b\rangle$,
3. Swap $|a\rangle|b\rangle \longmapsto |b\rangle|a\rangle$,
4. Multiplication gate: $0 \neq c \in F$: $|a\rangle \longmapsto |ac\rangle$,
5. generalized Phase Rotation: $\forall c \in F\ |a\rangle \longmapsto w^{ca}|a\rangle$,
6. generalized Hadamard (Fourier Transform) $|a\rangle \longmapsto \frac{1}{\sqrt{p}}\sum_{b\in F} w^{rab}|b\rangle, \forall 0<r<p$.
7. generalized Toffoli: $|a\rangle|b\rangle|c\rangle \longmapsto |a\rangle|b\rangle|c+ab\rangle$,

where all the addition and multiplication are in F_p(i.e. mod p). The fact that this set of gates is universal will be shown later, in section 6. The following theorem shows how to perform gates from $\mathcal{G}_2$ on encoded states fault tolerantly. Like $\mathcal{G}_1$, $\mathcal{G}_2$ is by no means a minimal universal set of gates, and in fact we make it as large as possible so that the fault tolerant procedures become more efficient.

Theorem 16 *There exists fault tolerant procedures which simulate the operations of all the gates from $\mathcal{G}_2$, on states encoded by polynomial codes, such that one error in a qubit or a gate effects only one qubit in each block at the end of the procedure. There exist such procedures also for encoding, decoding and error correction. Moreover, all these procedures use only gates from $\mathcal{G}_2$, and in particular do not use measurements.*

Proof: It turns out that all the gates can be applied pit-wise, and then applying degree reduction. The proof of this theorem, like that of theorem 15 is done by first showing how to perform the encoded gates in the cases in which pit-wise applications of the gate will do (without the need in reducing the degree of the polynomials.) Then we show how to apply the encoding, decoding and correction procedures. Finally, we show a procedure to reduce the degree of the polynomials, which enables us to construct the Fourier transform and the Toffoli gate. Both of these gates can be achieved by applying them pit-wise, and then applying degree reductions.

Pit-wise Fault Tolerant Procedures

Applying a generalized NOT gate on each one of the qupits in a block gives an generalized NOT on the entire block, as can be easily checked. Applying a generalized CNOT from the i'th qupit in the first block to the i'th of the second block, for all $1 \leq i \leq ms$, gives an encoded generalized CNOT from the first block to the second block. In the same way, applying the SWAP gate and the multiplication gate pit-wise, achieves an encoded SWAP and an encoded multiplication gate respectively.

Fault tolerant Procedure of General Phase Rotation

Define c_l as the interpolation coefficients such that

$$\forall\ f \in F[x],\ deg(f) \leq m-1, f(0) = \sum_{i=1}^{m} c_i f(\alpha_i). \tag{58}$$

To achieve a rotation by w^{ca}, we apply on the $l'th$ qupit the gate $|a\rangle \longmapsto w^{c_l a}|a\rangle$. This achieves the desired operation because:

$$\begin{array}{rcl} |S_a\rangle = & \sum_{f\in V, f(0)=a} |f(\alpha_1), ..., f(\alpha_m)\rangle & \longmapsto \\ & \sum_{f\in V1, f(0)=a} \Pi_{i=1}^{m} w^{c_l f(\alpha_l)} |f(\alpha_1), ..., f(\alpha_m)\rangle & = \\ & \sum_{f\in V, f(0)=a} w^a |f(\alpha_1), ..., f(\alpha_m)\rangle & . \end{array} \tag{59}$$

Fault Tolerant Error Correction Procedure

These procedures can be applied exactly as was done in the case of the CSS codes, except that everything is replaced by the natural analogue over the field F_p.

For the error correction procedure, we first generate the general cat state, which is a superposition of all strings of equal pits:

$$|cat_l^p\rangle = \sum_{a=0}^{p-1} |a^m\rangle \tag{60}$$

This is done by applying the generalized Fourier transform on a blank qupit, and then copying it using generalized CNOT gates to more blank qupits. We now apply a generalized Fourier transform on each of the coordinates, to achieve the state

$$\sum_{j, j\cdot\vec{1}=0\ mod\ p} |j\rangle, \tag{61}$$

which is simply an equal superposition of all strings j which satisfy $\sum_k j_k = 0\ mod\ p$, as a natural generalization of the corresponding state for qubits which was the superposition of all states with parity zero. The calculations of the syndrome, like in the CSS case, are done by computing the inner product of the corresponding rows in the parity check matrix of the code. To compute the inner product for the i'th row of the 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 no error occurred, the inner product is supposed to be zero, so the ancillary state does not change, since we have added to the ancilla a vector of which the coordinates sum up to 0 mod p. The rest of the error correction transforms smoothly to the case of computing over the field F_p. As a general rule, the gates NOT, $CNOT$ and Hadamard are replaced by their generalized version, i.e. generalized NOT, generalized CNOT and Fourier Transform over F_p, and this achieves the desired transformation.

5.2 Fault Tolerant Decoding and Encoding Procedures

The encoding procedure is achieved exactly like in the case of CSS codes: We generate $|S_0\rangle$ by correcting according to the code C_2, and then we add the input pits pit-wise. The decoding is also done exactly in the same way.

Fault Tolerant Procedure of the Fourier Transform Gate

The desired transformation is:

$$|S_a\rangle \longmapsto \frac{1}{\sqrt{p}}\sum_{b\in F} w^{ab}|S_b\rangle. \tag{62}$$

To achieve this, first fix $c_1, ..., c_m$ to be the interpolation coefficients, i.e. for any polynomial $f(x)$ over F_p with $deg(f) \leq m-1$, the zero coefficient of f satisfies $f_0 = \sum_i c_i f(\alpha_i)$. Denote by $w_l = w^{c_l}, l = 1, ..., m.$ and recall that in our notation

$$W(w_l) : |a\rangle \longmapsto \frac{1}{\sqrt{p}}\sum_{b\in F_p} w_l^{ab}|b\rangle. \tag{63}$$

We now apply $W(w_l)$ to the l'th qupit for all $1 \leq l \leq m$.

$$|S_a\rangle \longmapsto W(w_1)\otimes W(w_2)\otimes\cdots\otimes W(w_m)|S_a\rangle. \tag{64}$$

The transformation takes us from $|S_a\rangle$ to a state which we denote by $|\alpha\rangle$:

$$\begin{aligned} |S_a\rangle &= \frac{1}{\sqrt{p^d}}\sum_{f\in V, f(0)=a} |f(\alpha_1), ..., f(\alpha_m)\rangle \longmapsto \\ |\alpha\rangle &= \frac{1}{\sqrt{p^{d+m}}}\sum_{b1,b2,..bm\in F}\ \sum_{f\in V, f(0)=a} w^{\sum_{l=1}^m e_l f(\alpha_l) b_l}|b1, ..bm\rangle. \end{aligned} \tag{65}$$

For each string $b_1, ...b_m \in F_p$, associate the unique polynomial $b(x)$ which satisfies $b(\alpha_l) = b_l$, and has degree $deg(b) \leq m-1$. The exponent of w in equation 65 can be written in a much simpler form when $b(x)$ is of degree $deg(b) \leq m-d-1$. For such $b(x)$, the polynomial $h(x) = b(x)f(x)$ is of degree $deg(h) \leq m-1$ so:

$$\sum_{l=1}^m e_l f(\alpha_l)b(\alpha_l) = \sum_{l=1}^m e_l h(\alpha_l) = h(0) = f(0)b(0) \tag{66}$$

Hence, the sum over all b with $deg(b) \leq m-d-1$ in equation 65 gives:

$$\begin{aligned} &\frac{1}{\sqrt{p^{d+m}}}\sum_{b1,b2,..bm\in F, deg\ b(x)\leq m-d-1}\ \sum_{f\in V, f(0)=a} w^{b(0)f(0)}|b1, ..bm\rangle = \\ &\frac{1}{\sqrt{p^{m-d}}}\sum_{b1,b2,..bm\in F, deg\ b(x)\leq m-d-1} w^{b(0)a}|b1, ..bm\rangle = \\ &\frac{1}{\sqrt{p}}\sum_{b\in F_p} w^{ab}\frac{1}{\sqrt{p^{m-d-1}}}\sum_{b1,b2,..bm\in F, deg\ b(x)\leq m-d-1, b(0)=b} |b1, ..bm\rangle = \\ &\frac{1}{\sqrt{p}} w^{ab}\sum_{b\in F_p}|S'_b\rangle \end{aligned} \tag{67}$$

Where $|S'_b\rangle$ is the code word of the polynomial code when the degree of the polynomials is at most $m-d-1$, instead of d.

Now, we claim that the sum over the rest of the b's must vanish. The reason is that the norm of the above vector is 1. Now $|\alpha\rangle$ can be written as a sum of two vectors: The contribution from b's with $deg(b) \leq m-d-1$, and from the rest of the b's. The two are orthogonal, since different b's are orthogonal. Hence, the squared norm of $|\alpha\rangle$, which is 1, (because the operation is unitary and we started with a norm one vector) is the sum of the squared norms of the contribution of $deg(b) \leq m-d-1$, which is also 1, and the norm of the orthogonal vector. Thus, the norm of the sum over b's with $deg(b) > m-d-1$ must vanish.

Now, since $m = 3d+1$, the degree of the polynomials in $|S'_b\rangle$ is larger than that of $|S_b\rangle$. To fix this, we apply a degree reduction, which we will show shortly. The degree reduction takes the state $|S'_b\rangle$ to $|S'_b\rangle|S_b\rangle$. To complete the Hadamard transformation, we have to erase $|S'_b\rangle$. This is done by applying pit-wise subtraction of the second state from the first state. $|S'_b\rangle|S_b\rangle$ will then be taken to $|S'_0\rangle|S_b\rangle$. Then, we can discard the first register, and this completes the generalized Fourier transform.

Note that to achieve a generalized Fourier transform with w^r instead of w, we should simply replace w by w^r in all places where w appears in the above procedure.

Fault Tolerant Generalized Toffoli Gate

To apply the generalized Toffoli gate on $|S_a\rangle|S_b\rangle|S_c\rangle$, we will again use degree reduction. We first add another state, $|S'_0\rangle$ which is an encoded 0 using the same polynomial code except the degree is $2d$. Applying the general Toffoli gate pit-wise on the m coordinates, from the first two states $|S_a\rangle|S_b\rangle$ to the extra state gives $|S_a\rangle|S_b\rangle|S'_{ab}\rangle|S_c\rangle$, as is easy to check. We now apply degree reduction on the third register, using the fourth register as the target of the interpolation as is explained in the next section. This gives $|S_a\rangle|S_b\rangle|S'_{ab}\rangle|S_{ab+c}\rangle$. Applying the reverse of the generalized Toffoli gate, pit-wise on the first three registers gives $|S_a\rangle|S_b\rangle|S'_0\rangle|S_{ab+c}\rangle$, and we can discard the extra $|S'_0\rangle$ to obtain the desired state $|S_a\rangle|S_b\rangle|S_{ab+c}\rangle$.

5.3 Fault Tolerant Degree Reduction

The term *degree reduction* means the following. Let C' and C be two quantum polynomials codes of the same length m, which use polynomials of degree d' and d respectively, such that $d' > d$, where the polynomials are evaluated at the same m points in the field. A degree reduction procedure takes a word in the first code $|S'_a\rangle$ to the word $|S_a\rangle$ in the second code which encodes the same logical pit. The restriction on our procedure is that it is fault tolerant. This means that we cannot simply decode a and encode it again to get $|S_a\rangle$, because then even one error cannot be corrected, since the state of the environment will depend on the encoded pit.

The following solution, which generalizes classical degree reduction techniques by Ben-Or *et al*[46], is best illustrated in the following figure:

The idea goes as follows. Let

$$\sum_j c_j f(\alpha_j) = f(0) \tag{68}$$

so c_j are the interpolation coefficients for polynomials of degree less than m. We would have wanted to apply interpolation on the coordinates of the encoded state, so as to get a, and then to encode a by the code C. However, this is not fault tolerant. The trick is to first encode each coordinate in the original state $|S'_a\rangle$, so each

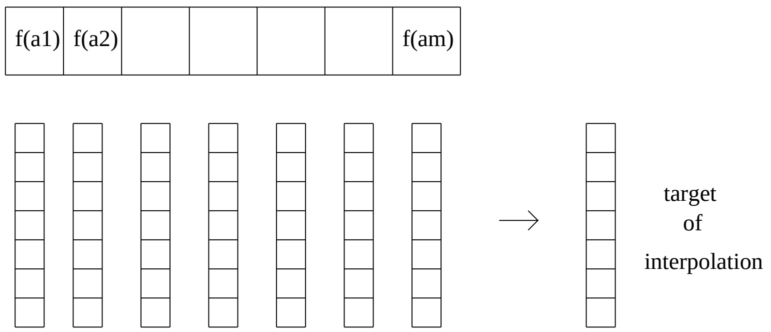


Figure IV.1: Scheme for degree reduction.

coordinate is *opened* to an encoded word in the code C. This is done by first adding m encoded states $|S_0\rangle$, and then, from each coordinate, say the i'th coordinate, apply m generalized CNOT gates to all the coordinates in the i'th state $|S_0\rangle$, so that the extra state now encodes the coordinate. This is illustrated in the figure by an $m \times m$ matrix, the columns of which are the encoded words, each encoding one coordinate in the original encoded word. We can now apply interpolation pit-wise from the opened coordinates to an extra state $|S_0\rangle$, by computing along the rows of the matrix. If the matrix elements are symbolically denoted by $a_{i,j}$, We apply the sum

$$\sum_j c_j a_{i,j} \tag{69}$$

over the elements in the i'th row of the matrix, and write it on the i'th coordinate of the extra state $|S_0\rangle$. Note that this computation is done separately on each row, or pit-wise, on the encoded opened words. It is easy to check that if no errors occurred, this indeed achieves the correct state $|S_a\rangle$ on the extra state which was the target of the interpolation. However, the state of the ancilla pits certainly depend on the original encoded pit. To take care of this, we "wrap" the encoded states by applying the reverse of the operations which we applied when we opened the states; To wrap the i'th coordinate, apply m generalized CNOT gates to subtract this coordinate from all the coordinates in the i'th opened state. If no error occurred, we get back the original m states $|S_0\rangle$, which can be discarded.

We would now like to take care of errors that might occur during this procedure, and this will force us to change the above procedure a little, and add error corrections. Note that an error during the opening of each coordinate might cause the entire encoded state to be wrong, since this part was not done fault tolerantly. Note that the interpolation is completely ruined, if even one of the columns of the matrix encodes a wrong pit. Thus, we must apply error corrections before we apply the interpolation. We first apply an error correction on each of the encoded states, so that it is a word in the code. Then, we apply an error correction according to the code C', *encoded* by the code C, to the overall word. We will immediately show how to do this. Given this transformation, it is easy to see that everything we have done is fault tolerant: We first open each coordinate, then apply error corrections, then apply the interpolation, and finally close each encoded state back to one coordinate, and discard the ancilla qubits. This achieves the transformation of $|S_a'\rangle$ to $|S_a'\rangle|S_a\rangle$ fault tolerantly, and completes the description of the degree reduction.

In the same way, one can achieve the transformation $|S'_a\rangle|S_b\rangle$ to $|S'_a\rangle|S_{a+b}\rangle$ by using the state $|S_b\rangle$ as the target of the interpolation, instead of $|S_b\rangle$. This will be useful in the construction of the Toffoli gate.

5.4 Fault Tolerant Encoded Error Correction

We show how to apply an encoded error correction on a doubly encoded word. This means the following. The state which we operate on is generated by encoding a state using the code C, and then encoding the resulting state again, using the code C'. We want to apply on the entire state an encoded error correction procedure. This means that the procedure is exactly the error correction procedure according to the code C, but each gate in the procedure is encoded by the code C'. We thus want to correct $C'-$encoded pit flips, which means the operator on words in C'

$$\mathcal{E}_{S_b(C')} : |S_a(C')\rangle \longmapsto |S_{a+1}(C')\rangle \tag{70}$$

and $C'-$encoded phase flips, operating on C' by

$$\mathcal{E}_{S_f(C')} : |S_a(C')\rangle \longmapsto w^a|S_a(C')\rangle \tag{71}$$

The natural way to correct these errors on words in the code C encoded by C' is simply to apply the error correction procedure of the code C, where each gate is replaced by a fault tolerant procedure according to the code C'. However, one must be careful, in order not to make a circular argument here. The problem is that we want to apply an encoded error correction in spite of the fact that we not yet have the fault tolerant procedures for the entire set of gates used in the error correction, and in particular, we cannot apply encoded Fourier transform, because in this procedure we use degree reduction, which uses encoded error correction. It turns out that this does not pose any difficulty since in this case we can omit the degree reduction everywhere. We proceed as follows.

For an encoded error correction, we need cat states as in equation 33, encoded by the code C'. Instead of generating cat states of length l as we do in the error correction procedure, we generate an encoded cat states on ml qupits. We first generate a state $|S_0(C'')\rangle$ in the polynomials code with degree $m-d-1$, which we denote by C'', using an encoding procedure according to this code. We then apply pit-wise generalized Fourier transforms, to get

$$\frac{1}{\sqrt{p}}\sum_{i=0}^{p-1}|S_a(C')\rangle \tag{72}$$

as is shown in section 5.2. We then copy it on $l-1$ ancilla states $|S_0(C')\rangle$ using generalized CNOT gates pit-wise. This generates the state

$$|S_{cat_l}\rangle = \frac{1}{\sqrt{p}}\sum_{i=1}^{p-1}|S_i(C')\rangle|S_i(C')\rangle\cdots|S_i(C')\rangle \tag{73}$$

(It might seem as if generating encoded cat states is much simpler than generating regular cat states, as we do in section 5.1, because we avoid here the main problem: verifying that the cat states are correct fault tolerantly. In fact, these verifications are not avoided here, but actually hidden in the generation of the encoded states, $|S'_0\rangle$ and $|S_0\rangle$, which are done using error corrections.)

From these cat states, it is easy to generate the encoded version of the state 61, using generalized Fourier transforms pit-wise. We thus have:

$$\sum_{j, j\cdot\vec{1}=0 \ mod \ p} |S_{j_1}(C')\rangle \cdots |S_{j_l}(C')\rangle \tag{74}$$

We now want to compute the j'th syndrome pit. Recall that this was done by computing the inner product of the string $(f(\alpha_1)...f(\alpha_m))$ with the j'th parity check matrix of the code. The un-encoded version of this operation is done by adding $h_j f(\alpha_j)$ to the j'th coordinate in the un-encoded version of the state 74. The natural encoded version of this is done by adding $h_j a_{i,j}$ to the $(mi+j)$'th coordinate in the state 74, where $a_{i,j}$ is the i'th coordinate in the state encoding $f(\alpha_j)$. We now apply a classical computation on the ancilla state, computing the sum of the encoded coordinates mod p, to give the corresponding syndrome bit. In this way we can compute all the pits in the syndrome, where like in the error correction procedure, everything is done all over again for each syndrome pit, to avoid propagation of errors. The syndrome is computed in this way for each one of the m^2 pits, and from the $(i,j)'$th copy we compute whether the j'th encoded coordinate suffered a pit-flip. If so we apply the reverse operation on the $(i,j)'$th pit.

A problem arises when we want to transform to the encoded $C-$basis, since the encoded Fourier transform gate is not yet in our repertoire. As was shown in the subsection 5.2, applying generalized Fourier transforms pit-wise (without degree reduction) takes us to the $C - basis$ of the polynomial code with degree $m-d-1$, if C' is of degree d. And so after the rotation to the $C-$basis we should actually correct pit-flips as in the first step, except we should replace C' everywhere by C''. At the end of this correction stage, we apply the inverse of the generalized Fourier transforms, pit-wise.

To show that this procedure indeed corrects general encoded errors, we first note that it suffices to correct encoded pit flips and encoded phase flips, just as in theorem 12. The proof of theorem 13 can be slightly modified to show that the procedure corrects and detects $C'-$encoded pit flips and $C'-$encoded phase flips. Denote by $\mathcal{H}$ the tensor product of these generalized Fourier transforms applied pit wise. The crucial fact is that a $C'-$encoded phase flip transforms by $\mathcal{H}$ to a C''-encoded pit flip:

$$\mathcal{H} \circ \mathcal{E}_{S_f(C')}|S_a(C')\rangle = \mathcal{H}(-1)^a|S_a(C')\rangle = (-1)^a|C_a(C'')\rangle = \mathcal{E}_{S_b(C'')}|C_a(C'')\rangle = \mathcal{E}_{S_b(C'')} \circ \mathcal{H}|S_a(C')\rangle \tag{75}$$

Hence, the sequence of equalities 19 holds also in our case, which shows that the above procedure corrects general encoded errors, if their number is smaller than that which the code C can correct.

6 Universality of the Sets of Gates $\mathcal{G}_1$ and $\mathcal{G}_2$.

The sets of gates $\mathcal{G}_1$ and $\mathcal{G}_2$ were not chosen arbitrarily; They were chosen so that any quantum computation can be expressed using these sets. In other words, these are universal sets of gates. A proof of this fact for $\mathcal{G}_1$ is missing in Shor's paper introducing the set $\mathcal{G}_1$, and we provide here a detailed proof, relying on a universality proof by Kitaev. A similar result was achieved independently by Boykin *et al.*[50]. The proof that the set of gates $\mathcal{G}_2$ is universal is more complicated, and is based on geometrical arguments on the groups $U(n)$, together with some basic facts from field theory. We start with a discussion of the issue of universality, and then show that our sets of gates $\mathcal{G}_1$ and $\mathcal{G}_2$ are indeed universal.

6.1 Universal Sets of Gates

We consider a set of gates, and ask which unitary operations can be constructed when gates from this set are applied on the qubits, in any order.

Definition 17 *Let $p \geq 2$. A set of gates $\mathcal{G}$ on $k > 1$ qupits is said to be universal if $\mathcal{G} \cup \{e^{i2\pi\theta} I\}_{real\ \theta}$ generates a dense subset in $U(p^k)$.*

Note that in quantum computation, one can multiply a gate by an overall scalar of absolute value 1 (which is sometimes called a phase factor) and the density matrices will remain the same. Therefore we can add to $\mathcal{G}$ all phase factors, without changing the set of operations on the quantum states. This definition indeed captures the notion of universality because it turns out that an algorithm using any set of gates can be translated to an algorithm which uses only gates from a universal set of gates $\mathcal{G}$, such that a), the new circuit computes a function which approximates to any given accuracy the original function (in total variation distance) b), the new circuit is only polylogarithmically larger and deeper than the original one, and c), the design of the new circuit can be efficiently computed given the design of the original circuit. This fact is based on two known results. The first is that a universal set of gates can be used to approximate matrices exponentially fast, and the sequence of gates from the universal set can be found efficiently:

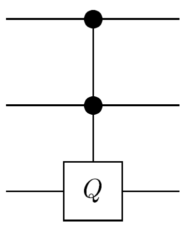
Theorem 17 *Kitaev, Solovay,:*

Consider a universal set of gates $\mathcal{G}$ over $U(m)$, for some integer m. Then there exists a polynomial p_m such that any element of $U(m)$ can be approximated up to ϵ by a word from G which is not longer than $p_m(log(1/\epsilon))$, and this word can be found efficiently (in time which is $p_m(log(1/\epsilon))$) by a classical Turing machine.

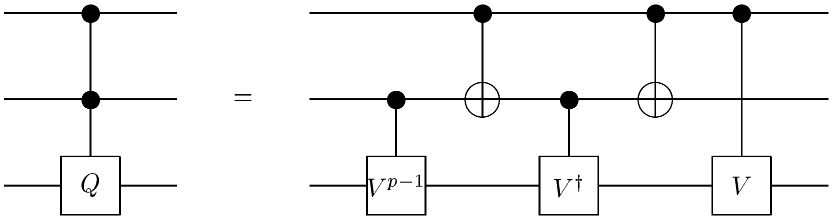
The proof of this theorem[135, 196] uses Lie groups and Lie Algebras, and will not be discussed here. The second result which we will need to justify the choice of definition of universality, is that an operation on any number of qubits (qupits) can be achieved using gates on two qubits (qupits). This was proved for qubits by DiVincenzo[93], and simplified by Barenco *et. al.*[20]. We give here a proof for the general case of qupits.

Theorem 18 *Let $\mathcal{G}$ be a universal set of gates on $k \geq 2$ qupits, for $p \geq 2$. Consider the Hilbert space of $m > k$ qupits. Then the gates in the set $\mathcal{G}$, extended to m qupits, generate a dense subset of $U(p^m)$.*

Proof: The proof generalizes ideas which were first used by Deutsch[88] and Barenco *et. al*[20]. We define a generalized Toffoli gate on m qupits, $T_m(Q)$, to be a gate which applies Q on the m'th qupit conditioned that the first $m-1$ qupits are in the state $p-1$:



Where the number of vertical wires is m. The conditioned Q can be applied on the k'th qupit, instead of the m'th one, in which case we denote $T_{m,k}(Q)$. Note that this definition is not the same generalization of the Toffoli gate as defined in the universal set of gates for polynomial codes, in subsection 5.1. Following Barenco *et. al.* [20] we show an explicit sequence of generalized Toffoli gates on $m-1$ qupits, $T_{m-1}(Q)'$s, which constructs $T_m(Q)$. Denote $V = Q^{\frac{1}{p}}$, so:



The picture should be interpreted as follows. The first wire is actually duplicated $m-2$ times. The sign $\oplus$ is actually the pit-flip, which adds 1 modulo p to every element in the field (This is a generalization of the NOT for F_2, and we sometimes call it the generalized NOT)

$$Pit - flip : |a\rangle \longmapsto |(a+1)\text{modp}\rangle \tag{76}$$

The sequence of gates which consists of the second and third gates, i.e. the controlled generalized NOT and the controlled $V^{\dagger}$, is repeated $p-1$ times. It is easy to check that the above circuit indeed gives the desired controlled Q, by considering what happens to the basis states, in two cases: all first $m-2$ qupits are equal to $p-1$, or not. Using the above scheme recursively we can construct a circuit which uses two-qupit gates and applies $T_m(Q)$ for any $m > 2$ and any one-qupit Q. Note that the recursion starts with two-qupit gates, which is the reason why we require $k \geq 2$.

The gate $T_m(Q)$ can be seen as applying a general Q on the subspace spanned by the last p basis vectors, while applying identity on the rest. The next step is to construct a generalization of the above gate, i.e. a gate which applies $Q \in U(p)$ on the subspace spanned by any p basis vectors $|i_1\rangle, .., |i_p\rangle$, while applying identity on the rest of the basis vectors. Denote this matrix by $T_m(Q, i_1, ..., i_p)$. We first note that $T_m(Q)$ with Q being any permutation on the basis vectors of the Hilbert space of one qupit, together with generalized NOT gates, generate all permutation matrices on m qupits. Generalized NOT gates are one-qupit gates, so we are allowed to use those. It suffices to construct all matrices of the form $\tau_{i,j}$, where $|i\rangle, |j\rangle$ are basis vectors for which the strings i, j differ in one coordinate. $\tau_{i,j}$ switches the two basis vectors $|i\rangle, |j\rangle$ and $\tau_{i,j}|k\rangle = |k\rangle$ for any $k \neq i, j$. W.l.o.g. let the coordinate on which i, j disagree be the last coordinate, and let this coordinate be equal to a in the string i, and to b in the string j. To construct $\tau_{i,j}$, apply sufficiently many generalized NOT gates on all the coordinates except the last coordinate, so that all these coordinates in both strings become equal to $p-1$. Now, apply $T_m(Q)$ with Q on the last coordinate being the matrix which permutes $|a\rangle$ and $|b\rangle$, leaving the rest of the basis vectors untouched. Then reverse all the generalized NOT gates on all the coordinates but the last one, which gives $\tau_{i,j}$, and therefore all permutations on basis vectors. The general $T_m(Q, i_1, ..., i_p)$ for any Q can clearly be achieved by first permuting the basis vectors $i_1, ..., i_p$ to the last p vectors, applying $T_m(Q)$, and then permuting back.

The last step is to use $T_m(Q, i_1, ..., i_p)$ to construct a general $p^m \times p^m$ unitary matrix U. Let us denote the p^m eigenvectors of U by $|\psi_j\rangle$ with corresponding eigenvalues $e^{i\theta_j}$. U is specified by $U|\psi_j\rangle = e^{i\theta_j}|\psi_j\rangle$. Define:

$$U_k|\psi_j\rangle = \begin{cases} |\psi_j\rangle & \text{if } k \neq j \\ e^{i\theta_k}|\psi_k\rangle & \text{if } k = j \end{cases} \tag{77}$$

Then $U = \Pi_{k=1}^{p^m} U_k$. It is left to show how to construct U_k: We first construct a transformation R which takes $|\psi_k\rangle$ to $\lambda|(p-1)^m\rangle$ where λ is a complex number of absolute value 1. We don't care what R does to the rest of the vectors. After applying R we apply the generalized Toffoli which takes $|(p-1)^m\rangle \longmapsto e^{i\theta_k}|(p-1)^m\rangle$ and does nothing on the rest of the basis states. Then we take $\lambda|(p-1)^m\rangle$ to $|\psi_k\rangle$ by applying R^{-1}. This indeed achieves U_k, as can be easily checked. To construct R, and similarly R^{-1}, we start with $|\psi_k\rangle$, and we first make the coefficient in front of $|0^m\rangle$ to be zero by a rotation in the plane spanned by $|0^m\rangle$ and $|(p-1)^m\rangle$, which is a special case of the $T_m(Q, i_1, ..., i_p)$ we have dealt with before. Thus, the weight of $|0^m\rangle$ has been shifted to $|(p-1)^m\rangle$. In this way, the weights in front of all basis vectors, one by one, are shifted to $|(p-1)^m\rangle$, and this achieves R. $\Box$

We proceed to prove a few geometrical lemmas which will be useful in proving universality of the sets of gates we use. The first and second lemmas were used by Kitaev[135] for proving universality of his set of gates. We will use these lemmas to prove universality of $\mathcal{G}_1$ and $\mathcal{G}_2$. It seems that the method which is used here to prove universality is quite general.

Lemma 20 *Let $n \geq 3$ Let $|\alpha\rangle \in \mathcal{C}^n$. Let H be the subgroup in $U(n)$, which fixes $|\alpha\rangle$, Let $V \in U(n)$ be a matrix not in H. Then, the subgroup generated by H and V is dense in $U(n)$.*

Proof: We use some ideas which were first used by Deutsch[88]. Let $G \subseteq U(n)$ be the closed subgroup generated by H and V. Let U be a $n \times n$ unitary matrix, and denote the n eigenvectors of U by $|\psi_j\rangle$ with corresponding eigenvalues $e^{i\theta_j}$. U is specified by $U|\psi_j\rangle = e^{i\theta_j}|\psi_j\rangle$. Define:

$$U_k|\psi_j\rangle = \begin{cases} |\psi_j\rangle & \text{if } k \neq j \\ e^{i\theta_k}|\psi_k\rangle & \text{if } k = j \end{cases} \tag{78}$$

Then $U = \Pi_{k=1}^{n} U_k$. It is thus enough to construct U_k. Let $|\psi_k\rangle = c|\alpha\rangle + d|\delta\rangle$. Let $|\gamma\rangle$ be perpendicular to both $|\alpha\rangle$ and $|\delta\rangle$. Such $|\gamma\rangle$ exists because the dimension is at least 3. To construct U_k, it suffices to rotate $|\psi_k\rangle$ to $f|\gamma\rangle$, where f is an additional phase factor, apply a matrix which multiplies $|\gamma\rangle$ by $e^{i\theta_k}$ and does nothing to vectors orthogonal to $|\gamma\rangle$, then rotate back $f|\gamma\rangle$ to $|\psi_k\rangle$. The middle operation of multiplication by the phase $e^{i\theta_k}$ of $|\gamma\rangle$ is clearly in H. The transformation which takes $|\psi_k\rangle$ to $f|\gamma\rangle$ can be approximated to any degree of accuracy as follows.

We first find some improved versions of V in G. We have $V|\alpha\rangle = a|\alpha\rangle + b|\beta\rangle$. By multiplying V on the left by a suitable element of H, we can arrange that the resulting element V_1 satisfies: $V_1|\alpha\rangle = a|\alpha\rangle + b|\delta\rangle$. There is a unit vector $|\eta\rangle$ which is mapped by V_1 onto $\bar{b}|\alpha\rangle - \bar{a}|\delta\rangle$. So $V_1|\alpha\rangle$ is perpendicular to $V_1|\eta\rangle$. Hence $|\eta\rangle$ is perpendicular to $|\alpha\rangle$. So we can find $h \in H$ such that $h|\delta\rangle = |\eta\rangle$. Let $V_2 = V_1 \cdot h$. Then $V_2|\alpha\rangle = a|\alpha\rangle + b|\delta\rangle$, and V_2 maps the space spanned by $|\alpha\rangle$ and $|\delta\rangle$ onto itself. $|\gamma\rangle$ and $V_2|\gamma\rangle$ are perpendicular to both $|\alpha\rangle$ and $|\delta\rangle$. Hence there is an $h_1 \in H$ that fixes both $|\alpha\rangle$ and $|\delta\rangle$ and that moves $V_2|\gamma\rangle$ onto $|\gamma\rangle$. Let $V_3 = h_1 \cdot V_2$. Then V_3 fixes $|\gamma\rangle$ and $V_3|\alpha\rangle = a|\alpha\rangle + b|\delta\rangle$.

We now construct a sequence of elements $W_i \in G$ (by induction on i) such that $W_i|\psi_k\rangle$ has the form $ca^i|\alpha\rangle + d_i|\gamma\rangle$. If we can do this, then for i large, ca^i tends to 0 since $|a| < 1$. Hence $|d_i|$ tends to 1 as i goes to ∞ and the lemma will be proved. For $i = 0$, we simply choose $W_0 \in H$ such that $W_0|\delta\rangle = |\gamma\rangle$. Now suppose W_i has been chosen to satisfy our inductive hypothesis. Then

$$V_3 W_i|\psi_k\rangle = ca^{i+1}|\alpha\rangle + bca^i|\delta\rangle + d_i|\gamma\rangle \tag{79}$$

The vector $bca^i|\delta\rangle + d_i|\gamma\rangle$ is clearly perpendicular to $|\alpha\rangle$ and non-zero. Hence we can find $h^\star \in H$ that moves this vector to a vector of the form $d_{i+1}|\gamma\rangle$. It suffices now to set $W_{i+1} = h^\star V_3 W_i$. $\Box$

Lemma 21 *Let U_1, U_2 be two non-commuting matrices in $SU(2)$, such that their eigenvalues are not integer roots of unity. The subgroup generated by U_1, U_2 is dense in $SU(2)$.*

Proof: If x is an element of $SU(2)$ not of finite order, then the closed subgroup generated by x is connected and of dimension 1. The closed group generated by both U_1, and U_2 is thus connected, and is non commutative. Any connected non-commutative subgroup of $SU(2)$ is *all* of $SU(2)$. $\Box$

Lemma 22 *Let A, B be two non orthogonal subspaces of C^n. Let G_A, G_B be dense subsets of $U(A)$,$U(B)$ respectively. Then the subgroup generated by $G_A \cup G_B$ is dense in $U(A \oplus B)$.*

Proof: We will first prove the lemma for the case $dim(A) = dim(B) = 1$, and $A \neq B$. Let G be the group generated by $U(A) \cup U(B)$. We consider the natural map from $U(2)$ to $SU(2)$, $U \longmapsto U/\sqrt{Det(U)}$, and look at the image of G in $SU(2)$ under this map. We can find in $U(A)$ and in $U(B)$ two matrices such that their images are of infinite order and do not commute, and by lemma 21, the image of G is dense in $SU(2)$. Hence, G contains all matrices in $SU(2)$ multiplied by a certain phase. Now, observe that taking the commutators of these matrices erases the phases. Using the fact that the commutator group of $SU(2)$ is $SU(2)$ itself, $[SU(2), SU(2)] = SU(2)$, we have that $[G, G] \subseteq G$ contains a dense subset of $SU(2)$. Using $U(A)$ we can generate all phase factors, which proves the lemma for the case $dim(A) = dim(B) = 1$.

Let us now consider the general case. Let $|\alpha_1\rangle, ...|\alpha_m\rangle$ be a basis for A. Let $|\beta_{m+1}, ...|\beta_s\rangle$ be a basis for B, such that each vector in the basis of B is not orthogonal to A. (To construct such a basis, let us start with any given basis. At least one basis vector is not orthogonal to A. Adding it to all the vectors in the basis which are orthogonal to A gives the desired basis.) Let B_i be the subspace spanned by $|\beta_i\rangle$, and denote by $A_i = A \oplus B_1 \oplus B_2 \oplus \cdots \oplus B_i$. We will show by induction on i that the group $U(A_{i-1})$ is in the subgroup generated by $U(A) \cup U(B)$ and all phase factors on $A \oplus B$.

For $i = 0$, the claim is trivial. We assume for $i - 1$, that the group $U(A_{i-1})$ is in the subgroup generated by $U(A) \cup U(B)$, and prove for i. If $|\beta_i\rangle \in A_{i-1}$, the induction step is proven. Hence, we assume $|\beta_i\rangle \notin A_{i-1}$. We have $|\beta_i\rangle \in B$, so we have in our repertoire $U(|\beta_i\rangle)$. We divide to two cases. If $dim(A_{i-1}) = 1$, then the induction step follows from the simple case in which $dim(A) = dim(B) = 1$. If $dim(A_{i-1}) \geq 2$, then $dim(A_{i-1} \oplus B_i) \geq 3$, and we can use lemma 20. Let $|\gamma_i\rangle \in A_i$ be the (non trivial) projection of $|\beta_i\rangle$ on the subspace orthogonal to A_{i-1}. $U(A_{i-1})$ is exactly the subgroup of $U(A_i)$ which fixes $|\gamma_i\rangle$. The matrix which multiplies $|\beta_i\rangle$ by a non trivial phase factor, leaving all orthogonal vectors untouched, does not fix $|\gamma_i\rangle$. Hence lemma 20 can be applied, and we have $U(A_i)$. $\Box$

6.2 Universality of the Set of Gates $\mathcal{G}_1$ used for CSS Codes

The set of gates used for CSS codes is shown here to be universal. This is done by a reduction to a proof of Kitaev[135], who used lemmas 20 and 21.

Theorem 19 *$\mathcal{G}_1$ together with all phase factors generate a dense subgroup in the group of special unitary matrices operating on five qubits, $U(2^5)$.*

Proof: The proof is based on a result by Kitaev[135], which asserts that the following set of gates is universal: CP is a two qubit gate which takes $|11\rangle \longmapsto i|11\rangle$ and applies identity on the rest, and the Hadamard gate. Let us first show how to construct CP from our set of gates: We will denote by T_k a generalized Toffoli on k qubits, which applies not on the k'th bit conditioned that the first $k-1$ bits are 1. We first construct T_4, using five qubits. This can be done using $T_3's$, as follows:

$$\begin{aligned} a,b,c,d,e &\longmapsto \\ a,b,c+ab,d,e &\longmapsto \\ a,b,c+ab,d,e+cd+abd &\longmapsto \\ a,b,c,d,e+cd+abd &\longmapsto \\ a,b,c,d,e+abd & \end{aligned} \tag{80}$$

which is exactly T_4 from qubits $1,2,4$ to 5. Define $X = P_4^3 T_4 P_4 T_4$, where P_4 applies the phase flip P on the fourth qubit. X takes $|1110\rangle \longmapsto i|1110\rangle$, $|1111\rangle \longmapsto -i|1111\rangle$ and does nothing to the other basis states. X^2 is the three qubit gate which gives $|111\rangle \longmapsto -|111\rangle$ and identity on the rest of the basis vectors, tensored with identity on the fourth qubit. Now, to construct CP we apply $P_3^3 T_3 P_3 T_3$ which gives $|110\rangle \longmapsto i|110\rangle, |111\rangle \longmapsto -i|111\rangle$ and identity on the rest, and applying X^2 we get CP tensor with identity on the third qubit.

The theorem now follows from Kitaev[135], who used the geometrical lemmas 20 and 21 to show that CP and the Hadamard gate are universal. Here is Kitaev's argument. Denote by

$$\begin{aligned} X_1 &= H_1 (CP)_{1,2} H_1 \\ X_2 &= H_2 (CP)_{2,1}^{-1} H_2 \end{aligned} \tag{81}$$

Define $Y_1 = X_1 X_2^{-1}$ and $Y_2 = X_2 X_1^{-1}$. Note that Y_1, Y_2 both operate as the identity on the two states $|00\rangle$, $|\eta\rangle = |01\rangle + |10\rangle + |11\rangle$. Denote by L the subspace orthogonal to $|00\rangle$ and $|\eta\rangle$. Then $Y_1, Y_2 \in SU(L)$. Y_1, Y_2 do not commute, and their eigenvalues are $\frac{1}{4}(1 \pm \sqrt{15})$. Hence, by lemma 21 they generate a dense subgroup in $SU(L)$. Now, add to Y_1, Y_2 also the gate CP itself. This gate fixes $|00\rangle$, but does not stabilize the space $|\eta\rangle$. Thus, in the space $L \oplus |\eta\rangle$, Y_1, Y_2 generate a dense subgroup in the subgroup which fixes $|\eta\rangle$, while CP is not in this subgroup. We can use lemma 22 to show that Y_1, Y_2, CP generate a dense subgroup in $SU(L \oplus |\eta\rangle)$. Finally, add H_1 to the set Y_1, Y_2, CP. We have seen that Y_1, Y_2, CP generate a dense set in the subgroup that fixes $|00\rangle$, while H_1 is not in this subgroup. Hence, H_1, Y_1, Y_2, CP, (which are all gates generated by H and CP), generate a dense subgroup of $SU(L \oplus |\eta\rangle \oplus |00\rangle) = SU(4)$. Together with all phase factors, we get the unitary group on two qubits. The result follows from theorem 18. $\Box$

6.3 Universality of the Set of Gates $\mathcal{G}_2$ for Polynomial Codes

We would now like to show that the set of gates for polynomial codes, $\mathcal{G}_2$, is universal. Note that in this case we are working with qupits, i.e over the field F_p, so universality is proved for matrices operating on qupits. We will show that $\mathcal{G}_2$ together with all phase factors generate a dense subgroup in $U(p^3)$ (theorem 20). We will first prove an analogue of the fact that one qubit gates and classical gates are universal for qubits[].

Lemma 23 *The set of gates consisting of all one-qupit gates $U(C^p)$ and all classical two-qupit gates generates all unitary matrices on two qupits, $U(C^{p^2})$.*

Proof: Let G be the closed subgroup in $U(C^{p^2})$ generated by all one-qupit gates $U(C^p)$ on each of the 2 qupits, and all classical two-qupit gates. We will use the fact that the lemma we are trying to prove is already known for the case of $p = 2$. Let S be the two dimensional subspace in C^p spanned by the first two basis vectors $|0\rangle$ and $|1\rangle$. Clearly, $U(S)$ is in $U(C^p)$. Let S_l be the two dimensional subspace in the Hilbert space of the l'th qupit, and let $A = S_1 \otimes S_2$. This subspace is isomorphic to the Hilbert space of 2 qubits, and by lemma 23 for the case of qubits, we have that $U(A)$ can be generated. In the same way, we can define all possible subspaces of the form A: For each qupit, pick two basis vectors out of the p possible basis vectors. Let A be the tensor product of the subspaces spanned by these pairs of vectors. There are $m = \binom{p}{2}^2$ subspaces of this form. By the same argument as before, $U(A)$ is in G for any such A.

We now want to apply lemma 22. First, we claim that the subspaces of the above form can be ordered, $A_1, ...A_m$ such that

$$\oplus_{i=1}^{j-1} A_i \not\perp A_j \tag{82}$$

as is easy to check. Second,

$$\oplus_{i=1}^{m} A_i = C^{p^k} \tag{83}$$

Since $U(A_i) \subset G$, the proof follows from lemma 22. $\Box$

It is thus enough to prove that all one qupit gates are in our repertoire. Denote by Q_0 the one qupit matrix of the form:

$$Q_0 = \begin{pmatrix} w & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{pmatrix} \tag{84}$$

For $0 \leq i < p$, we can similarly denote by Q_i the one qupit diagonal matrix which multiplies $|i\rangle$ by w and applies identity of the other basis states.

Lemma 24 *Q_i and Q_i^{-1} are in the subgroup generated by $\mathcal{G}_2$ on three qupits.*

Proof: We will generate $Q_i \otimes I \otimes I$, which applies the following transformation:

$$\begin{aligned} |i\rangle|a\rangle|b\rangle &\longmapsto w|i\rangle|a\rangle|b\rangle \\ |j\rangle|a\rangle|b\rangle &\longmapsto |j\rangle|a\rangle|b\rangle \quad j \neq i \end{aligned} \tag{85}$$

To achieve this transformation, we view this gate as applying multiplication by w of the second qupit, conditioned that the first qupit is i. Recall that in our notation, P was a one qupit gate which applies a phase shift, $P|a\rangle = w^a|a\rangle$, and B is a one qupit gate which applies a pit shift, $B|a\rangle = |a+1\rangle$. Both B and P are in our repertoire. We now claim that the controlled B, which we denote by CB, is also in our repertoire. CB is the gate which applies B on the second qupit, conditioned that the first qupit is i, and applies the identity on the second qupit if the state of the first qupit is anything but i. We can generate $CB \otimes I$, since the generalized Toffoli gate, together with the addition gate, generate all permutations on basis states of three qupits. Now consider the commutator

$$P^{-1} \cdot CB^{-1} \cdot P \cdot CB. \tag{86}$$

This is exactly the gate we want, since if the first qupit is i, the matrix which is applied on the second qupit is $P^{-1} \cdot B^{-1} \cdot P \cdot B = wI$. If the first qupit is in a basic state which is not $|i\rangle$, the matrix which is applied on the second qupit is simply the identity. $\Box$

We now consider the two commutator matrices in $< \mathcal{G}_2 >$:

$$X_i = HQ_iH^{-1}Q_i^{-1} \quad , \quad Y_i = HQ_i^{-1}H^{-1}Q_i \tag{87}$$

where H is the generalized Fourier transform. We also define the two dimensional subspace S_i:

$$S_i = span\{|i\rangle, \sum_{b\in F_p, b\neq i} w^{ib}|b\rangle\}. \tag{88}$$

The claim is that X_i and Y_i operate as the identity on the orthogonal subspace to S_i.

Lemma 25 *X_i and Y_i operate as the identity on $S_i^{\perp}$.*

Proof: It is easy to write down explicitly the matrix elements of $X_i = HQ_iH^{-1}Q_i^{-1}$ and $Y_i = HQ_i^{-1}H^{-1}Q_i$.

$$\begin{aligned} (X_i)_{ab} &= \begin{cases} \delta_{ab} + \frac{1}{p}(w-1)w^{(a-b)i} & \text{if } b \neq i \\ \delta_{ab} + \frac{1}{p}(w-1)w^{(a-b)i}]w^{-1} & \text{if } b = i \end{cases} \\ (Y_i)_{ab} &= \begin{cases} \delta_{ab} + \frac{1}{p}(w^{-1}-1)w^{(a-b)i} & \text{if } b \neq i \\ \delta_{ab} + \frac{1}{p}(w^{-1}-1)w^{(a-b)i}]w & \text{if } b = i \end{cases} \end{aligned} \tag{89}$$

To see that X_i and Y_i operate as the identity on the subspace orthogonal to S_i, consider the matrices $X_i - I$ and $Y_i - I$, which satisfy equation 89 where we subtract δ_{ab} from each term. It is easy to see that the orthogonal vectors to S_i are all in the kernel of $X_i - I$ and $Y_i - I$, since the vectors v orthogonal to S_i satisfy:

$$v_i = 0 \quad , \quad \sum_{b, b\neq i} v_b w^{-ib} = 0 \tag{90}$$

and thus

$$\begin{aligned} \sum_{b\in F_p} (X_i - I)_{ab} v_b &= \frac{1}{p}(w-1)w^{ai} \sum_{b\neq i} v_b w^{-bi} = 0 \\ \sum_{b\in F_p} (Y_i - I)_{ab} v_b &= \frac{1}{p}(w^{-1}-1)w^{ai} \sum_{b\neq i} v_b w^{-bi} = 0. \quad \Box \end{aligned} \tag{91}$$

We now consider the operation of X_i and Y_i on the subspace S_i. We claim that they generate a dense subgroup in the group of 2×2 unitary matrices $U(2)$ operating on S_i. We will want to use lemma 21, and the main effort is to prove that the eigenvalues of X_i and Y_i are not integer roots of unity. The proof of this fact is based on some basic results regarding cyclotomic fields and Galois fields, which can be found in "Introduction to Cyclotomic Fields" by Washington[214].

Lemma 26 *For $p > 3$, the eigenvalues of X_i and Y_i confined to S_i are not integer roots of unity.*

We now consider the operation of X_i and Y_i on the subspace S_i, which we span by the orthonormal basis vectors $|i\rangle$ and $|\alpha_i\rangle = \frac{1}{\sqrt{p-1}} \sum_{b \in F_p, b \neq i} w^{ib}|b\rangle$. By equation 89 and a little algebra we get:

$$X_i|i\rangle = (1 + \frac{1}{p}(w-1))w^{-1}|i\rangle + \frac{\sqrt{p-1}}{p}(w-1)w^{-i^2-1}|\alpha_i\rangle \tag{92}$$
$$X_i|\alpha_i\rangle = (\frac{\sqrt{p-1}}{p}(w-1)w^{i^2})|i\rangle + (1 + \frac{p-1}{p}(w-1))|\alpha_i\rangle$$

and for Y_i we get the transformation:

$$Y_i|i\rangle = (1 + \frac{1}{p}(w^{-1}-1))w|i\rangle + \frac{\sqrt{p-1}}{p}(w^{-1}+1)w^{-i^2-1}|\alpha_i\rangle \tag{93}$$
$$Y_i|\alpha_i\rangle = (\frac{\sqrt{p-1}}{p}(w^{-1}-1)w^{i^2})|i\rangle + (1 + \frac{p-1}{p}(w^{-1}-1))|\alpha_i\rangle$$

We denote by X_i', Y_i' the two matrices confined to S_i. Observe now that the determinant of X_i' is 1, because $|Q_i| = w, |H| = 1$ and $|X_i| = |H| \cdot |Q_i| \cdot |H^{-1}| \cdot |Q_i^{-1}| = 1$, and $|X_i'| = |X_i|$ since X_i operates as the identity on the subspace orthogonal to S_i. Hence X_i' and similarly Y_i' have determinant 1. The typical polynomial for X_i' is $\lambda^2 - Tr(X_i')\lambda + Det(X_i')$, which amounts to:

$$f(\lambda) = \lambda^2 - \frac{2 + (p-1)(w + w^{-1})}{p}\lambda + 1 \tag{94}$$

Y_i' has exactly the same typical polynomial. We want to show that the roots of this polynomial are not integer roots of unity. Let us assume that one of the roots of the above polynomial is a primitive n'th roots of unity, denoted by ζ_n. The other solution is the complex conjugate of ζ_n, and we have

$$\frac{2 + (p-1)(w + w^{-1})}{p} = \zeta_n + \zeta_n^{-1} \tag{95}$$

We will first prove that $n = p$. Denote by $Q(w), Q(\zeta_n)$ the Galois extensions of the field of rationals obtained by adjoining w, ζ_n, respectively, to the field of rationals Q. Also, denote by $Q(w)^+$, the maximal real subfield of $Q(w)$, obtained by extending Q by $w + w^{-1}$, and similarly denote the maximal real subfield of $Q(\zeta_n)$ by $Q(\zeta_n)^+$. The idea is that by equation 95, $Q(\zeta_n)^+ = Q(w)^+$.

The degree of the extension $deg(Q(w)/Q(w)^+)$ is exactly 2, since w is a root of the minimal two degree polynomial $x^2 - (w + w^{-1})x + 1$ over the field $Q(w)^+$. Similarly, $deg(Q(\zeta_n)/Q(\zeta_n)^+) = 2$. On the other hand, $deg(Q(w)/Q) = p - 1$ and $deg(Q(\zeta_n)/Q) = \phi(n)$, by theorem 2.5 in [214]. Now, for three fields, F_1, F_2, F_3 such that F_3 extends F_2 which extends F_1, we have $deg(F_3/F_1) = deg(F_3/F_2)deg(F_2/F_1)$. It follows that:

$$deg(Q(w)^+/Q) = \frac{p-1}{2} \quad , \quad deg(Q(\zeta_n)^+/Q) = \frac{\phi(n)}{2} \tag{96}$$

But $Q(w)^+ = Q(\zeta_n)^+$, which implies that the degrees of extensions are equal, so $\phi(n) = p - 1$.

Now if $p > 3$, then $w + w^{-1} \notin Q$, since $deg(Q(w + w^{-1})/Q) = (p-1)/2 > 1$. Since $w + w^{-1} \in Q(w) \cap Q(\zeta_n)$ we have that $Q \neq Q(w) \cap Q(\zeta_n)$. If p and n were relatively prime we would have $Q = Q(w) \cap Q(\zeta_n)$ (by proposition 2.4 in [214]) and so p must divide n, say $n = p^r m$, with m coprime to p. This implies:

$$\phi(n) = p^{r-1}(p-1)\phi(m) = p - 1. \tag{97}$$

This can only be satisfied if $n = p$. We get:

$$\frac{1 + (p-1)cos(2\pi/p)}{p} = cos(k * (2\pi/p)). \tag{98}$$

This equation is not satisfied by any integer k, for $p > 3$, since the left hand side is a convex combination of $cos(2\pi/p)$ and 1, and no real part of a p'th root of unity lies between these two points. This shows that the eigenvalues of X_i' and Y_i' are not integer roots of unity. □

We can now prove the theorem.

Theorem 20 *$\mathcal{G}_2$ together with all phase factors generate a dense subgroup in $U(p^3)$, for $p > 3$.*

Proof: Since we are allowed to operate on three qupits, we can generate Q_i by lemma 24. We can thus generate X_i and Y_i. It is easy to see that if $w \neq \pm 1$, or $p \neq 2$ the off diagonal terms of $X_i'Y_i' - Y_i'X_i'$ are not zero, so X_i' and Y_i' do not commute. We can apply lemma 21, using the fact that the eigenvalues are not roots of unity, by lemma 26. We thus have a dense subset of the unitary group on all subspaces S_i. We have that $\oplus_i S_i = C^p$, and S_i is not orthogonal to $\oplus_{j=0}^{i-1} S_j$. Using lemma 22 and induction on i, we show that we can generate a dense subgroup of $U(C^p)$, i.e. all operations on one qupit. Note that apart from all one-qupit gates, we also have in our repertoire all classical gates on three qupits, and in particular all classical gates on two qupits which act trivially on the third qupit. The theorem follows from lemma 23, which guarantees universality on two qupits, and theorem 18 which shows that these matrices can be used to construct all matrices on three qupits, $U(p^3)$. □

7 Fault Tolerance for Probabilistic Noise

In this section we show how to use the fault tolerant procedures described in the previous sections, in order to achieve robustness against probabilistic noise with constant error rate. We use qubits all along, but everything works for qupits in exactly the same way. The scheme is not specific for the quantum codes defined earlier but works with any quantum code, as long as it is a *quantum computation code* (to be defined shortly). We then define the recursive simulation of the unreliable circuit, using such codes. To analyze the propagation of errors, we define the notion of sparse errors and sparse fault paths. The threshold theorem is proved in two parts. First, we show that sparse fault paths are good, meaning that they cause sparse errors. Then we show that non-sparse fault paths are rare. We give the exact threshold condition on the error rate, which depends on the size of the fault tolerant procedures.

7.1 Quantum Computation Codes

In order to improve reliability of a quantum circuit, we need a quantum code accompanied with a universal set of gates which can be applied fault tolerantly on states encoded by C. The code should also be accompanied with fault tolerant decoding, encoding and error correction procedures. In the last three sections we have shown that CSS codes accompanied with the set of gates $\mathcal{G}_1$, and polynomial codes accompanied with the set of gates $\mathcal{G}_2$, are quantum computation codes. However, any quantum computation code can be used in order to improve the reliability of a quantum circuit, provided that its procedures satisfy two simple restrictions.

Definition 18 *A quantum code is called a quantum computation code if it is accompanied with a universal set of gates G, with fault tolerant procedures, and with fault tolerant encoding, decoding and correction procedures. Moreover, we require that (1) all procedures use only gates from G, and (2) The correction procedure takes any density matrix to some word in the code.*

The first restriction allows us to use this code recursively. The second restriction is required for a reason which will become clear in the proof of lemma 27. Note that the procedures for CSS codes and polynomial codes satisfy the above restrictions. We define the spread of the computation code.

Definition 19 *Let C be a computation code using the set of gates $\mathcal{G}$. Consider a fault tolerant procedure for a gate in $\mathcal{G}$ preceded by fault tolerant error corrections on each block participating in the procedure. The spread of the code is l if one fault which occurs during this sequence of gates, for any gate in $\mathcal{G}$, effects at most l qubits in each block at the end of the procedure.*

We require that the number of errors that the code can correct, d, is larger than the spread l, so that we can tolerate at least one error in a procedure preceded by error corrections.

$$l \leq d. \tag{99}$$

7.2 Recursive Simulations

From now on, fix a quantum computation code C. It encodes one qubit on m qubits, it corrects d errors, it is accompanied with a universal set of gates $\mathcal{G}$, and a set of fault tolerant procedures with spread l. We will use m which is constant and does not grow with n. Let M_0 be a quantum circuit using gates from $\mathcal{G}$. Then we simulate M_0 by a more reliable circuit M_1, as follows. Each qubit is replaced by a block of qubits. Each time step in M_0, transforms in M_1 to a working period, which consists of two stages. In the first stage, an error correction procedure is applied on each block. At the second stage, each gate which operated in the simulated time step in M_0 is replaced in M_1 by its procedure, operating on the corresponding blocks.

The input of M_1 is the input to M_0, where each input bit is duplicated m times. Before any computation is done on this input, we apply in M_1 a fault tolerant encoding procedure on each block, which takes $|0^m\rangle$ to $|o^m\rangle|S_0\rangle$ and similarly for 1. At the end of the computation we will again use redundancy, for each block, we apply a decoding procedure which decodes the state to m copies of the logical bit, and the output of M_1 is defined as the majority of the bits in each block. Note that redundancy in the input and output is unavoidable if we want robustness to noise, because otherwise the probability for the input and output to be correct is exponentially small. This is also assumed in the classical scenario[167].

The above mapping, denoted by $M_1 = \phi(M_0)$, is one level of the simulation. ϕ is then applied again, on M_1, to give M_2, and we repeat this r levels to get $M_r = \phi^r(M_0)$, an r-simulating circuit of M_0. The number of levels r will be $O(\mathrm{polyloglog(V(M_0))})$, where $V(M_0)$ is the volume of M_0, i.e. the number of locations in M_0. The output of M_r is defined by taking recursive majority on the outputs. This means that first we take the majority in each block of size m, then we take the majority, of m such majority bits, and so on for r levels, to give one output bit.

The advantage of using recursive simulations, instead of one step of simulation, as in Shor's scheme[190], is that in each level error corrections to the current level are added. This means that error corrections of all scales are applied frequently during the computation procedures, preventing accumulation of errors in all levels. This allows robustness against constant error rate, which seems impossible to achieve in one level of simulation.

7.3 Blocks and Rectangles

The recursive simulations induce a definition of s-blocks: Every qubit transforms to a block of m qubits in the next level, and this block transforms to m blocks of m qubits and so on. One qubit in M_{r-s} transforms to m^s qubits in M_r. This set of qubits in M_r is called an s-block. An 0-block in M_r is simply a qubit. In the same way, one can define $s-$working periods. Each time step in M_0 transforms to w time steps in M_1, and an $s-$working period is the time interval in M_r which corresponds to one time step in M_{r-s}.

Recall the definition of a location in a quantum circuit in subsection 8.1. The recursive simulation induces a partition of the set of locations in M_r to generalized rectangles. An $r-$rectangle in M_r is the set of locations which originated from one location in M_0. This is best explained by an example: Consider a CNOT gate which is applied in M_0 at time t on qubits q_1, q_2. The location $((q_1, q_2), t)$ in M_0 transforms in M_1 to error correction procedures on both blocks, followed by the procedure of the CNOT gate. The set of locations in these three procedures is the 1-rectangle in M_1 which originated from the location $((q_1, q_2), t)$ in M_0. More generally, an $s-$rectangle in M_r is the set of points in M_r which originated from one location in M_{r-s}. Note that the partition to s-rectangles is a refinement of the partition to $(s+1)$-rectangles. An 0-rectangle in M_r is just one location.

7.4 Sparse Errors and Sparse Faults

In a noiseless scenario, the state of M_r at the end of each r-working period encodes the state of M_0 at the end of the corresponding time step. However, we assume that errors occur in M_r and we want to analyze those. In order to analyze the propagation of errors in M_r, we need to distinguish between the actual faults that occur during the computation, and the errors that are caused in the state. First, we focus on the errors in the states, and define a distance between encoded states. The hierarchy of blocks requires a recursive definition.

Definition 20 *Let B be the set of qubits in n $r-$blocks. An (r,k)-sparse set of qubits A in B is a set of qubits in which for every $r-$block in B, there are at most k $(r-1)-$blocks such that the set A in these blocks is not $(r-1,k)$ sparse. An $(0,k)-$sparse set of qubits A is an empty set of qubits.*

Two density matrices ρ_1, ρ_2 of the set of qubits B are said to be (r,k)-deviated if there exists an (r,k)-sparse set of qubits $A \subseteq B$, with $\rho_1|_{B-A} = \rho_2|_{B-A}$. The deviation satisfies the triangle inequality since the union of two sets which are $(r,l_1),(r,l_2)$-sparse respectively is $(r, l_1 + l_2)$ sparse, by induction on r.

A computation is successful if the error at the end of each $r-$working period is sparse enough. The question is which faults keep the errors sparse. We define:

Definition 21 *The list of locations where faults had occurred, (in a specific run of the computation) is called a fault path.*

We will show in lemma 27 that the error is kept sparse if the fault path is sparse according to the following definition:

Definition 22 *A set of locations in an $r-$rectangle is said to be (r,k)-sparse if there are no more than k $(r-1)-$rectangles, in which the set is not $(r-1,k)$-sparse. An $(0,k)$-sparse set in an $0-$rectangles is an empty set. A fault path in M_r is (r,k)-sparse if in each $r-$rectangle, the set is $(r,k)-$sparse.*

7.5 The Good Part: Sparse Fault Paths Keep the Error Sparse

We claim that if the fault path is sparse enough, then the error corrections keep the deviation small. The number of faults allowed in one rectangle is bounded so that when taking into account the spread of the fault, the number of qubits effected in each block at the end of one working period is not too big, so that the density matrix can still be recovered.

Lemma 27 *Let C be a computation code that corrects d errors, with spread l. Let M_r be the $r-$simulation of M_0 by C. Consider a computation subjected to an (r,k)-sparse fault path with $kl \leq d$. At the end of each $r-$working period the error is (r,d)-sparse.*

Proof: It is instructive to first prove this lemma for $r = 1$. This is done by induction on the time t. For $t = 0$ the deviation is zero. Suppose that the density matrix at the end of the t'th working period is d-deviated from the correct matrix. If no errors occur during the t'th working period, the error corrections would have corrected the state to $\phi(\rho(t))$, and the procedures would have taken it to the correct state $\phi(\rho(t+1))$. However, due to the fact that k errors did occur in each rectangle, we have in each block at most kl qubits which are effected by this error, and the deviation is at most $kl \leq d$. This proves the theorem for $r = 1$. For general r, we prove two assertions together, using induction on r. The first assertion implies the desired result.

1. Consider n $r-$blocks, in a density matrix ρ_r which is $(r,d)-$deviated from $\phi^r(\rho_0)$, where ρ_0 is a density matrix of n qubits. At the end of an $r-$working period which $r-$simulates the operation g_0 on ρ_0, with an (r,k) sparse set of faults, the density matrix is (r,d) deviated from $\phi^r(g_0 \circ \rho_0)$.
2. Consider n $r-$blocks, in any density matrix, ρ_r. At the end of an r-working period which $r-$simulates the operation g_0 on ρ_0, with an (r,k) sparse set of faults, the density matrix is (r,d) deviated from some word $\phi^r(g_0 \circ \rho_0)$, where ρ_0 is a density matrix of n qubits.

For $r = 1$ the proof of the first assertion is as before, while the second assertion is true because of a similar argument, using the extra requirement that the error correction takes any word to a quantum code. Let us now assume both claims for r, and prove each of the claims for $r + 1$.

1. We consider an $(r + 1)-$working period operating on ρ_{r+1}, and $(r + 1)-$simulating the operation g_0 on ρ_0. Let us assume for a second two wrong assumptions. First, that all the $r-$rectangles in the $(r + 1)-$stage have (r, k) sparse set of faults. Second, that ρ_{r+1} is (r, d) deviated from $\phi^r(\rho_1)$, where ρ_1 is a density matrix which is $(1, d)-$deviated from $\phi(\rho_0)$. We can now use the induction assumptions.

The $(r + 1)-$simulation working period consists of two stages, one is the stage which $r-$simulates the error correction procedures in M_1, and the other part $r-$simulate one working period in M_1 which $1-$simulates the operation of g_0 in M_0. The first part $r-$simulates a computation in M_1, which, if no errors occur, is supposed to take ρ_1 to $\phi(\rho_0)$. We can use the induction assumption on the first assertion, to prove that at the end each one of the r-working periods in this $(r + 1)$-working period, the matrix is (r, d) deviated from what it is supposed to be. After w applications of this induction assumption, (where w is the number of time steps in the error correction procedure) we get that the matrix at the end of the error correction stage in the $r-$working period, is (r, d) deviated from $\phi^r(\phi(\rho_0)) = \phi^{r+1}(\rho_0)$. This is true under the two wrong assumptions.

Now, we release the second assumption. We actually start the computation with a matrix which is $(r + 1, d)-$deviated from $\phi^{r+1}(\rho_0)$. So most of the $r-$blocks are $(r, d)-$deviated from $\phi^{r+1}(\rho_0)$, except maybe d $r-$blocks in each $(r + 1)-$block which are problematic. By the induction stage on claim 2, after the first $r-$working period which r-simulates the operation of g_1 in M_1, the d problematic blocks in each $(r + 1)-$block are $(r, d)-$deviated from $\phi^r(g_1 \circ \rho_1')$. So after the first $r-$working period the density matrix is $(r, d)-$deviated from $\phi^r(g_1 \circ \rho_1)$, where ρ_1 is $(1, d)-$deviated from $\phi(\rho_0)$. Thus after the first $r-$working period we are in the same situation as if the second assumption holds, and we can proceed with the above argument.

We now consider the computation part of the $(r+1)-$working period. Still under the first wrong assumption, using the induction assumption on the first claim, we have that at the end of the $(r + 1)-$working period, the state is $(r, d)-$deviated from $\phi^{r+1}(g_0 \circ \rho_0)$. Thus under the first wrong assumption, the final density matrix is $(r, d)-$deviated from $\phi^{r+1}(g \circ \rho_0)$.

We now relax the first wrong assumption, and take into account the fact that there where k $r-$rectangles in each $(r + 1)$-rectangle where the faults where not $(r, k)-$sparse. By the fact that the $(r + 1)-$working period $r-$simulates a sequence of gates with spread l, these errors can effect only $kl \leq d$ $r-$blocks in each $(r+1)-$block, at the end of the $(r + 1)-$stage, so we have that the final density matrix at the end of the $(r + 1)-$stage is $(r + 1, d)-$deviated from the correct one, $\phi^{r+1}(\rho_0)$.

2. We consider one stage of $(r + 1)-$corrections on the n $(r + 1)-$blocks in an arbitrary density matrix. Again, let us assume that the faults in all the $r-$rectangles are $(r, k)-$sparse. By the induction stage on claim 2, after one $r-$working period, the density matrix is $(r, d)-$deviated from some $\phi^r(\rho_1)$. Let us consider the trajectory of $\phi^r(\rho_1)$ in the correction part of the working period. It is an $r-$simulation of an error correction procedure, which performs the computation which takes the density matrix ρ_1 to some word $\phi(\rho_0)$. As before, we can prove by induction on the w $r-$stages that at the end of the correction part of the $(r+1)$-working period, which is an $r-$simulation of the above error correction, we end up with a matrix which is $(r, d)-$deviated from $\phi^r(\phi(\rho_0))$. Taking into account the $r-$rectangles with faults which are not $(r, k)-$sparse, we end up with a

density matrix which is $(r+1,d)-$deviated from $\phi^{r+1}(\rho_0)$. $\Box$

Sparse errors are indeed "good", because if the error is sparse at the end of the computation, the majority of the bits will give the correct answer.

Lemma 28 *Let $2d+1 \leq m$. If the final density matrix of M_r is $(r,d)-$deviated from the correct one, then the distribution on the strings which are obtained when taking recursive majority on each $r-$block of its output is correct.*

Proof: Let ρ_0 be the correct final density matrix of M_0, Let ρ_r be the correct final density matrix of M_r, and let ρ'_r be the final density matrix which is $(r,d)-$deviated from ρ_r. First, we note that if we discard all the $r-$blocks of qubits which are not measured for output, the resulting two density matrices are still $(r,d)-$deviated. This remains true if we apply a measurement of all the qubits. Moreover, the output distribution remains the same after the measurements. We can thus assume that ρ_r and ρ'_r are density matrices of the output qubits, which are mixtures of basic states, and ρ'_r is $(r,d)-$deviated from ρ_r. ρ_r can thus be written as $\{p_i, |\alpha_i\rangle\}$, where $|\alpha_i\rangle = |i_1^{m^r} i_2^{m^r} ... i_n^{m^r}\rangle$. All pits in an $r-$block are equal, since the matrix is correct. The probability that the recursive majority string is i is exactly p_i. ρ'_r can be written as $\{q_j, |\beta_j\rangle\}$, where $|\beta_j\rangle$ are basis vectors. Let A be subset of qubits such that $\rho_r|_A = \rho'_r|_A$ and A contains all the qubits on which ρ_r operates on, except an $(r,d)-$sparse set. Note, that ρ'_r is supported only on $|\beta_j\rangle\}$ in which all the pits in A agree, and that

$$\sum_{j, \beta_j|_A = \alpha_i|_A} q_j = p_i. \tag{100}$$

because the reduced density matrix on A is the same as that of ρ_r. The probability that the recursive majority string is l is exactly the sum of probabilities of β_j which give the string l, but the only β_j that appear in the mixture agree on A, and we already know that the sum of the probabilities of those is correct. $\Box$

7.6 The Bad Part: Non-Sparse Fault Paths are Rare Below the Threshold

Let us first find the effective error rate in M_1. It is the probability for one rectangle to have more faults than the code can handle. If this effective error rate is smaller than the actual error rate η, this achieves an improvement in reliability from M_0 to M_1, and in fact from M_s to M_{s+1}. Let the code C correct d errors, and have spread l. Let A be the maximal number of locations in a rectangle. We require that the probability for more than $\lfloor d/l \rfloor = k$ faults in A locations is smaller than η. This is indeed the case if the following condition is satisfied.

Definition 23 *The threshold condition:*

$$\binom{A}{k+1} \eta^{k+1} < \eta \tag{101}$$

We can thus define the threshold of a code:

Definition 24 *Let C be a quantum computation code, which corrects d errors, has spread l, and where A is the maximal number of locations in a rectangle for simulations using this code. Let $k = \lfloor d/l \rfloor$. The threshold for probabilistic noise for the code C is*

$$\eta_c(C) = \begin{pmatrix} A \\ k+1 \end{pmatrix}^{-k}. \tag{102}$$

It is easy to see that any $\eta < \eta_c$ satisfies the threshold condition. The threshold can be computed given the parameters of the code. However, it is quite complicated to calculate the parameter A exactly. In our constructions we estimate the threshold to be $\approx 10^{-6}$ in both cases of CSS codes and polynomial codes, using codes of length $m = 7$, which can correct one error. This estimation is done assuming that measurements can be applied during the computation, classical operations in the procedures are error free, and this allows us to save some of the operations in the fault tolerant procedures, in a way which we do not describe here. Optimizations of the suggested procedures are certainly required and presumably can reduce this threshold by several orders of magnitudes.

We now show that below the threshold bad fault paths are rare.

Lemma 29 *If $\eta < \eta_c$, $\exists \delta > 0$ such that the probability $P(r)$, for the faults in an $r-$rectangle to be (r,k) sparse is larger than $1 - \eta^{(1+\delta)^r}$.*

Proof: Let δ be as follows.

$$\begin{pmatrix} A \\ k+1 \end{pmatrix} \eta^{k+1} < \eta^{1+\delta}. \tag{103}$$

Such δ exists for η below the threshold. The proof will follow by induction on r. The probability for an $0-$rectangle, i.e. one location, to have faults which are $(0,k)$ sparse, i.e. that in this location a fault did not occur, is $1-\eta$. Assume for r, and let us prove for $r+1$. For the faults in an $(r+1)$-rectangle not to be $(r+1,k)$ sparse, there must be at least $k+1$ $r-$rectangles in which the fault is not (r,k) sparse. So $P(r+1) \geq 1 - \begin{pmatrix} A \\ k+1 \end{pmatrix} (1-P(r))^{k+1} \geq 1 - \eta^{(1+\delta)^{r+1}}$, using the induction assumption, and the fact that $\eta^{(1+\delta)^r} < \eta_c$, so it satisfies the threshold condition. $\Box$

We can now prove the threshold theorem.

Theorem 21 The Threshold Theorem for Probabilistic Noise:

Let $\epsilon > 0$. Let C be a computation code with gates $\mathcal{G}$. There exists a threshold $\eta_c > 0$, and constants c_1, c_2, c_3 such that the following holds. Let Q be a quantum circuit, with n input qubits (qupits), which operates t time steps, uses s gates from $\mathcal{G}$, and has v locations. There exists a quantum circuit Q' which operates on $n\log^{c_1}(\frac{v}{\epsilon})$ qubits (qupits), for time $t\log^{c_2}(\frac{v}{\epsilon})$, and uses $s\log^{c_3}(\frac{v}{\epsilon})$ gates from $\mathcal{G}$ such that in the presence of probabilistic noise with error rate $\eta < \eta_c$, Q' computes a function which is ϵ-close to that computed by Q.

Proof: Q will be taken as M_0, and we generate M_r according to the above scheme, where r is chosen so that $v\eta^{(1-\delta)^r} < \epsilon$. By lemma 29, we have that the probability for a fault path to be bad is smaller than ϵ. By lemmas 27 and 28 the sparse fault paths give correct outputs. $\Box$

8 The Threshold Result for General Noise

We generalize the result to general local noise. The threshold condition is slightly different. Again, we fix a computation code and a set of gates $\mathcal{G}$. The proof consists of showing that the reliable circuit constructed for the case of probabilistic noise is robust against general noise, but the error rates which can be tolerated are slightly worth. Again, the proof consists of dealing with bad fault paths and good fault paths separately. The proof that good fault paths are indeed good uses a reduction to the case of the probabilistic noise. The proof that the bad part is negligible is more involved than in the probabilistic case.

8.1 Fault Paths in the Case of General Noise

The notion of fault paths is less clear in the case of general noise. To define fault paths, write the final density matrix of the noisy circuit as follows:

$$\rho(t) = \mathcal{E}(t) \cdot \mathcal{L}(t) \cdot \mathcal{E}(t-1) \cdot \mathcal{L}(t-1) \cdots \mathcal{E}(0) \cdot \mathcal{L}(0)\rho(0). \tag{104}$$

In the above equation, $\mathcal{E}(t)$ is the noise operator operating at time t, and $\mathcal{L}(t)$ is the computation operator at time t. According to our noise model, equation 38, $\mathcal{E}(t)$ can be written as a tensor product of operators, operating on the possible locations of faults at time t, $A_{i,t}$. Each such operator can be written as a sum of two operators, using equation 39:

$$\mathcal{E}_{A_{i,t}}(t) = (1-\eta)I + \mathcal{E}'_{A_{i,t}}(t) \quad , \quad \|\mathcal{E}'_{A_{i,t}}(t)\| \le 2\eta. \tag{105}$$

We can replace all the error operators in equation 104 by the products of operators of the form 105. We get:

$$\begin{aligned}\rho(t) \quad = \quad & \left(\otimes_{A_{i,t}}((1-\eta)I + \mathcal{E}'_{A_{i,t}})\right) \cdot \mathcal{L}(t) \cdot \left(\otimes_{A_{i,t-1}}((1-\eta)I + \mathcal{E}'_{A_{i,t-1}})\right) \cdot \mathcal{L}(t-1) \cdots \\ & \cdots \left(\otimes_{A_{i,0}}((1-\eta)I + \mathcal{E}'_{A_{i,t-1}})\right) \mathcal{L}(0)\rho(0).\end{aligned} \tag{106}$$

We can open up the brackets in the above expression. We get a sum of terms, where in each term, for each set of qubits $A_{i,t}$ at time t we either operate $(1-\eta)I$ or $\mathcal{E}'_{A_{i,t}}(t)$. Thus, each term in the sum corresponds to a certain fault path. More precisely, a fault path is a subset of the locations $\{A_{i,t}\}_{i,t}$, and the term in the sum which corresponds to this fault path is exactly the term in which we apply $\mathcal{E}'_{A_{i,t}}(t)$ on all locations in the fault path, and apply $(1-\eta)I$ on the rest. As was done in the probabilistic case, we can now divide the above sum 106 to two parts: the sum over the *good* fault paths, and the sum over the *bad* fault paths. We define the good fault paths to be those which are (r,k)-sparse, and the bad ones are all the rest. We write:

$$\rho(t) = \mathcal{L}_g \cdot \rho(0) + \mathcal{L}_b \cdot \rho(0) \tag{107}$$

We will treat each part separately.

8.2 The Bad Part: Non-Sparse Fault Paths are Negligible Below the Threshold

We show the trace norm of the *bad* part is negligible, when η is below the threshold for general noise.

Definition 25 *The threshold condition for general noise, $\eta'_c > 0$ is defined such that*

$$e \begin{pmatrix} A \\ k+1 \end{pmatrix} (2\eta)^{k+1} < 2\eta \tag{108}$$

Where as before, d is the number of errors which the code corrects, l is the spread, $k = \lfloor d/l \rfloor$ and A is the maximal number of locations in a rectangle. We can thus define the threshold of a code, for general noise:

Definition 26 *Let C be a quantum computation code, which corrects d errors, has spread l, and where A is the maximal number of locations in a rectangle for simulations using this code. Let $k = \lfloor d/l \rfloor$. The threshold for general noise for the code C is*

$$\eta'_c(C) = \frac{1}{2} e^{-k} \begin{pmatrix} A \\ k+1 \end{pmatrix}^{-k}. \tag{109}$$

It is easy to see that any $\eta < \eta'_c$ satisfies the threshold condition. Note that there is a slight difference from the threshold in the case of probabilistic noise. A factor of 2 is added to η, and the factor of e is added to the whole definition. These differences are due to slight technical difficulties which rise from the fact that the norm of the good operators in not smaller than 1, but can also be slightly larger than 1. η'_c will be smaller than the threshold for probabilistic noise, η_c.

Lemma 30 *Let $\eta < \eta'_c, \epsilon > 0$. Let M_0 use v locations. Let $r = \text{polyloglog}(\frac{v}{\epsilon})$. Then in M_r*

$$\|\mathcal{L}_b \cdot \rho(0)\| \leq \epsilon \quad , \quad \|\mathcal{L}_g \cdot \rho(0)\| \leq 1 + \epsilon \tag{110}$$

Proof: We shall rewrite the sum over all fault paths, by collecting together all the operations according to which $r-$procedures they were done in. We denote by $\mathcal{L}_b(i)$, the sum over all operators on the i'th procedure, when applying errors on bad fault paths. $\mathcal{L}_g(i)$ will thus be the sum over all operators on the i'th procedure, when applying errors on good fault paths. If there are v procedures, we can write:

$$\rho(t) = (\mathcal{L}_g(v) + \mathcal{L}_b(v)) \cdot (\mathcal{L}_g(v-1) + \mathcal{L}_b(v-1)) \cdots (\mathcal{L}_g(1) + \mathcal{L}_b(1))\rho(0) \tag{111}$$

We first prove that

$$\forall 1 \leq i \leq v \quad , \quad \|\mathcal{L}_b(i)\| \leq (2\eta)^{(1+\delta)^r} \quad , \quad \|\mathcal{L}_g(i)\| \leq 1 + (2\eta)^{(1+\delta)^r} \tag{112}$$

Where δ is defined by

$$e \begin{pmatrix} A \\ k+1 \end{pmatrix} (2\eta)^{k+1} < (2\eta)^{1+\delta}. \tag{113}$$

Such δ exists because η is below the threshold. The proof of inequality 112 for $\eta < \eta'_c$, follows the lines of lemma 29. We use induction on r, and denote by $\mathcal{L}_b^r(i)$ the sum over all bad fault paths in an $r-$procedure. For $r = 0$, a 0-procedure is simply one location, and $\mathcal{L}_b^0(i)$ is the sum over all *bad* fault paths in this procedure. For one location, there is only one term in this sum: the identity, or the gate applied in this location, followed by one noise operator. By equation 105, and the properties of the norm on super-operators in section 8 $\|\mathcal{L}_b^0(i)\| \leq 2\eta$, and $\|\mathcal{L}_g^0(i)\| \leq 1 + 2\eta$ Now assume for r and prove for $r+1$. For the faults in an $(r+1)$-rectangle *not* to be

$(r+1,k)$ sparse, there must be at least $k+1$ $r-$rectangles in which the fault is not (r,k) sparse. So by the induction assumption on both $\mathcal{L}_b(i)$ and $\mathcal{L}_g(i)$

$$\|\mathcal{L}_b^{r+1}(i)\| \leq \left(\begin{array}{c} A \\ k+1 \end{array}\right)((2\eta)^{(1+\delta)^r})^{k+1}(1+(2\eta)^{(1+\delta)^r})^{A-k-1} \leq e\left(\begin{array}{c} A \\ k+1 \end{array}\right)((2\eta)^{(1+\delta)^r})^{k+1}. \tag{114}$$

where we have used the fact $(1+(2\eta)^{(1+\delta)^r})^{A-k-1} < e$ since $(2\eta)^{(1+\delta)^r} < 2\eta$, and $2\eta A \leq 1$. The right hand side is $\leq (2\eta)^{(1+\delta)^{r+1}}$ using the fact that $(2\eta)^{(1+\delta)^r} < 2\eta$, and the threshold condition is satisfied for $\eta \leq \eta_c'$. This proves the induction step for $\mathcal{L}_b(i)$. Using $\|\mathcal{L}_g(i)+\mathcal{L}_b(i)\|=1$ proves the induction step also for $\mathcal{L}_g(i)$. To prove the statement, we consider bad fault paths, i.e. at least one $r-$rectangle is bad. If there are v rectangles, we have:

$$\|\mathcal{L}_b\| \leq v \cdot (1+(2\eta)^{(1+\delta)^r})^{v-1}(2\eta)^{(1+\delta)^r} \tag{115}$$

taking $r = \text{polyloglog}(\frac{v}{\epsilon})$ we get the desired result. □

8.3 The Good Part: Sparse Fault Paths Give Almost Correct Outputs

Lemma 31 *Let $\eta \leq \eta_c', \epsilon > 0$. Let M_0 use v locations. Let $r = \text{polyloglog}(\frac{v}{\epsilon})$, as in lemma 30. Let M_0 output $f(i)$ for a given i with probability of error ϵ'. Then for M_r,*

$$\sum_{bad\ j} [\mathcal{L}_g \cdot \rho(0)]_{j,j} \leq (1+\epsilon)\epsilon' \tag{116}$$

where bad j's are those basis states with the majority of the result qubits not equal to $f(i)$.

Proof: For any density matrix ρ which is $(r,d)-$deviated from the correct final density matrix of M_r we have $\sum_{bad\ i} \rho_{i,i} = \epsilon$. We will write $\mathcal{L}_g$ as a linear sum (not necessarily positive) of sparse physical fault paths, for which lemma 27 can be applied. We will get the desired result from linearity. We write

$$\mathcal{L}_g \cdot \rho(0) = \sum_E \mathcal{L}_E \cdot \rho(0) \tag{117}$$

where E runs over all general sparse fault paths, and $\mathcal{L}_E$ is the operator corresponding to the computation done in the presence of the general fault path E. Now each general fault in E is a linear sum of physical operators:

$$\mathcal{E}'_{A_{i,t}} = \mathcal{E}_{A_{i,t}} - (1-\eta)I \tag{118}$$

Inserting this to equation 117, and we get a linear sum over terms which correspond to physical fault paths, with sparse faults.

$$\mathcal{L}_g \cdot \rho(0) = \sum_f \lambda_f \mathcal{L}_f \cdot \rho(0) \tag{119}$$

Lemma 27 applies to each term in the sum. We get that each density matrix $\mathcal{L}_E \cdot \rho(0)$ is $(r,d)-$deviated from correct. This means that

$$\sum_{bad\ i} [\mathcal{L}_g \cdot \rho(0)]_{i,i} = \sum_f \lambda_f \sum_{bad\ i} [\mathcal{L}_f \cdot \rho(0)]_{i,i} = (\sum_f \lambda_f)\epsilon' = \text{Tr}(\mathcal{L}_g \cdot \rho(0))\epsilon' \leq \|\mathcal{L}_g\|\epsilon \leq (1+\epsilon)\epsilon', \tag{120}$$

using lemma 30 in the last inequality. □

8.4 The Threshold Theorem for General Noise

We can now prove the threshold result for general noise:

Theorem 22 The Threshold Theorem for General Noise:

Let $\epsilon > 0$. Let C be a computation code with gates $\mathcal{G}$. There exists a threshold $\eta_c' > 0$, and constants c_1, c_2, c_3 such that the following holds. Let Q be a quantum circuit, with n input qubits, which operates t time steps, uses s gates from $\mathcal{G}$, and has v locations. There exists a quantum circuit Q' which operates on $n\log^{c_1}(\frac{v}{\epsilon})$ qubits, for time $t\log^{c_2}(\frac{v}{\epsilon})$, and uses $s\log^{c_3}(\frac{v}{\epsilon})$ gates from $\mathcal{G}$ such that in the presence of general noise with error rate $\eta < \eta_c'$, Q' computes a function which is ϵ-close to that computed by Q. $\Box$.

Proof: Let Q compute f with accuracy ϵ', meaning that for any input i, the output is $f(i)$ with probability at least $1-\epsilon'$. We construct Q' which computes f with accuracy $\epsilon+\epsilon'$. Q' will be the $r-$simulation of Q, where r, is chosen such that the requirements of lemma 30 are satisfied with $\epsilon'' = \epsilon/(1+\epsilon')$. We write the probability to measure a bad basis state as the sum of the diagonal elements of the density matrix $\rho(t)$ corresponding to bad states.

$$\sum_{bad\ i} \rho(t)_{i,i} = \sum_{bad\ i} [\mathcal{L}_b \cdot \rho(0)]_{i,i} + \sum_{bad\ i} [\mathcal{L}_g \cdot \rho(0)]_{i,i} \tag{121}$$

And hence

$$|\sum_{bad\ i} \rho(t)_{i,i}| \le \sum_{bad\ i} |[\mathcal{L}_b \cdot \rho(0)]_{i,i}| + |\sum_{bad\ i} [\mathcal{L}_g \cdot \rho(0)]_{i,i}| \le \|\mathcal{L}_b \cdot \rho(0)\| + |\sum_{bad\ i} [\mathcal{L}_g \cdot \rho(0)]_{i,i}| \le \epsilon'' + (1+\epsilon'')\epsilon' \tag{122}$$

The first inequality follows from the fact that $\sum_i |\rho_{i,i}| \le \|\rho\|$ for any Hermitian matrix (see [10]. Lemmas 30 and 31 are used to derive the second inequality. Due to the choice of ϵ" This probability is indeed smaller than $\le \epsilon + \epsilon'$. $\Box$

9 Fault Tolerance with Any Universal Set of Gates

So Far, the reliable circuits which we have constructed can use only universal set of gates associated with a quantum computation code, such as the sets $\mathcal{G}_1$ and$\mathcal{G}_2$. This is an undesirable situation, both theoretically and practically. Theoretically, we would like to be able to show that the fault tolerance result is robust, meaning that quantum computation can be performed fault tolerantly regardless of the universal set of gates which we use. Practically, it is likely that the sets of gates $\mathcal{G}_1$ or $\mathcal{G}_2$ are difficult to implement in the laboratory, since they contain gates involving three qubits, and we would like to be able to use other, perhaps simpler, universal sets of gates. Indeed, in this chapter we provide the desired generalization, and show that the threshold result holds for any universal set of gates $\mathcal{G}$. We require, however, that $\mathcal{G}$ contains a gate which discards a qubit, and a gate which adds a blank qubit to the circuit.

Let $\mathcal{G}$ be a set of gates of a computation code, $\mathcal{G}'$ be a set of gates to be used for computation. After constructing the reliable circuit using $\mathcal{G}$, we will approximate each gate by gates from $\mathcal{G}'$, such that the sum of the errors of each gate is smaller than the allowed error. The threshold is of course worse, and will depend on the set of gates $\mathcal{G}'$.

Definition 27 *The Threshold condition for General unitary gates.*

Let $S(\delta)$ be the number of gates required to approximate a gate from $\mathcal{G}$ by gates from $\mathcal{G}'$ to accuracy δ. For any $\delta < \eta_c'$, we define η_c'' to be

$$\eta_c'' = \max_{\delta} \frac{\eta_c' - \delta}{S(\delta)} \tag{123}$$

$\eta < \eta_c''$ guarantees that $S(\delta_0)\eta + \delta_0 < \eta_c'$, thence the total error of all gates approximating one gate of the computation code will not exceed η_c', so the fault tolerance scheme for general noise will apply.

Theorem 23 The Threshold Result in Full Generality:

Let $\epsilon > 0$. Let $\mathcal{G}', \mathcal{G}''$ be two universal sets of quantum gates. There exists a threshold $\eta_c'' > 0$, and constants c_1, c_2, c_3, c_4 such that the following holds. Let Q' be a quantum circuit, with n input qubits, which operates t time steps, uses s gates from $\mathcal{G}'$, and has v locations. There exists a quantum circuit Q'' which operates on $c_1 n\log^{c_2}(\frac{v}{\epsilon})$ qubits, for time $c_3 t\log^{c_2}(\frac{v}{\epsilon})$, and uses $c_4 s\log^{c_2}(\frac{v}{\epsilon})$ gates from $\mathcal{G}''$ such that in the presence of general noise with error rate $\eta < \eta_c''$, Q'' computes a function which is ϵ-close to that computed by Q'. $\square$.

Proof: We first approximate Q' by a circuit M_0 which uses only gates from the a set of gates $\mathcal{G}$ of a computation code. Due to the Kitaev-Solovay theorem, this approximation can be done, with polylogarithmic cost of space and time, to an arbitrary accuracy. We will require that each gate is approximated up to $1/T$, so that M_0 computes a function which is $\epsilon/2$ close to that computed by Q'. We then construct M_r, the $r-$simulation of M_0, which again uses gates from $\mathcal{G}$. r is chosen such that the general threshold scheme will give an $\epsilon/2$ error when the error rate in is taken to be $\eta_e = S(\delta_0)\eta + \delta_0$. To construct Q'', we replace each gate in M_r by $S(\delta_0)$ gates from $\mathcal{G}''$, up to δ_0. $\square$

10 Fault Tolerance in a d-Dimensional Quantum Computer

So far, we allowed a gate to operate on any set of qubits, regardless on the actual location of these qubits in space. We call this case the no-geometry case. We consider also another case, in which the qubits are located a one array, and the gates can be applied only on nearest neighbor qubits. For more dimensions, the gates are restricted to operate on qubits which are nearest neighbors and lie in the same line. We show that the threshold result holds in full generality for $d-$dimensional quantum computers, for any $d \geq 1$. Note that for the geometry of the circuit to be preserved, we require that the number of qubits is preserved throughout the computation. Hence discarding and adding qubits are not allowed. However, we still need a gate which allows to put entropy out of the system. This will be the RESTART gate, which is constructed by discarding a qubit and then adding a blank qubit instead of the discarded qubit. We also need the SWAP gate, which switches between two qubit, i.e. it applies the unitary transformation $|i\rangle|j\rangle \leftrightarrow |j\rangle|i\rangle$. We thus require that the universal set of gates which is used in the final circuit, $\mathcal{G}''$, contains the SWAP and the RESTART gates.

Theorem 24 Threshold theorem for d-dimensional circuits::

Let $\epsilon > 0$. Let $d \geq 1$. Let $\mathcal{G}', \mathcal{G}''$ be two universal sets of quantum gates. There exists a threshold $\eta_c'' > 0$, and constants c_1, c_2, c_3, c_4 such that the following holds. Let Q' be a d-dimensional quantum circuit, with n input *qubits, which operates t time steps, uses s gates from $\mathcal{G}'$, and has v locations. There exists a d-dimensional quantum circuit Q'' which operates on $c_1 n\log^{c_2}(\frac{v}{\epsilon})$ qubits, for time $c_3 t\log^{c_2}(\frac{v}{\epsilon})$, and uses $c_4 s\log^{c_2}(\frac{v}{\epsilon})$ gates from $\mathcal{G}'' \cup \{SWAP\}$ such that in the presence of general noise with error rate $\eta < \eta_c''$, Q'' computes a function which is ϵ-close to that computed by Q'.*

Proof: The only difference from the proof of the threshold theorem for circuits without geometry is that the fault tolerant procedures need to apply gates on nearest neighbors only, and that all the qubits remain in the circuit throughout the computation. Here is how one level of the simulation is done. We first pick a preferred direction, and each qubit will be extended to an array of qubits lying in that direction. The simulation will "stretch" the simulated circuit only in the preferred direction, by a constant factor. Let a be the maximal number of ancilla qubits used in all the procedures of the computation code $\mathcal{G}$. Let m be the size of the block. A qubit in Q_1 will be replaced by $m+a$ qubits, placed in a line along the preferred direction. The ancilla qubits will serve as working space, but we will also SWAP qubits with ancilla qubits, in order to bring computation qubits closer and operate gates on them. The fault tolerant procedures are modified as follows. First, instead of adding ancilla qubits during the procedure, we only use the ancilla qubits that are already there, and apply a RESTART gate on an ancilla qubit one step before we use it in the procedure. Also, any gate g in the procedure, which operates on qubits which are far apart, is replaced by a sequence of SWAP gates which bring the qubits g operates on to nearest neighbor sites, followed by g, which is then followed by another sequence of SWAP gates to bring the qubits back to their original sites. Since the simulated circuit applies gates on nearest neighbors, say on k qubits in a raw, the number of SWAP gates required is at most $2(k+1)m$, i.e. a constant.

The claim is that the procedure is still fault tolerant. This might seem strange since the SWAP gates operate on many qubits, and seem to help in propagation of errors. However, note that a SWAP gate which operates on a faulty qubit and an unaffected qubit does not propagate the error to the two qubits, but keeps it confined to the original qubit, which is now in a new site. Hence, a SWAP gate which is not faulty does not cause propagation of error. If an error occurred in a SWAP gate, then the two qubits participating in it are contaminated. This will cause contamination of all the qubits on which g operates on, (in the worst case), so an error in a SWAP gate is equivalent to an error in all the original sites of the qubits participating in the gate, and the final site of the other qubit participating in the SWAP gate. This adds a factor of 2 to the original spread of the procedure. All other aspects of the theorem remain the same. $\Box$.

Chapter V

Conclusions and Open Problems

We cannot foresee which goals will be achieved, if quantum computers be the next step in the evolution of computation[128]. We are still very far from understanding the limitations of quantum computers as computation devices; The field of quantum complexity is still in its infancy. The main lesson from this thesis is that noisy quantum computers have almost the same computational complexity power as error free quantum computers, as long as the error rate is smaller than a certain threshold. We did not attempt to optimize the threshold in the fault tolerant scheme, but merely to prove its existence. However, a rough estimation of the threshold was given in the case in which the length of the code is $m = 7$, in both general CSS and polynomial codes, in a circuit with no geometry, and with noiseless classical operations and measurements allowed. The threshold then is estimated to be $\approx 10^{-6}$. After the threshold result was achieved, several papers generalizing and improving this result were published, and the current estimations of the threshold are between $\approx 10^{-4}$ to $\approx 10^{-2}$. This threshold is not so far from being practical. I hope that the threshold result will motivate physicists to achieve lower noise rates in their experimental implementations, and theoreticians to seek quantum computation codes with better parameters, in order to push the threshold as high as possible. The experimental difficulties of constructing a quantum computer are not only due to noise, but mainly due to the difficulty in putting a large number of qubits together and being able to control them. However, noise is no longer viewed as an obstacle to realization of quantum computers. The practical implications of realizing quantum computers physically are very hard to anticipate, but there is no doubt that they will change the way we think of computation.

A very important question is whether the assumptions used in the threshold result, of local noise, and Markovian environment, can be relaxed. An interesting direction to pursue is to consider different assumptions on the noise, such as very strong correlations in time and space between the noise process. Optimistic results in this direction were found by Lidar, Chuang and Whaley[152]. Dealing with non-local noise is an open and challenging problem. This is especially true since despite the optimistic message of the threshold result, quantum computers are still a purely theoretical object. Hence, a question still hangs over our heads: Is the assumption of local noise realistic? We will know for sure only when quantum computations are realized.

The threshold result raises a few open questions. Our scheme requires a polylogarithmic blow-up in the depth of the circuit. It might be possible to use a quantum analogue of multi linear codes[197], to reduce the multiplicative factor of $O(log(n))$ to a factor of $O(log(log(n))$. An open question is whether it is possible to reduce the time cost to a constant, as in the classical case. I conjecture that the answer is negative, due to information theoretical intuition, but was unable yet to prove this conjecture.

Let us confront the results concerning strong noise with those concerning weak noise. First, it was shown that if the noise is stronger than a certain threshold, the quantum computer is not stronger computationally than a classical system. This shows that there is a non trivial regime of noise in which quantum computers cannot perform tasks which are intractable for classical systems, in spite of the fact that their evolution is still quantum. An upper bound was given on the noise which any quantum computer can hope to tolerate. Thus we have an upper bound on the possible threshold for fault tolerance (assuming quantum computers are not polynomially equivalent to classical computers.) This upper bound, 0.97, is very far from the known threshold, 10^{-6}. Making these two bounds tight is a challenging task. One possible way to do this is to find a stronger simulation than the one suggested here, that is efficient for lower error rates.

The results for strong noise also point out several issues which should be thought of when constructing fault tolerant quantum computers. It was shown that any quantum computer can be simulated efficiently by classical computers, if it has limited amount of parallelism, where by parallelism I count only gates which operate on more than one qubit. This should be taken into account when planning implementations of quantum gates by low-fan-in gates[95, 20]. It might also be interesting to quantify, according to these observations, the stability of different algorithms against noise. At any rate, an inevitable conclusion is that in order to build fault tolerant quantum computers, a linear number of two-qubits gates should be applied every time step. Thus, the results presented here indicate important restrictions on possible realizations of quantum circuits.

Hand in hand with the practical and technical issues which are solved in this thesis, arise deep fundamental questions about quantum physics. Throughout this thesis, I tried to draw a strong connection between the amount of noise in the quantum computer, and its computational power compared to the classical computational power. I think that the results viewed in this way might have an important impact on a long standing question in quantum physics, regarding the transition from quantum to classical physics[224]. Traditionally, this transition is treated by taking the limit of Planck's constant to 0, or explained with decoherence, but is always viewed as a gradual transition (but see [132]). It is interesting to consider a different point of view, in which the definition of quantum versus classical behavior is computational. The computational power of all classical systems seem to be equivalent, whereas quantum complexity seems inherently different. Suppose that indeed quantum behavior cannot be simulated efficiently by classical systems, an idea suggested by Feynmann[102] which originated quantum computation. In other words, suppose $BPP \neq BQP$. Then, increasing the noise, a transition from the quantum computational behavior to classical computational behavior occurs. The "phase diagram" of quantum versus classical behavior can be viewed as follows:

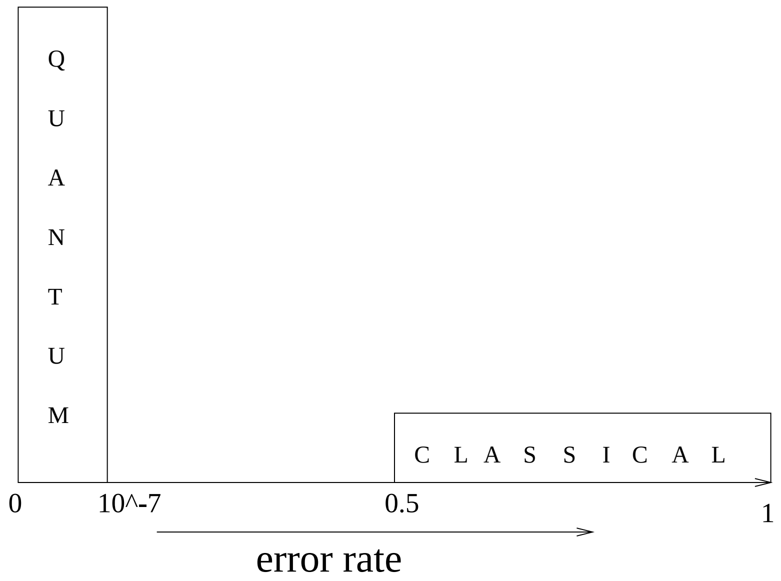


Figure V.1: In this figure, we see the computational behavior as a function of the error rate in the quantum circuit. The phase diagram is drawn for one dimensional circuits, but a similar diagram with different values of the error rate can be drawn for general circuits.

Changing the strength of the noise, the system transforms from quantum behavior to classical behavior. A fundamental question is what is the physical origin to the difference between the two regimes, which reflects itself in the difference between the computational powers. A hint for the answer is given in the analysis of the evolution of the clusters in the simulation of noisy quantum computers described in chapter III. The cluster analysis tells us that we should expect that the two phases are characterized by some *scale of quantum entanglement*, i.e. the typical number of qubits sharing entanglement. In the quantum regime this scale is macroscopic, and quantum computation is possible. On the right, "classical" regime, entanglement is confined to microscopic clusters. A very interesting question is to define a physical parameter which captures this notion. This would enable us to answer the following fundamental question: does the transition between the two regimes occur at a critical error rate η, or is it gradual. In chapter III there are strong indications to the fact that the transition is abrupt, and has many characteristics of a **phase transition** (and see also [132].) This transition clearly has a very strong connection to the old problem of understanding the transition from quantum to classical behavior. I conjecture that the transition from macroscopic quantum behavior to macroscopic classical behavior occurs as a phase transition, though a definition of the correct order parameter is still lacking. The idea that the transition from quantum to classical physics is abrupt stands in contrast to the standard view of a gradual transition due to decoherence[224]. I believe that the flippant frontier between quantum and classical physics will be better understood if we gain better understanding of the transition from quantum to classical computational behavior, and I view this connection between quantum complexity and quantum physics as extremely interesting.

The threshold result holds also in the case of circuits which allow to operate only on nearest neighbors. An interesting conclusion of this is that one dimensional quantum systems can exhibit a non trivial phase transition at a critical point η_c. The existence of a one dimensional phase transition is interesting because one dimensional phase transitions are rare, also in classical systems, though there exist several complicated examples[166, 109].

Perhaps a vague, but deeper, and more thought provoking question is that of the postulates of quantum mechanics. The possibility that the model will be realized will enable a thorough test of some of the more philosophical aspects of quantum theory, such as understanding the collapse of the wave function, the process of measurement, and other elements which are used as everyday tools in quantum algorithms. According to our current understanding, there are no theoretical obstacles to realizing quantum computers. But, it might be that the realization of quantum computation will reveal the fact that what we understand in quantum physics is merely an approximation holding only for small number of particles, which we extrapolated to many particles. Such questions are appealing motivations for this extremely challenging task of realizing the quantum computation model physically. It seems that successes, and also failures, in achieving this ambitious task, will open new exciting paths and possibilities in both computer science and fundamental physics.

Bibliography

[1] Aizenman M and Barsky D J, Sharpness of the phase transition in percolation models, *Commun. Math. Phys* **108** 489-526, 1987

[2] Abrams D S and Lloyd S, Simulation of Many-Body Fermi Systems on a Universal Quantum Computer, (quant-ph/9703054) *Phys.Rev.Lett.* **79** 2586–2589, 1997

[3] Abrams D S and Lloyd S, Non-Linear Quantum Mechanics implies Polynomial Time solution for NP-complete and #P problems, (quant-ph/9801041)

[4] Aizenman M, Chayes J T, Chayes L, Frohlich J and Russo L, On a sharp transition from area law to perimater law in a system of random surfaces, *Commun. Math. Phys* **92** 19-69, 1983

[5] Adleman L, Demarrais J and Huang M-D, Quantum Computability, *SIAM Journal of Computation* **26** 5 pp 1524–1540 October, 1997

[6] Adleman L, Molecular computation of solutions to combinatorial problems, *Science*, 266, 1021–1024, Nov. 11, 1994

[7] Aharonov D, Quantum Computation, In *annual reviews of computational physics*, Vol VI, ed. Dietrich Stauffer, World Scientific, 1999.

[8] Aharonov D and Ben-Or M, Fault-Tolerant Quantum Computation with Constant Error, (preprint quant-ph/9611025), *Proc. of the 29th Annual ACM Symposium on Theory of Computing (STOC)* 1997.

The journal version of this paper, Fault-Tolerant Quantum Computation with Constant Error Rate, (preprint quant-ph/9906129), submitted to SIAM journal of computation.

[9] Aharonov D and Ben-Or M, Polynomial Simulations of Decohered Quantum Computers (quant-ph/9611029), *37th Annual Symposium on Foundations of Computer Science* (FOCS) pp 46–55, 1996

[10] Aharonov D, Kitaev A Yu and Nisan N, Quantum Circuits with Mixed States, *Proc. of the 30th Annual ACM Symposium on Theory of Computing (STOC)* 1998

[11] Aharonov D, Ben-Or M, Impagliazo R and Nisan N, Limitations of Noisy Reversible Computation, (quant-ph/9611028) 1996

[12] Aharonov D, Beckman D, Chuang I and Nielsen M, What Makes Quantum Computers Powerful?

[13] Aharonov D, Two Phases of Entanglement in Noisy Quantum Computers, in preparation

[14] Alon N and Spencer J, The Probabilistic Method, Wiley-interscience, 1992

[15] Aspect A, Dalibard J and Roger G, Experimental test of Bell's inequalities using time-varying analyzers, *Phys. Rev. Lett.* **49**, 1804–1807, 1982

[16] Aspect A, Testing Bell's inequalities, *Europhys. News.* **22**, 73–75, 1991

[17] Barenco A A universal two-bit gate for quantum computation, *Proc. R. Soc. Lond. A* **449** 679–683, 1995

[18] Barenco A and Ekert A K Dense coding based on quantum entanglement, *J. Mod. Opt.* **42** 1253–1259, 1995

[19] Barenco A, Deutsch D, Ekert E and Jozsa R, Conditional quantum dynamics and quantum gates, *Phys. Rev. Lett.* **74** 4083–4086, 1995

[20] Barenco A, Bennett C H, Cleve R, DiVincenzo D P, Margolus N, Shor P, Sleator T, Smolin J A and Weinfurter H, Elementary gates for quantum computation, *Phys. Rev. A* **52**, 3457–3467, 1995

[21] Barenco A, Ekert A, Suominen K A and Torma P, Approximate quantum Fourier transform and decoherence, *Phys. Rev. A* **54**, 139–146, 1996

[22] Barenco A, Berthiaume A, Deutsch D, Ekert A, Jozsa R, and Macchiavello C, Stabilization of Quantum Computations by Symmetrization, (quant-ph/9604028) *SIAM J. Comp.***26**,5, 1541–1557, 1997

[23] Barenco A, Brun T A, Schak R and Spiller T P, Effects of noise on quantum error correction algorithms, *Phys. Rev. A* **56** 1177–1188, 1997

[24] Barnum H, Caves C, Fuchs C A, Jozsa R and Schumacher B, Non commuting mixed states cannot be broadcast, (quant-ph/9511010) *Phys. Rev. Lett.***76** 2818–2822, 1996

[25] Barnum H, Nielsen M and Schumacher B, (quant-ph/9702049)

[26] Beals R, Quantum computation of Fourier transform over symmetric groups *Proc. of the 29th Annual ACM Symposium on Theory of Computing (STOC)* 1997

[27] Beals R, Buhrman H, Cleve R, Mosca M and de Wolf R, Quantum Lower Bounds by Polynomials, (quant-ph/9802049) 1998

[28] Beckman D, Chari A, Devabhaktuni S and Preskill J Efficient networks for quantum factoring, *Phys. Rev. A* **54**, 1034–1063, 1996

[29] Bell J S On the Einstein-Podolsky-Rosen paradox, *Physics* **1** 195–200, 1964

[30] Bell J S On the problem of hidden variables in quantum theory, Rev. Mod. Phys. **38** 447-52, 1966 *Speakable and unspeakable in quantum mechanics* 1987 (Cambridge University Press)

[31] Benioff P, The Computer as a Physical Systems: A Microscopic Quantum Mechanical Hamiltonian Model of Computers as Represented by Turing Machines, *J. Stat. Phys.* **22** 563–591, 1980

[32] Benioff P Quantum mechanical Hamiltonian models of Turing machines, *J. Stat. Phys.* **29** 515-546 1982

[33] Bennett C H, Logical reversibility of computation, *IBM J. Res. Develop.* **17** 525–532, 1973

[34] Bennett C H, The Thermodynamics of Computation - a Review, *International Journal of Theoretical Physics,* **21**, No. 12, p 905, 1982

[35] Bennett C H, Demons, engines and the second law, *Scientific American* **257** no. 5 (November) pp 88–96, 1987

[36] C.H. Bennett, Time/Space Trade-offs for Reversible Computation, *SIAM Journal of Computation*, **18**, 4, pp 766–776, 1989

[37] Bennett C H, Bessette F, Brassard G, Savail L and Smolin J Experimental quantum cryptography, *J. Cryptology* **5**, pp 3–28, 1992

[38] Bennett C H, Brassard G, Crépeau C, Jozsa R, Peres A and Wootters W K Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels, *Phys. Rev. Lett.* **70** 1895–1898, 1993

[39] Bennett C H, DiVincenzo D P, Smolin J A and Wootters W K Mixed state entanglement and quantum error correction, (quant-ph/9604024) *Phys. Rev. A* **54** 3825, 1996

[40] Bennett C H, Bernstein E, Brassard G and Vazirani U Strengths and Weaknesses of quantum computing, (preprint quant-ph/9701001) *SIAM Journal of Computation* **26** 5 pp 1510–1523 October, 1997

[41] Berman G P, Doolen G D, Holm D D, Tsifrinovich V I Quantum computer on a class of one-dimensional Ising systems, *Phys. Lett.* **193** 444-450 1994

[42] Bernstein E and Vazirani U, 1993, Quantum complexity theory, *SIAM Journal of Computation* **26** 5 pp 1411–1473 October, 1997

[43] Berthiaume A, Deutsch D and Jozsa R, The stabilization of quantum computation, in *Proceedings of the Workshop on Physics and Computation, PhysComp 94* 60-62 Los Alamitos: IEEE Computer Society Press, 1994

[44] Berthiaume A and Brassard G, The quantum challenge to structural complexity theory, in *Proc. of the Seventh Annual Structure in Complexity Theory Conference* 1992 (IEEE Computer Society Press, Los Alamitos, CA) 132–137

[45] Berthiaume A and Brassard G, Oracle quantum computing, in *Proc. of the Workshop on Physics of Computation: PhysComp '92* (IEEE Computer Society Press, Los Alamitos, CA) 60–62, 1992

[46] Ben-Or M, Goldwasser S, and Wigderson A. Completeness theorems for fault-tolerant distributed computing, In Proc. for the 20th ACM Symp. on Theory of Computing, (STOC), Chicago, 1988, pp. 1-10.

[47] Boghosian B M and Taylor W, Simulating quantum mechanics on a quantum computer (preprint quant-ph/9701019) in *Proc. of the Workshop on Physics of Computation: PhysComp,* 1996

[48] Bouwmeester D, Pan J-W, Mattle K, Weinfurter H, Zeilinger A, Experimental quantum teleportation *Nature* **390**, 575-579 1997

[49] Boyer M, Brassard G, Hoyer P and Tapp A, Tight bounds on quantum searching, (preprint quant-ph/9605034) 1996

[50] P. Oscar Boykin, Tal Mor, Matthew Pulver, Vwani Roychowdhury, and Farrokh Vatan, "On Universal and Fault-Tolerant Quantum Computing", in preparation.

[51] Brassard G, Searching a quantum phone book, **Science 275** 627-628 1997

[52] Brassard G and Crepeau C, *SIGACT News* **27** 13-24 1996

[53] Brassard G, Hoyer P and Tapp A, Quantum Algorithm for the Collision Problem (quant-ph/9705002) 1997

[54] Brassard G and Hoyer P, An Exact Quantum-Polynomail Algorithm for Simon's Problem (quant-ph/9704027) *Proceedings of the 5th Israeli Symposium on Theory of Computing and Systems (ISTCS)*, 1997

[55] Brassard G, Teleportation as Quantum Computation, (quant-ph/9605035) 1996

[56] Brassard G, The dawn of a new era for quantum cryptography: The experimental prototype is working!, Sigact News, **20**(4), 78–82, 1989

[57] Buhrman H, Cleve R and van Dam W, Quantum Entanglement and Communication Complexity, (quant-ph/9705033) 1997

[58] Buhrman H, Cleve R and Wigederson A, Quantum vs. Classical Communication and Computation, (quant-ph/9802040) *Proc. of the 30th Annual ACM Symposium on Theory of Computing (STOC)* 1998

[59] Calderbank A R and Shor P W, Good quantum error-correcting codes exist, (quant-ph/9512032) *Phys. Rev. A* **54** 1098-1105, 1996

[60] Calderbank A R, Rains E M, Shor P W and Sloane N J A Quantum error correction and orthogonal geometry, *Phys. Rev. Lett.* **78** 405–408, 1997

[61] Calderbank A R, Rains E M, Shor P W and Sloane N J A, Quantum error correction via codes over $GF(4)$ (preprint quant-ph/9608006) 1996

[62] Chayes J T, Puha A and Sweet T, Independent and dependent percolation, Notes for a course, unpublished.

[63] Chernoff. See Feller W, An Introduction to Probability Theory and Its Applications, Wiley, New York,1957

[64] Choi M. D, linear algebra and its applications, **10** 286 (1975),

[65] Chuang I L, Laflamme R, Shor P W and Zurek W H, Quantum computers, factoring, and decoherence, *Science* **270** 1633–1635, 1995

[66] Chuang I L, Laflamme R and Paz J P, Effects of Loss and Decoherence on a Simple Quantum Computer, (quant-ph/9602018) 1996

[67] I.Chuang and W.C.D. Leung and Y. Yamamoto, Bosonic Quantum Codes for Amplitude Damping, (quant-ph/9610043), 1996

[68] Chuang I L and Yamamoto Creation of a persistent qubit using error correction Phys. Rev. A **55**, 114–127, 1997

[69] Chuang I L ,Vandersypen L M K, Zhou X, Leung D W and Lloyd S, Experimental realization of a quantum algorithm, (quant-ph/9801037) 1998

[70] Church A An unsolvable problem of elementary number theory, *Amer. J. Math.* **58** 345–363, 1936

[71] Cirac I J and Zoller P Quantum computations with cold trapped ions, *Phys. Rev. Let.*, 74: 4091-4094, 1995.

[72] Cirac J I, Pellizari T and Zoller P, Enforcing coherent evolution in dissipative quantum dynamics, *Science* **273**, 1207, 1996

[73] Cirac J I, Zoller P, Kimble H J and Mabuchi H Quantum state transfer and entanglement distribution among distant nodes of a quantum network, *Phys. Rev. Lett.* **78**, 3221, 1997

[74] Cirel'son (Tsirelson) B, Reliable storage of information in a system of unreliable components with local interactions. *Lecture notes in Mathematics***653** 15–30 ,1978

[75] Clausen M, Fast Generalized Fourier transforms, *Theoret. Comput. Sci.* **56** 55–63 1989

[76] Clauser J F, Holt R A, Horne M A and Shimony A Proposed experiment to test local hidden-variable theories, *Phys. Rev. Lett.* **23** 880–884, 1969

[77] Cleve R and Buhrman H, Substituting Quantum Entanglement for Communication (quant-ph/9704026) *Phys rev A*,**56** 2, 1201–1204, 1997

[78] Cleve R, van Dam W, Nielsen M and Tapp A, Quantum Entanglement and the Communication Complexity of the Inner Product Function (quant-ph/9708019) 1997

[79] Cleve R, Gottesman D, Lo H-K. How to share a quantum secret, quant-ph/9901025

[80] Cohen-Tanoudji C, *Quantum Mechanics*, Wiley press, New York (1977)

[81] Coppersmith D, An approximate Fourier transform useful in quantum factoring, IBM Research Report RC 19642, 1994

[82] Cormen T, Leiserson C and Rivest R, *Introduction to Algorithms*, (pp 776–800 for FFT, 837–844 for primality test, 812 for extended Euclid algorithm, 834–836 for RSA cryptosystem) MIT press, 1990

[83] Cory D G, Fahmy A F, and Havel T F, Nuclear magnetic resonance spectroscopy: an experimentally accessible paradigm for quantum computing, in *Proc. of the 4th Workshop on Physics and Computation* (Complex Systems Institute, Boston, New England) 1996

[84] Cory D. G, Fahmy A F, and Havel T F *Proc. Nat. Acad. of Sciences of the U. S.*, 94:1634–1639, 1997.

[85] Cory D G, Mass W, Price M, Knill E, Laflamme R, Zurek W H, and Havel T F and Somaroo S S, Experimental quantum error correction In *quant-ph*

[86] van Dam W, A Universal Quantum Cellular Automaton, *Proceedings of the Fourth Workshop on Physics and Computation*, 1996

[87] Deutsch D, Quantum theory, the Church-Turing principle and the universal quantum computer, In *Proc. Roy. Soc. Lond.* A **400** 97-117, 1985

[88] Deutsch D, Quantum computational networks, In *Proc. Roy. Soc. Lond.* A **425** 73-90, 1989

[89] Deutsch D and Jozsa R, Rapid solution of problems by quantum computation, In *Proc. Roy. Soc. Lond* A **439** 553-558, 1992

[90] Deutsch D, Barenco A and Ekert A, Universality in quantum computation, In *Proc. R. Soc. Lond.* A **449** 669-677, 1995

[91] Diaconis P and Rockmore D, Efficient Computation of the Fourier transform on finite groups, *J. AMS* **3** No. 2, 297–332, 1990

[92] Dieks D, Communication by EPR Devices, Phys Lett A, 92(6) 271–272 1982

[93] DiVincenzo D P, Two-bit gates are universal for quantum computation, *Phys. Rev. A* **51** 1015-1022 1995

[94] DiVincenzo D P, Quantum computation, *Science* **270** 255-261 1995

[95] DiVincenzo D P, Quantum Gates and Circuits, (quant-ph/9705009) 1997

[96] Durr C and Hoyer P, A Quantum Algorithm for Finding the Minimum. (quant-ph/9607014) 1996

[97] Durr C, LeThanh H and Santha M, A decision procedure for well formed linear quantum cellular automata, (quant-ph/9604007) *Proceeding of the 37th IEEE Symposium on Foundations of Computer Science*, 38–45, 1996, and *Random Structures & Algorithms*, 1997

[98] Einstein A, Rosen N and Podolsky B, *Phys. Rev.* **47**, 777 1935

[99] Ekert A and Jozsa R Quantum computation and Shor's factoring algorithm, *Rev. Mod. Phys.* **68** 733 1996

[100] Ekert A and Macchiavello C 1996 Quantum error correction for communication, Phys. Rev. Lett. **77** 2585-2588

[101] E. Farhi, J. Goldstone, S. Gutmann, M. Sipser A Limit on the Speed of Quantum Computation in Determining Parity quant-ph/9802045

[102] Feynman R P Simulating physics with computers, In *Int. J. Theor. Phys.* **21** 467-488, 1982

[103] Feynman R P, Quantum mechanical computers, In *Found. of Phys.* **16** 507-531, 1986 see also Optics News February 1985, 11-20.

[104] Fortnow L and Rogers J, Complexity Limitations on quantum computation Technical report 97-003, DePaul University, School of Computer science, 1997

[105] Fortuin C M, Kesteleyn P W and Ginibre J, Correlation inequalities on some partially ordered sets, *Commun. Math. Phys* **22** 89-103, 1971

[106] Fredkin E and Toffoli T 1982 Conservative logic, Int. J. Theor. Phys. **21** 219-253

[107] Freedman M, Logic, P/NP and the quantum field computer, preprint, 1997

[108] P. Ga'cs, Self Correcting Two Dimensional Arrays, in *Randomness and Computation*,1989, edited by S. Micali, vol 5, in series "Advances in Computing Research", pages 240-241,246-248, series editor: F.P.Preparata

[109] P. Ga'cs, one dimensional self correcting array.

[110] Gardiner C W, Quantum Noise, Springer-Verlag, Berlin, 1991

[111] Garey M R and Johnson D S, Computers and Intractability, published by Freeman and Company, New York, 1979

[112] Gaudi A. The set of ropes is presented in *la segrada familia* in Barcelona, Spain.

[113] Geroch R. and Hartle G. Computability and Physical theories in *Between Quantum and Cosmos* edited by Zurek and Van der Merwe and Miller, Princeton University Press, 1988, 549-566

[114] Gershenfeld N A and Chuang I L Bulk spin-resonance quantum computation, *Science*, 275:350–356, 1997.

[115] Gottesman D 1996 Class of quantum error-correcting codes saturating the quantum Hamming bound, Phys. Rev. A **54**, 1862-1868

[116] Gottesman D 1997 A theory of fault-tolerant quantum computation, 1997 (preprint quant-ph 9702029)

[117] Gottesman D, Evslin J, Kakade D and Preskill J, 1996, to be published

[118] Gottesman D, Fault-tolerant quantum computation with higher-dimensional systems," to appear in Proceedings of the 1st NASA International Conference on Quantum Computing and Quantum Communications (1998). Also quant-ph/9802007.

[119] D. Gottesman, Fault-Tolerant Quantum Computation with Local Gates, quant-ph/9903099.

[120] Greenberger D M, Horne M A and Zeilinger A 1989 Going beyond Bell's theorem, in *Bell's theorem, quantum theory and conceptions of the universe*, Kafatos M, ed, (Kluwer Academic, Dordrecht) 73-76

[121] R. B. Griffiths and C. S. Niu 1996 Semi classical Fourier transform for quantum computation in *Phys. Rev. Lett.* ,76, pp. 3228–3231

[122] Grover L K, Quantum mechanics helps in searching for a needle in a haystack, (quant-ph/9706033) Phys. Rev. Lett. **79**, 325-328 1997 and the original STOC paper: A fast quantum mechanical algorithm for database search *Proc. of the 28th Annual ACM Symposium on Theory of Computing (STOC)* 212–221, 1996

[123] Grover L K, A framework for fast quantum mechanical algorithms, (quant-ph/9711043) 1997

[124] Grover L K, Quantum computers can search arbitrarily large databases by a single query (quant-ph/9706005) 1997

[125] Grover L K, A fast quantum mechanical algorithm for estimating the median, (quant-ph/9607024)

[126] Hagley E et. al, Generation of Einstein Podolsky Rosen pairs of atoms, *Phys. Rev. Lett,* **79**, 1–5, 1997

[127] Hellwig K. and Kraus K. Communications in Mathematical Physics, **16** 142 (1970), Kraus K, *states, effects and*

operations: Foundamental notions of Quantum Theory (Springer Verlag, Berlin, 1983)
Schumacher B, Sending Entanglement Through Noisy Quantum Channels, quant-ph/9604023.

[128] Haroche S and Raimond J-M 1996 Quantum computing: dream or nightmare? Phys. Today August 51-52

[129] Hughes R J, Alde D M, Dyer P, Luther G G, Morgan G L ans Schauer M 1995 Quantum cryptography, Contemp. Phys. **36** 149-163

[130] Jones J A and Mosca M 1998 implementation of a Quantum Algorithm to Solve Deutsch's Problem on a Nuclear Magnetic Resonance Quantum Computer, (quant-ph/9801027)

[131] Jozsa R 1997 Entanglement and quantum computation, appearing in *Geometric issues in the foundations of science*, Huggett S *et. al.*, eds, (Oxford University Press)

[132] Khalfin L A and Tsirelson B S, Quantum/Classical Correspondence in the Light of Bell's Inequalities. *Foundations of physics,***22** No. 7, 879–948 July 1992

[133] Lord Kelvin, differential analyzer, 1870

[134] Keyes R W Science **168**, 796, 1970

[135] Kitaev A Yu, Quantum Computations: Algorithms and Error Corrections, in *Russian Math. Serveys*, **52**:6, 1191-1249

[136] Kitaev A Yu 1995 Quantum measurements and the Abelian stablizer problem, (preprint quant-ph/9511026)

[137] Kitaev. A. Yu Quantum error correction with imperfect gates, *Quantum Communication, Computing, and Measurement*, eds: Hirota, Holevo and Caves, 181–188, Plenum Press, New York, 1997.

[138] Kitaev A Yu 1997 Fault-tolerant quantum computation by anyons (preprint quant-ph/9707021)

[139] Kitaev A Yu 1997 Quantum NP Complete Problems, unpublished

[140] Knill E, Laflamme R, and Zurek W, Resilient quantum computation, *Science*, vol 279, p.342, 1998, see also: Knill E, Laflamme R and Zurek W H 1997 Resilient quantum computation: error models and thresholds in *LANL e-print* quant-ph/9702058, http://xxx.lanl.gov (1997), and Knill E and Laflamme R, Concatenated quantum codes, in *LANL e-print* quant-ph/9608012, http://xxx.lanl.gov (1996)

[141] Knill E, Laflamme R, and Zurek W, Resilient quantum computation,(quant-ph/9610011) *Science*, vol 279, p.342, 1998.

[142] Knill E and Laflamme R 1997 A theory of quantum error-correcting codes, Phys. Rev. A **55** 900-911

[143] Knill E, Laflamme R and Zurek W H 1997 Resilient quantum computation: error models and thresholds (preprint quant-ph/9702058)

[144] E. Knill, Non-Binary Unitary Error Bases and Quantum Codes, quant-ph/9608048, 1996.

[145] Kondacs A and Watrous J On the power of Quantum Finite State Automata *38th Annal Symposium on Foundations of Computer Science*,(FOCS) 1997

[146] Kwiat P G, Mattle K, Weinfurter H, Zeilinger A, Sergienko A and Shih Y 1995 New high-intensity source of polarization-entangled photon pairs Phys. Rev. Lett. **75**, 4337-4341

[147] Laflamme R, Miquel C, Paz J P and Zurek W H 1996 Perfect quantum error correcting code, Phys. Rev. Lett. **77**, 198-201

[148] Landauer R. Is quantum mechanics useful? *Phil. Trans. Roy. Soc. of London*, 353:367–376, 1995.

[149] Landauer R 1961 IBM J. Res. Dev. **5** 183, and 1970 IBM J. Res. Dev, volume 14, page 152.

[150] Lecerf Y 1963 Machines de Turing réversibles . Récursive insolubilité en $n \in N$ de l'equation $u = \theta^n u$, où θ est un isomorphisme de codes, C. R. Acad. Francaise Sci. **257**, 2597-2600

[151] Leung D W, Nielsen M A, Cuang I L and Yamamuto Y, (quant-ph/9704002)

[152] Daniel A. Lidar, Isaac L. Chuang and K. Birgitta Whaley (1) Phys. Rev. Lett. 81, 2594-7 (1998)

[153] van Lint J H Coding Theory, Springer-Verlag, 1982

[154] Lipton R J, Using DNA to solve NP-complete problems. *Science*, **268** 542–545, Apr. 28, 1995

[155] Lloyd S, Universal quantum simulators, *Science*, 273:1073–1078, 1996.

[156] Lloyd S 1993 A potentially realisable quantum computer, Science **261** 1569; see also Science **263** 695 (1994).

[157] Lloyd S 1995 Almost any quantum logic gate is universal, Phys. Rev. Lett. **75**, 346-349

[158] Lloyd S 1997 The capacity of a noisy quantum channel, Phys. Rev. A **55** 1613-1622

[159] Loss D and DiVincenzo D P Quantum Computation with Quantum Dots, Phys. Rev. A (preprint quant-ph/9701055) 1997

[160] MacWilliams F J and Sloane N J A 1977 *The theory of error correcting codes*, (Elsevier Science, Amsterdam)

[161] Mattle K, Weinfurter H, Kwiat P G and Zeilinger A 1996 Dense coding in experimental quantum communication, Phys. Rev. Lett. **76**, 4656-4659.

[162] Margolus N 1990 Parallel Quantum Computation, in *Complexity, Entropy and the Physics of Information, Santa Fe Institute Studies in the Sciences of Complexity,* vol VIII p. 273 ed Zurek W H (Addison-Wesley)

[163] Miquel C, Paz J P and Perazzo 1996 Factoring in a dissipative quantum computer Phys. Rev. A **54** 2605-2613

Miquel C, Paz J P and Zurek W H 1997 Quantum computation with phase drift errors, Phys. Rev. Lett. **78** 3971-3974

[164] Menshikov M V, Molchanov S A and Sidorenko A F, Percolation theory and some applications, *Itogi Nauki i Techniki (Series of Probability Theory, Mathematical Statistics, Theoretical Cybernetics)* **24** 53-110, 1986

[165] Monroe C, Meekhof D M, King B E, Itano W M and Wineland D J. Demonstration of a universal quantum logic gate, *Phys. Rev. Lett.*, 75:4714-4717, 1995.

[166] Muckamel D, one dimensional phase transition.

[167] von Neumann, Probabilistic logic and the synthesis of reliable organisms from unreliable components, in *automata studies(Shanon,McCarthy eds)*, 1956

[168] Nisan N and Szegedy M, On the degree of Boolean functions as real polynomials, *Proc. of the 24th Annual ACM Symposium on Theory of Computing (STOC)* 1992

[169] Nielsen M A and Chuang I L 1997 Programmable quantum gate arrays, Phys. Rev. Lett. **79**, 321-324

[170] J. Mod. Opt. **41**, no 12 1994 Special issue: quantum communication

[171] Ozhigov Y, Quantum computers cannot speed up iterated applications of a black box, (quant-ph/9712051)

[172] Palma G M, Suominen K-A & Ekert A K 1996 Quantum computers and dissipation, Proc. Roy. Soc. Lond. A **452** 567-584

[173] Papadimitriou C H, Computational Complexity, Addison-Wesley, 1994

[174] Pellizzari T, Gardiner S A, Cirac J I and Zoller P 1995 Decoherence, continuous observation, and quantum computing: A cavity QED model, Phys. Rev. Lett. **75** 3788-3791

[175] Peres A 1993 *Quantum theory: concepts and methods* (Kluwer Academic Press, Dordrecht)

[176] Prasad G., private communication

[177] Preskill J 1997 Fault tolerant quantum computation, (preprint quant-ph/9712048) to appear in *introduction to Quantum Computation,* edited by H.-K. Lo, S. Popescu, and T. P. Spiller

[178] Privman V, Vagner I D and Kventsel G 1997 Quantum computation in quantum-Hall systems, (preprint, quant-ph/9707017)

[179] Russo L, A note on Percolation, Z. Wahrsch. Geb. **43** 39-48, 1978

[180] Rabin M O, Probabilistic Algorithms *Algorithms and Complexity: New Directions and Recent Results*, pp. 21-39, Academic Press, 1976.

[181] Rains E, Hardin R H, Shor P W and Sloane N J A, A non additive quantum code, (quant-ph/9703002) Phys.Rev.Lett. 79 953–954 1997

[182] Rivest R, Shamir A and Adleman L 1979 On digital signatures and public-key cryptosystems, MIT Laboratory for Computer Science, Technical Report, MIT/LCS/TR-212

[183] Seymour P D and Welsh D J A, Percolation probabilities on the square lattice, *Ann Discrete Math.* **3** 227-245, 1978

[184] J.J.Saqurai Modern Quantum Mechanics, revised edition. Addison Wesley, 1994

[185] Schumacher B W and Nielsen M A 1996 Quantum data processing and error correction Phys Rev A **54**, 2629

[186] Shamir A 1979 Factoring Numbers in O(log(n)) Arithmetic Steps, in *Information Processing Letters 8(1)* 28-31.

[187] Shamir A How to Share a Secret. CACM 22(11): 612-613 (1979)

[188] Shannon C E 1948 A mathematical theory of communication Bell Syst. Tech. J. **27** 379; also p. 623

[189] Shor P W, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM J. Comp.*, **26**, No. 5, pp 1484–1509, October 1997

[190] Shor P W, Scheme for reducing decoherence in quantum computer memory, *Phys. Rev. A*, 52: 2493-2496, 1995.

[191] Shor P W, Fault tolerant quantum computation, In *Proceedings of the 37th Symposium on the Foundations of Computer Science*, pages 56–65, Los Alamitos, California, 1996. IEEE press. quant-ph/9605011.

[192] Shor P W and Laflamme R 1997 Quantum analog of the MacWilliams identities for classical coding theory, Phys. Rev. Lett. **78** 1600-1602

[193] Simon J 1977 On feasible numbers in *STOC 77,*, 195-207

[194] Simon D 1994 On the power of quantum computation, *SIAM J. Comp.*, **26**, No. 5, pp 1474–1483, October 1997

[195] Simon D 1998, private communication.

[196] R Solovay and A. C-C Yao, preprint, 1996

[197] Daniel A. Spielman, Highly Fault-Tolerant Parallel Computation Proceedings of the 37th Annual IEEE Conference on Foundations of Computer Science, 1996, pp. 154–163.

[198] Steane A, Multiple particle interference and quantum error correction, *Proc. Roy. Soc. of London A*, 452:2551-2577, 1996.

[199] Steane A M, Error correcting codes in quantum theory, Phys. Rev. Lett. **77** 793-797, 1996, Simple quantum error-correcting codes, Phys. Rev. A **54**, 4741-4751, 1996, Quantum Reed-Muller codes, submitted to IEEE Trans. Inf. Theory (preprint quant-ph/9608026), 1996 Active stabilization, quantum computation, and quantum state synthesis, Phys. Rev. Lett. **78**, 2252-2255, 1997

[200] Steane A M The ion trap quantum information processor, Appl. Phys. B **64** 623-642 1997

[201] Steane A M Space, time, parallelism and noise requirements for reliable quantum computing (preprint quant-ph/9708021) 1997

[202] Stern A, Aharonov Y and Imry Y, "Phase uncertainty and loss of interference: a general picture" Phys. Rev. A **41**, 3436 (1990). and "Dephasing of interference by a back reacting environment" in "Quantum coherence" ed. J. Anandan, World Scientific, 1990.

[203] W. Tittel, J. Brendel, B. Gisin, T. Herzog, H. Zbinden, N. Gisin Experimental demonstration of quantum-correlations over more than 10 kilometers, in *Phys. Rev. Lett* 1997

[204] Toffoli T 1980 Reversible computing, in *Automata, Languages and Programming*, Seventh Colloquium, Lecture Notes in Computer Science, Vol. 84, de Bakker J W and van Leeuwen J, eds, (Springer) 632-644

[205] Turchette Q A, Hood C J, Lange W, Mabushi H and Kimble H J 1995 Measurement of conditional phase shifts for quantum logic, Phys. Rev. Lett. **75** 4710-4713

[206] Turing A M 1936 On computable numbers, with an application to the Entschneidungsproblem, Proc. Lond. Math. Soc. Ser. 2 **42**, 230 ; see also Proc. Lond. Math. Soc. Ser. 2 **43**, 544

[207] Unruh W G, Maintaining coherence in quantum computers, *Phys. Rev. A*, 51:992–997, 1995.

[208] Valiant, unpublished

[209] Valiant L. G, Negation can be exponentially powerful. *Theoretical Computer Science,* 12(3):303-314, November 1980.

[210] Valiant L G and Vazirani V V. NP is as easy as detecting unique solutions. *Theoretical Computer Science,* 47(1):85-93, 1986

[211] Vedral V, Barenco A and Ekert A 1996 Quantum networks for elementary arithmetic operations, Phys. Rev. A **54** 147-153

[212] Vergis A, Steiglitz K and Dickinson B , "The Complexity of Analog Computation", Math. Comput. Simulation 28, pp. 91-113. 1986

[213] Warren W S, *Science*, 277:1688–1698, 1997.

[214] L. C. Washington, Introduction to Cyclotomic Fields, Spinger-Verlag, Graduate Texts in Mathematics 83, 1982.

[215] Watrous J, On one Dimensional quantum cellular automata, *Complex Systems* **5** (1), pp 19–30, 1991

[216] Wiesner S 1996 Simulations of many-body quantum systems by a quantum computer (preprint quant-ph/9603028)

[217] Wheeler J A and Zurek W H, eds, 1983 *Quantum theory and measurement* (Princeton Univ. Press, Princeton, NJ)

[218] Wigderson, private communication

[219] Wootters W K and Zurek W H 1982 A single quantum cannot be cloned, Nature **299**, 802

[220] Wootters W K A Measure of the Distinguishability of Quantum States *Quantum Optics, General Relativity, and Measurement* eds: Marlan O. Scully and Pierre Meystre, 145–154, Plenum Press, New York, 1983

[221] Yao A C-C, Quantum circuit complexity FOCS 1993, 352–361

[222] Zalka C, Efficient simulation of quantum systems by quantum computers *Proc. Roy. Soc. of London A*, in press, (preprint quant-ph/9603026)

[223] Zalka C, Grover's quantum searching algorithm is optimal, (quant-ph/9711070)

[224] Zurek W H, Decoherence and the transition from quantum to classical Physics Today 44(10), October, 1991 36–44.