FAULT-TOLERANT QUANTUM COMPUTATION WITH CONSTANT ERROR RATE *

DORIT AHARONOV† AND MICHAEL BEN-OR‡

Key Words: Quantum computation, Noise and Decoherence, Density matrices, Concatenated quantum error correcting codes, Polynomial codes, Universal quantum gates

Abstract. Shor has 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. This paper improves this result and shows that quantum computation can be made robust against errors and inaccuracies, when the error rate, η, is smaller than a constant threshold, η_c. The cost is polylogarithmic in space and time. The result holds for a very general noise model, which includes probabilistic errors, decoherence, amplitude damping, depolarization, and systematic inaccuracies in the gates. Moreover, we allow exponentially decaying correlations between the errors both in space and in time. 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.

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.

The paper contains new and significantly simpler proofs for most of the known results which we use. For example, we give a simple proof that it suffices to correct bit and phase flips, we significantly simplify Calderbank and Shor's original proof of the correctness of CSS codes. We also give a simple proof of the fact that two-qubit gates are universal. The paper thus provides a self contained and complete proof for universal fault tolerant quantum computation.

1. Outline. Quantum computation has recently gained a lot of attention, due to oracle results[65, 10, 11] and quantum algorithms[25, 65, 36], in particular Shor's factorization algorithm [62]. These results indicate a possibility that quantum computers are exponentially more powerful than classical computers. It is yet unclear whether and how quantum computers will be physically realizable[19, 22, 30, 37, 50, 51, 55, 59, 69] but as any physical system, they *in principle* will be subjected to noise, such as decoherence [76, 71], and inaccuracies. Thus, 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[72, 17, 54, 52, 7], and hence the computation must be protected. For classical circuits, von Neumann has shown already in 1956

---

[*] A preliminary version of this paper, under the name "Fault-Tolerant Quantum Computation With Constant Error", was published in *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC)* 1997

[†] School of Mathematics, Institute for Advanced Studies, Princeton, New Jersey

[‡] Department of Computer Science, The Hebrew University, Jerusalem, Israel

that the computation can be made robust to noise[53]. 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 quantum bits (qubits). It was argued by scientists that due to the continuity of quantum states, and to the fact that quantum states cannot be cloned[74], it will be impossible to protect quantum information from noise[72, 47]. Despite these pessimistic beliefs, Shor discovered a scheme to reduce the effect of decoherence[63]. Immediately after that, Calderbank and Shor[13] and Steane[68] showed that good quantum error correcting codes exist, a result which was followed by many explicit examples of quantum codes (e.g.[46, 68]). A theory for quantum error correcting codes was developed[43], and a group theoretical framework for almost all codes was found [14, 15, 31]. 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[64] 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 this paper 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, 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. This assumption can be slightly relaxed, by allowing exponentially decaying correlations in both space and time. Such assumptions are made also in the classical scenario, and are likely to hold in physical realizations of quantum computers. Thus, this paper settles the question of quantum computation in the presence of local noise.

Let us first describe the computational model with which we work. The standard model[23, 24, 75] of quantum circuits with unitary gates, allows only unitary operations on qubits. However, noisy quantum systems are not isolated from other quantum systems, usually referred to as the environment. Their interactions with the environment are unitary, but when restricting the system to the quantum computer alone the operation on the system is no longer unitary, and the state of the quantum circuit is no longer a vector in the Hilbert space. It is possible to stay in the unitary model, keeping track of the state of the environment. However, this environment is not part of the computer, and it is assumed that we have no control or information on its state. We find it more elegant to follow the framework of the physicists, and to work within the model of quantum circuits with mixed states defined by Aharonov, Kitaev and Nisan[4]. In this model, the state of the set of qubits is always defined: It is a probability distribution over pure states, i.e. a *mixed state*, or a density matrix, and not merely a pure state as in the standard model. The quantum gates in this model are not necessarily unitary: any physically allowed operator on qubits is a quantum gate. In general, it is easiest to think of a general physical operator as a unitary operation on the system of qubits and any number of extra qubits, followed by an operator which discards the extra qubits. In particular, one can describe in this model an operation which adds a blank qubit to the system, or discards a qubit. The model of quantum circuits with mixed states is equivalent in computational power to the standard model of quantum circuits[4], but is more appropriate to work with when dealing with errors.

The noise process can be described naturally in this model as follows. Since noise is a dynamical process which depends on time, the circuit will be leveled, i.e. gates will be applied at discrete time steps. Between the time steps, we add the noise process. The simplest model for noise is the probabilistic process: Each time step, each qubit or gate undergoes a fault (i.e. an arbitrary quantum operation) with independent probability η, and η is referred to as the *error rate.* The probabilistic noise process can be generalized to a more realistic model of noise, which is the following: Each qubit or (qubits participating in the same gate), each time step, undergoes a physical operator which is at most η far from the identity, in some metric on operators. This model includes, apart from probabilistic errors, also decoherence, amplitude and phase damping and systematic inaccuracies in the gates. Two important assumptions were made in this definition: Independence between different faults in space, i.e. locality, and independence in time, which is called the Markovian assumption. It turns out that for our purposes, we can release these two restrictions slightly, to allow exponentially decaying correlations in both time and space, and all the results of this paper will still hold. However, we will turn to the generalized noise model in a very late stage of this paper. Meanwhile, it is simpler to keep in mind the independent probabilistic noise model.

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[13] and Steane [68] 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.*[8], 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 paper we give an alternative proof for CSS codes, which generalized these ideas to quantum particles with $p > 2$ states, and is significantly simpler than the original proof of Calderbank and Shor[13].

We define a new class of quantum codes, which are called *polynomial codes.* The idea is based on a theorem by Schumacher and Nielsen[61] 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 Ben-Or, Goldwasser and Wigderson[9], 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. 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*[35] 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 [64], 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. We remark here that in this paper we make an effort to show that fault tolerance can be achieved without using classical operations or measurements. This fact is desirable because of two reasons. One is purely theoretical: One would like to know that measurements and 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 gates 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 in [9]. 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[39] and Solovay[66] 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[39]. 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 noise 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 [64] 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 goal of this paper is to improve this result and to show 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 noise 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 noise 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 noise 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 [38] and Gác's [28] 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 approximate 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.

Some of the assumptions we made in this work cannot be released. First, the choice to work with circuits which allow parallelism is necessary, because sequential quantum computation, such as quantum Turing machines, cannot be noise resistant[3], since error corrections cannot be applied fast enough to prevent accumulation of errors. It is also crucial that we allow qubits to be input at different times to the circuit unlike (Yao[75]), because this provides a way to discard entropy which is accumulating in the circuit. Without this assumption, i.e. under the restriction that all qubits are initialized at time 0, it is impossible to compute fault tolerantly without an exponential blowup in the size of the circuit[5]. An interesting question is whether other assumptions, such as the locality assumption, can be relaxed. A partial positive answer in this direction was given by Chuang *et. al.*[48]. In connection with the assumptions made for Fault tolerance, see also [1, 70].

A substantial part of the results in this paper were published in a proceeding version in [2]. The threshold result was independently described also by Knill, Laflamme and Zurek[42], and by Kitaev[40, 39]. This was done using different quantum correcting codes, and required measurements in the middle of the quantum computation. In this work we provide a complete proof to all the details of the fault tolerant quantum computation codes construction and in particular we rigorously show that the recursive scheme works. Kitaev has shown another, very elegant way to achieve fault tolerance in a different model of quantum computation, which works with anyons[41]. After the results were first published, it was shown by Gottesman[32] 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[33]. Fault tolerant quantum computation in low dimensions is also discussed by Gottesman[34]. Non binary codes were defined independently also by Chuang[18] and Knill[44]. For a survey of this area see Preskill[58].

Organization of paper: In section 2 we define the model of noisy quantum circuits. Section 3 is devoted to CSS and polynomial codes. In section 4 we describe how to apply fault tolerant operations on states encoded by CSS codes. In section 5 we describe fault tolerant procedures for polynomial codes. Section 6 proves the universality of the two sets of gates used in the previous two sections. In section 7 we present recursive simulations and the threshold result for probabilistic errors. In section 8 we generalize the result to general local noise, and in section 9 we generalize to circuits which use any universal set of gates. In section 10 we explain how to deal with exponentially decaying correlations in the noise. Section 11 proves the threshold result for one dimensional quantum circuit, and in 12 we conclude with possible implications to physics, and some open questions.

2. The Model of Noisy Quantum Circuits. In this section we recall the definitions of quantum circuits with mixed states as defined by Aharonov, Kitaev and Nisan[4], and define noisy quantum circuits. The model is based on basic quantum physics, and good references are Cohen-Tanudji[21], Sakurai [60], and Peres[56].

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: $|i\rangle, 0 \leq i < 2^n$, 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. Quantum Operators and Quantum Gates. Transformations of density matrices are linear operators on operators (sometimes called *super-operators*). $\mathbf{L}(\mathcal{N})$ denotes the set of all linear operators on $\mathcal{N}$, a finite dimensional Hilbert space. The most general quantum operator is any operator $T : \mathbf{L}(\mathcal{N}) \to \mathbf{L}(\mathcal{M})$ which sends density matrices to density matrices. A super-operator is called *positive* if it sends positive semi-definite Hermitian matrices to positive semi-definite Hermitian matrices, so T must be positive. 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 $T \otimes I : \mathbf{L}(\mathcal{N} \otimes \mathcal{R}) \to \mathbf{L}(\mathcal{M} \otimes \mathcal{R})$, and it is also required that the extended operator remains positive, and such T is called a *completely positive map.* Clearly, T should also be trace-preserving. It turns out that any super operator which satisfies these conditions indeed take density matrices to density matrices[45]. Thus quantum super-operators are defined as follows:

DEFINITION 1. *A* **physical super-operator**, *T, from k to l qubits is a trace preserving, completely positive, linear map from density matrices on k qubits to density matrices on l qubits.*

Linear operations on mixed states preserve the probabilistic interpretation of the mixture, because $T \circ \rho = T \circ (\sum_l p_l |\alpha_l\rangle\langle\alpha_l|) = \sum_l p_l T \circ (|\alpha_l\rangle\langle\alpha_l|)$.

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 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 $\mathcal{F}$. 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. In this paper, a discarding qubit gate will always be applied on qubits which are in tensor product with the rest of the system. Any quantum operation which does not operate on $\mathcal{F}$ commutes with this tracing out. One more useful quantum super-operator is the one which adds a blank qubit to the system, $V_0 : |\xi\rangle \mapsto |\xi\rangle \otimes |0\rangle : \mathbf{C}^{2^n} \to \mathbf{C}^{2^{n+1}}$. This is described by the super-operator $T : \rho \mapsto \rho \otimes |0\rangle\langle 0|$.

A lemma by Choi[16], and Hellwig and Kraus[45] asserts that any physically allowed super-operator from k to l qubits can be described as a combination of the above three operations: add $k+l$ blank qubits, apply a unitary transformation on the $2k+l$ qubits and then trace out $2k$ qubits. One can think of this as the definition of a quantum super-operator. We define a quantum gate to be the most general quantum operator.

DEFINITION 2. *A quantum gate of order (k,l) is a super-operator from k to l qubits.*

In this paper we will only use three types of quantum gates: discard or add qubits, and of course unitary gates. However, quantum noise will be allowed to be a general physical operator.

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 $\Pr(m) = \mathrm{Tr}(P_m \rho)$. Any measurement thus defines a probability distribution over the possible outputs. In this paper we will only need 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.

To describe a measurement as a super-operator, we replace the classical result of a measurement, m, 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:

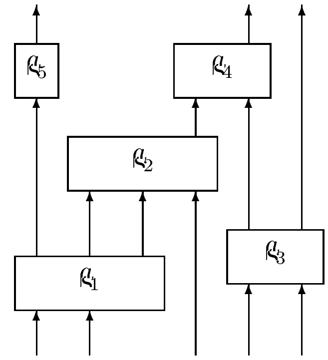
$$T\rho \;=\; \sum_m (P_m \rho P_m) \otimes \Big(|m\rangle\langle m|\Big) \tag{2.1}$$

2.5. The Trace Metric on Density Operators. A useful metric on density matrices, the *trace* metric, was defined in [4]. It is induced from the *trace norm* for general Hermitian operators. The trace norm on Hermitian matrices is the sum of the absolute values of the eigenvalues. $\|\rho\| = \sum_i |\lambda_i|$ It was shown in [4] that the distance between two matrices in this metric $\|\rho_1 - \rho_2\|$ is exactly the measurable distance between them: It equals to the maximal total variation distance between all possible outcome distributions of the same measurement done on the two matrices.

2.6. Quantum Circuits with Mixed States. We now define a quantum circuit:

DEFINITION 3. *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 $\mathfrak{g}_v \in \mathcal{G}$ of order (k_v, l_v). The in-degree and out-degree of v are equal k_v and l_v, respectively.*

Here is a schematic example of such a circuit.



The circuit operates on a density matrix as follows:

DEFINITION 4. **final density matrix***: Let Q be a quantum circuit. Choose a topological sort for Q: $\mathfrak{g}_t ... \mathfrak{g}_1$. Where $\mathfrak{g}_j$ are the gates used in the circuit. The final density matrix for an initial density matrix ρ is $Q \circ \rho = \mathfrak{g}_t \circ ... \circ \mathfrak{g}_2 \circ \mathfrak{g}_1 \circ \rho$.*

$Q \circ \rho$ is well defined and does not depend on the topological sort of the circuit [4]. At the end of the computation, the r output qubits are measured, and the classical outcome, which is a string of r bits, is the output of the circuit. For an input string i, the probability for an output string j is $\langle j|\Big(Q \circ |i\rangle\langle i|\Big)|j\rangle$.

DEFINITION 5. **Computed function:** *Let Q be a quantum circuit, with n inputs and r outputs. Q is said to compute a function f: $\{0,1\}^n \longmapsto \{0,1\}^r$, with accuracy ϵ if for any input $i \in \{0,1\}^n$, the probability for the output to be $f(i)$ is greater than $1 - \epsilon$.*

2.7. Quantum Circuits with probabilistic Errors. .

To implement noise in the quantum circuits, we divide the quantum circuit to time steps, and the gates are applied at integer time steps. We permit that qubits are input and output at different time steps. Faults occur in between time steps, in the *locations* of the circuit.

Definition 6. *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.*

Each location in the circuit exhibits a fault with with independent probability η.

Definition 7. *The list of times and places where faults had occurred, (in a specific run of the computation) is called a fault path.*

Each fault path, $\mathcal{F}$, is assigned a probability $Pr(\mathcal{F})$, which is a function of the number of faults in it. A fault at time t at a certain location means that an (arbitrary) general quantum operation is applied on the faulty qubits after time t. Given a certain fault path, we still have to determine the quantum operations that were applied on the faulty qubits. Denote by $\mathcal{E}(\mathcal{F})$ the choice of faults for the fault path, say $\mathcal{E}_t, ...\mathcal{E}_2, \mathcal{E}_1$. The final density matrix for this given choice $\mathcal{E}(\mathcal{F})$ is easily defined by applying the faults in between the gates: $Q(\mathcal{F}) \circ \rho = \mathcal{E}_t \circ U_t \circ ...\mathcal{E}_2 \circ U_2 \circ \mathcal{E}_1 \circ U_1 \circ \rho$. The final density matrix, $Q \circ \rho$ of the circuit is defined as a weighted average over the final density matrices for each fault path:

$$Q \circ \rho = \sum_{\mathcal{F}} Pr(\mathcal{F}) Q(\mathcal{F}) \circ \rho. \tag{2.2}$$

This final density matrix depends on the exact choice of quantum operations applied on the faulty gates. However, a noisy quantum computer is said to compute a function f with confidence ϵ in the presence of error rate η, if the circuit outputs the function f with accuracy ϵ regardless of the exact specification of the quantum operations which occurred during the faults.

2.8. Quantum Circuits with General Local Noise.

In most of the paper we will use only the probabilistic noise model described above. However, our result will be generalized in section 8 to a much more general noise model. In the general noise model, we replace the probabilistic faults which occur with probability η in each location by applying in each location in the quantum circuit an arbitrary general quantum operator, on which the only restriction is that it within η distance to the identity, in a certain norm of super-operators. Thus, in between time steps, a *noise operator* operates on all the qubits. For the time step t the noise operator is of the form

$$\mathcal{E}(t) = \mathcal{E}_{A_{1,t}}(t) \otimes \mathcal{E}_{A_{2,t}}(t) \otimes \cdots \otimes \mathcal{E}_{A_{l_t,t}}(t). \tag{2.3}$$

$A_{i,t}$ runs over all possible locations at time t, and for each one of them,

$$\|\mathcal{E}_{A_{i,t}}(t) - I\| \leq \eta \tag{2.4}$$

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

1. $||T\rho|| \leq ||T||||\rho||$
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 suggested by Kitaev[39, 4].

2.9. Connection to Specific Noise Processes. Independent and quasi independent probabilistic errors, amplitude and phase damping, decoherence, and systematic inaccuracies in gates, are all special cases of the general noise model defined above. For independent probabilistic errors, where an error $\mathcal{E}$ occurs with probability η, we can write for each one of the super-operators in the product2.3:

$$\mathcal{E}_{A_{i,t}}(t) = (1-\eta)I_{A_{i,t}} + \eta\mathcal{E}'_{A_{i,t}}(t) \tag{2.5}$$

where $\mathcal{E}'_{A_{i,t}}(t)$ is any physical super operator. This satisfies the conditions of local noise 2.3,2.4 with error rate 2η. This is true, in particular, also for probabilistic collapses of the wave function, a process in which each qubit randomly collapses to one of its basis states, for a given basis (not necessarily the computational basis $|0\rangle, |1\rangle$). $\mathcal{E}'_{A_{i,t}}(t)$ is then the physical super-operator which measures the qubit in the chosen basis. Depolarization, in which a qubit is randomly replaced by a completely random qubit, i.e. a qubit in the identity density matrix, is also a special case of probabilistic noise, in which the super-operator $\mathcal{E}'_{A_{i,t}}(t)$ replaces all the qubits in $A_{i,t}$ by completely random qubits. Decoherence[17], 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 above models 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. The generalizations described in the last subsection describe how to implement quasi independent probabilistic errors.

2.10. Adding correlations to the Noise Model. Two very important assumptions were made when introducing our general model of noise.

- Locality: No correlations between environments of 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.

Both of these assumptions can be slightly released, to allow exponentially decaying correlations in both space and time, while the results of this paper still hold.

To add exponentially decaying correlations to the probabilistic noise model, we generalize it in the following way. Instead of considering independent probabilities for error in each location, we require that the probability for a fault path which contains k locations is bounded by some constant times the probability for the same fault path in the independent errors model:

$$Pr(fault\ pat$$

adds some blank qubits, which are called the environment, at the beginning of the computation. Each time step, the adversary can operate a general operator on the environment and the set of qubits in the fault path at that time step. This model allows correlations in space, since the noise operator need not be of the form of a tensor product of operators on different locations. Correlations in time appear because the environment that the adversary added in not renewed each time step, so noise operators of different time steps are correlated. Note that the independent probabilistic noise process is a special case of this process.

3. Quantum Error Correcting Codes. Here we recall the definition of quantum codes, and of Calderbank-Shor-Steane (CSS) codes[13, 68]. 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[13], 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.*[8], 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[68].

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 based on ideas by Ben-Or *et. al.*[9] 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 [49].

3.1. Quantum Codes. DEFINITION 8. *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

$$\mathcal{R} \circ \mathcal{E}(q_1,..q_d) \circ \rho = \rho \tag{3.1}$$

DEFINITION 9. *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 $\mathcal{E}(q_1,..q_d)$*

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

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[8] 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:

$$\mathcal{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \sigma_y = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{3.2}$$

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:

$$\sigma_e = \sigma_{e_d} \otimes \cdots \sigma_{e_1}, \quad e \in \{1, x, y, z\}^d \tag{3.3}$$

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 10. *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*

$$\mathcal{R} \circ \sigma_e|\alpha\rangle\langle\alpha'|\sigma_e^\dagger = |\alpha\rangle\langle\alpha'| \tag{3.4}$$

Definition 11. *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, $\sigma_e \neq \sigma_{e'}$,*

$$\mathcal{R} \circ \sigma_e|\alpha\rangle\langle\alpha'|\sigma_{e'}^\dagger = 0 \tag{3.5}$$

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 rest of the paper, we will only use error correcting procedures which detect the errors. We can now prove the following:

Theorem 1. *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 $0 \leq i \leq 2^d - 1$,

$$|i\rangle \longmapsto |0^{d+d'}\rangle|i\rangle \longmapsto \sum_{j=0}^{2^d-1} |A_i^j\rangle \otimes |j\rangle \tag{3.6}$$

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:

$$\mathcal{L} = \sum_{e,e'} |A_e\rangle\langle A_{e'}| \otimes \sigma_e \cdot \sigma_{e'}^\dagger \tag{3.7}$$

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

$$\mathcal{R} \circ \mathcal{E} \circ |\alpha\rangle\langle\alpha'| = \mathcal{R} \circ \mathcal{T} \circ \sum_{e,e'} |A_e\rangle\langle A_{e'}| \otimes \sigma_e|\alpha\rangle\langle\alpha'|\sigma_{e'}^\dagger = (\mathcal{T} \circ \sum_e |A_e\rangle\langle A_e|) \otimes |\alpha\rangle\langle\alpha'| = |\alpha\rangle\langle\alpha'| \tag{3.8}$$

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 [13]. 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:

$$\forall a \in C_1/C_2 : |S_a\rangle = \frac{1}{\sqrt{2^{dim(C_2)}}} \sum_{w \in C_2} |w + a\rangle. \tag{3.9}$$

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:

$$H = H^{-1} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \tag{3.10}$$

Observe that

$$H\sigma_z H^{-1} = \sigma_x \tag{3.11}$$

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:

$$|C_a\rangle = H \otimes H \otimes \cdots \otimes H|S_a\rangle = \frac{1}{\sqrt{2^{m+dim(C_2)}}} \sum_{b=0}^{2^m-1} \sum_{w \in C_2} (-1)^{(w+a)\cdot b}|b\rangle = \frac{1}{\sqrt{2^{m-dim(C_2)}}} \sum_{b \in C_2^\perp} (-1)^{a\cdot b}|b\rangle \tag{3.12}$$

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[13]. We give here a simple proof of this theorem, based on theorem 1.

THEOREM 2. *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

$$\mathcal{R}_{C_1}|i\rangle = |w\rangle \otimes |e\rangle \tag{3.13}$$

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

$$R_{C_1} \circ \mathcal{E}_b \circ |\alpha\rangle\langle\alpha'| = |\alpha\rangle\langle\alpha'| \otimes |e_b\rangle\langle e_b| \tag{3.14}$$

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

$$\mathcal{R}_{C_2^{\perp}}|j\rangle = |w\rangle \otimes |e\rangle \tag{3.15}$$

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

$$R_{C_2^{\perp}} \circ \mathcal{E}_f \circ |\beta\rangle\langle\beta'| = |\beta\rangle\langle\beta'| \otimes |e_f\rangle\langle e_f| \tag{3.16}$$

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

$$\mathcal{R} = \mathcal{T}_f \circ \mathcal{T}_b \circ \mathcal{H} \circ \mathcal{R}_{C_2^{\perp}} \circ \mathcal{H} \circ \mathcal{R}_{C_1} \tag{3.17}$$

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 1, 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 3.17, the fact that $\mathcal{R}_{C_1}$ commutes with $\mathcal{E}_f$, equation 3.14,3.11 and 3.16, we have the desired result:

$$\mathcal{R} \circ \mathcal{E} \circ |\alpha\rangle\langle\alpha'| = \mathcal{T}_f \circ \mathcal{T}_b \circ \mathcal{H} \circ \mathcal{R}_{C_2^{\perp}} \circ \mathcal{H} \circ \mathcal{E}_f \circ \mathcal{R}_{C_1} \circ \mathcal{E}_b \circ |\alpha\rangle\langle\alpha'| = \tag{3.18}$$

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

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 $e_f \neq e'_f$ $\mathcal{T}_f \circ |e_f\rangle\langle e'_f| = 0$. $\square$.

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,

$$B^c P^{c'}, \quad \forall c, c' \in F_p. \tag{3.19}$$

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

$$W : |a\rangle \longmapsto \frac{1}{\sqrt{p}} \sum_{b \in F} w^{ab} |b\rangle. \tag{3.20}$$

And it can be easily checked that

$$\forall c \in F_p, \quad WP^cW^{-1} = B^c \quad , \quad WB^cW^{-1}P^c \tag{3.21}$$

We can therefore define CSS codes over F_p. The statements and proofs of theorems 1 and 2 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:

$$\begin{aligned}
C_1 &= \{(f(\alpha_1) \cdot f(\alpha_m)) | f(x) \in F(x), deg(f(x)) \leq d\} \subset F_p^m \\
C_2 &= \{(f(\alpha_1) \cdot f(\alpha_m)) | f(x) \in F(x), deg(f(x)) \leq d, f(0) = 0\} \subset C_1
\end{aligned} \tag{3.22}$$

We can now define the quantum code:

$$\forall a \in C_1/C_2, \quad |S_a\rangle = \frac{1}{\sqrt{p^d}} \sum_{f \in V_1, f(0)=a} |f(\alpha_1) \cdots f(\alpha_m)\rangle \tag{3.23}$$

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 3. *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 3 follows from theorem 2. □

4. Computing on States Encoded by CSS codes.

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

Definition 12. *A sequence of gates $P(\mathfrak{g})$ is said to encode a gate $\mathfrak{g}$ for the code C if for any superposition $|\alpha\rangle$, $P(\mathfrak{g})|S_{|\alpha\rangle}\rangle = |S_{g|\alpha\rangle}\rangle$.*

We will want $P(\mathfrak{g})$ 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 13. *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.

$$|S_a\rangle \longmapsto |S_{g(a)}\rangle \otimes |A\rangle \tag{4.1}$$

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

$$\begin{aligned} |S_{\vec{0}}\rangle &= \sum_{w \in C^{\perp}} |w\rangle \\ |S_{\vec{1}}\rangle &= \sum_{w \in C^{\perp}} |w + \vec{1}\rangle \end{aligned} \tag{4.2}$$

$|S_{\vec{0}}\rangle$,$|S_{\vec{1}}\rangle$, 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 4. *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 $\mathfrak{G}$ 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 $a, b \in C/C^{\perp}$.

$$|S_a\rangle = \sum_{w\in C^{\perp}} |a_1+w_1\rangle \otimes ...|a_m+w_m\rangle \longmapsto \sum_{w\in C^{\perp}} |a_1+1+w_1\rangle \otimes ...|a_m+1+w_m\rangle = |S_{a+\vec{1}}\rangle \tag{4.3}$$

For CNOT,

$$\begin{aligned} |S_a\rangle|S_b\rangle = &\sum_{w\in C^{\perp}} |a_1+w_1\rangle \otimes ...|a_m+w_m\rangle \sum_{w'\in C^{\perp}} |b_1+w'_1\rangle \otimes ...|b_m+w'_m\rangle \longmapsto \\ &\sum_{w\in C^{\perp}} |a_1+w_1\rangle \otimes ...|a_m+w_m\rangle \sum_{w'\in C^{\perp}} |a_1+b_1+w_1+w'_1\rangle \otimes ...|a_m+b_m+w_m+w'_m\rangle \\ &= |S_a\rangle|S_{a+b}\rangle \end{aligned} \tag{4.4}$$

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

$$|S_a\rangle = \sum_{w\in C^{\perp}} |a_1+w_1\rangle \otimes ...|a_m+w_m\rangle \longmapsto \sum_{w\in C^{\perp}} i^{3(\sum_k a_k+w_k)}|a_1+w_1\rangle \otimes ...|a_m+w_m\rangle. \tag{4.5}$$

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,

$$|S_{\vec{a}}\rangle|S_{\vec{b}}\rangle = \sum_{w,w'\in C^{\perp}} |\vec{a}+w\rangle|\vec{b}+w'\rangle \longmapsto \sum_{w,w'\in C^{\perp}} (-1)^{(\vec{a}+w)\cdot(\vec{b}+w')}|\vec{a}+w\rangle|\vec{b}+w'\rangle \tag{4.6}$$

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,

$$\begin{aligned} |S_a\rangle = &\sum_{w\in C^{\perp}} |a_1+w_1\rangle \otimes ...|a_m+w_m\rangle \longmapsto \sum_{x\in F_2^m}\sum_{w\in C^{\perp}} (-1)^{(a+w)\cdot x}|x\rangle = \\ &= \sum_{x\in C}(-1)^{a\cdot x}|x\rangle = \sum_{b\in C/C^{\perp}}\sum_{w\in C^{\perp}} (-1)^{a\cdot(b+w)}|b+w\rangle = \sum_{b}(-1)^{a\cdot b}|S_b\rangle. \end{aligned} \tag{4.7}$$

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 $j't$

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:

$$(c_0|S_0\rangle + c_1|S_1\rangle)\sum_i |i\rangle \longmapsto (c_0|S_0\rangle + c_1|S_1\rangle)\sum_i |i\rangle \tag{4.11}$$

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[64] 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

$$
\begin{aligned}
|0^m\rangle &\longmapsto |0^m\rangle|S_0\rangle \\
|1^m\rangle &\longmapsto |1^m\rangle|S_1\rangle
\end{aligned}
\tag{4.12}
$$

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 $i't$

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.

4.10.1. Construction of the Ancilla state $|A\rangle$. We would now like to show how to construct the state

$$|A\rangle = \frac{1}{2}|S_0S_0S_0\rangle + |S_0S_1S_0\rangle + |S_1S_0S_0\rangle + |S_1S_1S_1\rangle. \tag{4.15}$$

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

$$|B\rangle = \frac{1}{2}|S_0S_0S_1\rangle + |S_0S_1S_1\rangle + |S_1S_0S_1\rangle + |S_1S_1S_0\rangle. \tag{4.16}$$

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

The idea is that

$$\frac{1}{\sqrt{2}}(|A\rangle + |B\rangle) \tag{4.17}$$

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

$$\frac{1}{2\sqrt{2}}(|S_0\rangle + |S_1\rangle)(|S_0\rangle + |S_1\rangle)(|S_0\rangle + |S_1\rangle) \tag{4.18}$$

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,

$$|S_{cat}\rangle = \frac{1}{\sqrt{2}}(|S_0\rangle^m + |S_1\rangle^m) \tag{4.19}$$

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

$$\begin{aligned} |S_0\rangle^m|A\rangle &\longmapsto |S_0\rangle^m|A\rangle \\ |S_1\rangle^m|A\rangle &\longmapsto |S_1\rangle^m|A\rangle \\ |S_0\rangle^m|B\rangle &\longmapsto |S_0\rangle^m|B\rangle \\ |S_1\rangle^m|B\rangle &\longmapsto -|S_1\rangle^m|B\rangle \end{aligned} \tag{4.20}$$

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

$$\frac{1}{2}(|S_0\rangle^m + |S_1\rangle^m)(|A\rangle + |B\rangle) \tag{4.21}$$

and apply the transformation 4.20 to get

$$\frac{1}{2}(|S_0\rangle^m + |S_1\rangle^m)|A\rangle + \frac{1}{2}(|S_0\rangle^m - |S_1\rangle^m)|B\rangle \tag{4.22}$$

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 4.20 can be done by applying

$$|S_a\rangle|b\rangle|c\rangle|d\rangle \longmapsto (-1)^{a(bc+d)}|S_a\rangle|b\rangle|c\rangle|d\rangle \tag{4.23}$$

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 4.23 need not be fault tolerant, and it might ruin the entire block $|S_a\rangle$. Applying transformation 4.23 for $1 \leq i \leq m$ it is easy to check that we get

$$|S_a\rangle^m|S_b\rangle|S_c\rangle|S_d\rangle \longmapsto (-1)^{a(bc+d)}|S_a\rangle^m|S_b\rangle|S_c\rangle|S_d\rangle \tag{4.24}$$

which exactly achieves transformation 4.20.

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

$$\frac{1}{2}(|S_0\rangle^m + |S_1\rangle^m)^3|A\rangle + \frac{1}{2}(|S_0\rangle^m - |S_1\rangle^m)^3|B\rangle \tag{4.25}$$

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:

$$\begin{aligned} \frac{1}{\sqrt{2}}(|S_0\rangle^m + |S_1\rangle^m) &\longmapsto \frac{1}{\sqrt{2^{m-1}}} \sum_{i=0, i\cdot\vec{1}=0}^{2^m} |S_{i_1}\rangle|S_{i_2}\rangle...|S_{i_m}\rangle, \\ \frac{1}{\sqrt{2}}(|S_0\rangle^m - |S_1\rangle^m) &\longmapsto \frac{1}{\sqrt{2^{m-1}}} \sum_{i=0, i\cdot\vec{1}=1}^{2^m} |S_{i_1}\rangle|S_{i_2}\rangle...|S_{i_m}\rangle \end{aligned} \tag{4.26}$$

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

$$|a, b, c, d\rangle \longmapsto |a, b, c, d + maj(a, b, c)\rangle \tag{4.27}$$

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 4.20. 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 (4.25) 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 (4.25) 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.

4.10.2. 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

$$|\alpha\rangle = |S_a\rangle|S_b\rangle|S_c\rangle \longmapsto |S_a\rangle|S_b\rangle|S_{c+ab}\rangle \tag{4.28}$$

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

$$\begin{aligned}
|S_0\rangle|S_0\rangle|S_0\rangle &\longmapsto |S_0S_0S_0\rangle \\
|S_0\rangle|S_1\rangle|S_0\rangle &\longmapsto |S_0S_1S_0\rangle \\
|S_1\rangle|S_0\rangle|S_0\rangle &\longmapsto |S_1S_0S_0\rangle \\
|S_1\rangle|S_1\rangle|S_0\rangle &\longmapsto |S_1S_1S_1\rangle
\end{aligned} \tag{4.29}$$

A Toffoli gate can be generated from transformation (4.29) 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 (4.29) 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:

$$\begin{aligned}
|S_0\rangle|S_0\rangle|S_0\rangle|S_0\rangle &\longmapsto \frac{1}{\sqrt{2}}|S_0S_0S_0\rangle(|S_0\rangle + |S_1\rangle) \\
|S_0\rangle|S_1\rangle|S_0\rangle|S_0\rangle &\longmapsto \frac{1}{\sqrt{2}}|S_0S_1S_0\rangle(|S_0\rangle + |S_1\rangle) \\
|S_1\rangle|S_0\rangle|S_0\rangle|S_0\rangle &\longmapsto \frac{1}{\sqrt{2}}|S_1S_0S_0\rangle(|S_0\rangle + |S_1\rangle) \\
|S_1\rangle|S_1\rangle|S_0\rangle|S_0\rangle &\longmapsto \frac{1}{\sqrt{2}}|S_1S_1S_1\rangle(|S_0\rangle + |S_1\rangle) \\
|S_0\rangle|S_0\rangle|S_1\rangle|S_0\rangle &\longmapsto \frac{1}{\sqrt{2}}|S_0S_0S_1\rangle(|S_0\rangle - |S_1\rangle) \\
|S_0\rangle|S_1\rangle|S_1\rangle|S_0\rangle &\longmapsto \frac{1}{\sqrt{2}}|S_0S_1S_1\rangle(|S_0\rangle - |S_1\rangle) \\
|S_1\rangle|S_0\rangle|S_1\rangle|S_0\rangle &\longmapsto \frac{1}{\sqrt{2}}|S_1S_0S_1\rangle(|S_0\rangle - |S_1\rangle) \\
|S_1\rangle|S_1\rangle|S_1\rangle|S_0\rangle &\longmapsto \frac{1}{\sqrt{2}}|S_1S_1S_0\rangle(|S_0\rangle - |S_1\rangle)
\end{aligned} \tag{4.30}$$

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

$$|S_a\rangle|S_b\rangle|S_c\rangle \longmapsto (-1)^{ab+c}|S_a\rangle|S_b\rangle|S_c\rangle \tag{4.31}$$

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 (4.29). 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:

$$\begin{aligned}
|S_0\rangle|S_0\rangle|A\rangle &\longmapsto \frac{1}{2}(|S_0S_0S_0S_0S_0\rangle + |S_0S_1S_0S_1S_0\rangle + |S_1S_0S_1S_0S_0\rangle + |S_1S_1S_1S_1S_1\rangle)\\
|S_0\rangle|S_1\rangle|A\rangle &\longmapsto \frac{1}{2}(|S_0S_0S_0S_1S_0\rangle + |S_0S_1S_0S_0S_0\rangle + |S_1S_0S_1S_1S_1\rangle + |S_1S_1S_1S_0S_0\rangle)\\
|S_1\rangle|S_0\rangle|A\rangle &\longmapsto \frac{1}{2}(|S_0S_0S_1S_0S_0\rangle + |S_0S_1S_1S_1S_1\rangle + |S_1S_0S_0S_0S_0\rangle + |S_1S_1S_0S_1S_0\rangle)\\
|S_1\rangle|S_1\rangle|A\rangle &\longmapsto \frac{1}{2}(|S_0S_0S_1S_1S_1\rangle + |S_0S_1S_1S_0S_0\rangle + |S_1S_0S_0S_1S_0\rangle + |S_1S_1S_0S_0S_0\rangle)
\end{aligned} \tag{4.32}$$

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:

$$\begin{aligned}
|0,0\rangle|a,b,c\rangle &\longmapsto |0,0\rangle|a,b,c\rangle\\
|0,1\rangle|a,b,c\rangle &\longmapsto |0,1\rangle NOT(2)CNOT(1,2)|a,b,c\rangle\\
|1,0\rangle|a,b,c\rangle &\longmapsto |1,0\rangle NOT(1)CNOT(2,3)|a,b,c\rangle\\
|1,1\rangle|a,b,c\rangle &\longmapsto |1,1\rangle NOT(3)CNOT(1,3)CNOT(2,3)|a,b,c\rangle
\end{aligned} \tag{4.33}$$

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. Moreover, the procedures are shorter, and require less operations. Thus, as we will see later, the threshold which we achieve using polynomial codes is significantly better than that achieved using CSS codes. 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 bitwise, 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* [9] 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 5. *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 4 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.

5.1.1. 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 $1 \leq i \leq m$s, 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.

5.1.2. Fault tolerant Procedure of General Phase Rotation. Define c_l as the interpolation coefficients such that

$$\forall\ f \in F[x],\ deg(f) \leq m-1, f(0) = \sum_{i=1}^{m} c_i f(\alpha_i). \tag{5.1}$$

To achieve a rotation by w^{ca}, we apply on the $l't$

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, so:

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

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*[9], is best illustrated in the following figure:

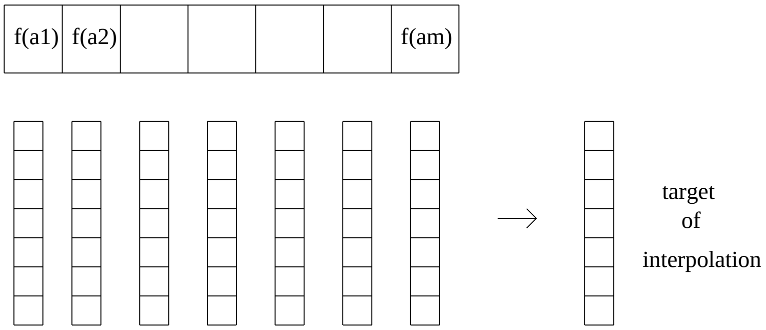


FIG. 5.1. *Scheme for degree reduction.*

The idea goes as follows. Let

$$\sum_j c_j f(\alpha_j) = f(0) \tag{5.11}$$

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 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

$$\sum_j c_j a_{i,j} \tag{5.12}$$

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'

$$\mathcal{E}_{S_b(C')} : |S_a(C')\rangle \longmapsto |S_{a+1}(C')\rangle \tag{5.13}$$

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

$$\mathcal{E}_{S_f(C')} : |S_a(C')\rangle \longmapsto w^a|S_a(C')\rangle \tag{5.14}$$

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 4.9, 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

$$\frac{1}{\sqrt{p}}\sum_{i=0}^{p-1}|S_a(C')\rangle \tag{5.15}$$

as is shown in section 5.2.1. We then copy it on $l-1$ ancilla states $|S_0(C')\rangle$ using generalized CNOT gates pit-wise. This generates the state

$$|S_{cat_l}\rangle=\frac{1}{\sqrt{p}}\sum_{i=1}^{p-1}|S_i(C')\rangle|S_i(C')\rangle\cdots|S_i(C')\rangle \tag{5.16}$$

(It might seem as if generating encoded cat states is much simpler than generating regular cat states, as we do in section 5.1.3, 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 5.4, using generalized Fourier transforms pit-wise. We thus have:

$$\sum_{j,j\cdot\vec{1}=0\ mod\ p}|S_{j_1}(C')\rangle\cdots|S_{j_l}(C')\rangle \tag{5.17}$$

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

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 1. The proof of theorem 2 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:

$$(5.18)\quad \mathcal{H} \circ \mathcal{E}_{S_f(C')} |S_a(C')\rangle = \mathcal{H}(-1)^a |S_a(C')\rangle = (-1)^a |C_a(C'')\rangle = \mathcal{E}_{S_b(C'')} |C_a(C'')\rangle = \mathcal{E}_{S_b(C'')} \circ \mathcal{H} |S_a(C')\rangle$$

Hence, the sequence of equalities 3.18 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.*[12]. 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 14. *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:

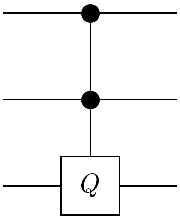
THEOREM 6. *Kitaev, Solovay,:*

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 $p_m(log^c(1/\epsilon))$, and this word can be found efficiently (in time which is $p_m(log^c(1/\epsilon))$) by a classical Turing machine.

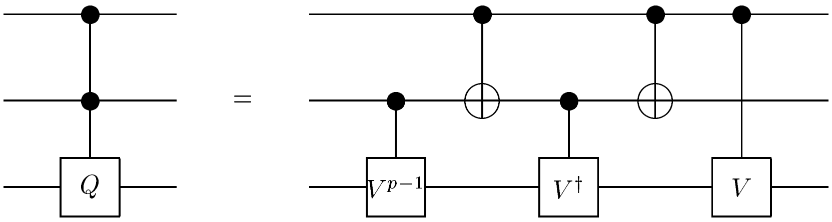
The proof of this theorem[39, 66] uses Lie groups and Lie Algebras, and will not be discussed here. The second result which we will need to justify the choice of definition of universality, is that an operation on any number of qubits (qupits) can be achieved using gates on two qubits (qupits). This was proved for qubits by DiVincenzo[26], and simplified by Barenco *et. al.*[6]. We give here a proof for the general case of qupits.

THEOREM 7. *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[24] and Barenco *et. al*[6]. 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 $T_{m,k}(Q)$. Following Barenco *et. al.* [6] 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)

$$Pit - flip : |a\rangle \longmapsto |(a+1)\text{mod}p\rangle \tag{6.1}$$

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:

$$U_k|\psi_j\rangle = \begin{cases} |\psi_j\rangle & \text{if } k \neq j \\ e^{i\theta_k}|\psi_k\rangle & \text{if } k = j \end{cases} \tag{6.2}$$

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. □

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[39] 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 1. *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[24]. 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:

$$U_k|\psi_j\rangle = \begin{cases} |\psi_j\rangle & \text{if } k \neq j \\ e^{i\theta_k}|\psi_k\rangle & \text{if } k = j \end{cases} \tag{6.3}$$

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 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 1. 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 1 can be applied, and we have $U(A_i)$. □

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[39], who used lemmas 1 and 2.

THEOREM 8. *$\mathcal{G}_1$ 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[39], 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'_3s, as follows:

$$\begin{aligned} &a, b, c, d, e && \longmapsto \\ &a, b, c + ab, d, e && \longmapsto \\ &a, b, c + ab, d, e + cd + abd && \longmapsto \\ &a, b, c, d, e + cd + abd && \longmapsto \\ &a, b, c, d, e + abd && \end{aligned} \tag{6.5}$$

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[39], who used the geometrical lemmas 1 and 2 to show that CP

and the Hadamard gate are universal. Here is Kitaev's argument. Denote by

$$X_1 = H_1(CP)_{1,2}H_1 \tag{6.6}$$

$$X_2 = H_2(CP)_{2,1}^{-1}H_2$$

Define $Y_1 = X_1X_2^{-1}$ and $Y_2 = X_2X_1^{-1}$. 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 2 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 3 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 7. □

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 9). We will first prove an analogue of the fact that one qubit gates and classical gates are universal for qubits[].

LEMMA 4. *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 4 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 3. First, we claim that the subspaces of the above form can be ordered, $A_1, ...A_m$ such that

$$\oplus_{i=1}^{j-1} A_i \not\perp A_j \tag{6.7}$$

as is easy to check. Second,

$$\oplus_{i=1}^{m} A_i = C^{p^k} \tag{6.8}$$

Since $U(A_i) \subset G$, the proof follows from lemma 3. □

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:

$$Q_0 = \begin{pmatrix} w & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{pmatrix} \tag{6.9}$$

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 5. *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:

$$\begin{aligned} |i\rangle|a\rangle|b\rangle &\longmapsto w|i\rangle|a\rangle|b\rangle \\ |j\rangle|a\rangle|b\rangle &\longmapsto \ \ |j\rangle|a\rangle|b\rangle \quad j \neq i \end{aligned} \tag{6.10}$$

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

$$P^{-1} \cdot CB^{-1} \cdot P \cdot CB. \tag{6.11}$$

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. □

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

$$X_i = HQ_iH^{-1}Q_i^{-1} \quad , \quad Y_i = HQ_i^{-1}H^{-1}Q_i \tag{6.12}$$

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

$$S_i = span\{|i\rangle, \sum_{b \in F_p, b \neq i} w^{ib}|b\rangle\}. \tag{6.13}$$

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

Lemma 6. *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 $Y_i = HQ_i^{-1}H^{-1}Q_i$.

$$(X_i)_{ab} = \begin{cases} \delta_{ab} + \frac{1}{p}(w-1)w^{(a-b)i} & \text{if } b \neq i \\ \delta_{ab} + \frac{1}{p}(w-1)w^{(a-b)i}]w^{-1} & \text{if } b = i \end{cases} \tag{6.14}$$

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

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 6.14 where we substruct 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:

$$v_i = 0 \quad , \quad \sum_{b, b\neq i} v_b w^{-ib} = 0 \tag{6.15}$$

and thus

$$\begin{aligned} \sum_{b\in F_p} (X_i - I)_{ab} v_b = &\ \frac{1}{p}(w-1)w^{ai} \sum_{b\neq i} v_b w^{-bi} = 0 \\ \sum_{b\in F_p} (Y_i - I)_{ab} v_b = &\ \frac{1}{p}(w^{-1}-1)w^{ai} \sum_{b\neq i} v_b w^{-bi} = 0. \quad \square \end{aligned} \tag{6.16}$$

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 2, 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[73].

LEMMA 7. *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 orthonornal 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 6.14 and a little algebra we get:

$$\begin{aligned} X_i|i\rangle &= (1 + \frac{1}{p}(w-1))w^{-1}|i\rangle + \frac{\sqrt{p-1}}{p}(w-1)w^{-i^2-1}|\alpha_i\rangle \\ X_i|\alpha_i\rangle &= (\frac{\sqrt{p-1}}{p}(w-1)w^{i^2})|i\rangle + (1 + \frac{p-1}{p}(w-1))|\alpha_i\rangle \end{aligned} \tag{6.17}$$

and for Y_i we get the transformation:

$$\begin{aligned} Y_i|i\rangle &= (1 + \frac{1}{p}(w^{-1}-1))w|i\rangle + \frac{\sqrt{p-1}}{p}(w^{-1}+1)w^{-i^2-1}|\alpha_i\rangle \\ Y_i|\alpha_i\rangle &= (\frac{\sqrt{p-1}}{p}(w^{-1}-1)w^{i^2})|i\rangle + (1 + \frac{p-1}{p}(w^{-1}-1))|\alpha_i\rangle \end{aligned} \tag{6.18}$$

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:

$$f(\lambda) = \lambda^2 - \frac{2 + (p-1)(w + w^{-1})}{p}\lambda + 1 \tag{6.19}$$

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

$$\frac{2+(p-1)(w+w^{-1})}{p} = \zeta_n + \zeta_n^{-1} \tag{6.20}$$

We will first prove that $n = p$. Denote by $Q(w), Q(\zeta_n)$ the Galois extentions 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 6.20, $Q(\zeta_n)^+ = Q(w)^+$.

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 [73]. 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:

$$deg(Q(w)^+/Q) = \frac{p-1}{2} \quad , \quad deg(Q(\zeta_n)^+/Q) = \frac{\phi(n)}{2} \tag{6.21}$$

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 [73]) and so p must devide n, say $n = p^r m$, with m coprime to p. This implies:

$$\phi(n) = p^{r-1}(p-1)\phi(m) = p-1. \tag{6.22}$$

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

$$\frac{1+(p-1)cos(2\pi/p)}{p} = cos(k*(2\pi/p)). \tag{6.23}$$

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 9. *$\mathcal{G}_2$ 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 5. 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 2, using the fact that the eigenvalues are not roots of unity, by lemma 7. 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 3 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 4, which gurantees universality on two qupits, and theorem 7 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 15. *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 8. Note that the procedures for CSS codes and polynomial codes satisfy the above restrictions. We define the spread of the computation code.

Definition 16. *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.

$$l \leq d. \tag{7.1}$$

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[53].

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[63], 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 2.7. 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 17. *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 fault paths keep the errors sparse. We will show in lemma 8 that this is guaranteed if the fault path is sparse:

Definition 18. *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 8. *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 $\mathfrak{g}_0$ on ρ_0, with an (r,k) sparse set of faults, the density matrix is (r,d) deviated from $\phi^r(\mathfrak{g}_0\circ\rho_0)$.
2. Consider n $r-$blocks, in any density matrix, ρ_r. At the end of an r-working period which $r-$simulates the operation $\mathfrak{g}_0$ on ρ_0, with an (r,k) sparse set of faults, the density matrix is (r,d) deviated from some word $\phi^r(\mathfrak{g}_0\circ\rho_0)$, 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 $\phi^{r+1}(\rho_0)$. □

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 9. *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

$$\sum_{j,\beta_j|_A=\alpha_i|_A} q_j = p_i. \tag{7.2}$$

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. □

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 19. *The threshold condition:*

$$\left(\begin{array}{c} A \\ k+1 \end{array}\right) \eta^{k+1} < \eta \tag{7.3}$$

We can thus define the threshold of a code:

DEFINITION 20. *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*

$$\eta_c(C) = \begin{pmatrix} A \\ k+1 \end{pmatrix}^{-k}. \tag{7.4}$$

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. Moreover, in our constructions of the procedures we did not attempt to optimize their size. 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, 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 10. *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.

$$\begin{pmatrix} A \\ k+1 \end{pmatrix} \eta^{k+1} < \eta^{1+\delta}. \tag{7.5}$$

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. □

We can now prove the threshold theorem.

Theorem 10. **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 10, we have that the probability for a fault path to be bad is smaller than ϵ. By lemmas 8 and 9 the sparse fault paths give correct outputs. □

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:

$$\rho(t) = \mathcal{E}(t) \cdot \mathcal{L}(t) \cdot \mathcal{E}(t-1) \cdot \mathcal{L}(t-1) \cdots \mathcal{E}(0) \cdot \mathcal{L}(0)\rho(0). \tag{8.1}$$

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 2.3, $\mathcal{E}(t)$ 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 2.4:

$$\mathcal{E}_{A_{i,t}}(t) = (1-\eta)I + \mathcal{E}'_{A_{i,t}}(t) \quad , \quad \|\mathcal{E}'_{A_{i,t}}(t)\| \leq 2\eta. \tag{8.2}$$

We can replace all the error operators in equation 8.1 by the products of operators of the form 8.2. We get:

$$\begin{aligned} \rho(t) = {} & \Big(\otimes_{A_{i,t}}((1-\eta)I + \mathcal{E}'_{A_{i,t}})\Big) \cdot \mathcal{L}(t) \cdot \Big(\otimes_{A_{i,t-1}}((1-\eta)I + \mathcal{E}'_{A_{i,t-1}})\Big) \cdot \mathcal{L}(t-1) \cdots \\ & \cdots \Big(\otimes_{A_{i,0}}((1-\eta)I + \mathcal{E}'_{A_{i,t-1}})\Big) \mathcal{L}(0)\rho(0). \end{aligned} \tag{8.3}$$

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 8.3 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:

$$\rho(t) = \mathcal{L}_g \cdot \rho(0) + \mathcal{L}_b \cdot \rho(0) \tag{8.4}$$

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

$$e\begin{pmatrix} A \\ k+1 \end{pmatrix}(2\eta)^{k+1} < 2\eta \tag{8.5}$$

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 22. *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*

$$\eta'_c(C) = \frac{1}{2} e^{-k} \left(\begin{array}{c} A \\ k+1 \end{array} \right)^{-k}. \tag{8.6}$$

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

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

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:

$$\rho(t) = (\mathcal{L}_g(v) + \mathcal{L}_b(v)) \cdot (\mathcal{L}_g(v-1) + \mathcal{L}_b(v-1)) \cdots (\mathcal{L}_g(1) + \mathcal{L}_b(1))\rho(0) \tag{8.7}$$

We first prove that

$$\forall 1 \leq i \leq v \quad , \quad \|\mathcal{L}_b(i)\| \leq (2\eta)^{(1+\delta)^r} \quad , \quad \|\mathcal{L}_g(i)\| \leq 1 + (2\eta)^{(1+\delta)^r} \tag{8.8}$$

Where δ is defined by

$$e \left(\begin{array}{c} A \\ k+1 \end{array} \right) (2\eta)^{k+1} < (2\eta)^{1+\delta}. \tag{8.9}$$

Such δ exists because η is below the threshold. The proof of inequality 8.8 for $\eta < \eta'_c$, follows the lines of lemma 10. We use induction on r, and denote by $\mathcal{L}^r_b(i)$ the sum over all bad fault paths in an $r-$procedure. For $r = 0$, a 0-procedure is simply one location, and $\mathcal{L}^0_b(i)$ 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 8.2, and the properties of the norm on super-operators in section 2.8 $\|\mathcal{L}^0_b(i)\| \leq 2\eta$, and $\|\mathcal{L}^0_g(i)\| \leq 1 + 2\eta$ 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 $\mathcal{L}_g(i)$

$$\|\mathcal{L}^{r+1}_b(i)\| \leq \left(\begin{array}{c} A \\ k+1 \end{array} \right) ((2\eta)^{(1+\delta)^r})^{k+1} (1 + (2\eta)^{(1+\delta)^r})^{A-k-1} \leq e \left(\begin{array}{c} A \\ k+1 \end{array} \right) ((2\eta)^{(1+\delta)^r})^{k+1}. \tag{8.10}$$

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 $2\eta A \le 1$. The right hand side is $\le (2\eta)^{(1+\delta)^{r+1}}$ using the fact that $(2\eta)^{(1+\delta)^r} < 2\eta$, and the threshold condition is satisfied for $\eta \le \eta_c'$. 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:

$$\|\mathcal{L}_b\| \le v \cdot (1+(2\eta)^{(1+\delta)^r})^{v-1}(2\eta)^{(1+\delta)^r} \tag{8.11}$$

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

8.3. The Good Part: Sparse Fault Paths Give Almost Correct Outputs. Lemma 12. *Let $\eta \le \eta_c', \epsilon > 0$. Let M_0 use v locations. Let $r = \text{polyloglog}(\frac{\text{v}}{\epsilon})$, as in lemma 11. Let M_0 output $f(i)$ for a given i with probability of error ϵ'. Then for M_r,*

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

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 8 can be applied. We will get the desired result from linearity. We write

$$\mathcal{L}_g \cdot \rho(0) = \sum_E \mathcal{L}_E \cdot \rho(0) \tag{8.12}$$

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:

$$\mathcal{E}'_{A_{i,t}} = \mathcal{E}_{A_{i,t}} - (1-\eta)I \tag{8.13}$$

Inserting this to equation 8.12, and we get a linear sum over terms which correspond to physical fault paths, with sparse faults.

$$\mathcal{L}_g \cdot \rho(0) = \sum_f \lambda_f \mathcal{L}_f \cdot \rho(0) \tag{8.14}$$

Lemma 8 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

$$\sum_{bad\ i} [\mathcal{L}_g \cdot \rho(0)]_{i,i} = \sum_f \lambda_f \sum_{bad\ i} [\mathcal{L}_f \cdot \rho(0)]_{i,i} = (\sum_f \lambda_f)\epsilon' = \text{Tr}(\mathcal{L}_g \cdot \rho(0))\epsilon' \le \|\mathcal{L}_g\|\epsilon \le (1+\epsilon)\epsilon', \tag{8.15}$$

using lemma 11 in the last inequality. □

8.4. The Threshold Theorem for General Noise. We can now prove the threshold result for general noise:

THEOREM 11. **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. □.

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 11 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.

$$\sum_{bad\ i} \rho(t)_{i,i} = \sum_{bad\ i} [\mathcal{L}_b \cdot \rho(0)]_{i,i} + \sum_{bad\ i} [\mathcal{L}_g \cdot \rho(0)]_{i,i} \tag{8.16}$$

And hence

$$|\sum_{bad\ i} \rho(t)_{i,i}| \leq \sum_{bad\ i} |[\mathcal{L}_b \cdot \rho(0)]_{i,i}| + |\sum_{bad\ i} [\mathcal{L}_g \cdot \rho(0)]_{i,i}| \leq \|\mathcal{L}_b \cdot \rho(0)\| + |\sum_{bad\ i} [\mathcal{L}_g \cdot \rho(0)]_{i,i}| \leq \epsilon'' + (1+\epsilon'')\epsilon' \tag{8.17}$$

The first inequality follows from the fact that $\sum_i |\rho_{i,i}| \leq \|\rho\|$ for any Hermitian matrix (see [4]. Lemmas 11 and 12 are used to derive the second inequality. Due to the choice of ϵ" This probability is indeed smaller than $\leq \epsilon + \epsilon'$. □

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 23. *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

$$\eta_c'' = \max_{\delta} \frac{\eta_c' - \delta}{S(\delta)} \tag{9.1}$$

$\eta < \eta_c''$ 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 12. **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'. $\Box$.

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. $\Box$

10. Robustness Against Exponentially Decaying Correlations. We would now like to show how the above results hold also in the case of exponentially decaying correlations between the noise processes, in both space and time, as is described in subsection 2.10. We observe that all the lemmas which we use to prove the threshold theorems hold in this case, except for one step which fails. It is the step that shows that bad fault paths are rare, in the case of probabilistic noise (lemma 10). The proof of this lemma rely on the independence of faults. We observe that the proof of lemma 10 is actually a union bounds. In this section we show that the same threshold as is used for probabilistic noise, (definition 7.4) guarantees that the bad fault paths are negligible also is the presence of exponentially decaying correlations, for probabilistic noise.

We use a union bound argument, as follows. Consider fault paths in v $r-$rectangles. If a fault path is bad, there must be at least one $r-$rectangle in which it is bad. In this rectangle, let us first count the number of minimal bad fault paths. If F_r is this number, we have:

(10.1)
$$F_1 \leq \left(\begin{array}{c} A \\ k+1 \end{array}\right)$$

and

(10.2)
$$F_{r+1} \leq \left(\begin{array}{c} A \\ k+1 \end{array}\right) F_r^{k+1}$$

We can solve the recursion, to get

(10.3)
$$F_r \leq \left(\begin{array}{c} A \\ k+1 \end{array}\right)^{\frac{(k+1)^r-1}{k}}$$

A minimal bad fault path contains exactly $(k+1)^r$ locations. Now, v r-rectangles contain vA^r locations. We can bound the number of bad fault paths in v r-rectangles consisting of $(k+1)^r+i$ locations by choosing one $r-$rectangle (This gives a factor of v). In this rectangle we pick one of the possible minimal bad fault paths (This gives a factor or F_r). We can then choose the rest of the locations arbitrarily. This gives that the number of bad fault paths in v r-rectangles consisting of $(k+1)^r+i$ locations is at most:

$$v\binom{A}{k+1}^{\frac{(k+1)^r-1}{k}}\binom{vA^r-(k+1)^r}{i} \tag{10.4}$$

Using the assumption 2.6 on the exponentially decaying probability of a fault path consisting of k locations, we have that the overall probability of the bad fault paths is at most:

$$\begin{aligned} &\sum_{i=0}^{vA^r-(k+1)^r} v\binom{A}{k+1}^{\frac{(k+1)^r-1}{k}}\binom{vA^r-(k+1)^r}{i} c\eta^{(k+1)^r+i}(1-\eta)^{vA^r-(k+1)^r-i} \leq \\ &cv(\binom{A}{k+1}^{\frac{1}{k}}\eta)^{(k+1)^r}\sum_{i=1}^{vA^r-(k+1)^r}\binom{vA^r-(k+1)^r}{i}\eta^i(1-\eta)^{vA^r-(k+1)^r-i} = \\ &cv(\binom{A}{k+1}^{\frac{1}{k}}\eta)^{(k+1)^r} \end{aligned} \tag{10.5}$$

This completes the proof for this model of noise, since the expression above decays exponentially fast to zero with r, if η is below the threshold for probabilistic noise from definition 7.4:

$$\binom{A}{k+1}\eta^k < 1. \tag{10.6}$$

11. 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 13. **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 $c_1 n \mathrm{log}^{c_2}(\frac{v}{\epsilon})$ qubits, for time $c_3 t \mathrm{log}^{c_2}(\frac{v}{\epsilon})$, and uses $c_4 s \mathrm{log}^{c_2}(\frac{v}{\epsilon})$ 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 $\mathfrak{g}$ in the procedure, which operates on qubits which are far apart, is replaced by a sequence of SWAP gates which bring the qubits $\mathfrak{g}$ operates on to nearest neighbor sites, followed by $\mathfrak{g}$, 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 $\mathfrak{g}$ 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. $\Box$.

12. Conclusions and Open Problems. The result implies that quantum computation might be practical if the noise in the system can be made very small. We hope these results motivate physicists to achieve lower error rates, and theoreticians to develop codes with better parameters, in order to push the threshold as high as possible. The point at which the physical data meets the theoretical threshold is where quantum computation becomes practical.

We did not attempt to optimize the threshold in the fault tolerant scheme. The reason is that this threshold depends on a lot of parameters, such as whether the system is $d-$dimensional or whether it has no geometry; Whether measurements are allowed or not, and which set of gates can be implemented in the laboratory. We did give a rough estimate of the threshold in one case, in which the length of the code is

$m = 7$, in both cases of 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 $\approx 10^{-6}$. This threshold is far from being practical, according to the state of the art, and optimization is certainly required. One can save space by distinguishing two types of qubits: regular qubits, and "classical" qubits which we do not need to protect from phase flips, but only from bit-flips. Such are the qubits involved in the classical computation of the error given the syndrome. These qubits can be encoded using classical error corrections. In practical cases, it might be possible to allow measurements on such qubits, during the computation, and apply noiseless classical computation instead of quantum computation on these qubits, followed by quantum gates conditioned on the classical results of these computations. Such ideas might reduce the threshold by several orders of magnitude.

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[67], to reduce the multiplicative factor of $O(log^c(n))$ to a factor of $O(log^c(log^c(n))$. An open question is whether it is possible to reduce the time cost to a constant, as in the classical case. We conjecture that the answer is negative.

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. Results in this direction were found by Lidar Chuang and Whaley[48].

The threshold result might have an impact on a long standing question in quantum physics, regarding the transition from quantum to classical physics[76]. Traditionally, this transition is treated by taking the limit of Planck's constant to 0, and it is viewed as a gradual transition (but see [38]). In [3] it was shown that for a very high noise rate, the quantum circuit behaves in a classical way. It is interesting to consider a different point of view, in which the definition of quantum versus classical behavior is computational. In this paper we show that for very small noise rate, quantum systems can maintain their quantum nature. In a previous paper, we have shown that quantum systems can be simulated efficiently by a classical Turing machine if the noise is large[3]. Suppose that indeed quantum behavior cannot be simulated efficiently by classical systems, an idea suggested by Feynmann[27] 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. Does this transition happen at a critical error rate? Indications for a positive answer are already shown in a previous paper of ours [3], which might suggest that the transition from quantum to classical physics occurs via a phase transition. We view this connection between quantum complexity and quantum physics as extremely interesting.

13. Acknowledgments. We wish to thank Peter Shor for discussions about his result, Thomas Beth for helping to construct the fault tolerant Toffoli gate without measurements for the CSS codes, Richard Cleve for asking the question of nearest neighbors and suggesting the solution, Noam Nisan for a fruitful discussion regarding the correct model to use, Prasad Gospel, Denis Gaitsgori and Erez Lapid for helpful discussions regarding lemmas 1 and 2. We are grateful to Bob Solovay for detecting and correcting crucial errors in lemmas 1, 2 and 3 in an early draft of this paper. Finally, we thank Alesha Kitaev for a very helpful remark regarding the generalization to the general noise model. This research was supported by The Israel

Science Foundation, grant number 69/96, and the Minerva Leibniz Center at the Hebrew University.

REFERENCES

[1] Aharonov D, Quantum computation, In *annual reviews of computational physics*, Vol VI, ed. Dietrich Stauffer, World Scientific, 1999.

[2] Aharonov D and Ben-Or M, Fault-Tolerant Quantum Computation with Constant Error, *Proc. of the 29th Annual ACM Symposium on Theory of Computing (STOC)* 1997

[3] Aharonov D and Ben-Or M, Polynomial Simulations of Decohered Quantum Computers *37th Annual Symposium on Foundations of Computer Science* (FOCS) pp 46–55, 1996

[4] 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

[5] Aharonov D, Ben-Or M, Impagliazo R and Nisan N, Limitations of Noisy Reversible Computation, in *LANL e-print* quant-ph/9611028, http://xxx.lanl.gov (1996)

[6] 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

[7] Barenco A, Ekert A, Suominen K A and Torma P, Approximate quantum Fourier transform and decoherence, *Phys. Rev. A* **54**, 139–146, 1996

[8] Bennett C H, DiVincenzo D P, Smolin J A and Wootters W K Mixed state entanglement and quantum error correction, *Phys. Rev. A* **54** 3825, 1996

[9] 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.

[10] Bernstein E and Vazirani U, 1993, Quantum complexity theory, *SIAM Journal of Computation* **26** 5 pp 1411–1473 October, 1997

[11] 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

[12] P. Oscar Boykin, Tal Mor, Matthew Pulver, Vwani Roychowdhury, and Farrokh Vatan, "On Universal and Fault-Tolerant Quantum Computing", in preparation.

[13] Calderbank A R and Shor P W, Good quantum error-correcting codes exist, *Phys. Rev. A* **54** 1098-1105, 1996

[14] 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

[15] Calderbank A R, Rains E M, Shor P W and Sloane N J A, Quantum error correction via codes over $GF(4)$ in *LANL e-print* quant-ph/9608006, http://xxx.lanl.gov (1996), To appear in *IEEE Transactions on Information Theory.*

[16] Choi M. D, linear algebra and its applications, **10** 286 (1975),

[17] Chuang I L, Laflamme R, Shor P W and Zurek W H, Quantum computers, factoring, and decoherence, *Science* **270** 1633–1635, 1995

[18] I.Chuang and W.C.D. Leung and Y. Yamamoto, Bosonic Quantum Codes for Amplitude Damping, in *Phys. Rev. A*, **56**, 2, (1997) pp. 1114-1125

[19] Cirac I J and Zoller P Quantum computations with cold trapped ions, *Phys. Rev. Let.*, 74: 4091-4094, 1995.

[20] 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

[21] C. Cohen-Tanoudji, *Quantum Mechanics*, Wiley press, New York (1977)

[22] 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

[23] Deutsch D, Quantum theory, the Church-Turing principle and the universal quantum computer, In *Proc. Roy. Soc. Lond.* A **400** 97-117, 1985

[24] Deutsch D, Quantum computational networks, In *Proc. Roy. Soc. Lond.* A **425** 73-90, 1989

[25] Deutsch D and Jozsa R, Rapid solution of problems by quantum computation, In *Proc. Roy. Soc. Lond* A **439** 553-558, 1992

[26] DiVincenzo D P, Two-bit gates are universal for quantum computation, *Phys. Rev. A* **51** 1015-1022 1995

[27] Feynman R P Simulating physics with computers, In *Int. J. Theor. Phys.* **21** 467-488, 1982

[28] 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

[29] Gardiner C W, Quantum Noise, Springer-Verlag, Berlin, 1991

[30] Gershenfeld N A and Chuang I L Bulk spin-resonance quantum computation, *Science*, 275:350–356, 1997.

[31] Gottesman D 1996 Class of quantum error-correcting codes saturating the quantum Hamming bound, Phys. Rev. A **54**, 1862-1868

[32] Gottesman D A theory of fault-tolerant quantum computation, in *Phys. Rev. A*,**57** 127–137

[33] D. Gottesman, 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.

[34] D. Gottesman, Fault-Tolerant Quantum Computation with Local Gates, quant-ph/9903099.

[35] D. Gottesman, How to share a quantum secret, Richard Cleve, Daniel Gottesman, Hoi-Kwong Lo, quant-ph/9901025

[36] Grover L K, Quantum mechanics helps in searching for a needle in a haystack, 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

[37] A. J. Jones, M. Mosca and R. H. Hansen, Implementation of a Quantum Search Algorithm on a Nuclear Magnetic Resonance Quantum Computer, in *Nature* 393 (1998) 344-346, and see also A. J. Jones and M. Mosca, Implementation of a Quantum Algorithm to Solve Deutsch's Problem on a Nuclear Magnetic Resonance Quantum Computer, in *J. Chem. Phys.* 109 (1998) 1648-1653

[38] 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

[39] Kitaev A Yu, Quantum Computations: Algorithms and Error Corrections, in *Russian Math. Serveys*, **52**:6, 1191-1249

[40] 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.

[41] Kitaev A Yu 1997 Fault-tolerant quantum computation by anyons, in *LANL e-print* quant-ph/9707021, http://xxx.lanl.gov (1997)

[42] 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)

[43] Knill E and Laflamme R 1997 A theory of quantum error-correcting codes, Phys. Rev. A **55** 900-911

[44] E. Knill, Non-Binary Unitary Error Bases and Quantum Codes, in *LANL e-print* quant-ph/9608048, http://xxx.lanl.gov (1996)

[45] 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.

[46] Laflamme R, Miquel C, Paz J P and Zurek W H 1996 Perfect quantum error correcting code, Phys. Rev. Lett. **77**, 198-201

[47] Landauer R. Is quantum mechanics useful? *Phil. Trans. Roy. Soc. of London*, 353:367–376, 1995.

[48] Daniel A. Lidar, Isaac L. Chuang and K. Birgitta Whaley (1) Phys. Rev. Lett. 81, 2594-7 (1998)

[49] van Lint J H Coding Theory, Springer-Verlag, 1982

[50] Lloyd S 1993 A potentially realisable quantum computer, Science **261** 1569; see also Science **263** 695 (1994).

[51] Loss D and DiVincenzo D P Quantum Computation with Quantum Dots, in *Phys. Rev. A*,**57**,1, pp. 120-126, 1997

[52] Miquel C, Paz J P and Perazzo 1996 Factoring in a dissipative quantum computer Phys. Rev. A **54** 2605-2613, and: Miquel C, Paz J P and Zurek W H 1997 Quantum computation with phase drift errors, Phys. Rev. Lett. **78** 3971-3974

[53] von Neumann, Probabilistic logic and the synthesis of reliable organisms from unreliable components, in *automata studies(Shanon,McCarthy eds)*, 1956

[54] Palma G M, Suominen K-A & Ekert A K 1996 Quantum computers and dissipation, Proc. Roy. Soc. Lond. A **452** 567-584

[55] 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

[56] Peres A 1993 *Quantum theory: concepts and methods* (Kluwer Academic Press, Dordrecht)

[57] Prasad G., private communication

[58] Preskill J 1997 Fault tolerant quantum computation, in *LANL e-print* quant-ph/9712048, http://xxx.lanl.gov (1997), to appear in *Introduction to Quantum Computation*, edited by H.-K. Lo, S. Popescu, and T. P. Spiller

[59] Privman V, Vagner I D and Kventsel G 1997 Quantum computation in quantum-Hall systems, in *Phys. Lett. A*, 239 (1998) 141-146

[60] J.J.Saqurai Modern Quantum Mechanics, revised edition. Addison Wesley, 1994

[61] Schumacher B W and Nielsen M A 1996 Quantum data processing and error correction Phys Rev A **54**, 2629

[62] 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

[63] Shor P W, Scheme for reducing decoherence in quantum computer memory, *Phys. Rev. A*, 52: 2493-2496, 1995.

[64] 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.

[65] Simon D 1994 On the power of quantum computation, *SIAM J. Comp.*, **26**, No. 5, pp 1474–1483, October 1997

[66] R Solovay and A. C-C Yao, preprint, 1996

[67] Daniel A. Spielman, Highly Fault-Tolerant Parallel Computation Proceedings of the 37th Annual IEEE Conference on Foundations of Computer Science, 1996, pp. 154–163.

[68] Steane A, Multiple particle interference and quantum error correction, *Proc. Roy. Soc. of London A*, 452:2551-2577, 1996.

[69] Steane A M The ion trap quantum information processor, Appl. Phys. B **64** 623-642 1997

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

[71] 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.

[72] Unruh W G, Maintaining coherence in quantum computers, *Phys. Rev. A*, 51:992–997, 1995.

[73] L. C. Washington, Introduction to Cyclotomic Fields, Spinger-Verlag, Graduate Texts in Mathematics 83, 1982.

[74] Wootters W K and Zurek W H 1982 A single quantum cannot be cloned, Nature **299**, 802

[75] Yao A C-C, Quantum circuit complexity, in *33th Annual Symposium on Foundations of Computer Science(FOCS)*, (1993) pp. 352–361

[76] Zurek W H, Decoherence and the transition from quantum to classical Physics Today 44(10), October, 1991 36–44.