Fault-Tolerant Quantum Computation With Constant Error

D. Aharonov* M. Ben-Or†

Abstract

In the past year many developments have taken place in the area of quantum error corrections. Recently Shor showed how to perform fault tolerant quantum computation when, η, the probability for a fault in one time step per qubit or per gate, is polylogarithmically small. This paper closes the gap and shows how to perform fault tolerant quantum computation when the error probability, η, is smaller than some constant threshold, η_0. The cost is polylogarithmic in time and space, and no measurements are used during the quantum computation. The same result is shown also for quantum circuits which operate on nearest neighbors only.

To achieve this noise resistance, we use concatenated quantum error correcting codes. The scheme presented is general, and works with any quantum code, that satisfies certain restrictions, namely that it is a "proper quantum code". The constant threshold η_0 is a function of the parameters of the specific proper code used.

We present two explicit classes of proper quantum codes. The first class generalizes classical secret sharing with polynomials. The codes are defined over a field with p elements, which means that the elementary quantum particle is not a qubit but a "qupit". The second class uses a known class of quantum codes and converts it to a proper code.

We estimate the threshold η_0 to be $\simeq 10^{-6}$. Hopefully, this paper motivates a search for proper quantum codes with higher thresholds, at which point quantum computation becomes practical.

1 Introduction

Quantum computation[11, 10, 31] is believed to be more powerful than classical computation, due to oracle results[26, 6] and Shor's algorithm[24]. It is yet unclear whether and how quantum computers will be physically realizable,[20, 12, 8] but as any physical system, they *in principle* will be subjected to noise, such as decoherence[30, 29, 21], and inaccuracies. Without error corrections, the effect of noise can accumulate and ruin the entire computation[29, 7], hence the computation must be protected. Even the simpler question of protecting quantum information is harder than the classical analog because one must also protect the quantum correlations between the quantum bits (qubits). However, it was shown [4, 28] that good quantum error correcting codes exist, a result which was followed by many explicit examples(ex: [22, 19]). This does not trivially 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. Recently Shor[23] showed how to use quantum codes in order to perform fault tolerant quantum computation when the *noise rate*, or the fault probability each time step, per qubit or gate, is polylogarithmically small. In this paper we improve this result and show how to perform fault tolerant quantum computation in the presence of constant noise rate, which is what one might expect in a real physical system. The cost is polylogarithmic cost in time and size.

The error corrections which have been described so far use a combination of classical and quantum operations. We would like to define a model of noisy quantum computation[1], such that errors and error corrections can be described entirely inside this model. In this paper we are able to prove the main result due to working in such a formal framework. Sequential quantum computation can not be noise resistant[1], so we work with quantum circuits[10, 31]. As the state of a noisy quantum system is in general a probability distribution over pure states, i.e. a *mixed state*[16], and not merely a pure state as in the standard model, we use quantum circuits with mixed states[3]. Since noise is a dynamic process which depends on time, the circuit will be divided to levels, or time steps. Unlike Yao[31] we allow qubits to be input and output at different times to the circuit. This is crucial, since with the restriction that all qubits are initialized at time 0, it is not possi-

*Institutes of Physics and Computer science, The Hebrew University, Jerusalem, Israel, E-mail: doria@cs.huji.ac.il

†Institute of Computer science, The Hebrew University, Jerusalem, Israel, E-mail: benor@cs.huji.ac.il

ble to compute fault tolerantly without an exponential blowup in the size of the circuit[2]. Between the time steps, we add the noise process, which is a probabilistic process: each qubit or gate undergoes a fault with independent probability η per step, and η is referred to as the *noise rate*. The list of times and places where faults had occurred, namely the *fault path*, is random, and naturally, the function that the circuit computes is a weighted average over the noise process.

Let us first describe how one can protect quantum information against noise, using quantum error correcting codes[4, 28]. As in classical linear block codes, a quantum error correcting code spreads the state of each qubit on a number of qubits, called a "block". The code is said to correct d errors if the whole state is recoverable given that not more than d errors occurred in each block. The difference from classical codes is that the quantum correlations should be recovered as well, and not only the logical states of the qubits. It turns out that applying classical error corrections in one basis of the Quantum space can correct the logical states, while applying classical error corrections in some other basis corresponds to corrections of the quantum correlations.

In order to protect a quantum computation against faults, one can try to compute on encoded states, and not on the states themselves, using quantum codes. The circuit M_1 which will compute on encoded states is defined as a simulation of the original circuit M_0. A qubit in the original circuit transforms to a block of qubits in the simulating circuit, and each time step transforms to a *working period* of several time steps. To simulate the computation done in the $t'th$ time step in M_0, we will apply in M_1 the analog computation, on encoded states. Each gate in M_0 will transform to some "procedure" in M_1, which computes this gate on encoded states. The procedure might require ancilla qubits, so these are added to the circuit M_1, and are initialized only when needed. If M_1 is initialized with some encoded state, we expect this state to evolve by the computation along some "trajectory", such that at the end of each working period it encodes the correct corresponding state of M_0. The input and output to M_1 will simply be a duplication of the inputs and outputs of M_0, on the corresponding blocks. We will therefore need to add on each block, before computation begins, an *input procedure*, that transforms the duplicated input, i.e a string of $0's$ or a string of $1'$s, to the encoding state $|S_0>$, or $|S_1>$. At the end of the computation we will need the opposite transformation, so we will use an *output procedure* that transforms a block in the state $|S_0>$ or $|S_1>$ back to a string of 0's or $1'$s.

Computing on encoded quantum states does not automatically provide protection against faults, since errors accumulate, and when the damage has effected too many qubits in one block, the correct state is no longer recoverable. In order to prevent accumulation of errors, error corrections should be applied all the time, and so in each working period in M_1 we will add a step of error corrections of each block. The working period will be divided to 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. One should notice, however, that this "hope" does not always come true. During the computation faults can *spread* to places not in the fault path. 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. If for example, a gate procedure consists of one gate operating on the whole block, then even one fault can cause an uncorrectable error. The procedures should be designed in such a way that a fault can not effect all the qubits in the block. In general, a fault in qubit q at time t can effect qubit q' at time $t' > t$ if there is a path in the circuit connecting the points (q, t) and (q', t'). Define the "spread" of a procedure as the maximal number of qubits in each block in the output of the procedure, which are effected by a fault that happened 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 computation stage in each procedure. 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, input, and output procedures are also subjected to faults, we need them to have small spread too. Such codes, with the corresponding universal set of gates, will be called *quantum computation codes.*

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 computation to fail. In contrast, the simulating circuit can tolerate a number of errors, say k, in each procedure, since they are immediately corrected in the following stage. The effective noise rate of M_1 is thus a function of the probability for more than k errors in a procedure, and it will be smaller than the actual noise rate η, if the procedures are not too large.

However, it seems that an improvement from a constant noise rate to polynomially small effective noise rate, as we need for fault tolerance, is hard to achieve in the above scheme. In [23] Shor shows how to apply the above scheme to achieve fault tolerance with polylogarithmically small noise rate, η.

To improve this result, we use *concatenated simulations*, which generalizes the works of Tsirelson [9] and Gac's [14] to the quantum case. The idea is that the effective noise rate of the simulating circuit can be decreased by simulating it again, and so on for several levels. 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, if the noise rate is smaller than some constant threshold η_0.

The picture is hierarchical: 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.

We are interested in how far the state of the noisy final circuit is, from the correct one. Naturally, due to the hierarchical structure of this scheme, we define the metric 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 the state is close to the correct state in this metric, 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. A major portion of the effort in this paper is applied to showing that if the set of faults is sparse enough, the distance of the state from the correct state, in the above metric, is kept bounded. It will be rather easy to show that the probability for the set of faults not to be sparse decays exponentially with the number of levels. In order to show that, we use the fact that the places where errors occur are random. The result holds also in a more general noise model, in which for all integers k and for any set of k points, the probability that a fault occured in all the points is bounded by η^k, and an adversary can pick a transformation that operates on all the damaged qubits together.

Not any quantum computation code can be used in the above scheme. There are two restrictions:(1) When applying the simulation, we replace the gates by fault tolerant procedures. Since we want to simulate the new circuit as well, we need that these procedures use gates that can be replaced by fault tolerant procedures as well. Hence the universal set of gates associated with the code, must have fault tolerant procedures which use gates from the same universal set of gates. This is the "closeness" restriction. (2) Let us consider a two level simulation. If a simulated error correction operates on an encoded wrong word, it clearly corrects it. But what happens if it gets as an input some state which does not encode any word? The simulated error correction "understands" only encoded words. If we demand that error correction in the lower level corrects any state to some word in the code, then the input for the error correction of the upper level might be wrong but it will be an encoded word, so it can be understood and corrected by the upper level. The second restriction is therefore that the error correction takes any state to some word in the code. Quantum computation codes which satisfy both restrictions are called *proper quantum computation codes.*

We describe two classes of proper quantum computation codes, with constant block size. We first describe a generalization of the quantum codes in [4], to codes over F_p, with p prime. These codes are defined for general quantum circuits which consist of particles with $p \geq 2$ possible states. We call such quantum particles *qupits*, as a generalization to qubits. The proofs that these are quantum codes[4] transform smoothly from F_2 to F_p. The first example of proper codes is with $p = 2$. In this example the set of gates and procedures described in [23] is used, modified to fit the definition of proper quantum codes. For the second class of proper codes, we recall the result in [25], showing that a quantum error correction exists if and only if the environment gains no information about the unencoded data from the damaged qubits. Hence there is a strong connection between quantum error correction codes to secret sharing schemes that are used to perform secure fault-tolerant distributed computation [5]. The second class of quantum codes is thus the quantum analog of random polynomial codes[5]. To adopt the techniques of [5] to the quantum setting one can use the same encoding but instead of selecting a random polynomial to share a secret we pick the superposition of all those polynomials. The noise threshold turns out to be larger than $\simeq 10^{-6}$.

The results hold also for quantum circuits which are allowed to operate only on nearest neighbor qubits (In this case the threshold will be smaller.) The scheme applies also for corrections of random inaccuracies, but not for systematic errors. Similar results to those of this paper where independently discovered by Knill, Laflamme and Zurek[18]. Non binary codes where defined independently also by Chuang[15] and Knill[17].

2 Noisy Quantum Circuits

In this section we recall the definitions of quantum circuits[16, 10, 31] with mixed states[3], and define noisy quantum circuits[1].

2.1 Pure states

We deal with systems of n two-state quantum particles, or "qubits". The *pure state* of such a system is a unit vector, denoted $|\alpha\rangle$, in the 2^n dimensional complex space $\mathcal{C}^{2^n}$. We view $\mathcal{C}^{2^n}$ as a tensor product of n two dimensional spaces, each corresponding to a qubit: $\mathcal{C}^{2^n} = \mathcal{C}^2 \otimes ... \otimes \mathcal{C}^2$. As a basis for $\mathcal{C}^{2^n}$, we use the 2^n orthogonal *basic states*: $|i\rangle = |i_1\rangle \otimes |i_2\rangle .... \otimes |i_n\rangle, 0 \leq i < 2^n$, where i is in binary representation, and each i_j gets 0 or 1. A general unit vector $|\alpha\rangle$ in $\mathcal{C}^{2^n}$ is called a "pure state", and is a *superposition* of the basic states: $|\alpha\rangle = \sum_{i=1}^{2^n} c_i|i\rangle$, with $\sum_{i=5}^{2^n} |c_i|^2 = 1$. $|\alpha\rangle$ corresponds to the vector $v_\alpha = (c_1, c_2, ..., c_{2^n})$. $v_\alpha^\dagger$, the complex conjugate of v_α, is denoted $\langle\alpha|$. The inner product between $|\alpha\rangle$ and $|\beta\rangle$ is $\langle\alpha|\beta\rangle = (v_\alpha, v_\beta^\dagger)$. The matrix $v_\alpha^\dagger v_\beta$ is denoted as $|\alpha\rangle\langle\beta|$. An isolated system of n qubits develops in time by a unitary matrix, of size $2^n \times 2^n$: $|\alpha(t_2)\rangle = U|\alpha(t_1)\rangle$. A quantum system in $\mathcal{C}^{2^n}$ can be *observed* by *measuring* the system. An important measurement is a *basic measurement* of a qubit q, of which the possible outcomes are $0, 1$. For the state $|\alpha\rangle = \sum_{i=1}^{2^n} c_i|i\rangle$, the probability for outcome 0 is $p_0 = \sum_{i,i|_q=0} |c_i|^2$ and the state of the system will *collapse* to $|\beta\rangle = \frac{1}{p_0}\sum_{i,i|_q=0} c_i|i\rangle$, (the same for 1). A unitary operation U on k qubits can be applied on n qubits, $n \geq k$, by taking the **extension** $\tilde{U}$ of U, i.e. the tensor product of U with an identity matrix on the other qubits. All definitions can be generalized to circuits which operate on $p-$state quantum particles, or *qupits*. (simply replace 2 by p in the definitions above).

2.1.1 Mixed states

A system which is not ideally isolated from it's environment is described by a *mixed state*. There are two equivalent descriptions of mixed states: mixtures and density matrices. We use density matrices in this paper. A system in the **mixture** $\{\alpha\} = \{p_k, |\alpha_k\rangle\}$ is with probability p_k in the pure state $|\alpha_k\rangle$. The rules of development in time and measurements for mixtures are obtained by applying **classical** probability to the rules for pure states. A pure state $|\alpha\rangle = \sum_i c_i|i\rangle$ is associated the density matrix $\rho_{|\alpha\rangle} = |\alpha\rangle\langle\alpha|$ i.e. $\rho_{|\alpha\rangle}(i,j) = c_i c_j^*$. A mixture $\{\alpha\} = \{p_l, |\alpha_l\rangle\}$, is associated the density matrix : $\rho_{\{\alpha\}} = \sum_l p_l \rho_{|\alpha_l\rangle}$. A density matrix is thus a $2^n \times 2^n$, hermitian positive semi definite complex matrix with $tr(\rho) = 1$. The operations on a density matrix are defined such that the correspondence to mixtures is preserved. If a unitary matrix U transforms the mixture $\{\alpha\} = \{p_l, |\alpha_l\rangle\}$ to $\{\beta\} = \{p_l, U|\alpha_l\rangle\}$, then $\rho_{\{\beta\}} = \sum_l p_l U|\alpha_l\rangle\langle\alpha_l|U^\dagger = U\rho_{\{\alpha\}}U^\dagger$. A basic measurement of the j'th qubit in ρ gives the outcome 0 with the probability which is the sum of the diagonal terms of ρ, which relate to the basic states i with $i_j = 0$: $Pr(0) = \sum_{i=1}^{2^n} \rho_{i,i}\delta(i_j = 0)$. conditioned that the outcome is the eigenvalue 0, the resulting density matrix is $O_0 \circ (\rho)$, which is the minor of ρ that includes only rows and columns which relate to basic states i with $i_j = 0$. (This minor should of course be normalized to have trace one). Without conditioning on the outcome the resulting density matrix will be $O \circ (\rho) = Pr(0)O_0 \circ (\rho) + Pr(1)O_1 \circ (\rho)$. which differs from ρ, only in that the entries in ρ which connected between 0 and 1 on the same qubit, or coordinate, are put to zero. Given a density matrix ρ of n qubits, the reduced density matrix of a subsystem,A, of, say, m qubits is defined as an average over the states of the other qubits: $\rho|_A(i,j) = \sum_{k=1}^{2^{n-m}} \rho(ik, jk)$.

2.2 Quantum circuits with mixed states

A quantum unitary gate, g, of order k is a complex unitary matrix of size $2^k \times 2^k$. A density matrix ρ will transform by the gate to $g \circ \rho = \tilde{U}\rho\tilde{U}^\dagger$, where $\tilde{U}$ is the extension of U. *A Quantum circuit* is a directed acyclic graph with n inputs and n outputs. Each node v in the graph is labeled by a quantum gate g_v. The in-degree and out-degree of v are equal to the order of g_v. Some of the outputs are labeled "result" to indicate that these are the qubits that will give the output of the circuit. The wires in the circuit correspond to qubits. An initial density matrix ρ transforms by a circuit Q to a final density matrix $Q \circ \rho = g_t \circ ... \circ g_2 \circ g_1 \circ \rho$, where the gates $g_t ... g_1$ are applied in a topological order. For an input string i, the initial density matrix is $\rho_{|i\rangle}$. The output of the circuit is the outcome of applying basic measurements of the result qubits, on the final density matrix $Q \circ \rho_{|i\rangle}$. Since the outcomes of measurements are random, the function that the circuit computes is a *probabilistic function*, i.e. for input i it outputs strings according to a distribution which depends on i.

2.3 Noisy Quantum Circuits

As any physical system, a quantum system is subjected to noise. The process of noise is dynamic and depends on time. We therefore divide the quantum circuit to time steps. We permit that qubits are input and output at different times, and we say a qubit is *alive* from t_1 to t_2 if it is input to the circuit at t_1 and output at t_2. The *space-time* of the noisy quantum circuit is a two dimensional array, consisting of all the pairs (q, t), of a qubit q and time t, where the qubit q is alive at time t. $V(M)$, the volume of the circuit M, is the number of points in it's space-time. In our model of noisy quantum circuits, between every two time steps, each qubit and each gate are damaged with independent probability η. The damage operates as follows: A unitary operation operates on the qubit, (or on the qubits that are output from the gate in the case of a gate damage) *and* on a state of the environment (The environment can be represented by m qubits in some state). This operation results in a density matrix of the $n+m$ qubits. We reduce this density matrix to the n qubits of the circuit to get the new density matrix after the damage. The density matrix of the circuit develops by applying alternately computation steps and noise steps. Each "run" of the computation is subjected to a specific *fault path*, which indicates where and when fault occured. Each run ends up with some output. The function computed by the noisy quantum circuit is naturally the average over the outputs, on the probabilistic process of noise.

3 Computing on encoded states

In the following section we define quantum codes and quantum computation codes. Then it is explained how to improve the reliability of the computation using quantum computation codes.

3.1 Quantum block codes

A quantum linear block linear code is a function ϕ from the Hilbert space of a qubit to a Hilbert space of m qubits: $\phi : \mathcal{C}^2 \longmapsto \mathcal{C}^{2^m}$. m is called the size of the block in the code. Such a code induces a linear function from $\mathcal{C}^{2^n}$ to $\mathcal{C}^{2^{mn}}$ in the following way: a pure state in $\mathcal{C}^{2^n}$, $|\alpha\rangle = \sum_i c_i|i\rangle$ will transform to $|\beta\rangle = \sum_i c_i\phi|i_1\rangle\phi|i_2\rangle...\phi|i_n\rangle$. A pure state in the image of ϕ is called a word in the code. The above definition can be extended to density matrices: A mixed state of n qubits will be encoded by the corresponding probability over the encoding of the pure states. A mixture of words in the code is also said to be in the code.

The error in the encoded state is sparse if not too many qubits in each block are damaged. Using this notion of sparse sets, we can define a metric on block density matrices. They will be close if the difference between them is confined to a sparse set of qubits:

Definition 1 *Let B be the qubits in n blocks of m qubits. A set $A \subseteq B$ of qubits is said to be $k-$sparse if in each block there are not more than k qubits in A. The deviation between ρ_1 and ρ_2 of the qubits B is the minimal k such that $\exists A$ which is a k-sparse set of qubits, and $\rho_1|_{B-A} = \rho_2|_{B-A}$.*

A quantum code is said to correct d errors if there is some correction procedure, such that when which operating it on all the blocks in any density matrix that deviates by d from a code word w, we get the correct word w.

3.2 Quantum computation codes

A computation code is a quantum code which provides a way to perform gates on the encoded states fault tolerantly. The procedure Pg that simulates a gate g with respect to a quantum code is a sequence of gates which transforms the encoded state to the encoded output of the gate: $Pg(\phi|i> \otimes|0>) = \phi(g \circ |i>) \otimes |a>$, where we have used extra ancilla qubits. These qubits are not counted as the inputs or outputs of the procedure. A quantum procedure is said to have *spread l* if no qubit or gate effects more than l outputs of the procedure. We will need procedures with small spread for fault tolerant computation. Since we want to convert any arbitrary circuit to a more reliable one, we need the set of gates that have l-spread procedures to be universal.

Definition 2 *A quantum block code C is said to be a quantum computation code with spread l if there exists a universal set of quantum gates G such that (1) for any gate $g \in G$ there exists a procedure P_g with respect to C, with spread l, and (2) There exist correction, input and output procedures with spread l.*

3.3 Improving reliability by block simulations

To simulate some circuit by a quantum computation code, we first convert it to a circuit which uses only gates from the universal set of the code. Then we simulate this new circuit as was explained: We now convert each qubit to a block of qubits, each time step to a working period, and each gate to the corresponding procedure, and besides that we add in each working period a correction procedure on each block. Apart from all that, we also add an input procedure before the first working period of each block and an output procedure after the last working period of each result block.

The space-time of the simulating circuit M_1 can be divided to rectangles, where each rectangle will correspond to one procedure, in the following way: First, divide the time to alternating stages: computation stages, in which one time step of M_0 is simulated, i.e. one gate procedures is applied (in parallel), and a correction stage, in which one error correction procedures is applied (in parallel). Each stage is a *strip* in the space time. Each strip can be divided to rectangles by dividing the qubits to sets: A correction strip will be divided such that in each rectangle a correction of one block is computed. In a computation strip, we divide the strip to rectangles by dividing the qubits to sets, where each set of qubits participates in exactly one procedure. Each rectangle thus corresponds to one procedure.

We show that if a fault path in M_1 is such that no more than a few faults occured in each rectangle, then indeed the computation succeeds. 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.

Definition 3 *A fault path of M_1 that block simulates M_0 is said to be a "$k-$sparse fault path" if no more than k faults occured in each rectangle.*

Lemma 1 *Let C be a quantum computation code that corrects d errors, with spread l. Let M_1 be a block simulating circuit. Consider a computation of M_1 subjected to a $k-$sparse fault path with $d \geq 2kl$. At the end of each working period the density matrix is $d-$deviated from the correct one.*

Proof: We will prove by induction on t a stronger assertion, that at the end of the t'th working period the density matrix is $d/2-$deviated from the correct one. For $t = 0$ the deviation is zero. Suppose that the density matrix at the end of the t'th working period is $d/2$-deviated from the correct matrix. After the computation stage, not more than kl qubits are effected in each block, so the density matrix is $kl + d/2$-deviated. Since $kl + d/2 \leq d$, the correction procedure indeed corrects the error, only that during the corrections new errors occur. Again, the number of effected qubits is not more than kl in each block, and all the other qubits transform as they should, so they are corrected. Hence after the correction procedure the matrix is kl-deviated. Since $kl \leq d/2$ this proves the induction step.▮

Lemma 2 *If the final density matrix of $M1$ is $d-$deviated from the correct final matrix of $M1$, then when measuring the result qubits of $M1$, a majority of them gives the correct answer of $M0$.*

Proof: Let $\rho0$ be the correct final density matrix of $M0$, describing the mixed state $\{\alpha\} = \{p_k, |\alpha_k>\}$, where $|\alpha_k>= \sum_i c_i^k|i>$. Due to the output procedures, the correct final density matrix, ρ_1, of $M1$, the mixed state $\{\beta\} = \{p_k, |\beta_k>\}$ where $|\beta_k>$ is generated from $|\alpha_k>$ by duplicating each qubit m times: $|\beta_k>= \sum_i c_i^k|i_1^m i_2^m ... i_n^m>$. Let σ_1 be a density matrix which is $d-$deviated from $\rho1$, where $2d < m$. The probability to get an n-string i when measuring $\rho0$, equals the probability to get an n-string i when measuring $\rho1$, and taking the majority. We claim that the distribution D on $n-strings$ generated by measuring all qubits in $\rho1$ and taking the majority is the same as the distribution D', generated when measuring $\sigma1$ and taking the majority. Take A to be the maximal set of qubits such that $\sigma1|_A = \rho1|_A$. Hence, when measuring the qubits in A, in $\sigma1$, one gets the same answer on all the qubits in one block, as in $\rho1$. Since $2d < m$, these qubits determine the majority vote. ▮

The above two lemmas together show that if the faults are sparse, a majority of the result qubits will give at the end the correct answer. We can now compute the effective noise rate of M_1. The probability of M_0 to be correct is $(1-\eta)^{V(M_0)}$. We define the effective noise rate of M_1 to be 1 minus the $V(M_0)$'th root of the probability of M_1 to be correct.

Theorem 1 *Let M_1 simulate M_0 by the computation code C, which corrects d errors, have spread l, with all rectangles smaller than a. The effective noise rate of M_1 is $\leq 2\begin{pmatrix} a \\ d/2l+1 \end{pmatrix}\eta^{\frac{d}{2l}+1}$.*

Proof: If the fault path in M_1 is $d/2l$ sparse, a majority of the result qubits will give the correct answer, by lemmas 1. and 2. The probability for a rectangle to have more than $d/2l$ faults is smaller than the number of possibilities to choose $d/2l+1$ points in the rectangle, times the probability for these points to have a fault. The number of the rectangles in M_1 is less than $2V(M_0)$. Computing the probability that no rectangle had more than $d/2l$ faults, and taking the $V(M_0)$'th root and subtracting from 1 gives the desired result. ▮

The effective noise is smaller than η if the parameters of the code are chosen correctly, and in this way one can improve the reliability of the computation. However, in the above scheme, it seems difficult to find a code which will give an improvement from a constant η to polynomially small effective noise rate. To achieve such an improvement, we use concatenated simulations.

4 Concatenated simulations

In this section, we define proper quantum code, and concatenated simulations by such codes. We prove that the reliability of the computation can be improved to a constant using $log(log(n))$ levels of simulations, when the noise is smaller than some constant imposed by the parameters of the code.

4.1 Improving reliability to a constant

We would now like to apply recursively simulations, using the simulation from the last section for several times. This scheme will work if certain restrictions are imposed on the quantum computation code:

Definition 4 *A quantum computation code which is associated with a set of gates G is said to be* **proper** *if (1) all procedures use only gates from G, and (2) The correction procedure takes any density matrix to some word in the code.*

Let M_0 be a quantum circuit. We define recursively M_r, an r-simulating circuit of a circuit M_0 by the proper quantum computation code C, as the simulation by C of M_{r-1}. 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. This hierarchy of blocks requires a definition of a metric which is recursive. A density matrix of M_r is recoverable, i.e close to the correct state, if it deviates on a "sparse" set of qubits:

Definition 5 *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, are said to be (r,k)-deviated if k is the minimum integer such that there exist an (r,k)-sparse set of qubits $A\subseteq B$, with $\rho_1|_{B-A}=\rho_2|_{B-A}$.*

The deviation is a metric, since the union of two sets which are $(r,l_1),(r,l_2)$-sparse respectively is (r,l_1+l_2) sparse. This is easily shown by induction on r.

The recursive simulations also induce a definition of s-rectangles: Each space time point in M_{r-s} transforms to a set of space time points in the following simulation $M_{(r-s+1)}$, which in their turn transform to more points in the following levels of the simulation. The set of all these points in M_r that originated from one space time point in $M_{(r-s)}$ are called an s-rectangle. The definition of s-rectangles defines a division of the space time of M_r, and this division is a refinement of the division to $(s+1)$-rectangles. An 0-rectangle is just a space time point in M_r. Using this hierarchy of rectangles, we define a notion of "sparse" fault paths. We will show in lemma 3 that given that the fault path is sparse, the deviation of the state from the correct state is kept bounded throughout the computation.

Definition 6 *A set of space time points 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 (which is one space time point) 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.*

We claim that if the fault path is sparse enough, then the error corrections keep the deviation small.

Lemma 3 *Let C be a proper 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 $kl(l+1)\le d$. At the end of each $r-$working period the density matrix is $(r,d/2)$-deviated from the correct one.*

Proof: We first prove by induction on the number of levels r three assertions, together. The first two assertions, when applied alternately for the $r-$computation and $r-$correction stages in M_r, will give the desired result.

1. Consider n $r-$blocks, in a density matrix which is $(r,kl)-$deviated from $\phi^r(\rho_0)$, where ρ_0 is a density matrix of n qubits. After applying one stage

of r – *computations* on these blocks, simulating the operation g on ρ_0, with an (r, k) sparse set of faults, the density matrix is (r, d) deviated from $\phi^r(g \circ \rho_0)$.

2. Consider n $r-$blocks, in a density matrix ρ_r, which is (r, d) deviated from a word $\phi^r(\rho_0)$, where ρ_0 is a density matrix of n qubits. After applying $r-$corrections, on all of the $r-$blocks, with an (r, k) sparse set of faults, the density matrix is (r, kl) deviated from $\phi^r(\rho_0)$.

3. Consider n $r-$blocks, in some density matrix, ρ_r. After applying $r-$corrections, on all of the $r-$blocks, with an (r, k) sparse set of faults, the density matrix is (r, kl) deviated from a word $\phi^r(\rho_0)$, where ρ_0 is a density matrix of n qubits.

For $r = 0$ the proof is trivial. The computations are just faultless, and 0$-$corrections are the identity. For instructiveness, let us consider also the case of $r = 1$. Claims 1,2 are satisfied merely because we use a computation code. Claim 3 is satisfied by the extra restriction on the error correction which a proper code satisfies. Let us now assume all the claims for r, and prove each of the claims for $r + 1$. In our proof, we refer to r-correction stages and r-computation stages. An r-correction stage will start and end with an $(r-1)$-correction stage, and an r-computation stage will start and end with an $(r - 1)$-computation stage.

1. We consider an $(r + 1)-$computation stage, $(r + 1)-$simulating the operation g on ρ_0. It can be viewed as a sequence of alternating $r-$computation stages and and r-correction stages. (The number of these r-stages is w.) Let us consider the density matrices in the trajectory of $\phi^{r+1}(\rho_0)$, at the end of each of these $r-$stages. These matrices can be written as $\phi^r(\rho_1^t)$. Where $\rho_1^0 = \phi(\rho_0)$ and $\rho_1^w = \phi(g \circ \rho_0)$. Let us assume for a second that all the $r-$rectangles in the $(r + 1)-$stage have (r, k) sparse set of faults. (This assumption is wrong- in each $(r + 1)-$rectangle we might have k $r-$rectangles in which the faults are not (r, k) sparse, but we will deal with this in a second.) Let us also assume that the density matrix we start with is (r, kl)-deviated from $\phi^{r+1}(\rho_0)$. (Again, a wrong assumption- there might be kl $r-$blocks in each $(r + 1)-$block that are not $(r, kl)-$deviated.) With these assumptions, we now prove by induction on t that applying an $r-$computation stage followed by an $r-$correction stage, on a matrix which is (r, kl)-deviated from $\phi^r(\rho_1^t)$, gives a matrix which is (r, kl)-deviated from $\phi^r(\rho_1^{t+1})$. This is true by applying the induction assumption of claims 1, and 2. Now, taking into account our wrong assumptions, the "bad" k r-rectangles and kl r-blocks which did not obey the assumptions can effect at most $l(k+kl)$ $r-$blocks in each $(r + 1)-$block, at the end of the $(r + 1)-$stage, since the $(r+1)$-computation stage $r-$simulates a procedure with spread l. So if $l(kl + l) < d$, we have that the final density matrix at the end of the $(r + 1)-$stage is $(r + 1, d)-$deviated from the correct one.

2. We consider an $(r + 1)-$corrections stage of n $(r + 1)-$blocks. Again, we view this stage as alternating $r-$correction stages and r-computation stages, where the number of r-stages is w. Let us consider the trajectory (i.e. no errors) starting with $\phi^r(\rho_1)$, where ρ_1 is a density matrix which is $(1, d)-$deviated from some word $\phi(\rho_0)$, and denote the density matrices in this trajectory at the end of each $r-$stages by $\phi^r(\rho_1^t)$. Then since the $(r + 1)-$rectangle simulates error correction, and the simulated matrix is not too deviated, the trajectory which starts with $\phi^r(\rho_1^0) = \phi^r(\rho_1)$ will end with $\phi^r(\rho_1^w) = \phi^r(\phi(\rho_0)) = \phi^{r+1}(\rho_0)$. Now, let us assume, again, that all the $r-$rectangles in the $(r + 1)-$stage have (r, k)-sparse set of faults, and that the density matrix we start with is (r, d)-deviated from $\phi^r(\rho_1)$. Under these assumptions, induction on t shows that applying an $r-$correction stage followed by an $r-$computation stage, on a matrix which is (r, d)-deviated from $\phi^r(\rho_1^t)$, gives a matrix which is (r, d)-deviated from $\phi^r(\rho_1^{t+1})$. This is true by applying the induction assumption of claims 1, and 2. So under the above assumptions, we end up with a matrix which is (r, kl)-deviated from $\phi^{r+1}(\rho_0)$, using the fact that the last $r-$stage is a correction stage. Now, 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 not. By the induction stage on claims 2 and 3, after the first stage of $r-$corrections, most of the $r-$blocks are $(r, kl)-$deviated from $\phi^{r+1}(\rho_0)$, except maybe d $r-$blocks in each $(r + 1)-$block which are $(r, kl)-$deviated from $\phi^r(\rho_1')$. So after the first $r-$correction the density matrix is $(r, kl)-$deviated from $\phi^r(\rho_1)$, where ρ_1 is $(1, d)-$deviated from $\phi(\rho_0)$. We can now use the induction from before and say that the final density matrix is (r, kl)-deviated from $\phi^{r+1}(\rho_0)$. We now 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 $(r + 1)-$correction $r-$simulates a correction procedure with spread l, these can effect only kl $r-$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 $(r + 1, kl)-$deviated from the correct one, $\phi^{r+1}(\rho_0)$.

3. 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 3, after one stage of r-corrections, the density matrix is (r,kl)-deviated from some $\phi^r(\rho_1)$. Let us consider the trajectory of $\phi^r(\rho_1)$ in the $(r+1)$-correction rectangle. Since it is an r-simulation of a correction, it takes the density matrix to some word $\phi^{r+1}(\rho_0)$. As before, we can prove by induction on the r-stages that at the end of the $(r+1)$-rectangle we end up with a matrix which is (r,kl)-deviated from $\phi^{r+1}(\rho_0)$, and taking into account the r-rectangles with faults which are not (r,k)-sparse, we end up with a density matrix which is $(r+1,kl)$-deviated from $\phi^{r+1}(\rho_0)$.

We can now use claims 2 and 1 alternately to prove by induction on the number of r-working period that at the end of each r-working period in M_r the density matrix is (r,kl)-deviated from the correct one. ∎

We can now prove the main result of this paper:

Theorem 2 *Let C be a quantum computation code, which corrects d errors, have spread l, and size of all procedures smaller than a. Let M_0 be a quantum circuit, with size s and depth t. There exists a quantum circuit M_r of size $O(s \cdot polylog(s))$ and depth $O(t \cdot polylog(t))$, such that in the presence of noise η which satisfies $\binom{a}{d/l(l+1)+1}\eta^{d/l(l+1)} < 1$ M_r outputs the correct answer with probability $\geq 2/3$.*

Proof: Set $k = d/l(l+1)$. If the fault path is (r,k)-sparse, then the final density matrix is indeed (r,d)-deviated from the correct one, by lemma 3. Measuring all result blocks in a density matrix (r,d)-deviated from the correct final density matrix, and taking majority in each r-block, gives the correct answer by an argument similar to that of the proof of lemma 2. Hence the probability for M_r to succeed is larger than the probability for a fault path to be (r,k)-sparse. Let us show by induction on r that the probability, $P(r)$, of the faults in an r-rectangle to be (r,k) sparse is higher than $1-\eta^{(1+\epsilon)^r}$, where we set $\binom{a}{k+1}\eta^{k+1} = \eta^{1+\epsilon}$. We can do that because of the assumption on the parameters, and ϵ is a positive constant. The probability for an 0-rectangle, i.e. one space time point, to have faults which are $(0,k)$ sparse, i.e. that in this point a fault did not occur, is $1-\eta$. For the step of the induction, assume for r, and let us prove for $r+1$. For the faults in an $(r+1)$-rectangle to be $(r+1,k)$ sparse, there must be at most k r-rectangles in which the fault is not (r,k) sparse. So $P(r+1) \geq 1-\binom{a}{k+1}(1-Pr)^{k+1} \geq 1-\eta^{(1+\epsilon)^{r+1}}$, using the induction assumption. This proves that the probability of success of Mr is $\geq (1-\eta^{(1+\epsilon)^r})^{2V(M_0)}$ since the number of r-rectangles is less than $2V(M0)$. Taking $r = O(log(log(V(M_0))))$ gives a constant probability of success. Since the growth in time and in space is exponential in r, the cost is polylogarithmic (We use codes of constant size). ∎

Remark: Theorem 2 requires that the code can correct $d > 1$ errors. A similar result holds for $d = 1$, with the threshold $\binom{b}{2}\eta < 1$ where b is the maximal size of slightly different rectangles, defined to contain a computation and a correction procedure together. The proof is almost the same. In some cases this threshold is better. ∎

5 Explicit proper codes

Linear quantum codes[4] are represented, using classical codes over F_p, and shown to be proper for $p = 2$. A subclass of linear codes, polynomial quantum codes, is defined and shown to be proper for $p > 2$.

5.1 Linear quantum codes over F_p.

A linear code of length m and dimension k over the field F_p is a subspace of dimension k in F_p^m, where F_p^m is the m dimensional vector space over the field of p elements. Given two linear codes C_1 and C_2 such that $\{0\} \subset C_2 \subset C_1 \subset F_p^m$ consider the following set of quantum states in the Hilbert space $\mathcal{C}^{p^m}$:

$$\forall a \in C1 : |S_a> = p^{-(m-k)/2}\sum_{v \in C_2}|a+v> .$$

If $(a1-a2) \in C_2$ then $|S_{a1}> = |S_{a2}>$, otherwise $<S_{a1}|S_{a2}> = 0$. Hence these states construct a basis for a linear subspace of the Hilbert space $\mathcal{C}^{p^m}$, with dimension $z = p^{dim(C1)-dim(C2)}$. This subspace is our quantum code. Define a second basis of this subspace to be:

$$\forall a \in C_2^{\perp} : |C_a> = \frac{1}{\sqrt{z}}\sum_{b \in C_1/C_2} w^{a \cdot b}|S_b> \ , \ w = e^{\frac{2\pi i}{p}} .$$

If $C1$ and $C_2^{\perp}$ both have minimum weight d, then the quantum code can correct for $t = \lfloor\frac{d-1}{2}\rfloor$, by applying classical error corrections with respect to the code C_1, first in the S-basis, and then in the C-basis. The proofs in [4] transforms smoothly to this general case.

Theorem 3 *For $p = 2$, linear block codes are proper. The universal set of gates associated with the code is :*

(1) $|a, b> \longmapsto |a, a+b>$,

(2) $|a> \longmapsto \frac{1}{\sqrt{2}} \sum_b (-1)^{ab} |b>$,

(3) $|a> \longmapsto |1-a>$,

(4) $|a> \longmapsto (i)^a |a>$, *and*

(5) $|a, b, c> \longmapsto |a, b, c+ab>$,

where all the addition and multiplication are in F_2(i.e. mod 2). These exist gate procedures, correction, input and output procedures, with respect to the code which have spread $l = 1$.

Proof: This set of gates is universal by [23]. We describe input, correction, output and gate procedures, all with spread 1.

Input procedure: It is enough to show how to generate a state $|S_0>$ in an ancilla block, since we can apply afterwards a controlled not (gate 1) from the i'th input bit to the i'th bit in the ancilla, and we get $|S_0>$ or $|S_1>$ in correspondence with the input. To generate the state $|S_0>$, we actually apply a correction procedure with respect to a code which consists of one word: $|S_0>$ alone. We start with a state $|0^m>$, and we want to correct it with respect to the code of one word, $|S_0>$. First, rotate each bit around the x axis, using gate 2. We now want to compute the syndrome of the error. We will compute the syndrome independently m times, one for each qubit. To compute the $j'th$ bit of the syndrome, we do the following: After [23], we will use a *cat state*, generated in the following way: Start with $|0^m>$, apply a rotation around the x axis (gate 2) on the first qubit and then a controlled not (gate 1) from this qubit to all the other $m-1$ qubits, to get $\frac{1}{\sqrt{2}}(|0^m> + |1^m>)$. Now apply a controlled not bitwise from the block we are initializing to the cat state only on the coordinates which in the $j'th$ raw of H, the parity check matrix of $C^{\perp}$, are 1. From this cat state apply a controlled not bitwise to m qubits in $|0>$, to imitate a measurement of the cat state. Now compute from the measured cat state the j'th syndrome bit, using only gates from the universal set. Compute in this way all the bits of the syndrome, and now from these bits we can compute whether the i'th bit has an error, as follows: the vector space can be divided to non-intersecting cosets of the subspace $C^{\perp}$. Each coset can be written as $C^{\perp} + e$ where e is a vector. Each such e gives one possible syndrome.($He = s$). Given the syndrome, we compute the **table** $s -> e$, and decide whether a qubit is wrong by asking whether it is in the support of e, meaning that the corresponding coordinate in e is 1. Finally apply a controlled not from the result to the $i'th$ qubit. This can be done independently for all the qubits. We have used $2m^3$ ancilla qubits. To see that we indeed get the state $|S_0>$, note that after the first rotations we have $\sum_{i=0}^{2^m-1} |i>$. 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. The spread of this procedure is 1: a fault in the cat states, in the first rotations and in the first controlled not gates can only effect one bit at the end.

Correction procedure: The correction is similar to what is done in the input procedure (Computing the syndrome with respect to $C1$, independently for each qubit, and correcting the qubits.) There is one difference: Since the code must be proper, any density matrix must be corrected to some word in the code. We guarantee this in the following way: Before starting the correction procedure, Generate another state $|S_0>$ on ancilla qubits, as in the input 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 larger than d, and write the answer on another qubit. The controlled not from the result to the $i'th$ qubit is replaced by a Toffoli gate (gate 5) which also checks if the number of faults is smaller than d. We also add a gate which swaps the qubit with a qubit from the state $|S_0>$ if the number of faults is indeed larger that d, using a Toffoli gate again. After applying classical error corrections, with respect to the code C_1, we transform to the $C-$basis by applying bitwise gate 2, correct again and rotate back to the $S-$basis.

Output procedure: The output procedure is applied by computing independently m times the a from the state $|S_a>$. This procedure requires $m(m+1)$ ancilla qubits. First we apply controlled not from each of the qubits to one of the last m ancilla qubits to simulate m basic measurements of these qubits[3]. We repeat this operation again to each one of the m blocks of m ancilla qubits, so we have m copies of the measured state. On each of these copy we apply an operation that computes the bit that is represented by this state, using only gates from the universal set. (This might require more ancilla qubits.) The resulting qubits are the m qubits which carry the results from these m computations. The spread of this procedure is 1. 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 faults is smaller than the critical number, the fault has no effect on the resulting qubits. A fault during the computation of one bit can only effect this bit. So the spread is 1, as long as the sum of damage in the block and number of faults in the first stage is smaller than half the minimal distance of the code.

Gate procedures: The procedures of gates $1-4$

are performed by applying bitwise the gates on each of the qubits in the block.

Toffoli gate: we define a Toffoli procedure as is described in [23], where all the measurements are replaced by controlled not gates. The only piece of this procedure which is not straight forward is creating the ancilla state $|A>$, without involving classical operations. To do that, generate m cat states, $|S_0>^m$, rotate the first block by applying gate 2 bitwise, and then copy this block bitwise on all the other blocks, giving the encoded cat state: $\frac{1}{\sqrt{2}}(|S_0>|S_0>...|S_0> +|S_1>|S_1>...|S_1>)$. Generate three such ancilla states. Also generate $|A>+|B>$ as in [23]. Now apply the transformation: $|S_a>|b>|c>|d> ---> (-1)^{a(bc+d)}|S_a>|b>|c>|d>$ on a block in the first encoded cat state and three bits in the three blocks of $|A>+|B>$, and do that for all blocks in the first encoded cat state, i.e. m times. Repeat with the second cat state and then the third. To measure the cat state, rotate all qubits in the encoded cat states in the x direction. Now compute from each block the bit it represents, and then independently m times the parity of these bits in each encoded cat state. For each qubit in the third block of $|A>$, compare three parity bits from the three cat states by a majority vote and apply a controlled not from the result to the qubit in $|A>$. If only one fault occurred in this procedure, then in each block of $|A>$ there is at most one effected qubit.∎

5.2 Polynomial quantum codes

Here we define the quantum analog of random polynomial codes[5]. To correct d errors, set $m = 4d+1$ and set $p > m+1$. Let $\alpha_1, \alpha_2, ..., \alpha_m$ be m distinct non zero elements of F_p such that the polynomial $G(x) = \Pi_{i=1}^{m}(x-\alpha_i)$ has a non-zero coefficient of x^{2d}. (Such α_i exist because $|F_p| > m+1$). Denote by

$$\begin{aligned} V_1 &= \{f(x) \in F(x) \mid deg f(x) \leq d\}, \\ V_2 &= \{f \in V_1 \mid f(0) = 0\}, \\ C_1 &= \{(f(\alpha_1), ..., f(\alpha_m)) \mid f \in V_1\} \subset F_p^m, \\ C_2 &= \{(f(\alpha_1), ..., f(\alpha_m)) \mid f \in V_2\} \subset C_1. \end{aligned}$$

As before, we use the codes C_1 and C_2 to define the quantum code:

$$\forall a \in F, \quad |S_a> = \frac{1}{\sqrt{|V2|}} \sum_{f \in V_1, f(0)=a} |f(\alpha_1), ..., f(\alpha_m)> .$$

Theorem 4 *Polynomial codes are proper quantum computation codes with spread $l = 1$. The universal set of gates is:*

(1) $|a>|b> \longmapsto |a>|a+b>$,

(2) $\forall\ c \in F$, $|a> \longmapsto |a+c>$,

(3) $0 \neq c \in F$: $|a> \longmapsto |ac>$,

(4) $|a>|b>|c> \longmapsto |a>|b>|c+ab>$,

(5) $\forall c \in F$ $|a> \longmapsto w^{ca}|a>$, *and the Fourier transform*

(6) $|a> \longmapsto \frac{1}{\sqrt{p}} \sum_{b \in F} w^{rab}|b>, \forall 0 < r < p$.

Proof:

Universality: Clearly, all classical reversible functions can be spanned by this set. We find an explicit unitary matrix in the group generated by this set, which has infinite order. We then use group representation theory to show that this group is dense in $SU(n)$. By [27], a general product of j such matrices, is exponentially(in j) close to any finite matrix, so the rate of approximation is exponential.

Input, output and correction procedure: These are exactly as in the general linear code, where transforming between the $S-$basis and the $C-$basis is done by the Fourier transform.

Gate procedures: The procedures of gates 1, 2 and 3 are performed by applying pitwise the corresponding gates.

Procedure of general Toffoli gate (4): This procedure we use interpolation techniques[5]. First we apply pitwise the general Toffoli gate on the m coordinates. On the third block we obtain the sum:

$$\sum |A(\alpha_1)B(\alpha_1)+C(\alpha_1), ..., A(\alpha_m)B(\alpha_m)+C(\alpha_m)>,$$

where the sum is over $A(x), B(x), C(x) \in V1, A(0) = a, B(0) = b, C(0) = c$. The polynomial $D(x) = A(x)B(x)+C(x)$ satisfies $D(0) = ab+c$, and it's degree $deg(D) \leq 2d$. To reduce the degree we use a quantum analog of the techniques in [5], where we keep the procedure fault tolerant. we can still correct d errors in D since $m = 4d+1$. We proceed as follows: we apply the input procedure to each coordinate in the third block, which will give the state

$$\sum_{D(x), D(x)=A(x)B(x)+C(x)} |S(d_1), ...S(d_m)> \ , \ d_j = D(\alpha_j).$$

We then first run the error correcting procedure of the code with degree $2d$, and after this compute the linear combination $S(\sum_l c_l d_l) = Sd$, where the c_l are the interpolation coefficients such that $\forall\ f \in F[x]$, $deg(f) \leq m-1, f(0) = \sum_{i=1}^{m} c_i f(\alpha_i)$.

Procedure of general rotation around the z axis by the angle π (5): This is done by applying on the $l'th$ qupit the gate $|a> \longmapsto w^{c_l a}|a>$. The proof that this achieves the desired operation is :

$$|S_a> = \frac{1}{\sqrt{|V2|}} \sum_{f \in V, f(0)=a} |f(\alpha_1), ..., f(\alpha_m)> \longmapsto$$

$$\sum_{f\in V1, f(0)=a} \Pi_{i=1}^{m} w^{c_i f(\alpha_i)} |f(\alpha_1),...,f(\alpha_m)>=$$

$$= \sum_{f\in V, f(0)=a} w^a |f(\alpha_1),...,f(\alpha_m)> .$$

Procedure of the Fourier transform:(6) $|Sa>\longmapsto \frac{1}{\sqrt{p}}\sum_{b\in F} w^{ab}|Sb>$. (This can be done with any w^r by replacing in the following w by w^r. This operation generalizes the rotation around the x axis by the angle π.)

To perform this procedure we first note that there are fixed non zero $e_1,...,e_m$ such that for any polynomial $f(x)$ over F_p with $deg(f) \leq m-1$, $f_{2d} = \sum_{e_i} e_i f(\alpha_i)$. This is true since interpolation via $\alpha_1,...\alpha_m$ is a linear functional. Denote $w_l = w^{e_l \alpha_l^{2d}}$, $l = 1,...,m$. Let us operate on each coordinate, that is qupit, by the Fourier transform

$$|a>\longmapsto \sum_{b\in F_p} w_l^{ab}|a> .$$

We claim that this indeed gives the desired operation. This is true since

$$|Sa>= \frac{1}{\sqrt{|V2|}} \sum_{f\in V, f(0)=a} |f(\alpha_1),...,f(\alpha_m)>\longmapsto$$

$$\frac{1}{\sqrt{p}} \sum_{b1,b2,..bm\in F} \sum_{f\in V, f(0)=a} w^{\sum_{l=1}^{m} e_l\alpha_l^{2d} f(\alpha_l) b_l} |b1,..bm> .$$

Let $b(x)$ be the unique interpolation polynomial $b(\alpha_l) = b_l$, with degree $deg(b) \leq m-1$, for some $b_1,...b_m \in F_p$. We distinguish two cases:

- CASE 1: $Deg(b) \leq d$. In this case the polynomial $h(x) = x^{2d}b(x)f(x)$ for $f(x) \in V1$ is of degree $\leq 4d = m-1$ and so

  $$\sum_{l=1}^{m} e_l\alpha_l^{2d} f(\alpha_l) b_l = f(0)b(0) = coeff\ of\ x^{2d}\ in\ h(x).$$

  In the above sum, we will have:

  $$\frac{1}{\sqrt{p}} \sum_{b1,b2,..bm\in F, b(x)\in V1} \sum_{f\in V, f(0)=a} w^{b(0)f(0)} x |b1,..bm>=$$

  $$= \sum_{b\in F_p} w^{ab}|Sb> .$$

- CASE 2: $Deg(b) > d$. We claim that the sum vanishes for the $b's$ in this case. Let $h(x)$ be the interpolating polynomial through the values $\alpha_l^{2d} f(\alpha_l) b_l$. Then $h(x) = x^{2d}f(x)b(x)(mod G(x))$. recall the definition of $G(x)$. The power of w in the sum is the coefficient of x^d in this polynomial. It is enough to show that this is not always the same value when summing over $f \in V1$ with $f(0) = a$, since then the sum vanishes. Let $r = deg(b) > d$. Picking $f(x) = a + cx^{2d-r+1}$, $deg(f) \leq d$, and deg $x^{2d}f(x)b(x) = 4d+1 = m$. Therefore $h(x) = x^{2d}f(x)b(x) - cB_rG(x)$ where Br is the leading coefficient in $b(x)$, i.e $b(x) = B_rx^r + ... + B_0$. $B_r \neq 0$. Looking at the coefficient of x^{2d} of $h(x)$ we have $aB_0 - cB_rg_{2d}$ where $g_{2d} \neq 0$ is the coefficient of x^{2d} in $G(x)$. This can obtain any value we want by selecting an appropriate $c \in F_p$.∎

6 Generalizations 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 noise rates, and theoreticians to develop a theory for proper quantum codes, and seek such 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.

The results of this paper hold also in the case of circuits which allow to operate only on nearest neighbors. (We thank Richard Cleve for pointing this out to us.) This is true since the procedures we use, which are of constant size, can be made, with constant cost, to operate only on nearest neighbors, by adding gates that swap between qubits. However, the bound on η in this case will be smaller.

Our scheme requires a polylogarithmic blow-up in the depth of the circuit. Reducing this to a constant, as in the classical case, remains an open problem.

This result might also have an impact on a long standing question in quantum physics, regarding the transition from quantum to classical physics[30]. In [1] it was shown that for a very high noise rate, the quantum circuit behaves in a classical way. In this paper we show that for very small noise rate, the system can still maintain it's quantum nature. If indeed the quantum nature can not be imitated by classical systems[13], or in other words if $BPP \neq BQP$, then, increasing the noise, a transition from the quantum behavior to classical behavior occurs. Does this transition happen at a critical noise-rate? Indications for a positive answer are already shown in [1]. We view this connection between quantum complexity and quantum physics as extremely interesting.

7 Acknowledgments

We wish to thank Noam Nisan, Peter Shor, Thomas Beth, David DiVincenzo and Ehud Friedgut for helpful discussions and essential remarks. We thank Richard Cleve for solving the question of nearest neighbor gates.

References

[1] D. Aharonov and M. Ben-Or. Polynomial simulations of decohered quantum computers. *37th Annual Symposium on Foundations of Computer Science*, pages 46–55, 1996.

[2] D. Aharonov, M. Ben-Or, R. Impagliazzo, and N. Nisan. Limitations of noisy reversible computation. In *preperation*, 1996.

[3] D. Aharonov and N. Nisan. Quantum circuits with mixed states. *in preparation.*

[4] A.R.Calderbank and P.W.Shor. Good quantum error correcting codes exist. In *phys.Rev.A, to appear*, pages quant–ph/9512032, 1995.

[5] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for fault-tolerant distributed computing. In *Proc. 20th ACM Symp. on Theory of Computing*, pages 1–10, Chicago, 1988. ACM.

[6] E. Bernstein and U. Vazirani. Quantum complexity theory. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on the Theory of Computing*, pages 11–20, 1993.

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

[8] I. Cirac and P. Zoller. *Phys.Rev.Lett*, 74:4091, 1995.

[9] B.S. Cirel'son. *Reliable storage of information in a system of unreliable components with local interactions*, volume 653 of *Lecture Notes in Mathematics*, pages 15–30. springer edition, 1978.

[10] D. Deutch. Quantum networks. In *Proc. Roy. Soc. Lond, Vol. A400*, 1989.

[11] D. Deutsch. Quantum theory, the church-turing principle and the universal quantum computer. In *Proc. Roy. Soc. Lond, Vol. A400*, pages 96–117, 1985.

[12] DiVincenzo. Quantum computation. *Science*, 270, 1995.

[13] R. Feynman. Simulating physics with computers. In *International Journal of Theoretical Physics, Vol. 21, No. 6/7*, pages 467–488, 1982.

[14] P. Ga'cs. Self correcting two dimentional arrays. In S. Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 240–241,246–248. 1989. series editor: F.P.Preparata.

[15] I.Chuang, W.C.D. Leung, and Y. Yamamoto. Bosonic quantum codes for amplitude damping. 1996.

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

[17] E. Knill. Non-binary unitary error bases and quantum codes. *quant-ph/9608048*, 1996.

[18] E. Knill, R. Laflamme, and W.H. Zurek. Threshold accuracy for quantum computation. *quant-ph/9610011*, 1996.

[19] R. Laflamme, C. Miquel, J.P. Paz, and W.H Zurek. Perfect quantum error correcting codes. *quant-ph/9602019*, 1996.

[20] S. Lloyd. A potentially realizable computer. In *Science, Vol 261*, pages 1569–1571, 1993.

[21] G. M. Palma, K.A. Suominen, and A. Ekert. Quantum computation and dissipation. *Proc. Roy. Soc. Lond.*, 1995.

[22] P. Shor. Scheme for reducing decoherence in quantum computer memory. *Phys.Rev.A*, 52(4), 1995.

[23] P. W. Shor. Fault-tolerant quantum computation. In *37th Annual Symposium on Foundations of Computer Science*, 1996.

[24] P.W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[25] B. Shumacher and M.A. Nielsen. Quantum data processing and error correction. pages quant–ph/9604022, 1996.

[26] D. Simon. On the power of quantum computation. In *35th Annual Symposium on Foundations of Computer Science*, pages 116–123, 1994.

[27] R. Solovay and A.C. Yao. in preperation.

[28] A.M. Steane. Multiple particle interference and quantum error. In *Proc.Roy.Soc.London.Ser, to appear. Also quant-ph/9601029.*

[29] W. G. Unruh. Maintaining coherence in quantum computers. Technical report, University of Vancouver, 1994. quant-ph/9406058.

[30] W.H.Zurek. Decoherence and the transition from quantum to classical. In *Physics today 44(10)*, pages 36–44, 1991.

[31] A. Yao. Quantum circuit complexity. In *34th Annual Symposium on Foundations of Computer Science*, pages 352–361, 1993.