Fault-Tolerant Quantum Computation

Peter W. Shor
AT&T Research
Room 2D-149
600 Mountain Ave.
Murray Hill, NJ 07974, USA
shor@research.att.com

Abstract

It has recently been realized that use of the properties of quantum mechanics might speed up certain computations dramatically. Interest in quantum computation has since been growing. One of the main difficulties in realizing quantum computation is that decoherence tends to destroy the information in a superposition of states in a quantum computer, making long computations impossible. A further difficulty is that inaccuracies in quantum state transformations throughout the computation accumulate, rendering long computations unreliable. However, these obstacles may not be as formidable as originally believed. For any quantum computation with t gates, we show how to build a polynomial size quantum circuit that tolerates $O(1/\log^c t)$ amounts of inaccuracy and decoherence per gate, for some constant c; the previous bound was $O(1/t)$. We do this by showing that operations can be performed on quantum data encoded by quantum error-correcting codes without decoding this data.

1. Introduction

It has recently been discovered that certain properties of quantum mechanics have a profound effect on abstract models of computation. More specifically, by using the superposition and the interference principles of quantum mechanics, one can devise a physics thought experiment giving a computing machine which is apparently more powerful than the standard Turing machine model of theoretical computer science. Using only polynomial resources, these quantum computers can compute certain functions which are not known to be computable on classical digital computers in less than exponential time [26, 23, 25]. The potentially most useful algorithms for quantum computers discovered so far include prime factorization and simulation of certain quantum mechanical systems.

Given these theoretical results, a natural question is whether such computers could ever be built. Ingenious designs for such computers have recently been proposed [13], and currently several experiments are underway in attempts to build small working prototypes [18]. Even if small quantum computers can successfully be built, scaling these up to computers that are large enough to yield useful computations could present formidable difficulties.

One of these difficulties is *decoherence* [16, 28, 12]. Quantum computation involves manipulating the quantum states of objects that are in coherent quantum superpositions. These superpositions, however, tend to be quite fragile and decay easily; this decay phenomenon is called decoherence. One way of thinking about decoherence is to consider the environment to be "measuring" the state of a quantum system by interacting with it [31].

A second potential obstacle to building quantum computers is inaccuracy [16, 9, 4]. Quantum computers are fundamentally analog-type devices; that is, the state of a quantum superposition depends on certain continuous parameters. For example, one of the common quantum gates used in quantum computations is a "rotation" of a quantum bit by an angle θ. When doing this transformation, there will naturally tend to be some inaccuracy in this angle θ. For the quantum computation to successfully yield the correct result, this inaccuracy must be less than the amount of inaccuracy which the computation can tolerate. All quantum gates are potentially analog in that there will be some amount of inaccuracy in any physical implementation; that is, the output quantum state will not be precisely the desired state. What must be shown is that the tolerance of the computation to these inaccuracies is large enough to permit quantum gates to be built.

These two difficulties are closely related. Decoherence can be expressed purely in terms of inaccuracies in the state of the quantum system and an auxiliary quantum system interacting with it called the "environment." Thus, decoherence-reduction methods can often be used to correct

0272-5428/96 $05.00

inaccuracy, and vice versa. It has already been shown that the use of quantum error correcting codes [24, 11, 27, 15, 7] can reduce both decoherence and inaccuracy dramatically during transmission and storage of quantum data. We build upon these techniques to show that the use of these codes can also reduce decoherence and inaccuracy while performing computations on quantum data.

Until now, the best estimate on the amount of inaccuracy required to permit t steps of quantum computation was $O(1/t)$ [9]. We show in this paper that quantum circuits can be made substantially more fault-tolerant. For any polynomial size quantum circuit, we show how to construct a fault-tolerant version of the quantum circuit which computes the same function and also has polynomial size. This circuit can tolerate $O(1/\log^c n)$ inaccuracy in the quantum gates, and decoherence averaging $O(1/\log^c n)$ per step.

More specifically, we use the quantum circuit model of computation, augmented with measurement operations during the computation. For noise-free quantum circuits, these measurement operations can always be delayed until the end of the computation [2]; thus previous definitions of quantum circuits have sometimes only permitted measurement steps at the end, as this model is easier to work with and was believed to be equivalent. It has not yet been shown that measurement operations can be delayed until the end of the computation for noisy quantum circuits (although this seems plausible since noisy quantum gates might be used to simulate measurement by noisy classical gates, which can in turn perform reliable classical computation [29, 20]). For now, it appears to be easier to provide fault-tolerance if measurement operations are allowed during the computation. There is no fundamental physical reason for requiring that measurement be delayed until the end of the computation.

The techniques used in this paper to build fault-tolerant quantum circuits rely heavily on quantum error correcting codes [24, 11, 27, 15, 7]. These codes can be used to encode k quantum bits (qubits) of data into n qubits of data so as to protect the data if errors occur in any t of these n qubits, where n, k and t are values which depend on the code used. (But note that t cannot exceed some upper bound depending on n and k, analogous to upper bounds in classical information theory [11, 7].) These codes were previously known to be potentially useful for storage and transmission of quantum data. It was not clear whether such codes could be used to prevent errors during quantum computation: more specifically, it was not known how to compute with encoded qubits without decoding them, and decoding the qubits in order to compute exposes them to potential errors. Further, decoding or correcting errors in quantum codes is in itself a quantum computation. It was not known how to correct errors using noisy quantum gates without possibly introducing worse errors.

Quantum error-correcting codes map qubits into blocks of qubits so that a small number of errors in the qubits of any block has little or no effect on the encoded qubits. We find circuits for correcting errors in the encoding qubits and for computing with the encoded qubits so that if these circuits are implemented with slightly noisy gates, only a small number of errors result in the encoding qubits in each block, and thus the encoded qubits are not disturbed.

This paper shows both how to correct errors in encoded qubits using noisy gates and also how to compute on these encoded qubits without ever decoding the qubits. We can thus alternate steps which perform computations on encoded qubits with steps that correct any errors that have occurred during the computation. If the error probability is small enough and the errors are corrected often enough, the probability that we have more errors than our quantum error correcting code can deal with remains small. This ensures that we have a high probability of completing our computation before it has been derailed by errors. Our results thus show that if quantum gates can be made only moderately reliable through hardware, substantial further improvements in reliability may be achievable through software.

At this point it may be informative to compare the classical and quantum situations with respect to computing with noisy gates. Classical storage and transmission of data through noisy media can be accomplished with relatively little overhead by using error correcting codes. However, performing classical computation with noisy gates is considerably harder. While *ad hoc* methods can be used to reduce error in classical computers at relatively low cost, general techniques for performing reliable computation with noisy gates require a logarithmic increase in the size of classical circuits [14, 21]. These techniques involve keeping several copies of every bit, and periodically reconciling them by setting them to the majority value [29, 20, 22].

Digital circuitry is reliable enough that these techniques are only cost-effective when reliability is of paramount importance [22]. The techniques given in this paper are even more costly in that they require a polylogarithmic increase in the size of quantum circuits. Quantum gates, however, are inherently less reliable than classical gates and thus in the quantum setting, the benefit of these techniques may justify their cost.

The techniques in this paper are also related to a different classical problem that occurs in cryptography. It is possible to perform computation on data that has been encoded and shared among several processors so that the data known collectively by any small subset of the processors gives no information about the unencoded data [8]. This is similar to the quantum mechanical requirement that measurement of the states of a small number of qubits in a fault-tolerant quantum circuit must give no information about the unencoded data, and the techniques used in [8] are similar to those used in this paper.

The rest of the paper is organized as follows. In Section 2, we briefly review the quantum circuit model of quantum computation that we use (which also allows additional measurement steps during the computation). In Section 3, we briefly review the quantum error correcting codes discovered independently by Calderbank and Shor and by Steane [11, 27], which are used in our construction. We also review some of the theory of classical error correcting codes. In Section 4, we show how errors in quantum information encoded in these quantum error correcting codes can be corrected using slightly noisy gates without introducing more error than is eliminated. In Sections 5 and 6, we show how to compute using encoded quantum data: Section 5 shows how to perform Boolean linear operations and certain $\pi/2$ rotations and Section 6 shows how to perform Toffoli gates. Together, these form a universal set of gates for quantum computation. In Section 7, we put all these pieces together to obtain robust quantum computation and we discuss some open problems.

Since our submission of this paper, we have learned that Zurek and Laflamme have independently investigated gates that calculate on encoded qubits [32]. Some of their ideas might be useful for simplifying our constructions.

2. Quantum circuits

The model of quantum computation we use is the quantum circuit model [30]. Our quantum computation will be done in the quantum state space of n two-state quantum systems (e.g., spin-$\frac{1}{2}$ particles). Each of these two-state quantum systems can be in a superposition of two quantum states, which we represent by $|0\rangle$ and $|1\rangle$. The quantum state space of n of these particles is a 2^n-dimensional complex space, with 2^n basis vectors $|b_1 b_2 \ldots b_n\rangle$ indexed by binary strings of length n. A (pure) quantum state is simply a unit vector in this space. Each of the n particles corresponds to one of the n bit positions in the indices of the 2^n basis vectors. That is, a quantum state is a sum

$$\sum_{b=0}^{2^n-1} \alpha_b |b\rangle \tag{2.1}$$

where the α_b are complex numbers with $\sum_b |\alpha_b|^2 = 1$. Each of the n particles is called a qubit.

A quantum gate on k qubits is a 2^k by 2^k unitary matrix which acts on the quantum state space of k qubits. To apply this transformation to the quantum state space of n qubits, we must first decide which k of the n qubits we wish to apply it to. We then apply the unitary transformation to these coordinate positions, leaving the binary values in the other coordinates untouched. For quantum computation, we take k to be some constant (such as 2 or 3). Quantum gates on k qubits then involve the interaction of only a constant number of quantum objects and thus are more likely to be physically realizable. It turns out that for constant $k \geq 2$, as long as a reasonably powerful set of quantum gates is realizable, the functions computable in quantum polynomial time does not depend on k. Such a set of quantum gates powerful enough to realize quantum computation is called a *universal* set of quantum gates; It has been shown that most sets of quantum gates are universal [3].

In a measurement operation, we measure the value (0 or 1) of one of the qubits. This will project the system into a superposition of states where this qubit has a definite value of either 0 or 1. If we measure qubit i, the qubit will be measured as 0 with probability

$$\sum_{b|b_i=0} |\alpha_b|^2 \tag{2.2}$$

and 1 with probability

$$\sum_{b|b_i=1} |\alpha_b|^2. \tag{2.3}$$

If b_i is observed to be 0, say, the relative values of the coefficients α_b are preserved on the states $|b\rangle$ with the ith bit of b being 0, but they are renormalized so the resulting state is a unit vector.

A quantum computation is a sequence of quantum gates and measurements on this 2^n-dimensional quantum state space. In order to produce a *uniform* complexity class, we need to require that this sequence can be computed by a classical computer in polynomial time. We allow the classical computer to branch depending on the measurement steps; that is, after a step which measures b_i, different quantum gates can be applied depending on whether b_i was observed to be 0 or 1.

For our fault-tolerant quantum circuits, we need results on universal sets of gates for quantum computation. One of the simplest universal sets contains the XOR (also called the controlled NOT) gate and all one-qubit gates [3]. The controlled NOT gate, which maps basis states as follows:

$$\begin{aligned} |00\rangle &\rightarrow |00\rangle \\ |01\rangle &\rightarrow |01\rangle \\ |10\rangle &\rightarrow |11\rangle \\ |11\rangle &\rightarrow |10\rangle\,, \end{aligned} \tag{2.4}$$

operates on two qubits and negates the target qubit if and only if the control qubit is 1. We say that this XORs the control qubit into the target qubit. Two canonical one-qubit gates are rotations around the x-axis by an angle θ, which take

$$\begin{aligned} |0\rangle &\rightarrow \cos(\theta/2)|0\rangle + \sin(\theta/2)|1\rangle \\ |1\rangle &\rightarrow -\sin(\theta/2)|0\rangle + \cos(\theta/2)|1\rangle\,, \end{aligned}$$

and rotations around the z-axis by ϕ, which take

$$\begin{array}{rcl} |0\rangle & \rightarrow & |0\rangle \\ |1\rangle & \rightarrow & e^{i\phi/2}|1\rangle \,. \end{array}$$

We need the result that the following set of three gates — rotations around the x-axis and the z-axis by $\pi/2$, and Toffoli gates — is a universal set of gates sufficient for quantum computation. The proof of this involves showing that these gates can be combined to produce a set of gates dense in the set of 3-qubit gates. Because of space considerations, the details are left out of this abstract.

The Toffoli gate is a three-qubit gate, as follows:

$$\begin{array}{rcl} |000\rangle & \rightarrow & |000\rangle \\ |001\rangle & \rightarrow & |001\rangle \\ |010\rangle & \rightarrow & |010\rangle \\ |011\rangle & \rightarrow & |011\rangle \\ |100\rangle & \rightarrow & |100\rangle \\ |101\rangle & \rightarrow & |101\rangle \\ |110\rangle & \rightarrow & |111\rangle \\ |111\rangle & \rightarrow & |110\rangle \,, \end{array} \qquad (2.5)$$

The Toffoli gate is a reversible classical gate which is universal for classical computation. The XOR is also a classical gate, but the only classical functions that can be constructed with it are linear Boolean functions; it takes three bits to provide a reversible classical gate which is universal for classical computation. (Recall that all quantum gates must be reversible). We will show that fault-tolerant quantum computation is possible by showing how to do both $\pi/2$ rotations and Toffoli gates fault-tolerantly.

This paper is too short to fully discuss error models in quantum circuits. We work with a simplified error model which is easy to analyze. We assume that no errors occur in quantum "wires" in our circuits, but only in the quantum gates. In practice, unless very stable quantum states are used to store data, quantum bits will degrade somewhat between their output by one quantum gate and their input into another. Practical large-scale quantum computation thus might require storage of quantum data using error correcting codes and periodic error correction of the memory in order to avoid excess accumulation of errors. For large amounts of memory, this may necessitate parallel processing to keep the memory from decaying faster than it can be accessed.

For the error model in our quantum gates, we assume for each gate that with some probability p, the gate produces unreliable output, and with probability $1 - p$, the gate works perfectly. This model thus assumes "fast" errors, which cannot be prevented by the quantum watchdog (quantum Zeno) effect [19]. This type of error encompasses a standard model for decoherence, where, with some small probability, the state of a gate is "measured" during its operation. Fault-tolerant circuits which can correct "fast" errors are also able to correct "slow" errors. These include the standard model of inaccurate gates where the unitary matrices the quantum gates implement are not precisely those specified. One way to analyze these error models is to use density matrices [19, 2].

3. Error-correcting codes

The construction of quantum error correcting codes relies heavily on the properties of classical error correcting codes. We thus first briefly review certain definitions and properties related to binary linear error correcting codes. We only consider vectors and codes over F_2, the field of two elements, so we have $1 + 1 = 0$. A binary vector $v \in \mathsf{F}_2$ with d 1's is said to have *Hamming weight d*. The *Hamming distance* $d_H(v, w)$ between two binary vectors v and w is the Hamming weight of $(v + w)$.

A *code* C of length n is a set of binary vectors of length n, called *codewords*. In a *linear code* the codewords are those vectors in a subspace of F_2^n (the n-dimensional vector space over the field F_2 on two elements). The *minimum distance* $d = d(C)$ of a binary code C is the minimum distance between two distinct codewords. If C is linear then this minimum distance is just the minimum Hamming weight of a nonzero codeword.

A linear code C with length n, dimension k and minimum weight d is called an $[n, k, d]$ code. A generator matrix G for a code C is an n by k matrix whose row space consists of the codewords of C. A parity check matrix H for this code is an n by $n - k$ matrix such that $Hx^T = 0$ for any x in the code. In other words, the row space of H is the subspace of F_2 perpendicular to C.

For a code C with minimum weight d, any binary vector in F_2^n is within Hamming distance $t = \lfloor \frac{d-1}{2} \rfloor$ of at most one codeword; thus, a code with minimum weight d can correct t errors made in the bits of a codeword; such a code is thus said to be a t error correcting code. Suppose we know a vector y which is a codeword with t or fewer errors. All the information needed to correct y is contained in the *syndrome* vector $s = Hy^T$. If the syndrome is 0, we know $y \in C$. Otherwise, we can deduce the positions of the errors from the syndrome. To correct the errors, we need then only apply a NOT to the bits in error. Computing the positions of the errors from the syndrome is in general a hard problem; however, for many codes it can be done in polynomial time.

The dual code $C^\perp$ of a code C is the set of vectors perpendicular to all codewords, that is $C^\perp = \{v \in \mathsf{F}_2^n : v \cdot c = 0 \; \forall c \in C\}$. It follows that if G and H are generator and parity check matrices of a code C, respectively, then H and G are generator and parity check matrices for $C^\perp$.

Suppose we have an $[n, k, d]$ linear code C such that

$$C^{\perp} \subset C \subset \mathrm{F}_2^n. \tag{3.1}$$

We can use C to generate a quantum error correcting code which will correct errors in any $t = (d-1)/2$ or fewer qubits. More details and proofs of the properties of these codes can be found in [11, 27]; in this abstract we only briefly describe results shown in these papers.

We will be using two different expressions for the codewords of our quantum codes. The first is described in [27]. Suppose that $v \in C$. We obtain a quantum state on n qubits as follows:

$$|s_v\rangle = 2^{-(n-k)/2} \sum_{w \in C^{\perp}} |v + w\rangle . \tag{3.2}$$

We refer to this as the s-basis, and will be using it in most of our calculations. It can easily be shown that $|s_v\rangle = |s_u\rangle$ if and only if $u + v \in C^{\perp}$. Thus, there are $\dim C - \dim C^{\perp} = 2k - n$ codewords in our quantum code.

By rotating each of the n qubits in a quantum codeword as follows:

$$\begin{aligned} |0\rangle &\rightarrow \tfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ |1\rangle &\rightarrow \tfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle), \end{aligned} \tag{3.3}$$

it is easy to verify that we obtain the quantum state

$$|c_v\rangle = 2^{-k/2} \sum_{w \in C} (-1)^{v \cdot w} |w\rangle . \tag{3.4}$$

As with the s-basis, it is easily checked that $|c_v\rangle = |c_u\rangle$ if and only if $u + v \in C^{\perp}$. We call this the c-basis for our code; it was first described in [11].

In quantum codes, in order to correct t arbitrary errors, all that is needed is be able to correct any t errors of the following three types [7, 15]:

1) bit errors, where $|0\rangle \rightarrow |1\rangle$ and $|1\rangle \rightarrow |0\rangle$ for some qubit,
2) phase errors, where $|0\rangle \rightarrow |0\rangle$ and $|1\rangle \rightarrow -|1\rangle$ for some qubit,
3) bit-phase errors, where $|0\rangle \rightarrow |1\rangle$ and $|1\rangle \rightarrow -|0\rangle$ for some qubit.

Phase errors can be converted to bit errors and vice versa by applying the change of basis in Equation (3.3). Note that in both the s- and the c-bases, the codewords are superpositions of codewords in our error correcting code C. Thus, to correct errors in the above quantum error correcting code, we can first correct bit errors in the s-basis using classical error correcting techniques, change bases as in (3.3), and then correct phase errors in the s-basis (which have become bit errors in the c-basis) by using the same classical error correcting code techniques. It remains only to show that these two correction steps do not interfere with each other; this is done in detail in [11].

4. Correcting errors fault-tolerantly

To correct errors in quantum codes, we first correct errors in the codewords seen in the $|s\rangle$ basis; we next rotate each qubit of the code as in Equation (3.3) (this could be done symbolically); and finally we correct the errors in the $|c\rangle$ basis. In either basis, the error correction can be done by determining which bits are in error by first computing the (classical) syndrome and then applying a NOT operation (in the appropriate basis) to those bits which are in error. The hard part of this procedure turns out to be the determination of which bits are in error.

Recall that to determine which bits are in error in classical codes, we need to compute the syndrome $S = Hx^T$, where x is the word we are trying to decode. This procedure will work equally well for our quantum codes, although in this case it must be done separately for the bit and the phase errors. It is this step which is hard to do fault-tolerantly. Computing the error from the syndrome is hard for arbitrary linear codes, but if we have made a good choice of our classical code C, then we have a polynomial-time algorithm for computing the error from the syndrome. Note that since we can measure the syndrome, this decoding step can be done on a classical computer. We discuss later in Section 7 classical codes which are both polynomial-time decodable and also strong enough to use to construct fault-tolerant quantum circuits.

The obvious way to measure the syndrome would be the following. For each row of the parity check matrix H, we take an ancillary qubit (or *ancilla*) which starts in the state $|0\rangle$, perform a controlled NOT from each of the qubits corresponding to 1's in that row of the parity check matrix into the ancillary qubit, and finally measure this ancillary qubit. Assuming that we do not make any errors in our quantum calculations, this works perfectly. Unfortunately, this method is not robust against quantum errors. One possible quantum error is a measurement error, where the qubit is measured as $|0\rangle$ instead of $|1\rangle$ or vice versa. If this were the only kind of error possible, it could be controlled by repeating the measurement several times. Unfortunately, this method also permits much worse types of error.

Suppose that we apply the above method, and halfway through the process (i.e., when we have XORed half of the 1's in that row of the parity check matrix H into our ancillary qubit), the state of the ancilla changes spontaneously. We now quite possibly obtain a wrong value for that bit of the syndrome, but much worse, we have also changed the state of the quantum codeword. This can most easily be seen by changing the basis for all the qubits in the codeword, as well as for our ancilla, by Equation (3.3). The effect of this change of basis on an controlled NOT is to reverse the roles of the control and the target qubits. Thus, in the rotated basis, our computation XORs the ancillary qubit $(|0\rangle + |1\rangle)/\sqrt{2}$ into certain qubits of our codeword. If everything proceeds without error, this measurement changes the state of our quantum codeword by XORing $|000\ldots00\rangle + |111\ldots11\rangle$ into the qubits corresponding to 1's in some row of the parity check matrix H. Since H is a generator matrix for the dual code $\mathcal{C}^{\perp}$, it follows after some calculation that this does not change our codeword. However, if the state of the ancillary qubit changes in the middle of our computation, we could end up XORing $|000\ldots11\rangle + |111\ldots00\rangle$ into our quantum codeword. In this scenario, one error during the quantum error correction would possibly lead to more errors in our quantum codeword than it is possible to correct. This technique thus cannot be used for error correction in quantum computation.

We use a slightly different technique to measure the syndrome without introducing too many errors in our quantum codewords. To measure the ith bit of the syndrome, we first construct a "cat state"

$$\frac{1}{\sqrt{2}}(|000\ldots0\rangle + |111\ldots1\rangle) \tag{4.1}$$

where the number of qubits in this state (say l) is equal to the number of 1's in the ith row of our parity check matrix H. (This is called a "cat state" after Schrödinger's renowned cat, as it is the one of the most unstable states of l qubits.) We next *verify* the cat state by measuring the XOR of random pairs of its qubits (this can be done using an auxiliary qubit). If all these measurements are 0, this will ensure that the cat state is a superpositions of states containing nearly all 0's or nearly all 1's, although the relative phase of the all-0's and the all-1's states may still be in error. If these measurements are not all 0's, we construct another cat state and repeat the process.

To use the cat state, we next rotate each qubit of the cat state as in transformation (3.3). If we do not make any errors, this gives a state

$$2^{-(l-1)/2} \sum_{b: b\cdot\bar{1}=0} |b\rangle \tag{4.2}$$

where l is the number of qubits in our cat state and $\bar{1}$ is the length l all-ones vector. In other words, this is the superposition of all states with an even number of 1's. Finally, we XOR each of the qubits of the ith row of the parity check matrix into one of the qubits of the rotated cat state. Since this rotated cat state was in the superposition of all even-parity states, if we now measure the qubits in this state, the parity of number of 1's observed will be the ith bit of the syndrome. More important, even if we have made r errors in our calculation, the back action of the XORs on the encoded state will not introduce more than r errors in qubits of our codeword. Thus, we can measure bits of the syndrome and keep our encoded states well protected. This allows us to correct errors while introducing on the average fewer errors than we correct.

Measurement of the syndrome using the method above is not guaranteed to give the right answer. What it does guarantee is that errors in the measurement operation are unlikely to produce catastrophic back action which destroys the encoded state beyond repair. We still need to ensure somehow that we obtain the right value for the error syndrome before attempting to correct the errors. One way to get the right error syndrome with probability $1 - 1/t$ is to repeat the above measurement $O(\log t)$ times. If we obtain the same error syndrome each time, the probability that we have made the same error repeatedly is very small. If we obtain different syndromes, we can keep repeating the measurement until the same error syndrome is obtained $O(\log t)$ times in a row. If the error rate is set low enough this guarantees that we correct the error with probability at least $1 - 1/t$,

5. XOR Gates and $\frac{\pi}{2}$ Rotations

In order to give our construction for fault-tolerant quantum circuits in detail, we first need to introduce more facts about error correcting codes. These can be found explained in more detail in coding theory books (such as [17]). We will be using codes with $\dim(\mathcal{C}) - \dim(\mathcal{C}^{\perp}) = 1$; they are thus rather inefficient in that they code one qubit into n. The codes we use can be constructed from self-dual codes with $\mathcal{C} = \mathcal{C}^{\perp}$ by deleting any one coordinate; such codes are called *punctured* self-dual codes. If a code $\mathcal{C}$ has minimum distance d, the punctured code has minimum distance at least $d - 1$ and can thus correct $\lfloor d/2 - 1 \rfloor$ errors. Binary self-dual codes have the property that all codewords have an even number of ones, since every codeword must be perpendicular to itself. Some binary self-dual codes have the additional property that the number of ones in all codewords is divisible by 4. These are called *doubly even* codes and their properties are useful in constructing fault-tolerant quantum circuits.

Suppose that we have a punctured self-dual code $\mathcal{C}$ with length n, dimension $k = (n + 1)/2$ and minimum distance d. Consider the corresponding quantum codewords $|s_v\rangle$. From the previous section, we have that the number of different quantum codewords is $2k - n = 1$. It is easy to verify that one of these consists of the superposition of all codewords in $\mathcal{C}$ with even weight (these are the codewords of $\mathcal{C}^{\perp}$) and the other consists of the superposition of all codewords of $\mathcal{C}$ of odd weight. We label these $|s_0\rangle$ and $|s_1\rangle$ respectively. It is also easy to see that the c-basis of our code looks like:

$$\begin{aligned} |c_0\rangle &= \frac{1}{\sqrt{2}}(|s_0\rangle + |s_1\rangle), \\ |c_1\rangle &= \frac{1}{\sqrt{2}}(|s_0\rangle - |s_1\rangle). \end{aligned} \tag{5.1}$$

Recall that we get from the s-basis to the c-basis of our quantum codes by applying the transformation (3.3) to each qubit of our codewords. It is clear from the above equations that applying this transformation to each qubit individually also applies this transformation to the encoded states. Further, this transformation is fault-tolerant. Suppose that there were at most r errors in qubits of the input state. This transformation is applied separately to each qubit, so the output state will also contain r errors in it if the transformation was applied perfectly. Even if the transformation is imperfect, an error in one application of a quantum gate can only affect one qubit since the transformation is applied bitwise, so even with noisy gates, this transformation can introduce only a small number of errors,

A number of other transformations of the encoded qubits can also be performed by applying them bitwise to the codeword. We demonstrate this with an XOR gate. Suppose we have two different qubits encoded, $|s_a\rangle$ and $|s_b\rangle$. Expanding these, we have the quantum state

$$|s_a\rangle|s_b\rangle = 2^{-(n-k)} \sum_{w\in\mathcal{C}^\perp} |w+a\rangle \sum_{w'\in\mathcal{C}^\perp} |w'+b\rangle \quad (5.2)$$

Applying an XOR from the ith qubit of the first codeword into the ith qubit of the second codeword, we obtain

$$2^{-(n-k)} \sum_{w\in\mathcal{C}^\perp} \Big(|w+a\rangle \sum_{w'\in\mathcal{C}^\perp} |w'+b+w+a\rangle\Big). \quad (5.3)$$

If w' ranges over all codewords in $\mathcal{C}^\perp$, then for any fixed $w \in \mathcal{C}^\perp$, $w'+w$ also ranges over all codewords in $\mathcal{C}^\perp$. Thus, the above sum can be rewritten as $|s_a\rangle|s_{a+b}\rangle$. This operation works for any $\mathcal{C}$ with $\mathcal{C}^\perp \subset \mathcal{C}$ as in Equation (3.1). For punctured self-dual codes $\mathcal{C}$, with $\dim \mathcal{C}^\perp = \dim \mathcal{C} - 1$, this gives an XOR gate.

Other operations which can also be done bitwise are the phase change operation $|s_a\rangle|s_b\rangle \rightarrow (-1)^{a\cdot b}|s_a\rangle|s_b\rangle$ and (for punctured doubly even self-dual codes), the rotation

$$\begin{aligned} |s_0\rangle &\rightarrow |s_0\rangle \\ |s_1\rangle &\rightarrow i|s_1\rangle. \end{aligned} \quad (5.4)$$

The calculations for both of these cases are straightforward.

These above operations which can be done bitwise are not enough to provide a universal set of gates for quantum computation. They only generate unitary matrices in the *Clifford group*, which is a finite group of unitary transformations in 2^n-dimensional complex space that arises in several areas of mathematics [10]. The transformations in this group corresponding to classical computation are the *linear Boolean* transformations, which can be built out of XOR and NOT gates. To obtain a set of gates universal for quantum computation, we add the Toffoli gates as in Equation (2.5). This construction is discussed in the next section.

6. Toffoli gates

We construct our Toffoli gate in two stages. We first show how to construct a fault-tolerant Toffoli gate given a set of ancillary quantum bits known to be in the encoded state $\frac{1}{2}(|s_0s_0s_0\rangle + |s_0s_1s_0\rangle + |s_1s_0s_0\rangle + |s_1s_1s_1\rangle)$. This procedure is done using only linear Boolean operations and $\pi/2$ rotations on the encoded qubits. We next show how to fault-tolerantly put a set of ancillary qubits into the above state. This operation will be somewhat harder, as it cannot be done using operations in the Clifford group. To construct this state, we use a "cat" state $(|000\ldots0\rangle+|111\ldots1\rangle)/\sqrt{2}$ as we did in Section 4.

A technique used both in this section and in Section 4 is that of first constructing an ancillary set of qubits known to be in a certain state and then using them to perform operations on another set of qubits. This is reminiscent of techniques used in several quantum communication papers. In quantum teleportation [5], if two researchers share an EPR pair, they can use this pair and classical communication to "teleport" the quantum state of a particle from one researcher to another. In [6], a small number of "USDA" pairs of qubits known to be in perfect EPR states can used to purify a set of noisy EPR states, sacrificing some of them but yielding a large set of good EPR pairs. This paradigm may prove useful in other quantum computations.

6.1 Using the ancillary state

Suppose we had an ancillary set of qubits known to be in the encoded state

$$|A\rangle = \frac{1}{2}(|s_0s_0s_0\rangle+|s_0s_1s_0\rangle+|s_1s_0s_0\rangle+|s_1s_1s_1\rangle). \quad (6.1)$$

We now show how to use these to make a Toffoli gate on 3 other encoded qubits, using Boolean linear operations and $\pi/2$ rotations. Recall that the Toffoli gate transforms qubits by negating the third qubit if and only if the first two are 1's. We first build a gate that makes the following transformation taking two encoded qubits to three encoded qubits.

$$\begin{aligned} |s_0s_0\rangle &\rightarrow |s_0s_0s_0\rangle \\ |s_0s_1\rangle &\rightarrow |s_0s_1s_0\rangle \\ |s_1s_0\rangle &\rightarrow |s_1s_0s_0\rangle \\ |s_1s_1\rangle &\rightarrow |s_1s_1s_1\rangle. \end{aligned} \quad (6.2)$$

Note that this gate adds a third (encoded) qubit, which is a 1 if and only if the first two are both 1's, and which is 0 otherwise. This gate uses the ancillary state $|A\rangle$ described above, as well as linear operations, which can be performed robustly as in Section 5.

To perform these transformations, we first append the ancilla $|A\rangle$ to the first two qubits. We next XOR the third qubit into the first, and the fourth qubit into the second. This produces the transformation

$$\begin{array}{rcl}
|s_0s_0\rangle|A\rangle & \to & \frac{1}{2}\big(\, |s_0s_0s_0s_0s_0\rangle + |s_0s_1s_0s_1s_0\rangle \\
& & +|s_1s_0s_1s_0s_0\rangle + |s_1s_1s_1s_1s_1\rangle\,\big) \\
|s_0s_1\rangle|A\rangle & \to & \frac{1}{2}\big(\, |s_0s_1s_0s_0s_0\rangle + |s_0s_0s_0s_1s_0\rangle \\
& & +|s_1s_1s_1s_0s_0\rangle + |s_1s_0s_1s_1s_1\rangle\,\big) \\
|s_1s_0\rangle|A\rangle & \to & \frac{1}{2}\big(\, |s_1s_0s_0s_0s_0\rangle + |s_1s_1s_0s_1s_0\rangle \\
& & +|s_0s_0s_1s_0s_0\rangle + |s_0s_1s_1s_1s_1\rangle\,\big) \\
|s_1s_1\rangle|A\rangle & \to & \frac{1}{2}\big(\, |s_1s_1s_0s_0s_0\rangle + |s_1s_0s_0s_1s_0\rangle \\
& & +|s_0s_1s_1s_0s_0\rangle + |s_0s_0s_1s_1s_1\rangle\,\big)\,.
\end{array} \tag{6.3}$$

Finally, we measure the first and second encoded qubits.

Suppose we measure them to be $|s_0s_0\rangle$, so both encoded qubits are 0. Focusing on the elements of the superposition where the first and second encoded qubits are both 0, we get the transformation given by (6.2), which is what we wanted in the first place. Suppose, however, that we measure the first and second encoded qubits to be in the state $|s_0s_1\rangle$. Pulling out the relevant elements of the superposition, we get

$$\begin{array}{rcl}
|s_0s_0\rangle & \to & |s_0s_1s_0\rangle \\
|s_0s_1\rangle & \to & |s_0s_0s_0\rangle \\
|s_1s_0\rangle & \to & |s_1s_1s_1\rangle \\
|s_1s_1\rangle & \to & |s_1s_0s_0\rangle\,.
\end{array} \tag{6.4}$$

This transformation can be converted to the one we want by first applying a controlled NOT from the first qubit to the third qubit, and then applying a NOT to the second qubit. These are both Boolean linear operations and so can be applied fault-tolerantly. Putting these transformations together, we get

$$\begin{array}{l}
|s_0s_0\rangle \to |s_0s_1s_0\rangle \to |s_0s_1s_0\rangle \to |s_0s_0s_0\rangle \\
|s_0s_1\rangle \to |s_0s_0s_0\rangle \to |s_0s_0s_0\rangle \to |s_0s_1s_0\rangle \\
|s_1s_0\rangle \to |s_1s_1s_1\rangle \to |s_1s_1s_0\rangle \to |s_1s_0s_0\rangle \\
|s_1s_1\rangle \to |s_1s_0s_0\rangle \to |s_1s_0s_1\rangle \to |s_1s_1s_1\rangle.
\end{array} \tag{6.5}$$

It is easy to check that the other two cases (where we observe $|s_1s_0\rangle$ or $|s_1s_1\rangle$), can also be corrected to the desired gate by linear operations. Thus, by observing two of these five qubits and applying linear operations, we have achieved what is nearly a Toffoli gate.

We still need to show how to get the complete Toffoli gate on three qubits, as in Equation (2.5). To do this, we start with three qubits to which we want to apply the Toffoli gate, and apply transformation (6.2) to the first two. We next apply a controlled NOT from our original third qubit (represented in fourth place below) to the newly introduced qubit (represented in third place below). We finally apply $|s_0\rangle \to (|s_0\rangle + |s_1\rangle)/\sqrt{2}$, $|s_1\rangle \to (|s_0\rangle - |s_1\rangle)/\sqrt{2}$ to the original third qubit (represented in fourth place below). This gives the transformation:

$$\begin{array}{rcl}
|s_0s_0s_0\rangle & \to & \frac{1}{\sqrt{2}}|s_0s_0s_0\rangle\big(|s_0\rangle + |s_1\rangle\big) \\
|s_0s_1s_0\rangle & \to & \frac{1}{\sqrt{2}}|s_0s_1s_0\rangle\big(|s_0\rangle + |s_1\rangle\big) \\
|s_1s_0s_0\rangle & \to & \frac{1}{\sqrt{2}}|s_1s_0s_0\rangle\big(|s_0\rangle + |s_1\rangle\big) \\
|s_1s_1s_0\rangle & \to & \frac{1}{\sqrt{2}}|s_1s_1s_1\rangle\big(|s_0\rangle + |s_1\rangle\big) \\
|s_0s_0s_1\rangle & \to & \frac{1}{\sqrt{2}}|s_0s_0s_1\rangle\big(|s_0\rangle - |s_1\rangle\big) \\
|s_0s_1s_1\rangle & \to & \frac{1}{\sqrt{2}}|s_0s_1s_1\rangle\big(|s_0\rangle - |s_1\rangle\big) \\
|s_1s_0s_1\rangle & \to & \frac{1}{\sqrt{2}}|s_1s_0s_1\rangle\big(|s_0\rangle - |s_1\rangle\big) \\
|s_1s_1s_1\rangle & \to & \frac{1}{\sqrt{2}}|s_1s_1s_0\rangle\big(|s_0\rangle - |s_1\rangle\big)\,.
\end{array} \tag{6.6}$$

We now observe the fourth qubit in the expression above. If we observe $|s_0\rangle$, it is easy to see that we have performed a Toffoli gate. If we observe $|s_1\rangle$, we need to fix the resulting state up. This can be done by applying the transformation

$$|s_as_bs_c\rangle \to (-1)^{a\cdot b}(-1)^c|s_as_bs_c\rangle \tag{6.7}$$

to the three remaining encoded qubits (recall that we measured the last qubit). This is the composition of the two linear operations $|s_as_b\rangle \to (-1)^{a\cdot b}$ and $|s_c\rangle \to (-1)^c$, so it can be done fault-tolerantly using the methods of Section 5.

6.2 Constructing the ancillary state

For the last piece of our algorithm, we need to show how to construct the ancillary state $\frac{1}{2}(|s_0s_0s_0\rangle + |s_0s_1s_0\rangle + |s_1s_0s_0\rangle + |s_1s_1s_1\rangle)$ fault-tolerantly. To obtain this, we use the technique we used in Section 4 of introducing a "cat" state $(|000\ldots0\rangle + |111\ldots1\rangle)/\sqrt{2}$, which we can check to ensure that it is in a state close to the desired state before we use it. We use two states on $3n$ qubits, the state $|A\rangle$ defined in (6.1) and the state $|B\rangle$, which are as follows:

$$\begin{array}{rcl}
|A\rangle & = & \frac{1}{2}(|s_0s_0s_0\rangle + |s_0s_1s_0\rangle + |s_1s_0s_0\rangle + |s_1s_1s_1\rangle) \\
|B\rangle & = & \frac{1}{2}(|s_0s_0s_1\rangle + |s_0s_1s_1\rangle + |s_1s_0s_1\rangle + |s_1s_1s_0\rangle)\,.
\end{array}$$

Note that $|A\rangle$ is the state we want, and $|B\rangle$ is easily convertible to it by applying a NOT operation to the third encoded qubit. Note also that

$$\tfrac{1}{\sqrt{2}}(|A\rangle+|B\rangle) = \tfrac{1}{2\sqrt{2}}(|s_0\rangle+|s_1\rangle)(|s_0\rangle+|s_1\rangle)(|s_0\rangle+|s_1\rangle)\,, \tag{6.8}$$

which is easily constructible through operations discussed in Section 5. Let us now introduce some more notation: let $|\bar{0}\rangle = |000\ldots0\rangle$ and $|\bar{1}\rangle = |111\ldots1\rangle$, where these states are both on n qubits. Thus, $(|\bar{0}\rangle + |\bar{1}\rangle)/\sqrt{2}$ is the cat state described above. Suppose we could apply a transformation that changed states as follows:

$$\begin{aligned} |\bar{0}\rangle|A\rangle &\rightarrow |\bar{0}\rangle|A\rangle \\ |\bar{1}\rangle|A\rangle &\rightarrow |\bar{1}\rangle|A\rangle \\ |\bar{0}\rangle|B\rangle &\rightarrow |\bar{0}\rangle|B\rangle \\ |\bar{1}\rangle|B\rangle &\rightarrow -|\bar{1}\rangle|B\rangle\,, \end{aligned} \tag{6.9}$$

We would then be able to construct state $|A\rangle$ as follows. First, construct

$$\tfrac{1}{2}(|\bar{0}\rangle + |\bar{1}\rangle)(|A\rangle + |B\rangle)\,, \tag{6.10}$$

where the cat state has been verified as in Section 4 to make sure it is close to the desired cat state (i.e., nearly all the probability amplitude is concentrated in states with either nearly all 0's or nearly all 1's). We next apply the transformation (6.9) to obtain the state

$$\tfrac{1}{2}(|\bar{0}\rangle + |\bar{1}\rangle)|A\rangle + \tfrac{1}{2}(|\bar{0}\rangle - |\bar{1}\rangle)|B\rangle\,. \tag{6.11}$$

Finally, we observe whether the "cat qubits" are in the state $|\bar{0}\rangle \pm |\bar{1}\rangle$. This tells us whether the unobserved qubits contain $|A\rangle$ or $|B\rangle$.

The probability of being in state $|A\rangle$ can be estimated using the probabilities of error during the quantum calculation. If this probability is not high enough, we repeat this step, applying the transformation (6.9) not to the state (6.10) but to the output state from the previous step along with a newly constructed "cat state." Repeating this step logarithmically many times can be shown to increase this probability of being in state $|A\rangle$ to polynomially close to 1.

To apply the transformation (6.9) to a superposition $|\bar{a}\rangle|s_b\rangle|s_c\rangle|s_d\rangle$, it is sufficient to apply bitwise the operation

$$|a_i\rangle|b_i\rangle|c_i\rangle|d_i\rangle \rightarrow (-1)^{a_i(b_ic_i+d_i)}|a_i\rangle|b_i\rangle|c_i\rangle|d_i\rangle \tag{6.12}$$

to the ith qubit of $|\bar{a}\rangle$, $|s_b\rangle$, $|s_c\rangle$ and $|s_d\rangle$ for $1 \le i \le n$. This operation is easily accomplished by elementary quantum gates and as it is a bitwise operation, it is fault-tolerant.

The one piece of the computation which we have not yet described is how to construct an encoded state $|s_0\rangle$. This can be done by techniques similar to those described in this paper, but there is no space to describe this in detail.

7. Conclusions

We now estimate the accuracy required to make quantum circuits fault-tolerant with these methods. The only large binary self-dual codes I know of which are also decodable in polynomial time are the $[2^{m+1}, 2^m, 2^{m/2}]$ self-dual Reed-Muller codes [17]. These codes are indeed doubly even, but unfortunately their minimum distance (and error correction capacity) grows as the square root of their length n. This will be enough for our purposes, but these codes are substantially worse than both the theoretical maximum and than the known constructions without all the required properties.

In order to do s steps of quantum computation with a low probability of failure, we need a quantum code which can correct $O(\log s)$ errors. Using Reed-Muller codes, this means we need codewords of length $O(\log^2 s)$. In measuring the syndrome, to be relatively sure that we have not made an error in computing it, we measure each bit of the syndrome $O(\log s)$ times (this is probably overkill). Using the measuring technique described in Section 4, even if we make errors while measuring the syndrome, we do not substantially affect our encoded qubits.

Computing the error syndrome requires a number of quantum gates proportional to the number of 1's in the parity check matrix, which in this case is $O(\log^3 s)$. Since we measure the syndrome $O(\log s)$ times in our correction step, the entire correction step takes $O(\log^4 s)$ operations. We need to set the error rate low enough so that there will be less than one error on the average throughout this process, which means we must have error rate less than $O(1/\log^4 s)$ per gate operation. Computation operations on encoded qubits take at most $O(\log^4 s)$ steps, so these can also be accomplished while ensuring that there is a very small probability of making more errors than we can correct.

One additional result which would be nice is the discovery of better binary self-dual error correcting codes which are also efficiently decodable; this could substantially increase the asymptotic efficiency of the fault-tolerant quantum circuits described in this paper. Another interesting result would be a method for performing rotations on encoded bits directly, rather than going through Toffoli gates. This could be accomplished using techniques similar to those in Section 6 if the ancillary state $\cos(\theta)|s_0\rangle + \sin(\theta)|s_1\rangle$ could be constructed fault-tolerantly for arbitrary θ.

The techniques in this paper pay a moderate penalty in both space and time for making quantum circuits fault-tolerant. An interesting question is how much time and space are really required. The space, for example, could likely be reduced significantly by using more efficient quantum error correcting codes for memory. Another interesting question is how much noise in quantum gates can be tolerated while still permitting quantum computation. A lower bound on this quantity is shown in [1]. Better error-correcting codes could possibly increase the maximum allowed error rate considerably from $O(1/\log^4 t)$, but it appears that to get results better than $O(1/\log t)$, substantially different techniques may be required.

Finally, the analysis in this paper is purely asymptotic. An analysis needs to be done to see how much fault-tolerance these techniques provide for quantum computations using specific numbers of gates, and at what cost in space and time this could be accomplished.

Acknowledgements

I would like to thank Rolf Landauer and Bill Unruh for their skepticism, which in part motivated this work. I would also like to thank David DiVincenzo for insightful discussions on quantum fault tolerant computing, David DiVincenzo and David Johnson for helpful comments on earlier versions of this paper, and Rob Calderbank and Neil Sloane for helpful discussions on classical error correcting codes.

References

1. D. Aharonov and M. Ben-Or, "Polynomial simulations of decohered quantum computers," in this *Proceedings.*
2. D. Aharonov and N. Nisan, "Quantum circuits with mixed states," in this *Proceedings.*
3. A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin and H. Weinfurter, "Elementary gates for quantum computation," *Phys. Rev. A* **52,** pp. 3457–3467 (1995).
4. C. H. Bennett, E. Bernstein, G. Brassard and U. Vazirani, "Strengths and weaknesses of quantum computation," *SIAM J. Computing,* to appear.
5. C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres and W. K. Wooters, "Teleporting an unknown quantum state via dual classical and EPR channels," *Phys. Rev. Lett.* **74,** pp. 2619–2622 (1993).
6. C. H. Bennett, G. Brassard, S. Popescu, B. Schumacher, J. A. Smolin and W. K. Wooters, "Purification of noisy entanglement and faithful teleportation via noisy channels," *Phys. Rev. Lett.* **76,** pp. 722–725 (1996).
7. C. H. Bennett, D. DiVincenzo, J. A. Smolin and W. K. Wooters, "Mixed state entanglement and quantum error correction," *Phys. Rev. A,* submitted; also LANL e-print* `quant-ph/9604024`.
8. M. Ben-Or, S. Goldwasser and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," *Proc. 20th Ann. ACM Symp. on Theory of Computing,* ACM Press, New York, pp. 1–10 (1988).
9. E. Bernstein and U. Vazirani, "Quantum complexity theory," manuscript (1996); a preliminary version appeared in *Proc. 25th Ann. ACM Symp. on Theory of Computing,* ACM Press, NY, pp. 11–20 (1993).
10. A. R. Calderbank, P. J. Cameron, W. M. Kantor, and J. J. Seidel, "$\mathbb{Z}_4$ Kerdock codes, orthogonal spreads, and extremal Euclidean line-sets," *Proc. London Math. Soc.,* to appear.
11. A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist," *Phys. Rev. A,* to appear (August 1996); also LANL e-print* `quant-ph/9512032`.
12. I. L. Chuang, R. Laflamme, P. W. Shor and W. H. Zurek, "Quantum computers, factoring and decoherence," *Science* **270,** pp. 1633–1635 (1995).
13. J. I. Cirac and P. Zoller, "Quantum computations with cold trapped ions," *Phys. Rev. Lett.* **74,** pp. 4091–4094 (1995).
14. R. L. Dobrushin and S. I. Ortyukov, "Lower bound for the redundancy of self-correcting arrangements of unreliable functional elements," *Probl. Inf. Transm.* **13**(3), pp. 203–218 (1977) [English translation of *Probl. Peredachi Inf.*].
15. A. Ekert and C. Macchiavello, "Error correction in quantum communication," LANL e-print* `quant-ph/9602022` (1996).
16. R. Landauer, "Is quantum mechanics useful?" *Phil. Trans. Roy. Soc. London Ser. A* **353,** pp. 367–376 (1995).
17. F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes,* North-Holland, Amsterdam (1977).
18. C. Monroe, D. M. Meekhof, B. E. King, W. M. Itano and D. J. Wineland, "Demonstration of a universal quantum logic gate," *Phys. Rev. Lett.* **75,** pp. 4714–4717 (1995).
19. A. Peres, *Quantum Theory: Concepts and Methods,* Kluwer Academic, Dordrecht (1993).
20. N. Pippenger, "Invariance of complexity measures for networks with unreliable gates," *J. ACM* **36,** pp. 531–539 (1989).
21. N. Pippenger, G. D. Stamoulis and J. N. Tsitsiklis, "On a lower bound for the redundancy of reliable networks with noisy gates," *IEEE Trans. Inf. Theory* **37,** pp. 639–643 (1991).
22. D. A. Rennels, "Fault-tolerant computing — Concepts and examples," *IEEE Trans. Comput.* **C-33,** pp. 1116–1129 (1984).
23. P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th Ann. Symp. on Foundations of Computer Science,* IEEE Computer Society Press, Los Alamitos, CA, pp. 124–134 (1994).
24. P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Phys. Rev. A* **52,** pp. 2493–2496 (1995).
25. P. W. Shor, "Polynomial time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Computing,* to appear; also LANL e-print* `quant-ph/9508027`.
26. D. Simon, "On the power of quantum computation," in *Proc. 35th Ann. Symp. on Foundations of Computer Science,* IEEE Computer Society Press, Los Alamitos, CA, pp. 116–123 (1994).
27. A. M. Steane, "Multiple particle interference and quantum error correction," *Proc. Roy. Soc. London Ser. A,* to appear; also LANL e-print* `quant-ph/9601029`.
28. W. G. Unruh, "Maintaining coherence in quantum computers," *Phys. Rev. A* **51,** pp. 992–997 (1995).
29. J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," in *Automata Studies* (C. E. Shannon and J. McCarthy, eds.), Princeton University Press, Princeton, NJ, pp. 329–378 (1956).
30. A. Yao, "Quantum circuit complexity," in *Proc. 34th Ann. Symp. on Foundations of Computer Science,* IEEE Computer Society Press, Los Alamitos, CA, pp. 352–361 (1993).
31. W. H. Zurek, "Decoherence and the transition from quantum to classical," *Physics Today* **44**(10), pp. 36–44 (1991).
32. W. H. Zurek and R. Laflamme, "Quantum logical operations on encoded qubits," LANL e-print* `quant-ph/9605013` (1996).

*LANL e-prints are available at `http://xxx.lanl.gov` from the Los Alamos National Laboratory e-print server.