

Preliminary Results On Simulating a Scalable Fault Tolerant Ion-Trap System for Quantum Computation

Tzvetan Metodiev[‡], Andrew Cross[†], Darshan Thaker[‡], Kenneth Brown[†], Dean Copsey[‡], Frederic T. Chong[‡], and I.L. Chuang[†]

[†]Center for Bits and Atoms, Massachusetts Institute of Technology, Cambridge MA 02139 and

[‡]University of California Davis, Davis, CA

Quantum information processors face the challenge of operating reliably despite the intrinsic component unreliability due to decoherence. This can be avoided by strategic use of quantum error correction to build a fault-tolerant quantum computer, as long as the component failure probability p_0 is less than a critical threshold p_{th} . This threshold value is an excellent, comprehensive measure of the scalability of a particular physical implementation, and although it can theoretically be as high as 10^{-3} , its actual value depends highly on implementation specifics, such as T_1 and T_2 lifetime parameters, qubit couplings, and control and communication errors. Here, we numerically evaluate p_{th} for an ion trap quantum computer architecture, using a Monte-Carlo simulation built on existing experimentally determined performance parameters, including geometric layout constraints, and techniques adopted from classical computer architecture design.

I. INTRODUCTION

Quantum information processing offers the promise of computational speedups through the clever utilization of hitherto untapped quantum-mechanical resources available with nanoscale engineering. However, the most crucial challenge facing practical experimental realization of such quantum computing systems is *reliability*, because quantum effects intrinsically suffer from degradation of quantum behavior.

This loss can be countered efficiently by systematic application of quantum error correction, and remarkably, the theory of fault-tolerant quantum computation predicts that once component error probabilities fall below some threshold p_{th} , which may be as high as 10^{-4} , an arbitrarily reliable system can be constructed, using only such faulty parts. For real systems, however, the actual value of p_{th} can vary wildly; it depends heavily on fundamental performance parameters, including qubit T_1 and T_2 lifetime values, qubit-qubit coupling strengths, one and two-qubit logic gate operation times, gate control error probabilities, and errors due to movement of quantum information. Many other underlying system assumptions, such as the availability of fast, parallel classical computation, also factor into the final value of p_{th} .

There are many physical implementation candidates for quantum computation and each one has the potential to one day be realized. A requirement for any implementation is to be able to compute over a very large number of quantum bits (qubits), a feat that can only be realized if the system can achieve p_{th} sufficiently high that component failure probabilities can be below this threshold. If such a threshold value exists then the quantum system will be scalable enough to undertake an arbitrary length of computation. For this reason, determining p_{th} is a very interesting, and central question for each physical implementation candidate.

Here, we take a first step to explicitly and numerically determine p_{th} for a leading implementation candidate for quantum computers – the ion trap quantum information processor [3–5, 7, 8, 10, 13, 14, 23, 24, 30]. We obtain this result through a detailed architectural simulation of the quantum charge-coupled device (QCCD) ion trap architecture proposed by Kielpinski et al [3], founded on ongoing experimen-

tal work at NIST in the Wineland group [10, 11], using parameters from real experimental data for qubit lifetimes, gate fidelities, and communication errors. Quantum computers, almost by definition, are exponentially hard to simulate with a classical computer, but we sidestep this difficulty here, using the stabilizer method of Gottesman and Knill [17, 18, 25] to allow efficient classical simulation of just a subset of quantum circuits, those which perform quantum error correction. Thus, we simulate systems with up to hundreds of qubits, and can vividly explore the interplay of device performance parameters with system-level geometric layout constraints. Building on earlier architectural studies of quantum computers [9, 15], we find a natural architectural structure for recursive error correction in the ion trap QCCD framework, allowing us to obtain numerical values for p_{th} .

These results are presented as follows. Section II reviews the current status of ion-trap technology and introduces the model we have adopted for subsequent simulations. This is followed by a brief introduction to fault-tolerant quantum computation in Section III. Section IV gives simulation methods together with an approach to concatenated coding and recursive layout in ion trap QCCD architectures. Finally, section V describes results and expectations for recursive error correction in these systems.

II. ION-TRAPS AND THEIR REPRESENTATION

We begin by reviewing the current state of ion trap quantum information processor implementations, focusing on the QCCD model employed by the NIST group. Included is a description of the abstract physical model we have adopted to accurately simulate trapped-ion systems, which follows closely with the current experimental results [3, 8, 10, 12, 14, 16, 21, 22, 24].

A. The Ion-Trap System

Ion-trap quantum computation was initially proposed by Cirac and Zoller [7]. They suggested that a number of ions

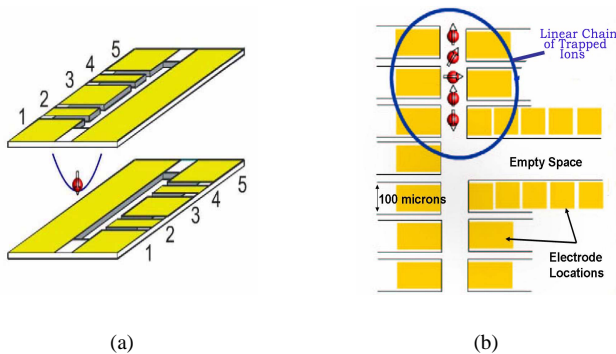


FIG. 1: (a) Schematic of a dual ion-trap constructed using $200\mu\text{m}$ thick alumina wafers spaced $360\mu\text{m}$ apart. Electrodes 2 and 4 are the trap locations. (b) The proposed QCCD structure is composed of large arrays of interconnected traps. Each trap length is approximately $100\mu\text{m}$ and is represented by a pair of orange squares. Both images courtesy of D. Wineland and D. Kielpinski [21].

trapped in a linear RF trap interacting with laser beams can be used as a quantum information processor. Two internal atomic states of each ion are identified with a qubit. However, this proposal does not scale to extremely large number of qubits. As the length of the ion chain increases, the vibrational modes become progressively harder to identify [14], decreasing the gate fidelities. In this section we briefly describe a more elaborate proposal by D. Kielpinski *et al.*, [3, 11, 14, 21] that suggests using a large-scale quantum charge-coupled device (QCCD) consisting of interconnected ion-traps. Multiple traps allow for smaller ion chains and thus greater control over logical operations. In this model, ions are stored in memory regions and moved to interaction regions for manipulation.

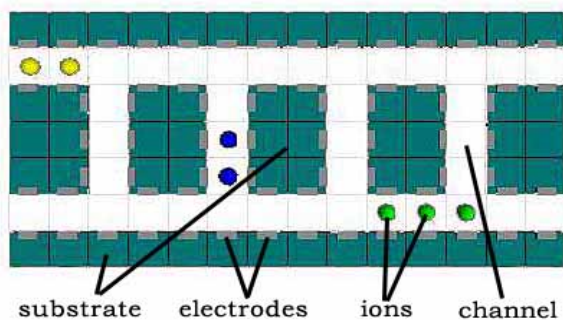


FIG. 2: Layout of a QCCD-style architecture as drawn by the simulation software. Each non-empty location in the grid can contain electrodes on an alumina plaquette or a single ion. In the simulation, ions are colored according to their function.

The physical realization of the QCCD model is possible with current micro-machining techniques in alumina. Figure 1(a) shows a schematic of a dual trap made with $200\mu\text{m}$ thick alumina wafers separated by $360\mu\text{m}$ [14, 16]. Individual ions are trapped near electrodes 2 and 4.

Figure 1(b) shows the proposed trap micro-array as a two dimensional grid. Kielpinski *et al.* suggests that the grid can

be physically realized by stacking silicon wafers together in three layers [21]. The middle layer carries a DC voltage used for axial confinement of the ions, and the outer layers contain an RF voltage needed for the radial confinement. The length of each electrode would be about $100\mu\text{m}$. Two hyperfine sub-levels $|\downarrow\rangle$ and $|\uparrow\rangle$ of an electronic state of Be^+ define the qubit. Optical Raman transitions [8, 14] provide the coupling between $|\downarrow\rangle$ and $|\uparrow\rangle$ to generate superposition states.

B. Ion-Trap Physical Model and Representation

The simulator's representation of an ion-trap processor is shown in Figure 2 and is designed to capture the essential elements of the proposed scheme of Figure 1(b). The QCCD architecture is a two-dimensional grid that specifies placement of trap structures such as the alumina substrate, electrodes, and ions. Internally the layout is a matrix of cells where each is filled with a substrate material or is empty. Electrodes can be attached to the substrate cells, and empty cells can be occupied by ions.

A single qubit of information is carried by the ion-qubits. Multiple ion-qubits can be associated and grouped into a linear *chain* and are coupled via shared vibrational modes. A two qubit gate can be applied to any two chained qubits regardless of their distance from each other, however physical limitations restrict the number of qubits in a given chain to less than ten. Chains can be split apart into two or more smaller chains.

1. Gates and Measurement

Single qubit gates consist of any arbitrary rotations in the x-y plane of the Bloch sphere and are performed by applying a laser pulse on the target ion-qubit. For simplicity, the model assumes that this single laser beam performs all single qubit gates with the same failure rate and gate duration. For single qubit gates, the failure probability is $p_f \approx 10^{-4}$ and the execution time is $1\mu\text{s}$.

Two qubit gates are physically implemented with several laser pulses on each of the two ion-qubits. For example a controlled-phase gate that changes the sign of the qubit's phase if the control qubit is $|1\rangle$ consists of three laser pulses. As with the one qubit gate, we have made the simplification that all two qubit gates have the same duration and failure rate. For two qubit gates the execution time is on the order of $10\mu\text{s}$ and the failure probability is about 0.03.

A readout laser is used to measure ion-qubits. If the ion is in the $|0\rangle$ state many photons are scattered and measurement is recorded, otherwise very few are scattered and a measurement of $|1\rangle$ is recorded. The graphical representation of measurement in the simulator cannot be distinguished from that of a single gate, since both apply a laser to a single qubit. Measurement is a rather lengthy process that takes around $100\mu\text{s}$ and fails about 1% of the time.

Operation	Time	Prob. Failure
Single Gate	1 μ s	0.0001
Double Gate	10 μ s	0.03
Measure	100 μ s	0.01
Movement	10ns/ μ m	0.005/ μ m
Split	10 ⁻³ s	
Cooling	10 ⁻³ s	
Memory time	100s	

TABLE I: Summary of the experimentally determined physical parameters for ion-traps. Note that cooling and splitting ions are by far the two slowest operations in the table. Architectures must be designed to avoid unnecessarily cooling and splitting ion chains. The movement, splitting, and cooling times and the failure rates are taken from [14]. The gate parameters are taken from [10] and [12].

2. Quantum Communication via Ballistic Transport

In one experiment [14], a single ion was transferred 10⁶ times between two trapping electrodes separated by 1.2mm while preserving coherence. The physical displacement of the ion-qubit is termed as *ballistic transport*. Each transfer process took $\approx 50\mu$ s with a $\approx 50\mu$ s wait between transfers. We model the rate at which the qubit dephases during transport as a failure probability of ~ 0.005 per micron moved. The main cause of decoherence during movement is not movement itself, but the exposure of the ion-qubit to fluctuating background fields as it moves.

Another cause of decoherence is the *heat* acquired by the ions as they move (i.e. their vibrational motion intensifies), thus ions must be cooled upon arrival. If specific individual ions in a large linear ion chain need to be moved, a potential wedge must be created to separate those ions from the rest of the chain. Chain splitting heats all of the ions involved and takes a comparably large amount of time (see Table I).

Table I summarizes current physical parameters as observed experimentally. Single qubit gates are fast and precise – from the table they are performed with a probability of failure $p_f = 10^{-4}$ and a 1 μ s execution time. On the other hand, two qubit gates fail 3% of the time and are a factor of 10 slower than their single qubit counterparts.

III. FAULT-TOLERANCE AND THE THRESHOLD

In this section we briefly introduce the theory of fault tolerant computation in the context of ion-traps. Prior knowledge of some error correction terminology is assumed. The reader is referred to [2, 19, 25, 27, 29] for a full description and derivation of the theory of fault tolerance in quantum computation.

The theory of fault-tolerant quantum computation has been developed to prevent and even counteract the build-up of errors in a quantum computer. The ability to arbitrarily improve the reliability of the computation in the presence of faulty system components is the definition of a fault-tolerant architec-

ture.

Fault tolerance is achieved through concatenated coding and repeated error correction at each level of encoding after each step or group of steps. To see how this works, let p_0 be the component failure rate for all the components of some architecture. The failure probability of a collection of components will be a polynomial in p_0 . For the remaining discussion, assume p_0 is small enough that only the lowest order term of the Taylor expansion contributes significantly to the overall failure probability of the circuit. If the code can correct at most one error, then the probability that there will be two or more errors in the data qubits after one level of encoding is approximately cp_0^2 , where c is the number of ways two or more errors can occur. As the recursion level is increased, the probability of failure scales as $\frac{1}{c}(cp_0)^{2^k}$, where k denotes the level of recursion.

If we have a computation requiring N operations with total failure rate p_f and desire an overall success rate $1 - \epsilon$, then we can write $1 - \epsilon = (1 - p_f)^N$ assuming that each step fails independently of prior and future steps. Solving for p_f we see that $p_f = 1 - (1 - \epsilon)^{1/N}$, thus we must increase the level of encoding k until:

$$\frac{1}{c}(cp_0)^{2^k} \leq 1 - (1 - \epsilon)^{1/N} \quad (3.1)$$

Note that such a k cannot be found if $p_0 > 1/c$. Such a result is known as the *threshold condition* for fault tolerance [2, 28]. Provided that the component failure rate $p_0 < p_{th} \equiv (1/c)$, computation can be made arbitrarily reliable. Threshold results exist for both classical and quantum computation.

Gottesman has shown that a threshold exists for systems with local gates in two dimensions [20]. The major requirement is that ancillary qubits be placed near data qubits. As levels of concatenation increase, higher level ancilla will be further and further from their respective data qubits. The rate at which the added communication introduces error must not overwhelm the correction rate.

The implications of the above restrictions to ion-traps are that ancilla must be re-initialized close to the point of use to avoid long shuttling and must be near the data. Also, error correction must be frequent enough to compensate for communication costs and it must occur on all lower levels simultaneously. Since the ions we are simulating are contained in fairly large traps, at least 10 microns apart, we assume that single ion addressing is possible and thus logical operations can be applied at any time on any ion. Simultaneous lower error correction is a built-in feature of our simulator. The most important and general implication is that geometric constraints cannot be ignored when designing a large-scale ion-trap computer if fault tolerance is to be achieved.

IV. SIMULATING LARGE-SCALE QUANTUM SYSTEMS

Having reviewed ion traps, quantum error correction, and fault tolerance, we review a method to simulate large-scale systems by mentioning the specific examples of the error-correction layouts and networks we plan to study. This sec-

tion first describes a method for simulating a special class of quantum *stabilizer* circuits in polynomial time on a classical machine. The stabilizer method described and developed by Gottesman and others [17, 18] is based on the Gottesman-Knill theorem and consists of the evolution of stabilizer generators. This is followed by a discussion of how circuits are mapped to QCCD ion trap arrays.

A. Simulating Quantum Circuits via the Heisenberg Representation

The challenge of engineering classical computer systems is met with the creation of sophisticated software tools for computer-aided design (CAD). The approach can also be applied to the quantum problem. Unfortunately for the same reason that quantum computers can solve problems that are thought to be hard, quantum circuits are hard to simulate. In general, such a simulation requires resources that grow exponentially in the number of qubits. A fact particularly crucial in our simulator where we study concatenated quantum codes, which involve an exponential increase in the number of qubits with the level of recursion.

One such model is related to the Gottesman-Knill theorem [18] which states that any quantum circuit performing only the Clifford group operators and measurements of Pauli group operators can be simulated in classical polynomial time. Such circuits are called stabilizer circuits and are not universal, but are sufficient to simulate encoding, decoding, syndrome extraction, and error correction for stabilizer codes.

A state $|\psi\rangle$ is stabilized by an operator when the state of the system $|\psi\rangle$ remains unchanged after the action of the operator. Each n qubit quantum state $|\psi\rangle$ can be represented uniquely by a group of stabilizers that can be generated by an n element generating subset of that group. As the system evolves, we only need to keep track of the change in the n generators, which comes at a cost linear in n rather than 2^n if we were to keep track of the wavefunction using probability amplitudes of $|\psi\rangle$ after each gate. The following simple example demonstrates the main idea behind the stabilizer formalism.

If U is a Clifford group operation such as the Hadamard gate H and the operator Z stabilizes a single qubit state $|\psi\rangle$ such that $Z|\psi\rangle = |\psi\rangle$, then after H is applied we have:

$$H|\psi\rangle = HZ|\psi\rangle = HZH^\dagger H|\psi\rangle = XH|\psi\rangle \quad (4.1)$$

Thus the new state of the system is stabilized by X . Similarly we can deduce a set of transformations for all Clifford group operations. For a detailed description of the stabilizer formalism the interested reader can refer to chapter 10 of [25].

We simulate stabilizer code networks under local uncorrelated noise. To model this noise, compose tensor products of single qubit positive operators

$$\mathcal{E}(\rho) = (1 - p_x - p_y - p_z)\rho + p_x X\rho X + p_y Y\rho Y + p_z Z\rho Z \quad (4.2)$$

with probability parameters p_x , p_y , and p_z . For the remainder of this paper, assume these parameters are constants – i.e. every qubit experiences the same noise process for all time.

B. Mapping Networks to Layouts

The notion of recursive error correction described earlier must be mapped to the QCCD ion trap architecture. The gate network is clearly self-similar, and we mentioned earlier that ancilla need to be stored close to their point of use. Both of these observations suggest that the two-dimensional layout should take the form of some kind of space filling fractal curve. We borrow the notion of an H-tree from modern computer architecture.

An H-tree is a recursive structure that is constructed by iteratively scaling and rotating an H-like shape. H-trees have been widely used in classical fault-tolerant studies, particularly in VLSI layout design [26] and binary tree layouts [6], so they may prove to be a natural structure to apply to our problem.

In fact, H-trees are even more natural than they might first seem. Each leaf of the tree may hold one computation and one corresponding sympathetic ion. The sympathetic ion never needs to move, and the associated computational ion can return to this “home” location frequently throughout the computation.

Consider now the specific case of the 3 qubit code. The first level of this code stores one logical qubit in 3 physical qubits (Figure 3). These three data qubits can be stored in separate leaves. The opposing leaves hold three ancillary qubits, where two are used to measure the error syndrome from the data qubits. The third one is additional and serves no purpose but to provide symmetry. Error-correction involves initializing the ancilla, interacting them with the (stationary) data, then measuring the ancilla to obtain an error syndrome.

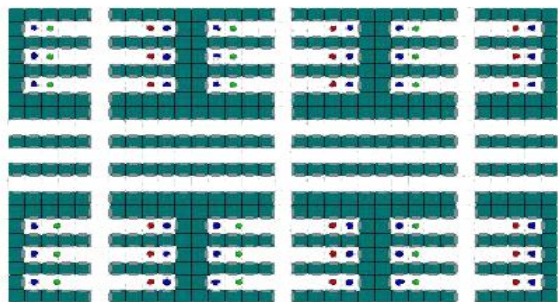


FIG. 3: H-tree layout for the 3-bit code. There are 6 blocks at level 1 encoding. The top 3 blocks represent the logical data qubits and the bottom 3 are the logical ancillae. Within each level 1 we also have 3 data and 3 ancilla qubits. Next to each single physical ion-qubit is a sympathetic ion used to cool the corresponding ion-qubit.

At the next level (i.e. the first level of recursion), one large logical qubit is encoded into three smaller logical qubits, which are each in turn encoded as three physical qubits. Here the benefits of the H-tree structure are greatest. The outer H contains three logical data qubits and their associated logical ancilla qubits. Each of these logical qubits resides in an inner H that contains three physical qubits and their associated physical ancilla. Error-correction first occurs in parallel in each of the inner H’s. After this massively parallel step, error-correction proceeds in the outer H using *exactly*

the same movement and interaction patterns as in the inner H's. In principle the H-tree layout can be used with codes concatenated to arbitrary depth.

Our simulator presents fault tolerant results by simulating the semi-classical 3-bit error correction code and the quantum Steane $[[7, 1, 3]]$ code [29] using the H-tree layout described above. These are only two of the many possible error correction codes available for simulation. The basic idea behind each code is outlined with the following few steps:

- Prepare a needed number of ancilla qubits for syndrome extraction.
- Interact the ancilla qubits with the data qubits via CNOT gates which forces the parity of the data qubits into the ancilla.
- At each gate recursively error correct the lower levels.
- Measure each ancilla qubit to extract the syndrome. This avoids direct interaction with the data qubits.

The next section presents our results and an estimated value of the threshold parameter p_{th} for the ion-trap system.

V. RESULTS AND ANALYSIS

This section describes threshold results that have been obtained through the simulation methods of Section IV. The first section describes some of the essential details of the simulation procedure. The second section presents the threshold results accompanied by some theoretical expectations.

A. Procedure

Quantum circuits constructed from noisy components may fail to produce the correct result. We model the reliability of a noisy quantum circuit by a Bernoulli random variable R with parameter p_f . The failure probability $p_f(\vec{p})$ of the circuit depends on the component failure probabilities vector \vec{p} . The elements of \vec{p} are the failure probabilities of each component in the circuit. A Monte-Carlo simulation samples R given these input failure probabilities \vec{p} , producing an estimate \hat{p}_f of p_f .

Large numbers of samples are collected using multiple runs of the simulator, executed on a 32 node Beowulf cluster to give accurate estimates of p_f . The standard error σ_s of \hat{p}_f is simply

$$\sigma_s = \frac{\sigma}{\sqrt{N_s}} \quad (5.1)$$

where N_s is the number of samples taken. Given an accuracy target, the required number of samples can be calculated.

The component failure probabilities \vec{p} can be chosen arbitrarily, but as discussed in Section II, all of the parameters can be selected based on experimental results. Specifically, we take the parameters in Table I as baseline parameters \vec{p}_b . To estimate bounds on the threshold for fault-tolerance, we apply a global exponential scaling factor M transforming the component failure probabilities with respect to the baseline

parameters,

$$\vec{p} = 10^{-M} \vec{p}_b. \quad (5.2)$$

By varying M in the neighborhood of zero, the circuit failure probability $p_f(\vec{p})$ can be estimated over a range of \vec{p} close to the experimental parameters to determine a point at which recursion improves reliability.

The syndrome extraction procedure adopted by us for both the 3-qubit and the $[[7, 1, 3]]$ code is as follows. If the first syndrome measurement results in no error, that syndrome is kept; otherwise another syndrome is collected. If both of these syndromes agree, the syndrome is accepted. However, if both syndromes disagree, new syndromes continue to be collected up to some maximum number. Any time the last two syndromes agree, that syndrome is accepted. If no syndromes ever agree, no correction is applied.

B. Simulation Results

1. 3-qubit code with gate errors

By counting the number of places where an error may occur in the 3-qubit code, we determine that c from Equation 3.1 should be no less than 59 without considering errors that could have propagated from prior cycles and initialization errors. The first few terms of the network failure probability as a function of the gate failure probability p are

$$p_f(p) = 59p^2 - 531p^3 + 1852p^4 + 3260p^5 - \dots \quad (5.3)$$

for one detection and recovery cycle. The complete expression has nonzero coefficients for all orders of p up to and including p^{59} . As mentioned in Section III, only the lowest order term of p_f is relevant for the purpose of determining a threshold. For $p \ll \frac{59}{531} \approx 10\%$, the $59p^2$ term dominates and all the approximations in Section III are valid. The threshold $c = 59$ corresponds to $p_{th} = 1.7\%$.

Our experimental results for the 3-qubit code are plotted on Figure 4, where it can be seen that recursion does indeed improve reliability when gates fail with probability p_0 less than approximately 2% and all other sources of error are set to zero. The experimental percentage of p_0 is very close to the estimated theoretical value of 1.7%.

From the plot and a calculated 95% confidence interval we can compute the value for the constant c as defined in Section III to be:

$$c \approx \frac{(422 \pm 114)}{20} = 21 \pm 6$$

which corresponds to $p_{th} = 0.05 \pm 0.02$, a slight over estimate. However, the intersection point of the $k = 1$ and $k = 2$ curves gives an actual $p_{th} \leq 0.018$ in excellent agreement with the theoretical result. If we add a level of encoding and choose $p_0 < p_{th}$, then there will be improvement in p_f for the system. This is exactly what Figure 4 shows for $k = 3$.

Plotting the logarithm of the data (Figure 5) clarifies the exponents of the dominant terms. These plots show that the

k	N	M	MEASUREMENT		MOVEMENT		BOTH	
			p_f	σ_s	p_f	σ_s	p_f	σ_s
1	20	0.00	0.363	0.006	0.610	0.004	0.633	0.007
1	20	0.25	0.134	0.004	0.230	0.003	0.244	0.006
1	20	0.50	0.042	0.002	0.068	0.003	0.067	0.004
1	20	0.75	0.014	0.002	0.019	0.002	0.021	0.002
2	20	0.00	0.53	0.01	0.993	0.002	0.994	0.001
2	20	0.25	0.071	0.004	0.51	0.01	0.529	0.009
2	20	0.50	0.006	0.001	0.053	0.003	0.051	0.004
2	20	0.75	0.0008	0.0004	0.0031	0.0006	0.003	0.001

TABLE II: 3-bit code failure probabilities for gate errors and errors from measurement, movement, and both. Following our convention, N is the number of error correction cycles, M is the scaling parameter, k is the recursion level, p_f is the network failure probability, and σ_s is the standard error of the estimate p_f .

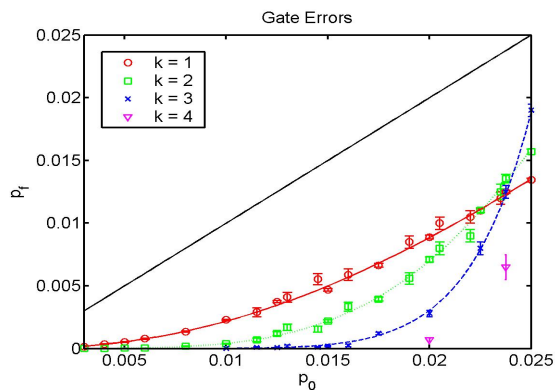


FIG. 4: p_f vs p_0 for the 3-bit code with gate errors for $k = 1, 2, 3, 4$. The solid line corresponds to $k = 1$, the dotted line to $k = 2$, and the dashed line to $k = 3$, and the two triangular points to $k = 4$. This plot shows that recursion improves reliability for gate failure rate p_0 slightly below $p_0 = 0.025$. Furthermore, each curve behaves like $p_0^{2^k}$ beneath this threshold.

dominant term of $p_f(p_0)$ is $p_0^{2^k}$ as expected from the discussion in Section III. The linear fits to these plots give us slopes of 1.99 ± 0.03 , 3.93 ± 0.08 , and 7.96 ± 0.12 as expected for levels 1, 2 and 3 respectively.

2. 3-qubit code with other error sources

Table II shows results when other sources of error are considered. The independent variable is now a scaling parameter M of the base failure probabilities shown in Table I. For example, a scaling parameter of $M = 1$ corresponds to reducing each failure probability by one order of magnitude. Note that adding a level of recursion once again improves reliability in all cases when M is sufficiently large.

As our results show, the current failure rates are not enough to achieve reliable quantum computation. However, computations using the 3 qubit bit flip (or phase flip) code can be made reliable with parameters very close to current experimental pa-

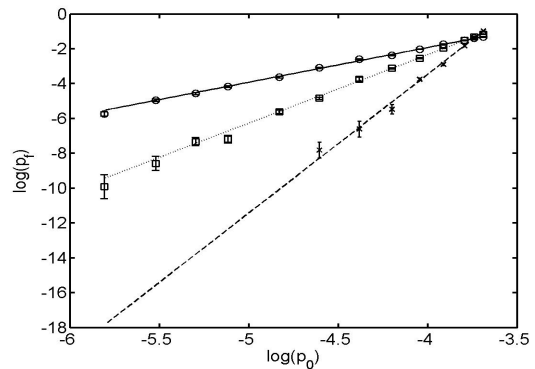


FIG. 5: $\log(p_f)$ vs $\log(p_0)$ for the 3-bit code with gate errors for $k = 1, 2, 3$. The solid line corresponds to $k = 1$, the dotted line to $k = 2$, and the dashed line to $k = 3$. The slopes of these lines are approximately 2, 4, and 8 from top to bottom, demonstrating the $p_0^{2^k}$ behavior of the dominant term.

rameters. Considering polynomial and/or exponential fits to the data, threshold scaling parameter estimates can be made based on the intersection point of the $k = 1$ and $k = 2$ curves:

$$\text{Measurement and Gate Error: } M \geq 0.077$$

$$\text{Movement and Gate Error: } M \geq 0.47$$

$$\text{Measurement, Movement, and Gate Error: } M \geq 0.48$$

Referring back to the baseline parameters, $M = 0.48$ corresponds to failure probability of 3.3×10^{-5} , 9.9×10^{-3} , 3.3×10^{-3} , and 1.7×10^{-3} per μm for a single qubit gate, two-qubit gate, measurement, and movement respectively.

Movement error is qualitatively different from gate and measurement failure and it is the dominant mode of failure. The failure probability of a single movement action at level k scales like $d^{\alpha k}$ for constants d and α . For k larger than some critical value, the reliability will begin to decrease with k because of this scaling.

The exponential scaling of movement failure with recursion level can be overcome by executing error correction every time qubits move a constant distance. Implementing this

requires a slightly different layout to accommodate intermediate error correction. In addition, ancilla will likely have to move with their associated data so that error correction can be performed recursively at each stopping point. Otherwise the area requirements of the intermediate error correction blocks would contribute too greatly to the total movement distances. Further discussion of intermediate error correction will be postponed until Section VI.

3. Some $[[7, 1, 3]]$ code results

Our results for the Steane $[[7, 1, 3]]$ code are an ongoing process due to the amount of resources involved in simulating a fairly large quantum code, but we have been able to notice the possibility of a region where the failure rate of level 2 encoding is better than level 1. Figure 6 displays our result. The $\log(p_f)$ vs $\log(p_0)$ plot, when fitted, gives us a slope of 2.04 ± 0.07 for level 1 and 4.02 ± 0.07 for level 2. This result hints that the $[[7, 1, 3]]$ circuit simulated with gate failure probability of 1×10^{-4} through 5×10^{-4} is fault tolerant and we expect to see the corresponding slopes for higher levels. It is important to note that in Figure 6 each data point represents 50 error correction cycles. Thus, to see that at the intersection point of the two curves $p_f = p_0 = p_{th}$, we must divide p_f by 50. Due to the recursive nature of the theory this may not be so evident between levels 1 and 2, but as higher levels of encoding are sampled the probabilities of failure should converge to the same value at the intersection region.

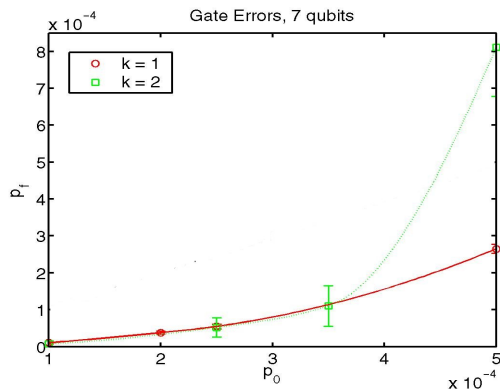


FIG. 6: p_f vs p_0 for the $[[7, 1, 3]]$ code with gate errors for $k = 1, 2$. The solid line corresponds to $k = 1$, the dotted line to $k = 2$. This plot shows that recursion improves reliability for gate failure rate p_0 slightly below $p_0 = 2.5 \times 10^{-4}$. Furthermore, each curve behaves like $p_0^{2^k}$ beneath this threshold.

VI. CONCLUSIONS AND FUTURE DIRECTIONS

A. Conclusion

We have presented an initial stage in the development of a set of tools to simulate ion-trap architectures. Given variable-

length error correction networks, this simulator estimates upper bounds on thresholds for fault-tolerance and network execution times. The preliminary results show that there exists a component failure rate for which added levels of recursion improve the reliability of the system for the classical 3-qubit code. Furthermore, the threshold behavior as a function of recursion level can be observed for previously unanalyzed architectures.

However, there is still much to be done. First, analysis of a quantum code, the Steane $[[7, 1, 3]]$ code, is not complete. A quantum code will correct against arbitrary quantum errors unlike the 3-qubit code, which can protect against either bit-flip errors or phase-flip errors. Second, we currently simulate a basic H-Tree structure and constrain movement to predetermined global paths based on the self-similarity of the H-Tree. To achieve the goal of simulating *arbitrary* large-scale quantum architectures, carefully chosen algorithms need to be applied to create a high level geometrical independence.

Our physical ion-trap model needs further improvement. The noise model we have adopted is not as rigorous as it should be. There are also sources of error with the ion-trap architecture that are just coming to light and involve some fine but important details that need to be taken into consideration if a real threshold value is to be evaluated.

B. Future Directions

1. Teleportation versus Intermediate Error Correction

The scalability of fault-tolerant architectures for ion trap quantum information processors is a central question at the heart of quantum architecture design. We have analyzed this issue through detailed simulations, which model parameters from current experiments, suggesting that for certain error regimes, scalability can indeed be achieved. More open questions remain, however, which must be addressed.

A problem with the current H-tree architecture is that movement errors increase exponentially with recursion level k , so the system failure probability p_f scales as $(p_0 d^k)^{2^k}$, where d is the distance the ions travel when $k = 1$. To overcome this, either intermediate error correction must be applied during movement with frequency rising exponentially in k , or a new architecture must be found where movement errors do not scale exponentially with the distance. Both solutions are reasonable, but the latter, accomplished using quantum teleportation, may result in a higher threshold.

2. Circuit and Movement Optimization

A large factor in the stability of the system is the efficiency of the circuits that are being simulated. In our results we have chosen the widely accepted 3-qubit and $[[7, 1, 3]]$ error correction circuits; however there are many possible error correction circuits. Some circuits that we will ultimately need to simulate require a substantially large amount of operations and would

thus demand a careful and robust compile time optimization techniques.

As mentioned earlier, a novel approach for minimizing data movement solves the problem of exponential increase in movement errors. An approach that leaves the data entirely stationary, except perhaps for local error correction could transform the movement problem into a problem of EPR pair distribution by using quantum teleportation. The goal is then to sustain a large pair current to the local region, then distill a critical bandwidth of high fidelity pairs using classical communication channels.

With the above goals in mind, we hope to find an efficient fault-tolerant architecture that may motivate the development of mid-scale quantum information processors. In fact, the 3-qubit code may be a useful example for ion-trap quantum in-

formation processors since the dominant source of errors are only phase flip errors. The ability to demonstrate thresholds for scalable fault-tolerant ion-trap architectures will further motivate the realization of larger quantum information processors.

VII. ACKNOWLEDGEMENTS

The authors would like to thank Scott Aaronson from UC Berkeley whose contribution of an optimized stabilizer algorithms was very helpful [1]. We would also like to thank Francois Impens for his insightful discussions on the topic of fault tolerance.

-
- [1] S. Aaronson and D. Gottesman. Improved simulation of stabilizer circuits - in preparation, 2004.
 - [2] D. Aharonov and M. Ben-Or. Fault tolerant computation with constant error. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, pages 176–188, 1997.
 - [3] D. Kielpinski and C. Monroe and D. J. Wineland. Architecture for a large-scale ion-trap quantum computer. *Nature*, 417:709–711, 2002.
 - [4] G. K. Brennen, I. H. Deutsch, and C. J. Williams. Quantum logic for trapped atoms via molecular hyperfine interactions. *Phys. Rev. A*, 65(022313(R)), 2002.
 - [5] G. K. Brennen, D. Song, and C. J. Williams. Quantum computer architecture using nonlocal interactions. *Phys. Rev. A*, 67(050302(R)), 2003.
 - [6] R. P. Brent and H. T. Kung. On the area of binary tree layout. *Inform. Contr.*, 65:45–52, 1982.
 - [7] J. I. Cirac and P. Zoller. Quantum computations with cold trapped ions. *Phys. Rev. Lett.*, 74:4091–4094, 1995.
 - [8] C. Monroe et. al. Scalable entanglement of trapped ions. *AIP Conf.*, 551:173–186, 2001.
 - [9] D. Copley et al. Toward a scalable, silicon-based quantum computing architecture. *To appear, Selected Topics, Journal of Quantum Electronics*, 2004.
 - [10] D. J. Wineland et al. Experimental issues in coherent quantum-state manipulation of trapped atomic ions. *Journal of Research of the National Institute of Standards and Technology*, 103:259–328, 1998.
 - [11] D. Kielpinski et. al. Recent results in trapped-ion quantum computing at nist. *Experimental Implementation of Quantum Computation (Sydney, 2001)*, 2001.
 - [12] D. Leibfried et al. Experimental demonstration of a robust, high-fidelity geometric two ion-qubit phase gate. *Nature*, 422:412–415, 2003.
 - [13] J. V. Porto et. al. Quantum information with neutral atoms as qubits. *Phil. Trans. R. Soc. Lond.*, A361:1417–1427, 2003.
 - [14] M. A. Rowe et. al. Transport of quantum states and separation of ions in a dual rf ion trap. *arXive e-print quant-ph/0205094*, 2002.
 - [15] M. Oskin et al. Building quantum wires: The long and the short of it. In *the 30th Annual International Symposium on Computer Architecture*, 2003.
 - [16] Q. A. Turchette et. al. Heating of trapped ions from the quantum ground state. *Phys. Rev. A*, 61:063418–1–8, 2000.
 - [17] D. Gottesman. A class of quantum error-correcting codes saturating the quantum hamming bound. *Phys. Rev. A*, 54:1862, 1996.
 - [18] D. Gottesman. The heisenberg representation of quantum computers. *arXive e-print quant-ph/9807006*, 1998.
 - [19] D. Gottesman. Theory of fault-tolerant quantum computation. *Phys. Rev. A*, 57(1):127–137, 1998. *arXive e-print quant-ph/9702029*.
 - [20] D. Gottesman. Fault-tolerant quantum computation with local gates. *arXive e-print quant-ph/9903099*, 1999.
 - [21] D. Kielpinski. Entanglement and decoherence in a trapped-ion quantum register. *Thesis, Univ. Colorado*, 2001.
 - [22] D. M. Meekhof, C. Monroe, B. E. King, W. M. Itano, and D. J. Wineland. Generation of nonclassical motional states of a trapped atom. *Phys. Rev. Lett.*, 76:1796, 1996.
 - [23] K. Molmer and A. Sorensen. Multiparticle entanglement of hot trapped ions. *Phys. Rev. Lett.*, 82:1835–1838, 1999.
 - [24] C. Monroe, D. M. Meekhof, B. E. King, W. M. Itano, and D. J. Wineland. Demonstration of a fundamental quantum logic gate. *Phys. Rev. Lett.*, 75:4714, 1995.
 - [25] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, UK, 2000.
 - [26] W.L. Ruzzo and L. Snyder. Minimum edge length planar embeddings of trees. In *Proceedings of the CMU Conference on VLSI Systems and Computations*, pages 119–123. Computer Science Press, October 1981. Kung, Sproul, Steele, eds.
 - [27] P. W. Shor. Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A*, 54:2493, 1995.
 - [28] P. W. Shor. Fault-tolerant quantum computation. *37'th Symposium on Foundations, Los Alamitos, CA, IEEE Computer Society Press*, pages 56–65, 1996.
 - [29] A. Steane. Simple quantum error correcting codes. *Phys. Rev. Lett.*, 77:793–797, 1996.
 - [30] A. M. Steane and D. M. Lucas. Quantum computing with trapped ions, atoms and light. *AIP Conference Proceedings*, 551(1):158–172, 2001.