Quantum Information and Computation, Vol. 7, No. 4 (2007) 297–318
© Rinton Press

NOISE THRESHOLD FOR A FAULT-TOLERANT TWO-DIMENSIONAL LATTICE ARCHITECTURE

KRYSTA M. SVORE[a]
Dept. of Computer Science, Columbia University, 1214 Amsterdam Ave., MC:0401
New York, NY 10027 USA

DAVID P. DIVINCENZO[b]
IBM Research, PO Box 218,
Yorktown Heights, NY 10570 USA

BARBARA M. TERHAL[c]
IBM Research, PO Box 218,
Yorktown Heights, NY 10570 USA

Received June 5, 2006
Revised December 1, 2006

We consider a model of quantum computation in which the set of operations is limited to nearest-neighbor interactions on a 2D lattice. We model movement of qubits with noisy `SWAP` operations. For this architecture we design a fault-tolerant coding scheme using the concatenated $[[7, 1, 3]]$ Steane code. Our scheme is potentially applicable to ion-trap and solid-state quantum technologies. We calculate a lower bound on the noise threshold for our local model using a detailed failure probability analysis. We obtain a threshold of 1.85×10^{-5} for the local setting, where memory error rates are one-tenth of the failure rates of gates, measurement, and preparation steps. For the analogous nonlocal setting, we obtain a noise threshold of 3.61×10^{-5}. Our results thus show that the additional `SWAP` operations required to move qubits in the local model affect the noise threshold only moderately.

Keywords: quantum fault tolerance, quantum architectures, quantum error correction

Communicated by: I Cirac & G Milburn

1 Introduction

In order to perform large-scale quantum computation reliably, any quantum computer technology will require error correction. Error-correcting quantum data is useful if the logical noise level is lowered by coding and error-correcting. Since error correction is noisy, a decrease in logical noise level only occurs for sufficiently low initial noise rates, below a certain *noise threshold* [1, 2, 3, 4] . It is clear that it is of great importance for the future of quantum computation to determine the values of such noise thresholds since they set the level of accuracy to be achieved by experimental programs.

[a]kmsvore@cs.columbia.edu
[b]divince@watson.ibm.com
[c]bterhal@gmail.com

Over the past 10 years estimates for noise thresholds for various *code architectures* [5, 6, 7, 8] have been determined. These estimates, depicted in the overview in Fig. 1 of [9], have varying degrees of accuracy and rigor. In Ref. [10], entirely rigorous criteria for fault-tolerant design and threshold analysis were formulated. These rigorous criteria allow, at least for small error-correcting codes, an accurate determination of the noise threshold for simple stochastic noise models.

In many threshold analyses it has been assumed that interactions between qubits can be performed in a manner independent of their spatial separation. In reality, interactions between stationary qubits are typically restricted to occur between nearest neighbors in some low-dimensional spatial layout, the *spatial architecture*. A transport mechanism will thus be required to bring qubits together. An example is the scalable ion-trap architecture where ions are shuttled between interconnected ion traps [11]. Transport mechanisms will also be required in solid-state implementations, such as the method involving electron spins in quantum dots [12], phosphorus spin qubits in silicon [13], Josephson-junction qubits [14] or neutral atoms in a 2D optical lattice [15].

Previous work [8, 16, 17] has established the existence of a noise threshold in a local model. In Ref. [8] we made a first attempt to also estimate the effect of locality on the noise threshold, in particular the dependence on a code-dependent scale parameter and the failure rates of qubit transportation. In Ref. [18], the authors derive a noise threshold of $O(10^{-7})$ for the [[7,1,3]] code for a quasi-one-dimensional architecture. Some thresholds related to ion-trap systems are also analyzed in Refs. [19, 20]. For spin-qubits, Refs. [21] and [22] estimate noise thresholds taking into account transport mechanisms in conjunction with explicit two-dimensional layouts. None of these papers employ the rigorous methods of analysis outlined in Ref. [10]. We believe the use of rigorous methods can considerably impact the threshold values that are found. We briefly argue in Section 6 why the methodology laid out in Ref. [10] gives fairly tight bounds for the fault-tolerance threshold. This implies that some of the previous threshold estimates for the [[7,1,3]] code have been too optimistic.

In this paper we consider a two-dimensional nearest-neighbor lattice architecture using logical noisy `SWAP` gates as the basic qubit transport mechanism. It is clear that any viable layout for stationary qubits has to be in a two-dimensional plane so that classical control fields can access the qubits from the third dimension. In the case of a quasi-1D or self-similar architecture (such as the H-tree), the classical controls could also lie in the two-dimensional plane itself.[d] In our analysis, we consider the simplest possible noise model, namely independent adversarial stochastic noise. In this noise model, each location ℓ independently undergoes an error with some probability γ_ℓ. The error that occurs can be chosen adversarially so that in our threshold analysis we assume that for every location the worst among the three Pauli errors `X`, `Y`, `Z` occurs.

We find, somewhat surprisingly, that the threshold noise value for our 2D lattice architecture is close to the threshold noise value for a nonlocal architecture — both are $O(10^{-5})$. We understand this small dependence on locality to arise from (1) the dominance of the noisy

[d] A purely one-dimensional fault-tolerant architecture is impossible if one uses a distance-3 code. The reason is that such an architecture necessitates swapping data qubits inside a block which may generate a two-qubit error due to one failed `SWAP`. For a distance-3 code, such errors cannot be corrected. This implies that the level-1 error rate can never be smaller than the base error rate. A purely one-dimensional fault-tolerant architecture is not excluded for higher distance codes, but the threshold noise value may be very poor.

two-qubit gates and (2) the fact that we have optimized the circuits to be space-efficient in a dense, 2D square lattice, with a surprisingly small cell size of 6×8 for the [[7,1,3]] code, resulting in a small amount of moving.

Given the good performance of our `SWAP`-based architecture, we believe that it is unnecessary to seek alternative means of transportation (teleportation and entanglement purification for example), at least for the [[7,1,3]] code in this 2D layout. As the reader will observe, the noisy `SWAP` gates simply model any physical scheme in which qubits are transported through some quantum channel. In our model, this channel will have a fixed time-delay and a noise level that depends on the choice of noise level for the `SWAP`s. It will also be clear from our analysis that no really long-distance means of transportation is needed for fault-tolerant quantum computation.

1.1 A Local 2D Lattice Architecture

Let us assume that we have modified an unencoded circuit M so that interactions among the qubits are nearest-neighbor on a 2D lattice. This requires adding `SWAP` gates in the unencoded circuit, which can be implemented using three controlled-NOT (`CNOT`) gates. We will also be using `SWAP` gates in our encoded circuit, but we will treat these gates as elementary (and not composite) gates.

After one level of encoding using Steane's [[7,1,3]] code, every qubit in M is replaced by a 6×8 cell of qubits, 48 qubits in total, as depicted in Fig. 1. In a concatenated architecture, the next level of encoding is obtained by replacing each qubit in the cell again by a 6×8 cell, and so on. In this way, two-qubit `CNOT` gates between nearest-neighbor qubits in the cell become (transversal) `CNOT` gates between two neighboring cells. The cell functions as (1) the spatial unit in which error correction takes place, (2) the 'resting place' for a block of data qubits, and (3) the space to create a special type of ancillary encoded state $|\overline{a}_\theta\rangle$ (see Sections 3.2 and 5.3) that is used in the computation.

For error correction, the 48 qubits of the cell are divided into three sets, see Fig. 1. There are seven physical data qubits labeled by d in the figure. In the snapshot, the data qubits sit at a special location in the cell that we refer to as *home base*. When the data qubits interact with ancilla qubits for syndrome extraction, the data qubits will always be sitting at home base. The cell also contains at most ten ancilla qubits at a time. Three of these ancilla qubits are verification qubits, labeled by v in the figure. The other seven are the proper ancilla qubits (labeled by a) that will receive the error syndrome. The rest of the cell is filled by dummy qubits (labeled by O). The state of the dummy qubits is irrelevant for the computation; they are only used for swapping with data or ancilla qubits. One can view the dummy qubits as forming channels through which the other qubits can pass. Since ancilla qubits remain local to the cell and only data qubits interact with data qubits in neighboring cells, we place data qubits on the exterior of the cell and devote the interior region of the lattice cell to ancilla preparation and verification for error correction.

Interactions between data qubits in neighboring cells can occur between two lattice cells situated on a horizontal axis, i.e., data qubits must move horizontally from home base to interact, or on a vertical axis, i.e., data qubits must move vertically from home base to interact. We include both a vertical and a horizontal transportation channel, consisting of dummy qubits, to be used for horizontal and vertical transportation of data qubits. The

channels occupy the first row and first column of the lattice cell and they can be used by qubits in neighboring cells as well (see the discussion in Section 3.3).

We assume `SWAP` to be our mechanism for qubit transport, where a qubit moves by swapping with an adjacent qubit. To guarantee the fault tolerance of each `SWAP` operation we only *swap a data or ancilla qubit with a dummy qubit*. Thus, in the layout shown in Fig. 1, we alternate dummy qubits with data or ancilla qubits to allow for proper fault-tolerant movement.

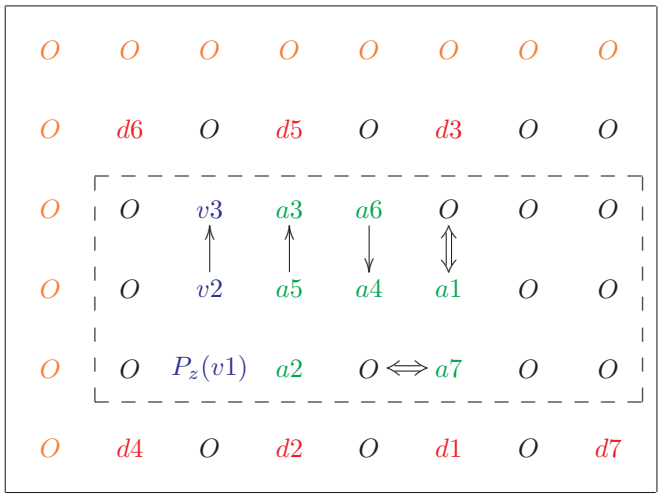


Fig. 1. Snapshot of the operations and state of a 6×8 lattice cell during the ancilla preparation part of error correction. Ancilla qubits are labeled with a and data qubits are labeled with d. Ancilla qubits labeled with v are used for verification. O locations represent dummy qubits that are used for swapping operations. The inner dashed region is the area used for ancilla preparation and verification. The vertical and horizontal dummy qubit border regions represent channels (labeled as O) used for qubit transportation. Single arrows represent `CNOT` gates, double arrows represent `SWAP` operations, and P_z represents preparation of a $|0\rangle$ state.

2 Review of Definitions and Analysis Method for the [[7,1,3]] Code

We now review some definitions and properties of the standard fault-tolerant analysis for the [[7,1,3]] code; this has been described in detail in Ref. [10]. For the [[7,1,3]] code, the Clifford-group gates (two-qubit `CNOT`, single-qubit Hadamard (`H`), and single-qubit phase gate (`S`), i.e., $|0\rangle\langle 0| + i\,|1\rangle\langle 1|$) are transversal. In order to make the gate set universal, one needs a non-Clifford-group gate; we choose the single-qubit gate $\mathtt{T} = e^{-i\pi \mathtt{Z}/8}$. We believe that this choice for a non-Clifford-group gate is better than trying to implement, by local means, a three-qubit Toffoli gate.

For reasons discussed later, we choose to implement the gate `T`, as well as that gate `S`, as in Fig. 2. Note that $\mathtt{T} = R_{\mathtt{Z}}(\pi/4)$ and $\mathtt{S} = R_{\mathtt{Z}}(\pi/2)$, where $R_Z(\theta) = e^{-i\frac{\theta}{2}\mathtt{Z}}$. In these circuits, the state $|a_\theta\rangle = R_{\mathtt{Z}}(\theta)\,|+\rangle$. The corrective operation $R_{\mathtt{Z}}(2\theta)$ for the `T` gate is the `S` gate and the corrective operation for the `S` gate is `Z`. We emphasize that with this construction, the only gates in the *unencoded* circuit are `X`, `Y`, `Z`, `H`, and `CNOT` (in addition to state preparations and measurements).

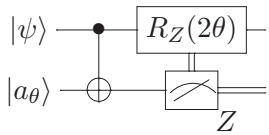


Fig. 2. Circuit for implementing the single-qubit gate $R_Z(\theta)$. If the measurement result is $|1\rangle$, then we correct the state by applying $R_Z(2\theta)$. The output state is $R_Z(\theta)\,|\psi\rangle$.

Locations in a quantum circuit are defined to be gates, single-qubit state preparations, measurement steps, or memory (wait) locations; they can all be executed in a single timestep. After one level of encoding, every location (denoted as 0-Ga) is mapped onto a 1-rectangle (1-Rec), a space-time region in the encoded circuit, which consists of the encoded gate (1-Ga) followed by error correction (1-EC), as shown in Fig. 3. For transversal gates, the 1-Ga consists of performing the 0-Gas on each qubit in the block(s). The encoded version of a preparation location is called a 1-Prep. For the [[7,1,3]] code, it consists of at most two attempts at preparing the desired encoded state; each attempt is followed by a verification step. If the verification of the first preparation fails, we repeat the preparation and verification. If it fails again, we continue with the failed encoded state which may contain two errors. To make a 1-Rec, we follow the 1-Prep by a 1-EC. The encoded version of a measurement location is called a 1-Meas. In the $[[7,1,3]]$ code, it is implemented as seven parallel measurements. In our model, we consider both measurement in the `X` basis and measurement in the `Z` basis to be single locations (rather than converting `X` measurement into `Z` measurement using a `H` gate). For the measurement 1-Rec, the 1-EC is simply the classical processing of the measurement outcomes.

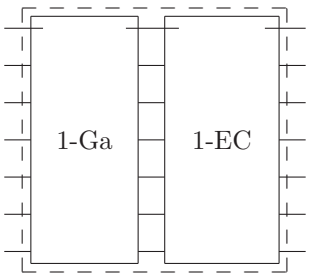


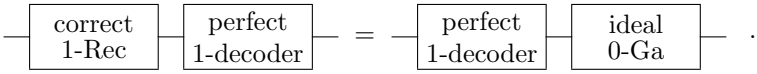
Fig. 3. A 1-rectangle (1-Rec), indicated by a dashed box, that replaces a single-qubit 0-Ga location. The 1-Rec consists of the encoded fault-tolerant implementation of the 0-Ga (1-Ga) followed by an error-correction procedure (1-EC).

For the fault-tolerance analysis, one also defines an extended 1-Rec (1-exRec) which consists of a 1-Rec with its preceding 1-EC(s) on the input block(s). The 1-exRec for a preparation or measurement location coincides with the 1-Rec for preparation or measurement.

In order to be fault-tolerant, the 1-EC, 1-Ga, and 1-Prep must satisfy the fault-tolerance requirements listed in Ref. [10] and reproduced in Appendix 1. Our constructions obey these requirements with the caveat that the 1-EC does not satisfy the second property. As we will show, it can leave an `X` error and a `Z` error on the data after its application. Thus for fault tolerance, it is necessary that the 1-Gas never transform a `X` or `Z` error into a `Y` error since this

would correspond to having two `X` (or `Z`) errors in a block. This is achieved by only having 0-Gas which are `X`, `Y`, `Z`, `H`, and `CNOT` (and not `S`).

If the fault-tolerance requirements are fulfilled, it can be shown that if a 1-exRec is good — for a distance-3 code, such as the [[7,1,3]] code, this means that the 1-exRec *contains at most one fault*— then the dynamics induced by the 1-Rec contained in the 1-exRec is correct [10], as defined in this picture:



A 1-exRec containing two faults can be bad, but not all higher-order faults lead to two errors on the data. In order to give an accurate estimate of the threshold, Ref. [10] introduced the notion of benign and malignant fault patterns. A set of locations is *benign* in a 1-exRec if the 1-Rec contained in the 1-exRec is correct. Otherwise, the set of locations is called malignant. With this definition one can redefine a bad or *failed* 1-exRec, namely, as a 1-exRec that contains faults at malignant locations. In our analysis, we assume any pattern of three faults or more is always malignant. We discuss this assumption in Section 4.1 since it has a small but non-zero effect on the threshold estimate.

We denote the failure probability of a location of type ℓ as $\gamma_\ell \equiv \gamma_\ell^0$. The failure probability at encoding level r for location ℓ can be computed with a multi-dimensional map $\mathbf{F}$, with $\mathbf{F}_\ell(\gamma^{r-1}) = \gamma_\ell^r$. The map $\mathbf{F}$ is independent of concatenation level and is determined by the number of malignant and higher-order errors in the 1-exRec for the different locations. An initial vector of probabilities γ^0 is said to be below the threshold if there is a concatenation level r_ϵ such that

$$\forall \epsilon, \exists r_\epsilon, \forall \ell, \;\; \gamma_\ell^{r_\epsilon} = \left(\mathbf{F}^{r_\epsilon}(\gamma^0)\right)_\ell \leq \epsilon. \tag{1}$$

In Sections 4 and 5, we describe how we calculate this map and the threshold results that we obtain for the 2D local and the nonlocal model. We now describe the local fault-tolerant circuits that we use in our analysis.

3 Local Fault-tolerant Circuitry

Before we describe our error-correction circuitry, we first describe local 1-exRecs. In a local architecture, a 1-Ga which replaces a two-qubit 0-Ga includes the necessary `SWAP` operations to move the two encoded qubits together. At the next level of concatenation, each new location, including a `SWAP`, is replaced by its 1-Rec. Our `SWAP` gates involve one dummy qubit and one data qubit; it is unnecessary to do error correction on the dummy qubit. Therefore, the `SWAP` 1-Rec effectively behaves as a 1-Rec for a single-qubit gate since it requires a 1-EC on only one block of qubits.

3.1 Local Steane Error Correction (1-EC)

For $[[7,1,3]]$ error correction, we have a choice between (1) Shor-type error correction, where the syndrome for each stabilizer operator is measured separately, (2) Steane-type error correction, where the entire `X`- or `Z`-error syndrome is copied onto a verified ancilla block, or (3) Knill-type error correction, where the data qubits are teleported into a new block and `X`- and `Z`-error-syndrome information is obtained via a Bell measurement. For our local layout, it is important to minimize the use of space and thus movement, potentially at the cost

of increased waiting and memory errors. The reason is that we assume memory errors will typically occur at lower rates than `SWAP` errors. This optimization would seem to point to a sequential execution of the most space-efficient error correction, which is Shor's. However, in Shor-type error correction, the number of gates between data and ancilla qubits is more than in Steane-type error correction, which can negatively affect the threshold.

Therefore, we opt for a variant of Steane's error correction, schematically drawn in Fig. 4. We use a *deterministic* error-correction network, in which we sequentially prepare three ancilla blocks in the encoded 0-state $|\bar{0}\rangle$. If a single failure causes one ancilla block to fail verification, then two ancilla blocks remain, one each for `X`- and `Z`-error correction. If two ancilla blocks fail verification, which by construction can only happen when two faults occur, we cannot perform full error correction and we count that as a failure of the data in this rectangle. If the first two ancilla blocks pass verification, then there is no need to prepare a third ancilla block. Instead, the data block waits for the same number of timesteps that would have been required for the third ancilla block preparation.

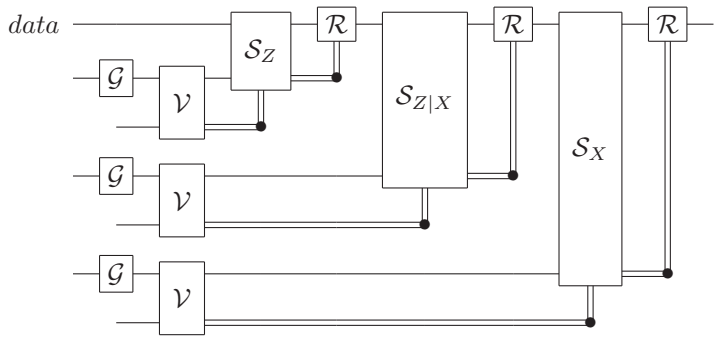


Fig. 4. The error-correction network (1-EC) based on Steane's $[[7,1,3]]$ code, performing first `Z`-error correction, and then `X`-error correction. Error correction consists of ancilla preparation ($\mathcal{G}$), ancilla verification ($\mathcal{V}$), syndrome extraction ($\mathcal{S}$), and a recovery operation ($\mathcal{R}$). In many circumstances, the application of the corrective gate $\mathcal{R}$ can be postponed and is carried forward classically as a change of 'Pauli frame' [7].

The local versions of $\mathcal{G}$ and $\mathcal{V}$ are shown in Fig. 5. Our ancilla verification scheme is less extensive than the standard CSS method described, for example, in Ref. [10]. In the standard method, the ancilla is verified using another block in the state $|\bar{0}\rangle$. For the fault tolerance of our error-correction network, we only need to measure the single stabilizer `IZIZIZI` on the ancilla block $|\bar{0}\rangle$.[e] The ancilla needs to be verified since the encoding uses two-qubit gates which can put two errors on the block. If these errors are `X` errors they can propagate to the data. However, not every possible pair of `X` errors will be generated by the encoder circuit. In fact, we only need to check for errors on the second, fourth, and sixth qubits of the ancilla block. Our verification method prevents an ancilla from passing with the errors $\mathtt{X}_2\mathtt{X}_7$, $\mathtt{X}_3\mathtt{X}_6$, $\mathtt{X}_4\mathtt{X}_5$, each of which can be caused by a single faulty location in the preparation circuit. An ancilla with single `X` and/or single `Z` errors can still pass verification. To parallelize the necessary measurement, verification is performed by preparing a 3-qubit cat state $(|000\rangle + |111\rangle)/\sqrt{2}$

[e]Thanks to Panos Aliferis for suggesting this simplification.

and applying controlled-Z (CZ) gates between the three verification qubits and the second, fourth, and sixth qubits of the ancilla block. Each verification qubit is then measured in the X basis. If the parity of the measurement results is 1, the ancilla block fails verification. If an ancilla block passes verification, it is used for syndrome extraction ($\mathcal{S}$), shown in Figure 6. During ancilla preparation the data qubits are moved to home base in order to interact with the prepared ancilla.

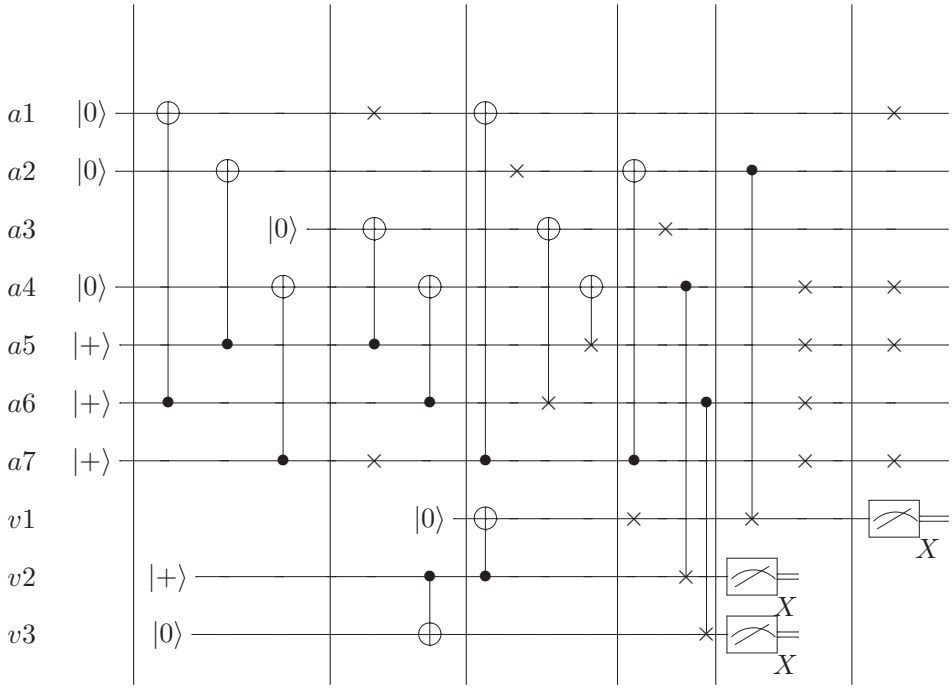


Fig. 5. The local preparation and verification networks $\mathcal{G}$ and $\mathcal{V}$. The lower three qubits are the verification qubits of network $\mathcal{V}$. The seven timesteps in which this circuit is executed are delineated by the long vertical lines. The third timestep is shown in Fig. 1. The single-qubit $\times$ symbol indicates that in this timestep, the qubit has been swapped with a dummy qubit. The CNOT symbol with the control indicated with a $\times$ means a CNOT followed by SWAP on the same qubits. The two-qubit gate indicated by a solid circle on one end and a $\times$ on the other is a CZ followed by SWAP. It is important to note that the qubit labels on the left correspond to the associated wires throughout the circuit (and thus are never swapped) and thus qubit a6, for example, interacts with qubit v3 in the 5th timestep. Thus the fourth qubit travels the farthest, being swapped four spaces along the square lattice during the operation of this circuit.

3.2 Local Preparations (1-Preps)

The local 1-Prep for $|\bar{0}\rangle$ consists of at most two attempts to create a verified ancilla as in the local network in Fig. 5. We obtain the 1-Prep for $|\bar{+}\rangle$ by applying transversal Hadamards on the passed ancilla. For universality, we also need 1-Preps for the state $|\bar{a}_\theta\rangle$, for $\theta = \pi/2, \pi/4$ (see Fig. 2). We can produce such states as follows (see Ref. [10]: The unencoded state $\left|a_{\pi/2}\right\rangle$ is correctly generated by measuring the operator $\mathtt{SXS}^\dagger$ on the state $|0\rangle$, since $\mathtt{SXS}^\dagger \left|a_{\pi/2}\right\rangle =$

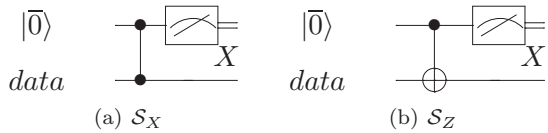


Fig. 6. The network $\mathcal{S}$ for syndrome extraction during (a) X-error correction and (b) Z-error correction. During syndrome extraction the data qubits sit at home base in the cell. The two-qubit gate with solid circles on both ends is the CZ gate.

$\left|a_{\pi/2}\right\rangle$. If we find eigenvalue $+1$, we are done, otherwise we apply Z. Similarly, for $\left|a_{\pi/4}\right\rangle$, we measure $\mathtt{TXT}^\dagger$ and correct with Z if eigenvalue -1 is found. As we will see later, $|\overline{a}_\theta\rangle$ preparations are fundamentally different from Clifford-group preparations, in that it is not necessary to make them fault tolerant at every level of concatenation.

3.3 Local Rectangles and Their Use of Space and Time

Table 1. Types and number of locations contained in a local 1-EC and two local 1-Gas. The first numbers for the 1-EC are based on the assumption that the first and second ancillas pass verification; the numbers in parentheses result from the first or second ancilla failing verification. The last column lists the number of locations in a horizontal CNOT 1-Ga.

ℓ	Description	# in 1-EC	# in h-CNOT 1-Ga
0	horizontal CNOT (h-CNOT)	2 (3)	0
1	vertical CNOT (v-CNOT)	30 (38)	7
2	horizontal CZ (h-CZ)	0	0
3	vertical CZ (v-CZ)	14 (14)	0
4	h-CNOT followed by SWAP	0	0
5	v-CNOT followed by SWAP	0	0
6	h-CZ followed by SWAP	6 (9)	0
7	v-CZ followed by SWAP	0	0
8	H	0	0
9	horizontal SWAP (h-SWAP)	22 (33)	112
10	vertical SWAP (v-SWAP)	8 (12)	14
11	$\|+\rangle$-preparation	8 (12)	0
12	$\|0\rangle$-preparation	12 (18)	0
13	X-measurement	34 (37)	0
14	Z-measurement	0	0
15	wait during gate/prep	18 (27)	196
16	wait during measurement	0	14

In Table 1, we list the universal set of location types considered in our scheme. Preparation of $|+\rangle$, measurement in the X basis, the CZ gate, and CNOT and CZ combined with SWAP are considered single locations which can be implemented in one fundamental timestep. We will assume that these composite gates have error probabilities similar to the other elementary gates. In practical implementations, some gates may have to be constructed from multiple gates and may thus have higher error rates. We assume that the time it takes to do a measurement is the same as the single-qubit gate time and that classical post-processing does not take any additional time. Note in Table 1 that we use two different types of wait locations, one that happens in parallel with a gate and one that happens in parallel with a measurement. The reason for having separate locations is that the encoded version of the wait location that

occurs in parallel with a gate needs many additional wait locations in order to be synchronized with the encoded version of a gate.

We include both horizontal and vertical two-qubit gate locations. Two-qubit gates occurring between two logical qubits x and y on a horizontal axis use the horizontal channel internal to lattice cell x or lattice cell y as well as the horizontal channel internal to the lattice cell belonging to the neighboring qubit directly south of lattice cell x or lattice cell y (i.e., directly below it). We will actually use *either* the channel south of lattice cell x *or* the channel south of lattice cell y, but not both. For example, in Fig. 1(a), data qubits $d6$, $d5$, and $d3$ move into the channel internal to their lattice cell, while qubits $d4$, $d2$, $d1$, and $d7$ move into the neighboring channel directly south. To perform a two-qubit gate, the neighboring horizontal qubits then move across the lattice cell by swapping (for example, along row 2 and 6 in the figure) while the qubits that moved into the channels remain stationary and wait. Vertical two-qubit gates only use the vertical channel internal to the lattice cells of the two qubits.

It follows that two vertical two-qubit gates on two adjacent pairs of cells can be executed in parallel since there is no common use of channels. For two horizontal two-qubit gates on pairs of cells that are above each other we need some scheduling since a common channel is used. This scheduling is easy to achieve, since we can designate which of the two qubits involved in the two-qubit gates will move into the channel, and which will remain within the cell. The qubits remaining within the cell can swap across the internal row freely, while the qubits moving into the channel remain stationary and wait until the data qubits moving along the interior of the lattice cell arrive. For a more detailed explanation, a complete snapshot of the movement just described is available at

http://www.research.ibm.com/quantuminfo/svore/movies.html .

In the local model, we must clearly define to what rectangle some of the `SWAP` operations and additional wait locations belong since this can be ambiguous. During ancilla preparation in the 1-EC, the data qubits move around in the cell and reside at home base once syndrome extraction begins. This movement and possibly waiting of data qubits is included in the definition of the previous 1-Ga, since it can be viewed as movement back to home base after the 1-Ga was performed. Similarly, the 1-EC does not contain any `SWAP` or wait locations on the *data* once the last syndrome extraction (and possibly error correction) has been performed. Thus the 1-Ga includes all data locations between the last time the data was in home base for syndrome extraction and the next time the data is in home base for syndrome extraction. The 1-EC includes all locations on the data between the first and last syndrome extraction, including these syndrome extractions (and possibly error correction) themselves.

With these definitions, we list the number of locations in a few essential routines in Table 1. Note that the total number of `SWAP` gates in the h-`CNOT` 1-Ga is $112 + 14 = 126$ which is substantial. As we will see in Section 5 the effect on the threshold of these additional gates is quite small.

In Table 2, we list the number of timesteps and locations in a 1-Rec and 1-exRec for different locations ℓ. The local 1-EC takes 27 timesteps; this may be compared to the same 1-EC without `SWAP`, i.e., the nonlocal 1-EC, which takes 21 timesteps. Note that in Table 2 the difference between the number of timesteps in a 1-Rec and a 1-exRec is 26, which is one less than the number of timesteps in a 1-EC. This is because there is an overlap of one timestep between the leading 1-EC and the 1-Rec.

Of all gates the horizontal `CNOT` 1-Rec takes the longest, a total of 35 timesteps (compared to 22 timesteps in the similar nonlocal model). The single-qubit 1-Rec takes 28 timesteps (compared to 22 timesteps in the nonlocal model). For synchronization, we have added wait locations to the 1-Recs that take fewer than 35 timesteps, so that the time for all 'gate' 1-Recs is 35 timesteps. Notice that the preparation 1-Recs take more time, but these 1-Recs can be started at the appropriate time in advance so that the prepared states will be ready when necessary.

Since it is hard to represent what happens in three-dimensional space-time on a two-dimensional sheet of paper, we only partially represent in this paper the local fault-tolerant circuits that we have developed. Supplementary material in the form of complete sequences of time snapshots of the 2D layouts can be found at

http://www.research.ibm.com/quantuminfo/svore/movies.html.

Table 2. The number of timesteps in the 1-Rec and the 1-exRec for different locations ℓ.

ℓ	Time of 1-Rec	Time of 1-exRec	Total # loc. in 1-exRec
[0–7]	35	61	1225
[8–10]	35	61	616
[11–12]	41	41	469
[13–14]	1	27	196
[15]	35	61	616
[16]	28	54	378

4 Methodology

Our threshold analysis relies on counting the number of malignant fault pairs in a 1-exRec; thus it is important to understand this notion. In principle, we have to assume an arbitrary input to the 1-exRec in order to analyze malignancy of faults. Given the fact that the [[7,1,3]] code is a perfect code, incoming errors can be modeled by letting the input block have at most 1 `X` and 1 `Z` error. Now one considers two Pauli faults in the 1-exRec. If both faults occur in a single leading 1-EC, the output of that 1-EC can again be viewed as some codeword plus at most one `X` and `Z` error. This implies that the subsequent 1-Rec is correct, and thus the fault-pattern is benign. If the 1-exRec is a `CNOT` 1-exRec, both leading 1-ECs can have faults and the transversal `CNOT` gate could spread these so that the 1-Rec may not be correct.

Let us consider how the possible input errors affect the output of a 1-EC with at most one fault. If the 1-EC has no fault, the incoming errors will be corrected and the output is a codeword, just as if there were no incoming errors. If the 1-EC has a single fault, it is guaranteed that one of the incoming errors (`X` or `Z`) will be corrected. As was argued in Ref. [10], the overall effect of the other incoming error and the fault in the 1-EC is a possible logical operation in the code space plus an error whose position and character only depends on the fault in the 1-EC. In other words, the errors at the end of the leading 1-EC with at most one fault do not depend on the pattern of incoming errors. This property still holds with our modified 1-EC where the 1-EC can create one `Z` and one `X` error in the outgoing data block. This implies that for determining malignancy, we only need to place two faults inside the 1-exRec and not consider incoming errors. Since we are using an adversarial noise model, we consider a pair of locations malignant if it is malignant for some Pauli errors occurring at those locations.

4.1 Calculation of Failure Probabilities

We calculate the failure probability $\gamma_\ell^r = \mathbf{F}_\ell(\gamma^{r-1})$ of a type-ℓ 1-exRec. Note that $\mathbf{F}$ is level-independent and we drop the superscript r in what follows. We can bound the failure probability γ_ℓ as

$$\begin{aligned} P(\text{failure of a 1-exRec}) &= P(\text{1-Rec is incorrect}) \\ &\leq P(1^+ \text{ trailing 1-ECs do not occur}) \\ &\quad + P(\text{1-Rec is incorrect \& trailing 1-ECs occur}). \end{aligned} \tag{2}$$

Remember that a 1-EC does not occur when less than two ancilla blocks pass verification. Consider P(1-Rec is incorrect & trailing 1-ECs occur). Because of the fault tolerance of the quantum circuits, the 1-Rec is only incorrect when at least two malignant faults occur in the 1-exRec. We can upper-bound this probability as

$$\begin{aligned} P(\text{1-Rec is incorrect \& trailing 1-ECs occur}) &\leq P(\text{1-Rec incor. due to mal. pair} \\ &\quad \text{\& trailing 1-ECs occur}) \\ &\quad + P(3^+ \text{ faults \& trailing 1-ECs occur}) \end{aligned} \tag{3}$$

Consider the failure probability due to malignant pairs. In this probability it is not specified whether the leading 1-ECs in the 1-exRec occurred or not. But the only way for a leading 1-EC to fail is when there are two faults in it. As argued earlier in this section, for the [[7,1,3]] code it follows that the 1-Rec is correct and thus we can safely assume that the leading 1-ECs always occur. We estimate this probability using a numerical malignancy count

$$\begin{aligned} &P(\text{1-Rec incor. due to mal. pair in 1-exRec of type } \ell \\ &\quad \text{\& all 1-ECs occur in 1-exRec}) \leq \sum_{i \geq j} \alpha^\ell_{[i],[j]} \gamma_i \gamma_j. \end{aligned} \tag{4}$$

Here $\alpha^\ell_{[i],[j]}$ is the number of malignant pairs of type i and j for a 1-exRec of type ℓ, where we count only those malignant pairs that do not lead to failure of a 1-EC. The numerical malignant pair counts are obtained using the QASM Toolsuite [23]. With this software, one simulates the propagation of Pauli errors in a quantum circuit and thus determines whether a pair of faults is malignant. A selection of these malignancy matrices $\alpha^\ell_{[i],[j]}$ are reproduced in Appendix B.

We upper-bound the triple-fault term as $P(3^+ \text{ faults \& trailing 1-ECs occur}) \leq \binom{N}{3}\gamma_{\max}^3$, where N is the total number of locations in the full 1-exRec (including all 1-ECs) and $\gamma_{\max} = \max_\ell(\gamma_\ell)$ is the maximum failure probability. This expression is probably a large over-estimate of the higher-order terms. For our largest local 1-exRec, the `CNOT` 1-exRec, $N = 1225$. This implies that, including *only* these triple malignancies, the threshold cannot be above $p = 5.7 \times 10^{-5}$ (which is the solution of the equation $p = p^3\binom{1225}{3}$). Since the third term is a very steep function of p, the effect of the term around the thresholds that we find below is not so large. In the results in Section 5, we see that the effect of omitting this term altogether gives similar thresholds in the $O(10^{-5})$ range. Alternative methods of bounding this higher-order term exist, but they either involve triple malignancy counting (which is very time consuming) or heuristic strategies such as malignancy sampling.

Table 3. Number of bad locations of type ℓ in the $\mathcal{G}$ and $\mathcal{V}$ networks that make the ancilla not pass verification. Location types not listed are all good.

Description of type ℓ	Number
h-`CNOT`	1
v-`CNOT`	7
h-`CNOT` followed by `SWAP`	2
h-`CZ` followed by `SWAP`	2
h-`SWAP`	3
$\|0\rangle$-preparation	2
$\|+\rangle$-preparation	1
`X`-measurement	3
wait during gate/prep	2

Let us return to the first term in Eq (2). We calculate the probability that one or more of the 1-ECs fail as

$$P(1^{+}\text{trailing 1-ECs fail}) = 1 - P(\text{1-EC occurs})^{q}, \tag{5}$$

where $q = 0$ for a measurement 1-exRec, $q = 1$ for single-qubit 1-exRecs (including `SWAP` and 1-Prep), and $q = 2$ for the two-qubit 1-exRecs. Furthermore,

$$P(\text{1-EC occurs}) = P(\text{anc. pass})^2 + 2P(\text{anc. pass})^2 P(\text{anc. fail}). \tag{6}$$

How do we estimate the probability for an ancilla to pass? We have $P(\text{anc. pass}) = 1 - P(\text{anc. does not pass})$, where we bound

$$P(\text{anc. does not pass}) \leq P(\text{no pass due to 1 fault in } \mathcal{G}, \mathcal{V}) + P(2^{+} \text{ faulty loc. in } \mathcal{G}, \mathcal{V}), \tag{7}$$

where $P(2^{+} \text{ faulty loc. in } \mathcal{G}, \mathcal{V}) \leq \binom{49}{2}\gamma_{\max}^2$. Thus we assume any two faulty locations in preparation and verification cause the ancilla to fail verification. This is a slight overcount, but the majority of single faulty locations in fact cause the ancilla to fail verification. We have

$$P(\text{no pass due to 1 fault in } \mathcal{G}, \mathcal{V}) = \sum_{\ell \in \mathcal{G}, \mathcal{V}} \alpha_\ell \gamma_\ell, \tag{8}$$

where α_ℓ is the number of bad locations of type ℓ occurring in the $\mathcal{G}, \mathcal{V}$ network as determined by the QASM toolsuite, listed in Table 3.

5 Threshold Studies

We determine a lower bound on the noise threshold for each location type using the formulas for failure probabilities given in the previous sections. In order to estimate thresholds, we repeatedly apply the 17-dimensional flow map $\mathbf{F}_\ell(\gamma)$ to the initial vector of failure probabilities. The only flow that depends on the $a_{\pi/2}$- and $a_{\pi/4}$-preparations is the flow map for these preparations themselves. This implies that the threshold for all the other (Clifford) gates is *independent* of the initial failure probabilities for these two preparations. Thus we can estimate the threshold for these preparations separately and see whether or not they determine the overall threshold.

5.1 Clifford-group Location Thresholds

Let us first consider a much studied case in the literature in which all initial probabilities are the same, i.e., set to some value γ, except the failure probability for a memory location which is set to $\gamma/10$. The threshold of all these gates (except the memory location) will be denoted as γ_c. For our 2D lattice architecture, we find a lower bound of $\gamma_c \geq 1.85 \times 10^{-5}$. We compare our local 2D architecture with a nonlocal architecture with exactly the same circuits (except for the `SWAP`s). For the nonlocal setting, we find $\gamma_c \geq 3.61 \times 10^{-5}$.

In Section 4.1 we argued that we considerably overestimate the three-or-more-fault terms in our failure probability calculation. To see how this affects the threshold, we perform our analysis omitting this triple-fault term entirely in Eq (3). The local threshold then becomes 2.15×10^{-5} and the nonlocal threshold becomes 4.32×10^{-5}, which are small corrections on the numbers given above.

To test the correctness of our local analysis, we also analyze our local 2D architecture in a variant where at all levels of concatenation the error rates of `SWAP` gates are set to zero by hand. Then the only difference between the local and nonlocal case is the additional wait locations in the local architecture. In this scenario, the threshold is $\gamma_c \geq 3.0 \times 10^{-5}$, in which we see the effect of additional waiting in the local model. To eliminate the possibility that the threshold for the nonlocal architecture is suboptimal due to the choice of deterministic error correction, we also analyze the nonlocal circuits in [10]; this gives a lower bound of $\gamma_c \geq 4.3 \times 10^{-5}$. This shows that our choice of circuits is close to optimal.

One may think that the closeness of the local and nonlocal thresholds could be due to the fact that we let the memory error rates be 1/10 of the gate error rates and thus eliminate the effect of additional waiting on the local threshold. Thus we also consider the local and nonlocal architecture where all error probabilities are identical. For the local architecture we obtain a lower threshold, $\gamma_c \geq 1.1 \times 10^{-5}$; for the nonlocal architecture the threshold becomes 1.9×10^{-5}.

To understand the behavior of the failure probabilities at different levels of encoding we use a *threshold reliability information plot* (TRIP) [9]. The behavior of the failure probability for the horizontal `CNOT` in the local model is given in Fig. 7. The value of γ at the crossing point with the 45-degree line is the threshold at level r, also called the level r *pseudothreshold*, for the given location type. In Figure 8, we have plotted the behavior of the `SWAP` location. From these figures it is clear that the horizontal `CNOT` dominates the threshold. The pseudothresholds of the `SWAP` location get worse as a function of r due to the noisiness of the `CNOT`, whereas the pseudothresholds of the `CNOT` get better as a function of r. Another way of seeing this is that after one level of encoding there is a region in which one is below the pseudothreshold of the `SWAP` gate, but not below the pseudothreshold of the `CNOT` gate. In this region, the `CNOT` gets worse, which feeds into a worse pseudothreshold of the `SWAP` at the second level of encoding.

5.2 Approximate Closed Form for Thresholds

Even though our map is high-dimensional, we can capture the essential features by a two-dimensional map. We do this by making some drastic approximations: the 1-EC failure probabilities of Eq (2), and the 3+ fault probabilities of Eq (3), are ignored. Many distinct rectangles are treated as having the same error probabilities. Despite these approximations, the thresholds can be estimated very simply to about a factor of two.

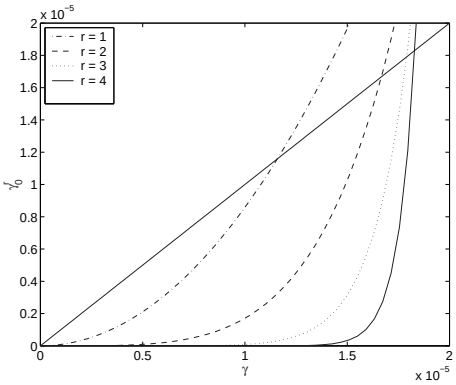


Fig. 7. Initial failure probability γ versus the failure probability γ_0^r at level r of a horizontal `CNOT` location, shown for levels $r = 1, \ldots, 4$.

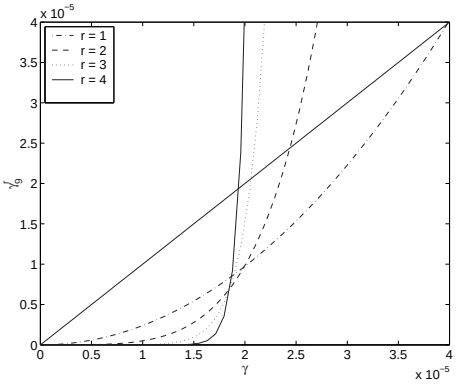


Fig. 8. Initial failure probability γ versus the failure probability at level r of a horizontal `SWAP` location type γ_9^r, shown for levels $r = 1, \ldots, 4$.

We assume that all error levels of gates, preparations, and measurements are comparable (but not necessarily identical) at the base level. If some of the error rates are much larger than others (say, by a factor of 100), it may take more levels of concatenation for the two-dimensional representation to become a good approximation, but these initial conditions will be eventually marginalized.

We only consider the Clifford gates. After a single iteration, information about the locations in each 1-exRec is absorbed into the new error probabilities of simulated locations. Then we find if we start with comparable error rates for all gates, all subsequent iterations have the property that the failure probabilities fall into three distinct groups: (1) those of two-qubit gates (locations [0–7]) (2) those of single-qubit gates (locations [8–10], [15], [16]), (3) those of the preparations (locations [11], [12]) and those of measurement (locations [13], [14]). In other words, the failure probabilities of members of each group are very close in value. We see that this grouping basically corresponds to the number of 1-ECs in the 1-exRec.

We also observe that the measurement failure probabilities are very small compared to

the other failure probabilities and that the behavior of the map changes very little if the measurement failure probability is always set to zero. After the second iteration, the failure probability of the preparation rectangles gets close to those of the single-qubit gates, so we group them together. Thus, we end up with a two-dimensional map described by the equations:

$$x^{i+1} = \left(x^i \; y^i\right) \cdot \begin{pmatrix} a & b \\ b & c \end{pmatrix} \cdot \begin{pmatrix} x^i \\ y^i \end{pmatrix}, \tag{9}$$

$$y^{i+1} = \left(x^i \; y^i\right) \cdot \begin{pmatrix} d & e \\ e & f \end{pmatrix} \cdot \begin{pmatrix} x^i \\ y^i \end{pmatrix}, \tag{10}$$

where i is the iteration level. Here x^{i+1} is the failure probability of the two-qubit gate group at level $i+1$ and y^{i+1} is the failure probability of the single-qubit gate group. The numbers a, b, etc. can be extracted by summing entries in the malignancy matrices $\alpha^{\ell}_{[i],[j]}$ (see Appendix B for two examples of single-qubit and two-qubit gates). For the local model, the numbers for $\ell = 0$ (horizontal `CNOT`) and $\ell = 9$ (horizontal `SWAP`) are $a = 7907$, $b = \frac{55997}{2}$ and $c = 93488$ and $d = 1956$, $e = \frac{18424}{2}$ and $f = 35886$.

The equations for the fixed point of this map are two coupled second-order equations for x and y. By elementary elimination methods, one can derive from these a fourth-order polynomial in x whose roots give the fixed point. One root is at $x = 0$; factoring this out leaves a cubic polynomial. We find that, for the range of parameters we have, the next root, giving the threshold for x, is accurately given by truncating this cubic polynomial to linear order. The resulting expression for the threshold value of x is

$$x = \frac{c}{ac + 4ce - 2bf - f^2}. \tag{11}$$

A parallel analysis for y gives an expression for its fixed point:

$$y = \frac{d}{a^2 + 4bd - 2ae + df}. \tag{12}$$

Now we analyze the first iteration of the full map. For simplicity, we try to follow the analysis in the previous section where all error rates are the same, except for the memory error rates. At the physical level, we start with a gate failure rate γ_g and a wait failure rate γ_w. The x^1 and y^1 are given by quadratic expressions

$$x^1 = \left(\gamma_g \; \gamma_w\right) \cdot \begin{pmatrix} F & G \\ G & H \end{pmatrix} \cdot \begin{pmatrix} \gamma_g \\ \gamma_w \end{pmatrix}, \tag{13}$$

$$y^1 = \left(\gamma_g \; \gamma_w\right) \cdot \begin{pmatrix} J & K \\ K & L \end{pmatrix} \cdot \begin{pmatrix} \gamma_g \\ \gamma_w \end{pmatrix}. \tag{14}$$

For the local model, the parameters are $F = 71779$, $G = \frac{77899}{2}$, $H = 26842$ (taken from the horizontal `CNOT` malignancy matrix) and $J = 8318$, $K = \frac{32843}{2}$ and $L = 21632$ (extracted from the single-qubit wait malignancy matrix).

We have been interested in the case $\gamma_w = \frac{1}{10}\gamma$. We find this case is numerically similar to setting $\gamma_w = 0$, causing these equations to become very simple: $x^1 = F\gamma^2$, and $y^1 = J\gamma^2$.

Setting $x^1 = x$, the threshold expression, and $y^1 = y$ (the threshold expressions Eqs (11,12) and solving for γ, we get two different expressions for γ:

$$\gamma = \sqrt{\frac{c}{F(ac + 4ce - 2bf - f^2)}}, \tag{15}$$

$$\gamma = \sqrt{\frac{d}{J(a^2 + 4bd - 2ae + df)}}. \tag{16}$$

We will be below threshold if γ is lower than either of these two expressions, since then both x^1 and y^1 will be below their threshold values. Therefore,

$$\gamma_c = \min\left(\sqrt{\frac{c}{F(ac + 4ce - 2bf - f^2)}}\ ,\ \sqrt{\frac{d}{J(a^2 + 4bd - 2ae + df)}}\right). \tag{17}$$

Note that these expressions satisfy the required homogeneity property, namely that if all malignant pair counts are scaled by the same constant κ, the threshold changes by the factor $1/\kappa$. Putting in values for these malignant pair counts, we find that the estimated threshold values are $\gamma_c \geq 1.94 \times 10^{-5}$ for the local model and $\gamma_c \geq 4.67 \times 10^{-5}$ for the nonlocal model. This agrees very well with the threshold values of the actual approximate map, which are 1.94×10^{-5} and 4.43×10^{-5}; these are in turn in reasonable agreement with the threshold values for the exact map, which were found to be 1.85×10^{-5} and 3.61×10^{-5}.

Our formulas explain the factor-of-two difference between the local and nonlocal cases in the following way. The malignant pair counts differ between the two models, in different cases by a factor of 5 at most, and a factor of 1 (unchanged, that is) at the least. These formulas represent a kind of weighted average among these different counts, so the actual difference is somewhere between a factor of 5 and a factor of 1.

5.3 a_θ-preparation Threshold and Injection by Teleportation

It would be possible to approach a_θ-preparation in exactly the same way as all other locations in the quantum circuit. A deterministic, fault-tolerant extended rectangle for this preparation is known, which is based on the fact that the state $|\bar{a}_\theta\rangle$ is an eigenstate of the encoded operator $\bar{R}_{\mathsf{Z}}(\theta)\bar{\mathsf{X}}\bar{R}_{\mathsf{Z}}(-\theta) = \bar{R}_{\mathsf{Z}}(2\theta)\bar{\mathsf{X}}$. Fault tolerance is achieved by repeated measurement of this operator followed by error correction, exactly as in Fig. 13 of Ref. [10]. Since we require this rectangle to be deterministic, we must allow for three repetitions, rather than the two repetitions used in Ref. [10].

Unfortunately, this triple-repetition exRec, with all the moves and waits required by locality, has an extremely large number of locations, and the threshold, if we determine it using the formalism discussed for the Clifford gates, is very poor. But we can take advantage of the fact that the $|a_\theta\rangle$ preparations are not mandatory at the lowest level (because these preparations are not needed in 1-ECs) to obtain fault tolerance by a different strategy.

This strategy is *injection by teleportation*, an idea due to Knill [7]. The key circuit is shown in Fig. 9; here, an encoded EPR pair is created and one half of the pair is decoded. This decoded half is used to teleport an unencoded $|a_\theta\rangle$. After teleportation, the resulting state is the encoded $|\bar{a}_\theta\rangle$. The encoded state produced by this circuit may be at any level of concatenation. This circuit is not fault-tolerant, so $|\bar{a}_\theta\rangle$ may be noisier than $|a_\theta\rangle$. But this

circuit will only suffer from single faults in gates in the latter part of the decoding circuit and in the measurements. The total number of such gates is on the order of ten, no matter what the level of concatenation of the final, coded state. Thus, if we inject to level k of teleportation, the encoded error rate γ_θ^k is given approximately by $\gamma_\theta^k = \gamma_\theta^0 + 10\gamma_{\text{CNOT}}^0$.

For purposes of analysis, we imagine that the injection level k is chosen so that the Clifford-group gates are essentially noiseless. At this level, we may use a fault tolerant circuit related to Fig. 13 of Ref. [10], as described above. Upon further concatenation, the probability flows will be determined just by the preparation counts in this 1-exRec. There would be 84 a_θ-preparations in the triple-repetition rectangle; the a_θ-threshold, then, is estimated by the equation $\gamma_\theta = \binom{84}{2}\gamma_\theta^2$. Assuming $\gamma_{\text{CNOT}} = 10^{-5}$, these equations give an estimate for the γ_θ threshold of 1.0×10^{-4}.

Thus, the a_θ-threshold value is not unduly low; we expect, in fact, that this number can be improved with modifications of the circuit in Ref. [10]. We note that this threshold is of course far from the theoretical upper maximum of about 17% established by the analysis of the magic-state distillation technique [24]. We also note that, with the scheme just outlined, there is a serious efficiency cost, in that more levels of concatenation are typically needed to achieve effectively noiseless operation. We expect that this, too, may be improved upon.

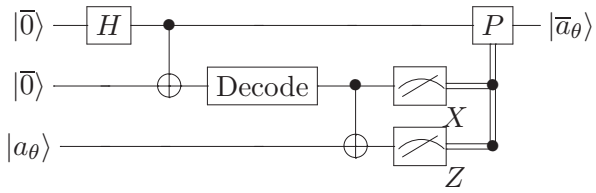


Fig. 9. Injection by teleportation. An encoded EPR state is created and half of the state is decoded. A Bell measurement is carried out on the state $|a_\theta\rangle$ and the decoded half followed by encoded corrective Paulis on the encoded half of the EPR pair. There is ample room in the two-dimensional cell to lay out this circuit, whatever the level of concatenation desired for the final state.

6 Concluding Remarks on the Noise Threshold of the [[7,1,3]] Code

In the fault-tolerance literature, various estimates of the nonlocal threshold for the [[7,1,3]] code have been given. These numbers vary between an optimistic value of $O(10^{-3})$ [5], an estimate of $O(10^{-4})$ [8] and the most recent rigorous *lower bound* of 2.73×10^{-5} established in Ref. [10]. In this section, we argue why, for stochastic independent noise models, the noise threshold of the [[7,1,3]] code *is* realistically $O(10^{-5})$.

The main reasons that the earlier estimates were more optimistic lies in the method of analysis. In Ref. [10] one estimates the failure probability of an *extended* rectangle, which includes error correction before and after the encoded gate. The previous work analyzed the failure probability of a rectangle and tried to take into account incoming errors into the rectangle in a heuristic fashion. The reason that the analysis in Ref. [10] is superior to the other analyses is that the definition of failure introduced in Ref. [10] has a direct interpretation in terms of the failures in the unencoded quantum circuit that the noisy quantum circuit simulates. Namely, by pushing perfect decoders from the end of the computation through to

the front, one generates the unencoded circuit in which failed 1-exRec corresponds to failed elementary gates.

In Ref. [10], the malignancy method was used to establish a *lower bound* of 2.73×10^{-5} for the $[[7, 1, 3]]$ code in the presence of independent adversarial stochastic noise. This derivation of the lower bound assumed, for simplicity, that all gates in the circuit are (worst-case) `CNOT` gates. One may thus think that such an analysis would be far too pessimistic. This conclusion is falsified by two observations. The first observation is that there are already 13245 malignant `CNOT` pairs *alone* in the `CNOT` 1-exRec (see the malignancy matrix in Ref. [10]). Thus just assuming that all gates are noiseless except the `CNOT` would set a threshold of $\frac{1}{13245} \approx 7.55 \times 10^{-5}$. The second observation is that the full nonlocal analysis of the present paper that takes into account all types of locations similarly produces a $O(10^{-5})$ threshold. The threshold changes slightly if we change adversarial noise to depolarizing noise as shown in Ref. [10]; their lower bound for depolarizing noise is then close to 10^{-4}.

These observations lead to two conclusions that are of interest in further studies. Firstly, it is highly desirable to look at different code architectures and see whether one can obtain a threshold in the $O(10^{-3})$ range. One example of such a promising architecture is the C_4/C_6 scheme by Knill [7]. Secondly, as witnessed by the analysis of the [[7,1,3]] code, thresholds have certain crucial dependencies but are otherwise remarkably robust against small variations (in the encoding circuitry, in the noise rates of some gates, and in adding `SWAP` gates for locality, for example). To identify these crucial dependencies, by studying a whole range of codes and understanding in what part of 'code-space' the most promising codes lie, will be an important task ahead.

Acknowledgements

We would like to thank Panos Aliferis for many interesting discussions and insightful comments. KMS would like to thank John Preskill for the invitation to visit the Institute for Quantum Information at Caltech. KMS also thanks Andrew Cross for his work on the QASM toolsuite and also many engaging discussions. DPDV and BMT acknowledge support by the NSA and the ARDA through ARO contract number W911NF-04-C-0098. Some of our quantum circuit diagrams were made using the Q-circuit LaTeX macro package by Steve Flammia and Bryan Eastin.

References

1. D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error. In *Proceedings of 29th STOC*, pages 176–188, 1997, `http://arxiv.org/abs/quant-ph/9611025`.
2. E. Knill, R. Laflamme, and W. Zurek. Resilient quantum computation: Error models and thresholds. *Proc. R. Soc. Lond. A*, 454:365–384, 1997, `http://arxiv.org/abs/quant-ph/9702058`.
3. J. Preskill. Fault-tolerant quantum computation, pp. 213–269 in *Introduction to Quantum Computation*, eds. H.-K. Lo, S. Popescu and T.P. Spiller (1998, World Scientific, Singapore). `http://arxiv.org/abs/quant-ph/9712048`.
4. B.M. Terhal and G. Burkard. Fault-tolerant quantum computation for local non-markovian noise models. *Phys. Rev. A*, 71:012336, 2004, `http://arxiv.gov/abs/quant-ph/0107045`.
5. A. Steane. Overhead and noise threshold of fault-tolerant quantum error correction. *Phys. Rev. A*, 68(4):42322–1–19, 2003, `http://arxiv.org/abs/quant-ph/0207119`.
6. B.W. Reichardt. Improved ancilla preparation scheme increases fault-tolerant threshold. `http:`

//arxiv.gov/abs/quant-ph/0406025.
7. E. Knill. Quantum computing with realistically noisy devices. *Nature*, 434:39–44, 2005.
8. K.M. Svore, B.M. Terhal, and D.P. DiVincenzo. Local fault-tolerant quantum computation. *Phys. Rev. A*, 72:022317, 2005, http://arxiv.org/abs/quant-ph/0410047.
9. K.M. Svore, A.W. Cross, I.L. Chuang, and A.V. Aho. A flow-map model for analyzing pseudothresholds in fault-tolerant quantum computing. *Quantum Information and Computation*, 6(3):193–212, 2005, http://arxiv.org/abs/quant-ph/0508176.
10. P. Aliferis, D. Gottesman, and J. Preskill. Quantum accuracy threshold for concatenated distance-3 codes. *Quant. Inf. Comput.* 6(2), 97165, 2006. http://arxiv.org/abs/quant-ph/0504218.
11. D. Kielpinski, C. Monroe, and D. J. Wineland. Architecture for a large-scale ion-trap quantum computer. *Nature*, 417:709–711, 2002.
12. J.R. Petta, A.C. Johnson, J.M. Taylor, E.A. Laird, A. Yacoby, M. D. Lukin, C.M. Marcus, M.P. Hanson, and A.C. Gossard. Coherent manipulation of coupled electron spins in semiconductor quantum dots. *Science*, 309:2180, 2005.
13. A. J. Skinner, M. E. Davenport, and B. E. Kane. Hydrogenic spin quantum computing in silicon: A digital approach. *Physical Review Letters*, 90:087901, 2003.
14. R. H. Koch, J. R. Rozen, G. A. Keefe, F. M. Milliken, C. C. Tsuei, J. R. Kirtley, and D. P. DiVincenzo. Experimental observation of an oscillator stabilized Josephson qubit. *Phys. Rev. Lett.* 96, 127001, 2006.
15. J. Sebby-Strabley, M. Anderlini, P. S. Jessen, and J. V. Porto. A lattice of double wells for manipulating pairs of cold atoms. *Phys. Rev. A* 73, 033605, 2006. http://arxiv.org/abs/cond-mat/0602103.
16. D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error rate. To appear in the SIAM Journal of Computation, http://arxiv.org/abs/quant-ph/9906129.
17. D. Gottesman. Fault-tolerant quantum computation with local gates. *Jour. of Modern Optics*, 47:333-345, 2000, http://arxiv.org/abs/quant-ph/9903099.
18. T. Szkopek, P.O. Boykin, H. Fan, V. Roychodhury, E. Yablonovitch, G. Simms, M. Gyure, and B. Fong. Threshold error penalty for fault tolerant computation with nearest neighbor communication. *IEEE Trans. Nano.*, 5:42-49, 2006, http://arxiv.org/abs/quant-ph/0411111.
19. T. Metodi, D.D. Thaker, A.W. Cross, F.T. Chong, and I.L. Chuang. A quantum logic array microarchitecture: scalable quantum data movement and computation. In *Intl. Symp. on Mircoarchitecture (MICRO-38)*, 2005, http://arxiv.org/abs/quant-ph/0509051.
20. T. Metodiev, A. Cross, D. Thaker, K. Brown, D. Copsey, F. Chong, and I. Chuang. Preliminary results on simulating a scalable fault tolerant ion-trap system for quantum computation. In *3rd workshop on Non-Silicon Computing (NSC) held in conjunction with the 31st Annual International Symposium on Computer Architecture (ISCA)*, 2004.
21. L.C.L. Hollenberg, A.D. Greentree, A.G. Fowler, and C.J. Wellard. Spin transport and quasi 2d architectures for donor-based quantum computing. *Phys. Rev. B* 74:045311, 2006, http://arxiv.org/abs/quant-ph/0506198.
22. J.M. Taylor, H.-A. Engel, W. Dür, A. Yacoby, C.M. Marcus, P. Zoller, and M.D. Lukin. Fault-tolerant architecture for quantum computation using electrically controlled semiconductor spins. *Nature Physics*, 1:177–183, 2005.
23. A. W. Cross and K.M. Svore. A QASM Toolsuite, http://web.mit.edu/awcross/www/qasm-tools/.
24. S. Bravyi and A. Kitaev. Universal quantum computation with ideal Clifford gates and noisy ancillas. *Phys. Rev. A*, 71:022316, 2005, http://arxiv.org/abs/quant-ph/0403025.

Appendix A Fault-tolerance Requirements

The $[[7,1,3]]$ code is a perfect code which means that all states in the 2^7 dimensional space can be represented as some codeword with at most 1 X error and at most 1 Z error.

The fault-tolerance properties that the constructions of 1-Gas, 1-Meas's, 1-Preps, and 1-

Table A.1. The six stabilizers and logical operations of the [[7,1,3]] code.

Name	Operator
s_1	IIIXXXX
s_2	IXXIIXX
s_3	XIXIXIX
s_4	IIIZZZZ
s_5	IZZIIZZ
s_6	ZIZIZIZ
$\bar{X}$	XXXXXXX
$\bar{Z}$	ZZZZZZZ

ECs must satisfy for a distance-3 code, as stated in Ref. [10], are:
1. If a 1-EC contains no fault, it takes any input to an output in the code space.
2. If a 1-EC contains one fault, it takes any input to a valid output. (The output of a level-1 block is 'valid' if it deviates from the code space by the action of a weight-1 operator).
3. If a 1-EC contains no fault, it takes an input with at most one error to an output with no errors.
4. If a 1-EC contains at most one fault, it takes an input with no errors to an output with at most one error.
5. If a 1-Ga contains no fault, it takes an input with at most one error to an output with at most one error in each output block.
6. If a 1-Ga contains at most one fault, it takes an input with no errors to an output with at most one error in each output block.
7. A 1-Meas with no faults applied to an input with one error agrees with an ideal measurement.
8. A 1-Meas with at most one fault applied to an input with no errors agrees with an ideal measurement.
9. A 1-Prep with at most one fault produces an output with at most one error.

Appendix B Selected Malignant Pair Matrices

In these matrices the columns and rows corresponding to location i are indicated by [i]. Columns (and rows) with only zero elements are omitted.

For the horizontal `SWAP`, location 9, we have

$$\alpha^{9}_{[i],[j]} = \begin{array}{c} \\ [0] \\ [1] \\ [3] \\ [4] \\ [6] \\ [9] \\ [10] \\ [11] \\ [12] \\ [13] \\ [15] \\ [16] \end{array} \begin{pmatrix} [0] & [1] & [3] & [4] & [6] & [9] & [10] & [11] & [12] & [13] & [15] & [16] \\ 16 & 186 & 68 & 40 & 56 & 592 & 72 & 42 & 52 & 106 & 1043 & 68 \\ & 467 & 312 & 204 & 276 & 2856 & 347 & 200 & 244 & 480 & 4931 & 312 \\ & & 63 & 70 & 90 & 942 & 123 & 90 & 96 & 210 & 1782 & 126 \\ & & & 18 & 60 & 618 & 78 & 48 & 60 & 116 & 1080 & 70 \\ & & & & 30 & 812 & 106 & 64 & 76 & 144 & 1404 & 90 \\ & & & & & 3345 & 1088 & 716 & 780 & 1704 & 10572 & 606 \\ & & & & & & 60 & 84 & 100 & 192 & 1890 & 123 \\ & & & & & & & 12 & 44 & 72 & 1338 & 90 \\ & & & & & & & & 24 & 96 & 1398 & 96 \\ & & & & & & & & & 84 & 3114 & 210 \\ & & & & & & & & & & 11612 & 1824 \\ & & & & & & & & & & & 84 \end{pmatrix}$$

For the horizontal `CNOT`, location 0, we have

$$\alpha^{0}_{[i],[j]} = \begin{array}{c} \\ [0] \\ [1] \\ [3] \\ [4] \\ [6] \\ [9] \\ [10] \\ [11] \\ [12] \\ [13] \\ [15] \\ [16] \end{array} \begin{pmatrix} [0] & [1] & [3] & [4] & [6] & [9] & [10] & [11] & [12] & [13] & [15] & [16] \\ 60 & 761 & 241 & 144 & 198 & 2018 & 421 & 144 & 166 & 364 & 2567 & 54 \\ & 2106 & 1295 & 810 & 1116 & 10561 & 2309 & 806 & 918 & 1938 & 13487 & 266 \\ & & 210 & 250 & 324 & 3264 & 692 & 306 & 312 & 714 & 4572 & 84 \\ & & & 68 & 216 & 2103 & 448 & 164 & 190 & 392 & 2668 & 56 \\ & & & & 108 & 2799 & 594 & 224 & 236 & 504 & 3496 & 72 \\ & & & & & 11748 & 5460 & 2237 & 2469 & 5328 & 29008 & 216 \\ & & & & & & 593 & 508 & 532 & 1176 & 6831 & 98 \\ & & & & & & & 48 & 152 & 288 & 3260 & 72 \\ & & & & & & & & 74 & 336 & 3268 & 72 \\ & & & & & & & & & 336 & 7584 & 168 \\ & & & & & & & & & & 25240 & 1560 \\ & & & & & & & & & & & 42 \end{pmatrix}$$