NOISE THRESHOLDS FOR THE $[4, 2, 2]$-CONCATENATED TORIC CODE

Ben Criger

JARA Institute for Quantum Information, RWTH Aachen, Otto-Blumenthal-Straße 20
Aachen, Nord-Rhein Westfalen, 52074 Germany

Barbara M. Terhal

JARA Institute for Quantum Information, RWTH Aachen, Otto-Blumenthal-Straße 20
Aachen, Nord-Rhein Westfalen, 52074 Germany

Received April 18, 2016
Revised August 11, 2016

We analyze the properties of a 2D topological code derived by concatenating the $[4, 2, 2]$ code with the toric/surface code, or alternatively by removing check operators from the 2D square-octagon or 4.8.8 color code. We show that the resulting code has a circuit-based noise threshold of $\sim 0.41\%$ (compared to $\sim 0.6\%$ for the toric code in a similar scenario), which is higher than any known 2D color code. We believe that the construction may be of interest for hardware in which one wants to use both long-range two-qubit gates as well as short-range gates between small clusters of qubits.

Keywords: quantum error correction, color codes, noise threshold

Communicated by: R Cleve & R Laflamme

1 Introduction

Quantum error correction (QEC) is believed to be a necessity for quantum computing. Two-dimensional topological quantum error correcting codes are the leading contenders for the implementation of quantum error correction due to the practical appeal of 2D qubit connectivity, relatively high noise thresholds, and universal fault-tolerant gate constructions (see e.g. the review [1]). Two popular families of 2D topological codes are toric/surface codes [2] and color codes [3], which we briefly introduce in Sections 1.2 and 1.3 for completeness. The surface code architecture is based on planar qubit connectivity of low degree (each qubit participating in four parity checks, each parity check involving four qubits), and has a very high noise threshold $0.6\% - 1\%$ [4,5], when all gate error rates are identical. These advantages make it the focus of current experimental research.

In this manuscript, we consider a scenario in which multi-qubit gates may have different error rates, with long-range gates having higher rates than short-range gates. Previous work [6] has considered extremely noisy long-range operations which connect relatively large regions of low-noise surface code tiles. We consider a scenario in which only a small region, called a cluster, is connected by short-range gates, with error rates for short- and long-range operations that are similar, though not necessarily identical.

In order to arrive at a more clustered layout, we simply concatenate the toric code with a four-qubit code which encodes two qubits, the $[4, 2, 2]$ code. This code by itself is a natural
Noise thresholds for the \([4, 2, 2]\)-concatenated toric code

testbed to assess the quality of parity check measurements as it is the smallest code which can detect a single error. Alternatively, the \([4, 2, 2]\) code can correct a single amplitude-damping error \([7, 8]\), thus providing protection against \(T_1\) errors. For these reasons implementing the \([4, 2, 2]\) code is a natural choice for early quantum error correction experiments. One can imagine that the four qubits in each cluster are coupled to ancilla qubits for parity check measurement using short-range gates. Once the capacity of each four-qubit cluster to detect errors is established, one can tie these clusters together by surface code parity checks, concatenating the toric code with the \([4, 2, 2]\) code. The inter-cluster toric code parity check measurements could be performed using long-range gates. Long-range connections could also be established by creating and distilling entangled states between clusters. The point of this work is not to prescribe how to use such a \([4, 2, 2]\)-concatenated surface code (see Section 5 for ideas on superconducting qubit hardware, \([9, 10]\) for implementations of similar codes using perturbative gadgets and Majorana fermions), but to show that it may offer more flexibility in designing a physical layout while attaining a high noise threshold, similar to the surface code.

Coincidentally, the \([4, 2, 2]\)-concatenated toric/surface code (or \([4, 2, 2]\)-toric code for short) is identical to a reduced or ‘light’ version of a 2D color code. Two-dimensional color codes \([3]\) are 2D topological codes which have an advantage over the surface code in the transversality, and thus \(O(1)\) time and space overhead, of the Hadamard and the \(S\) gate (note that these gates are not transversal in the \([4, 2, 2]\]-toric code, see the discussion at the end of Section 2). It has also been argued in \([11]\) that, despite the lower noise threshold, the color code qubit overhead is lower than that of the surface code.

Another possible advantage of the \([4, 2, 2]\)-toric code over the surface code is that it can interface, using lattice surgery \([12]\), with a 3D color code in which the \(T\) gate is implemented transversally \([13]\). An encoded \(T\) state ancilla can, through this method, be transferred fault-tolerantly to a planar array for further Clifford processing. The Clifford processing itself consists of \(\text{CNOT}\) gates which can be realized on qubits encoded in surface code or triangular color code sheets using lattice surgery along boundaries of the sheets \([14,11]\). It may be possible to do a phase \(S = \text{diag}(1, i)\) and Hadamard gate in the \([4, 2, 2]\)-toric code or a plain surface code if one encodes in lattice defects as in \([15]\).

2D color codes have lower thresholds than the surface code, limiting their experimental feasibility in the near term. This is often attributed to the increased weight of the check operators (six for a hexagonal color code, and four or eight for the square-octagon color code). In the circuit-based noise model, which we also use in this paper (see Section 3), one assumes that all elementary operations in the parity check circuits undergo errors. For example, using a non-scalable decoder based on integer programming, a threshold of 0.082% has been obtained for the triangular square-octagon (4.8.8) color code \([16]\), although the noise threshold assuming noiseless parity check measurements was found to be almost identical to that of the toric code. Earlier work \([17]\) estimated the noise threshold of triangular color codes to be approximately 0.1%. More recent work \([18]\) has shown, using a decoder which maps the decoding problem onto three copies of a surface code decoding problem \([19]\), that a circuit-based noise threshold for the 4.8.8 color code of 0.14% can be attained. The high threshold of the 2D color codes for noiseless parity check measurements (also called the code-capacity threshold) suggests that their poor performance against circuit-based noise is due to
Fig. 1. The toric code on a 5-by-5 lattice with periodic boundary conditions in both dimensions. In this code a qubit is associated with each edge of the lattice, with stabilizer check operators associated with the vertices and faces. The boundaries of faces (also called plaquettes) support a check of the form $Z^\otimes 4$, the co-boundaries of vertices (also called stars), support checks of the form $X^\otimes 4$. Sets of edges that wrap around the torus support the two sets of logical $X$ and $Z$ operators, as shown above.

the diminished reliability of the syndrome information. In a circuit-based error model, the reliability of the syndrome is directly affected by the number of qubits involved in the parity check measurement as it determines the number of CNOT gates. In this paper, we show that a ‘light’ version of the 2D color code, namely the $[4, 2, 2]$-toric code, in which we measure fewer parity checks than in a standard color code, has quite a high noise threshold, almost the same as the toric code itself. This shows that the lower threshold of the color code is not solely due to measuring high-weight check operators. We arrive at these results by showing how the decoding problem is identical to that of the toric code and by carefully optimizing the parity check circuits.

1.1 Overview

In Sections 1.2 and 1.3 we review the toric/surface code and 2D color codes. In Section 2 we detail the check operators and logical operators of the $[4, 2, 2]$-toric code. We show how it can be obtained from the square-octagon color code on a torus. In Section 3 we show that the $[4, 2, 2]$-toric code can be decoded using minimum-weight perfect matching with a deformed Manhattan metric. In Section 4 we present the noise thresholds obtained against various error models. We conclude in Section 5 with a discussion.

1.2 Toric/Surface Codes

The standard example of a 2D topological code is the toric code [2, 20]. It is defined by placing weight-four parity check operators on an $l$-by-$l$ square lattice as in Figure 1. A toric code on an $l$-by-$l$ lattice has $2l^2$ physical qubits, two logical qubits and a distance $d = l$; it is a $[2l^2, 2, l]$ code. The surface code which encodes a single qubit can be obtained by imposing boundary conditions producing an $[l^2 + (l - 1)^2, 1, l]$ code [21]. A qubit-overhead optimized surface code encoding a single qubit can also be constructed with parameters $[l^2, 1, l]$, see Figure 2.
1.3 Color Codes

The two-dimensional color codes are obtained by placing qubits on the vertices of a lattice, with both an $X$ and $Z$ stabilizer check $S_u(X)$ (resp. $S_u(Z)$) on each face $u$ of the lattice [3]. The lattice must be trivalent (all vertices having degree three) and face three-colorable (the set of faces must be divisible into three subsets such that no face from a given subset is adjacent to another face from the same subset). This guarantees that each face is supported on an even number of qubits, and that adjacent faces share exactly two vertices [22], implying that all check operators commute. Examples of lattices obeying these trivalent and three-colorable constraints are a lattice with hexagonal plaquettes and the square-octagon lattice (see Figure 3). Note that one can associate a color to each edge so that the edge joins two plaquettes of that color. For a lattice on the torus there is a linear dependency between the check operators, namely the product of all $Z$ stabilizer (resp. $X$) checks of one color equals the product of $Z$ (resp. $X$) check operators of another color, i.e. $\Pi_{u \in \text{Blue}} S_u(X) = \Pi_{u \in \text{Green}} S_u(X) = \Pi_{u \in \text{Red}} S_u(X)$ and $\Pi_{u \in \text{Blue}} S_u(Z) = \Pi_{u \in \text{Green}} S_u(Z) = \Pi_{u \in \text{Red}} S_u(Z)$. In order to calculate the number of logical qubits which are encoded by the lattice, one can use the expression for the Euler characteristic, $\chi = |V| - |E| + |F|$ ($\chi = 0$ for the torus). Here
Fig. 4. Triangular color code encoding one qubit with $d = 7$: the logical string operators (either $X$ or $Z$) can run along the three boundaries and commute with all checks.

$V$ is the set of vertices, $E$ the set of edges and $F$ the set of faces of the lattice. Trivalence of the lattice implies that $|E| = \frac{3}{2}|V|$ so that $|V| = 2|F|$; the total number of qubits is even. The number of logical qubits for the torus is the difference between the number of physical qubits and the number of linearly-independent checks, in this case $|V| - 2|F| + 4 = 4$. The logical operators $X_i, Z_i, i = 1, \ldots, 4$ form non-trivial loops, running over edges of a specific color, around the torus, similar to the toric code.

One can also define a color code with open boundaries, for example a triangular color code, see Figure 4. One can obtain this triangular code by puncturing a trivalent lattice which covers the sphere. Puncturing means that one removes a qubit and all the check operators which act on it. For the sphere one has $|V| - |E| + |F| = 2$ so that $|V| = 2(|F| - 2)$, again even. The unpunctured code then encodes no qubits. When one removes a single qubit (and its associated checks, a total of six), the total number of physical qubits is odd. The triangular code encodes a single qubit and the oddness of the number of physical qubits implies that $X = X_{\text{all}}$ and $Z = Z_{\text{all}}$ form a pair of anti-commuting logical operators ($X_{\text{all}}$ acts as Pauli $X$ on all qubits in the lattice). A multi-qubit color code can be obtained using a polygon of higher degree [23]. By multiplying these logical operators with check operators, one can deform the logical operators to operators on a boundary of the triangular lattice. One can associate a color to each boundary; the color of the removed check operator.

For this lattice, the parameters of the triangular code are $[n = \frac{1}{2}d^2 + d - \frac{1}{2}, 1, d]$ [16]. 2D triangular color codes allow a transversal logical Hadamard and phase gate, that is, $H = H^\otimes n$, and $S = S^\otimes T \otimes S^\dagger^\otimes T$ with $T$ being a subset of the vertices [22]. In addition, the logical CNOT can be performed using lattice surgery [14, 11, 11].

2 The $[4, 2, 2]$-toric code

We introduce a topological code on a square-octagon lattice with qubits on its vertices. We can create this code by concatenating the toric code (or a surface code) with the $[4, 2, 2]$ code [24]. The stabilizer group of the $[4, 2, 2]$ code is generated by the check operators $XXX$ and $ZZZZ$ and the code encodes two qubits with logical operators:

$$X_1 = XIXI, X_2 = IIXX, Z_1 = IIZI, Z_2 = ZZII.$$
Noise thresholds for the $[4, 2, 2]$-concatenated toric code

Fig. 5. The two stabilizer checks and four logical operators for the $[4, 2, 2]$ code, placed on the vertices of a square.

To concatenate, we replace every qubit of the toric code with a square or cluster with the $[4, 2, 2]$ code defined on its vertices:

Fig. 6. Each edge on the original toric code lattice is replaced with a square containing four qubits.

To produce the concatenated code check operators, the check operators of the toric code are written in terms of the logical operators of the $[4, 2, 2]$ code, and the check operators of each cluster are added. We have the option of choosing one of the logical qubits of the $[4, 2, 2]$ code to be a gauge qubit, i.e. a qubit that will not be used and whose state can be left to vary or be fixed. We will select the logical qubit such that the logical $Z$ (to become part of a $Z^4$ check) is on a side of the square which is parallel to the edge of the underlying square lattice. With this selection, the toric code check operators become the check operators seen in Figure 7 when written using the logical operators of the $[4, 2, 2]$ code.

This concatenated code, the $[4, 2, 2]$-toric code, encodes 2 logical qubits (the logical qubits of the toric code) and 2 gauge qubits (one gauge qubit per cluster of four qubits). The total number of qubits is $8l^2$ and the distance of the code is $2l$ as the logical operator of the toric code must have support in $l$ separate clusters and on two qubits per cluster. Thus the parameters of this code are $[8l^2, 2, 2l]$, showing that the qubit overhead for a given distance is identical to the toric code.

A ‘full’ color code on the 4.8.8 lattice can also be obtained by concatenating the $[4, 2, 2]$ code and the toric code. This requires two toric codes for concatenation, one with $Z$ checks on plaquettes and $X$ checks on stars as usual, and one with $Z$ checks on stars and $X$ checks on plaquettes. This construction can also use two rotated surface codes (see Figure 2), resulting in a color code on a lattice with four open boundaries, encoding two logical qubits.

General relations showing that $D$-dimensional color codes can be viewed as multiple copies of surface codes up to local unitary transformations and addition and removal of ancilla qubits exist, even in the presence of boundaries, see e.g. [23, 25, 26]. Thus, we see a very concrete realization of this mapping using code concatenation: the code concatenation perspective directly suggests ways of doing noisy syndrome measurements and implementing a decoder.
Fig. 7. Check operators and logical Pauli operators for the $[4, 2, 2]$-toric code. The blue octagon supports a $Z$ check, the green octagon supports an $X$ check, the red squares support both $X$ and $Z$ checks.

One can obtain the 4.6.12 color code lattice [16] from an identical concatenation step of two copies of the toric code with the $[4, 2, 2]$ code: one starts with a hexagonal toric code lattice with qubits on edges. The code has hexagonal weight-six plaquettes and weight-three stars which upon concatenation with the $[4, 2, 2]$ code become weight-twelve checks and weight-six checks respectively. Similarly, it is possible to obtain the 6.6.6 color code from concatenating the toric code and the $[6, 4, 2]$ code, see Appendix A.

The difference between the full 4.8.8 color code and the $[4, 2, 2]$-toric code lies in the fact that the full color code encodes an additional logical qubit whose logical operators are string operators on the gauge qubits of $[4, 2, 2]$-toric code. In the $[4, 2, 2]$-toric code, we do not correct for errors on these gauge qubits, nor do we fix their states by check measurements. In other words, a simpler construction for the $[4, 2, 2]$-toric code begins with a color code defined on a square-octagonal tiling and removes $X$ checks from blue octagons and $Z$ checks from green octagons. In doing so, $XX$ edges between green $X$-octagons become undetectable, as do $ZZ$ edges between blue $Z$ octagons. These operators are precisely the gauge operators of the $[4, 2, 2]$ code defined earlier, as seen in Figure 8.

Given this relation between the surface code and the 4.8.8 color code, one can ask about the properties of the triangular version of the concatenated surface code. This code is defined as the triangular color code, with $X$ and $Z$ checks removed from blue and green plaquettes, respectively. Unfortunately, one can now multiply the logical operator of the encoded qubit by the weight-two gauge operators to reduce its weight to one, see Figure 9. To see this, take a logical $Z$ running along the green boundary in Figure 5 on each square touching this boundary we can remove its support by multiplication with a blue $ZZ$ edge. Thus, for the qubit encoded in the triangular code to have high distance, one needs to fix (and thus measure) all the stabilizer checks of the color code. This is unfortunate, as the triangular color code has a transversal $S$ gate, which arises partially due to the oddness of the number of physical qubits. It is obvious that any concatenation of a surface code or copies of a surface
code with the $[4, 2, 2]$ code always results in a logical $\overline{X}$ which acts on an even number of qubits, thus simple transversality of the $S$ gate is excluded for any $[4, 2, 2]$-concatenated surface code.

In the following section, we compare the performance of the toric code and the $[4, 2, 2]$-toric code, focusing on the error threshold.

3 Decoding

To determine the ability of topological codes to correct errors, they are commonly subjected to random errors using three scenarios:

*Data-Only Errors* in which every qubit in the lattice is acted upon by a Pauli $X$ and $Z$ with probability $p$. These errors are detected at the points where they anticommute with local stabilizer checks.

*Data & Syndrome Errors* in which errors act on the lattice as above, and syndrome measurements are subject to symmetric bit-flip with probability $q$ (usually set to $p$ for simplicity). In this scenario, faulty measurements are repeated, typically $d$ times. The resulting measurement record is then decoded.

*Circuit-Based Errors* in which errors act on the lattice and syndrome measurements as
above, but faulty operations are modeled by perfect operations, followed by a random Pauli, sampled uniformly from the set of one- or two-qubit Paulis [4]. In this work, we assume that each operation (including the identity) is subject to the same error rate \( p \), unless otherwise noted.

In each of these scenarios, a logical error rate can be estimated by performing a large number of Monte Carlo trials of the appropriate decoding algorithm (\( 10^4 \) for Figures 14, 15, 17, 19, and 20 below).

The pattern of syndromes generated by errors acting on the color code is different from that of the toric code. Instead of indicating the endpoints of connected chains, the violated stabilizer checks may also be at the endpoints of Y-shaped string-nets (see Figure 10). This additional complication in the structure of the syndromes makes the problem of finding optimal decoders for color codes more difficult. There are decoders based on the renormalization group [27], and decoders which decompose the decoding problem into multiple instances of minimum-weight perfect matching [19]. Using a non-scalable decoder based on integer programming, thresholds of 10.56\% (data-only errors), 3.05\% (data & syndrome errors) and 0.082\% (circuit-based errors) have been obtained for the color code on the square-octagon lattice [16]. In [28] the highest possible noise threshold against data & syndrome errors occurring with equal probability was found to be 4.5\% for the 6.6.6 2D color code.

For the toric code the decoding problem is well understood. To restore the code state, it is necessary to assign an error to the observed syndrome such that, with high probability, the product of the assigned error with the actual error is an element in the stabilizer group. A generated pattern of syndromes does not correspond uniquely to the error which caused it, because any connected chain of errors on data qubits or syndrome bits produces syndromes at its endpoints (see Figure 11).

Thus, decoding for the toric code can be accomplished by noting that the negative log-likelihood of a pattern of errors forming a continuous chain is proportional to a weighted Manhattan distance between the endpoints of the chain [20]. The problem is then reduced to finding a set of pairs which minimizes the total length of the assigned error chains; a well-studied combinatorial problem called minimum-weight perfect matching [29]. Under the three error models described above, using an MWPM decoder, the toric code has threshold error parameters of 10.3\%, 2.9\% and 0.6\%, respectively [4, 30], identical to our results for the toric
Noise thresholds for the $[[4, 2, 2]]$-concatenated toric code

Let us now examine the decoding problem for the $[[4, 2, 2]]$-toric code: it is also solvable by minimum-weight perfect matching of a set of points in 2D or 3D. As in the toric and color codes, the CSS nature of the $[[4, 2, 2]]$-toric code permits $X$ and $Z$ syndromes to be decoded as though they are caused by uncorrelated errors, with a small loss in threshold if errors are in fact correlated (though decoders exist which can compensate for this [31]). As in the toric code, these syndromes occur at the endpoints of error chains. Figure 12 shows some examples of error chains and their endpoints, which can be a pair of squares, a pair of octagons or a square-octagon pair. Again, we see that we only need to find a minimum-length path between non-trivial syndromes and paths which are the same in length (touch the same number of qubits) are equivalent given the gauge qubit degrees of freedom. We can thus consider two sub-lattices, one for correction of $Z$ and one for the correction of $X$ errors: the sub-lattices are formed by $X$-squares and $X$-octagons and $Z$-squares and $Z$-octagons resp., see one sub-lattice in Figure 13.

On the sub-lattice, one marks the checks with non-trivial syndromes (putting defect points on the dual lattice) and one constructs a matching of all defects such that it minimizes the total distance (over the lattice) between them. For a pair of defects on octagons, the minimum path length can still be calculated using the Manhattan distance, taking into account toric boundary conditions as for the toric code. For defect pairs where one or both defects are on a square, the Manhattan distance may be smaller than the minimum path length, corresponding to a path which traverses a forbidden octagon. However, the correct distance will be given by the Manhattan-metric path connecting octagons neighbouring the square endpoints, with an additional unit of length per square endpoint. Each square has two nearest-neighbour octagons per syndrome type, so at most four octagon-octagon distances must be calculated per syndrome pair. This adds minimal overhead to decoding.
Fig. 12. Examples of $Z$ and $X$ error chains ending at defects (stabilizer checks with a syndrome indicating the error type). Consider a single $X$ error which is detected by the adjacent square and octagon, see top-left. An $X$ error on the qubit immediately to the right would have caused the same syndrome, but we do not need to distinguish between the two errors as their product is a gauge qubit logical $X = XX$. Two adjacent $Z$ errors on a square can either be a gauge qubit $Z$, going undetected (on the right), or detected by two adjacent octagons (on the left). In the latter case, three other $ZZ$ errors on the same square could have caused the same syndrome, but all are related by stabilizer $ZZZZ$ check and gauge qubit $Z = ZZ$ on the square, hence we do not need to resolve this ambiguity.

Fig. 13. Square-octagon lattice, with faces shaded over which a path may be constructed to assign a length to a pair of syndromes indicating $X$ errors.
Thresholds

For scenarios in which the stabilizer check measurements may themselves be in error, we consider the process of assigning a distance to changes in the measurement record which are separated in time and space. In principle, we would assign different edge weights to time-like edges representing errors in the measurement of different check operators. In order to provide an accurate comparison with the toric code, however, we use uniform edge weights to obtain thresholds in both the toric and $[4, 2, 2]$-toric codes in the following section.

We note that the mapping onto a minimum-weight matching problem (first used in [32]) is the same as for the full color code decoder in [19, 18] where matchings are sought for three different sub-lattices, i.e. one sub-lattice of red squares and blue octagons, one sub-lattice of red squares and green octagons and one sub-lattice of green and blue octagons. We have thus shown that due to the gauge qubit degree of freedom the reduced color code can be decoded by solving only one minimum-weight perfect matching problem.

4 Thresholds

We begin by comparing the $[4, 2, 2]$-toric code to the toric code in the “Data-Only Errors” scenario in Figure [14]. Given that check operator weight has no effect on the data-only error model, it is perhaps expected that both possess a threshold near 10.3%. Nevertheless, this similarity serves to demonstrate the efficacy of the punctured metric introduced in Section [3]. We consider the “Data & Syndrome Errors” scenario in Figure [15]. Given that the addition to the error model is identical in both cases, it is not surprising to see that the threshold is the same, about 3%.

A more detailed treatment of the circuit-based error model is necessary, in part to show the dependence of the error threshold on syndrome extraction circuit design. We treat the circuit-based model in the following subsection.
4.1 Circuit-Based Errors

If we consider circuit-based errors, the form of the parity check circuits is relevant. For the $[[4, 2, 2]]$-toric code, one could in principle do quantum error correction with the typical procedure for concatenated codes. This means that one encodes the parity check circuits for the toric code with the $[[4, 2, 2]]$ code (since the data qubits are also encoded with the $[[4, 2, 2]]$ code). This means that a single ancilla qubit in $|+\rangle$ in a parity check circuit for the toric code gets replaced by four qubits encoded in $|+\rangle$ of the $[[4, 2, 2]]$ code. This four-qubit ancilla could also be verified so that any single error in the ancilla preparation can only lead to a single error on the four outgoing qubits (modulo gauge qubit errors). A single CNOT gate in the toric code parity check circuit would be replaced by two CNOT gates. Between each CNOT gate one can possibly insert a quantum error correction step for the $[[4, 2, 2]]$ code, a measurement of $XXXX$ and $ZZZZ$, to detect whether errors have occurred. We have chosen not to consider this fully concatenated parity check circuitry here as it requires more overhead, and the efficacy of this scheme is not immediately clear. The question to be solved is how to optimally use the additional error detection information in the toric code decoder. It is not excluded that the threshold of this kind of circuitry is higher than that of the toric code itself; after all the use of the $[[4, 2, 2]]$ code in concatenated form has been shown to numerically lead to a threshold of 3% by Knill in [24].

First we assume that we extract the syndrome of all checks (octagon and square) using single-qubit ancillas and one round of quantum error correction consists of measuring all $Z$ and $X$ checks. One can optimize the circuits used for syndrome extraction such that they take a minimal number of timesteps. It is also possible to constrain the circuit to produce weight-two gauge qubit errors (of $X$ or $Z$ type) on clusters of four qubits. For the toric code, interleaving leads to a single round of quantum error correction taking four timesteps (not counting preparation and measurement of the ancilla qubit) [33]. We obtain a circuit which contains eight CNOT timesteps, shown in Figure [17]. This increases the total probability of an error occurring during measurement, which is reflected in the lower threshold for the
Fig. 16. CNOT schedule for syndrome extraction (shown are eight steps in which CNOTs occur). Not shown are initial state preparation (|0⟩ for Z checks and |+⟩ for X checks) and measurement (in the X basis for X checks and the Z basis for Z checks). The ancillas associated with the weight-four check operators whose CNOT schedules are contained in the top row are measured (in one additional timestep) before the CNOT gates in the bottom row are executed (the remaining qubits can undergo errors during this single step measurement). The square X checks are measured first, followed by a measurement of the square Z checks.}

[[4, 2, 2]]-toric code seen in Figure 17. This threshold can be increased, however, by noting that the gate schedule in Figure 16 does not use every data qubit at every timestep. It is prevented from doing so by the full use of every ancilla qubit at every timestep. With this in mind, we consider an alternative parity check circuit in which the weight-eight octagon checks are measured using a Bell state ancilla, i.e. |+⟩ is replaced by $\frac{1}{\sqrt{2}}(|00⟩ + |11⟩)$. One can decrease the length of the CNOT schedule to four, with two data qubits interacting with each octagon ancilla state at every timestep. The resulting circuit is shown in Figure 18. Using this CNOT schedule, the threshold of the [[4, 2, 2]]-toric code is appreciably increased, as seen in Figure 19.

Though this threshold is still lower than that of the surface code, it far exceeds previous thresholds for color codes. In the following section, we discuss the implications of this moderate threshold value for other error-correcting codes and for hardware architectures with clustered layouts.

5 Discussion & Future Work

To obtain more information about when the [[4, 2, 2]]-toric code could be preferred over the toric code, we examine the consequences of tripling the error rates of CNOT gates associated with octagons. In other words, the depolarizing noise rate on all gates is $p$ except for the CNOT gates used to perform the octagon parity checks where it is $3p$. As we see in Figure 20, the threshold $p_c$ of the [[4, 2, 2]]-toric code becomes $\sim 0.21\%$. This implies that one can tolerate a ‘long-range’ CNOT error rate of $\sim 0.63\%$, above the toric threshold, as long as the short-range CNOT error rate (and all other operations) is a factor of 3 lower. As a result we expect that an error rate difference appreciably higher than a factor of 3 will favour the use of a code like the [[4, 2, 2]]-toric code. Greater precision in this estimate depends on altering the minimum-weight perfect matching algorithm to use edge weights derived from error propagation through the syndrome extraction circuit [5].
Fig. 17. Circuit-based noise threshold comparison using the parity check circuits in Figure 16. Error bars are 95% confidence intervals.

Fig. 18. Four-step data-ancilla gate schedule for syndrome extraction in the $[4, 2, 2]$-toric code. Pairs of octagon ancillas are prepared in Bell states prior to interaction with the data qubits, and their measurement outcomes ($\pm 1$) are multiplied to obtain the syndrome. Note that the square $X$ and $Z$ checks are measured simultaneously. This syndrome extraction requires two ancillas per square and octagon, for a total of $2d^2$ ancillas, identical to the toric code.
Fig. 19. Logical error probability of the $[4, 2, 2]$-toric code with the four-step schedule in Figure 18. The threshold error probability has increased to $\sim 0.41\%$. Error bars are 95% confidence intervals.

Fig. 20. Logical error probability of the $[4, 2, 2]$-toric code with the four-step schedule in Figure 18, with a tripled error rate for all CNOT gates supported on an ancilla inside an octagon. This represents a possible error rate discrepancy for long-range gates. Error bars are 95% confidence intervals.
An open question is whether our results can be extended to show that the full square-octagon color code can have a threshold close to that of the toric code when correcting circuit-based noise. This would imply that one can both get the advantage of transversal gates and a good threshold. However we suspect that the good performance may relate to using the gauge qubit degree of freedom in the decoder and in designing the parity check circuits. Further optimization of the color code decoder in [19, 18], possibly using insights of this paper, would be able to shed light on this question.

Let us add a few comments on a potential hardware advantage of the $[4, 2, 2]$ toric code for superconducting transmon qubits. With these qubits, several types of two-qubit gates have been realized. The two-qubit $cz$ gate which uses a direct capacitive coupling between two transmon qubits has been reported to have a low error rate of 0.6% and short duration $< 50$ ns [34]. The range of this gate is determined by the size of the transmon qubits, approximately 200 $\mu$m in [35] and the gate requires flux-tunable qubits (hence space for flux lines). A surface code implementation which only uses this $cz$ gate seems to hard to achieve in a purely 2D fashion, as the qubits will be closely packed together: see for example Figure 1 in [35] in which one has a 1D array of capacitively-coupled qubits which can be used to implement the parity checks of the repetition code but not those of the 2D surface code. The alternative is to use a bus-resonator-mediated two-qubit gate. Such a resonator-mediated gate can be activated by only microwave control such as in the cross-resonance gate used by the IBM group, see [36, 37] with a currently reported error rate of 1.4% [38]. The resonator-mediated coupling can also be used to enact an iSWAP gate where flux-tunable qubits are brought into resonance, or the qubit state can be explicitly put on the resonator bus as in the two-qubit gate employed in [39] (a variety of other two-qubit gates are listed in [38]). A resonator-mediated gate is the basic building block in the surface code layout for transmon qubits as described in [40]. We may thus speculate that it is of interest to use short-range, short-duration, high-fidelity $cz$ gates between directly coupled qubits, for measuring, say, the square $XXXX$ and $ZZZZ$ checks. To measure the octagonal checks between the four-qubit cluster one can consider using longer-range bus-resonator mediated gates (a bus resonator at 8GHz has physical length of $\sim$ 1 cm which can be compactly wrapped-up) allowing for more flexibility. The octagonal checks could also be measured as direct parity measurements (see references to direct parity check measurements in [11]). As long as long-range operations have error rates appreciably higher than those at short range, we suspect that codes such as the $[4, 2, 2]$-toric code will be applicable.

Acknowledgments

The authors thank Nikolas Breuckmann and Kasper Duivenvoorden for useful discussions, and Tobias Hölzer for his early work on this project. This work is supported by the European Research Council (EQEC, ERC Consolidator Grant No: 682726). B. Criger acknowledges financial support through the program SCALEQIT, and computing support from the RWTH ITC.

Comments

Numerical simulations were performed using py-qcode, which can be found at github.com/bcriger/py-qcode.
Noise thresholds for the $[4, 2, 2]$-concatenated toric code

References

1. Barbara M. Terhal. Quantum error correction for quantum memories. *Reviews of Modern Physics*, 87:307–346, Apr 2015.
2. A. Y. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303:2–30, January 2003, quant-ph/9707021.
3. Hector Bombin and Miguel Angel Martin-Delgado. Topological quantum distillation. *Physical Review Letters*, 97(18):180501, 2006.
4. A. G. Fowler, A. M. Stephens, and P. Groszkowski. High-threshold universal quantum computation on the surface code. *Physical Review A*, 80(5):052312, November 2009, 0803.0272.
5. David S. Wang, Austin G. Fowler, and Lloyd C. L. Hollenberg. Surface code quantum computing with error rates over 1%. *Phys. Rev. A*, 83:020302, Feb 2011.
6. Y. Li and S. C. Benjamin. Hierarchical surface code for network quantum computing with modules of arbitrary size. *ArXiv e-prints*, September 2015, 1509.07796.
7. D. W. Leung, M. A. Nielsen, I. L. Chuang, and Y. Yamamoto. Approximate quantum error correction can lead to better codes. *Physical Review A*, 56:2567–2573, October 1997, quant-ph/9704002.
8. A. S. Fletcher, P. W. Shor, and M. Z. Win. Channel-adapted quantum error correction for the amplitude damping channel. *IEEE Transactions on Information Theory*, 54(12):5705–5718, Dec 2008.
9. Courtney G Brell, Steven T Flammia, Stephen D Bartlett, and Andrew C Doherty. Toric codes and quantum doubles from two-body Hamiltonians. *New Journal of Physics*, 13(5):053039, 2011.
10. Barbara M. Terhal, Fabian Hassler, and David P. DiVincenzo. From Majorana fermions to topological order. *Phys. Rev. Lett.*, 108:260504, Jun 2012.
11. A. J. Landahl and C. Ryan-Anderson. Quantum computing by color-code lattice surgery. Technical Report SAND2014-15911J, Sandia National Laboratories, July 2014.
12. C. Vuillot, D.B. Criger, and B.M. Terhal. Versatile lattice surgery for color codes (March 2016, in preparation).
13. H. Bombin and M. A. Martin-Delgado. Topological Computation without Braiding. *Physical Review Letters*, 98(16):160502, April 2007, quant-ph/0610024.
14. C. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter. Surface code quantum computing by lattice surgery. *New Journal of Physics*, 14(12):123011, December 2012.
15. Matthew B. Hastings and A. Geller. Reduced space-time and time costs using dislocation codes and arbitrary ancillas. *Quantum Information & Computation*, 15(11-12):962–986, September 2015.
16. A. J. Landahl, J. T. Anderson, and P. R. Rice. Fault-tolerant quantum computing with color codes. *ArXiv e-prints*, August 2011, 1108.5738.
17. David S. Wang, Austin G. Fowler, Charles D. Hill, and Lloyd C. L. Hollenberg. Graphical algorithms and threshold error rates for the 2d color code. *Quantum Information & Computation*, 10(9):780–802, September 2010.
18. A. M. Stephens. Efficient fault-tolerant decoding of topological color codes. *ArXiv e-prints*, February 2014, 1402.3037.
19. N. Delfosse. Decoding color codes by projection onto surface codes. *Physical Review A*, 89(1):012317, January 2014, 1308.6207.
20. E. Dennis, A. Kitaev, A. Landahl, and J. Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43:4452–4505, September 2002, quant-ph/0110143.
21. S. B. Bravyi and A. Y. Kitaev. Quantum codes on a lattice with boundary. November 1998, quant-ph/9811052.
22. Aleksander Kubica and Michael E. Beverland. Universal transversal gates with color codes: A simplified approach. *Physical Review A*, 91:032330, 2015.
23. A. Kubica, B. Yoshida, and F. Pastawski. Unfolding the color code. *New Journal of Physics*, 17(8):083026, August 2015, 1503.02065.
24. E. Knill. Quantum computing with realistically noisy devices. *Nature*, 434(7029):39–44, Mar 2005.
25. H. Bombin, G. Duclos-Cianci, and D. Poulin. Universal topological phase of two-dimensional
stabilizer codes. *New Journal of Physics*, 14(7):073048, July 2012.

26. Héctor Bombín. Structure of 2d topological stabilizer codes. *Communications in Mathematical Physics*, 327(2):387–432, 2014.

27. G. Duclos-Cianci and D. Poulin. A renormalization group decoding algorithm for topological quantum codes. In *Information Theory Workshop (ITW), 2010 IEEE*, pages 1–5, Aug 2010.

28. R. S. Andrist, H. G. Katzgraber, H. Bombin, and M. A. Martin-Delgado. Tricolored lattice gauge theory with randomness: fault tolerance in topological color codes. *New Journal of Physics*, 13(8):083006, August 2011, 1005.0777.

29. Jack Edmonds. Paths, Trees, and Flowers. *Canadian Journal of Mathematics*, 17:449–467, February 1965.

30. James William Harrington. *Analysis of quantum error-correcting codes: symplectic lattice codes and toric codes*. PhD thesis, California Institute of Technology, May 2004.

31. N. Delfosse and J. P. Tillich. A decoding algorithm for CSS codes using the X/Z correlations. In *2014 IEEE International Symposium on Information Theory*, pages 1071–1075, June 2014.

32. Tobias Hölder. Study of [[4,2,2]]-concatenated toric code for a scalable circuit-QED architecture. Bachelor’s thesis, RWTH Aachen, 2013.

33. A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A*, 86(3):032324, September 2012.

34. Barends et al. Superconducting quantum circuits at the surface code threshold for fault tolerance. *Nature*, 508:500–503, April 2014.

35. Kelly et al. State preservation by repetitive error detection in a superconducting quantum circuit. *Nature*, 519:66–69, 2015.

36. Corcoles et al. Demonstration of a quantum error detection code using a square lattice of four superconducting qubits. *Nature Communications*, 6:6979, April 2015, 1410.6419.

37. Chow et al. Characterizing a four-qubit planar lattice for arbitrary error detection. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 9500, page 1, May 2015, 1509.02815.

38. J. M. Gambetta, J. M. Chow, and M. Steffen. Building logical qubits in a superconducting quantum computing system. *ArXiv e-prints*, October 2015, 1510.04375.

39. Ristè et al. Detecting bit-flip errors in a logical qubit using stabilizer measurements. *Nature Communications*, 6:6983, April 2015, 1411.5542.

40. D. P. DiVincenzo. Fault-tolerant architectures for superconducting qubits. *Physica Scripta Volume T*, 137(1):014020, December 2009.
Appendix A  Concatenation Construction of the 6.6.6 Color Code

We place the physical qubits of the $[6, 4, 2]$ code on the vertices of red plaquettes of a hexagonal lattice, expressing the checks and logical operators as in Figure 5 above:

![Stabilizer checks and logical operators for the $[6, 4, 2]$ code, placed on the vertices of a hexagon.](image)

The operators $XX$ and $ZZ$, when placed on an edge, are logical operators, though some are products of two generators.

The color code checks associated with green and blue plaquettes can be written as tensor products of these edge operators:

![Green and blue checks of the 6.6.6 color code, expressed using edge operators of the $[6, 4, 2]$ code given in Figure 21. When expressed in this fashion, each of these checks has weight four.](image)

The green and blue checks of the 6.6.6 color code divide into disjoint pairs, those supported on logical qubits 1 and 4 for every red hexagon, and those supported on logical qubits 2 and 3. Placing two physical qubits on each red hexagon, we can see that the weight-four checks on this lattice create a toric code:
Fig. 23. Toric code stabilizers supported on a 6.6.6 lattice with two physical qubits per red hexagon, qubits 2 and 3 from Figure 22.