Title

Low-overhead fault-tolerant quantum computing using long-range connectivity.

Journal

Science Advances, 8(20)

Authors

Cohen, Lawrence

Bartlett, Stephen

Brown, Benjamin

et al.

Publication Date

2022-05-20

DOI

10.1126/sciadv.abn1717

Peer reviewed

PHYSICS

Low-overhead fault-tolerant quantum computing using long-range connectivity

Lawrence Z. Cohen[1], Isaac H. Kim[1,2], Stephen D. Bartlett[1], Benjamin J. Brown[1]*

Vast numbers of qubits will be needed for large-scale quantum computing because of the overheads associated with error correction. We present a scheme for low-overhead fault-tolerant quantum computation based on quantum low-density parity-check (LDPC) codes, where long-range interactions enable many logical qubits to be encoded with a modest number of physical qubits. In our approach, logic gates operate via logical Pauli measurements that preserve both the protection of the LDPC codes and the low overheads in terms of the required number of additional qubits. Compared with surface codes with the same code distance, we estimate order-of-magnitude improvements in the overheads for processing around 100 logical qubits using this approach. Given the high thresholds demonstrated by LDPC codes, our estimates suggest that fault-tolerant quantum computation at this scale may be achievable with a few thousand physical qubits at comparable error rates to what is needed for current approaches.

INTRODUCTION

Quantum computing devices are now capable of outperforming even the fastest conventional supercomputers at certain tasks (*1*). However, to execute many quantum algorithms of practical interest, it is widely believed that a fault-tolerant architecture will be required to identify and correct errors in noisy quantum hardware. Fault-tolerant architectures come with a substantial overhead cost, using a large number of low-noise physical qubits to encode and process quantum information with even a small number of protected qubits. Specifically, it has been estimated that millions of qubits will be needed to solve relevant problems in quantum chemistry (*2*–*4*), to break cryptosystems (*5*, *6*), or to get an advantage over classical algorithms using polynomial speedups (*7*, *8*). These large overheads provide a daunting challenge for scaling up from today's noisy devices to large-scale fault-tolerant quantum computers.

The enormous resource estimates mentioned above are all obtained using fault-tolerant architectures based on quantum error-correcting codes with local check operators (*9*–*12*). These codes have a number of highly desirable features for quantum computation, including high thresholds and fast decoders (*5*, *13*). The locality of these codes means that quantum error correction can proceed using only entangling gates between neighboring qubits arranged in a two-dimensional layout, i.e., on a chip. Thus, while local codes provide a clear pathway to demonstrate the principles of fault tolerance using existing quantum technology, these overheads mean that useful fault-tolerant quantum computing with this approach will likely remain out of reach in the near term.

Locality of gate operations is a physically well-motivated constraint. Recently, however, there has been substantial progress in developing long-ranged entangling gate operations in a variety of quantum processing systems, including those based on superconductors (*14*), semiconductors (*15*–*17*), and trapped ions (*18*, *19*). Optical photons provide an approach that is not naturally constrained to a local two-dimensional layout (*20*, *21*) and can also allow for other qubit systems to be connected into complex quantum networks (*22*–*25*). Recent work has also considered emulating long-range interactions using a local quantum architecture and classical communication (*26*), and architectures have been proposed where long-range interactions are constrained on interconnected planar arrays of matter-based qubits (*27*). The possibility of long-range connectivity opens the door to a new class of quantum codes and fault-tolerant architectures that can harness this connectivity to our advantage.

Here, we show how to perform fault-tolerant quantum computation with an architecture that exploits long-range connectivity to greatly reduce the overhead, compared with local approaches. Rather than focusing on asymptotic behavior, we consider the overhead savings that may be possible in the scale of devices expected in the near term, where, for example, fault-tolerant quantum computing on 50 logical qubits may be possible with only a few thousand physical qubits while maintaining a code distance of $d = 14$ to 16. For comparison, a surface code–based architecture requires at least 10,000 qubits to attain a similar number of logical qubits and code distances. Provided that long-range coupling becomes sufficiently reliable to go below the fault tolerance threshold of our scheme, we anticipate that such an architecture will be capable of performing nontrivial quantum algorithms at a scale compatible with current roadmaps for quantum devices under development during the next few years.

Our approach uses quantum low-density parity-check (LDPC) codes, which efficiently encode a large amount of logical information for a given number of physical qubits. There has been a recent surge of interest in this subject [see (*28*) for a recent review], spurred by Gottesman's remarkable observation (*29*) that quantum LDPC codes meeting certain criteria can be used to achieve fault-tolerant quantum computing with constant overhead. While research into quantum LDPC codes is still in its infancy, they are showing promise. Codes that fulfill Gottesman's criteria are now known (*30*). Moreover, recent numerical studies indicate that LDPC codes can achieve reasonably high thresholds (*31*–*33*). Recent breakthroughs in achieving high code distances indicate that there is room for further development (*34*–*36*).

To use these LDPC codes for quantum computation, one must be able to fault-tolerantly implement a universal set of protected logic gates. While Gottesman (*29*) establishes a method to perform

[1]Centre for Engineered Quantum Systems, School of Physics, University of Sydney, Sydney, New South Wales 2006, Australia. [2]Department of Computer Science, UC Davis, Davis, CA 95616, USA.
*Corresponding author. Email: b.brown@sydney.edu.au

quantum computation using fault-tolerant gate teleportation (*37*), the cost associated with the distillation of the requisite resource state (*38*) is not understood well in the practical regime of interest.

Here, we introduce a flexible method to perform low-overhead quantum logic gates for a general class of quantum LDPC codes. Our work can be thought as a generalization of lattice surgery (*39*), where an ancillary system is coupled to a quantum error-correcting code to measure logical Pauli operators in a fault-tolerant way. Our approach to low-overhead quantum logic builds on an extensive literature into the use of code deformations to perform Clifford gates via measurement that have been well studied for topological codes (*5*, *13*, *39*–*42*), and which have recently been generalized to certain classes of quantum LDPC codes (*43*–*46*). To use this approach for quantum LDPC codes in a way that maintains the desirable low overheads, we construct the required ancillary system by adapting weight reduction methods proposed in (*47*, *48*) to measure the desired logical operators of a given quantum LDPC code. These logical operations then yield a universal gate set for fault-tolerant quantum computing when supplemented with noisy ancilla state injection via magic state distillation (*49*). Thus, our scheme provides an explicit way of performing low-overhead fault-tolerant quantum computing using quantum LDPC codes, which is applicable to codes of even modest size.

Before presenting our detailed results, we briefly illustrate the potential overhead improvements that our construction enables. We will make use of existing quantum LDPC codes with explicit constructions and efficient decoders, together with our fault-tolerant approach to performing logic gates on these codes. Table 1 shows the overhead required to complete a round of logical operations with error correction for a given number of logical qubits k and code distance d for a number of quantum LDPC codes, specifically, hyperbicycle and hypergraph product codes explicitly constructed in (*31*, *50*). By one round of logical operations, we mean a set of logical Pauli measurements such that each logical qubit is acted on nontrivially by at most one measurement. We directly compare the qubit resources for our construction against the use of surface codes encoding the same number of logical qubits and with the same code distance, with the latter serving as a proxy for how well the codes protect logical quantum information. The surface code is currently the predominant candidate for a quantum architecture, and considerable effort has been spent optimizing its overhead for fault-tolerant computation. Our analysis thus shows the potential overhead improvement that can be achieved using a nonlocal architecture as compared to a local architecture.

All codes we have used, and their fault-tolerant operations, use check operators involving no more than 13 qubits for the largest code in Table 1. (However, it is not the case that all stabilizer generators will have this weight, and most generators will have lower weight.) This weight is larger than that of the surface code, but not by a substantial margin. Because this number is fixed at a small constant value, errors do not spread substantially during measurement of these check operators, which would otherwise affect the threshold of the scheme. However, we emphasize that the stabilizer weights and the circuits used to measure the stabilizer generators will still affect the performance of this scheme. In particular, higher weight stabilizers will increase the failure rate under circuit-level noise, and this should be taken into account when considering the estimates in Table 1. Efficient decoders for the codes in Table 1 have also been designed that perform comparably to minimum weight perfect matching decoding on surface codes of similar distance (*31*).

We implement Clifford gates through parity measurement of logical qubits in the Pauli basis. To keep the overhead low, we restrict the number of logical qubits that can participate in a single measurement round, and we call this number the parallelism of the scheme (see Fig. 1). For a given level of parallelism and given error correcting code, we require n_{anc} physical qubits to create the ancilla systems used in logical measurement, and n_{data} is the size of the code used to store the logical information. Our analysis shows that at very small code sizes, quantum LDPC codes give only a modest overhead improvement when compared to surface code architectures. However, as the size of the system increases, we see that the improvement in overhead for the quantum LDPC codes becomes very substantial,

Table 1. Overhead estimates. Estimates of the overhead required to perform a round of logic, including those qubits needed to encode the data as well as additional ancilla qubits required to perform fault-tolerant gates. We use LDPC codes constructed in (*31*, *50*), which all have initial check weights of no more than 10. We denote the number of logical qubits as k and the distance of the code as d. Comparisons are made against the surface code with the same distance. Here, "parallelism" denotes the number of logical qubits that can be acted upon nontrivially in one round of error correction, and which determines the number of required ancilla qubits. The number of data, ancillary, and total physical qubits needed to perform one round of logical measurements with error correction is denoted n_{data}, n_{anc}, and n_{tot}, respectively. We do not include any ancilla qubits that may be used for error syndrome extraction. Estimates for the surface code were obtained using the compact block scheme from (*40*).

k	d	Parallelism	Code family	n_{data}	n_{anc}	n_{tot}
18	8	2	Hyperbicycle	294	500	800
			Surface	1,152	128	1,300
50	14	2	Hyperbicycle	900	1,400	2,300
			Surface	9,800	300	10,000
	16	20	Hypergraph	1,922	5,000	7,000
			Surface	12,800	2,000	15,000
578	16	578	Hypergraph	7,938	120,000	130,000
			Surface	150,000	75,000	225,000
		68	Hypergraph	7,938	15,000	23,000
			Surface	150,000	10,000	160,000

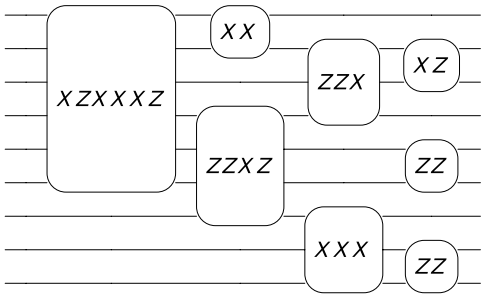

Fig. 1. Parallelism. An example circuit consisting of Pauli measurements on an architecture with a parallelism of six. This circuit contains four rounds of error-corrected logical measurements. In each round, at most, six logical qubits in total can be involved in logical measurements.

offering an order of magnitude improvement at several hundred logical qubits. Current algorithms for minimal nontrivial quantum chemistry calculations (*51*) require around 100 to 200 logical qubits, and so, even at this regime, it may be beneficial to use quantum LDPC codes. We expect that for larger algorithms, such as more complex quantum chemistry algorithms (*2*–*4*) or Shor's algorithm (*6*, *52*), where several thousand logical qubits are required, the overhead gains will be substantial.

These encouraging reductions in overhead motivate experimental work toward the design and realization of quantum LDPC codes in the laboratory. At the physical level, it remains to find efficient ways to measure check operators extending over distant qubits with high fidelity. With long-range coupling now demonstrated using a number of very different approaches (*14*–*19*), as well as proposals allowing all-to-all coupling (*53*), there is substantial room for innovation here. It is also critical that we identify check operator readout circuits that maintain the fault tolerance of the scheme, as well as account for cross-talk that may be present in long-range interactions, and this will require the development of quantum LDPC codes at the level of the circuit error model (*27*, *54*).

Last, further study into compilation of quantum algorithms will allow us to determine what level of parallelism is required to efficiently execute a quantum computation. A commonly used elementary fault-tolerant gate set consists of nondestructive measurements of arbitrary Pauli strings (*4*, *40*, *55*). If the parallelism is strictly less than the number of logical qubits, clearly, not all the Pauli strings can be measured directly. In particular, if the weight of the Pauli string exceeds the parallelism, one would need to break that measurement down into a sequence of (lower-weight) Pauli measurements, leading to a reduced speed. However, if most of the Pauli strings have small weights, the speed of the two approaches will not differ substantially. We anticipate this to be the case if the goal is to simulate a locally interacting many-body quantum spin Hamiltonian using the Trotter-Suzuki method (*56*), but it is unclear how the two approaches will differ for methods such as qubitization (*57*).

RESULTS

We now present our main result: a procedure to implement fault-tolerant logic gates in quantum LDPC codes via a generalization of lattice surgery, in a way that preserves the low overheads. We propose a method for implementing fault-tolerant gates on quantum LDPC codes by using multilogical Pauli measurements (*40*). Our result expands on a set of techniques originally devised to reduce the weight of stabilizer generators of quantum codes (*47*, *48*). We extend these results to include measurement of all logical Pauli operators, allowing for implementation of the full logical Clifford group. Our method is a form of code deformation, in which we transform our code into a new code and in doing so obtain logical information about our original code (*5*, *13*, *39*–*42*).

Our construction enables us to perform single-qubit Pauli measurements, as well as parity measurements between logical qubits on one or multiple LDPC blocks in an arbitrary choice of Pauli basis. This capability gives us a measurement-based approach for realizing the full Clifford gate set (*40*). This gate set can be supplemented with magic state distillation and state injection to achieve universal quantum computing (*49*). We will discuss universal quantum computing in more detail in the "Low-overhead fault-tolerant quantum computation" section. Our construction also guarantees that the distance of the code is preserved during the code deformation, and so, we retain the error-correcting capabilities of our code.

We begin by setting some basic notation and terminology in the next section. The "Code deformation" section presents our construction for measuring logical multi-qubit Pauli operators. We then prove that our construction preserves the distance of the code throughout the process in the "Fault tolerance" section.

Notation and terminology

We describe quantum error-correcting codes with the stabilizer formalism. Let $\mathcal{P} = \langle I, X, Y, Z\rangle$ be the Pauli group and $\mathcal{P}_n = \mathcal{P}^{\otimes n}$ the Pauli group acting on n qubits. A stabilizer code is defined by an Abelian group $\mathcal{S} \subset \mathcal{P}_n$ such that $-I \notin \mathcal{S}$. The code $\mathcal{C}$ is a subspace spanned by the common +1 eigenvalue eigenstates of the operators in $\mathcal{S}$. The logical Pauli operators are operators in $\mathcal{P}_n$ that commute with every operator in $\mathcal{S}$ but are not themselves in $\mathcal{S}$. If $\mathcal{S}$ is generated by an independent set of generators $\{g_1, \ldots, g_m\}$, the number of logical qubits of the code is $k = n - m$. The distance of the code is equal to the weight of the least-weight nontrivial logical operator where the weight of an operator in $\mathcal{P}_n$ is the number of qubits on which it acts nontrivially, i.e., with nonidentity support. Of particular interest is a class of stabilizer codes known as Calderbank-Shor-Steane (CSS) codes. These are codes with a stabilizer group that can be generated by a set that includes only two types of elements: those that are the product of Pauli-X operators only and those that are the product of only Pauli-Z operators.

We consider families of stabilizer codes $\mathcal{S}_n$ such that each member of family is indexed by the number of qubits n. Let w_n be the maximum weight of a stabilizer of a generating set of $\mathcal{S}_n$, and let q_n be the maximum number of stabilizer generators that act on any given qubit for a specified generating set. A family of LDPC codes then is a sequence $\mathcal{S}_n$ that can be generated by a set of stabilizer generators such that $w_n = O(1)$ and $q_n = O(1)$. For the remainder of the paper, we will drop the explicit dependence on n and assume that these quantities are constants.

We can describe LDPC codes using the Tanner graph. Let $\mathcal{T} = (V, C, E)$ be a bipartite graph. Each node in V corresponds to a physical qubit in the code, and each node in C corresponds to a generator of the stabilizer group of the code. We draw an edge between $c \in C$ and $v \in V$ if the generator corresponding to c acts nontrivially on the qubit corresponding to v. We label each edge with either an X, Z, or Y depending on how the generator acts on the qubit. For the

case of CSS codes, we can instead label the nodes in C with X or Z, as opposed to labeling the edges. We will often abuse notation and use the labels for variable and check nodes of a Tanner graph to refer to their respective physical qubits and stabilizer generators directly.

Code deformation

We now present our main technical contribution, starting with an outline of the basic idea behind implementing measurements using code deformation. Suppose we have a stabilizer code $\mathcal{S} = \langle g_1, \ldots, g_m \rangle$ along with a logical operator $\widetilde{L}$ that we wish to measure. We can interpret our procedure as adding $\widetilde{L}$ to the generating set of a new code that includes this operator and then removing it again once we have reliably obtained the measurement outcome. Note, however, that simply adding $\widetilde{L}$ to $\mathcal{S}$ will not yield an LDPC code because, in general, $\widetilde{L}$ is a high-weight operator. To maintain the key properties of an LDPC code, we need a stabilizer code that includes $\widetilde{L}$ in its stabilizer group whereby $\widetilde{L}$ can be generated with constant- and ideally low-weight generators. This can be achieved by creating an extended system with additional qubits that include a set of low-weight stabilizer generators $S_1, \ldots, S_l$ whose product gives $\widetilde{L}$. The product of the measurement results for each stabilizer generator gives us the measurement result of $\widetilde{L}$. It is also important that the stabilizers of the new code do not generate any other logical operators of $\mathcal{S}$ so that we do not make any unwanted logical measurements that may affect our computation.

Logical CSS measurement

Here, we describe our construction for the fault-tolerant measurement of logical Pauli operators in CSS codes, illustrated with an example in Fig. 2 for concreteness. We concentrate on the case for measuring X logical operators but remark that an analogous procedure will hold for measuring Z logical operators by reversing the X and Z terms in the following discussion. Let $\mathcal{C}$ be a CSS stabilizer code described by Tanner graph $\mathcal{G} = (V, C, E)$. We show how to measure a specific instance of an X logical operator $\widetilde{X}$. To do so, we deform code $\mathcal{C}$ onto a new code $\mathcal{C}_{\text{merged}}$ described by the Tanner graph that includes $\mathcal{G}$ as a subgraph, $\mathcal{G}_{\text{merged}} \supset \mathcal{G}$.

Our goal is to define a new LDPC code whose stabilizer group includes $\widetilde{X}$. We can use this new code to infer the measurement result of $\widetilde{X}$ by measuring its stabilizer generators. The construction of $\mathcal{G}_{\text{merged}}$ is expressed in terms of $\mathcal{G}$ and a subgraph of $\mathcal{G}$ specified by $\widetilde{X}$ that we call $\mathcal{G}_{\widetilde{X}} = (V_{\widetilde{X}}, C_{\widetilde{X}}, E_{\widetilde{X}})$, together with a dual graph $\mathcal{G}^T_{\widetilde{X}} = (V^T_{\widetilde{X}}, C^T_{\widetilde{X}}, E^T_{\widetilde{X}})$. The variable nodes $V_{\widetilde{X}}$ of the subgraph $\mathcal{G}_{\widetilde{X}}$ are the qubits that support $\widetilde{X}$, checks C_{eX} are Z type stabilizers that share an edge of the Tanner graph with variable nodes $V_{\widetilde{X}}$, and edges $E_{\widetilde{X}} \subseteq E$ are those of E that are incident to nodes included in both $V_{\widetilde{X}}$ and $C_{\widetilde{X}}$ (see Fig. 2A). The dual graph $\mathcal{G}^T_{\widetilde{X}}$ is such that for each $v \in V_{\widetilde{X}}$, we have a corresponding node $v^T \in C^T_{\widetilde{X}}$, and likewise for each $c \in C_{\widetilde{X}}$, we have a corresponding vertex $c^T \in V^T_{\widetilde{X}}$. For each edge $e \in E_{\widetilde{X}}$ with $e = (v, c)$, we have $e^T \in E^T_{\widetilde{X}}$ with $e^T = (v^T, c^T)$ such that the nodes $v^T \in C^T_{\widetilde{X}}$ and $c^T \in V^T_{\widetilde{X}}$ are those that correspond to the nodes of the original subgraph (see Fig. 2B). For now, we assume that there is no strict subset of qubits $V' \subset V$ such that V' supports a distinct X logical operator, before explaining the differences with the more general case.

We define the Tanner graph $\mathcal{G}_{\text{merged}}$ in terms of $\mathcal{G}$ and our new graphs $\mathcal{G}_{\widetilde{X}}$ and $\mathcal{G}^T_{\widetilde{X}}$. Specifically, we combine the Tanner graph $\mathcal{G}$ with r copies of $\mathcal{G}^T_{\widetilde{X}}$ and $r - 1$ copies of $\mathcal{G}_{\widetilde{X}}$ using additional edges $E_{\text{extra}} \subset E_{\text{merged}}$. We layer the copies of $\mathcal{G}_{\widetilde{X}}$ and $\mathcal{G}^T_{\widetilde{X}}$ in an alternating fashion (see Fig. 2C). The additional edges E_{extra} connect adjacent layers of $\mathcal{G}^T_{\widetilde{X}}$ and $\mathcal{G}_{\widetilde{X}}$ as shown in Fig. 2C. To explicitly describe the edges of E_{extra}, we index copies of these graphs and their corresponding objects $\mathcal{G}^T_{\widetilde{X}}[j]$ with $1 \le j \le r$ and $\mathcal{G}_{\widetilde{X}}[j]$ with $2 \le j \le r$. We can regard $\mathcal{G}_{\widetilde{X}} \subseteq \mathcal{G}$ as $\mathcal{G}_{\widetilde{X}}[1]$. In later sections, we will refer to the layers corresponding to $\mathcal{G}^T_{\widetilde{X}}[j]$ as dual layers, and the layers corresponding to $\mathcal{G}_{\widetilde{X}}[j]$ as primal layers. We will also refer to the final layer $\mathcal{G}^T_{\widetilde{X}}[r]$ as the boundary layer.

Let us also append indices to the objects of $\mathcal{G}_{\widetilde{X}}[j] = (V[j], C[j], E[j])$ and $\mathcal{G}^T_{\widetilde{X}}[j] = (V^T[j], C^T[j], E^T[j])$ to define E_{extra}. Recall that for each variable and check in $\mathcal{G}_{\widetilde{X}}$, we have a corresponding check or variable, respectively, in $\mathcal{G}^T_{\widetilde{X}}$. Thus, we also have that each $v[j] \in V[j]$ has a corresponding vertex $v^T[k] \in C^T[k]$, and likewise for each $c[j] \in C[j]$, we have $c^T[k] \in V^T[k]$. The appending edges E_{extra} then include all edges $(v[j], v^T[j])$ and $(c[j], c^T[j])$ for all v, c, and $1 \le j \le r$,

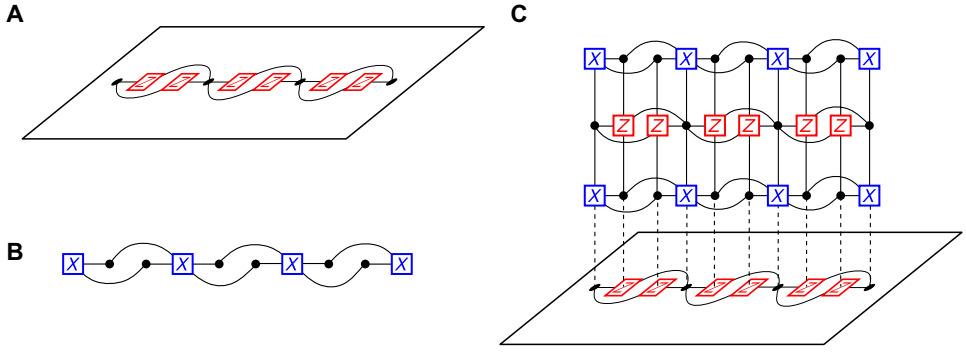

Fig. 2. Measurement of a logical $\widetilde{X}$ operator of the code $\mathcal{C}$. (**A**) Bipartite subgraph $\mathcal{G}_{\widetilde{X}}$ of the Tanner graph of $\mathcal{C}$ on the support of $\widetilde{X}$. Black nodes are the variable nodes corresponding to qubits in the support of $\widetilde{X}$. Red nodes are the check nodes corresponding to Z-type stabilizers in $\mathcal{C}$ that act on qubits in the support of $\widetilde{X}$. (**B**) Dual graph $\mathcal{G}^T = (V^T_{\widetilde{X}}, C^T_{\widetilde{X}}, E^T_{\widetilde{X}})$ of the logical $\widetilde{X}$ in (A). There is a one-to-one mapping between the X-type generators and the qubits in (A), and the qubits and the Z-type generators in (A). (**C**) Measurement of $\widetilde{X}$ using the ancilla system $\mathcal{G}_{\text{anc}} = \mathcal{G}_{\text{merged}} \backslash \mathcal{G}$. The Tanner graph $\mathcal{G}_{\text{anc}}$ is constructed by taking alternating layers of the subgraph $G^T_{\widetilde{X}}$ in (B) and the subgraph $\mathcal{G}_{\widetilde{X}}$ in (A). The vertical edges are the set E_{extra}, which connect adjacent layers. The product of the X generators gives the logical $\widetilde{X}$, and hence, the product of the measurement results for each X generator gives the measurement result of the logical $\widetilde{X}$. After merging the codes and measuring $\widetilde{X}$, we then split the codes by measuring the stabilizers for $\mathcal{C}$ and measuring the qubits in $\mathcal{C}_{\text{anc}}$ in the Z basis, returning us to the original code space.

and edges $(v[j+1], v^T[j])$ and $(c[j+1], c^T[j])$ for all v, c, and $2 \leq j \leq r-1$.

To summarize, with $E_{\text{extra}} \subset E_{\text{merged}}$ defined, we have now specified all of the objects of $\mathcal{G}_{\text{merged}}$. We have that $\mathcal{G}$ is a subgraph of $\mathcal{G}_{\text{merged}}$. Likewise, the variables of $\mathcal{G}_{\bar{X}}^T[j]$ and $\mathcal{G}_{\bar{X}}[j]$ are variables of $\mathcal{G}_{\text{merged}}$. The checks of subgraphs $\mathcal{G}_{\bar{X}}[j]$ are Z stabilizers for $\mathcal{G}_{\text{merged}}$ and the checks of dual graphs $\mathcal{G}_{\bar{X}}^T[j]$ are X stabilizers of the merged Tanner graph. Edges E_{extra} fix graphs $\mathcal{G}$, $\mathcal{G}_{\bar{X}}[j]$, and $\mathcal{G}_{\bar{X}}^T[j]$ together. It will sometimes be helpful to refer to $\mathcal{G}_{\text{anc}} = \mathcal{G}_{\text{merged}} \backslash \mathcal{G}$ as the ancilla system—the resource used to make parity measurements.

Having constructed our new code defined by the Tanner graph $\mathcal{G}_{\text{merged}}$, we now present the following lemma demonstrating that $\bar{X}$ is an element of the stabilizer group of this new code.

Lemma 1. Let $\mathcal{C}$ be a CSS LDPC code and $\bar{X}$ an X logical operator in $\mathcal{C}$ such that there is no other X logical operator supported on a strict subset of the qubits in $\bar{X}$. Then, the construction for $\mathcal{G}_{\text{merged}}$ above gives a code $\mathcal{C}_{\text{merged}}$ that contains $\bar{X}$ in its stabilizer group whereby

$$\prod_{\substack{j=1,\ldots,r \\ v^T[j] \in C^T[j]}} v^T[j] = \bar{X} \tag{1}$$

We recall that we have used the notation where $v_T[j]$ is the stabilizer generator represented by this check node.

Proof. We prove this lemma by showing that the support of $\prod_{j,\, CT[j]} v^T[j]$ on the qubits of $\mathcal{G}_{\text{anc}}$ is trivial, and is nontrivial on $V[1]$, which is the support of $\bar{X}$. By definition, stabilizers $v^T[j]$ are supported on variables $v[j] \in V[j]$ and $c^T[j] \in V^T[j]$. We therefore concentrate on these qubits.

Observe that each Z generator $c \in C[1]$ must be connected to an even number of qubits in $V[1]$ in order for the stabilizers to commute with $\bar{X}$. Consequently, each physical qubit $c^T[k] \in V^T[k]$ must be connected to an even number q_X of X generators in $C^T[k]$. Hence, each qubit $c^T[k]$ supports the term $(X_{c^T[k]})^{q_x} = \mathbb{1}$ for the operator $\prod_{j,\, CT[j]} v^T[j]$.

Furthermore, each physical qubit $v[k] \in V[k]$ for $k \geq 2$ is connected to exactly two X generators, $v^T[k-1]$ and $v^T[k]$. It follows that each qubit $v[k] \in V[k]$ supports the term $(X_{c^T[j]})^2 = \mathbb{1}$ for the operator $\prod_{j,CT[j]} v^T[j]$.

Last, each physical qubit $v[1] \in V[1]$ is connected to exactly one X generator in $\mathcal{G}_{\text{anc}}$, specifically $v^T[1]$. Consequently, the product of all the X generators in $\mathcal{C}_{\text{anc}}$ gives precisely the logical operator $\bar{X}$, thus completing the proof of Lemma 1.

To make a measurement of $\bar{X}$ in a practical way, we first determine $\mathcal{G}_{\text{merged}}$ and prepare each physical qubit in $\mathcal{G}_{\text{anc}}$ in the 0 state. We then measure all the stabilizer generators in $\mathcal{G}_{\text{merged}}$ and perform a round of error correction. To ensure that this procedure is fault tolerant in the presence of noisy measurements, we can repeat this step d times (*42*). Once we have fault tolerantly obtained the result of the measurement of $\bar{X}$, we can return to the original code space $\mathcal{C}$ by measuring each physical qubit in $\mathcal{G}_{\text{anc}}$ in the Pauli-Z basis.

Lemma 1 provides a mechanism to perform a measurement of an X logical operator $\bar{X}$; however, it is restricted to the special case where there is no other X logical operator within its support. In the general case, we may have another X logical operator $\bar{X}'$ supported entirely on a strict subset V' of physical qubits in $V[1]$. Following the construction as given above, we will make an unwanted measurement of $\bar{X}'$, which will result in an entirely different computation. As an example of this situation, consider the measurement of a two-logical-qubit operator $\bar{X}_1\bar{X}_2$, where $\bar{X}_1$ and $\bar{X}_2$ are canonical logical operators that do not intersect at any physical qubits. Following the procedure outlined above will give us separate measurements of $\bar{X}_1$ and $\bar{X}_2$.

We now generalize our construction of $\mathcal{G}_{\text{merged}}$ to address this general situation, which we illustrate in Fig. 3. To measure $\bar{X}_1\bar{X}_2$ without measuring the value of $\bar{X}_1$ or $\bar{X}_2$ individually, we must construct $\mathcal{G}_{\text{merged}}$ such that the separate logical operators are connected. First, we must construct the separate ancilla systems, $\mathcal{G}_{\text{anc.},1}$ and $\mathcal{G}_{\text{anc.},2}$, for $\bar{X}_1$ and $\bar{X}_2$. Let $c_1[k] \in C_1^T[k]$ and $c_2[k] \in C_2^T[k]$ be two arbitrary X generators for $1 \leq k \leq r$. We introduce a new physical qubit $a[k]$ for $1 \leq k \leq r$ and a new Z generator $z[j]$ for $1 \leq j \leq r$. There are the variable and check nodes in the highlighted region of Fig. 3. The nodes of $\mathcal{G}_{\text{merged}}$ will consist of all the nodes in $\mathcal{G} \cup \mathcal{G}_{\text{anc.},1} \cup \mathcal{G}_{\text{anc.},2}$ as well as the nodes $a[k]$ and $z[j]$. The edge set of $\mathcal{G}_{\text{merged}}$ will also contain all the edges in $\mathcal{G} \cup \mathcal{G}_{\text{anc.},1} \cup \mathcal{G}_{\text{anc.},2}$ as well as the edges connecting $\mathcal{G}_{\bar{X}_1}^T[2]$ and $\mathcal{G}_{\bar{X}_2}^T[2]$ to $\mathcal{G}_{\bar{X}_1}[1]$ and $\mathcal{G}_{\bar{X}_2}[1]$, respectively. We then add the following edges, which are the edges in the highlighted region of Fig. 3: $(a[j-1], z[j])$ and $(z[j], a[j])$ for $2 \leq j \leq r$, $(c_1[k], a[k])$ and $(a[k], c_2[k])$ for $1 \leq k \leq r$, and $(c_1^T[j], z[j])$ and $(z[j], c_2^T[j])$ for $2 \leq j \leq r$. Then, the product of all X checks in $\mathcal{G} \backslash \mathcal{G}_{\text{merged}}$ gives the measurement of $\bar{X}_1\bar{X}_2$ and the product of the X checks exclusively in $\mathcal{G}_{\text{anc},1}$ or $\mathcal{G}_{\text{anc},2}$ does not give measurements of $\bar{X}_1$ or $\bar{X}_2$ since they will have support on the physical qubits $a[k]$.

Before continuing, we offer some orienting remarks. First, if we choose to measure a logical operator supported at the boundary of the planar code defined using the lattice geometry presented in (*9*), we recover the lattice surgery construction given in the original work (*39*). The gates we obtain are also similar in spirit to those proposed in (*43*), where a surface code embedded on a torus is used as a resource to measure logical CSS operators of constant rate hyperbolic surface codes. We expect that the additional resources that will be needed to perform these logical operations will scale similarly with code

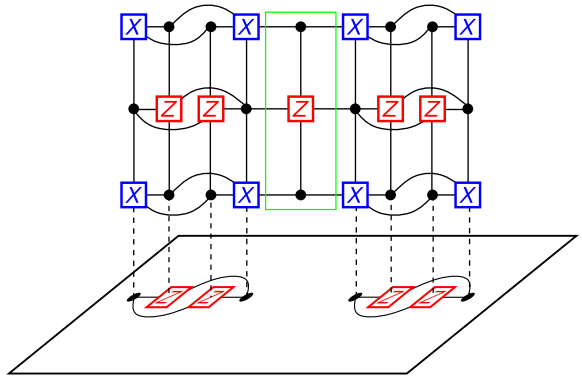

Fig. 3. Measurement of the logical operator $\bar{X}_1\bar{X}_2$. First, ancilla systems for the logical operators $\bar{X}_1$ and $\bar{X}_2$ are constructed as in Fig. 2. These ancilla systems are connected together as highlighted (green box). This ancilla system is then connected to the logical $\bar{X}_1\bar{X}_2$ as described previously. Observe that the product of the X stabilizer generators in the ancilla system gives the logical X_1X_2. Furthermore, if we take the product of the X stabilizers on the left ancilla system, we do not obtain X_1, since this product will include qubits in the highlighted (green) region. The same holds for the stabilizers in the right ancilla system. Hence, the product of the measurement results for these generators gives the measurement result for X_1X_2. After obtaining the measurement result, we can again measure the original stabilizers to return to the code space.

distance to our scheme, up to constant factors. Our techniques, however, do not require the logical operators to have a specific structure, and are thus more broadly applicable to quantum LDPC codes. Second, the reader familiar with hypergraph product codes (*58*) can check that $\mathcal{G}_{\text{anc}}$ is the hypergraph product of $\mathcal{G}_{\bar{X}}$ and the Tanner graph of a repetition code with r variables. From this observation, it is easy to verify that the stabilizers supported on the variables of G_{anc} commute.

Logical non-CSS measurement

To implement the entire logical Clifford group using logical Pauli measurements, we must also be able to measure non-CSS logical operators that are the product of Pauli-X and Pauli-Z measurements (*40*, *41*, *55*). We now demonstrate how our construction can be adapted to such measurements.

First, we demonstrate how to measure the logical operator $\bar{Y}$, which we illustrate in Fig. 4. Let $\bar{X}$ be a logical operator in $\mathcal{C}$ and let $\bar{Z}$ be the corresponding Z logical operator. The corresponding Y logical operator is given by $\bar{Y} = i\bar{X}\bar{Z}$, since $\bar{X}$ and $\bar{Z}$ must anticommute, any support of $\bar{X}$, and any support of $\bar{Z}$ must intersect at an odd number of qubits.

To measure $\bar{Y}$, we prepare an ancilla system by combining two ancilla systems, $\mathcal{G}_{\text{anc}\bar{X}}$ and $\mathcal{G}_{\text{anc}\bar{Z}}$, used to measure the CSS measurements $\bar{X}$ and $\bar{Z}$, respectively, using a fusion procedure as follows. We denote $A = V_{\bar{X}}[1] \cap V_{\bar{Z}}[1]$ as the set of qubits in the intersection of $\bar{X}$ and $\bar{Z}$. Their check operators are denoted $C_{\bar{X}}[k]$ and $C_{\bar{Z}}[k]$, respectively.

For each physical qubit $v \in A$, there are corresponding X generators $v_{\bar{X}}^T[k] \in C_{\bar{X}}^T[k]$ and Z generators $v_{\bar{Z}}^T[k] \in C_{\bar{Z}}^T[k]$ for $1 \le k \le r$. Furthermore, each $v[1]$ has corresponding qubits $v_{\bar{X}}[j]$ and $v_{\bar{Z}}[j]$, for $2 \le j \le r$, in $\mathcal{G}_{\text{anc}\bar{X}}$ and $\mathcal{G}_{\text{anc}\bar{Z}}$, respectively. We then form $\mathcal{G}_{\text{merged}}$ using all the nodes and edges in $\mathcal{G}_{\text{anc}\bar{X}}$ and $\mathcal{G}_{\text{anc}\bar{Z}}$, except for the generators $v_{\bar{X}}^T[k]$ and $v_{\bar{Z}}^T[k]$ for all vertices in A, which we replace with $y[k] = i v_{\bar{X}}^T[k] v_{\bar{Z}}^T[k]$. Merging the generators $v_{\bar{X}}^T[k]$ and $v_{\bar{Z}}^T[k]$ creates an extra degree of freedom, which we fix by introducing check nodes $g[j]$ for $2 \le j \le r$ and adding a Z edge $(v_{\bar{X}}[j], g[j])$ and an X edge $(v_{\bar{Z}}[j], g[j])$. The product of the X generators in $\mathcal{G}_{\text{anc}\bar{X}}$, the Z generators in $\mathcal{G}_{\text{anc}\bar{Z}}$, and the generators $y[k]$ allows us to infer the measurement of $\bar{Y}$. To check that these stabilizer generators commute, note that all the generators in $\mathcal{G}_{\text{anc}\bar{X}}$ and $\mathcal{G}_{\text{anc}\bar{Z}}$ commute since they act on different qubits, except $v_{\bar{X}}^T[1]$ and $v_{\bar{Z}}^T[1]$, which we rectified by combining them. This procedure can also be straightforwardly adapted to measure a logical operator of the form $\bar{X}_1\bar{Z}_2$ when the intersection of the supports of $\bar{X}_1$ and $\bar{Z}_2$ is not empty.

Last, we require measurement of non-CSS product operators such as $\bar{X}_1\bar{Z}_2$ when $\bar{X}_1$ and $\bar{Z}_2$ do not intersect at any physical qubits, as illustrated in Fig. 5. As in the general case for CSS measurements, we must first connect the logical operators to make a parity measurement. We calculate the ancilla systems $\mathcal{G}_{\text{anc1}}$ and $\mathcal{G}_{\text{anc2}}$ for $\bar{X}_1$ and $\bar{Z}_2$. Let $c_1[k] \in C_1^T[k]$ and $c_2[k] \in C_2^T[k]$ be X and Z generators, respectively, for $1 \le k \le r$. As we did for the measurement of $\bar{X}_1\bar{X}_2$, we extend out each ancilla system so that $\mathcal{G}_{\text{anc1}}$ has a boundary of X generators coming out of the column defined by $c_1[k]$ and $\mathcal{G}_{\text{anc2}}$ has a boundary of X generators coming out of the column defined by $c_2[k]$. As for the Y measurement, we then merge these boundaries by merging the corresponding X and Z generators at these boundaries and creating weight-two XZ generators between corresponding physical qubits at these boundaries. Then, if we take the product of the X generators in $\mathcal{G}_{\text{anc1}}$, the Z generators in $\mathcal{G}_{\text{anc2}}$, and the merged stabilizers, we obtain a measurement of $\bar{X}_1\bar{Z}_2$ without measuring $\bar{X}_1$ and $\bar{Z}_2$ separately.

Simultaneous measurement of commuting logical operators

Suppose we wish to simultaneously measure two commuting logical operators. If the supports of the logical operators do not intersect, then it is easy to see that we can make each measurement independently at the same time. If the supports do intersect but both logical operators are of the same type (all X or all Z), then we can still independently measure each logical operator using our construction. Last, if the two commuting logical operators are not of the same type (say a Pauli-X–type measurement $\bar{X}_1$ and a Pauli-Z–type measurement $\bar{Z}_2$) and intersect, then they must intersect at an even number of qubits. Let $\bar{L}_1$ and $\bar{L}_2$ be the two commuting Pauli operators and let a and b be two physical qubits in the intersection $V_1[1] \cap V_2[1]$ such that $\bar{L}_1$ and $\bar{L}_2$ act with different Pauli operators on a and similarly on b. Then, the generators $a^T[1] \in C_1^T[1]$ and $a'^T[1] \in C_2^T[1]$ will not commute. Similarly, the generators $b^T[1] \in C_1^T[1]$ and $b'^T[1] \in C_2^T[1]$ will not commute. To rectify this situation, we replace the generators $a^T[k]$ and $b^T[k]$ with $a^T[k]b^T[k]$ for $1 \le k \le r$ and create weight-two Z stabilizer generators acting on

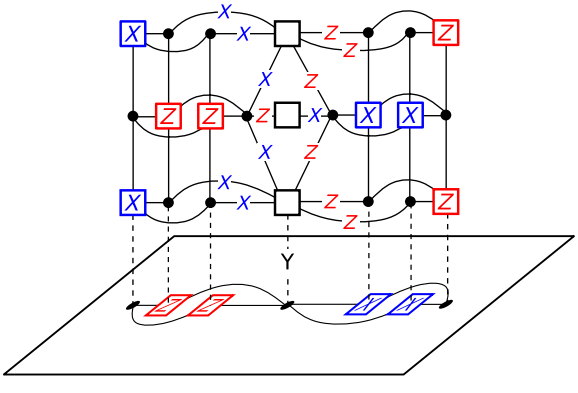

Fig. 4. Measurement of a logical $\bar{Y} = i\bar{X}\bar{Z}$ operator. This measurement closely follows that of logical $\bar{X}_1\bar{X}_2$ shown in Fig. 3, with the key difference being that ancilla systems for the logical operators X_1 and Z_2 are connected using non-CSS generators. Observe that the product of the X stabilizer generators on the left, the Z stabilizer generators on the right, and the mixed stabilizer generators in the dual layers gives the logical $\bar{Y}$.

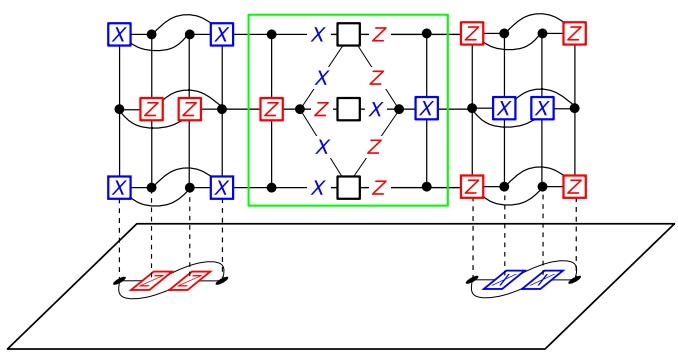

Fig. 5. Measurement of the logical operator $\bar{X}_1\bar{Z}_2$. This measurement closely follows that of logical $\bar{X}_1\bar{X}_2$ shown in Fig. 3, with the key difference being that ancilla systems for the logical operators X_1 and Z_2 are connected using non-CSS generators (highlighted green box). Observe that the product of the X stabilizer generators on the left, the Z stabilizer generators on the right, and the mixed stabilizer generators in the highlighted region gives the logical $\bar{X}_1\bar{Z}_2$.

$a[j]$ and $b[j]$ for $2 \leq j \leq r$ to fix the degrees of freedom created from merging the stabilizer generators. Similarly, we replace the generators $a'^T[k]$ and $b'^T[k]$ with $a'^T[k]b'^T[k]$ and create weight-two X stabilizer generators acting on $a'[j]$ and $b'[j]$. Since there are always an even number of intersections, noncommuting stabilizers can always be paired up.

Fault tolerance

Having presented our construction to perform logical Clifford gates via logical Pauli measurement, we now show that the merged code still has error-correcting capabilities and therefore retains its fault tolerance. First of all, it follows straightforwardly from our construction that the LDPC nature of the code remains intact throughout the deformation from $\mathcal{C}$ to $\mathcal{C}_{\text{merged}}$.

Lemma 2. Let $\widetilde{L}$ be a logical operator on a CSS quantum LDPC code $\mathcal{C}$. Let w be the maximum weight of a stabilizer generator in $\mathcal{C}$ and let q be the maximum number of stabilizer generators acting on a physical qubit in $\mathcal{C}$. Let w' and q' be the equivalent quantities for the code $\mathcal{C}_{\text{merged}}$ obtained after making a measurement of $\widetilde{L}$ in $\mathcal{C}$. Then, $w', q' \leq \max(w+3, q+3)$.

This fact is important for reducing complexity of the stabilizer measurement and for limiting the spread of errors throughout the deformation procedure.

Distance of $\mathcal{C}_{merged}$

To preserve error-correcting capabilities, it is important that our code maintains a nontrivial distance throughout the code deformation. We show that the code distance of $\mathcal{C}_{\text{merged}}$ is no less than that of $\mathcal{C}$. There are two main concerns regarding the distance of our construction. The first is whether the construction can reduce the weight of the logical operators. Through consideration of the structure of logical operators on the ancilla system, we demonstrate that logical operators terminating at the boundary layers of the ancilla systems must necessarily maintain their weight because these boundaries are kept well separated. Second, our construction can add new logical degrees of freedom to the code. As we are not interested in the state of these additional qubits, we refer to them as gauge qubits. This is consistent with terminology used in the context of subsystem codes [see (*59*) for an introduction]. We demonstrate that the logical operators associated to the gauge qubits (that we refer to as gauge operators) do not decrease the distance of the logical operators of interest. We call an operator that acts simultaneously on logical qubits and gauge qubits as a dressed logical operator. It is important to check that there are no dressed logical operators with weight smaller than d, as the existence of such an operator will mean that the code distance is decreased.

Our following arguments will also use the notion of "cleaning" (*60*). We say that a logical operator $\widetilde{L}$ is cleaned from some set of physical qubits A to some set of physical qubits B, if we can multiply $\widetilde{L}$ by an element S of the stabilizer group such that the equivalent logical operator $\widetilde{L}' = S\widetilde{L}$ has trivial support on A and nontrivial support on B.

The distance of $\mathcal{C}_{\text{merged}}$ will depend upon the value of r, where r was defined as the number of copies of $\mathcal{G}_{\widetilde{L}}^T$ in the ancilla system. Let d be the distance of $\mathcal{C}$. Consider the example of measuring the logical operator X_1X_2 of the surface code that encodes two logical qubits shown in Fig. 6. Then, letting $r = 1$ will substantially reduce the distance of the code, whereas letting $r = d$ will preserve the distance. In general, we will show that if we let $r = d$, where d is the distance of $\mathcal{C}$, then we can guarantee that the distance of the code during deformation does not drop below d.

Measuring a logical operator in $\mathcal{C}$ may introduce gauge degrees of freedom for which we must account. Suppose we measure a logical operator $\widetilde{L}$ in our code. We construct the ancilla system using the primal and dual graphs, $\mathcal{G}_{\widetilde{L}}[j]$ and $\mathcal{G}_{\widetilde{L}}^T[j]$, and merge them together with $\mathcal{C}$ to create $\mathcal{C}_{\text{merged}}$. If the number of independent check nodes in $\mathcal{G}_{\widetilde{L}}^T$ is less than or equal to the number of variable nodes, then $\mathcal{G}_{\widetilde{L}}^T$ will have nontrivial degrees of freedom and making the measurement of $\widetilde{L}$ will add extra gauge qubits to our code. In the case of an $\widetilde{X}$ measurement, we can choose a canonical set of Z gauge operators that are entirely contained in any dual layer of the ancilla system. Suppose that $r = 1$ and $\mathcal{G}_{\widetilde{L}}^T[1]$ contains n' variable nodes and m' check nodes. Then, if we interpret $\mathcal{G}_{\widetilde{L}}^T[1]$ as a classical code, it contains at least $n' - m'$ logical bits. This is a lower bound since some of the checks in $\mathcal{G}_{\widetilde{L}}^T[1]$ may be a linear combination of other checks. However, in our case, there are exactly $n' - m' + 1$ logical bits in $\mathcal{G}_{\widetilde{L}}^T[1]$. This is due to the fact that if there were a subset of check nodes in $\mathcal{G}_{\widetilde{L}}^T[1]$ whose product gives the identity, then the equivalent qubits in $\widetilde{L}$ would be a logical operator, and we have enforced the requirement that $\widetilde{L}$ contains no subsets that support a logical operator. We will call the logical operators of $\mathcal{G}_{\widetilde{L}}^T[1]$ cycle operators. However, we stress that the exact structure of these operators will not be important. Now, when we create $\mathcal{G}_{\text{merged}}$ with $r = 1$, we add n' qubits and m' stabilizer generators, and remove one logical qubit, thus adding at least $n' - m' + 1$ new gauge qubits. However, the same constraint for $\mathcal{G}_{\widetilde{L}}^T[1]$ carries to $\mathcal{G}_{\text{merged}}$, and hence, there are exactly $n' - m' + 1$ new gauge qubits, and we can choose the canonical set of Z gauge operators to be the logical operators of $\mathcal{G}_{\widetilde{L}}^T[1]$. These define all of the gauge degrees of freedom.

The cycle operators on $\mathcal{G}_{\widetilde{L}}^T$ are independent of the original logical operators on $\mathcal{C}$ and correspond to stabilizers of $\mathcal{C}$. Each qubit in a cycle in $\mathcal{G}_{\widetilde{L}}^T[1]$ has a corresponding stabilizer generator in $\mathcal{G}_{\widetilde{L}}[1]$. By applying these generators, we can clean the cycle operator so that it is supported entirely on $\mathcal{C}$. In this case, we can see that the cycle operator is thus equivalent to a product of old stabilizers in $\mathcal{G}_{\widetilde{L}}[1]$.

A cycle gauge operator in the bottom layer $\mathcal{G}_{\widetilde{L}}^T[1]$ can be cleaned to other layers $\mathcal{G}_{\widetilde{L}}^T[k]$ through the application of stabilizer generators

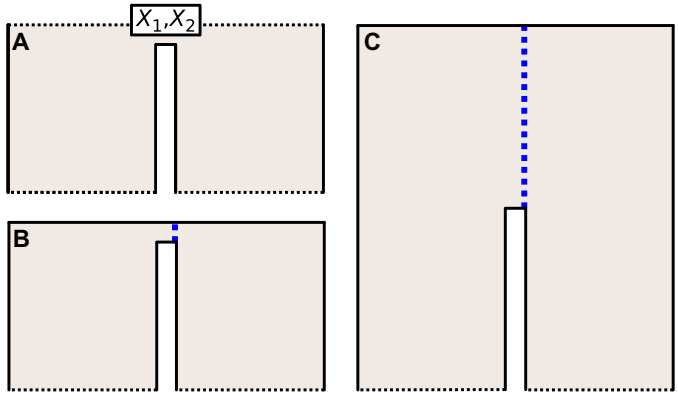

Fig. 6. Distance of $\mathcal{C}_{\text{merged}}$. (**A**) Suppose we have a code created by adjoining two surface code patches at the corner and we wish to measure the logical operator X_1X_2 along the top boundary. In (**B**), the ancilla system used for the measurement only has one layer, creating a low-weight logical between the top and bottom smooth boundaries. To mitigate this, we must use an ancilla with d layers, as in (**C**). This ensures that the distance of the code is preserved during the merge. It is worth noting that it may not always be the case that d layers are needed to preserve the distance, and for certain codes, it may suffice to only use one layer.

in $\mathcal{G}_{\widetilde{L}}[k]$, and so, equivalent cycles in different layers of $\mathcal{G}^T_{\widetilde{L}}[k]$ correspond to the same gauge qubit. We illustrate this in Fig. 7, where we show the measurement of an X logical operator. Here, we can construct a canonical set of logical operators of $\mathcal{G}^T_{\widetilde{L}}[1]$ with two cycles, and so, there are two gauge qubits. In this case, the cycles correspond to Z gauge operators. The following lemma tells us about the nature of the X gauge operators.

Lemma 3. Let $\mathcal{C}_{\text{merged}}$ be the code obtained after measuring an $\widetilde{X}$ logical operator. Let $\widetilde{Z}_g$ be a Z gauge operator in $\mathcal{C}_{\text{merged}}$. If a logical or gauge operator $\widetilde{O}$ anti-commutes with $\widetilde{Z}_g$, then it has weight at least r.

Proof. If $\widetilde{O}$ anti-commutes with $\widetilde{Z}_g$, then it must intersect every possible support of $\widetilde{Z}_g$. There are equivalent cycles all representing $\widetilde{Z}_g$ on each dual layer of $\mathcal{G}_{\text{anc}}$. These equivalent cycles are disjoint and so the X gauge operator must have support on every dual layer. Since there are r dual layers, $\widetilde{O}$ must have weight at least r, and furthermore must have weight at least one on each dual layer of $\mathcal{G}_{\text{anc}}$.

We can thus think of the $\widetilde{X}$ gauge operators as "strings" that travel from the top boundary $\mathcal{G}^T_{\widetilde{L}}[r]$ to the bottom of $\mathcal{G}^T_{\widetilde{L}}[1]$ and terminate in $\mathcal{C}$. We now show that $\mathcal{C}_{\text{merged}}$, when treated as a subsystem code, has distance at least d.

Theorem 1. Let $\mathcal{C} = [\![n, k, d]\!]$ be a quantum CSS LDPC code and let $\widetilde{L}$ be a logical operator in $\mathcal{C}$. Let $\mathcal{C}_{\text{merged}}$ be the code obtained after making a measurement of $\widetilde{L}$ in $\mathcal{C}$ using an ancilla system $\mathcal{C}_{\text{anc}}$. Then, $\mathcal{C}_{\text{merged}}$, when treated as a subsystem code, has distance $\geq d$ as long as the ancilla system has at least $2d - 1$ layers.

Proof. See "Materials and Methods."

Low-overhead fault-tolerant quantum computation

The previous section presents our construction for fault-tolerant measurements of logical Pauli operators. These measurements allow for logical Clifford gates as well as initialization and measurement in the logical Pauli basis.

Here, we consider what this construction means for performing low-overhead fault-tolerant quantum computing. We first consider the space and time overheads associated with our approach to fault tolerance, with the parallelism arising as a key quantity in determining these overheads. We then discuss the overheads associated with non-Clifford gates implemented via magic states and the decoders available for quantum LDPC codes.

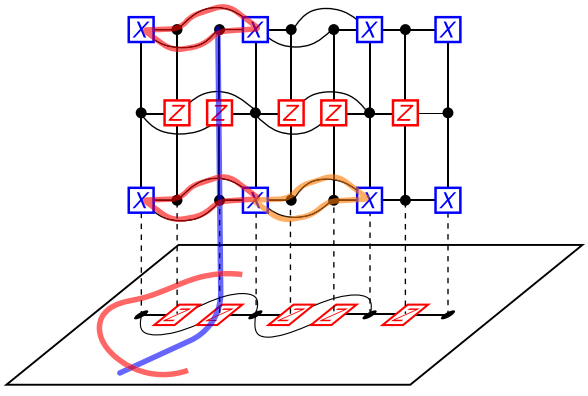

Fig. 7. Gauge operators of $\mathcal{C}_{\text{merged}}$. The red and orange cycles in the first layer of the ancilla system give a canonical set for the Z gauge operators. All red cycles correspond to the same gauge operator up to stabilizers. We can see that the Z gauge operators can be deformed so that they lie entirely in $\mathcal{C}$ and are equivalent to the product of the original Z stabilizers that are in $\mathcal{G}_{\widetilde{X}}$. The blue string is the X gauge operator corresponding to the red cycles. It must anti-commute with any representation of the conjugate Z gauge operator, and so, it must intersect with every red gauge operator in the graph. Since the red gauge operator can be cleaned to any dual layer of the ancilla system, the blue X logical must have support at least on every dual layer of the ancilla system. Since we use an ancilla system with d dual layers, this means that an X gauge operator must have weight at least d.

Ancilla system size

We now analyze the space overhead associated with making Pauli measurements using ancilla systems. Let $\widetilde{L}$ be a logical of weight w_L in a code $\mathcal{C}$ with distance d. Again, let q be the maximum number of stabilizer generators connected to a physical qubit in $\mathcal{C}$. Then, there are at most $qw_L/2$ check nodes in the graph $\mathcal{G}_{\widetilde{L}}$ and to leading order the number of physical qubits in the ancilla system used to measure $\widetilde{L}$ is

$$n_a = (1 + q/2)\, w_L r \tag{2}$$

In particular, when $r = d$, as in Theorem 1, this means that the ancilla system is proportional to $w_L d$. This makes our scheme particularly applicable to codes such as the hypergraph product code, with distance scaling of $O(\sqrt{n})$. Note that for the codes considered in Table 1, which are constructed from cyclic classical codes, we can choose a canonical set of logical operators so that $w_L = d$ for all of the logical operators in this set.

Parallelism

Let us now look at how the space and time overheads of our fault-tolerant scheme depend upon the weight of the logical parity measurements. We will see that there is a trade-off between the space and the time overheads and that certain codes allow us to improve these overheads when compared to a general LDPC code.

First, note that the techniques for making parity measurements between two logical qubits can easily be generalized to make parity measurements between multiple logical qubits. For instance, suppose we wish to measure $\widetilde{X}_1\widetilde{X}_2\widetilde{Z}_3$. This can be accomplished by first constructing the ancilla systems to measure $\widetilde{X}_1\widetilde{X}_2$ and $\widetilde{Z}_3$ separately. We can then connect these ancilla systems to create the ancilla system to measure $\widetilde{X}_1\widetilde{X}_2\widetilde{Z}_3$. We can then continue this construction to implement Pauli measurements of an arbitrary number of logical qubits. It is important that when we implement this construction, we do not connect all the ancilla systems at the same qubit, as otherwise there can potentially be a qubit that is in the support of many stabilizer generators, and $\mathcal{C}_{\text{merged}}$ will not be an LDPC code.

There is, however, a limit to how many logical qubits we can include in a parity measurement if we want the code rate to remain constant during the deformation. Suppose the code has $\Theta(n)$ logical qubits. Then, if we want to make a parity measurement of all X logical operators in the code, we would require $\Omega(nd)$ ancilla qubits. If $d > \Omega(1)$, then the number of ancillas we must add is $>\Omega(n)$, and hence, the resource cost of computation will diminish the savings made by choosing a finite rate code. From an asymptotic standpoint, this means that the number of logical qubits measured in a single parity measurement should be kept at a constant. This restriction will limit the number of logic gates that can be performed in parallel, and consequently, the time overhead of the computation is increased. In general, there is a trade-off between space and time overhead. If we increase the space overhead by adding many ancilla systems, then we can decrease the time overhead by allowing very large multilogical measurements.

We encapsulate these notions in the parallelism of our fault-tolerant quantum computing scheme, which we defined in the

summary of results. The parallelism is not inherent to a code and is instead chosen depending on the available space and time overhead. For instance, imagine a quantum computing platform for which the space overhead, i.e., required number of qubits, is the primary constraint. In this case, it will be advantageous to reduce the parallelism at the expense of time. In this space-constrained regime, the most natural choice is to use a parallelism of two, performing a sequence of weight-two logical Pauli measurements, which can be easily converted to an entangling gate.

However, certain codes, e.g., hypergraph product codes constructed from good classical LDPC codes, have an extra structure that allows us to achieve the same parallelism with a more modest resource. We can lay out a hypergraph product code on a two-dimensional grid so that the support of a canonical logical operator is contained entirely in one row or one column (*50*). Each row and column can support multiple logical qubits, and so, we can configure an ancilla system for an entire row or column that can measure multiple logical qubits. In the asymptotic limit, each row or column contains $\Theta(\sqrt{n})$ physical qubits and $O(\sqrt{n})$ logical operators, and so, for a $d = O(\sqrt{n})$ hypergraph product code, a single ancilla system of size $O(n)$ can be used to measure Pauli operators with logical weight $O(\sqrt{n})$. A naive scheme would have yielded an ancilla system of size $O(wd^2) = O(n^{3/2})$. As an explicit example, the ⟦7938,578,16⟧ code from Table 1 can be arranged on a two-dimensional grid such that each row or column contains 63 physical qubits and 17 logical operators of the same type. Each qubit is in the support of at most five stabilizer generators, and the code has distance 16. Hence, using Eq. 2, we find that we can construct an ancilla system for an entire row or column, and measure up to 17 logical operators, using around 3528 physical qubits. Furthermore, in such a layout, logical operators of different type only intersect if they correspond to the same logical qubit, and hence, if we assume that each logical qubit is only acted upon nontrivially by one logical operator in each round of error correction, then we can keep the stabilizer generator weight low by avoiding the construction for the simultaneous measurement of commuting logical operators.

If we wish to be able to measure all the logical operators on the hypergraph product code, then we require $O(n^{3/2})$ ancilla qubits. This reduces the rate and distance of the scheme to $O(n^{2/3})$ and $O(n^{\frac{1}{3}})$, respectively. Note that these code parameters cannot be achieved by a strictly local code in two dimensions because such codes necessarily obey the constraint $kd^2 = O(n)$ (*61*).

Magic states

With a scheme to perform Pauli measurements on quantum LDPC codes, we consider how this approach can be integrated into a broader scheme for performing universal fault-tolerant quantum computing with low overhead. There is one main requirement that we still need in addition to Pauli measurement, which is the distillation of magic states (*49*).

As we have already mentioned, Clifford gates alone are not sufficient to perform universal quantum computing, and in addition, we require a non-Clifford gate such as a T gate. A standard approach to fault-tolerant non-Clifford gates is through injection of magic states. For example, the T gate can be implemented using the magic state $|T\rangle$ and Clifford gates. Unless the code being used has transversal T gates, it is generally difficult to prepare these magic states in a fault-tolerant way. For this reason, magic state distillation, which prepares a small number of low error magic states from a larger number of noisy magic states, is often required for fault-tolerant architectures.

Here, we assume that all the data qubits are stored on one LDPC code block, such as a hypergraph product code, along with ancilla systems to make Pauli measurements on the data block. To distill magic states, we use a separate magic state factory and then inject the distilled magic states into our data block using the ancilla systems. We will consider previously designed magic state factories that use surface codes. As an example computation, suppose we want to perform 10^{10} T gates, the number required in (*6*), with a tolerance of 1%, and each noisy $|T\rangle$ has an error of 10^{-3}. Then, we require a distillation scheme with an output error rate of 10^{-12}. Such a magic state distillation scheme can be implemented using ∼15,000 physical qubits where we do not count the qubits needed for stabilizer measurements. Such a distillation scheme in conjunction with the ⟦7938,578,16⟧ data block would still render a favorable overhead when compared to a full surface code scheme. Of course, this would render a fairly slow scheme, since we would only be producing one magic state at a time. In Discussion, we consider the possibility of further reducing the overhead required for magic state distillation by using LDPC codes, allowing us to increase the frequency of magic state production while maintaining low overhead.

Decoders

There has been extensive work in designing efficient algorithms for decoding quantum LDPC codes. Many of these adapt known algorithms for classical LDPC codes, modifying them to deal with the nuances of quantum codes. One simple decoding algorithm for classical LDPC codes is the bit-flip algorithm. Leverrier *et al.* (*62*) adapted this decoder for quantum LDPC codes and designed the small-set flip decoder, which was then shown to be able to correct a linear number of errors on quantum LDPC codes with sufficient expansion properties (*30*). The predominant algorithm for classical LDPC codes is belief propagation (BP) decoding. BP works by envisioning the code as a graph and transmitting likelihoods between the nodes. While BP works well on classical codes, its performance is not as consistent on quantum codes due the degeneracy present in quantum LDPC codes, which results in split beliefs, where the decoder is not able to choose between two equivalent corrections. As a result, several modifications have been proposed to adapt BP decoding for quantum LDPC codes. In particular, Panteleev and Kalachev (*31*) combined BP with ordered statistic decoding to design a decoder that appears to perform well on a variety of quantum LDPC codes. Hastings (*63*) created an efficient greedy decoding algorithm that is able to correct a constant number of errors on the hyperbolic surface codes. There is thus a sufficient body of work showing that efficient decoding of quantum LDPC codes is possible. See also (*43*, *64*, *65*). For decoding, we will assume the use of one of the decoders outlined above. Given a good choice of code, these decoders appear to have performance comparable or even superior to that of the surface code.

DISCUSSION

We have shown that it is possible to use LDPC codes to achieve fault-tolerant quantum computing with overheads favorable to surface code schemes, even at reasonable scales. Our scheme uses a generalized form of lattice surgery, which when coupled with magic state distillation can implement universal quantum computing.

Our construction highlights parallelism as a key constraint in maintaining low overheads, and we note that this role of parallelism has also been identified in other schemes for quantum computing

using LDPC codes, in particular, the scheme by Gottesman (*29*). In (*29*), quantum computation proceeds by preparing encoded resource states and then using teleportation to execute quantum gates. A concatenation scheme is used to prepare arbitrary resource states into the LDPC code blocks, which has a nonnegligible overhead, and thus requires a fixed value of the parallelism to maintain a constant overhead. In contrast, our proposal allows for parallelism to increase with distance for certain code families, such as the hypergraph product codes. Furthermore, while the scheme in (*29*) has the required asymptotic behavior, concatenation often suffers from poor error thresholds as well as undesirable overheads in the practical regimes of interest. For our proposal, based on code deformation, we can expect similar thresholds to those obtained for LDPC codes used as quantum memories (*27*).

There are two main bottlenecks, in terms of overhead, in our scheme, and we now address how these may be overcome. One of the main contributions to the overhead in our scheme is magic state distillation. Although we argued that it is possible to use surface code schemes while still maintaining an overhead advantage, it is generally necessary to use several magic state factories to achieve a high enough rate of magic state production. There is a potential for LDPC codes to be used to improve the overhead of magic state distillation schemes. We can look to some ideas from surface codes. Litinski (*40*) presents several surface code distillation schemes that achieve low overheads using the following idea. Circuits for magic state distillation using codes with transversal T gates can be rewritten to use only Z measurements. This means that Z logical errors are far less destructive than X logical errors, and hence, it is sufficient to use surface codes with low d_z. It is very straightforward to generalize this to LDPC codes such as the hypergraph product code. By taking the product of a high-rate, low-distance classical LDPC code and a high-distance repetition code, we can construct LDPC codes with low d_z, high d_x, and higher encoding rates than the surface code. While this will not improve the overhead for constructing a single magic state factory, it may allow us to fit several identical factories in the same space as one surface code factory.

One advantage that such a magic state scheme would offer is that the hypergraph product code used is only nonlocal in one dimension. This may be a desirable simplification in some quantum systems. In general, a promising line of research is to consider how much we can gain from LDPC codes while restricting the level of nonlocality allowed. It has been shown that there is a connection between the distance of the code and the connectivity (*26*, *46*), and that limiting the nonlocality will limit the attainable distances (*66*). This may, however, be a sacrifice that is necessary to see quantum LDPC codes physically implemented.

The other main contribution to the overhead comes from the ancilla systems used to make logical measurements. As we established, to maintain the distance of our code during the code deformation, we require that the ancilla systems have a height of at least d. Lowering this overhead while maintaining the required distance may be achieved using a technique established in (*48*). In this reference, Hastings proposes increasing the expansion of the graph $\mathcal{G}_{\tilde{L}}$, and as a result, the number of layers in the ancilla system would only need to be constant in size, as opposed to scaling with d. To implement this approach, we would require a deterministic method for creating graphs with sufficient expansion. While constructions for Ramanujan graphs with high expansion exist, further work is needed to see how these can be integrated into our scheme. We also note that it is unclear that this proposal will offer a substantial overhead improvement at the scales that we have considered, and instead give improvements to the resource cost of quantum computing at larger scales. Furthermore, it will be worthwhile to investigate the time overhead of fault-tolerant gates. The discovery of ancilla systems that enable code deformations by single-shot error correction (*30*, *67*, *68*) would permit measurement-based gates in constant time, leaving open the tantalizing prospect of fault-tolerant quantum computing with constant-space and constant-time overhead.

MATERIALS AND METHODS

We now restate and prove Theorem 1.

Theorem 1. Let $\mathcal{C} = [\![n, k, d]\!]$ be a quantum CSS LDPC code and let $\tilde{L}$ be a logical operator in $\mathcal{C}$. Let $\mathcal{C}_{\text{merged}}$ be the code obtained after making a measurement of $\tilde{L}$ in $\mathcal{C}$ using an ancilla system $\mathcal{C}_{\text{anc}}$. Then, $\mathcal{C}_{\text{merged}}$, when treated as a subsystem code, has distance $\geq d$ as long as the ancilla system has at least $2d - 1$ layers.

Proof. We prove the theorem for the simple case where $\tilde{L}$ is an X logical operator. Let us also assume without loss of generality that $\tilde{L}$ does not contain any X-type stabilizers as a subset of $V_{\tilde{L}}[1]$. The proof of more general logical parity measurements can be obtained with the methods we develop here with the most simple case.

Let $\tilde{Z}$ be a Z logical operator contained entirely in $\mathcal{C}$. First, note that the X stabilizers in $\mathcal{C}$ are left unchanged in $\mathcal{C}_{\text{merged}}$ and so $\tilde{Z}$ must still have weight at least d on the physical qubits in $\mathcal{G}_{\text{merged}} \backslash \mathcal{G}_{\text{anc}}$. We now need to show that the application of a gauge operator and an arbitrary stabilizer does not reduce the weight of $\tilde{Z}$ below d. We are only interested in Z-type stabilizers and gauge operators, as X-type stabilizers and gauge operators will not reduce the weight of a Z logical operator. Let us consider what happens when we apply the Z generators $C[1] \subset \mathcal{G}_{\tilde{L}}[1]$. These generators are identical to the original generators from $\mathcal{C}$ with the addition of one extra physical qubit from $V^T[1] \subset \mathcal{G}_{\tilde{L}}^T[1]$ in each generator. Hence, after applying these generators, $\tilde{Z}$ will still have weight at least d on the physical qubits of $\mathcal{G}_{\text{merged}} \backslash \mathcal{G}_{\text{anc}}$. Stabilizer generators $C[k]$ for $k \geq 2$

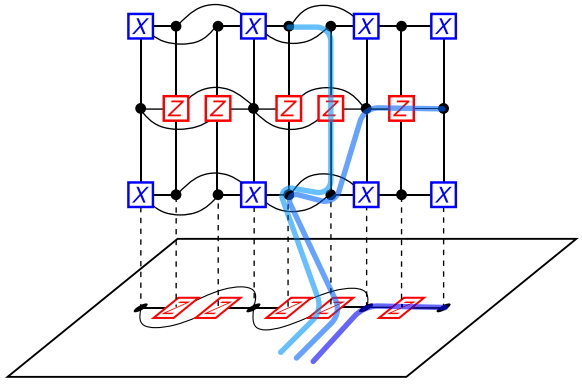

Fig. 8. Equivalent logical operators. The three curves here all represent equivalent X logical operators. The dark curve is an X logical operator on the original code $\mathcal{C}$. The other two logical operators are obtained by successive application of X stabilizer generators in the dual layers of the ancilla system. In particular, after each application of X stabilizer generators in a dual layer, the logical operator has support on the same dual layer since any subset of X stabilizer generators in a dual layer is independent.

can only increase the weight of the logical operator beyond d, as they have no common support with $\mathcal{G}_{\text{merged}} \backslash \mathcal{G}_{\text{anc}}$.

The Z gauge operators when we measure an X logical operator are cycles in $\mathcal{G}_{\widetilde{Z}}^{T}[1]$. As discussed above, these cycles can be cleaned entirely into $\mathcal{C}$. These gauge operators are stabilizers of $\mathcal{C}$, and since the support of $\widetilde{Z}$ in $\mathcal{C}$ remains the same in $\mathcal{C}_{\text{merged}}$, the application of a stabilizer of $\mathcal{C}$ will not reduce the weight of $\widetilde{Z}$ below d since the weight of a logical operator is lower bounded by d even with arbitrary stabilizer generators applied.

Let us now check that we do not change the weight of any X logical operators by adding the ancilla system. Let $\widetilde{X} \neq \widetilde{L}$ be an X logical operator contained entirely in $\mathcal{C}$. We must show that the weight of $\widetilde{X}$ remains above d with arbitrary application of X-type stabilizers and gauge operators. Let $\widetilde{X}_g$ be an arbitrary, nontrivial X gauge operator and let $\widetilde{Z}_g$ be any Z gauge operator that anti-commutes with $\widetilde{X}_g$. Then, $\widetilde{X}\,\widetilde{X}_g$ anti-commutes with Z_g and so, by Lemma 3, $\widetilde{X}\,\widetilde{X}_g$ has weight at least d, even with application of arbitrary stabilizers.

The last case to consider is when no gauge operators are applied, and we apply arbitrary stabilizers. This is only relevant if $\widetilde{X}$ intersects with $\widetilde{L}$ at some physical qubits. In this case, we can apply an X generator in the dual-layer $\mathcal{G}_{\widetilde{L}}^{T}[1]$, which will clean $\widetilde{L}$ from the qubits at the intersection of $\widetilde{L}$ and $\widetilde{X}$ onto the equivalent qubits in the first primal layer of the ancilla system $\mathcal{G}_{\widetilde{L}}[2]$.

Doing this will also create nontrivial support on the first dual layer $\mathcal{G}_{\widetilde{L}}^{T}[1]$ (see Fig. 8). To see why this is the case, suppose there is a subset of X generators in $C_{\widetilde{L}}^{T}[1]$ such that their product gives trivial support on $V_{\widetilde{L}}^{T}[1]$. That is, there is a subset A^T of $C_{\widetilde{L}}^{T}[1]$ such that each qubit in $V_{\widetilde{L}}^{T}[1]$ is connected to an even number of generators in A^T. This implies that there is a subset A of $V_{\widetilde{L}}[1]$ such that each check node in $C_{\widetilde{L}}[1]$ is connected to an even number qubits in A. This means that if we apply X to each physical qubit in A to form the operator $\widetilde{X}_A$, then $\widetilde{X}_a$ commutes with each Z generator in $C_{\widetilde{L}}[1]$, and hence, $\widetilde{X}_A$ is an X logical operator in $\mathcal{C}$, but we have assumed that $\widetilde{L}$ contains no logical operators as a subset, and hence, there is no such set A^T that gives trivial support on $V_{\widetilde{L}}^{T}[1]$. Note that in the general case we can have subsets of $V_{\widetilde{L}}[1]$ that support a logical operator; however, applying all the equivalent stabilizers in the first dual layer of lG_{anc} will create support on the qubit connecting the two ancilla systems.

The above argument tells us that we can clean $\widetilde{L}$ from qubits in $V_{\widetilde{L}}[1]$, replacing them with qubits in $V_{\widetilde{L}}[2]$ while creating nontrivial support on $V_{\widetilde{L}}^{T}[1]$. Now, we can continue this process to clean $\widetilde{L}$ from qubits in $V_{\widetilde{L}}[2]$ to qubits in $V_{\widetilde{L}}[3]$ in the support of $\widetilde{L}$ while creating nontrivial support on $V^T[2]$ and so on. For each qubit in the intersection of $\widetilde{X}$ and $\widetilde{L}$ that we cleaned from $\widetilde{L}$, there will always be an equivalent qubit in the primal layer $\mathcal{G}_{\widetilde{L}}[j]$, until we reach the top boundary, at which point there will be support on at least one physical qubit in $V_{\widetilde{L}}[j]$, and so, the weight of $\widetilde{X}$ will always be at least d.

REFERENCES AND NOTES

1. F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. M. Ewen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, J. M. Martinis, Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
2. V. von Burg, G. H. Low, T. Häner, D. S. Steiger, M. Reiher, M. Roetteler, M. Troyer, Quantum computing enhanced computational catalysis. *Phys. Rev. Res.* **3**, 033055 (2021).
3. J. Lee, D. W. Berry, C. Gidney, W. J. Huggins, J. R. McClean, N. Wiebe, R. Babbush, Even more efficient quantum computations of chemistry through tensor hypercontraction. *PRX Quantum* **2**, 030305 (2021).
4. I. H. Kim, Y.-H. Liu, S. Pallister, W. Pol, S. Roberts, E. Lee, Fault-tolerant resource estimate for quantum chemical simulations: Case study on Li-ion battery electrolyte molecules. *Phys. Rev. Res.* **4**, 023019 (2022).
5. A. G. Fowler, M. Mariantoni, J. M. Martinis, A. N. Cleland, Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A* **86**, 032324 (2012).
6. C. Gidney, M. Ekerå, How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum* **5**, 433 (2021).
7. E. Campbell, A. Khurana, A. Montanaro, Applying quantum algorithms to constraint satisfaction problems. *Quantum* **3**, 167 (2019).
8. Y. R. Sanders, D. W. Berry, P. C. S. Costa, L. W. Tessler, N. Wiebe, C. Gidney, H. Neven, R. Babbush, Compilation of fault-tolerant quantum heuristics for combinatorial optimization. *PRX Quantum* **1**, 020312 (2020).
9. E. Dennis, A. Kitaev, A. Landahl, J. Preskill, Topological quantum memory. *J. Math. Phys.* **43**, 4452–4505 (2002).
10. H. Bombin, M. A. Martin-Delgado, Topological quantum distillation. *Phys. Rev. Lett.* **97**, 180501 (2006).
11. D. Bacon, Operator quantum error-correcting subsystems for self-correcting quantum memories. *Phys. Rev. A* **73**, 012340 (2006).
12. J. P. Bonilla Ataides, D. K. Tuckett, S. D. Bartlett, S. T. Flammia, B. J. Brown, The XZZX surface code. *Nat. Commun.* **12**, 2172 (2021).
13. R. Raussendorf, J. Harrington, Fault-tolerant quantum computation with high threshold in two dimensions. *Phys. Rev. Lett.* **98**, 190504 (2007).
14. P. Magnard, S. Storz, P. Kurpiers, J. Schär, F. Marxer, J. Lütolf, T. Walter, J.-C. Besse, M. Gabureac, K. Reuer, A. Akin, B. Royer, A. Blais, A. Wallraff, Microwave quantum link between superconducting circuits housed in spatially separated cryogenic systems. *Phys. Rev. Lett.* **125**, 260502 (2020).
15. A. R. Mills, D. M. Zajac, M. J. Gullans, F. J. Schupp, T. M. Hazard, J. R. Petta, Shuttling a single charge across a one-dimensional array of silicon quantum dots. *Nat. Commun.* **10**, 1063 (2019).
16. F. Borjans, X. G. Croot, X. Mi, M. J. Gullans, J. R. Petta, Resonant microwave-mediated interactions between distant electron spins. *Nature* **577**, 195–198 (2020).
17. J. Yoneda, W. Huang, M. Feng, C. H. Yang, K. W. Chan, T. Tanttu, W. Gilbert, R. C. C. Leon, F. E. Hudson, K. M. Itoh, A. Morello, S. D. Bartlett, A. Laucht, A. Saraiva, A. S. Dzurak, Coherent spin qubit transport in silicon. *Nat. Commun.* **12**, 4114 (2021).
18. S. Debnath, N. M. Linke, C. Figgatt, K. A. Landsman, K. Wright, C. Monroe, Demonstration of a small programmable quantum computer with atomic qubits. *Nature* **536**, 63–66 (2016).
19. J. M. Pino, J. M. Dreiling, C. Figgatt, J. P. Gaebler, S. A. Moses, M. S. Allman, C. H. Baldwin, M. Foss-Feig, D. Hayes, K. Mayer, C. Ryan-Anderson, B. Neyenhuis, Demonstration of the trapped-ion quantum ccd computer architecture. *Nature* **592**, 209–213 (2021).
20. H. Bombin, I. H. Kim, D. Litinski, N. Nickerson, M. Pant, F. Pastawski, S. Roberts, T. Rudolph, Interleaving: Modular architectures for fault-tolerant photonic quantum computing. arXiv:2103.08612 [quant-ph] (15 March 2021).
21. S. Bartolucci, P. Birchall, H. Bombin, H. Cable, C. Dawson, M. Gimeno-Segovia, E. Johnston, K. Kieling, N. Nickerson, M. Pant, F. Pastawski, T. Rudolph, C. Sparrow, Fusion-based quantum computation. arXiv:2101.09310 [quant-ph] (22 January 2021).
22. S. D. Barrett, P. Kok, Efficient high-fidelity quantum computation using matter qubits and linear optics. *Phys. Rev. A* **71**, 060310 (2005).
23. N. H. Nickerson, Y. Li, S. C. Benjamin, Topological quantum computing with a very noisy network and local error rates approaching one percent. *Nat. Commun.* **4**, 1756 (2013).
24. N. Kalb, A. A. Reiserer, P. C. Humphreys, J. J. W. Bakermans, S. J. Kamerling, N. H. Nickerson, S. C. Benjamin, D. J. Twitchen, M. Markham, R. Hanson, Entanglement distillation between solid-state quantum network nodes. *Science* **356**, 928–932 (2017).
25. L. J. Stephenson, D. P. Nadlinger, B. C. Nichol, S. An, P. Drmota, T. G. Ballance, K. Thirumalai, J. F. Goodwin, D. M. Lucas, C. J. Ballance, High-rate, high-fidelity entanglement of qubits across an elementary quantum network. *Phys. Rev. Lett.* **124**, 110501 (2020).
26. N. Delfosse, M. E. Beverland, M. A. Tremblay, Bounds on stabilizer measurement circuits and obstructions to local implementations of quantum LDPC codes. arXiv:2109.14599 [quant-ph] (29 September 2021).
27. M. A. Tremblay, N. Delfosse, M. E. Beverland, Constant-overhead quantum error correction with thin planar connectivity. arXiv:2109.14609 [quant-ph] (29 September 2021).

28. N. P. Breuckmann, J. N. Eberhardt, Quantum low-density parity-check codes. *Quantum* **2**, 040101 (2021a).
29. D. Gottesman, Fault-tolerant quantum computation with constant overhead. *Quantum Inf. Comput.* **14**, 1338–1372 (2014).
30. O. Fawzi, A. Grospellier, A. Leverrier, Constant overhead quantum fault tolerance with quantum expander codes. *Commun. ACM* **64**, 106–114 (2021).
31. P. Panteleev, G. Kalachev, Degenerate quantum LDPC codes with good finite length performance. *Quantum* **5**, 585 (2021).
32. J. Roffe, D. R. White, S. Burton, E. Campbell, Decoding across the quantum low-density parity-check code landscape. *Phys. Rev. Res.* **2**, 043423 (2020).
33. A. Grospellier, L. Grouès, A. Krishna, A. Leverrier, Combining hard and soft decoders for hypergraph product codes. *Quantum* **5**, 432 (2021).
34. M. B. Hastings, J. Haah, R. O'Donnell, Fiber bundle codes: Breaking the $n^{1/2}$ polylog(n) barrier for quantum LDPC codes, in *Proceedings of the 53rd Annual ACM SIGACT symposium on theory of computing. Association for computing machinery*, New York, NY, USA, 2021, pp. 1276–1288; https://doi.org/10.1145/3406325.3451005.
35. P. Panteleev, G. Kalachev, Quantum ldpc codes with almost linear minimum distance. *IEEE Trans. Info. Theory* **68**, 213–229 (2022).
36. N. P. Breuckmann, J. N. Eberhardt, Balanced product quantum codes. *IEEE Trans. Info. Theory* **67**, 6653–6674 (2021b).
37. E. Knill, Scalable quantum computation in the presence of large detected-error rates. arXiv:quant-ph/0312190 (23 December 2003).
38. Y.-C. Zheng, C.-Y. Lai, T. A. Brun, L.-C. Kwek, Constant depth fault-tolerant clifford circuits for multi-qubit large block codes. *Quantum Sci. Technol.* **5**, 045007 (2020).
39. C. Horsman, A. G. Fowler, S. Devitt, R. Van Meter, Surface code quantum computing by lattice surgery. *New J. Phys.* **14**, 123011 (2012).
40. D. Litinski, A game of surface codes: Large-scale quantum computing with lattice surgery. *Quantum* **3**, 128 (2019).
41. B. J. Brown, K. Laubscher, M. S. Kesselring, J. R. Wootton, Poking holes and cutting corners to achieve clifford gates with the surface code. *Phys. Rev. X* **7**, 021029 (2017).
42. C. Vuillot, L. Lao, B. Criger, C. G. Almudéver, K. Bertels, B. M. Terhal, Code deformation and lattice surgery are gauge fixing. *New J. Phys.* **21**, 033028 (2019).
43. N. P. Breuckmann, C. Vuillot, E. Campbell, A. Krishna, B. M. Terhal, Hyperbolic and semi-hyperbolic surface codes for quantum storage. *Quantum Sci. Technol.* **2**, 035007 (2017).
44. A. Lavasani, M. Barkeshli, Low overhead clifford gates from joint measurements in surface, color, and hyperbolic codes. *Phys. Rev. A* **98**, 052319 (2018).
45. A. Krishna, D. Poulin, Topological wormholes: Nonlocal defects on the toric code. *Phys. Rev. Res.* **2**, 023116 (2020).
46. A. Krishna, D. Poulin, Fault-tolerant gates on hypergraph product codes. *Phys. Rev. X* **11**, 011023 (2021).
47. M. B. Hastings, Weight reduction for quantum codes. arXiv:1611.03790 [quant-ph] (11 November 2016).
48. M. B. Hastings, On quantum weight reduction. arXiv:2102.10030 [quant-ph] (19 February 2021).
49. S. Bravyi, A. Kitaev, Universal quantum computation with ideal clifford gates and noisy ancillas. *Phys. Rev. A* **71**, 022316 (2005).
50. A. A. Kovalev, L. P. Pryadko, Quantum kronecker sum-product low-density parity-check codes with finite rate. *Phys. Rev. A* **88**, 012311 (2013).
51. R. Babbush, C. Gidney, D. W. Berry, N. Wiebe, J. McClean, A. Paler, A. Fowler, H. Neven, Encoding electronic spectra in quantum circuits with linear T complexity. *Phys. Rev. X* **8**, 041015 (2018).
52. É. Gouzien, N. Sangouard, Factoring 2048-bit rsa integers in 177 days with 13 436 qubits and a multimode memory. *Phys. Rev. Lett.* **127**, 140503 (2021).
53. J. Ramette, J. Sinclair, Z. Vendeiro, A. Rudelis, M. Cetina, V. Vuletić, Any-to-any connected cavity-mediated architecture for quantum computing with trapped ions or rydberg arrays. *PRX Quantum* **3**, 010344 (2022).
54. O. Higgott, N. P. Breuckmann, Subsystem codes with high thresholds by gauge fixing and reduced qubit overhead. *Phys. Rev. X* **11**, 031039 (2021).
55. C. Chamberland, E. T. Campbell, Universal quantum computing with twist-free and temporally encoded lattice surgery. *PRX Quantum* **3**, 010331 (2022).
56. D. Wecker, B. Bauer, B. K. Clark, M. B. Hastings, M. Troyer, Gate-count estimates for performing quantum chemistry on small quantum computers. *Phys. Rev. A* **90**, 022305 (2014).
57. G. H. Low, I. L. Chuang, Hamiltonian simulation by qubitization. *Quantum* **3**, 163 (2019).
58. J.-P. Tillich, G. Zémor, Quantum LDPC codes with positive rate and minimum distance proportional to the square root of the blocklength. *IEEE Trans. Info. Theory* **60**, 1193–1202 (2014).
59. D. Poulin, Stabilizer formalism for operator quantum error correction. *Phys. Rev. Lett.* **95**, 230504 (2005).
60. S. Bravyi, B. Terhal, A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes. *New J. Phys.* **11**, 043029 (2009).
61. S. Bravyi, D. Poulin, B. Terhal, Tradeoffs for reliable quantum information storage in 2d systems. *Phys. Rev. Lett.* **104**, 050503 (2010).
62. A. Leverrier, J.-P. Tillich, G. Zémor, Quantum expander codes, in *2015 IEEE 56th Annual Symposium on Foundations of Computer Science* (IEEE, 2015) pp. 810–824.
63. M. B. Hastings, Decoding in hyperbolic spaces: Quantum ldpc codes with linear rate and efficient error correction. *Quantum Inf. Comput.* **14**, 1187–1202 (2014).
64. N. P. Breuckmann, B. M. Terhal, Constructions and noise threshold of hyperbolic surface codes. *IEEE Trans. Inf. Theory* **62**, 3731–3744 (2016).
65. N. P. Breuckmann, V. Londe, Single-shot decoding of linear rate ldpc quantum codes with high performance. *IEEE Trans. Inf. Theory* **68**, 272–286 (2022).
66. N. Baspin, A. Krishna, Connectivity constrains quantum codes. arXiv:2106.00765 [quant-ph] (1 June 2021).
67. H. Bombín, Single-shot fault-tolerant quantum error correction. *Phys. Rev. X* **5**, 031043 (2015).
68. N. Delfosse, B. W. Reichardt, Short Shor-style syndrome sequences. arXiv:2008.05051 [quant-ph] (12 August 2020).

Acknowledgments: We acknowledge D. Poulin for discussions on using quantum LDPC codes for computation. We thank A. Saraiva, M. Vasmer, and P. Webster for discussions and comments on the manuscript. **Funding:** This work was supported by the Australian Research Council via the Centre of Excellence in Engineered Quantum Systems (EQUS) project number CE170100009 and by the ARO under grant number W911NF-21-1-0007. **Author contributions:** All authors conceived the methodology, derived the main results, and contributed to the writing of the manuscript. **Competing interests:** The authors declare that they have no competing interests. **Data and materials availability:** All data needed to evaluate the conclusions in the paper are present in the paper and/or the Supplementary Materials.

Submitted 8 November 2021
Accepted 6 April 2022
Published 20 May 2022
10.1126/sciadv.abn1717