Notes on Quantum Error Correction

Cheng-Yu Liu
316497z@gmail.com

Abstract

Quantum error correction (QEC) protects against both coherent and incoherent errors by encoding logical quantum information into a larger Hilbert space of physical qubits, where errors can be detected and corrected without disturbing the logical state. In this note, we provide short introductions to key concepts in QEC, including the stabilizer formalism, code concatenation, and subsystem codes, with illustrative examples. Finally, using Python and the Qiskit framework, we reproduce a simulation of the Bacon–Shor code on a trapped-ion chain as presented in [6], and discuss the results in relation to the error models analyzed in that work.

Contents

1 Introduction

Quantum algorithms potentially speed up calculations exponentially but at the same time require thousands of gate operations. High-fidelity gates can be realized in experiments [16], but accumulation of errors can be drastic in even larger systems as required in many applications [2]. Assuming a simple model where each gate has independent stochastic errors with fidelity f, the probability that a circuit of m gates has no errors is f^m, which can be around 50% with 140 consecutive gates with fidelity $f = 99.5\%$. The probability of error-free execution drops rapidly as the circuit depth increases. For practical algorithms requiring thousands or millions of operations, such raw physical error rates are clearly insufficient. This motivates the use of quantum error correction (QEC), in which logical qubits are redundantly encoded into multiple physical qubits to actively detect and correct errors. The threshold theorem ensures that if the physical error rate is below a certain threshold value, then logical errors can be suppressed arbitrarily by increasing the code size. For surface codes, one of the most studied QEC schemes, this threshold is on the order of 1% and follows the relation [4]:

$$P_L \sim \left(\frac{P}{P_{\text{thre}}}\right)^{\frac{d+1}{2}}$$

to suppress the logical error rate P_L per stage, where d is the code distance, P is the physical error rate per stage, and P_{thre} is the error threshold. This relation also shows how quantum error correction mitigates logical errors P_L by mapping physical operations into logical operations.

Suppose one physical qubit in the Shor code suffers a small coherent error:

$$U_X(\theta) = e^{-i\frac{\theta}{2}X} = \cos\left(\tfrac{\theta}{2}\right) I - i\sin\left(\tfrac{\theta}{2}\right) X.$$

For small θ, this can be approximated as

$$U_X(\theta) \approx I - i\tfrac{\theta}{2}X.$$

Action on an encoded state

Let $|\psi_L\rangle$ denote the encoded logical state. After the error, the state becomes

$$U_X(\theta)|\psi_L\rangle = \cos\!\left(\tfrac{\theta}{2}\right)|\psi_L\rangle - i\sin\!\left(\tfrac{\theta}{2}\right)X_j|\psi_L\rangle,$$

where X_j denotes a bit-flip on physical qubit j.

Thus, the corrupted state is a superposition of:

- the "no error" branch with amplitude $\cos(\frac{\theta}{2})$, and
- the "error on qubit j" branch with amplitude $-i\sin(\frac{\theta}{2})$.

In this article, we will first highlight the importance of quantum error correction. We will then examine the work based mainly on [6], aiming to provide explanations, reproduce key results, and draw inspiration from their findings.

2 Stabilizer Formalism

A well-developed framework used to efficiently characterize quantum error correction codes is the *stabilizer formalism.* It describes a code space as the simultaneous $+1$ eigenspace of a set of commuting Pauli operators, called *stabilizers.* Errors are detected by measuring these stabilizers: if an error anticommutes with a stabilizer, the corresponding measurement outcome flips, providing an error syndrome.

Stabilizer code. A *stabilizer code* on n qubits is specified by an abelian subgroup $\mathcal{S} \subseteq \mathcal{P}_n$ (the n-qubit Pauli group) that does not contain $-I$. The *codespace* $\mathcal{C}$ is the joint $+1$ eigenspace of all elements of $\mathcal{S}$:

$$\mathcal{C} = \left\{\, |\psi\rangle \in (\mathbb{C}^2)^{\otimes n} \;:\; S\,|\psi\rangle = |\psi\rangle \;\; \forall\, S \in \mathcal{S} \,\right\}.$$

Dimension. If the code encodes k logical qubits into n physical qubits (an $[[n,k,d]]$ code), then with $n-k$ independent stabilizer generators we have

$$\dim \mathcal{C} = \frac{2^n}{2^{\,n-k}} = 2^k.$$

Logical basis inside the codespace. Because $\dim \mathcal{C} = 2^k$, we may choose an orthonormal *logical basis* of $\mathcal{C}$,

$$\mathcal{B}_L = \left\{|x_L\rangle : x \in \{0,1\}^k\right\},$$

such that each basis vector lies in the codespace (hence is stabilized):

$$S\,|x_L\rangle = |x_L\rangle \qquad \forall\, S \in \mathcal{S},\; \forall\, x \in \{0,1\}^k.$$

Arbitrary logical state (spanning by the logical basis). Any generic codespace vector $|\psi\rangle \in \mathcal{C}$ can be expressed *in the logical basis* as

$$|\psi\rangle \equiv |\psi_L\rangle = \sum_{x \in \{0,1\}^k} \alpha_x\,|x_L\rangle, \qquad \sum_x |\alpha_x|^2 = 1.$$

Thus the logical basis $\{|x_L\rangle\}$ spans the same subspace that was defined abstractly by the condition $S|\psi\rangle = |\psi\rangle$.

Expansion in the physical (computational) basis. Each logical basis vector is itself a vector in the n-qubit Hilbert space and typically expands as a superposition of computational basis states:

$$|x_L\rangle = \sum_{i=0}^{2^n-1} c_i^{(x)} |i\rangle,$$

with coefficients $\{c_i^{(x)}\}$ constrained by the stabilizer conditions $S|x_L\rangle = |x_L\rangle$ for all $S \in \mathcal{S}$. These constraints select which computational-basis components may appear and with what relative phases or amplitudes.

Remark (stabilizers vs. logical operators). Stabilizers act *trivially* on every codespace vector (eigenvalue $+1$) and thus define $\mathcal{C}$. By contrast, *logical operators* act *nontrivially* within $\mathcal{C}$; they lie in the normalizer $N(\mathcal{S})$ of $\mathcal{S}$ in $\mathcal{P}_n$ but not in $\mathcal{S}$ itself.

There are $n-k$ independent stabilizer generators, which generate the full stabilizer group of size $|\mathcal{S}| = 2^{n-k}$. These $n-k$ constraints reduce the full 2^n-dimensional Hilbert space to the 2^k-dimensional code space. In other words:

- n physical qubits provide a Hilbert space of dimension 2^n,
- $n-k$ stabilizer constraints remove $n-k$ degrees of freedom,
- leaving k logical qubits, i.e. a code space of dimension 2^k (same as logical state dimension).

Within this framework, many of the most important quantum error correction codes—including repetition codes, concatenated codes (e.g., Shor code), the color code of Hamming codes (e.g., Steane code), surface codes, and subsystem codes (e.g., Bacon-Shor code)—can be described in a unified and elegant way.

3 Homomorphic Logical Measurements (Notes on the Talk and Paper [5, 7])

3.1 Surface code

Each stabilizer act on neighbor local qubits. The error threhold is low but the code distance cannot be well increased even with larger physical qubits number. The relation write $kd^2 = O(n)$ with n, k, d being the typical [[n, k ,d]] definition error code. Here we can see that if restricting on encoding rates $\frac{k}{n} \sim 1$, code distance d scales as $O(1)$. Noted that for linear code, $n \geq k + d - 1$

3.2 Quantum LDPC (Low-Density Parity-Check) code

Decoding time is large to cost computation delays, while fast decoding is an essential ingredient to fault-tolerant computation. Sparce stabilzers (low weight hamming weight) can improve the problem [14]. Quantum LDPC code provide nonlocal stabilzers, measurements. The code distance d can be increased faster not following $kd^2 = n$ (code rate: $\frac{k}{n}$). In addition, one motivation comes from when standard Shor and Steane style logical measurement cannnot be performed on large quantum LDPC code.

For typical surface code, code rate scales asymtotically to zero and with sqaure root of code distance when enlarging code block. Improvement gives nonvanishing encoding rate for different surfaces (more non-trivial loops), but with code distance logarithmic in the blocklength. **Hypergraph product construction** improve this problem: First of all, we have

$$\text{Toric code} \subset \text{Hypergraph product codes} \subset \text{Homological codes} \subset \text{Stabilizer codes}.$$

Noted that homological codes belong to mutually orthogonal binary codes, and stabilizer codes belong to additive self-orthogonal code over GF (4) with respect to the trace Hermitian inner product

Theorem 1: it guarantees that from any full-rank classical LDPC parity-check matrix H, you can systematically build a quantum LDPC code whose parameters are exactly those given.

Classical		**Quantum (constructed)**	**Notes**
Code $[n, k, d]$	$\longrightarrow$	$[[n^2 + (n-k)^2,\ k^2,\ d]]$	Quantum code parameters
LDPC (sparse)row weight i, column weight j	$\longrightarrow$	LDPC (row weight $\approx i + j$)	Sparsity preserved
Parity-check matrix H	$\longrightarrow$	$(H_X,\ H_Z)$ built from $H \otimes I,\ I \otimes H^T$	CSS-type stabilizers
Distance d	$\longrightarrow$	Distance d	Same as classical code
Rate k/n	$\longrightarrow$	$\dfrac{(k/n)^2}{1 + (1 - k/n)^2}$	Quantum rate expression

LDPC codes linear codes with sparce parity check matrix and can also be described by Tanner graph denoted by bipartite $\mathcal{T}\,(V, C, E)$. For $H = \mathbb{F}_2^{r \times n}$, $V = 1, ...n$ (called varaible nodes) is the columns of H and $C = \otimes_1, ... \otimes r$ (check nodes) with column indices i and row indices j. There is an edge set E when $H_{ij} = 1$v
Generalizations from Toric code An $m \times m$ toric code (V, E) can be represented as $\mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z}$, where the two-dimensional vertex set consists of coordinates (x, y) with each coordinate ranging over $\{0, 1, 2, \ldots, m-1\}$. The vertex–edge incidence matrix $\mathbf{H_1}$ is defined such that $(\mathbf{H_1})_{ij} = 1$ if vertex i is incident to edge j. Each i-th row of $\mathbf{H_1}$ corresponds to a vertex (an X-stabilizer), and each j-th column corresponds to an edge, which represents a physical qubit. Pictorially, for a four qubits repetition code (building block of toric code) can be denoted as in Table 1. Let $H_1 \in \{0, 1\}^{r_1 \times n_1}$ and $H_2 \in \{0, 1\}^{r_2 \times n_2}$ be classical parity-check

X stabilizer (row)	**edge_0**	**edge_1**	**edge_2**	**edge_3**
X_0	1	1	0	0
X_1	0	1	1	0
X_2	0	0	1	1
X_3	1	0	0	1

Table 1: Toric code H_r matrix for 4 edges

matrices. Define identity matrices I_a of the indicated sizes, and use the Kronecker product $\otimes$. Then, the CSS stabilizer matrices are given by

$$H_X = \left[\, H_1 \otimes I_{n_2} \;\middle|\; I_{r_1} \otimes H_2^{\mathsf{T}} \,\right], \qquad H_Z = \left[\, I_{n_1} \otimes H_2 \;\middle|\; H_1^{\mathsf{T}} \otimes I_{r_2} \,\right].$$

Toric code as a special case. If both classical codes are chosen as the length-L repetition code with parity-check $H_r \in \{0, 1\}^{L \times L}$ (representing a cyclic ring), then the toric-code stabilizer matrices become

$$H_X = \left[\, H_r \otimes I_L \;\middle|\; I_L \otimes H_r^{\mathsf{T}} \,\right], \qquad H_Z = \left[\, I_L \otimes H_r \;\middle|\; H_r^{\mathsf{T}} \otimes I_L \,\right].$$

Here the rows of H_X correspond to plaquette (face) X-stabilizers and the rows of H_Z correspond to vertex Z-stabilizers, while the columns index the $2L^2$ edge qubits of the lattice.

3.3 Logical measurements of Shor and Steane type

Standard approach will encounter two possible limitations. First, if an error occur on the ancilla qubtis, the error will propagate to data qubtis and cause higher weight errors. Below shows a graph of common error propgations extracted from [15]

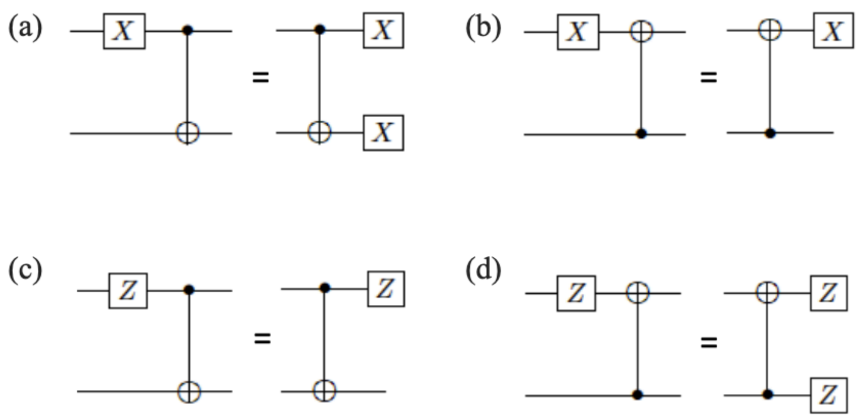


Fig. 3. Propagation of X and Z errors through the CNOT gates.

Figure 1:

Shor's fault-tolerant logical measurements are implemented by applying transversal gates between data qubits and ancilla GHZ (cat) states. The procedure requires multiple rounds, where each GHZ ancilla interacts transversally with the data qubits and is then measured in the X (Z) basis, corresponding to initial input state $|\overline{+}\rangle$ ($|\overline{0}\rangle$).

These repeated measurements allow one to perform majority voting on the syndrome outcomes, thereby suppressing the effect of measurement errors. Fault tolerance requires that errors arising at any stage do not propagate uncontrollably to the data qubits. Figure 2 illustrates this process.

One potential issue is that ancilla faults during syndrome extraction can propagate in such a way that errors mimic measurement errors. To avoid this mixing, each round of syndrome extraction must itself be implemented fault-tolerantly. By performing fault tolerant error correction in each state, a single fault can only corrupt the outcome and then be fixed during that round. This guarantees that majority voting across repeated rounds of cat state measurements produces valid syndrome information.

Shor's method requires repitions of each stage to alleviate an probability

$$P = \frac{1}{2} - (1-2p)^d = \frac{1}{2} - \Delta$$

of logical error occurs, where p is the single qubit error probability and d is the circuit depth of each stage. The majority vote requires $O(e^{2d})$ repitions.

In Steane method, logical measurement is performed by preparing an ancilla block encoded in the same CSS code (e.g., $|0_L\rangle = \frac{1}{\sqrt{2}}\big(|+++\rangle + |---\rangle\big)^{\otimes 3}$ or $|+_L\rangle$ for for the Shor code.) and coupling it to the data block with transversal CNOTs, realizing

$$\mathrm{CNOT}^{\otimes n} \;=\; \overline{\mathrm{CNOT}}^{\otimes k}$$

for an $[\![n,k,d]\!]$ CSS code. This is in fact mapping the measurement outcome from data code block to ancilla code block:

Let the data block be $|\psi\rangle = \alpha|0_L\rangle + \beta|1_L\rangle$ and the ancilla be $|0_L\rangle$.After transversal CNOTs: $|\psi\rangle|0_L\rangle \;\mapsto\; \alpha|0_L\rangle|0_L\rangle + \beta|1_L\rangle|1_L\rangle$. Measuring the ancilla block in the Z basis reveals the eigenvalue of Z_L on the data block, while collapsing it into $|0_L\rangle$ or $|1_L\rangle$ accordingly.

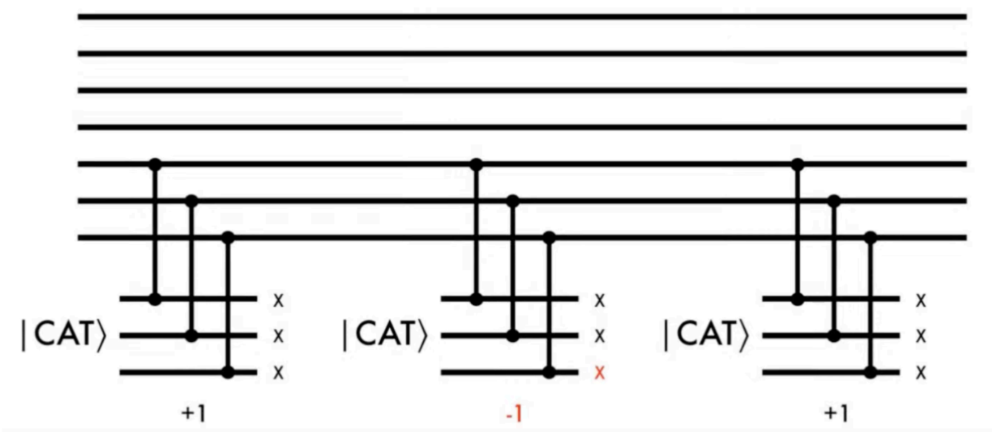


Figure 2:

Unlike Shor's cat-state method, which measures stabilizers one by one, Steane's method allows all stabilizers of one error type (either X-type or Z-type) to be extracted in a single round. One can imagine that once an measurement error ocurrs in Steane's parity checks, more parity checks outsome can be used to infer the syndromes compared with Shor's method that require more qubits for multiple stages measurements for one set of syndrome in each stage. The logical error rate yields

$$P = \mathcal{O}\left(p^{\frac{d-1}{2}}\right)$$

. Here, d is the code distance.

For an general $[\![n, k, d]\!]$ code, it is easily to generalize the ancilla states to $|0\rangle = \overline{|+_1 ... 0_i ... +_k\rangle}$ for a Z type measurements, since $X_j|+_k\rangle$ leaves no change of the state, the measurement of Z_i will only extract information from i qubits (the state $|+_k\rangle$, which treats all Z measurement outcomes on an equal footing). $|0_i\rangle$ is just the ancilla state (measurement state) for Z_i stabilizers. Dimensions of $|0_i\rangle$ state is the weight of Z_i stabilizers. Here, we have noted that for even weight of stabilizers, they are related by local Hadamard gate, called Clifford-equivalent. Also, the choice of codewords are designed by both logical operators and stabilzers. A density matrix for logical state of one-qubit encoding can be found as [1]

$$|0_L\rangle\langle 0_L| = \frac{1}{2^n}\left(I + \overline{Z}_L\right)\prod_j (I + S_j)\,.$$

For $|1_L\rangle$, one can change the plus sign to minus sign.

Problem in Steane code could be ancilla states $|0_L\rangle$ preparation [12]. It can be comprised of a non-fault tolerant preapation process combined with a verification stage. The verification stage Fig. 3 requires and additional ancilla qubit to flag a successful preparation, like post-process. The whole process then become fault-tolerant but with successful rate e^{-np}, with n being number of gates and p the successtul probability of each gate. For state other than $|0_L\rangle$ can be prepared combined with Clifford operations. Noted that apart from Clifford operations, magic state injection ($T|+\rangle$) is also required to fulfill the universal quantum operations. Similarly, by state distillation or code concatenation, desired ancilla qubit states can be obtained but with large overheads [20].

Another trick for ancilla states creation in Steane code is by performing X-type measurements. It seems like we can create the codewords $|0_L\rangle$ by following a state projection from logical operators and stabilizers:

$$\frac{1}{2^J}\left(I + \overline{Z}_L\right)\prod_j^J (I + S_j)\,.$$

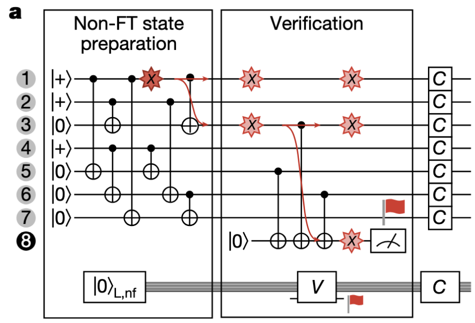


Figure 3:

If we perform X type measurements (mathematically described by above formula, while S_j being Z type measurements can be trivial), the mathematical description of this projecting process can be:

$$|0_L\rangle = \frac{1}{2^{J/2}}\cancel{(I+\overline{Z}_L)} \prod_{j\in X\text{-type}} (I+S_j)\,|0^{\otimes n}\rangle + \xcancel{\frac{1}{2^{J/2}}(I+\overline{Z}_L) \prod_{j\in Z\text{-type}} (I+S_j)\,|0^{\otimes n}\rangle}. \tag{1}$$

Noted that Z type measurements and logical Z_L measurement act trivially on $|0^{\otimes n}\rangle$ (They are already in stabilizer group or commute with stabilizers). The formula requires projective operator which could only be done unitarily. More explicity, an arbitrary state can be written as combination of projective states with different observables, hence we could write $|0^{\otimes n}\rangle = \frac{I+S_j}{2}|0^{\otimes n}\rangle + \frac{I-S_j}{2}|0^{\otimes n}\rangle$. This also demonstrate that Eq. 1 have implicitly selectively choose the projective states $\frac{I+S_j}{2}|0^{\otimes n}\rangle$ with some probability. For Steane code, this probability is $(\frac{1}{2})^3 = \frac{1}{8}$ for three consecutive projecting process.

From $|0^{\otimes n}\rangle = \frac{I+S_j}{2}|0^{\otimes n}\rangle + \frac{I-S_j}{2}|0^{\otimes n}\rangle$, we could infer that the correponding error correction of Z-type could fix the problem when projecting into wrong states. Hence, the process require further fault-tolerant error correction (FTEC) following the stabilzers measurement to deterministically generate logical $|0_L\rangle$ state(Above are my current understanding which may not correspond to what paper really trying to convey.). There is no need post-selection for Steane's ancilla qubit preparation as claimeed in the video for logical qubtis number $k = 1$. Also, for $k > 1$ the process can be used to generate $|0^{\otimes k}\rangle$ (all Z measurement at once) but not $\overline{|+_1 ... 0_i ... +_k\rangle}$ (If we want particular Z_i measurement). The reason is that $\overline{|+_1 ... 0_i ... +_k\rangle}$ are not easily prepared anymore. This may make the whole preparation process as hard as directly measuring logical operators in data block.

A natural thoughts then will be can a new choice of ancilla code such that it can achieve a LDPC measurement on particular logical qubit. The next question is, is there any other choices of ancilla code to achieve non-postselection, no repition like Steane's method for an [[m,1,d]] (Steane ancilla code: [[n,1,d]] or [[n, k ,d]])ancilla code. Here, the speaker aims to build a new code that could perform with $m < d$ that could be more resource freiendly.

The speaker introduced a measurement process called *homomorphic measurement.* A toric code is an

$$[[n,k,d]] = [[2L^2,2,L]]$$

defined on a torus, which can be represented as a square sheet with periodic boundary conditions. The stabilizers all commute, and the corresponding logical operators are shown in Fig. 10. The horizontal loops $\overline{X}_1, \overline{Z}_2$ and the vertical loops $\overline{Z}_1, \overline{X}_2$ correspond to logical operators that wrap around the torus in the horizontal or vertical directions.

3.4 Binary vector spaces

They construct the homomorphism between data qubits and the ancilla qubits by using CSS codes chain complexes.

An $r \times n$ binary matrix defines a linear map

$$H : \mathbb{F}_2^n \to \mathbb{F}_2^r,$$

where $\mathbb{F}_2 = \{0, 1\}$ with addition and multiplication modulo 2. The transpose H^T is the $n \times r$ matrix with rows and columns swapped. The kernel (null space) is

$$\ker(H) = \{v \in \mathbb{F}_2^n : Hv = 0\},$$

the image (column space) is

$$\mathrm{im}(H) = \{Hv : v \in \mathbb{F}_2^n\},$$

and the row space is the span of the rows of H, denoted $\mathrm{rs}(H)$. Note that $\dim(\mathrm{im}(H)) = \dim(\mathrm{rs}(H)) = \mathrm{rank}(H)$.

Given a finite set S, the vector space $\mathbb{F}_2[S]$ consists of all formal binary sums of elements in S,

$$v = \sum_{e \in S} v_e e, \qquad v_e \in \mathbb{F}_2,$$

which can be naturally identified with subsets of S (element e is present if $v_e = 1$). If $H : \mathbb{F}_2[A] \to \mathbb{F}_2[B]$, then the transpose defines a map $H^T : \mathbb{F}_2[B] \to \mathbb{F}_2[A]$ under the corresponding bases.

As an example, consider

$$H = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

The image is spanned by the columns $(1, 0)^T$, $(0, 1)^T$, and $(1, 1)^T$, which generate all of $\mathbb{F}_2^2$. The row space is spanned by $(1, 0, 1)$ and $(0, 1, 1)$, giving the subspace

$$\{(0, 0, 0), (1, 0, 1), (0, 1, 1), (1, 1, 0)\} \subseteq \mathbb{F}_2^3.$$

In the language of quantum error correction, the row space $\mathrm{rs}(H)$ often corresponds to the stabilizer group (constraints on codewords), the image $\mathrm{im}(H)$ corresponds to possible syndrome outcomes, and the kernel $\ker(H)$ corresponds to valid codewords with no detected error.

A CSS code with stabilizer X type and Z type will have corresponding stabilizer group isomorphic to $\mathrm{rs}(H_X)$ and $\mathrm{rs}(H_Z)$.

The quantum code can be described using two families of Pauli stabilizers. The X-type stabilizer group corresponds to parity checks that involve X operators, and it is isomorphic to the row space of H_X. Similarly, the Z-type stabilizer group corresponds to parity checks that involve Z operators, and it is isomorphic to the row space of H_Z.

The X-type logical operators are elements of $\ker(H_Z)$ (like centralizer), meaning they commute with all Z-type checks and therefore preserve the Z-stabilizer constraints. Likewise, the Z-type logical operators are elements of $\ker(H_X)$, since they commute with all X-type checks since a logical operator will stay in the codespace.

The number of encoded logical qubits is the number of independent logical degrees of freedom that remain after imposing all stabilizer constraints:

$$k = \dim\big(\ker(H_X)/\mathrm{rs}(H_Z)\big) = \dim\big(\ker(H_Z)/\mathrm{rs}(H_X)\big)$$

(quotient subgroup: The elements of the quotient space V/W are the cosets of W. Each coset is of the form $v + W$ for some $v, w \in V$. Algebraically, forming the quotient space V/W, V/W means we treat all vectors that differ by an element of W as equivalent. Topologically, V/W is like shrinking W space into a point. it is also like finding logical qubits dimension using $\dim(2^n/2^{n-k}) = k$). This formula says that logical qubits live in the space of operators that preserve one type of stabilizer (the kernel) but are not redundant with the other type (the row space). The X *distance* d_X measures how resilient the code is against bit-flip

(X-type) errors: it is the minimum number of qubits that must be flipped to implement a nontrivial logical X operation. Formally,

$$d_X := \min\{\, |c| : c \in \ker(H_Z) \setminus \mathrm{rs}(H_X) \,\}.$$

Similarly, the Z *distance* d_Z quantifies protection against phase-flip (Z-type) errors:

$$d_Z := \min\{\, |c| : c \in \ker(H_X) \setminus \mathrm{rs}(H_Z) \,\}.$$

Finally, the overall *code distance* is

$$d = \min\{d_X, d_Z\},$$

which sets the maximum number of arbitrary single-qubit errors the code can reliably detect and correct. Physically, the larger the distance, the more robust the code is against noise.

Quantum error correction uses this framework because stabilizers operators naturally form abelian groups modulo phases (self-commute).

3.5 Algebraic Topology

Notes from the lecture [3], the **2-dimensional disk** is defined as

$$D^2 = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 \leq 1\}.$$

It consists of all points in the plane whose distance from the origin is less than or equal to 1. **Interior and boundary**

$$\mathrm{Int}(D^2) = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 < 1\}, \qquad \partial D^2 = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\} = S^1.(\partial D^n = S^{n-1})$$

Topological Meaning In a CW complex, D^2 serves as a **2-cell**. Attaching a 2-cell means gluing a copy of D^2 along its boundary S^1 via a continuous map:

$$f : S^1 \to X^1.$$

For example:

- S^2 is formed by attaching one D^2 to a point ($X^1 = X^0$ here, one D^0 zero D^1, one $D^{2]}$, $\chi(S^2) = 2$ (χ defined below)) .
- A torus T^2 is formed by attaching D^2 along a loop that winds in two directions (X^0: a point, X^1 add two D^1 lines, $f : \partial D^1 = S^0 \to X^0$. X^2: add a D^2 two dimensional disk $f : S^1 \to X^1$) ($X^2 : ab^{-1}a^{-1}b$, the direction of loop are glued will result in different shape, if $X^2 : ab^{-1}ab$ is a Klein bottle). $\chi(T^2) = 0$

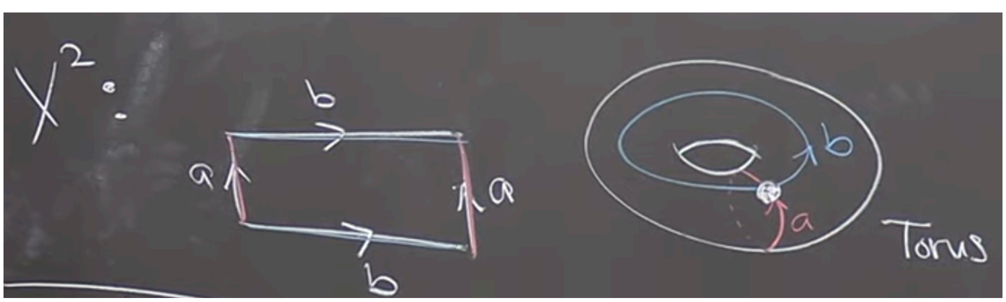


Figure 4: S^1 for torus. X^1: add

Generalization The n-dimensional disk is

$$D^n = \{(x_1, \ldots, x_n) \in \mathbb{R}^n \mid x_1^2 + \cdots + x_n^2 \leq 1\},$$

$$f : S^{n-1} \to X^{n-1}.$$

Euler characteristic Vertices $D^0 = V$, Edges $D^1 = E$, Faces $D^2 = F$,...

$$\chi = \#\text{even } \dim(D) - \#\text{odd } \dim(D)$$

$$\chi(T^2) = V - E + F = 1 - E + 1 = 0 \quad \Rightarrow \quad E = 2$$

or a torus can be build from $V = 4$, $E = 8$, $F = 4$ and similar goes for S^2 but with χ fixed.

Product and homology. Noted that D^n is contractible and S^n are not.
homotopy $\simeq$

Spaces (X, Y)	**Relationship**	**Intuition**
D^n and a point *	$D^n \simeq *$	A disk can be shrunk to a point (contractible).
S^1 and a circle-shaped wire loop	$S^1 \simeq$ any loop	All circles have the same homotopy type
S^1 and a torus (T^2)	Not homotopy equivalent	A torus has more "holes."
$\mathbb{R}^n$ and a point	$\mathbb{R}^n \simeq *$	Can contract the entire space to a point.
A hollow cylinder and a circle	$S^1 \times I \simeq S^1$	The cylinder retracts onto its circular core.

The torus (solid torus: $D^2 \times S^1$) is defined as $T^2 = S^1 \times S^1$ and its *fundamental group* is $\pi_1(T^2) \cong \mathbb{Z} \times \mathbb{Z}$. In contrast, for the circle we have$\pi_1(S^1) \cong \mathbb{Z}$. Since the integer group $\mathbb{Z}$ is not isomorphic to the product group $\mathbb{Z} \times \mathbb{Z}$, it follows that $T^2 \not\simeq S^1$. Geometrically, if one tries to shrink the torus T^2 into a circle S^1, one must collapse or "break" one of the gluing directions that form T^2. Since this cannot be done continuously without tearing the surface, T^2 and S^1 are not homotopy equivalent. $D^1 \times D^2$: a solid cylinder (sphere) Some identities:

$$D^n \times D^m = D^{n+m}$$

$$\partial(X \times Y) = (\partial X \times Y) \cup (X \times \partial Y)$$

$\cup$ is called union. For example, calculate $\partial \left(D^2 \times [1, 0]\right) = \left(\partial D^2 \times [1, 0]\right) \cup \left(D^2 \times \partial[1, 0]\right) =$ $\left(S^1 \times [1, 0]\right) + D^2 \times \{0, 1\}$ It is exactly the surface of the cylinder. Or simply, $\partial \left(D^2 \times [1, 0]\right) = \partial D^3 \; // = S^2$ So calculate $\partial \left(S^1 \times S^1 \times [1, 0]\right) = S^1 \times S^1 \times \{0, 1\}$ is two copies of torus surface.

Another example: $S^3 = \partial(D^4) = \partial(D^2 \times D^2) = S^1 \times D^2 \cup D^2 \times S^1$ (two tori formed by looping around different directions, pictorially, draw S^1 first for first qubit and then draw D^2 connected on S^1 similar for second torus but with opposite order.). Union can be think of gluing, hence gluing two tori is S^3.

Examples of **Quotients** in topology: $D^1/S^0 = S^1$, $D^2/S^1 = S^2$, $S^2/S^1 = S^1 \vee S^1$, $\vee$ (pronounce: wedge), also examples in Fig. 5

Homology group is used to describe

3.5.1 Homology Groups

Vector spaces over $\mathbb{F}_2$ are abelian groups C_i under addition. Boundary operators ∂_i are group homomorphisms (Like in toric code, logical Z_L is noncontractible loops around the torus.). Groups here are in topological sense not the same as Stabilizers group in physical Pauli sense. A chain complex is just a sequence of abelian groups with compatible homomorphisms, typically written as

$$C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0, \quad \text{with } \partial_1 \circ \partial_2 = 0.$$

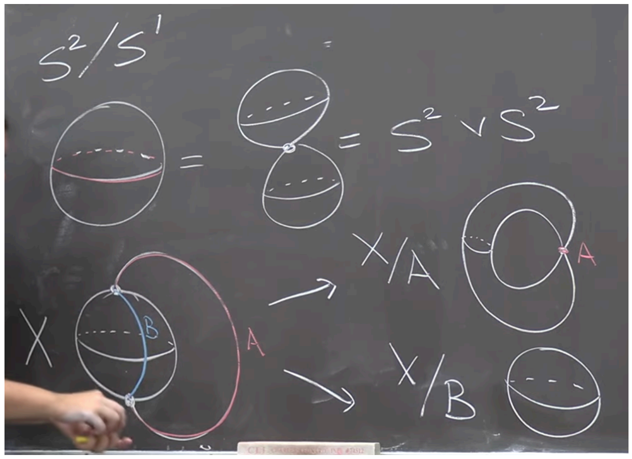


Figure 5:

In CSS codes, $H_X = \partial_1$ and $H_Z^T = \partial_2$ naturally satisfy $H_X H_Z^T = 0$ is the stabilizers. The above **homology group** is used to describe data qubits C_1 and logical operators $(\ker(\partial_1)/\mathrm{im}(\partial_2))$

Logical operators are homology classes H_i. They are cycles Z_n (commute with stabilizers) but not boundaries B_n themselves (not product of stabilizers). Mathematically:

$$H_n = Z_n/B_n \tag{2}$$

$$Z_n := \ker \partial_n := \{\, c \in C_n \mid \partial_n(c) = 0 \,\} \tag{3}$$

$$B_n := \mathrm{im}\, \partial_{n+1} := \{\, \partial_{n+1}(c) \mid c \in C_{n+1} \,\} \tag{4}$$

This correspond to $\dim(\ker(\partial_n)/\mathrm{im}(\partial_{n+1})) = \dim(\ker(H_z)/\mathrm{rs}(H_x)) = \mathrm{k}$. The algebra links with Fig. 10 toric code. The toric code can be expressed as the chain complex $C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0$, where qubits live on edges (C_1), X-stabilizers are associated with vertices (C_0), and Z-stabilizers with faces (C_2). The logical operators are characterized by the first homology group

$$H_1 = \ker(\partial_1)/\,\mathrm{im}(\partial_2) \cong \mathbb{Z}_2 \oplus \mathbb{Z}_2 = (0,0),(1,0),(0,1),(1,1),$$

which corresponds to the two nontrivial loops around the torus that encode the logical qubits, while torus requires two loops to deecribe its topology. In general,

$$\ker(H_Z)/\,\mathrm{im}(H_X)$$

corresponds to the X-type logical operators, while

$$\ker(H_X)/\,\mathrm{im}(H_Z)$$

corresponds to the Z-type logical operators. Pictorially, one can imagine that all closed loops of errors on qubits in Fig. 10 lie in $\ker(\partial_{H_Z})$, but many of them can also be formed as products of X stabilizers. The only exceptions are loops that connect opposite edges (loop around) of the torus, which give nontrivial errors that cannot be detected and by design act as logical Z operators.

3.6 Homomorphic logical measurements

As we have elaborated on Shor and Steane measurement downsides and limitations, here we dorecctly go to the arthor main points, homomorphic logical measurements. They are trying to find a new code [[m, 1 ,d]] that could unifying or improve before mentioned downsides. The process first start from preparing 1. preparing ancilla in $\overline{|0^{\otimes k}\rangle}$ 2. perform interaction Γ between ancilla and data block. 3. measured Z basis on ancilla block.

Data–Ancilla Interaction

Applying the homomorphism for CSS codes into their ancilla code construction by considering possible interaction between data-ancilla interaction (typically utlising similar mathematical but applying on different purposes.):

We have two CSS codes: - Data: (H_X, H_Z) of length n, - Ancilla: (H'_X, H'_Z) of length m. Before interaction, stabilizer groups are written as

$$T_Z = \mathrm{rs}\begin{pmatrix} H_Z & 0 \\ 0 & H'_Z \end{pmatrix}, \quad T_X = \mathrm{rs}\begin{pmatrix} H_X & 0 \\ 0 & H'_X \end{pmatrix}.$$

After Interaction (Γ a gate matrix $\Gamma : \mathbb{F}_2^m \to \mathbb{F}_2^n$ (CNOTs)), stabilizer groups can be written as

$$T'_Z = \mathrm{rs}\begin{pmatrix} H_Z & 0 \\ H'_Z\Gamma^T & H'_Z \end{pmatrix}, \quad T'_X = \mathrm{rs}\begin{pmatrix} H_X & H_X\Gamma \\ 0 & H'_X \end{pmatrix}.$$

To explain T'_Z further, it is like the outcome of $H_Z^{'}$ on ancilla qubits, is determined not only by the state initial state $|0_L\rangle$ lie in ancilla block but also $H_Z^{'}\Gamma^T$ when performing interaction, which is like a different mapping other than Steane style, from my understanding, Steane style measurement follows $H'_Z\Gamma^T = H'_Z$ (Since Γ here is like identity for transversal gates in Steane measurement) and also $H'_Z = H_Z$ since they are using same logical codewords, hence same stabilizers. Just like the author mentioned, for Shor's style measurment, $H'_Z \neq H_Z$ since $H_{Z'}$ should correspond to cat states stabilzers (1D).

The role interchange between target and controlled of X type and Z type errors can be explained by the error propagation shown in Fig. 1.

We also required conditions such $T'_Z = T_Z$, $T'_X = T_X$, i.e.

$$\mathrm{rs}(H'_Z\Gamma^T) \subseteq \mathrm{rs}(H_Z), \quad \mathrm{rs}(H_X\Gamma) \subseteq \mathrm{rs}(H'_X).$$

This ensures the interaction preserves the stabilizer groups.

Definition (Homomorphic gadget). An $[[m, k', d']]$ homomorphic gadget (H'_X, H'_Z, Γ) for an $[[n, k, d]]$ CSS code (H_X, H_Z) consists of: (i) an ancilla $[[m, k', d']]$ CSS code with checks (H'_X, H'_Z); (ii) a gate matrix $\Gamma : \mathbb{F}_2^m \to \mathbb{F}_2^n$; such that

$$\mathrm{rs}(H'_Z\Gamma^T) \subseteq \mathrm{rs}(H_Z), \qquad \mathrm{rs}(H_X\Gamma) \subseteq \mathrm{rs}(H'_X). \tag{5}$$

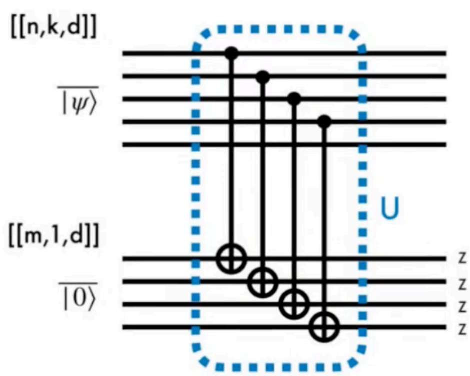


Figure 6:

To summarize, a stabilizer element (in fact also logical error) $v \in \ker H'_Z$ are transformed as $\Gamma v \oplus v$, acted on data block (Γv) and ancilla block (v), since ancilla state is prepared in logical $|0^{\otimes k}\rangle$, the outcome will be Γv. There are two cases: where $v \in \mathrm{rs}\,(H'_Z)$ or $v \notin \mathrm{rs}\,(H'_Z)$, the former under homomorphic gadget setting will preserve the structure of $v \in \mathrm{rs}\,(H'_Z)$ and act as a X error detection for data block. The latter are in fact mapping logical Z operation into ancilla block. As mentioned, the outcome will be Γv (it is measured in ancilla block but in fact bring based on data block information. One can simply assume a vector acting by matrix T'_Z to see this) which will isomorphic to $\Gamma \ker(H_X)$ (seems might encounter vector space outside $\Gamma \ker(H_X)$)

3.7 Homomorphic measurements on surface codes

Surface codes are defined as cellulations of a manifold $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, where the boundary maps $\partial_2 : \mathcal{F} \to \mathcal{E}$ and $\partial_1 : \mathcal{E} \to \mathcal{V}$ obey the CSS code condition $\partial_1 \partial_2 = 0$. (Can LDPC CSS codes, such as hypergraph product codes, have different homomorphic gadgets?) Linear maps $\gamma : \mathcal{A} \to \mathcal{D}$ connect the ancilla and data surface codes. In fact, the gate matrix is given by $\Gamma = \gamma_1$ in the paper, where $\gamma_1 : \mathcal{E}' \to \mathcal{E}$ is the linear map between qubits. The data and ancilla surface codes are defined respectively as $\mathcal{D} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, and $\mathcal{A} = (\mathcal{V}', \mathcal{E}', \mathcal{F}')$. Explicitly, the relation between data block and ancilla block:

$$\begin{array}{ccccc} \mathbb{F}_2[\mathcal{F}'] & \xrightarrow{\partial_2'} & \mathbb{F}_2[\mathcal{E}'] & \xrightarrow{\partial_1'} & \mathbb{F}_2[\mathcal{V}'] \\ \downarrow \gamma_2 & & \downarrow \gamma_1 & & \downarrow \gamma_0 \\ \mathbb{F}_2[\mathcal{F}] & \xrightarrow{\partial_2} & \mathbb{F}_2[\mathcal{E}] & \xrightarrow{\partial_1} & \mathbb{F}_2[\mathcal{V}] \end{array}$$

The above relation naturally gives homomorphic gadget conditions shown in Eq. 5 , as $\gamma_1 \partial_2' = \partial_2 \gamma_2 \subseteq \partial_2$ and $\gamma_0 \partial_1' = \partial_1' \gamma_1 \subseteq \partial_1$. The paper seems like weakening the global homeomorphism constraints of a usual linear map Γ (γ_i) such as Steane or Shor to local homeomorphism. This generalization gives more degree of freedom to represent logical operators to a single non-contractable loop in a new manifold. This generalization do not preserve transversal gates, as we can see that γ_i local homeomorphism, or covering spaces can be many-to-one linear maps. There are also certain boundaries for manifold M, with two rough boundaries and two smooth boundaries is the planar surface codes [4].

The paper contructs homomorphic gadgets into two categories: **subspaces of data code space** $\mathcal{D}$ and **covering space** of $\mathcal{D}$. For the first one, it is natural that homomorphic gadget can be constructed given Γ is injective (one-to-one, hence transversal and fault tolerant.$\mathcal{A}(\mathcal{V}', \mathcal{E}', \mathcal{F}') \in \mathcal{D}(\mathcal{V}, \mathcal{E}, \mathcal{F})$). Shor code can be thought of as $A = l \subseteq (\mathcal{V}, \mathcal{E})$ and l loops not intersecting (loops here generally mean logical operators, so not restricted on toric code loops, if loop intersects, it could involve two logical operators which is not in cat state gadget.), with $F = \emptyset$ and this indicates repition code $0_L = \frac{1}{\sqrt{2}}(|+++...\rangle + |---...\rangle)$ will have only X stabilizers, for repetition code of Z stabilizers, one use T'_X which interchange the controlled and target qubits between data and ancilla qubits.

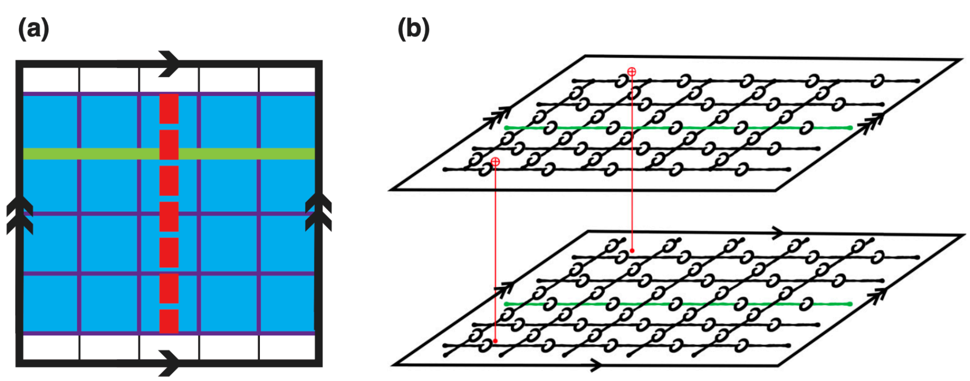


Figure 7:

Fig. 7 (a) describes a toric code $\mathcal{D}$ and the blue region describes surface code with smooth boundaries. Red line connecting two boundaries are then logical X(or by seeing red lines crossing sets of Z stabilizers.).

The logical operator dimension of $\mathcal{A}$ are then reduced from four (complete mapping correspond to Steane measurement) to two. This ensures simpler preparation of ancilla states as mentioned in 3.3, which is also a problem associated with Steane measurement if utilising a complete mapping between data (might involve two or more logicals) and ancilla block.

Here we move on to homomorphic gadgets from **covering spaces**. Emphasizing the motivation again, if we want to perform a single-shot nondestructive logical CSS measurements on multiple logical operators (ancilla block), then the direct mapping such as Steane code or $A \in l$ inevitably support two logicals degree of freedom but with overlapped qubits, furthermore, multiple logical qubits ancilla is hard to prepare. The idea is to unfold the manifold to make logical operators uniquely represented by a non-intersecting loop in ancilla sheets. This resolves all of the problems mentioned.

Groups acting on spaces: The infinite *simply connected* covering space U (for example, $\mathbb{R}^2$) is equipped with a regular tiling (cellulation: divided into cells like vertices, edges, faces, $[i,\ i{+}1]\times[j,\ j{+}1]$ for $(i,j) \in \mathbb{Z}^2$). A group G of symmetries(leave the grid-structure intact), such as translations or rotations, acts on U, and for each point $u \in U$, its orbit $Gu = \{g(u) \mid g \in G\}$ consists of all symmetry-related copies of u (all Gu collapses to one point.). The orbit space U/G is the corresponding quotient manifold (for instance, a torus), and the quotient map (continuous and open) $p_G : U \to U/G$ sends each point u to its orbit Gu. The resulting manifold $M = U/G$ is the compact surface on which the surface code is defined. Because each element of G preserves the tiling of U, the quotient map p_G induces a cellulation of M; that is, every k-cell in U maps to a k-cell in M, preserving the lattice structure.

As discussed in Sec. 3.5 and illustrated in Fig. 4, one can regard the **first example: torus** as the quotient of the real plane $\mathcal{U} = \mathbb{R}^2$ by the integer translation group $G \cong \mathbb{Z} \times \mathbb{Z}$ (translations $t_{r,s}\,(x, y) : (x, y) \to (x + dr, y + ds)$ for an $[2d^2, 2, d]$ toric code). Intuitively, this corresponds to identifying points that differ by integer shifts, i.e., taking 0 and 1 as the same point in each direction. The quotient $\mathbb{R}^2/(\mathbb{Z}\times\mathbb{Z})$ can thus be represented by the unit square $[0, 1) \times [0, 1)$, where opposite edges—labeled a and b in Fig. 4—are glued together to form the torus topologically. Because the torus is constructed as this quotient, its *fundamental group* is isomorphic to the translation group itself,

$$\pi_1(T^2) \cong \mathbb{Z} \times \mathbb{Z},$$

with each generator corresponding to one of the two noncontractible loops along the a and b directions.

We can also consider **second example: hyperbolic surface codes**, with the universal $\mathcal{U} = \mathbb{H}^2$ which are defined on regular tilings characterized by a Schläfli symbol $\{r, s\}$ (note that this is unrelated to the integer coordinates (r, s) used earlier). Here, r indicates that each face (tile) is a regular polygon with r sides, and s means that s such faces meet at each vertex. The pair $\{r, s\}$ determines both the curvature of the surface and the stabilizer structure: if $(r - 2)(s - 2) < 4$, the surface is spherical; if $(r - 2)(s - 2) = 4$, it is Euclidean (flat, as in the toric code); and if $(r - 2)(s - 2) > 4$, it is hyperbolic. In the code, each Z-type stabilizer acts on r qubits (around a face), and each X-type stabilizer acts on s qubits (around a vertex). The **Coxeter group** $G_{r,s}$ preserve the tilling structure. Group G is chosen as the normal subgroup of $G_{r,s}$ (like relation between Pauli group and Clifford group). The parameters $[\![n,\ k,\ d]\!]$ satisfy $k = O(n)$ and $d = O(\log n)$.

(This passage formalizes how one can form a quotient manifold $\mathcal{U}/G$ by identifying points under a group of **local homeomorphism**, in a way that preserves the cellulation and thus the qubit and stabilizer structure of the original topological code.)

The *image* of N_u, $g\,(N_u)$ is the set of all points in $\mathcal{U}/G$ that are reached when applying the map p_G to every point in $N_\nu : p_G(N_u) = \{\, p_G(x) \mid x \in N_u \,\}$. Therefore, if N_ν is a small open patch around u in the original space $\mathcal{U}$, then the set $N_v := p_G(N_\nu)$ is the corresponding small open patch and disjoint in the quotient space $\mathcal{U}/G$. N_ν and $g(N_u)$ are homeomorphic. Also, no nontrivial $g\,(u) = u$ (Not fixed points mean mapping all of the points to N_ν, bijective: injective and surjective)

Third example: [[2d^2, 2, d]] toric code. if we choose U_u for any $u = (x, y) \in \mathcal{U} = \mathbb{R}^2$

The *lifting property* is the key topological feature they rely on, since it allows any logical operator—represented by a noncontractible loop on the base surface to be lifted to a non-self-intersecting path on a multi-sheeted covering manifold. The loop on $\mathcal{U}$ starts at u and ends at some translated copy of $g(u)$.

Fourth example: [[2d^2, 2, d]] toric code. Lifting a horizontal loop l on $\mathcal{U}/G$ to $\tilde{l}$. Mathematically, denoted as $g(u) = t_{10}(u)$, where $u = (0, 0) \in \mathcal{U}$. One can imagine logical operator correspond to $\mathcal{U}/G$ is like viewing $(0, y)$ and (d, y) as same point. Also, $\tilde{l}$ is guranteed to be a loop if and only if l is contractble on $\mathcal{U}/G$ given U is simply connected.

Consider another covering map $p_G^H : \mathcal{U}/H \to \mathcal{U}/G$ defined as $p_G^H H(u) = G(u)$. When $H = \langle t_{1,0} \rangle$ (horizontal translations), the intermediate covering space $\mathcal{U}/H$ is an infinite *cylinder*, obtained by identifying points along the horizontal direction of the universal cover $U = \mathbb{R}^2$. The base space $\mathcal{U}/H$, where $G = \langle t_{1,0}, t_{0,1} \rangle$, is the *torus*, obtained by identifying both horizontal and vertical directions. On the torus $\mathcal{U}/H$, the horizontal and vertical logical loops correspond to $t_{1,0}$ and $t_{0,1}$, respectively. When lifted to the cylinder $\mathcal{U}/H$, the horizontal loop remains closed since $t_{1,0} \in H$, while the vertical loop becomes an open segment as $t_{0,1} \notin H$. This pictorizes the general relation

$$g \in H \iff \text{the lifted loop } \ell \text{ is closed on } \mathcal{U}/H.$$

Fifth example: [[2d^2, 2, d]] toric code is same as the previous example for relation $g \in H \iff$ the lifted loop ℓ is closed on $\mathcal{U}/H$.

3.8 Homomorphic gadgets for covering spaces

Now we can start to construct homomorphic gadgets for covering spaces. Until now, we make some remarks: $g(u)$ lives in $\mathcal{U}$ and $p_G(u)$ lives in $\mathcal{U}/G$. For loops, $p_G(g(u)) = p_G(u)$ on $\mathcal{U}/G$ but $g(u) \neq u$ on $\mathcal{U}$ could be possible. This coule directly be seen $p_G(u) = Gu = \{\, g(u) \mid g \in G \,\}$ while p_G represented all possible $g(u) \in \mathcal{U}$ and collapse to one point in space $\mathcal{U}/G$ by definition.
The task is to find $\mathcal{A} \subseteq \tilde{D} = \mathcal{U}/H$ (where H is defined previously) such that $\mathcal{A} \subset l'$ and satisfies $d_{\mathcal{A}} = d_{\mathcal{D}}$.

As discussed before, the subgroup $H \supseteq G$, and $p = p_G^H$ is the covering map from $\mathcal{U}/H$ to $\mathcal{U}/G$. If we pick $H = \langle g \rangle$ ($g \in G$), then all the loops are unfolded except the loop l corresponding to the g-translation.

Specifically, we map two non-contractible loops to one non-contractible loop in the ancilla block $A \subseteq \tilde{D} = \mathcal{U}/H$. This ensures that we only have one unique logical operator in $\mathcal{U}/H$, where H is chosen to be $\langle t_{1,1} \rangle$ (i.e., no overlapping qubits like in the toric code with different logicals). This unique logical operator can be designed to represent $\overline{Z_1 Z_2}$, enabling single-shot measurement. Noted that ancilla block $\mathcal{A}$ is chosen such that $d_A = d_D$(minimum weight of a nontrivial X logical operator of $\mathcal{A}$, is the red line part in Fig. 9(c)

The induced homomorphic gadget (not necessarily transversal for covering maps) is induced by map $\gamma := p \circ \tilde{\gamma}$, where $\quad \tilde{\gamma} : \mathcal{A} \to \tilde{D}$, $\gamma : \mathcal{A} \to \mathcal{D}$

One can obtain a clearer physical picture from Fig. 9. Panel (a) shows the data-qubit manifold $\mathcal{U}/G = \mathbb{T}^2$, where the green loops represent the logical operators $\overline{Z_1 Z_2}$. In panel (b), the corresponding logical loop ℓ is lifted to the covering space $\mathcal{U}$, forming a path that connects the two points (x, y) and $(x + d, y + d)$ (connecting two grids), pictorially, imagine two grids collapse into one grid due to translation symmetry, then the green lines in Fig. 9(b) become Fig. 9(a). Finally, panel (c) illustrates the ancilla-qubit manifold $\mathcal{U}/H$, with $H = \langle t_{1,1} \rangle$ which takes the form of a cylinder. The covering spaces correspond to Fig. 9(c) is shown in Fig. 8. Noted that red line in Fig. 9(c) is logical X operator, when depicting in Fig. 8, it will become a line connected two smooth boundaries.

3.9 Fault tolerance

Since there is no transversal mapping for $\gamma := p \circ \tilde{\gamma}$, while homeomorphism between data sheets and ancilla sheets in standard measurement method is levergaed to local homeomorphism between them. The mapping between edges then might encounter many-to-one coupling, $\gamma_1^T(e) \in E'$. Even under these correlations, it is shown it still have fault tolerance with X error $\min\{d_{\mathcal{A}},\, d_{\mathcal{D}}\}$.

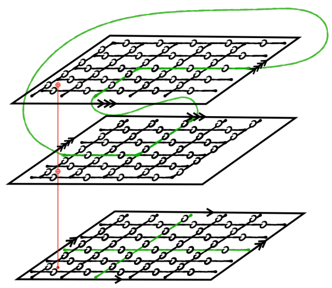


Figure 8:

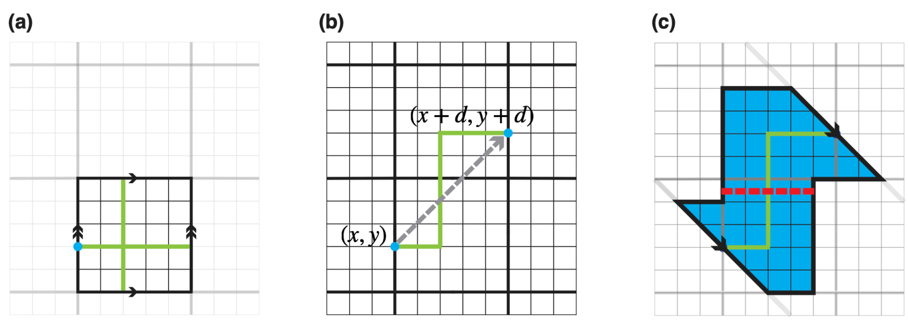


Figure 9: (a) Data-qubit manifold $\mathcal{U}/G = \mathbb{T}^2$, where the green loops represent the logical operators $\overline{Z_1 Z_2}$. (b) The covering space $\mathcal{U}$, showing the lifted path connecting (x, y) and $(x + d, y + d)$. (c) The ancilla-qubit manifold $\mathcal{U}/H$, which is topologically equivalent to a cylinder.

3.10 Joint measurement

Considering two disjoint loops l_1 and l_2 on $\mathcal{U}/G$, if the manifold $\mathcal{M}$ is path connnected, then logical operator can be $l_1 p l_2 p^{-1}$.

For two separate codes, say two ancilla blocks $\mathcal{A}_1$, $\mathcal{A}_2$, in order to prepare ancilla, one uses a lattice surgery approach to entangle two blocks from the initial state $|+\rangle_1|+\rangle_2$ into the logical Bell state $|+\rangle_L = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ by measuring $Z_{\mathcal{A}_1} Z_{\mathcal{A}_2}$ with some surface code A' satisfying $\partial\mathcal{A}' = l'_1 \cup l'_2$. (Note that the results will be either $|+\rangle_L = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ or $|-\rangle_L = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$. One then applies X_1 for correction.) Just like the layer of ancilla blocks depicted in Fig. 8, $Z_{\mathcal{A}_i}$ can be a closed loop on the boundary, $l'_i \subseteq \partial\mathcal{A}_i$. After ancilla preparation, one could construct homomorphic gadget (entangle data block and ancilla block) and perform logical measurement afterwards. Ancilla states can be prepared *offline*.

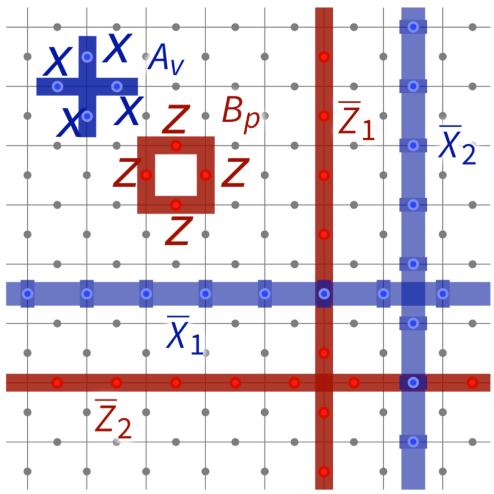


Figure 10:

4 Summary

They first establish the algebraic conditions under which the interaction matrix $\Gamma = \gamma_1$ between a data block $[\![n, k, d]\!]$ and an ancilla block $[\![n', k', d']\!]$ preserves the stabilizer structure. Motivated by topological intuition, the authors represent intersecting logical loops as a single noncontractible loop. This construction achieves two goals: (1) it enables single-shot measurement of multiple logical operators, and (2) it simplifies the logical state preparation of the ancilla block.

The intuition is formalized through *covering map* between the topological structures (vertices, edges, and faces) of the data and ancilla codes. Such a map induces corresponding linear mappings between their chain complexes, ensuring that the homomorphic gadget conditions are automatically satisfied.

In this framework, the Steane measurement corresponds to a homeomorphic (one-to-one) chain map, while the homomorphic logical measurement generalizes it to a *covering map* (locally bijective but globally many-to-one). This broader formulation naturally supports more general and scalable constructions of logical measurements across CSS codes.

5 Improved noisy syndrome decoding of quantum ldpc codes with sliding window (Notes on [8])

The paper introduces a sliding-window decoding approach for qLDPC codes (including hypergraph product codes and lifted product codes) and demonstrates a significant improvement in the logical error rate. While sliding-window decoding has previously been studied for surface codes, its application to qLDPC codes has been lacking. The results are compared with single-shot qLDPC which requires using larger code size. The results are believed to be faster and more accurate for qLDPC codes instead of single-shot decoding.

Single-shot decoding: You can decode correctly (or nearly so) using only a single round of syndrome measurements instead of multiple rounds of syndrome measurements where is typically used. This require decoder can distinguish bewteen **data-qubit** errors and **measurement errors** and apply correction without repeated measurement rounds. Example: 3D color code with extra local containts among syndromes let you detect measurement errors. However, single-shot decoding often fail to preserve code distancen and logical error rate, this requires further larger code size, which is not resource-preserving also in a long run.

Sliding-window: Correct previous measurement syndromes and leave the latest errors for future correction. This increase logical memory lifetime and the code distance for hypergraph-product codes and lifted-product codes by using single-shot decoding. This decoding turns out to be more desirable for fast and accurate decoding for qLDPC codes.

5.1 Single-shot decoding

$m \times n$ parity-check matrix H (In the article, Z-type generators to correct X-tyoe error). An X error set represented by $e \in \mathbb{F}_2^n$ with ideal syndrome $\sigma_{ideal} := He \in \mathbb{F}_2^m$ and realistic syndrome denoted as $\sigma := \sigma_{ideal} + u$, where $u \subset \mathbb{F}_2^m$ represents the set of measurement errors.

To deal with readout errors, repetitive measurement (In [7], repetitive measurements are unnecessary because the logical operator is extracted using an ancilla block encoded in a distance-d code. The logical measurement outcome is therefore protected to the same degree as the data, allowing a single-shot procedure analogous to Steane's method but with substantially lower ancilla preparation overhead. Importantly, encoding a single logical operator into a distance-d ancilla is far more resource-efficient than attempting to eliminate repetition in standard syndrome extraction. In conventional error-correction protocols, avoiding repetitive stabilizer measurements would require encoding each stabilizer into its own distance-d ancilla block, which is prohibitively expensive for LDPC codes with $O(n)$ stabilizers. Thus, homomorphic logical measurement avoids repetition through an efficiently encoded ancilla, whereas repetitive measurement remains essential for syndrome extraction.) is common strategy but with lower error correction speed.
One could then consider **single-shot decoding** which correct the single shot decoder immediately after a noisy (only X errors can be detected) syndrome extraction. After a noisy decoding process, error set e be extracted as $\widetilde{e}$, we then have $H\widetilde{e} + \widetilde{u} = \sigma \neq \sigma_{ideal}$ with $e \neq \widetilde{e}$ (if is the ideal model, e and $\widetilde{e}$ will be different up to X-type stabilizers.). However, in single-shot correction, it is shown that if $e + \widetilde{e} \in r$ where r is an X error set with $|r|$(weight of r) is bounded.

Usually if fault-tolerance threshold exists, single-shot decoding is considered success (repetitive code ususally give $\to 0$ error threhold.). Therefore, this naturally gives weight$|r| \leq \alpha|u|$ with $\alpha > 0$, α is an constant independent of code size n. ($|r| = e + \widetilde{e}$ is the weight with one-shot decoding and correction completed, therefore, if $|r|$ is smaller than measurement weight $\alpha|u|$, then somehow produce effective correcting.).

Threshold is not the only metric for QEC code, while it is naturally to think of space-time overheads to characterize the effectiveness of QEC code. For example, if a code performs with high distance but required a extremely large amounts of gates in encoding process and decoding process, then this might not be a good code. Another metric is called *effective distance*, d_{eff} which is upper bounded by code distance and determines the size of the code. We have logical error rate $p_L \propto p^{d_{eff}+1/2}$. This metric not only determines the circuit depths (Toric code stabilizer measurement: 4 CNOT layers + measurement depth = O(1), which is quite good.) of the code, but also make it like space-time dependent (circuit depths $\propto$ computation time) which partially related to the overall computation of a round of error correction.

For $\alpha \leq 1$, then $d_{eff} = d$. For $\alpha >> 1$, this shows extreme noisy decoding process with generated qubit errors $\alpha|u| >> |u|$. The worst case would have $d_{eff} = d/\alpha$. This naturally leads to building larger code, while overall space overhead will not change using constant-rate qLDPC codes. As more logical qubits eccoded in a single block code, the space-time of logical computation overhead and decoding complexity is increased. Above seems like trying to characterize what they mention before while single-shot decoding can degrade code distance d to d_{eff}, although overall space overhead (logical qubit per qubits (encoding rate)), this still requires large overheads for improving logical memory time and code distance.

5.2 Sliding window decoding

Instead of using large codeblock for an improved effective distance, the author considers sliding-window decoding, which use sets of syndromes accumulated multiple rounds of stabilzer measurements. A sliding window decoding (W(width), F(offset)) means for each W syndrome measurements and decoding we apply correction on F rounds and leaving $W - F$ rounds for the future correctin cycles.

The measured syndrome σ_t at time t obeys

$$\sigma_t = H\left(\sum_{j=1}^{t} e_j\right) + u_t.$$

Then the decoder $\mathcal{D}_{\text{win}}$ takes $(\sigma_1, \ldots, \sigma_W)$ as input to estimate $\widetilde{e}_1, \ldots, \widetilde{e}_W$ and $\widetilde{u}_1, \ldots, \widetilde{u}_W$ such that

$$H\left(\sum_{j=1}^{t} \widetilde{e}_j\right) + \widetilde{u}_t = \sigma_t.$$

The decoding part in this paper uses BP-OSD method which will be introduced in next Section [19] and with phenomenological error model with measurement error u_t and qubit error e_j and proabability p and independent to each other.

To emphasize, the decoder cannot recover the exact errors e_t and u_t (while exact errors seem like could be traced in classical computers as random generating lists), but only get approximations $\widetilde{e}_t$ and $\widetilde{u}_t$. After getting σ_t from decoder $\mathcal{D}_{\text{win}}$, we then apply

$$\xi := \sum_{j=1}^{F} \widetilde{e}_j$$

on the qubits. Then update each syndrome σ_t for $t \in [F+1, W]$ as

$$\sigma_t' := \sigma_t + H\xi.$$

which can simply be changed manually with actual operaion in code. For next round, we change

$$\sigma_t = H\left(\sum_{j=W+1}^{t}\right) + u_t$$

with t ranges from $W+1, \ldots , W+F$ which we the generates sets of syndrome $(\sigma_{W+1}, ..., \sigma_{W+F})$. Combined with previous round generated $(\sigma_{F+1}, ...\sigma_W)$. We can then take W inputs $(\sigma_{F+1}, ...\sigma_{W+F})$ to compute the next round decoder $\mathcal{D}_{\text{win}} := \sigma_t = H\left(\sum_{j=F+1}^{W+F} \widetilde{e}_j\right) + \widetilde{u}_t$.

Different choices of W, F might benefit the qLDPC codes compared with single-shot $(W, F) = (1, 1)$. While non-overlapping decoding will necessarily suffer from a residual error where they are not able to correct measurement/qubit errors occur on $W - 1$ timestep.

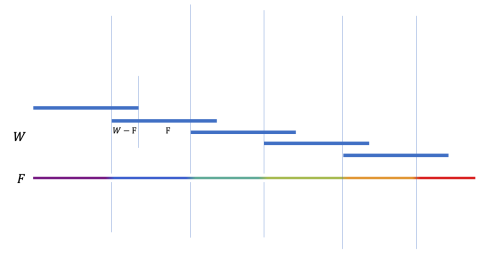


Figure 11:

5.3 Numerical simulations

Consider a phenomenological error model in which a qubit error e_j occurs with probability p, independently of a measurement error u_j, which also occurs with probability p. The criteria for a good performance of (W, F)-decoding is perform multiple rounds of syndrome extractions, decoding, and correction when an actual logical error occurs. The memory *lifetime* $T := (N-1)F$ if the logical fail occurs when after N EC. Also, T is a function of error probability p.

Along the cycles, we keep track on a quantity *residual qubit error* $r := \sum_{i=1}^{NF} e^i + \sum_{j=1}^{N} \xi_j$, once the quantity r is not a **correctable** error set even in the absence of measurement errors.

To determine whether r is correctable or not, an *ideal decoder* $\mathcal{D}_{\text{ideal}} : \mathbb{F}^m \to \mathbb{F}^n$ is applied, namely $\widetilde{r} := \mathcal{D}_{ideal}(Hr)$ to get $\widetilde{r}$. If $r + \widetilde{r}$ is non-trivial logical operator, then the logical error occurs.

The strategy is to check whether the following relation holds:

$$r + \widetilde{r} \in \ker(H_X) \setminus \text{rs}(H_Z).$$

Note that now $r + \widetilde{r}$ is guranteed to lie in normalizer group.

BP-OSD is used for characterizing D_{ideal} and D_{win} (BP: *belief propagation* OSD: *ordered-statistics decoding*, a post-processing algorithm), with open source [13] and combination sweep method [11]. Combination-sweep parameter ( ) is set to 40 in this paper.

Instead of using $(\sigma_1, \sigma_2, ...\sigma_W)$, we use $(\sigma_1, \sigma_2 - \sigma_1, ...\sigma_W - \sigma_W)$ which is by mapping relation $\left(\sigma_t = H_{win}\left(\sum_{j=1}^{t} e_j\right) + u_t\right)$ with $H_{win} = [I_W \otimes H | B \otimes I_m]$, where B is a $W \times W$ matrix and H is a $m \times n$ matrix.

Hypergraph Product Codes: a linear code with $m_A \times n_A$ parity-check matrix A and $HGP(A)$ has X and Z parity-check matrices:

$$\begin{aligned} H_X &= \left[A \otimes I_{n_A} \quad | \quad I_{m_A} \otimes A^T\right], \\ H_Z &= \left[I_{n_A} \otimes A \quad | \quad A^T \otimes I_{m_A}\right]. \end{aligned}$$

An $(r, s) = (3, 4)$ LDPC matrix means that each column has exactly 3 ones and each row has exactly 4 ones, with all other entries equal to zero. The left and middle panels of Fig. 12 show comparisons between non-overlapping decoding and overlapping decoding. The panel on the right shows that, for larger/more complex quantum codes, copies of overlapping decoding with same logical qubits as larger quantum codes still outperforms single-shot decoding in terms of logical lifetime.

Decoding volume $V = W(n-k)/2$ is a measure of decoding complexity. While 4 copies of [[625,25,8]] have same decoding volume as [2500,100], it still yields longer logical memory lifetime.

Lifted product codes: Single-shot, overlapping, and non-overlapping decoding results for the $[714, 100, \leq 16]$ and $[1428, 184, \leq 24]$ lifted product codes are shown in Fig. 13, reproduced from [17].

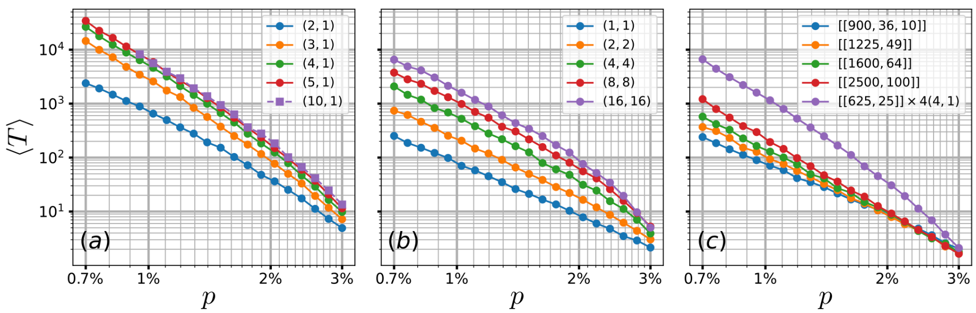


Figure 12:

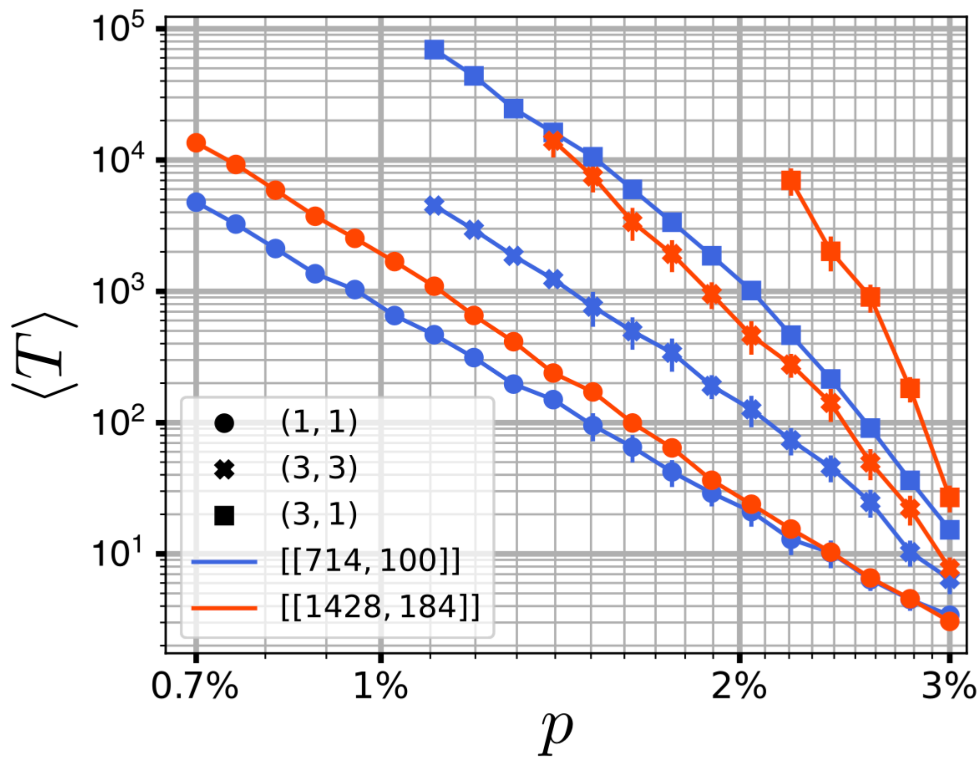


Figure 13:

6 Decoding across the quantum low-density parity-check code landscape (Notes on [13])

Numerical simulations for previous paper based on studying paper [13].

1.Three families of hyperproductc code: topological codes, fixed-rate random codes, semitopological codes (shared properties of topological and hypergraph code allows for a trade-off between code threshold and stabilizer locality). 2. Exponential supression in lower error regime for all three families

3. Compared with previous belief propagation decoders, the paper provides threshold comparable with minimum-weight perfect matching algorithm (golden standard on threshold, but) for toric codes which is also expected and subsequently demonstrated in the paper.

4. BP-OSD can be generalized to all QLDPC codes that can be contructed by hypergraph product while MWPM, union-find (UF) cannot.

topological QLDPC codes have local stabilizers embedded in some D-dimensional space with high error threshold but lower logical encoding rate.

Random QLDPC codes are hyperproduct of high-performance classical LDPC codes with higher encoding rates(Unlike tend to zero with increasing block length on topological codes) and lower threhold than topological counterparts but with distant stabilzers that is harder to implement in various quantum platform without full connectivity.

The paper exapnds on the results that BP(belief propagation) + OSD(ordered statistics decoding) decoder is a general decoder for all QLDPC codes that can be constructed from hypergraph product.

semitopologiccal codes: Interplay between local topological codes and nonlocal random QLDPC codes. Firstep: *adge augmentation*: replacing each parity check edge with a length-g section of repetition code. Then semitopologiccal code is obtained by hypergraph product. It can be thought of as surface codep patches connected to one another at their boundaries enabling long-range interaction.

For random QLDPC codes, quantum degeneracy (multiple equivalent solutions to the decoding problem) can be resolved by combining BP and OSC post-processing. When BP fails, OSD with matrix inversion is introduced to resolve ambiguities in quantum degeneracy.

Not only random QLDPC codes, but also topological QLDPC codes and semitopological codes are improved by using BP+OSD in this paper. Toric code threshold is found to be near the minimum weight-perfect-mathcing algorithm. For large code block size of semitopological codes (more local stabilziers.) the threshold approaches toric code results.

6.1 Low-density parity-check codes

Classical error correction.

Classical error correction. A classical error-correction code $\mathcal{C}_H$ describes a redundant encoding $b \to c$ from a k-bit data string b to an n-bit codeword c (with $n > k$). The codewords $c \in \mathcal{C}_H$ are defined as the null-space (kernel) vectors of an $m \times n$ binary parity-check matrix H such that

$$Hc \bmod 2 = 0,$$

that is,

$$\mathcal{C}_H = \ker(H) = \{\, c \in \mathbb{F}_2^n \mid Hc = 0 \,\},$$

where the arithmetic is taken over the finite field $\mathbb{F}_2$.

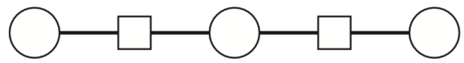
By the rank–nullity theorem, a parity-check matrix permits $k = n - \operatorname{rank}(H)$ linearly independent codewords. If a codeword is subject to an error e, the parity-check matrix yields an m-bit syndrome

$$s = H(c + e) = He.$$

The syndrome is nonzero for all errors of Hamming weight less than the code distance, i.e. $|e| < d$. In general, classical codes are labeled using the notation $[n, k, d]$, where n is the codeword length, k is the number of encoded bits, and d is the code distance. The code rate is given by

$$R = \frac{k}{n}.$$

Factor graphs: Data nodes $V = \{v_j | j = 1, ...n\}$ drawn as circles parity nodes $U = \{u_i | i = 1, ..., m\}$ drawn as squares graph edge $\Lambda_{ij} \in \Lambda$ is drawn between when $H_{ij} = 1$ From bipartite graph $G = (V, U, \Lambda)$, one can easily visualize the corresponding parity-check matrix (adjacency matrix) H. *LDPC codes*.



(a)

Figure 14: [3,1,3] repetition code with parity check matrix (adjacency matrix) $H = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

LDPC codes A family of (l, q)-LDPC codes with column and row weights upper bounded by l and q. Either randomly producing column and row with wiughts set (l, q) and systematically modifying factor graph from a base code is possible to construct an (l, q) LDPC codes.

Edge-augmented LDPC codes Edge augmentation is a tool to create an LDPC code family from any "parent" factor graph $G = (V, U, \Lambda)$. **semitopological codes** are created by taking the hypergraph product of such augmented codes.

Connecting nodes with a single edge λ_{ij} in the parent code, the augmentation process involves adding a graph chain segment $G^g = \{V^g, U^g, \Lambda^g\}$. The adjacency matrix of the graph chain segment for $g = 4$ is

$$H^{g=4} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The form is constructed by adding a diagonal $g - 1$ entries to the identity matrix with dimension g. To paraphrase again and make it intuitively consider (iv_j, λ_{ij}) which naturally gives a matrix with $i \times j$ elements with u_i and v_j indicating the row and column indices and λ_{ij} determine the either being 0 or 1. Now, if we consider

$$G' = (V \cup V_g,\ U \cup U_g,\ \{\lambda_{ij}\} \cup \Lambda^g \cup \Lambda^w),$$

it is like we concatenate every data nodes v_j (column) and parity nodes u_i (row) (Graphically, every circles and squares) with additional two edges $\Lambda^w = \left\{\lambda^g_{1j}, \lambda^g_{ig}\right\}$ connected to $\{v_j,\, u^g_1\}$ and $\{v^g_g,\, u_i\}$ and nodes (defined as Λ^g which is graph chain segment.) , for example, $(1 \cup V^g, 2 \cup U^g, \Lambda\ \lambda_{ij} \cup \Lambda^g \Lambda^w)$

Therefore, the g-augmented factor graph $G^{*g} = (V^{*g},\, U^{*g},\, \Lambda^{*g})$ is obtained by edge-augmenting every edge of the parent graph $G = (V,\, U,\, \Lambda)$ with a length-g **graph chain segment**.

If the parent graph G corresponds to an $[n, k, d]$ code, then the augmented graph G^{*g} corresponds to an $[n + g|\Lambda|,\ k,\ d']$ code, where the new distance satisfies

$$d' \geq (1 + g\mu)\, d,$$

where $|\Lambda|$ is the number of all edges in the parent graph and μ is the minimum degree over all data nodes in G.

The resulting code rate is

$$R^{*g} = \frac{k}{n + g|\Lambda|} = \frac{R}{1 + g|\Lambda|/n},$$

which reflects the trade-off between increasing the code distance (from d to d') and decreasing the rate.

An exmaple of $(2,3)$-LDPC parent code with check matrix

$$H^{g=4} = \begin{pmatrix} 1 & 1 & & \\ & 1 & 1 & \\ & & 1 & 1 \\ & & & 1 \end{pmatrix}.$$

augmented with $g=1$, $g=2$ graph chain segment shows g-augmented graph G^{*1}, G^{*2} in Fig. [**?**] with code parameters $[9,2,6]$, $[15,2,10]$ It is easy to see that the check matrix shown her

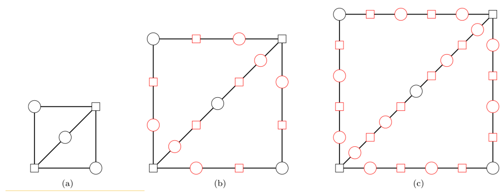


Figure 15: The left graph is the parent code. the red part correspond to $g=1$ and $g=2$ with check matrices being $\begin{pmatrix}1\end{pmatrix}$ and $\begin{pmatrix}1 & 1\\ 0 & 1\end{pmatrix}$, where Λ^w connected nodes $\{v_j,\, u_1^g\}$ and $\{v_g^g,\, u_i\}$ (connected to circle and square)

6.2 Quantum coding

The binary representation of a three-qubit Pauli operator $K = X_2Z_1$ (or error $(\mathbf{x},\mathbf{z})$) is $k = (010, 100)$. The CSS code checking matrix $H_{\text{CSS}} = \begin{pmatrix} H_Z & 0 \\ 0 & H_X \end{pmatrix}$. The corresponding quantum syndrome $\mathbf{s}_Q$ will be $\mathbf{s}_Q = (\mathbf{s}_X, \mathbf{s}_Z) = (H_Z \cdot \mathbf{x}, H_X \cdot \mathbf{z})$

CSS codes then can be described as two seperately classical codes $\mathcal{C}(H_Z)$ and $\mathcal{C}(H_X)$to detect X and Z type errors.

Hypergraph product codes$\mathcal{HGP}$. For a classical code n, k, d,HGP(C_H) is a CSS code with

$$H_X = \left(H \otimes I_n \mid I_m \otimes H^T\right), \qquad H_Z = \left(I_n \otimes H \mid H^T \otimes I_m\right).$$

The quantum code $\mathcal{HGP}(\mathcal{C}_H)$ parameters are

$$[[n^2 + m^2, k^2 + (k^T)^2, \min(d, d^T)]]$$

where H^T is the check matrix for code $[m, k^T, d^T]$.

6.3 Quantum LDPC codes

If the check matrix H_{CSS} has row and column weights bounded by l_Q and q_Q, then it defines a family of (l_Q, q_Q)-QLDPC CSS codes(Hypergraph product preserve the sparsity).

There are two important classes of hypergraph codes: (1) topological QLDPC codes or surface code family. (2) Random QLDPC codes which is constructed by randomly generated classical LDPC codes. The paper propose a new class of semitopological codes contructed by hypergraph product of augmented LDPC codes aims to combine both of its advantage. One can think of the surface-code family as having local stabilizers described by a sparse parity-check matrix, and these stabilizers form a CSS code defined by two matrices H_X and H_Z. Moreover, CSS codes naturally satisfy a chain-complex structure, which provides an intuitive way to view the surface code: the stabilizers arise from taking the boundary operators between faces, edges, and vertices of the underlying lattice. This correspondence between algebra and topology can also be extended to the notion of covering spaces, which likewise obey chain-complex relations. Such structure strengthens certain surface-code families [7] in their ability to support homomorphic logical measurements. To sum up, the paper [7] found an "analytic" way to systematically find the useful check matrix H_X and H_Z that are restricted in local stabilizers parity check matrix drived by topological intuition.

With random generated parity check matrix, is there an insight to construct "useful" parity matrices? Seems like [18] provides the answer.

Simply from Eq. 6.2, A (l_Q, q_Q)-QLDPC code family will have $l_Q = \max(2l, 2q)$ and $q_Q = l + q$.

Topological (4,4)-QLDPC codes. Each stabilizers are composed of four qubits and each qubit is involved in four stabilizers (With boundary, it might involve three or two qubits.). **hypergraph product** of $[n, 1, n]$ full-rank (The number of independent rows equals the number of rows. While $k = n - \text{rank}(H)$, hence parity check H is full rank.) repetition code gives an surface code with paramters $[[n^2 + (n-1)^2, 1, n]]$. Toric code $[2n^2, 2, n]$ is given by taking hypergraph product of ring code $[n, 1, n]$. The shortcoming of *Topological (4,4)-QLDPC codes* is the encoding rate scales poorly as $R = k/n \to 0$ as $d \to 0$ Noted that for CSS codes, we have relations $k = n - \text{rank}(H) = n - \text{rank}(H_X) - \text{rank}(H_Z)$.

Random QLDPC codes. The merit of random QLDPC codes is the finite encoding rate per block compared with surface code due to the usage distant stabilizers. **4-cycle** in tanner graph negatively impacts iterative decoding in error correction, therefore, have to be avoided using Mackay-Neal method in the paper [10] (might be involved in their provided Python package). The corresponding $(8, 7)$-QLDPC codes from $(3, 4)$-LDPC codes produced encoding rate $R = k/n = 0.4$. In realistic hardware and in codespace, typically require more qubits to realize distant interaction in stabilizer checks. A mean weight for random codes is 7 in $(8, 7)$-QLDPC codes, while $(4, 4)$-topological QLDPC codes are 4.

semitopological codes. As mentioned, semitopological codes are formed by the hypergraph product of augmented LDPC codes. In the paper, it is constructed by taking the hypergraph product from $(2, 3)$-LDPC codes with repetition code, which is the building blocks of surface code (local behavior). The codes combine the features of two types of codes. The comparisons can be seen in Fig. 16

$\mathcal{C}_H$	$\mathcal{C}_H^T$	$\mathcal{HGP}(\mathcal{C}_H)$	$R = k/n$	$\bar{w}$
[16, 4, 6]	$[12, 0, \infty]$	[[400,16,6]]	0.04	7.0
[20, 5, 8]	$[15, 0, \infty]$	[[625,25,8]]	0.04	7.0
[24, 6, 10]	$[18, 0, \infty]$	[[900,36,10]]	0.04	7.0

g	$\mathcal{C}_H^{*g}$	$(\mathcal{C}_H^{*g})^T$	$\mathcal{HGP}(\mathcal{C}_H^{*g})$	R	$\bar{w}$
0	[3,2,2]	[2,1,1]	[[13,5,2]]	0.385	5.00
1	[9,2,6]	[8,1,8]	[[145,5,6]]	0.0345	4.25
2	[15,2,10]	[14,1,14]	[[421,5,10]]	0.0119	4.14
3	[21,2,14]	[20,1,20]	[[841,5,14]]	0.00595	4.10
9	[57,2,38]	[56,1,56]	[[6385,5,38]]	0.000783	4.04

Figure 16: Tables of random LDPC codes and Semitopological codes performances, where $\mathcal{HGP}$ is the hyperproduct code and with the classical codes C, encoding rate R, and mean weights $\bar{w}$

6.4 Belief propagation decoding

Receiving a syndrome $\mathbf{f} = H \cdot \mathbf{e}$, we look for the maximum–likelihood error

$$\mathbf{e}_{\text{MW}} = \arg\max_{\mathbf{e}} P(\mathbf{e} \mid \mathbf{s}), \qquad \mathbf{e} = (e_1, e_2, e_3, \ldots),$$

which ranges over 2^n possible error strings. The marginal probability (sofe decisions) $P(e_i)$ is calculated to be

$$P(e_i = 1 \mid s) = \sum_{\substack{e_1,\ldots,e_n \\ e_i=1 \\ He=s}} P(e_1, \ldots, e_n \mid s).$$

and subsequently with final decoding estimate (hard decoders) by:

$$P^1(e_i) = \sum_{\sim e_i} P(e_1, e_2, e_i = 1, e_3, \ldots, e_n \mid s).$$

BP marginals perform less accurate with short loops in tanner graph and become better with long loops. BP decoder take check matrix H and $\mathbf{s}$ as input and also suffer from quantum degeneracy mainly due to the quantum codes seperate errors into cosets from stabilzier formalism but classical codes will have a nearly direct bijective relation between error and syndrome.

BP decoding of quantum codes:

(soft decoding (passing belief) –> hard decision –> check syndrome if $H\mathbf{e} = \mathbf{s}$)$^{\text{iterate unitil last step is satisfied}}$

In detail, BP algorithms is trying to compute fast for $P_1(\mathbf{e})$ first. We extracted the process from [19], while we follow notations from this paper might be different from the $BP - OSD$ paper.

Consider a $[7, 1, 3]$ Steane code shown in Fig. 17, μ_{v_i} represents each bit flip error for each qubit node v_i, and each check node c_i is connected to syndrome bit $s_{x,i}$. First, an initial LLR (log-likelihood ratio) of X_i (correspond to error on node)

$$\mu_{v_i} = \log \frac{\Pr(X_i = 1)}{\Pr(X_i = 0)} = \log \frac{p_x}{1 - p_x}.$$

is sent from each varaible/parity node v_i to check node/syndrome node c_j as $m^{(0)}_{v_i \to c_j} = \mu_{v_i}$.

Then each iteration contains two steps of propagating back and forth between v_i and c_i. The governing equations are as follows:

$$m^{(t)}_{c_j \to v_i} = (-1)^{s_{x,j}} \tanh^{-1}\left(\tanh\left(\prod_{v_k \in \partial c_j \setminus v_i} m^{(t)}_{v_k \to c_j}\right)\right).$$

$$m^{(t+1)}_{v_i \to c_j} = \mu_{v_i} + \sum_{c_k \in \partial v_i} m^{(t)}_{c_k \to v_i}.$$

where $s_{x,j} \in \{0, 1\}$ is the syndrome corresponding to Pauli Z measurement. ∂c_i and ∂v_i denotes nodes which is connected v_i and c_i.

After numerous iterations, we compute marginal belief/bias $m^{(t)}_{v_i}$ as

$$m^{(t)}_{v_i} = \mu_{v_i} + \sum_{c_k \in \partial v_i} m^{(t)}_{c_k \to v_i} \approx \log \frac{\Pr(X_i = 1 \mid S_x = s_x)}{\Pr(X_i = 0 \mid S_x = s_x)}.$$

.

and check for each syndrome measurement $\hat{x}^{(t)} H_1^T = s_x$. by using

$$x^{(t)}_i = \begin{cases} 0, & m^{(t)}_{v_i} > 0, \\ 1, & m^{(t)}_{v_i} \leq 0. \end{cases}$$

to chracterize errors $\hat{x}^{(t)} = \left(\hat{x}^{(t)}_1, \hat{x}^{(t)}_2, \ldots, \hat{x}^{(t)}_n\right)$. Noted that some notations and differnet from the essential main paper I refer to due to definitions of probability.

Qrdered statistics decoding:

Going back to the main paper, it aims to show that BP–OSD also applies to toric codes and semitopological codes, in addition to random QLDPC codes, with the corresponding open-source implementation available on GitHub.

`https://github.com/quantumgizmos/bp_osd`

In this section, the authors describe OSD as a classical decoding procedure that can be applied equally well to the H_X and H_Z parity-check matrices of a CSS code.

Method:

For a $m \times n$ matrix H, we will always have $\operatorname{rank}(A) \leq \min(m, n)$.

There is no full column rank for $LDPC$ codes check matrix H as $n > m$ which can make $H^{-1} \cdot \mathbf{s} = \mathbf{e}$ impossible. We then find subset of columns $[S]$ which is linearly independent and with associated submatrix $H_{[S]}$ with full column rank. Then the submatrix $H_{[S]}$ can be invertible following $H^{-1}_{[S]} \cdot \mathbf{s} = \mathbf{e}_{[S]}$.

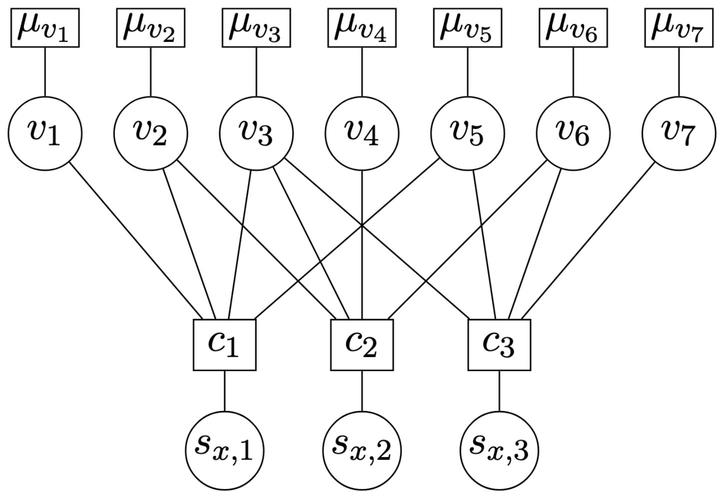


Figure 1: Tanner graph of H_1 for the $[[7,1,3]]$ Steane code

Figure 17: Tanner graph for $[7,1,3]$ Steane code H

Noted that different choice of basis $[S]$ will lead to differnet unique solution $\mathbf{e}_{[S]}$ which could eliminate possible quantum degeneracy.

OSD post-procesisn method is by modifying soft decisions in BP to obtain the basis set [S] (qubits that involved in errors) with the highest probability of error occurs.

OSD-0 algorithm: Belown show the following steps for BP-OSD when BP decoder fails.

(1) use BP soft decision form vector $P_1(\mathbf{e})$ to order the qubits which are being flipped or not in terms of probability, $[O_{BP}]$.

(2) Then rearrange the check matrix $H \rightarrow H_{[O_{BP}]}$ which is the matrix following the order of O_{BP}

(3) Use the first RANK(H) linearly independent columns of $H_{[O_{BP}]}$ as $H_{[S]}$

(4) Calculate $\mathbf{e}_S = H^{-1}_{[S]} \cdot \mathbf{S}$, which is the OSD-O solution.

(5) The OSD-O solution can then be expressed as $\mathbf{e}_{S,T} = (\mathbf{e}_{[S]}, \mathbf{e}_{[T]}) = (\mathbf{e}_{[S]}, 0)$ while $[T] \notin [S]$. Then above gurantees $H_{[S,T]} \cdot \mathbf{e}_{[S,T]} = \mathbf{s}$ (while $H_{[S,T]}$ is the original H in (S,T) order)

(6) Mapping back to the original ordering, $\mathbf{e}_{[S,T]} \rightarrow \mathbf{e}_{OSD-0}$

It is like modifying check matrix from the original to improve the part where original BP will fail to predict.

Higher-order OSD: Considering where $e_T \neq 0$. While perfomring the 6 steps as in BP-OSD-0. They now give solution

$$\left(\mathbf{e}_{[S,T]} = H^{-1}_{[S]}\,\mathbf{e}_{[S]} + H^{-1}_{[S]}\,H_{[T]}\,\mathbf{e}_{[T]}, \mathbf{e}_{[T]}\right)$$

while simply $H_{[S,T]} \cdot \mathbf{e}_{[S,T]}$ will give $\mathbf{e}_{\mathbf{s}} + 2H_{[T]} \cdot \mathbf{e}_{[T]} = \mathbf{s}$ for all possible $\mathbf{e}_{[T]}$.

In principle, there are $2^{n-\mathrm{RANK(H)}}$ configurations for $\mathbf{e}_T$, which gives difficulty to find the lowest wieght of $\mathbf{e}_{S,T}$. However, combine with BP-OSD-0 ordering for matrix H. The task can be facilitated by applying weighted greedy search routine which prioritizes the more probable configurations of $\mathbf{e}_T$ according to the soft decisions $P^1(\mathbf{e})$

Greedy searching strategies for higher-oder OSD:

For the numerical simulations, a strategy steps are listed as follows:

(1) As mentioned, $\mathbf{e}_T$ is put in the order decided by BP soft decisions.

(2) Search over all weight-one configurations of $e_{[T]}$

(3) And search for weight-two configurations in the first λ bits. This combined with weight-one searching lead to $n - \mathrm{RANK(H)} + \begin{pmatrix} \lambda \\ 2 \end{pmatrix}$

The decoders using this combination sweep greedy search algorithm is called $BP + OSD - CS$. The simulations in this paper use parameter $\lambda = 60$.

6.5 Numerical Simulations

Simulation methodology for BP+OSD decoding. If consider symmetric hypergraph product code, the decoding problems for X and Z type errors are the same.

The residual error is given by $\mathbf{x}_R = \mathbf{x} + \mathbf{x}_{\mathrm{OSD(BP)}}$. By checking the commutation relations with logical Z operator sets $L_Z : L_Z \cdot \mathbf{x}_R = \mathbf{0}$. it shows zero when it $\mathbf{x}_R$ and L_Z commutes, and with possible elemens being one if anticommute. Noted that errors anticommutes with stabilizers and stabilizers commutes with logicals. Correctable errors commute with logicals, while one can think of correctable errors do not change the subspace outside the encoding qubit. While errors anticommute with logicals is itself logicals up to possible stabilizers.

Topological QLDPC codes

Below error threshold, you can decrease logical error rate by increasing code distance or code concatenation.

The papers aims to conctruct a new type of codes which could have finite encoding rate and have low logical error rate and high error threshold.

7 Notes on Fault-Tolerant Belief Propagation for Practical Quantum memory [9]

A fault-tolerant belief propagation decoder that utilise a **sapce-time tanner graph** across **multiple rounds of syndromes measurement** with mixed alphabet error variables.

(1) **Probabilistic error consolidation** to mitigate degeneracy effects and short cycles

(2) Adaptive sliding window that can capture long error events.

(3) high error thresholds of $0.4\% - 0.87\%$ and strong error-floor performance for various types of topological codes.

(4) Reduction of generalized check matrix for computation complexity.

(5)The fundamental purpose of generalized check matrix is by adding temporal and genral faults (circuit-level) in to the matrix. If standard raw syndrome do not involve temporal and fault-tolerance. Hence the design may not directly tackle temporal and fault-tolerance.

(6) Above temperal addition make them possible to connect variable node connects to at most two rounds of check nodes. This extend each stabilizer lives in each stabilizer. They can termporally addind effects to refine the code. This might make errors propagate through rounds, but overall the effects are better.

(7) They merge errors temporally if with two intuition: sparse matrix computation or when errors are degenerate (errors consolidation). Noted that this does not change the real circuits but computational complexity. However, computational complexity do benefit real hardware fidelity if the decoding time can be lower. The computation can then carry on if decoding process is fast. While adaptive sliding window and their circuit-level choice do benefit real physics in a more fundamental way. Since it is about adding more degrees of freedom to change the system.

(8) Qubit *rightarrow location* in this paper. For examply, there many locations for an error $E = (E_1, E_2, ...)$. However, E_1, E_2, E_3, E_4 correspond to different errors in the first qubit.

(9) Representing errors in terms of binary bits, which requires 4-bit strings in circuit level model. The real computations involve enlarging locations into column vectors.

(10) These all process, first starting from quantum computation, suddenly, we are trying to decode the circuits. First, they apply decoders on not one round but three round in a role. This process also includes circuit-level model and allow temporal connection between nodes and checks, called space-time tanner graph.

So this "choice" of decoding instead of each round give them degree of freedom to optimize the decoding process, which could be more efficient, for example, adding temporal checks between different rounds. Now, we are enterring the decoding process, not only merging errors, they also contructed careful intuition on how to decompose errors and observe thier degeneracy, by merging degeneracy, the decoder will become more efficient. At last, move into a larger picture when we are performing quantum error correction, they further improve the standard sliding window to adaptive sliding window, which further improve the decoding performance, this could lead to a high threhold. Here I have think of something, if we can follow the same temporal analogy like syndrome extraction, we memorize the adpative syndrome, and we do not correct it in each adaptive sliding window, we are further optimizing the overall error threhold in a even higher percentage. Since correction also includes time. is it like passive error correction. The sliding window process involve multiple syndrome measurements, all of them but not just the last one are used in BP to find good correction.

Toric code is LDPC codes but not QLDPC codes, while QLDPC codes should have finite encoding rate.

7.1 Introduction

Typically, a code distance d is protecting the quantum memory but requires $O(d)$ rounds of syndrome measurements resulting in $O(d^3)$ potential error locations for a code length $O(d^2)$ ($O(d^2)$ qubits).

A decoder with complexity nearly in linear with the number of error variables. Apart from perfect error syndromes, quatnum data-syndrome codes, and the phenomenological noise model. Here introduce a fault-tolerant belief propagation (FTBP) for general QLDPC, CSS, and no-CSS codes under noise model.

By incorporating both spatial and temporal correlations, this sparse graph representation allow more effective error correction. This also mitigates effect when gates are supressed by probability p realistic spontaneous emission error which is related to time.

Noted that errors related by stabilizers must be in same syndrome and errors note related by stabilzers can also be in same syndromes.

Two-qubit gates introduces in the method cause more short cycles. In this paper, **probabilistic error consolidation** is introduced. The process put degenery errors and errors with same residue errors into a new single representative set. This process decouples higher- order error variables into lower-order ones for probabilistic consolidation. This can reduce short cycles in the Tanner graph.

In addition, they introduce dynamic sliding window to make widow offset being effective and efficient. Noted that non-overlapping sliding window in temporal majority voting is useless while (W, W) is same as $(1, 1)$ but with overlapping (W, F) could help inprove code distance [8].

FTBP applies to general QLDPC codes and with nearly linearly scaling results in $O(d^3 \log(d))$ computationally intensive for a a code of distnace d.

7.2 Quantum stabilizer codes and circuit-level noise model

7.3 generalized check matrix for syndrome extraction circuit and circuit-level decoding problem and FTBP and sparse generalized check matrix for space-time Tanner graph and error merging and probabilisitc error consolidation

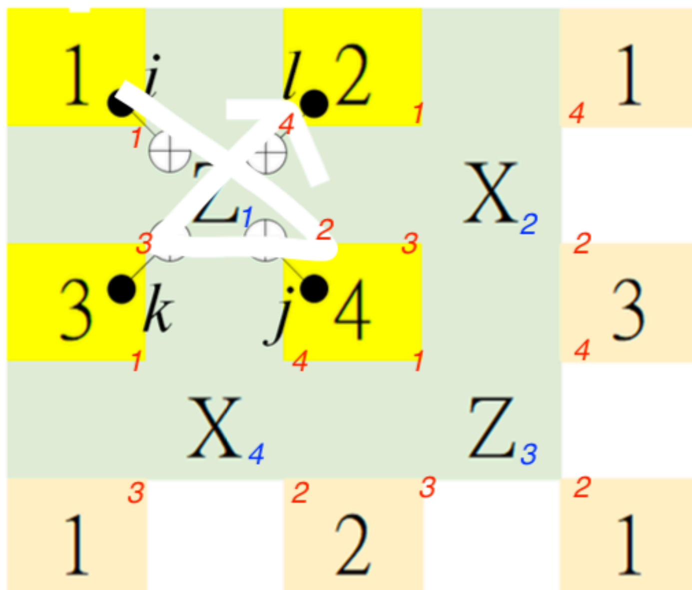
Stabilzer S is abelian group. Degenerate errors can be thought of as the elements in the cosets of same syndrome measurement results.

With multiple rounds of raw syndrome extraction for codes $[[d^2, 2, d]]$ rotated toric codes (with even d) [**?**, **?**], the $[[\frac{9}{8}d^2, 4, d]]$ rotated 6.6.6 toric color codes (with d a multiple of 4) [**?**, **?**], and the $[\![\frac{d^2+1}{2}, 1, d]\!]$ twisted XZZX toric codes (with odd d) [**?**].

Lattice representation of $[[d^2, 2, d]]$ for $d = 2$. The entangling gate in order $(i, j, k, l) = (1, 4, 3, 2)$ for ancilla 1 withn in a Z type stabilizers ((2, 3, 4, 1) for ancilla 2) which is shown in . 18) .But each stabilizer $S_1, S_2, ...$ can be performed in parallel.

Circuit-level noise model

-errors $\{I, X, Y, Z\}$



(a)

Figure 18: Blue numbers are the ancilla order, and red numbers are the entangling gate order. This involve four depths.

Assuming a circuit-level noise model in the quantum memory, each potential error source in the syndrome extraction circuit is referred to as a *location*. After perfect ancilla preparation, it experiences subsequent bit-flip or phase flip errors. Additionally, idle qubits will also sufferred from Pauli errors which have to be taken into account.

7.4 Generalized check matrix

a M syndrome bits $s = \mathfrak{M} = \mathcal{H}\star E$, where $\mathcal{H}$ is with dimension M (Belief propagation decoding of quantum LDPC codes with N(N locations) as generalized check matrix.

$$E \star F = \sum_{k\colon E_k = D_i \text{ or } C_{ij}} E_k * F_k \; + \sum_{k\colon E_k = b_i \text{ or } m_i} E_k \cdot F_k \pmod 2.$$

Here, H, E, and F take values in a mixed alphabet

$$\{I, X, Y, Z\}, \quad \{I, X, Y, Z\}^2, \quad \text{and} \quad \{0, 1\},$$

corresponding to the four types of circuit-level errors we consider. Specifically, the ancillary qubit preparation error is given by

$$b_i \in \{0, 1\},$$

which corresponds to either $\{I, X\}$ or $\{I, Z\}$. The CZ or CNOT gate error acting on ancilla i and data qubit j is represented by

$$C_{ij} \in \{I, X, Y, Z\}^2.$$

The idle-qubit error on data qubit i is denoted by

$$D_i \in \{I, X, Y, Z\},$$

and the measurement error at ancilla i is specified by

$$m_i \in \{0, 1\}.$$

The result is a binary syndrome $s \in \{0, 1\}$, with

$$E * F = \begin{cases} 0, & \text{if } E \text{ and } F \text{ commute,} \\ 1, & \text{if } E \text{ and } F \text{ anticommute.} \end{cases}$$

$$E \star F = \sum_{k\colon E_k = D_i \text{ or } C_{ij}} E_k * F_k \; + \sum_{k\colon E_k = b_i \text{ or } m_i} E_k \cdot F_k \pmod 2.$$

Here H, E, F take values in a mixed alphabet $\{I, X, Y, Z\}$, $\{I, X, Y, Z\}^2$, and $\{0, 1\}$. The result is a binary syndrome $s \in \{0, 1\}^M$.

$$E * F = \begin{cases} 0, & \text{if } E \text{ and } F \text{ commute,} \\ 1, & \text{if } E \text{ and } F \text{ anticommute.} \end{cases}$$

Proposition 1. The generalized check matrix H is constructed as follows:

1. If location k corresponds to an ancilla preparation error b_i or a measurement error m_i, then the k-th column of H lies in $\{0, 1\}^M$ and is

$$H_k = \mathfrak{M}(\{E_k = 1\}) = (0, 0, 0, 0, .., E_k = 1, ...0).$$

This is kth role of the check matrix.

2. If location k corresponds to an idle qubit error D_i, then the k-th column of H lies in $\{I, X, Y, Z\}^M$. Since both X and Z syndromes must be captured, set

$$H_k = Z^u X^v, \qquad u = \mathfrak{M}(\{E_k = X\}), \quad v = \mathfrak{M}(\{E_k = Z\}),$$

where

$$u = \{u_1, u_2, \ldots, u_M\}, \qquad v = \{v_1, v_2, \ldots, v_M\}.$$

3. If location k corresponds to a CZ or CNOT gate error C_{ij}, then the k-th column of H lies in $\{I, X, Y, Z\}^{M \times 2}$. The relevant error bases are

$$u = \mathfrak{M}(\{E_k = X \otimes I\}), \qquad v = \mathfrak{M}(\{E_k = Z \otimes I\}),$$
$$u' = \mathfrak{M}(\{E_k = I \otimes X\}), \qquad v' = \mathfrak{M}(\{E_k = I \otimes Z\}).$$

Thus,

$$H_k = \begin{pmatrix} Z^{u_1} X^{v_1} & Z^{u'_1} X^{v'_1} \\ \vdots & \vdots \\ Z^{u_M} X^{v_M} & Z^{u'_M} X^{v'_M} \end{pmatrix},$$

which is interpreted as a length-M vector over $\{I, X, Y, Z\}^2$.

The above relation are utilised to construct "generalzied parity check matrix" which is shown in Fig. 19 with raw check matrix being $\begin{bmatrix} Z & Z \\ X & X \end{bmatrix}$

(a)

$D_1^{(1)}$	$D_2^{(1)}$	$C_{1,1}^{(1)}$	$C_{2,2}^{(1)}$	$C_{1,2}^{(1)}$	$C_{2,1}^{(1)}$	$m_1^{(1)}$	$m_2^{(1)}$	$b_1^{(1)}$	$b_2^{(1)}$	$D_1^{(2)}$	$D_2^{(2)}$	$C_{1,1}^{(2)}$	$C_{2,2}^{(2)}$	$C_{1,2}^{(2)}$	$C_{2,1}^{(2)}$	$m_1^{(2)}$	$m_2^{(2)}$	$b_1^{(2)}$	$b_2^{(2)}$	$D_1^{(3)}$	$D_2^{(3)}$	$C_{1,1}^{(3)}$	$C_{2,2}^{(3)}$	$C_{1,2}^{(3)}$	$C_{2,1}^{(3)}$	$m_1^{(3)}$	$m_2^{(3)}$	$b_1^{(3)}$	$b_2^{(3)}$
Z	Z	ZI	IZ	ZI		1		1																					
X	X	IX	XI		XI		1		1																				
Z	Z	IZ	ZZ	IZ	IZ					Z	Z	ZI	IZ	ZI		1		1											
X	X	XX	IX	IX	IX					X	X	IX	XI		XI		1		1										
Z	Z	IZ	ZZ	IZ	IZ					Z	Z	IZ	ZZ	IZ	IZ					Z	Z	ZI	IZ	ZI		1		1	
X	X	XX	IX	IX	IX					X	X	XX	IX	IX	IX					X	X	IX	XI		XI		1		1
Z	Z	IZ	ZZ	IZ	IZ					Z	Z	IZ	ZZ	IZ	IZ					Z	Z	IZ	ZZ	IZ	IZ				
X	X	XX	IX	IX	IX					X	X	XX	IX	IX	IX					X	X	XX	IX	IX	IX				

Figure 19:

The circuits for three rounds of raw syndrome extraction of check matrix $\begin{bmatrix} Z & Z \\ X & X \end{bmatrix}$ and corresponding error variables is shown in Fig. 20.

Let $R(\mathcal{M}, E) \in \{I, X, Y, Z\}^n$ denote the residual Pauli error on the data qubits of the n-qubit code after the execution of circuit $\mathcal{M}$.

Circuit-level decoding problem

Given a syndrome extraction circuit $\mathfrak{M}$, syndrome bits $s = \mathfrak{M}(E) \in \{0, 1\}^M$, find an estimate $\hat{E}$ such that

$$\mathfrak{M}(\hat{E}) = s$$

and

$$R(\mathfrak{M}, \hat{E})\, R(\mathfrak{M}, E)$$

is correctable by perfect QEC.

Definition 3

After building stabilizers, they constructed error vector E from the following settings:

1) Each single-qubit Pauli variable D_j is independently generated with depolarizing rate $\epsilon \in [0, 3/4)$, following the distribution $(p_j^I, p_j^X, p_j^Y, p_j^Z) = (1 - \epsilon, \epsilon/3, \epsilon/3, \epsilon/3)$.

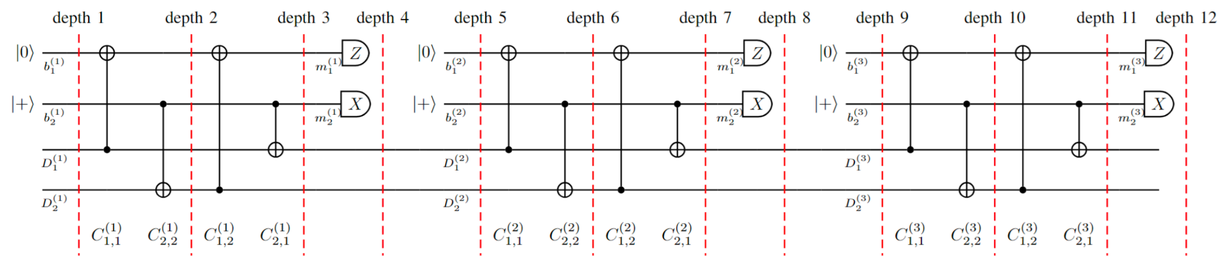


Figure 20:

2) Each two-qubit Pauli variable C_{ij} is independently generated with depolarizing rate $\epsilon \in [0, 3/4)$, such that $C_{ij} = I$ with probability $1 - \epsilon$ and C_{ij} is a non-identity two-qubit Pauli with probability $\epsilon/15$.

3) Each syndrome bit error m_j or ancillary preparation error b_j is an independent bit-flip or phase-flip error with rate $\epsilon_b \in [0, 1/2)$, following the distribution $(q_j^{(0)}, q_j^{(1)}) = (1 - \epsilon_b, \epsilon_b)$.

The log-likelihood ratio then can be defined consequently based on above.

In addition, the Tanner graph has N variable nodes corresponding to $E_1, \ldots, E_N$ and M check nodes corresponding to the rows of H. The neighbors of a check node i are

$$N(i) = \{\, j : H_{ij} \neq I \text{ or } H_{ij} \neq 0 \,\},$$

Here j mean collect variable nodes (in index j) directly connected to check node i. For exmaple, $N(i)$ is the set such that $H_{i1} = X$, $H_{i2} = Z, \ldots$

The neighbors of a variable node j are

$$M(j) = \{\, i : H_{ij} \neq I \text{ or } H_{ij} \neq 0 \,\}.$$

This shows trivial operator in $\mathcal{H}$ for identity I and 0 actually do not contribute to "stabilize", therefore no need to perform entangling gates corresponding to this entry.

Belief propagation generates LLR vectors $\Gamma_1, \ldots, \Gamma_N$ for estimating the error variables. The FTBP algorithm performs binary, quaternary, and 16-ary message computations.

FTBP Algorithm

Variable-to-check (V-to-C): $\Gamma_{j \to i}$ Check-to-variable (C-to-V): $\Delta_{j \to i}$

If the syndrome does not match, then the process continue until the maximum of $T_{\max}$ iterations.

We first quickly summarize the main differnce between FTBP and standard BP:

(1)Space–time Tanner graph while standard BP is spatial only.

(2)FTBP variable update (Pauli case): instead of Z or X type error, here with full circuit level error which would include 2/4/16-ary instead of all 2 ary, since we have to characterize errors like two-qubit errors. A matrix as shown in Fig. 19 shows this for label C denoting rows and columns.

(3) Horizontal step do not change from BP

(4) Vertical Step (Marginal Distribution Part):

$$\Gamma_j^W = \Lambda_j^W + \frac{1}{\alpha} \sum_{\substack{i \in \mathcal{M}(j) \\ W \star H_{ij} = 1}} \Delta_{i \to j}, \qquad W \neq I.$$

This is the quantum twist.

For each Pauli $W \in \{X, Y, Z\}$:

- Add only the check messages for checks where $W \in \{X, Y, ZXI, IX, XY, XZ, ..\}$ anticommutes with the check-matrix entry H_{ij}. In fact, this can be thought of as if error is X and the parity check matrix element are also X, then this do not cause an errror. But in classical BP, while there are just $0 \cdot 0$, $0 \cdot 1$, $1 \cdot 0$, $1 \cdot 1$, which naturally can be expressed as

Because anticommute = 1 in the syndrome.

7.5 BP complexity discussion for circuit-level decoding problem

Window W size is smaller to reduce generalizefd check matrix. A n qubit stabilizer code with m stabilizers with weight w requires mw CNOT or CZ gates, and m ancilla parepations. The deo=coding procedure is applied to r rounds of syndrome extraction. The generalized matrix will then be $rm \times r(n + m\omega + 2m)$ (where n qubits plus possible errors on m ancilla preparations and m measurements and $m\omega$ gates)

Lemma

Reduce the complexity of $FTBP$ generalized check matrix. The complexity of FTBP for an n-qubit quantum code with O(1) stabilizer weights and a window of O(√n) rounds of syndrome extraction is O(n1.5 log n), achieved using a sparse generalized check matrix. Additionally, each variable node connects to at most two rounds of check nodes in the corresponding Tanner graph.

7.6 Probabilistic error consolidation

The trick is to changing the input of errors that could combine degenerate errors into one with two times probability and the other degenerate one being zero. Noted that errors vector and check matrix are invertible. So labeling degenerate errors on parity check elements is like telling you which type of error should be merged.

7.7 adaptive sliding window

7.8 simulations

8 Notes on Dan Browne topological codes

8.1 Toric codes

all boundaries are cycles as $\partial_{n+1}\partial_n = 0$

Boundaries $B_n = \operatorname{im} \partial_{n+1}$ **Cycles (closed loops)** $Z_n = \ker \partial_n$

$$B_n = \operatorname{im}(\partial_{n+1}), \qquad Z_n = \ker(\partial_n), \qquad B_n \subseteq Z_n.$$

In the toric code, $Z_1 = \ker(\partial_1)$ is the group of 1-chains with zero boundary, (Mathematically, centralizer of stabilzer group) pictorially it can be thought of as all closed loops that pass through X stabilizers (vertices), this form sets of logical Z and other closed loops formed by Z stabilizers (when pass through $Z_1 = \ker(\partial_1)$ $= Z_1 = \ker(H_x)$, it means formed by Z stabilizers, and vice versa, one can pictorially imagine that.) In the chain complex, this is consistent with the relation $\partial_2 \circ \partial_1 = 0$, which ensures that every boundary is a cycle. However, there are loops that are not contractible, represented by $\ker(\partial_1)/\operatorname{im}(\partial_2)$, pictorially $\operatorname{im}(\partial_2)$ is all loops that can be formed by stabilizers multiplication, and $\ker(\partial_1)/\operatorname{im}(\partial_2)$ satisfies $\ker(\partial_1) - \operatorname{im}(\partial_2) \subset \operatorname{im}(\partial_2)$. Or simply thinking, make group generated by LS, S, L with $S = I$, then the remaining will be only logicals L.

Noted that $\ker(\partial_1)/\operatorname{im}(\partial_2)$ characterize Z logical and $\ker(\partial_2)/\operatorname{im}(\partial_1)$ characterize X logical, pictorially and algebarically imagine no trivial commuted with their dual types of operators.

This can further deduce, that Z logicals lies on the "edges" and X logicals lies on the "faces" as shown in Fig. 21 toric codes pictorial interpretations.

Two error models: **depolarizing**: each qubit indepently suffer from three types of error and **uncorrelated model**: usually problems can be seperated into X and Z type decoding problems.

Minimum weight checking: graph characterize the weights between syndrome measurement that show errros.

Fig. 22 shows a process when decoding a syndrome measurement based on MWPM.

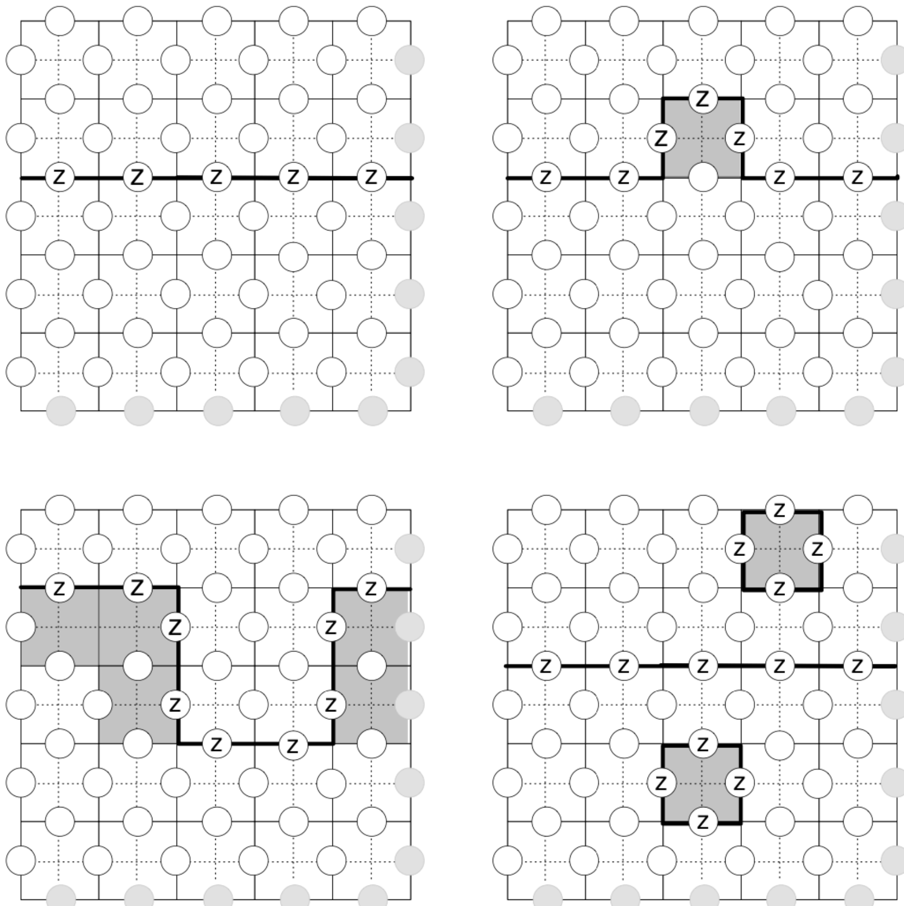


Figure 21:

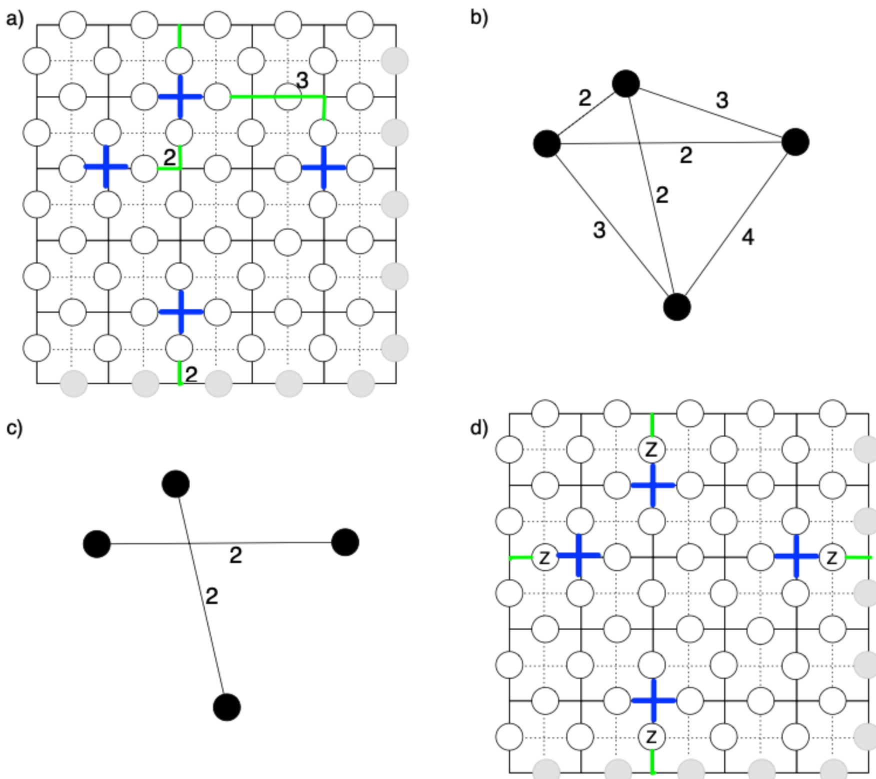


Figure 22:

The runtime scales with V^3 V is the vertices of the graph shown in Fig. 22(b).

8.2 Elements of Topology and Homology

Homeomorphism: an object can include stretching, bending, but not (in general) tearing or glueing. Formally: Continous mapping between two objects or topology: continous, bijective, continous inverse function. Function mapping is shown in Fig. 23.

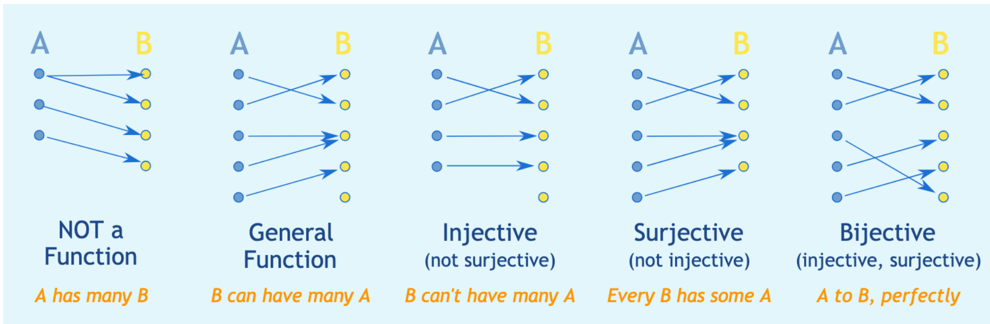


Figure 23:

Cellulation (CW complex): Euler characteristic of any cellulation of the sphere is 2, and that more generally, for any surface, χ is independent of the cellulation chosen and is a topological invariant.

The surface of any convex polyhedron is homeomorphic to a sphere.

References

[1] https://sites.google.com/site/danbrowneucl/teaching/lectures-on-topological-codes-and-quantum-computation.

[2] Quantum error correction below the surface code threshold. *Nature*, 638(8052):920–926, 2025.

[3] Anthony Bosman. Algebraictopology, 2023.

[4] Austin G Fowler, Matteo Mariantoni, John M Martinis, and Andrew N Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A—Atomic, Molecular, and Optical Physics*, 86(3):032324, 2012.

[5] Shilin Huang. Homomorphic logical measurements | qiskit seminar series with shilin huang. 2023.

[6] Shilin Huang, Kenneth R. Brown, and Marko Cetina. Comparing shor and steane error correction using the bacon-shor code. *Science Advances*, 10(45):eadp2008, 2024.

[7] Shilin Huang, Tomas Jochym-O'Connor, and Theodore J Yoder. Homomorphic logical measurements. *PRX Quantum*, 4(3):030301, 2023.

[8] Shilin Huang and Shruti Puri. Improved noisy syndrome decoding of quantum ldpc codes with sliding window. *arXiv preprint arXiv:2311.03307*, 2023.

[9] Kao-Yueh Kuo and Ching-Yi Lai. Fault-tolerant belief propagation for practical quantum memory. *arXiv preprint arXiv:2409.18689*, 2024.

[10] David JC MacKay and Radford M Neal. Near shannon limit performance of low density parity check codes. *Electronics letters*, 32(18):1645–1646, 1996.

[11] Pavel Panteleev and Gleb Kalachev. Degenerate quantum ldpc codes with good finite length performance. *Quantum*, 5:585, 2021.

[12] Lukas Postler, Sascha Heuβen, Ivan Pogorelov, Manuel Rispler, Thomas Feldker, Michael Meth, Christian D Marciniak, Roman Stricker, Martin Ringbauer, Rainer Blatt, et al. Demonstration of fault-tolerant universal quantum gate operations. *Nature*, 605(7911):675–680, 2022.

[13] Joschka Roffe, David R White, Simon Burton, and Earl Campbell. Decoding across the quantum low-density parity-check code landscape. *Physical Review Research*, 2(4):043423, 2020.

[14] Jean-Pierre Tillich and Gilles Zémor. Quantum ldpc codes with positive rate and minimum distance proportional to the square root of the blocklength. *IEEE Transactions on Information Theory*, 60(2):1193–1202, 2013.

[15] Andre Van Rynbach, Ahsan Muhammad, Abhijit C Mehta, Jeffrey Hussmann, and Jungsang Kim. A quantum performance simulator based on fidelity and fault-path counting. *arXiv preprint arXiv:1212.0845*, 2012.

[16] Ye Wang, Stephen Crain, Chao Fang, Bichen Zhang, Shilin Huang, Qiyao Liang, Pak Hong Leung, Kenneth R Brown, and Jungsang Kim. High-fidelity two-qubit gates using a microelectromechanical-system-based beam steering system for individual qubit addressing. *Physical Review Letters*, 125(15):150505, 2020.

[17] Qian Xu, J Pablo Bonilla Ataides, Christopher A Pattison, Nithin Raveendran, Dolev Bluvstein, Jonathan Wurtz, Bane Vasić, Mikhail D Lukin, Liang Jiang, and Hengyun Zhou. Constant-overhead fault-tolerant quantum computation with reconfigurable atom arrays. *Nature Physics*, 20(7):1084–1090, 2024.

[18] Qian Xu, Hengyun Zhou, Guo Zheng, Dolev Bluvstein, J Pablo Bonilla Ataides, Mikhail D Lukin, and Liang Jiang. Fast and parallelizable logical computation with homological product codes. *Physical Review X*, 15(2):021065, 2025.

[19] Hanwen Yao, Waleed Abu Laban, Christian Häger, Alexandre Graell i Amat, and Henry D Pfister. Belief propagation decoding of quantum ldpc codes with guided decimation. In *2024 IEEE International Symposium on Information Theory (ISIT)*, pages 2478–2483. IEEE, 2024.

[20] Yi-Cong Zheng, Ching-Yi Lai, and Todd A Brun. Efficient preparation of large-block-code ancilla states for fault-tolerant quantum computation. *Physical Review A*, 97(3):032331, 2018.