$Prelude; Classical error-correcting code Lecture 1 (2h)

$$Repetition code as a ferromagnet

Repetition code interpretation/realization

Before discussing quantum codes, let us recall the physical ~~meaning~~ of classical codes. A prime example is the repetition code which encodes 1 logical bit into n physical bits.

$0 \to 00\,0\cdots 00$

$1 \to 11\,1\cdots 11$

This encoding is robust against bit flip errors. When less than a half of bits were flipped, one can still recover the original information 0 or 1.

In the coding theory language, this is because the Hamming distance (number of bits with different entries) between two codewords 0000… and 1111… is large (n, the number of physical bits). This example encodes only 1 bit, but we generally want to encode k logical bits (with large k) into n physical bits while keeping the code distance (the minimal Hamming distance between pairs of codewords) large. That is, we hope to have [n,k,d] with large k and d.

Decoding

Another important feature of a code is its decodability. In the repetition code, we can easily recover the initial information by counting the total number of 0s and 1s (majority vote). In general, however, decoding can be computationally challenging. We want to construct a code with efficient decoders, which is the holy grail of classical coding theory.

Hamiltonian

The repetition code can be physically realized as a ground state of a simple local spin Hamiltonian, a ferromagnet, or Ising model. The Hamiltonian and ground states are given by

$H = -\sum_{i,j} Z_i Z_j \qquad gs = |00\cdots 0\rangle, |11\cdots 1\rangle$

In physics language, we are simply encoding 0 and 1 into repetition of up and down spins, in other words, in a giant magnet. This is essentially how we encode classical bits in our HDD memory (in a very simplified language). Decoding can be done by simply measuring the total magnetization:

$M = \sum_j Z_j$

Phase error

The codeword space is defined by the space of ground states. If we were to use the ferromagnet as a quantum code, then we would be encoding a superposition state as follows

$|0\rangle + |1\rangle \to |00\cdots 00\rangle + |11\cdots 11\rangle \qquad |0\rangle - |1\rangle \to |00\cdots 00\rangle - |11\cdots 11\rangle$

These states are often known as GHZ states or cat states, being a macroscopic superposition of classical states. Here bit flip errors corresponds to Pauli X operators. In a quantum system, phase errors, corresponding to Pauli Z operators may also occur. Here we see that the above two states are connected by a single Pauli Z error

$Z_1 \otimes I_2 \otimes \cdots \otimes I_n (|00\cdots 00\rangle + |11\cdots 11\rangle) = |00\cdots 00\rangle - |11\cdots 11\rangle$

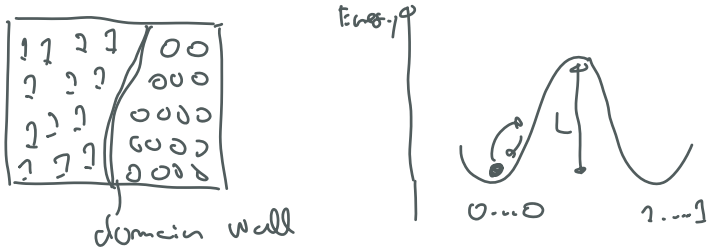
so the information can be lost by damaging only a single qubit. So, unfortunately this system cannot protect a quantum information.

Memory time

One another important physical question concerns how long one can retain a classical memory. In the repetition code, suppose bit flip errors occur randomly at each spin with some finite probability p per 1 second. Then, after ~1/p seconds, most of spins will be flipped at least once, and the original information is no longer recoverable. Hence, one needs to keep performing error-correction before it's too late, namely every 1/p seconds.

Thermal noises

But in actual physical systems, errors do not occur randomly. Namely, in a Hamiltonian system coupled with thermal bath, errors which increase the energy of the system are not very likely to occur. In the 2D Ising Hamiltonian, let us consider transforming 0000 state into 1111 state by flipping spins one by one. One then notices that there will be a domain wall separating regions with 0 and regions with 1. The Hamiltonian assigns excitation energy to mismatch of neighboring spins. Since the length of the domain wall will be at least L, in order to transform 0000 state into 1111 state, the system needs to climb a large energy barrier of height L.



At low temperature, thermal fluctuations cannot cross a large energy barrier (unless there is some entropic enhancement from other local minima, which is the case in some of glassy models).

For 2D Ising model, there is a finite critical temperature, and below that, the classical memory time scales exponentially with respect to the system size.

$$T_c \sim \frac{2J}{k_B \ln(1+\sqrt{2})} \qquad T_{memory} \sim \exp(\text{const.} \cdot L) \quad T < T_c$$

Here Tc can be analytically computed for Ising model. For a certain model of stochastic thermal noises, the memory time can be rigorously lower bounded. Lectures on Glauber dynamics for discrete spin systems (Fabio Martinelli)

Self-correction

The 2D Ising model is often called self-correcting since the thermal fluctuation tends to return the system to neighbors of the original state. The upshot is that a ferromagnet is stable at finite temperature, and that's why we use it everyday. One open problem is whether such a self-correcting quantum memory exists or not. There is a no-go theorem, which we will prove in later lectures, ruling out 2D self-correcting stabilizer code. 4D generalization of the Toric code is a self-correcting quantum memory. At this moment, there is no known 3D self-correcting memory. Some of the fracton models are "marginally" self-correcting, in a sense that the memory size increase can persist up to some length scale, but then the growth stops there. Recent status is summarized in a review.

$$Quantum repetition code does not exist

One might consider trying the same strategy to securely encode quantum information.

$$|\psi\rangle \to |\psi\rangle \otimes |\psi\rangle \otimes \cdots \otimes |\psi\rangle \quad ?$$

However, a quantum repetition code is prohibited because we are not allowed to clone a quantum state due to the no-cloning theorem. So, we will need a different strategy to protect a quantum information from errors. In the next section, we will introduce the Toric code, a prototypical example of quantum codes, and examine its coding and physical properties from various perspectives.

Local code bound

Comment: In a real world, classical information is first stored as a ferromagnet which is already quite robust. Then this encoded ferromagnetic bit is further encoded into some good classical error-correcting code. There is not much study on how to physically realize classical codes as ground states of local Hamiltonians. For stabilizer-like frustration-free Hamiltonians, the following bound is known,

$$k d^{1/D} \leq O(n) \quad \text{for local code.}$$

And examples which asymptotically saturate the bound was introduced.

Comment: Another interesting physical interpretation of classical error-correcting code is its relation to spin glass physics. See Nishimori.

Ex 1 Prove the no-cloning theorem. Namely, there is no unitary operator which does

$$|\psi\rangle \otimes |0\rangle \xrightarrow{U} |\psi\rangle \otimes |\psi\rangle$$

for arbitrary psi.

~30min

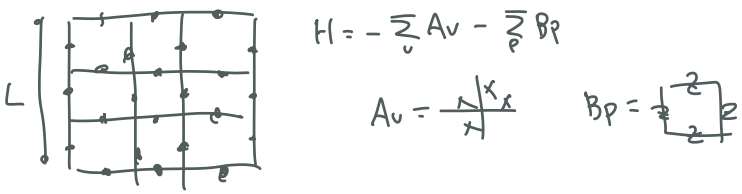
$Physics of Toric code

In this section, we will discuss coding and physical properties of the Toric code from various perspectives.

$$Toric code

$Logical qubits

The Toric code is defined on an L by L lattice on a torus where qubits live on edges. There are n=2L^2 qubits in total. The Hamiltonian consists of vertex terms and plaquette terms:



This Hamiltonian is solvable because all the terms Av and Bp commute with each other:

$$[A_v, B_p] = 0 \quad \text{for all } v, p$$

Av and Bp have +1 or -1 eigenvalues, and ground state satisfies

$$A_v|\psi\rangle = +|\psi\rangle \qquad B_p|\psi\rangle = +|\psi\rangle \qquad \text{for all } v, p$$

There are L^2 constraints from Av, but only L^2-1 of them are independent. Namely we see

$$\prod_v A_v = I \qquad \prod_p B_p = I$$

So, in total, there are 2L^2 - 2 constraints. There are 2^n states in the whole Hilbert space, and each constraint picks half of the states, so we have 4 degenerate ground states:

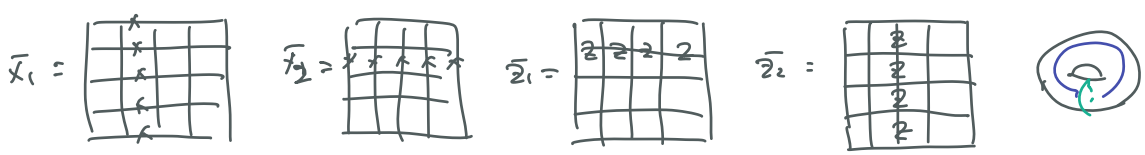
$$\#gs = 2^n / 2^{2L^2-2} = 2^2 = 4$$

This means that the ground state space of the Toric code can encode 2 logical qubits.

Here Av and Bp are stabilizer generators of the Toric code. In general, a stabilizer code can be realized as a ground state space of a Hamiltonian H = - \sum_j Sj.

$Logical operators

Next, we want to know how to manipulate logical qubits. Logical operators are unitary operators which transform encoded states of logical qubits in a non-trivial manner. We have the following logical operators which wind around the torus:



The (Pauli) logical operators commute with the Hamiltonian, so their action preserves the ground state space, but they may act non-trivially on each state. We can check that logical operators obey commutation relations of 2 qubits:

$$[H, \bar{X}_j] = [H, \bar{Z}_j] = 0 \qquad \{\bar{X}_1, \bar{Z}_1\} = 0 \qquad \{\bar{X}_2, \bar{Z}_2\} = 0.$$

So, these logical operators are nothing but Pauli operators for logical qubits. It is often convenient to represent logical operators in the following canonical form:

$$\begin{Bmatrix} \bar{X}_1 & \bar{X}_2 \\ \bar{Z}_1 & \bar{Z}_2 \end{Bmatrix}$$

where operators in the same column anti-commute, and operators in different columns commute [Beni-Chuang].

Transversal logical ops

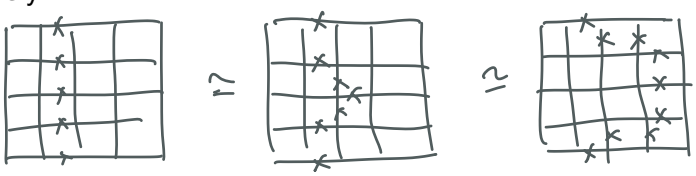
One can label 4 degenerate ground states as eigenstates of Z1 and Z2, as |0,0⟩ |0,1⟩ |1,0⟩ |1,1⟩. With these logical operators, we can perform arbitrary Pauli-like operations on logical qubits by applying string of Pauli operators. These Pauli logical operators are fairly easy to implement because they are written in a tensor product form, and all the Pauli operators in logical operator can be implemented simultaneously.

One question is how to perform non-Pauli operators. It turns out that not all the logical operators can be implemented in a tensor product form, a result known as the Eastin-Knill no-go theorem [Eastin-Knill]. This, and how to get around this, is the topic of the next section.

as well as strengthening of EK theorem.

$Deformability of logical operators

(Deformability) The most important property of a quantum error-correcting code is non-uniqueness, meaning that there are multiple different expressions of logical operators which act in the same manner in the ground state space. In the Toric code, one can deform the shape of logical operators continuously:



Here, applying stabilizer generators Av deforms the shape of logical operators. Since Av=1 in the ground state space, deformed logical operators are equivalent.

(Winding and code distance) Even though one can deform the shape of logical operators locally, one cannot eliminate the topological winding around the torus. That is, non-trivial logical operators must wind around the torus. This guarantees that the minimal weight of logical operators is L.

In stabilizer code, the minimal logical operator size is defined as the code distance. For a stabilizer code, it is formally defined as follows:

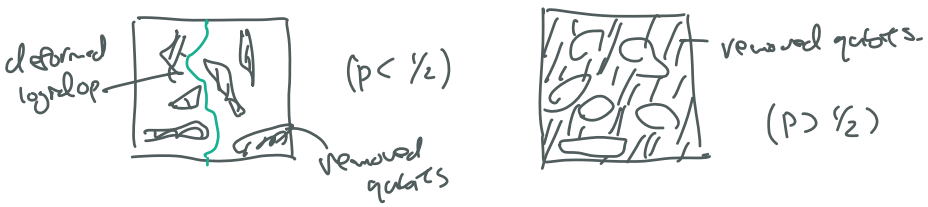
$$d = \min_{O} w(O) \qquad [O, S_j] = 0 \qquad O \notin S$$

Here w(O) is the weight (number of non-identity Pauli operators) in O, Sj are stabilizer generators, and S is the stabilizer group generated by Sj. For the Toric code, d = L.

This is the number of Pauli operators needed to transform one ground state to another orthogonal ground state, a natural generalization of the Hamming distance.

(Erasure threshold)

Here we will demonstrate that this redundancy of logical operator expression provides a protection of logical qubits. We focus on the simplest model of errors where qubits are lost from the system (that is, we eliminate each qubit with probability p>0). If p < 1/2, the regions with removed qubits will form isolated clusters because it is below the percolation threshold.



So, one should be able to find a line which avoids removed qubits, but still winds around the torus. Hence, one can access the logical qubits by measuring these deformed logical operators.

More precisely, the probability of successfully recovering the logical qubit is (for $p < 1/2$)

$$\text{prob(fail)} \leq \exp(-\text{const} \cdot N^{\delta}) \longrightarrow 0$$

not important for us.

Where delta is some positive constant. What is important for us is that failure probability asymptotes to 0 as n goes to infinity. But if p>1/2, the removed qubits will proliferate and one cannot draw a line wrapping around the torus. In this case, logical qubits are lost from the system. Instead, logical qubits are present in the discarded qubits. (For unknown Pauli errors, this problem is more complicated. The problem of finding a threshold can be mapped to spin glass problem.)
(Classical vs quantum) As one can see, the deformability of logical operators (which is a manifestation of non-uniqueness of logical operators) is the key for error-correction. In the case of classical code, we protected information by redundancy 0 to 00000 etc. In quantum case, we will protect quantum information by redundancy of logical operators. In general, when we restrict our attention to a sub space of the whole Hilbert space, two different operators may have equivalent action. This fact played a crucial role in recent development in the physics of AdS/CFT correspondence through the lens of QEC codes.

$String-net picture

We have seen that the ground state space of the Toric code protects logical qubits by looking at logical operators, but we didn't discuss physical properties of ground states in a direct manner. Here we study the ground state.

Let us explicitly construct one ground state of the Toric code.

Claim:

$|g.s\rangle = \prod_v (1+A_v)|0\rangle^{\otimes n}$ is a ground state.

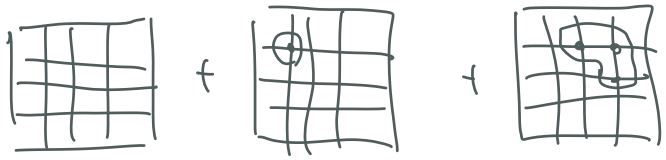
Proof: $B_p|gs\rangle = B_p \prod_v (1+A_v)|0\rangle^{\otimes n} = \prod_v (1+A_v) B_p |0\rangle^{\otimes n} = \prod_v (1+A_v)|0\rangle^{\otimes n}$ $([A_v, B_p]=0)$

$A_v|gs\rangle = A_v \prod_v (1+A_v)|0\rangle^{\otimes n} = \prod_v (1+A_v)|0\rangle^{\otimes n}$ because $A_v(A_v+I) = A_v^2 + A_v = I + A_v$.

Here we used the fact that (1+Av) is a projector onto Av=+1 subspace. This is a useful trick in discussing entanglement properties of stabilizer states. We will use it again in the problem.

$$\left[\frac{I+A_v}{2}\right]^2 = \left[\frac{I+A_v}{2}\right]$$

Let us now examine each term as shown below:

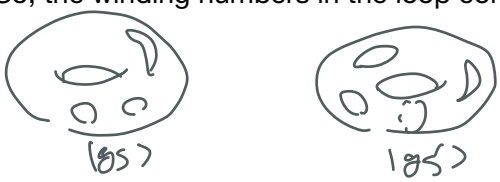


Note that applying Av will create 1's. Here we draw closed loops along qubits in 1 states. When multiple loops overlap with each other, they will form a bigger loop. Hence, this ground state can be written as

$$|gs\rangle = \sum_{\text{closed loops}} |\sigma\rangle$$

trivial

The ground state can be written as a superposition of all the possible loop configuration. Note that closed loops in this ground state can be continuously deformed to a smaller top. Other 3 ground states can be constructed by applying logical X1 and X2 which inserts closed loops which wind around the torus. So, the winding numbers in the loop configurations can distinguish 4 ground states.



The Toric code is often called a Z2 string-net condensation. Here, our strings are Z2 like object. But they can carry more degrees of freedom in general. This way, one can significantly generalize the Toric code. If your construction is based on a finite group G, then you will get quantum double model. You can go further and define (commonly believed to be) the most generic 2D model (non-Chiral), string-net condensation.

$Stability of ground states

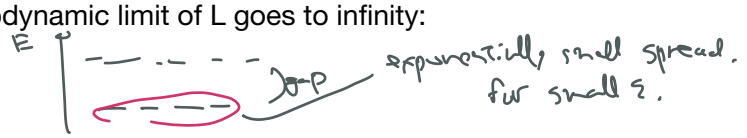
We have discussed that logical qubits in the Toric code are robust against errors. Here we discuss physical consequences of the fault-tolerance which emerges as the stability of the ground state space.

Suppose we somehow engineered the Hamiltonian of the Toric code in a laboratory. In reality, our implementation of the Hamiltonian is not perfect, so the resulting Hamiltonian may be written as follows

Where V consists of local terms only. \epsilon controls the error of our implementation of the Toric code Hamiltonian. The question here is whether the ground state degeneracy (2 logical qubits) is still present or not. This is often called the stability of the Hamiltonian (or the ground state space, or the quantum phase).

Here we first address this question by using a simple perturbative analysis. In order to lift the ground state degeneracy, the perturbation should be able to distinguish ground states and assign different values of energy penalties. Recall that we will need string-like logical operators to distinguish ground states.

Since V is a local perturbation, in the first order perturbative approximation, V cannot lift the degeneracy. Only at the O(L)-th order perturbative approximation, where V^L contributions may appear, the ground state degeneracy can be lifted. But the L-th order perturbative effect is exponentially suppressed with respect to L. As such, we expect that, for sufficiently small \epsilon, the ground state energy splitting is exponentially suppressed, and the degeneracy is protected at the thermodynamic limit of L goes to infinity:



Here in this heuristic argument, it is very important that logical operators are high weight. So, the fact that the Toric code is a quantum code renders the stability of the Hamiltonian.

This intuition was rigorously proven by Bravyi-Hastings-Michalakis where they proved that the Toric code's ground state's energy splitting is exponentially suppressed and the energy gap between the excited state and ground states remains finite for sufficiently small \epsilon.

Their proof is technical tour-de-force, so we will not review it here. Instead, we will look at the key ingredient. In their proof, Bravyi-Hastings-Michalakis was able to identify a certain necessary condition

for the stability of ground states. This is called the local indistinguishability.

Claim: Consider an arbitrary pair of ground states of the Toric code, and look at their reduced density matrices defined for some local region R. For any connected region R which can be continuously deformed to small region, two density matrices are indistinguishable:

$|\psi_{GS}\rangle, |\phi_{GS}\rangle \longrightarrow \rho, \sigma \qquad \rho_R = \sigma_R$

This may be intuitively clear, but proving it requires 2 tools which are very useful!!

This is a formal statement saying that any local operator cannot distinguish ground states.

For simplicity, we will prove this for Pauli logical states. (eigenstates of $\bar{X}_j$ or $\bar{Z}_j$'s)

- Proof: Recall that Pauli operators form basis of quantum states in the following sense.

Tool 1

We can always decompose ρ on n qubit system as

$$\rho = \sum_{P \in Pauli} c_P P \qquad \text{where} \quad P = P_1 \otimes \cdots \otimes P_n$$

with some coefficient c_P. ($P_1, \ldots P_n$ are I, X, Y, Z). One can solve for c_P by using orthogonality of Pauli ops: (This is always possible)

$$\mathrm{tr}(P\theta) = \frac{\delta_{P,\theta}}{2^n} \qquad (P, \theta \in Pauli)$$

Hence we have

$$c_P = \frac{\mathrm{tr}(P\rho)}{2^n}$$

Note we always have

$$c_I = \frac{1}{2^n} \quad \text{because} \quad \mathrm{tr}(\rho) = 1.$$

Tool 2

- We now represent the gs in this way. One useful technique here.

$S_j|\psi\rangle = +|\psi\rangle$ for $j = 1, \ldots n$. (S_j : indep generators). Then

$$|\psi\rangle\langle\psi| = \frac{1}{2^n}\prod_j (1+S_j) = \frac{1}{2^n}\sum_{S \in \mathcal{S}} S.$$

(Given $\mathcal{S}$, choice of S_j is not unique. But we can choose any indep S_j)

$\frac{1}{2}(1+S_i)$ is a projector.

- Let's look at $|0,0\rangle$ state with $\bar{Z}_1, \bar{Z}_2 = +1$.

This state is stabilized by $S_1, S_2, \ldots, S_{n-2}, \bar{Z}_1, \bar{Z}_2$,

Defing $\mathcal{S}_{\bar{Z}_1,\bar{Z}_2} \equiv \langle \mathcal{S}, \bar{Z}_1, \bar{Z}_2 \rangle$, we have

$$|0,0\rangle\langle 0,0| = \frac{1}{2^n}(1+S_1)\cdots(1+S_{n-2})(1+\bar{Z}_1)(1+\bar{Z}_2) = \frac{1}{2^n}\sum_{S \in \mathcal{S}_{\bar{Z}_1,\bar{Z}_2}} S.$$

(Here we are free to pick any $\bar{Z}_1, \bar{Z}_2$ deformed etc)

proof

$$\mathrm{Tr}_{\bar{R}}|0,0\rangle\langle 0,0| = \frac{1}{2^n}\sum_{S \in \mathcal{S}_{\bar{Z}_1,\bar{Z}_2}} \mathrm{Tr}_{\bar{R}}(S)$$

Recall $\mathrm{tr}(P) = 0$ for $P \in Pauli$ $P \neq I$.

So, $\mathrm{Tr}_{\bar{R}}(S) \neq 0$ only when S has no support on $\bar{R}$. (only on R)

Let's define the restriction of $\mathcal{S}_{\bar{Z}_1,\bar{Z}_2}$ on R by $\mathcal{S}_{\bar{Z}_1,\bar{Z}_2}|_R$.

This consists of $S \in \mathcal{S}_{\bar{Z}_1,\bar{Z}_2}$ supported strictly on R.

Note this forms a group. product of two ops on R is still on R.

The key observation is

$S_{\bar{z}_1 \bar{z}_2} |_R = S|_R$. This is because.

$S_{\bar{z}_1 \bar{z}_2}$ consists of S, $\bar{z}_1$ times S. $\bar{z}_2$ times S. $\overline{z_1 z_2}$ times S.

↑ stabilizers ↑ $\bar{z}_1$ equivalent ↑ $\bar{z}_2$ equivalent ↑ $\overline{z_1 z_2}$ equivalent

Logical ops wrap around the torus, so none of them can be supported on R

One can repeat a similar argument on show that

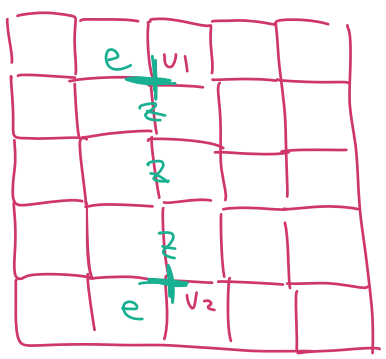
$\mathrm{Tr}_R |\psi\rangle\langle\psi| = \frac{1}{\dim(S|_R)} \sum_{s \in S|_R} s$. for stabilizer ground states.

For other states, we use tool 1 to show the similar statement. □

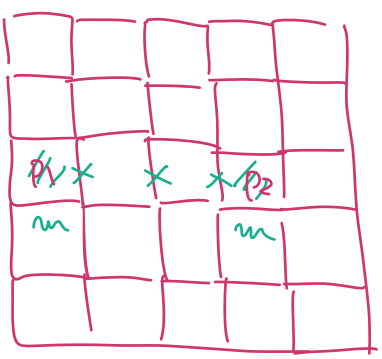
$Anyons

So far we discussed properties of ground states of the Toric code. Next, let us discuss properties of excitations in the Toric code. In 3 spatial dimensions, particles can be either bosons or fermions. Exchanging bosons/fermions generate additional +1 or -1 phase factors. In 2 spatial dimensions, particles can exhibit richer properties under exchanges and braidings, which go beyond fermions/ bosons. These are called anyons. Toric code is the simplest example which supports anyonic excitations.

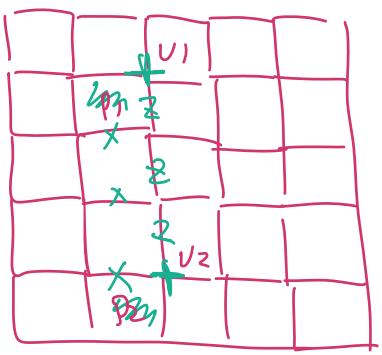
Anyonic excitations in the Toric code can be created by applying a segment of logical operator, as shown below.



$A_{v_1} = -1$
$A_{v_2} = -1$



$B_{p_1} = -1$
$B_{p_2} = -1$



$A_{v_1} = -1$ $B_{p_1} = -1$
$A_{v_2} = -1$ $B_{p_2} = -1$

Since logical operators commute with all the terms in the Hamiltonian, a segment of logical operators creates excitations only at its endpoints.

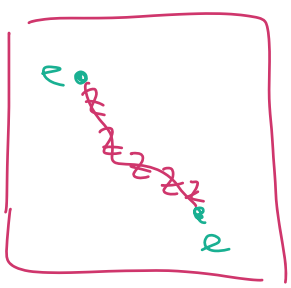
These excitations can be interpreted as quasi-particles (which are not particles in the original construction of the Toric code, but can "emerge" as particles when the system gets excited.) In the Toric code, there are 3 types of anyonic excitations:

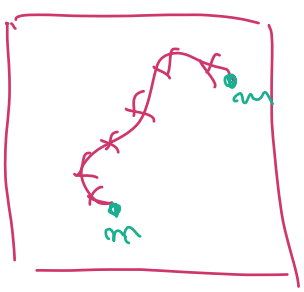
- "$A_v = -1$" electric charge e
- "$B_p = -1$" magnetic flux m
- "$A_v = -1$" & "$B_p = -1$" fermion f (= e×m)

The names "electric charge" and "magnetic flux" come from an analogy with lattice gauge theory. It is arbitrary to call which of Av or Bp, charge or flux. As we will see later, e and m are bosons, in a sense that exchanging two e's (two m's) do not generate non-trivial phase. But a composite of e and m (f = e*m) terms out to be a fermion, with additional -1 phase.

(Properties) Let us study key properties of anyons in the Toric code.

- They appear as a pair. This is because a segment of logical operators have even number of end points. So, the total number of electric charge is always even:





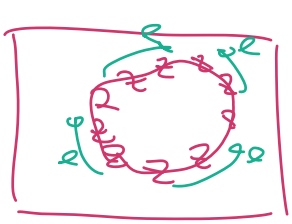
total # of e
$= 0 \pmod 2$
total # of m.
$= 0 \pmod 2$

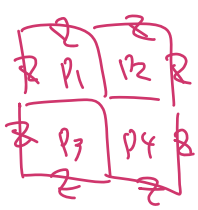
This can be shown by observing the following:

$\prod_v A_v = I$ $\prod_p B_p = I$.

Here, we are identifying Av=-1 as an excitation, so the total number must be even.

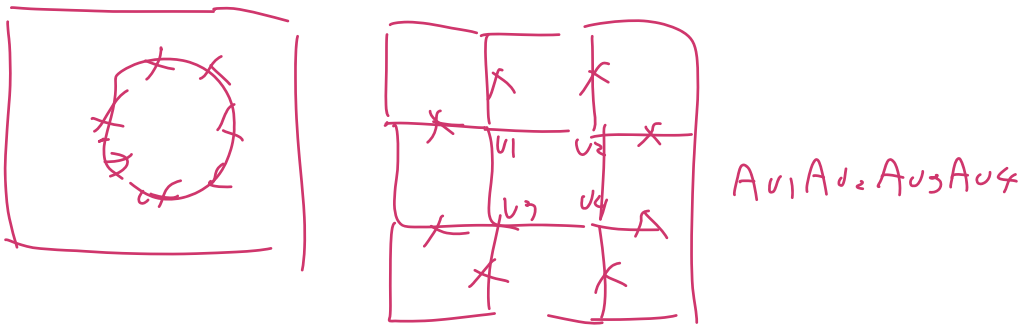
- Stabilizer operators corresponds to creating a pair of anyons and then annihilating them. Here let us consider electric charges, which are excited by Pauli Z operators. Notice that multiplying Bp terms together can create a closed string of Z operators. Let us think of applying Pauli Z operators one by one so that it eventually forms a closed string. At the beginning, a pair of electric charges are created, and then propagate along the loop. Eventually, they meet and get annihilated.





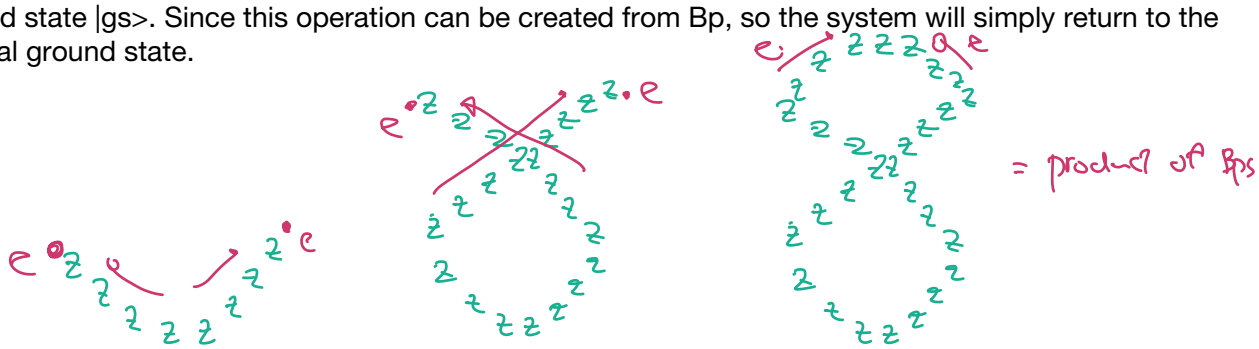
$= B_{p_1} B_{p_2} B_{p_3} B_{p_4}$

A similar observation holds for magnetic fluxes. Combining Av operators will create a closed loop of X operators on a dual lattice. This represented a pair creation of m, propagation and their annihilation.

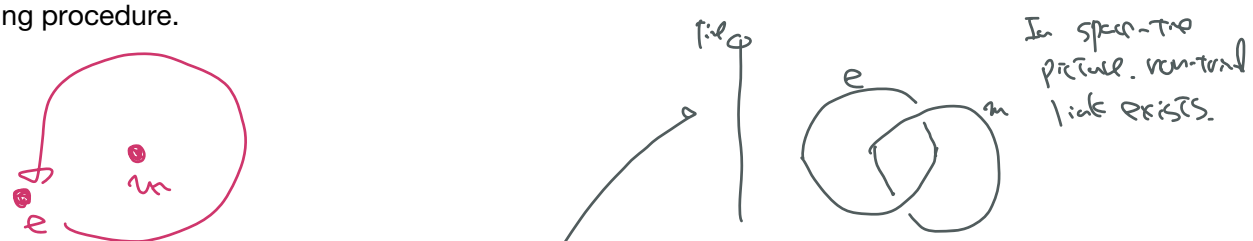


- Finally let us verify that these are indeed anyons.

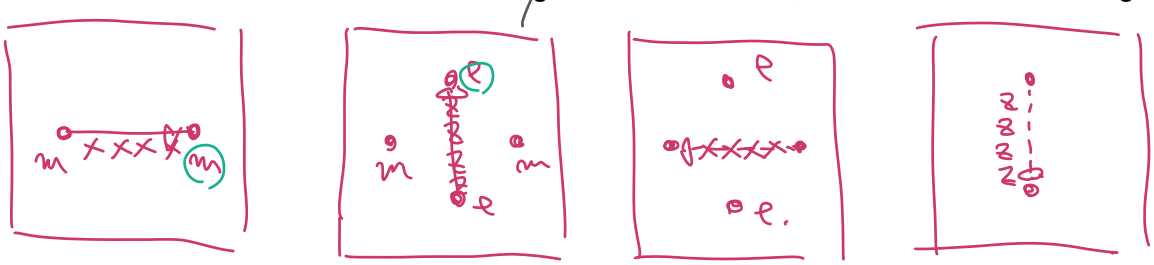
e and m are bosons. This can be checked by creating a pair of e, exchange them, and annihilate them. The overall process can be generated by the following trajectory of Pauli Z operators, acting on a ground state |gs>. Since this operation can be created from Bp, so the system will simply return to the original ground state.



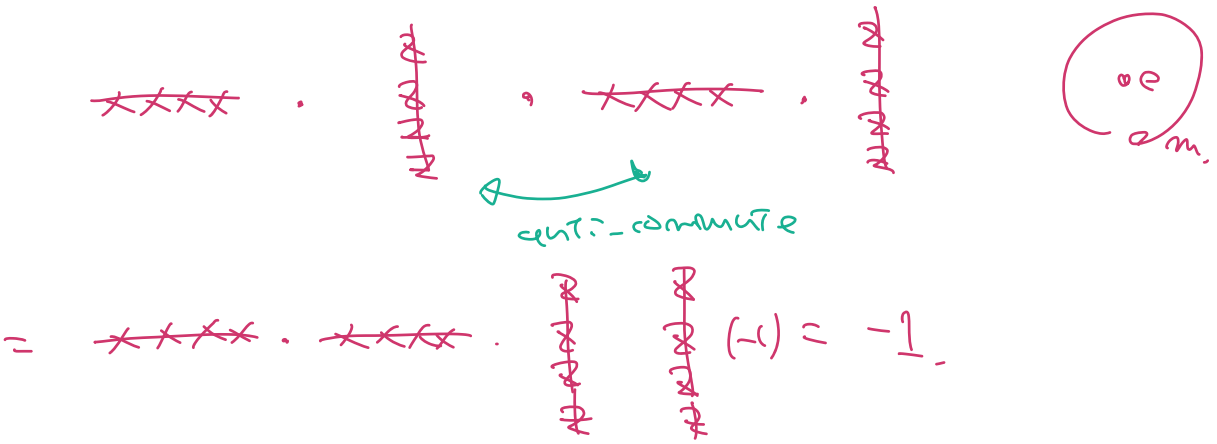
E and m have non-trivial braiding statistics. This can be studied by looking at an overall effect of the following procedure.



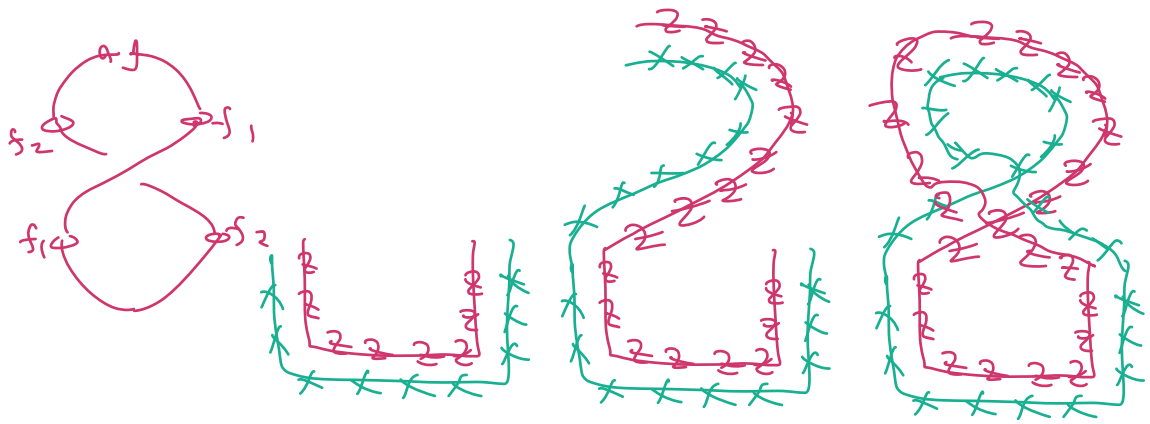
We will show that there will be an additional -1 sign. To braid e and m, we consider the following steps:



Here, a pair of two e's and a pair of two m's are created, and then annihilated eventually. The net effect is to make e circle around m by 360 degree. The overall effect of this procedure is given by



Finally, we will show that f is indeed fermion. This is a bit complicated to see, but we again need to look at the following procedure.



We see that the net effect consists of a loop of X and a loop of Z, but on one location, X needs to be applied before Z, and on the other location, Z needs to be applied before X. This mismatch generates an additional -1 sign.

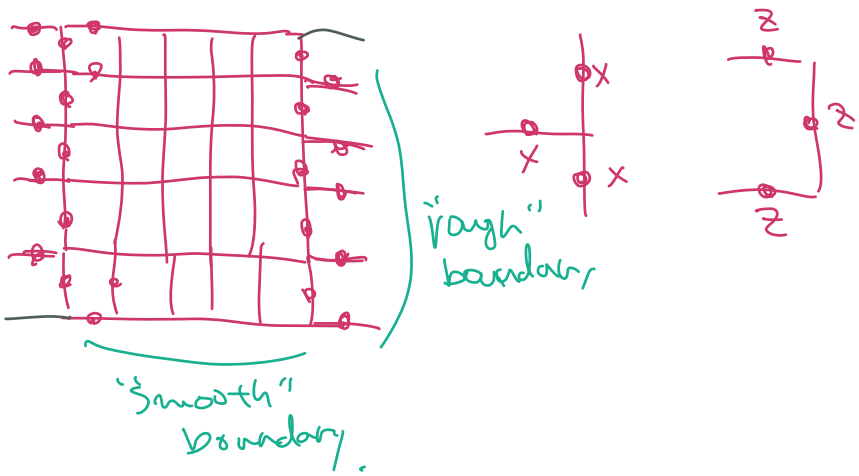
Combining these observations, one can construct the following braiding statistics table:

	1	e	m	f
1	1	1	1	1
e	1	1	-1	-1
m	1	-1	1	-1
f	1	-1	-1	-1

$Toric code with boundaries

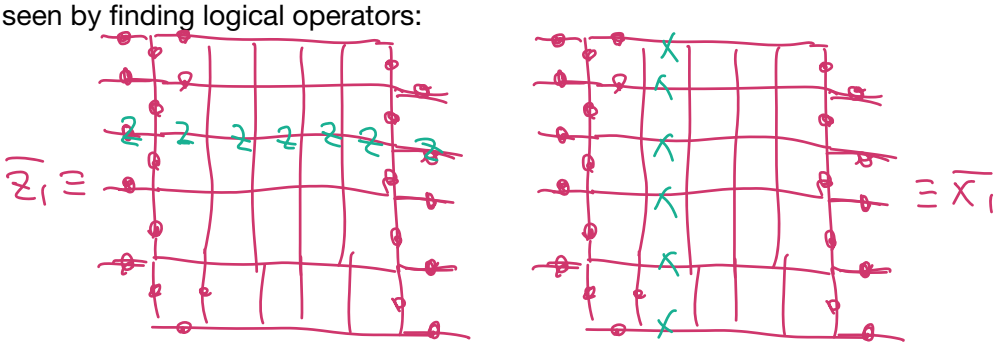
In a real world, it is a bit challenging to put the Toric code on a torus. Can we build a quantum code on a more conventional lattice?

In the Toric code, there are two types of consistent boundary conditions, called rough and smooth boundaries, originally found by Bravyi and Kitaev. These are depicted below:



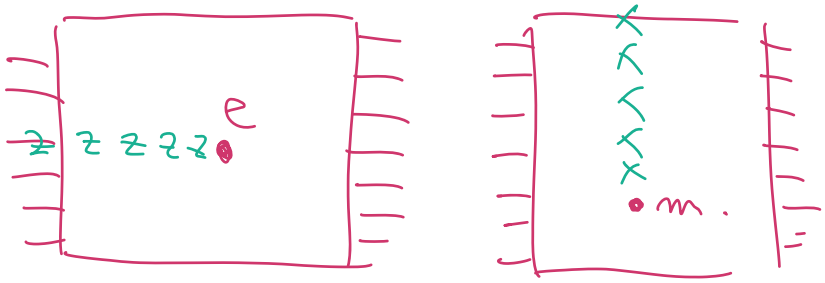
You can check that all the terms, including boundary ones, commute with each other. So, this model is still a stabilizer code, and has a gapped energy spectrum.

It turns out that this code encodes only 1 logical qubit, instead of 2 in the original one on a torus. This can be seen by finding logical operators:



Z-type logical operators can be anchored at rough boundaries, but not at smooth boundaries. Similarly, X-type logical operators can be anchored at smooth boundaries, but not at rough boundaries. Hence, we only find a single pair of X and Z logical operators, corresponding to a single logical qubit.

One interesting physical consequence is that, depending on the type of boundaries, anyonic excitations can be absorbed. Namely, by attaching a segment of Z-type logical operator, you can create a single e excitation from the rough boundary.



In a fanciful term, this phenomena is called condensation of anyons. It simply says that a certain type of anyons can be created from the boundary (without creating pair excitations). Here you see that

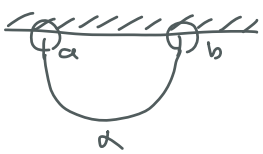
Smooth boundary: m condenses
Rough boundary: e condenses.

In studies of topological phases of matter, it is an important problem to classify what kinds of gapped boundaries can be attached. An important result is a “theorem” due to Levin and Gu. It is not a rigorous mathematical proof, but very beautiful physical “proof”.

Claim: If a set of anyons can condense into a gapped boundary, this set must be mutually bosonic. Here, by mutually bosonic, it means that their self-statistics is bosonic, and their mutual braiding-statistics is trivial.

Proof:

- Let i, j are particles in $\mathcal{M}$ ($\mathcal{M}$: set of condensing particles).



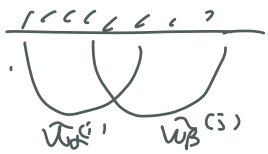
consider an open path α.

$\hat{W}_\alpha^{(i)} = U_a^{(i)} W_\alpha^{(i)} U_b^{(i)}$ U_a, U_b are local ops since boundary is gapped.

$\hat{W}_\alpha^{(i)} |gs\rangle = |gs\rangle$ — ①

Notice that $\hat{W}_\alpha^{(i)} \hat{W}_\beta^{(j)} |gs\rangle = e^{i\Theta_{ij}} \hat{W}_\beta^{(j)} \hat{W}_\alpha^{(i)} |gs\rangle$

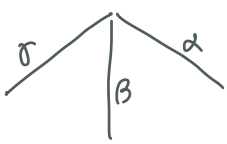
Using ①, we have $e^{i\Theta_{ij}} = 1$ $(i \neq j)$

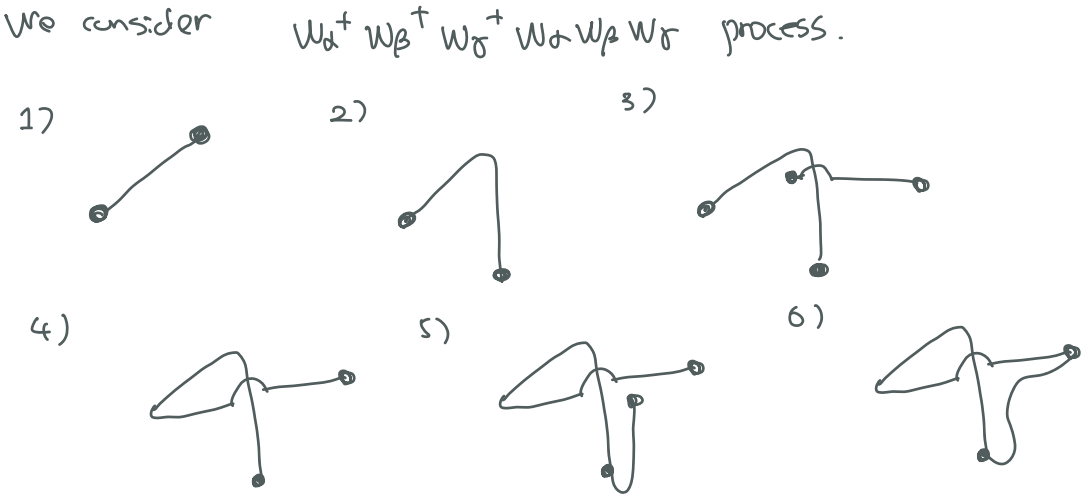


- As for self-statistics, we need a certain trick, called loop-op algebra. due to Levin & Wen.

$W_\alpha W_\beta W_\gamma |gs\rangle = e^{i\Theta} W_\sigma W_\beta W_\alpha$

where α, β, γ are open strs with one common endpoint.





Then we can see that this exchanges the particle of the same type (as we did in fermion in the Toric code). Since all the W operators commute with each other, we find that the self-statistics is trivial…

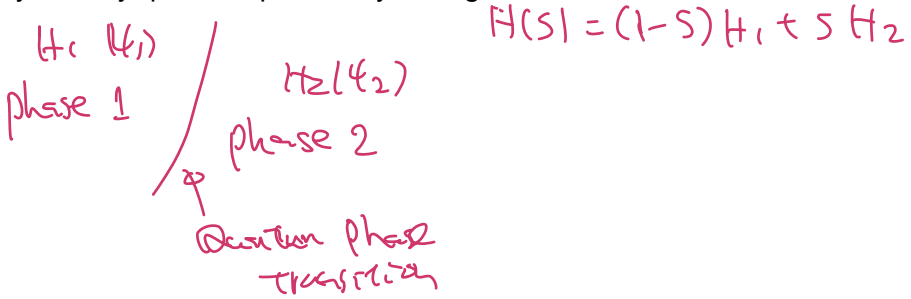
In the case of the Toric code, you have only e or m as bosons, so there are 2 possible types of boundaries. Namely, you cannot put a boundary which absorbs f.

Classification of gapped boundaries is an important subject in studies of topological phases of matter. It turns out that this problem has a lot of resemblance to another important problem in quantum error-correction, namely the classification of transversal logical gates. We will briefly comment on this in later lectures.

$Circuit complexity of the Toric code ground state

The Toric code is a prototypical example of non-trivial quantum phase with topological order. But what does it mean by “quantum phase”? What is a good definition of quantum phase? Fortunately, quantum information theory provides a precise answer to this question.

Conventionally, quantum phases were used to characterize properties of ground states of a gapped Hamiltonian (whose energy gap remains finite at the thermodynamic limit) at zero temperature. We typically classify quantum phases by asking whether two Hamiltonians can be smoothly interpolated or not:



If two Hamiltonians belong to different quantum phases, we expect that one cannot go from one to the other, and we must encounter quantum phase transition in between which involve some non-analytic change of ground state properties (at the thermodynamic limit). If two belong to the same quantum phase, we should be able to smoothly deform one to the other.

There are two essential difficulties here. First, in order to classify quantum phases, we need to check every possible path connecting two Hamiltonians. So, it is not easy to show that two Hamiltonians are actually in different phases. Second, a quantum phase refers to a family of Hamiltonians with similar ground state properties (in some appropriate sense). But, in order for the notion of “phase” to make sense, the Hamiltonian must be stable.

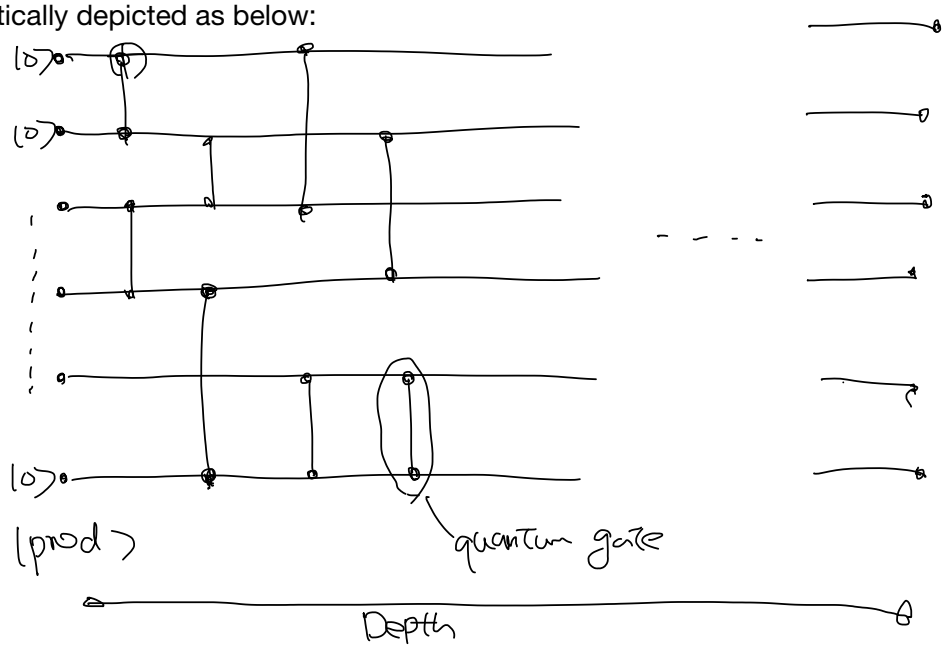
To gain some intuition, here let’s think about an ordinary phase of matter in classical world, a gas and a solid phase of water. They are different phases because we cannot smoothly connect them by changing pressure or temperature. At the same time, they can be identified as “phases” because their properties are more or less stable against small fluctuations of temperature and pressure. Otherwise, we won’t be able to perceive them as phases of water.

As for the stability issue, we have already seen that the Toric code is stable against small perturbations. So, the Toric code can serve as a representative of a family of Hamiltonians which form a quantum phase! Now, the question is whether the Toric code belongs to a non-trivial quantum phase or not.

In order to attack this question, the notion of quantum circuit complexity will be useful.

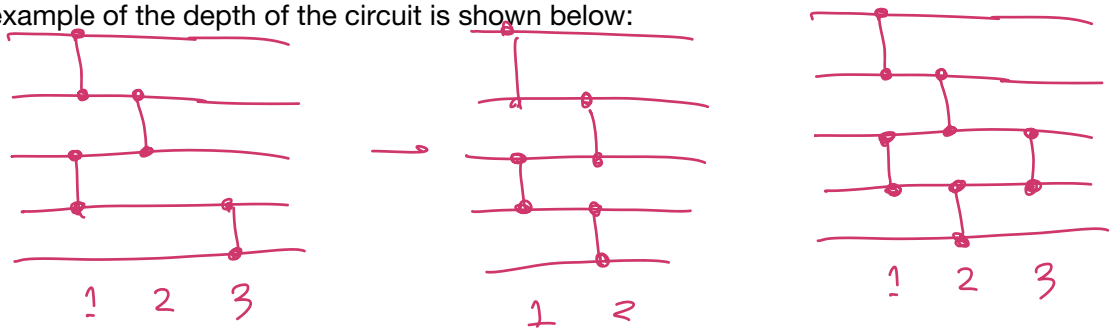
(Quantum complexity definition)

Starting from a reference state |psi>, a circuit complexity of |phi> is defined roughly as the number of time steps one needs to convert psi to phi. To be concrete, let us ask the circuit complexity of |toric> with reference being |0>. The Toric code ground state can be prepared by some quantum circuit, schematically depicted as below:



Notice that some quantum gates, namely commuting ones, can be implemented simultaneously. The minimal time step for implementing a circuit U is called the depth. Here, we assumed that we are allowed to implement 2-qubit gates on neighboring qubits only on the toric code lattice.

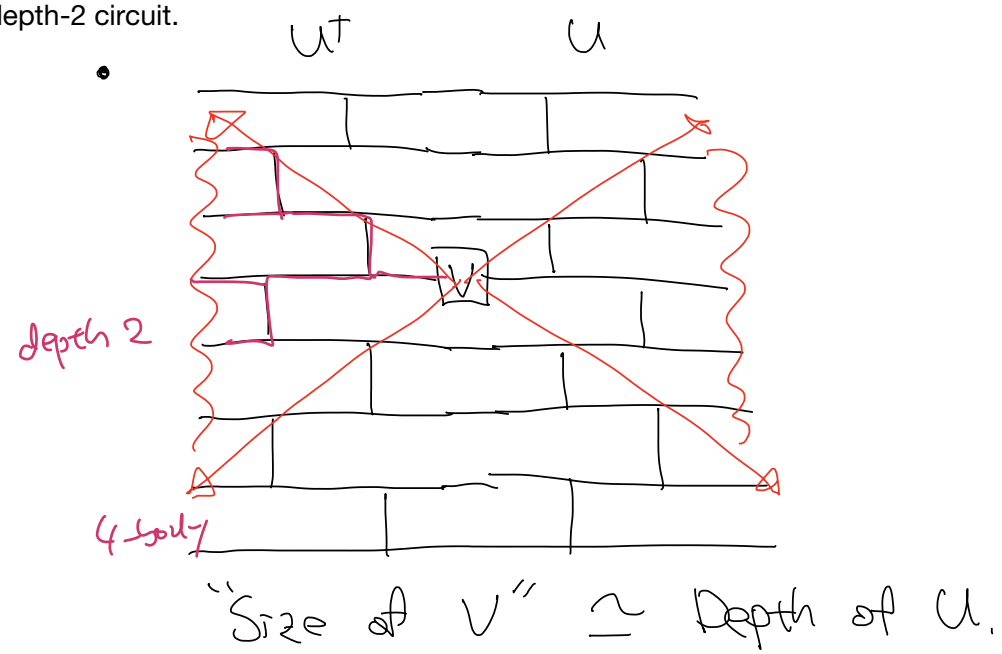
An example of the depth of the circuit is shown below:



(Size of operators)

Our goal is to prove that the Toric code's circuit complexity is O(L). To prove this statement, it is useful to use the notion of the operator size.

Given one-body unitary operator V, we ask how large UVU is. Below, we show the case which evolves by depth-2 circuit.



We see that the size of the operator can grow at most by finite size. Namely, in 2D, its size can grow at most as depicted below:

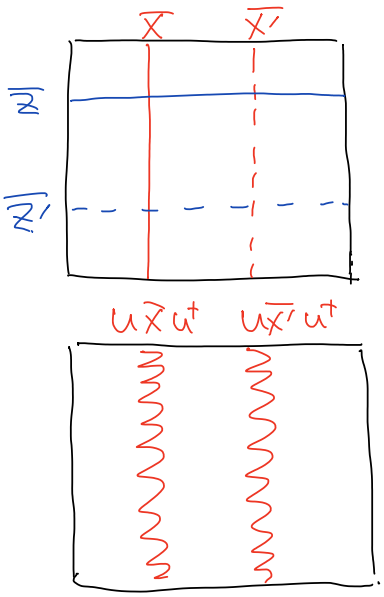
Hence, if U is O(1) depth circuit, UVU will be O(1)-size operator.

Finally, we prove that the toric code ground state has O(L) circuit depth.

proof

Suppose there exists short-depth U s.t

$|toric\rangle = U|prod\rangle$ — (*)



$\langle toric|\bar{X}\bar{X}'|toric\rangle = 1$ — ①

$\langle toric|\bar{Z}\bar{Z}'|toric\rangle = 1$

$\langle toric|\bar{X}\bar{Z}\bar{X}'\bar{Z}'|toric\rangle = -1$

anti-commutation

$\langle prod|U^\dagger\bar{X}U\,U^\dagger\bar{X}'U|prod\rangle = 1$

No overlap (*) used.

$\langle prod|U^\dagger\bar{X}U|prod\rangle\langle prod|U^\dagger\bar{X}'U|prod\rangle = 1$

$= \pm 1$ $= \pm 1$

$(U^\dagger\bar{X}U)^2 = I \Rightarrow$ eigvalues ± 1

$\langle prod|U^\dagger\bar{X}U\,U^\dagger\bar{Z}U\,U^\dagger\bar{X}'U\,U^\dagger\bar{Z}'U|prod\rangle = 1$

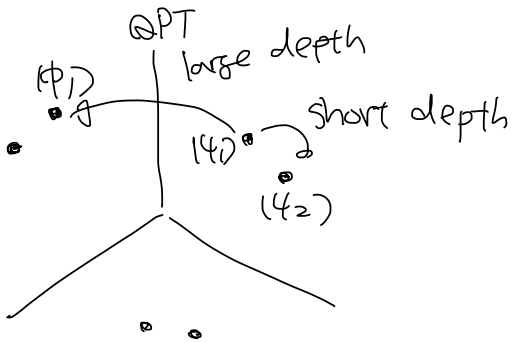
contradiction U must be at least $O(L)$ depth. □

On the other hand, if two Hamiltonian can be smoothly interpolated, then we can just adiabatically transform one ground state to the other. In general, we need the following time scale:

$$t \sim O\left(\frac{1}{\Delta}\right)$$

Where \Delta is the smallest energy gap. If the system remains gapped throughout the interpolation, then one can switch between two states by O(1) time.

(Summary) In summary, the notion of quantum circuit complexity gives a precise definition of quantum phases of matter. Namely, ground states in the same phase can be interpolated via short-depth circuit, while ground states in different phases cannot be.



(There is a caveat in this classification. Consider ground states of the Toric code. There are 4 orthogonal states, and we can consider arbitrary superposition of them. It turns out that not all the pairs of ground states can be connected by local unitary transformations. This is the result due to Bravyi and Koenig (based on Eastin-Knill theorem)).

So, even in the same Hamiltonian, two states may not be smoothly connected. For this reason, it is usually better to think if one can smoothly map the ground state space to the other ground state space, forgetting about transformations inside the ground state space. We will return to this beautiful result by Bravyi and Koenig later.

Putting this subtlety under the rug, the circuit complexity is a versatile and precise way to classify quantum phases.

$Symmetry-protected topological phases

Finally, let us discuss one possible generalization of the aforementioned scheme of classifying quantum phases. Realistic physical systems often have symmetries such as a conservation of the total particle number U(1) symmetry, or Z2 symmetry. In the presence of symmetries, we usually consider the interpolation problem by imposing such symmetries throughout.

The classification of quantum phases become richer when symmetries are imposed. Namely, we can have the so-called symmetry protected topological phases. Here in classifying quantum phases, we demand that each local gate in the circuit obeys the symmetry constraint. That is, we ask if two ground states are connected by symmetric local unitary circuit or not. It turns out that there are examples of short-range entangled state which can be transformed into a product by a constant depth quantum circuit, but cannot be done so by a constant depth symmetric quantum circuit.

consider a 1D chain of 2n qubits.

Here we will look at a prototypical example of an SPT phase, the 1D cluster state. The 1D cluster state is a ground state of the following Hamiltonian:

$$H = -\sum_j S_j \qquad S_j = Z_{j-1} X_j Z_{j+1}. \qquad [S_i, S_j] = 0$$

This ground state can be created by a constant depth circuit. Namely, let us begin with |+> states which are stabilized by X operators. We then apply the CZ gates:

$$CZ|a,b\rangle = (-1)^{ab}|a,b\rangle \qquad a,b = 0,1.$$

When CZ gates act on two qubits, it transforms Pauli operators in the following manner:

$$(CZ)^\dagger = CZ \qquad (CZ)(XI)(CZ) = XZ \qquad (CZ)(ZI)(CZ) = ZI$$
$$(CZ)(IX)(CZ) = ZX \qquad (CZ)(IZ)(CZ) = IZ.$$

As such, if we apply CZ on neighboring qubits in 1D chain, we obtain the stabilizer generators of the cluster state:

$|\text{prod}\rangle = |+\rangle^{\otimes 2n} \longrightarrow |\text{cluster}\rangle$. X_j was stabilizer for $|\text{prod}\rangle$.

$\overline{CZ} = \prod_{j=1}^{2n} CZ_{j,j+1}$ (Note $CZ_{j,j+1}$ commute with each other because they are diagonal in comp basis state). $\overline{CZ}\, X_j\, \overline{CZ} = S_j$.

Hence, the cluster state and a trivial product state belong to the same quantum phase according to the definition based on quantum circuits.

Let us now impose symmetries to the system. Observe that the cluster state Hamiltonian has two sets of Z2 symmetries. Here we label qubits by $A_1, \cdots, A_n, B_1, \cdots, B_n$ in an alternating manner.

$S_A = \bigotimes_j X_{Aj}$ $S_B = \bigotimes_j X_{Bj}$.

One can see that the Hamiltonian commutes with these symmetry generators:

$[S_A, S_j] = [S_B, S_j] = 0$

The ground state (cluster state) is +1 eigenstate of these symmetry generators because:

$S_A = \prod_{j:\text{odd}} S_j$ $S_B = \prod_{j:\text{even}} S_j$.

We now ask if the cluster state and the product state belong to the same phase or not in the presence of this symmetry. It turns out that they are different. To gain some intuition, let us first observe that CZ is not a symmetric gate. This is because CZ does not commute with symmetry operators:

Claim e.g. $CZ\, S_A\, CZ = X\,I\,X\,I\,X\,Z\,X\,Z\,X\,I\,X\,I \neq S_A$.

(but note $\overline{CZ}\, S_A\, \overline{CZ} = S_A$. so, $\overline{CZ}$ is symmetric unitary, but cannot be implemented via local sym gates)

Here we prove that the cluster state is a non-trivial Z2 * Z2 SPT phase.

Proof:

- The proof strategy is somewhat similar to toric code case.

  Suppose there exists symmetric local unitary U s.t

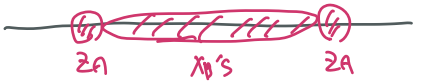
  $|\text{cluster}\rangle = U|\text{prod}\rangle$ where $|\text{cluster}\rangle = |+\rangle^{\otimes 2n}$

  We will consider the following operators:

$\equiv V_A$ (acting on $\frac{n}{2}$ qubits)

$\equiv V_B$

$= V_A$



$= V_B$

  Here V_B is a product of $Z_A \otimes X_B \otimes Z_A$, so $\langle\text{cluster}|V_B|\text{cluster}\rangle = +1$.

- We will use

  $\langle\text{cluster}|\, V_A\, V_B\, V_A\, |\text{cluster}\rangle = -1$ to derive a contradiction.

  ↯

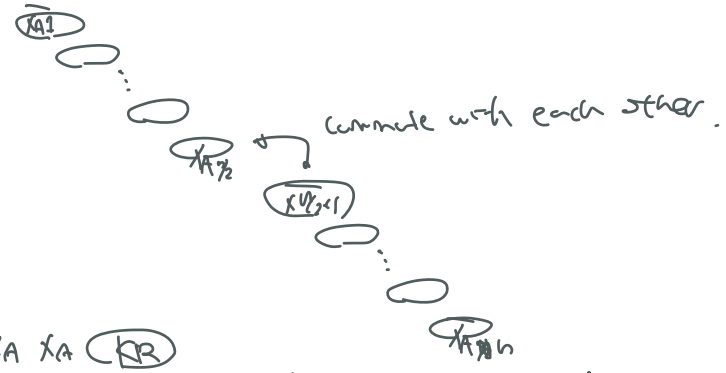
- We study the property of $U^\dagger V_A U \equiv \bar{V}_A$.

$$V_A = \prod_{j=1}^{N_2} X_{Aj}$$

$$\bar{V}_A = \prod_{j=1}^{N_2} \underbrace{U^\dagger X_{Aj} U}_{\bar{X}_{Aj}} = \prod_{j=1}^{N_2} \bar{X}_{Aj} \qquad \bar{X}_{Aj} = U^\dagger X_{Aj} U \leftarrow O(1) \text{ ops.}$$

Note $\bar{X}_{Aj}$ commute with each other.

- Note $U^\dagger S_A U = S_A$ because U is symmetric.

then. $\prod_{j=1}^{n} \bar{X}_{Aj} = S_A =$ 

Hence we have

$$\bar{V}_A = K_L \, X_A \, X_A \, X_A \, X_A \, K_R$$

where the middle part is X_A's. and $K_L \cdot K_R$ $O(1)$-body

Similarly by $\bar{V}_A^2 = 1$ & $K_L^2 = 1$, $K_B^2 = 1$.

$$\bar{V}_B = M_L \, X_B \, X_B \cdots X_B \, M_R \quad \text{with } M_L \text{ \& } M_R \; O(1)\text{-body.}$$

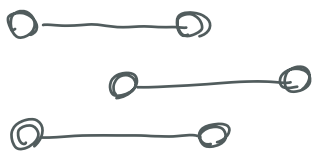
- Note $\langle SPT | \bar{V}_B | SPT \rangle = \langle prod | U^\dagger \bar{V}_B U | prod \rangle$

$$= \langle + |^{\otimes 2n} \; I \; M_L \; X_B\text{'s} \; M_R \; I \; |+\rangle^{\otimes 2n}$$

$$= \langle M_L \rangle \langle M_R \rangle = +1$$

- $\langle SPT | V_A V_B V_A | SPT \rangle = \langle prod | \bar{V}_A \bar{V}_B \bar{V}_A | prod \rangle$

$$= \langle K_L^2 \rangle \langle M_L \rangle \langle M_R \rangle \langle K_R^2 \rangle = +1 \quad \cdots$$



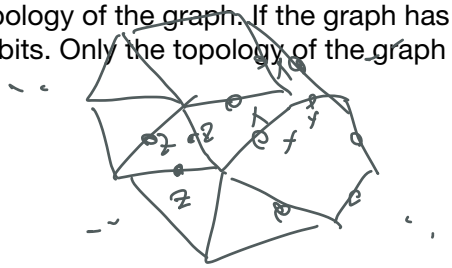
This contradicts with (✡) !!

Hence $|cluster\rangle$ is non-trivial !!

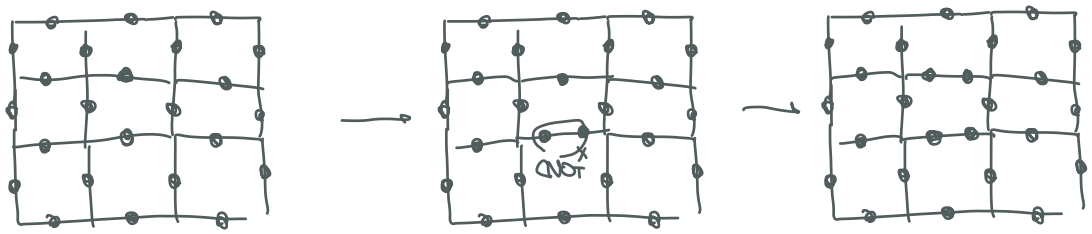
$Toric code preparation circuit

Let's return to the Toric code. We have shown that it takes at least O(L) time to create the Toric code ground state. (Dennis et al gave L^2 algorithm.) Here we show that it is indeed possible to do so in O(L) time. There are various ways, and we will introduce an approach which can be applied to multiple cases.

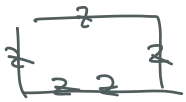
The method utilizes the fact that the toric code is topological, and can be defined on any graph. Given a graph, we place qubits on edges. Then plaquette terms and vertex terms can be naturally defined. We can easily see that all the terms commute with each other. The number of logical qubits depends on the topology of the graph. If the graph has the topology of the torus, then the code will support 2 logical qubits. Only the topology of the graph determines the number of logical qubits.



Given the toric code on an arbitrary graph, one can change the shape of the graph locally by applying local unitary transformations and adding ancilla qubits (in a way not changing the topology of the graph). The strategy is to start from a small toric code, and then make it bigger. First, we introduce a method of adding an extra vertex.



Here ancilla was in |+⟩ state. we applied CNOT



Stabilizers are

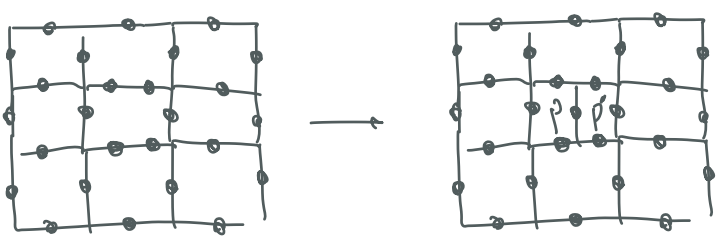
CNOT
ZI → ZZ
IZ → ZI
ZZ → IZ
XI → IX
IX → XX
XX → XI

(Recall we are free to multiply stabilizers)

This is like a "vertex" with 2 edges only.

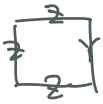
(We will introduce a "decoration" trick, which is very useful in dealing with stabilizer codes in general.)

Reapply the same procedure, we obtain the 3rd figure. We now insert a new edge.

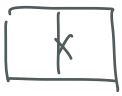
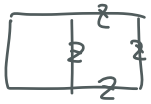


Here the ancilla is in (+) state (X stabilizer)

To create a toric code on this new graph with extra edge, we apply the following unitary.

$U = \exp\left(\frac{\pi}{4} B_p \cdot \boxed{X}\right) = \exp\left(i\frac{\pi}{4}$  $\right)$

This unitary operator is carefully chosen so that

U  $U^\dagger =$  $= B_p$

$U = \exp(i\theta P)$ | decorations

$= \cos\theta\, I + i\sin\theta\, P$

$[P, \theta] = 0 \Rightarrow U\theta U^\dagger = \theta$

$\{P, \theta\} = 0$

$\Rightarrow U\theta U^\dagger = \cos(2\theta)\theta + \sin(2\theta)\, iP\theta$

$\theta = \frac{\pi}{4} \Rightarrow iP\theta$.

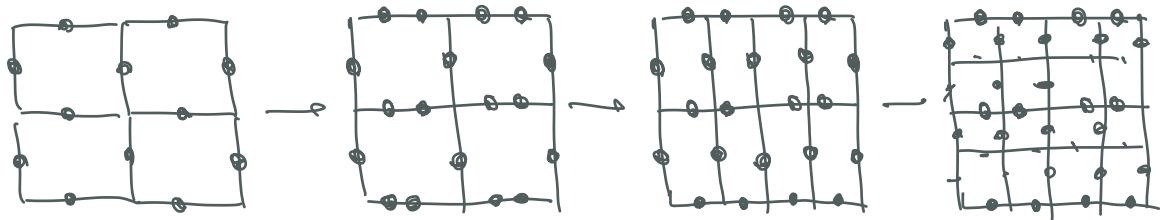
Note that $B_p B_{p'}$ was already in the stab group from the previous lattice. $[B_p B_{p'}, U] = 0$ so it remains. Thus we obtained B_p and $B_{p'}$.

We can also check

U  $U^\dagger =$  $\cdot B_p$

B_p is already in the stab group. So we obtain A_v terms

This algorithm makes twice bigger toric code

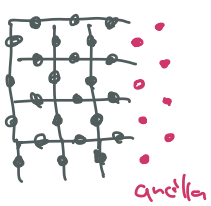


This algorithm prepares the Toric code of size L=2^m in a time step m. One might think that this is quicker than O(L). Here, the point is that we are inserting ancilla qubits at each step, and this procedure actually make qubits travel by the distance O(L). So, in a completely localized geometry, this procedure's circuit depth is O(L).

This encoding circuit generates a structure called MERA where encoding occurs in a hierarchical way, like the opposite of RG transformation!

The aforementioned procedure can be applied to create the toric code in any arbitrary geometry. The strategy is to prepare the smallest possible graph of the desired topology. Then we keep adding qubits and expand the graph to the desired size.

(Truly local method)



put ancilla in the middle by SWAP gates. Then merge them into the toric code.

$Generating the toric code via measurement

We have shown that it takes at least O(L) time to create the Toric code ground state. There is a clever way to “avoid” this lower bound on the circuit complexity by utilizing measurements.

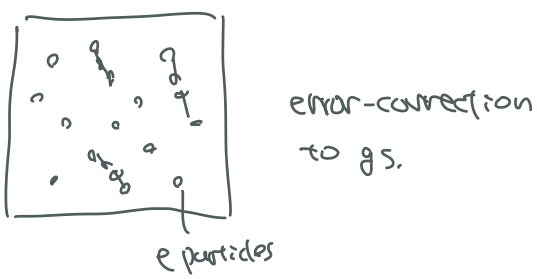
The simplest strategy is to start from a product state |0>, and then protectively measure Av. Letting av = +1, -1 as measurement outcomes. We can then prepare the following state:

$$|\psi\rangle = \prod_v (1 + a_v A_v) |0\rangle^{\otimes n}$$

This is toric code with prob $\sim \frac{1}{2^{n/2}}$!!

The resulting state has Bp = +1 and Av = av. In order to create the Toric code ground state, one needs to make all av = +1. Recall that av = -1 corresponds to the presence of electric charges e at v. Then, by pairing up all the charge excitations, we can create a ground state. Here, pairing up e’s can be done by applying Pauli Z operators along suitable paths. Since the measurement can be performed simultaneously, the whole procedure can be applied in O(1) time step, pretty fast!

So, even though one cannot create the Toric code ground state via local unitary circuit, one can still create the ground state in O(1) time by utilizing measurements. But how did the measurement avoid the O(L) lower bound?



The essential point is that, in order to eliminate anyonic e excitations, one needs to know the locations of all the e particles on the lattice in advance. In other words, after all the Av measurements were performed, the outcome date av needs to be communicated to one location where a classical computer can tell you what is the appropriate way of removing e particles. Here, the measurement outcome needs to travel across the whole lattice, which in principle can take a time proportional to L.

One might think that we could eliminate e particles locally without using any global communications (e.g. by annihilating them with nearby excitations). The circuit complexity lower bound for the toric code essentially says that one cannot eliminate e particles locally. That is, even if some pairs can be annihilated in this way, at the end of the day, some e particles will remain in an isolated manner. In order to annihilate them, we need O(L) time communication.

(One may try to split the lattice into small grids of squares, and annihilate particles inside each square locally. But a particle will remain with probability 1/2. We then move to a larger grid, but again, particle will remain with probability 1/2. Unfortunately, some excitation will remain until the grid becomes order of the system size.)

So, the O(L) lower bound is an absolute thing. The measurement strategy cannot avoid this lower bound. Practically, however, the classical measurement outcomes can travel quite fast (close to the speed of light). So, the non-local interactions are much easier with measurement strategy. For this reason, creating the Toric code ground state via measurement is a promising approach.

Here we point out one caveat in this approach. This approach will create a ground state of the Toric code in |0,0>. One can generalize this and create any stabilizer state of the ground states, stabilized by Pauli logical operators. But creating other superposition states seems not easy with this method. In contrast, the aforementioned algorithm with graph deformation can prepare any codeword state.

$Toric code from a cluster state

We have discussed a method of preparing the Toric code state by measuring 4-body stabilizer generators. This method can be applied to any stabilizer code as long as one can perform error-correction efficiently and bring the system back to Sj=+1 states.

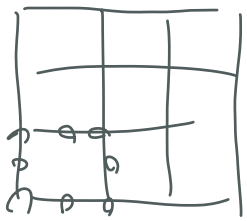
But fault-tolerantly measuring 4-body stabilizer operators can be tricky in real experiments. It turns out that a simpler method exists where one first prepares the cluster state, and then perform local projective measurements. This idea plays a crucial role in fault-tolerant version of the measurement-based quantum computation.

Let us begin with a warm up exercise. Suppose that we have a two qubit state |0,0> which is stabilized by Z1 and Z2. We then perform a measurement of X1X2 (a tensor product, not single Pauli operators). There are two possible outcomes
(1+X1X2)|00> = 00 + 11
(1-X1X2)|00> = 00 - 11
These states can be represented as a stabilizer state. For the case of X1X2=+1, we have the following stabilizers:
Z1Z2, X1X2.
For the other state, we just need to put -X1X2. Here we notice that the stabilizer group has changed as follows because of the measurement:
Z1, Z2 to Z1Z2, X1X2.
Why did this happen? It is because Z1 and Z2 do not commute with X1X2, but their product Z1Z2 commute. In other words, from the stabilizer group, only those which commute with X1X2 survived.

This observation can be generalized to n qubit systems. Consider a state |psi> which is stabilized by the stabilizer group S. We then perform Pauli P measurement. There are two cases to consider.

1. P commutes with S. In this case, P must be inside S (if this is a code, then P may be logical operators. But this is a state, so there is no non-trivial logical operator.) We can then write P as a product of stabilizer generators. As such, the measurement outcome +1, -1 is determined a priori by S (or |psi>). And the state remains the same after the measurement.
2. P anti-commutes with some operator in S. In this case, one can always choose independent stabilizer generators S1, … , Sn such that P commutes with S1, …, Sn-1, and anti-commutes only with Sn [BY-Chuang]. The original state is given by
   (1+S1)…(1+Sn).
   If the measurement outcome was P=+1, we have
   (I+P)(1+S1)…(1+Sn-1)(1+Sn)(1+P) = (1+S1)…. (1+P)(1+Sn)(1+P) = (1+P)(1+S1)… (1+Sn-1)
   where Sn disappears due to the anti-commutation, PSn=-SnP.
   So, the stabilizer group is replaced by S1, … , Sn-1, P.
   If the measurement result was -1, it will be S1, … , Sn-1, -P.

With this result in hand, let us discuss how to convert the cluster state into the Toric code. We will consider a lattice as shown below



where qubits live on both edges and vertices. Starting from a product state |+>, we create the cluster state by applying CZ gates on neighboring qubits. The resulting stabilizer generators are:

Then we perform projective measurements of all the qubits at vertices in X basis. Let's find the resulting stabilizer generators. We notice that the following stabilizer generators commute with X operators. Here we used the fact that one-body X on vertices are in the stabilizer group:

These are stabilizer generators of the Toric code after exchanging X and Z by applying Hadamard gates. Also, string-like Z-type logical operators can be formed:

So, this procedure will create the Toric code in Z-logical basis states. There is a way to modify this protocol so that one can prepare any encoded state, but I will skip this.

In the past few years, there have been significant interests in what kind of entanglement can be generated via measurements. Suppose we start from a product state, and then let the system evolve with some interacting dynamics. We may consider repeated applications of random 2-qubit Clifford operators on neighboring qubits. This will create a highly entangled state. But what happens if we perform measurements during the time evolution? Naively, we would think that measurement will destroy the entanglement, so the system cannot retain high entanglement in the presence of finite rate measurement. But surprisingly, people found that the volume-law entanglement (not area-law) can survive even under the measurement. It turns out that the notion of QEC is central to this understanding. The aforementioned example of the Toric code state is perhaps the simplest example of creating a highly entangled (complex) state via projective measurement (although it is not volume-law). These findings, together with old results on the Toric code, suggest that involving measurements will give rich research avenue which should be explored further, and these may have practical advantages.

$Transversal gate

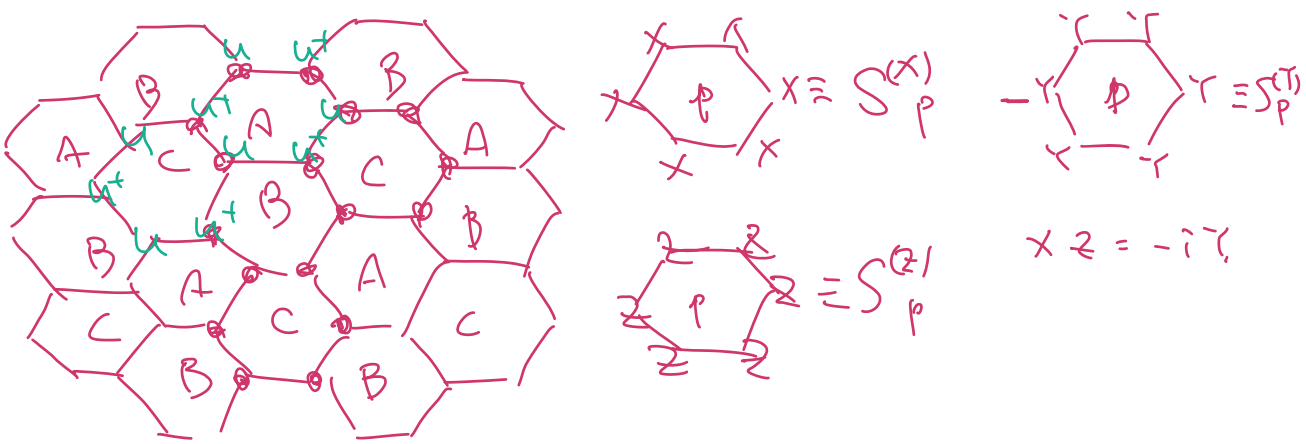
We discussed coding and physical properties of the Toric code in rather details and showed that it can indeed store quantum information securely. In order to perform some useful computational task, however, we need to perform quantum computation by transforming encoded states of logical qubits. In the Toric code, one can perform logical Pauli X, Y, Z transformations easily by implementing string-like logical operators. Since they are written in tensor product form, one can implement them in a unit time step. Logical operators with tensor product form are called transversal operators, and are particularly appealing because, even if one introduces some errors during the implementation of string-like logical operators, this error does not propagate to other qubits. More, generically, fault-tolerant logical gates, which can be implemented by constant depth quantum circuits, will be useful.

We cannot do anything interesting just by applying Pauli operators. In fact, we can't even entangle different logical qubits if we are restricted to Pauli operators. The central question here is what kinds of logical gates can be implemented by a constant-depth quantum circuit. Ideally, we would hope to implement arbitrary logical operators, including the one of the form of superposition of large string-like logical operators, cos(theta) X + I sin(theta)Z, in a fault-tolerant manner. Unfortunately, this is not possible with transversal logical gates, due to the Eastin-Knill theorem. (While one can avoid the Eastin-Knill theorem by looking at approximate error-correction, the gain is still small, see [] for instance.) In fact, in the case of stabilizer codes, one can make a stronger statement that fault-tolerant logical gates, implementable by constant depth circuits, cannot form a universal gate set. The latter result is due to the theorem by Bravyi and Koenig, which we will prove later.

Of course, there are several clever tricks to avoid this restriction and achieve fault-tolerant quantum computation (e.g. code switching, magic state distillation). But here, we will focus on the problem of finding and classifying what kinds of logical gates can be fault-tolerantly implemented. We will focus on a certain generalization of the Toric code, which is called the color code, and discuss its physical and coding properties, including its transversal logical gates.

$Color code

Here we will consider another interesting model of stabilizer Hamiltonians, called the topological color code. The color code is defined on a honeycomb lattice and qubits live on vertices. For now, we consider a model on a torus. The honeycomb lattice has a very special feature that one can put color labels, A, B, C, to each plaquette so that neighboring plaquettes have different color labels. For this reason, we say that the honeycomb lattice is 3-colorable. The color code can be constructed on any 3-colorable lattice. The stabilizer generators of the color code are given as follows:



Here, we have two stabilizer generators (X-type and Z-type) per each plaquette. By multiplying them,

one can construct Y-type stabilizer generators as well. You can check that all the stabilizers commute with each other because they share zero or two qubits. This is actually a generic property of 3-colorable lattice, so the color code can be defined on arbitrary 3-colorable lattice.

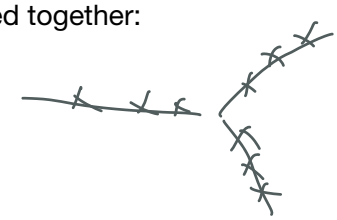
(Logical qubits) The color code on a torus supports 4 logical qubits. To see this, we need to see how many of stabilizer generators are independent. Observe that multiplying X-stabilizers on color A generates X operators acting on all the qubits:

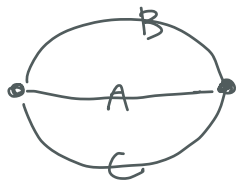
$$\prod_{p\in A} S_p^{(x)} = \prod_{p\in B} S_p^{(x)} = \prod_{p\in C} S_p^{(x)} = X^{\otimes n} \Rightarrow \prod_{p\in A}\prod_{p\in B} S_p^{(x)} = 1 \quad \text{etc.}$$

And a similar relation for Z stabilizers. In total, there are 4 independent ways of creating identity operator from stabilizer generators, so n-4 stabilizer generators are independent, and we have k = 4.

(Logical operators) Next, let us find logical operators of the color code. As shown below, there are 3-types of X-string and Z-string logical operators, associated with 3 different color labels:

However, these three logical operators are not independent from each other. As shown below, they can be fused together:





is a stabilizer.

Hence there are only 2 independent X-type and Z-type logical operators.

For each type of logical operator, one can define anyonic excitations. EA, EB, EC etc. Here unlike the Toric code, we have the following fusion channel of anyons to the vacuum:

$$e_A \times e_B \times e_C = \mathbb{1}$$
$$m_A \times m_B \times m_C = \mathbb{1}$$

Their mutual braiding statistics can be found in the following way:

	m_A	m_B	m_C
e_A	1	−1	−1
e_B	−1	1	−1
e_C	−1	−1	1

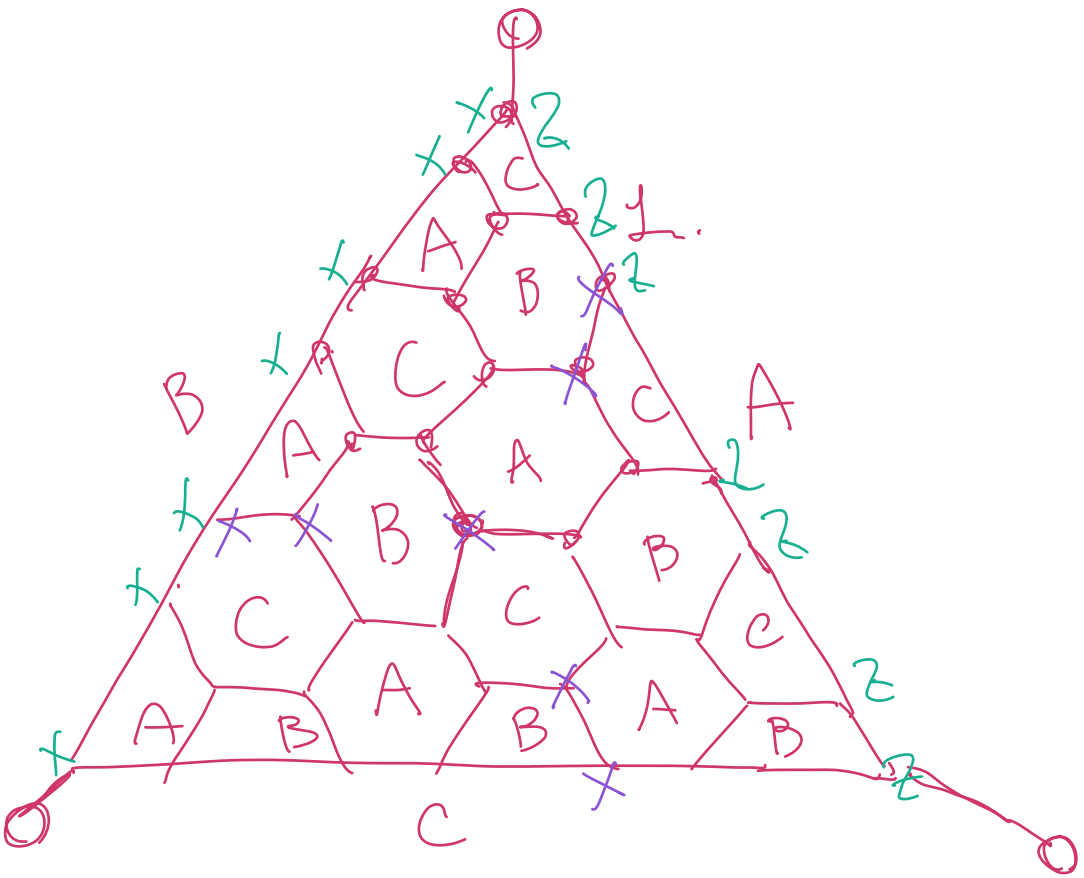
(Toric code) It turns out that the color code on a torus can be converted into 2 copies of the Toric code. This may be guessed by the fact that k=4 is twice of k=2 for the Toric code. Another way is to observe that the above braiding statistics allows us to find the following correspondence:

$$e_A = e_1, \quad m_A = m_2 \quad e_C = e_1 \times e_2$$
$$e_B = e_2, \quad m_B = m_1 \quad m_C = m_1 \times m_2$$

A formal procedure to convert the color code into 2 copies of the Toric code was described in [].

(Properties) Then, why do we bother to consider the color code if it is just 2 copies of the Toric code? One can see that the color code has some symmetries which were not present in the Toric code. For instance, applying a Hadamard gate on every qubit will leave the Hamiltonian invariant.
X to Z, Y to -Y, Z to X.
More generally, we see that any local exchanges of X, Y, Z Pauli operators will leave the ground state space invariant. This suggests that we can implement Clifford logical gates in the color code. (Recall Clifford transform Pauli into Paulis).
$Color code with boundaries

Here we focus on a version of the color code with boundaries which encodes k=1, and show that all the Clifford gates can be implemented in this case. The model is defined as follows:

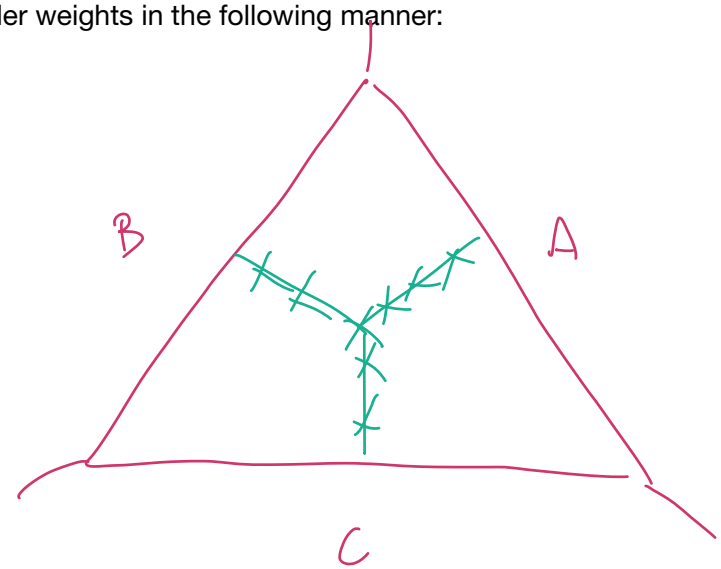


Here it is useful to imagine that there is an additional vertex at “spatial infinity” represented by a white dot. With this vertex, the system is supported on a geometry of a closed sphere. We can also assign colors A, B, C to boundaries by looking at the plaquettes including the vertex at the spatial infinity.

One can compute the number of logical qubits by using combinations tricks, such as the Euler characteristic of a closed sphere (i.e. V-E+F=2). Skipping the detail of this calculation, we find that the color code with boundaries can support a single logical qubit: k=1. Also, the total number of vertices (excluding the spatial infinity) is always odd.

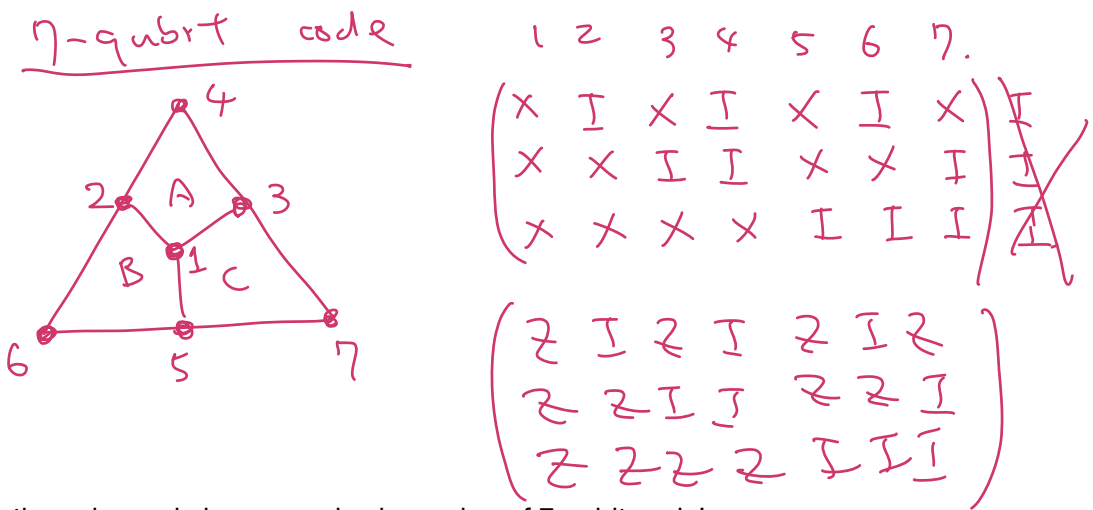
(Logical operators)
Let us find logical operators of the code. It turns out that X^n, Y^n, Z^n are logical operators. To see that, observe that they commute with all the stabilizer generators (which have even weights). Also, one can show that, in this geometry, the total number of qubits is always odd. So, they satisfy the commutation relations of Pauli X, Y, Z. One can then easily see that any Clifford transformation can be implemented in this code, by applying transversal Clifford operators on every qubit.

These are not expressions of minimal weight logical operators. One can construct logical operators with smaller weights in the following manner:



So, the code distance is O(L) wher L is the linear length. One can interprete this as creating mA, mB, mC from each boundary and fuse them into the vacuum in the middle of the system.

Here it is worth noting that the topological color code is a generalization of the 7-qubit code. Indeed, the smallest possible color code with 3 boundaries is given by



So, the color code is a many-body version of 7 qubit code!

$3D color code

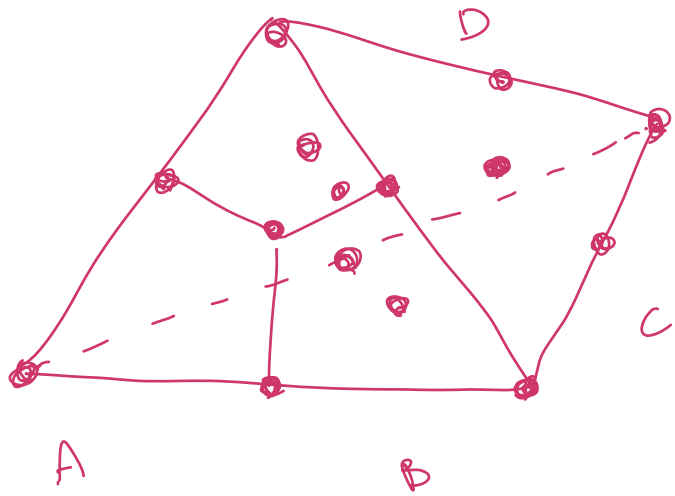
The topological color code can be defined in higher-dimensional systems as well. In D-dimension, one needs to consider D+1 colorable graph where D-dim volumes can have D+1 different color labels. The model is equivalent to D copies of the D-dimensional Toric code. They admit the transversal RD gate, defined as

$$R_D = \begin{pmatrix} 1 & 0 \\ 0 & \exp(i\frac{\pi}{2^D}) \end{pmatrix}$$

For D=2, 3, we have

$$R_2 = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} = S \qquad R_3 = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix} = T$$

Since T is outside the Clifford group, it is very appealing that we can implement transversal T in 3D color code.
Here, we look at the smallest realization of 3D color code:



$$\left(\!\!\left(\begin{matrix} X \text{ on volumes} \\ Z \text{ on faces} \end{matrix} \right)\!\!\right)$$

Here X-stabilizers are defined on volumes while Z-stabilizers are defined on faces. There are in total 15 qubits. One may notice that this code is equivalent to 15-qubit Reed Muller code.

The code does not have the exchange symmetry of X and Z anymore, so not a whole Clifford gate set can be applied. But it admits a transversal logical T gate. Here we prove this.

To see this, it is convenient to label logical states as follows:

$$|+\rangle_L = \left(\frac{1+S_A^X}{2}\right)\left(\frac{1+S_B^X}{2}\right)\cdots|0\rangle^{\otimes 15} \qquad \text{by using the trick of projectors.}$$

$$|1\rangle_L = X^{\otimes 15}|0\rangle_L.$$

Let us study the action of logical T. Notice that X-type stabilizer generators (including products of them) always have weight 8. Hence |0L> consists of terms with the number of 1s being multiple of 8

(zero or 8). So,

$T^8 |0_L\rangle = |0_L\rangle$

As for |1L>, we have 7 or 15 of |1>. Hence we have

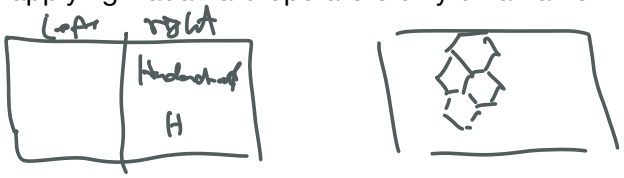
$T^{15} |1_L\rangle = (\sqrt{T})^{-1} |1_L\rangle$

Therefore, we find that the action of T is T^-1. T^{-1}

so implicitly T^{-1} on every qubit will do $\bar{T}$.

$ Transversal gate and domain wall

Transversal logical gates in quantum error-correcting codes have interesting correspondence with classification of domain walls, which is an important topic in studies of topological phases of matter. Consider the 2D color code. We have seen that the transversal application of Hadamard operators leave the Hamiltonian and the ground state space invariant. Here we think of transforming the Hamiltonian by applying Hadamard operators only on a half of the system:



This generated two copies of the 2D color code on the left and the right connected by a gapped domain wall. Namely the terms in the Hamiltonian remains the same except along the domain wall.

Properties of a gapped domain wall can be classified by studying how anyonic excitations behave after crossing the domain wall. In the above example, we have

$m_A \to e_A \qquad m_C \to e_C.$

$m_B \to e_B$

Where anyon labels change upon crossing the domain wall.

One can create a different kind of domain walls by applying different transversal gates partially. If you implement S gate, which transform Pauli operators as follows:

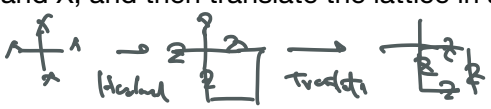
$X \to Y \qquad Y \to -X \qquad Z \to Z$

Anyon labels change as follows:

$m_A \to m_A e_A, \ etc.$

As the above observation implies, when a fault-tolerantly implementable logical gate (by a constant-depth quantum circuit) exists, then one can utilize it to construct a gapped domain wall.

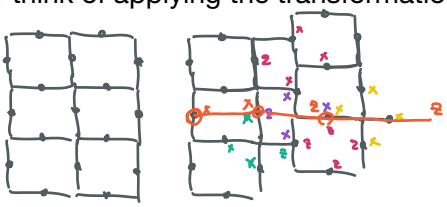
In fact, this correspondence is not restricted to constant-depth quantum circuit. To see an example, we begin by showing that in the toric code, one can implement a Hadamard-like gate by using a lattice translation (which preserves the locality of operators, but cannot be implemented by a local unitary transformation due to the chiral index theorem). Here we implement transversal Hadamard gate to exchange Z and X, and then translate the lattice in a diagonal direction.



This implements the following transformation on logical operators: $\bar{X}_1 \to \bar{Z}_2$, $\bar{X}_2 \to \bar{Z}_1$

Which is Hadamard followed by SWAP.

We now think of applying the transformation partially on the lattice. This will create the following system:



Upon crossing the domain wall, e and m particles get exchanged. This can be seen from the fact that logical operator gets transformed from X to Z. (Putting the boundaries becomes a bit tricky, but we will skip this subtlety).

The key lesson is that domain walls and fault-tolerant logical gates are closely related. This physics becomes even richer in higher dimensions.

$$ Theory of local stabilizer code

We have studied coding and physical properties of local stabilizer codes by focusing on specific examples of the Toric code and the color code. Here, we develop generic tools to study properties of local stabilizer codes.

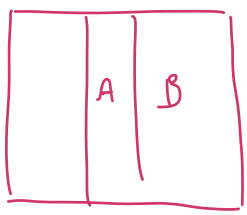
$ Duality lemma

We will begin with the so-called duality lemma which holds for arbitrary stabilizer codes.

Lemma

Consider an arbitrary stabilizer code with k logical qubits. Then we always have
GA + gB = 2k
Where B is a complementary subsystem of A, and gA is the number of independent non-trivial logical operators.

Let us looks at the Toric code to gain some intuition. There are in total 4 independent logical operators. There are 2 string-like logical operators wrapping the torus in the vertical direction, so we have

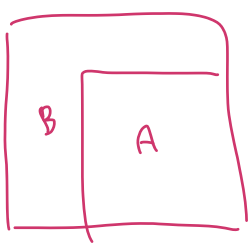


$g_A = 2$

$g_B = 2$

Here note that gA and gB may count the equivalent logical operators. This formula essentially says that one cannot deform the horizontal logical operators and put them in vertical strips.

Next, recall that vertical and horizontal string-like logical operators generate the full non-trivial logical operators. This implies the following:



$g_A = 0$

$g_B = 4$

So, one cannot find any logical operator on A, which is a topologically trivial region.

.

proof

- $n_a + n_b = n$
- we can arrange $n-k$ indep stabilizers as follows

A		B	#	
	⊗	I	S_a	– (a)
I	⊗		S_b	– (b)
	⊗		S_{ab}	– (ab)
			$n-k$	

- logical op in A
  - i) Need to commute with (a) & (ab)

$$2^{2n_a} / 2^{S_a + S_{ab}} = 2^{2n_a - (S_a + S_{ab})}$$

  - ii) But must be non-trivial

$$= 2^{2n_a - 2S_a - S_{ab}}$$

$g_A = 2n_a - 2S_a - S_{ab}$ $\quad g_B = 2n_b - 2S_b - S_{ab}$

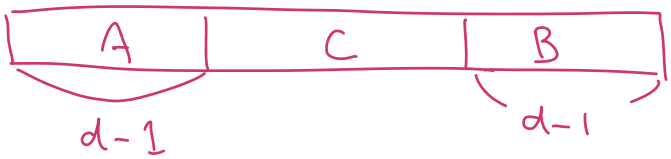
$g_A + g_B = 2(n_a + n_b) - 2(S_a + S_b + S_{ab}) = 2k$

We now discuss two applications of the duality lemma.

Quantum singleton Bound (application 1)

Singleton bound for stabilizer codes.

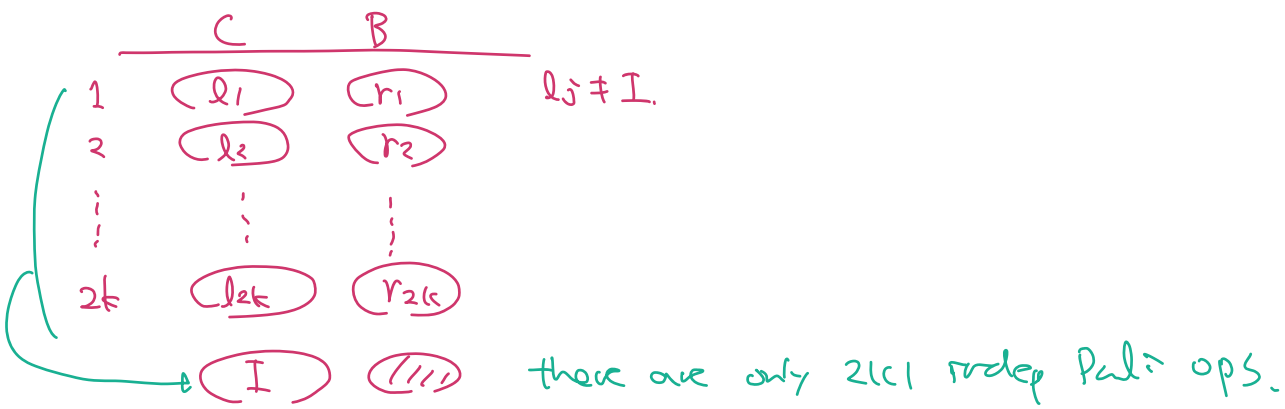
$n-k \geq 2(d-1)$



$g_A = 0 \quad g_B = 0 \quad g_{BC} = 2k$ then $|C| \geq k$

- If $|C| \leq k-1$.

  list all $2k$ logical ops on CB

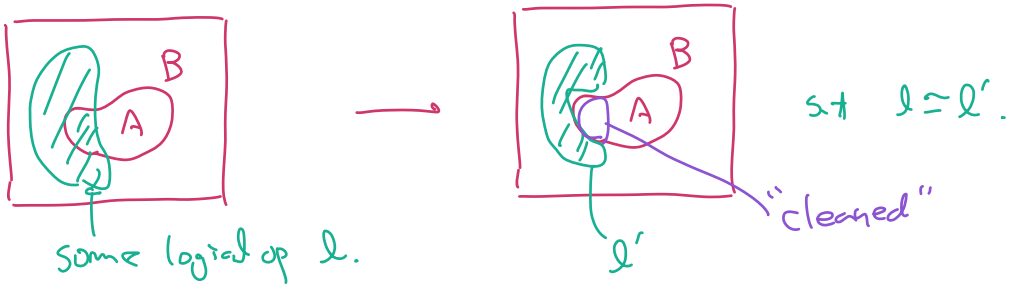


Cleaning Lemma (application 2)

Consider an arbitrary stab code. If there is no (non-trivial) logical op on a subsystem A, then all the logical ops can be supported on $B = A^c$.

proof

$g_A + g_B = 2k$ $\quad g_A = 0 \Rightarrow g_B = 2k$ (there are $2k$ indep logical ops).

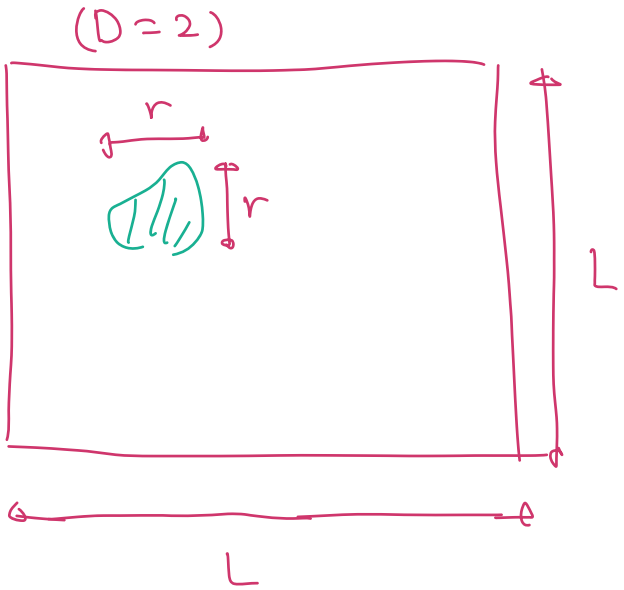


$ Bound on code distance

We now use the developed tools to derive bound on code distance of a local code.

We begin by defining a local code.

Def of local code.



- L is a big integer.
- S_j are geometrically local (in O(1)-box)
- Square (or cubic) lattice
- r is held fixed.

$$H = -\sum_j S_j$$

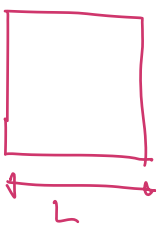
- Strictly, $L \to \infty$ family.

upperbound on code distance

consider a local stabilizer code in D-dimension, then.

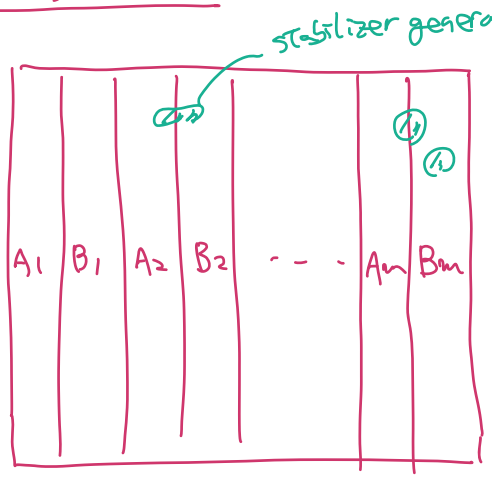
$$d \leq O(L^{D-1})$$

$\uparrow$ code distance



(D=2).

proof (D=2)



$A = \bigcup_j A_j$ $B = \bigcup_j B_j$

$\partial_A + \partial_B = 2k$

WLOG $\partial_A \neq 0$.

$\Rightarrow \exists$ logical op ℓ supported on A

$\ell = \ell_1 \otimes \cdots \otimes \ell_m$ // supported on A_m

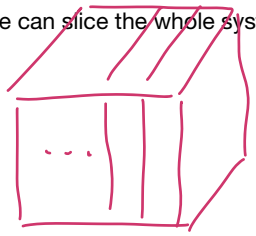
for some j. ℓ_j must be logical

i) $[\ell_j, S] = 0$

ii) $\{\ell_j, r\} = 0$ (r pair of ℓ)

iii) supported on A_m

In 3D, we can slice the whole system as shown below.

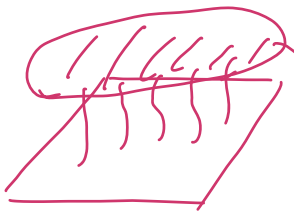


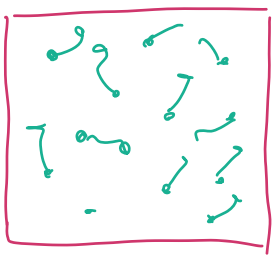
3D

Comment on self-correcting memory

- 2D code (including toric code) has stray logical op

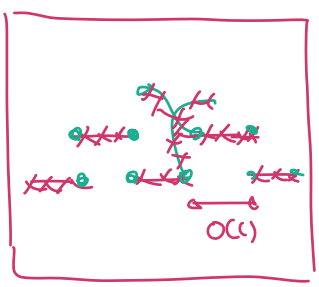
  A quantum memory at finite temperature??





Finite energy density state

Anyons are particles with constant energy. can move without energy penalty

- 

  Anyon trajectory crosses after O(1) time

  ⇒ Quantum Memory time $\tau \sim O(\beta)$

  • Energy Barrier $\tau \simeq \exp\left(\frac{\text{Barrier}}{T}\right)$

Need to perform error-correction before τ.

Self-correcting memory

$\tau \to \infty$ as $n \to \infty$ (No error-correction needed)

i) 4D toric code is self-correcting. $\exp(L)$.

ii) 2D local code is not! (string op)

iii) 3D ?? fracton code ?? Welded code ??

Storage Capacity of local codes

In 2-dim, any constraints on $[n, k, d]$?

Claim

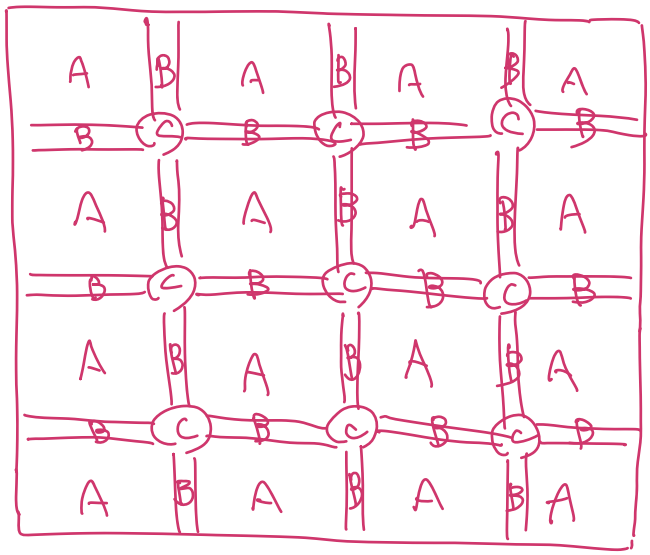
$k d^2 \leq O(n)$ $\qquad$ $k d^{\frac{2}{D-1}} \leq O(n)$ for D-dim

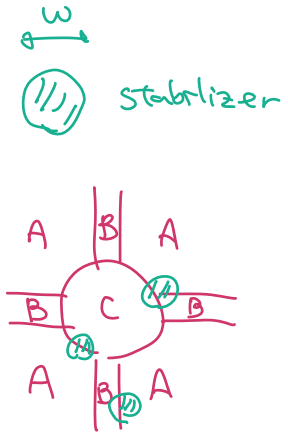
comment

2-dim toric code

$k = 2$ $\quad d = \sqrt{n}$ saturating the bound.

proof





- stabilizer overlaps with neighbouring regions only
- No stab acts on two different A's.

(a) $g_B = 0$ $\quad g_C = 0$ because B & C are smaller than d.

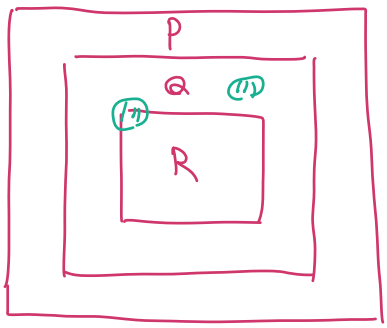
It turns out $g_A = 0$ too !! — (i).

If so, $g_{BC} = 2k$

(b). $g_B = 0$ $\quad g_{BC} = 2k$ implies $k \leq |C| \sim O(\frac{n}{d^2})$ — (ii)

use the proof of the singleton bound.

Lemma (holographic principle)



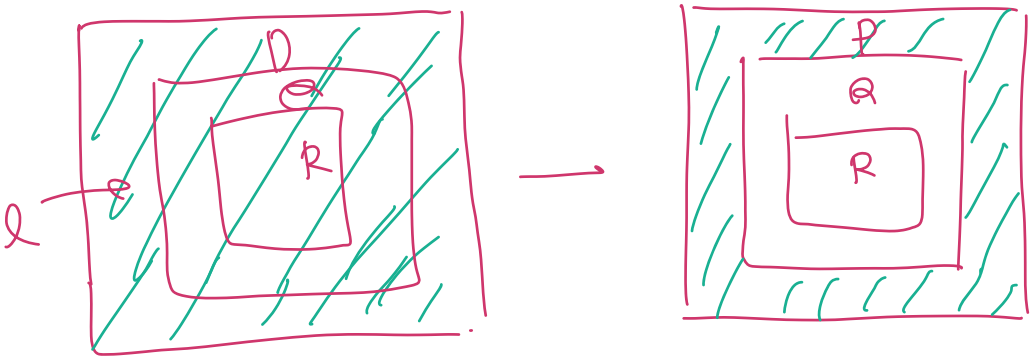
$\exists w$.
- Assume that there is a logical op on $P \cup Q \cup R$. call it l.
- Assume that $g_R = 0$ $g_Q = 0$.

$\Rightarrow$ l can be supported on P.

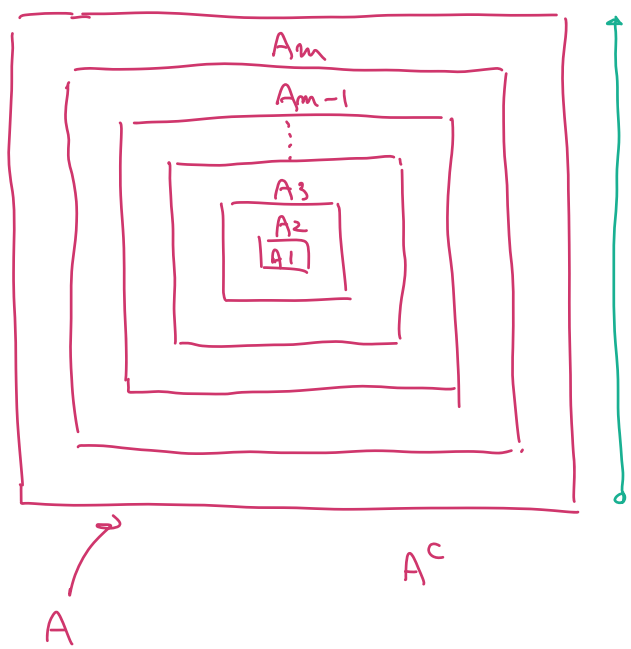
Proof

For simplicity $g_{PQR} = 1$ $g_Q = 0$ $g_R = 0$

$g_{\overline{PQR}} = 2k-1$ $g_{\overline{PQR}Q} = 2k-1 \Rightarrow g_{PR} = 1$.



proof of (i)



A does not support a logical op. Use the holographic principle from the interior.

$ Restriction on fault-tolerant logical gates.

Next, we will discuss what kinds of logical gates can be implemented in a fault-tolerant manner in D dim local code.

$ Clifford Hierarchy

Here we begin by introducing the notion of Clifford hierarchy.

One confusing point is that Clifford hierarchy is a set, not a group. This can be seen easily from the fact that C3 contains all the Clifford operators, as well as T gate. If C3 were a group, these would generate dense operators. But C3 is just a set, so C3 is still a finite set of operators.

Clifford hierarchy

0th level	U(1) phase
1st level	Pauli ops
2nd level	Clifford
3rd level	T gate, CCR etc
⋮	⋮

Def

commutator $K(A,B) \equiv ABA^\dagger B^\dagger$

$C_{m+1} \equiv \{U : \forall P \in \text{Pauli} \quad K(U,P) \in C_m\}$

Examples

- $C_1 = \{U : K(U,P) \in U(1)\}$ C_1 is Pauli op set
- $C_2 = \{U : K(U,P) \in C_1\}$ $UPU^\dagger P^\dagger = \text{Pauli}$

  C_2 is Clifford (transform Pauli to Pauli)

Examples

- $R_m = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/2^m} \end{pmatrix}$

$R_0 = \mathbb{1}$ $\in C_0$

$R_1 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = Z$ $\in C_1$

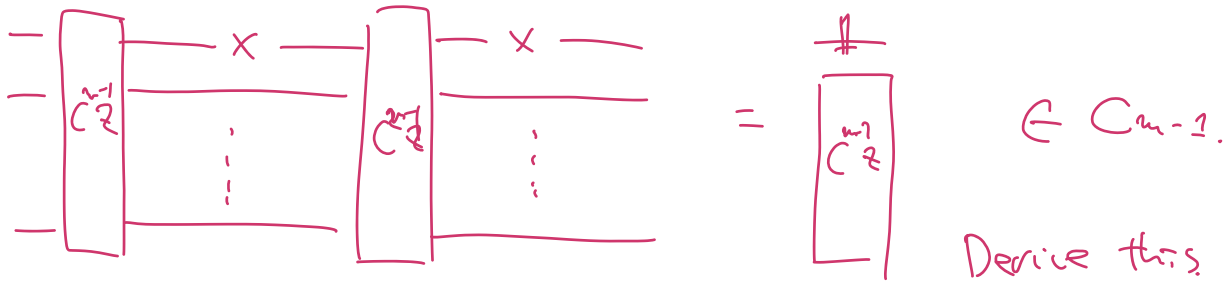
$R_2 = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} = S$ $\in C_2$

$R_3 = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{\sqrt{i}} \end{pmatrix} = T$ $\in C_3$

$R_m \times R_m^\dagger \times \propto R_{m-1}$ ← Derive this

- $C^{\otimes m-1} Z \,|\, i_1, \cdots, i_m) = (-1)^{i_1 \cdots i_m} |i_1, \cdots, i_m)$

  add -1 only when $i_1 = \cdots = i_m = 1$.



$\in C_{m-1}$.

Derive this.

Remarks

- Reed-Muller code ($2^{m+1}-1$ qubit) ($k=1$) ($d=3$)
  R_m gate
- Hypercube code (2^m qubit) ($k=1$) ($d=2$)
  $C^{m-1}Z$ gate.
- $C_m \supset C_{m-1}$
- C_m is not a group in general. (explain why).

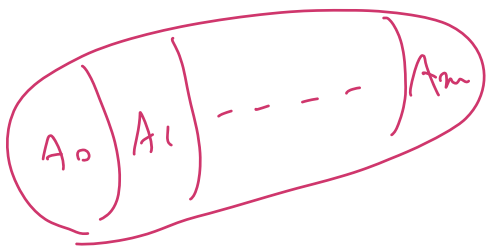
Transversal implementable logical operators are quite restricted as summarized in the following lemma.

(Partition lemma for transversal gates)

Transversal gate

Lemma

Consider an n-qubit stabilizer code,



Assume that

$g_{A_j} = 0$ (no logical op).

Then a transversal gate $\leq C_m$

proof

- Recall that

$$g_A + g_{\bar{A}} = 2k$$

$$g_{A_j} = 0 \Rightarrow g_{\overline{A_j}} = 2k.$$

Induction.

A transversal logical op supported on $s+1$ regions belongs to C_s.

Let's assume this for s & prove this for $s+1$.

$\bar{P} \equiv 1\ P_1 \cdots P_m$

$\ell \equiv \ell_0\ \ell_1 \cdots \ell_s\ 1 \cdots 1.$

$k(\ell, \bar{P}) \equiv 1 \cdots 1 \cdots 1 \in C_s.$

$\Leftrightarrow \ell \in C_{s+1}.$

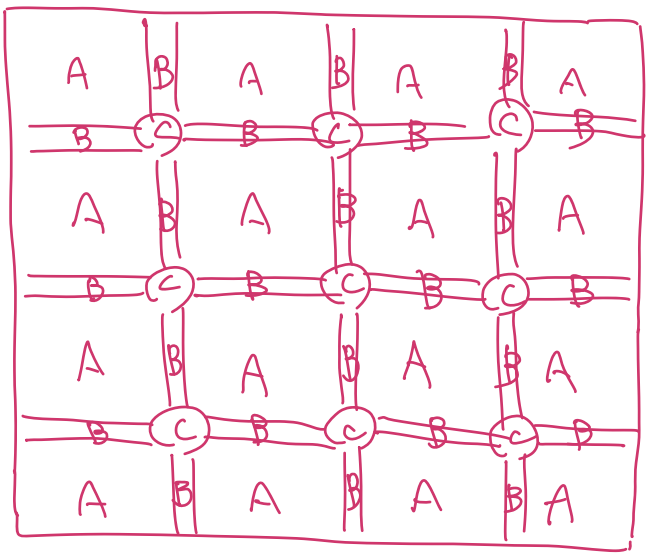
The above lemma enables us to prove a fundamental result due to Bravyi and Koenig. For simplicity, we will focus on transversal logical gates, but argument can be applied to locality preserving gates.

- **Bravyi-Koenig Bound.**

  In D spatial dimension, transversal gates are in C_D.

  eg). topological color code R_D is transversal.

  proof ($D=2$)

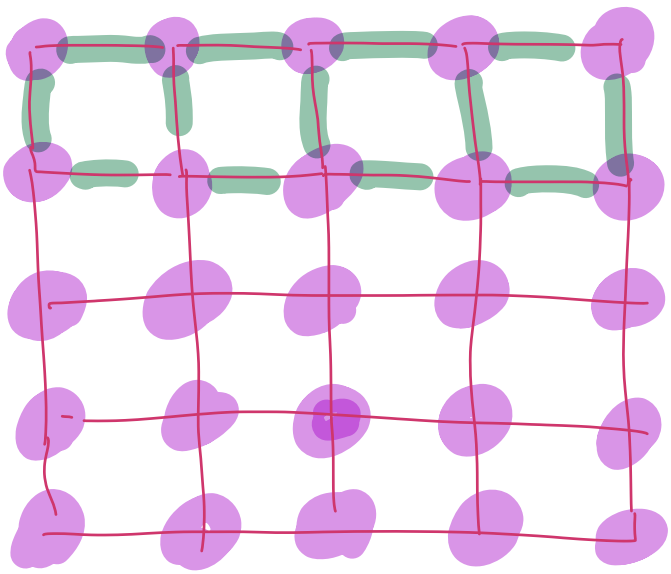


$g_A = 0$

$g_B = 0$

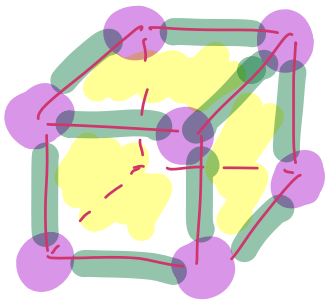
$g_C = 0$

$\Rightarrow$ 2nd level !!



- Fatten 0-cells
- Fatten 1-cells

- $D = 3$



- Fatten 0-cells
  1-cells
  2-cells

$g_A = g_B = g_C = g_D = 0$

$\Rightarrow$ 3rd level !!

Erasure threshold

If the code has a loss tolerance rate $p_L > 1/m$, then gate is restricted to C_{m-1}.

proof

All m regions should satisfy

$g_{A_0} = \cdots = g_{A_{m-1}} = 0$ (with high probability)

$\Rightarrow$ gate is in C_{m-1}.

No-go for self-correction

If 3d code has non-Clifford, then sym op exists.

proof

If no sym., then the code has Clifford only.

Distance Bound

In general $d \leq L^{D-1}$.

If the code has C_m, then

$$d \leq L^{D+1-m}.$$