Quantum Information Theory - 67749
Recitation 3, June 4, 2025

1 Recitation Overview - Quantum Error Correction Codes.

In the last lectures, we introduced quantum channels - physical maps between density matrices and mentioned that they are used to represent noise, an error that might occur during either computation or communication. Faults and errors are not unique to the quantum regime and were studied before in the context of (classical) error correction codes and fault tolerance. In this recitation, we are first going to represent what classical error correction codes are, and we will give the Tanner codes as an example of the construction of such codes. Then we will talk about the standard way to model quantum noise, give a basic example of a first quantum code, and then eventually we will give the Toric code.

Part I
Classicl Error Correction Code.

2 Codes in General. Notations and Definitions.

A code is a subset, which one can think of as the valid points. Here we focus only on linear binary codes, which are linear subspaces of $\mathbb{F}_2^n$. As implied at the beginning, codes are used to correct errors, a correctable error is one that gets a valid point into its local environment. A common way to measure resilience is to ask how many bits an evil entity needs to flip such that the corrupted vector will be closer to another vector in that space than the original one. Those ideas were formulated by Hamming [Ham50], who presented the following definitions.

Definition 2.1. *Let $n \in \mathbb{N}$ and $\rho, \delta \in (0,1)$. We say that C is a **binary linear code** with parameters $[n, \rho n, \delta n]$. If C is a subspace of $\mathbb{F}_2^n$, and the dimension of C is at least ρn. In addition, we call the vectors belong to C codewords and define the distance of C to be the minimal number of different bits between any codewords pair of C.*

From now on, we will use the term code to refer to linear binary codes, as we don't deal with any other types of codes. Also, even though it is customary to use the above parameters to analyze codes, we will use their percent forms called the relative distance and the rate of code, matching δ and ρ correspondingly.

Definition 2.2. *A **family of codes** is an infinite series of codes. Additionally, suppose the rates and relative distances converge into constant values ρ, δ. In that case, we abuse the notation and call that family of codes a code with $[n, \rho n, \delta n]$ for fixed $\rho, \delta \in [0,1)$, and infinite integers $n \in \mathbb{N}$.*

Notice that the above definition contains codes with parameters attending to zero. From a practical view, it means that either we send too many bits, more than a constant amount, on each bit in the original message. Or that for big enough n, adversarial, limited to changing only a constant fraction of the bits, could disrupt the transmission. That distinction raises the definition of good codes.

Definition 2.3. *We will say that a family of codes is a **good code** if its parameters converge into positive values.*

Example 2.1 (The Check-sum/Even-parity Code.)**.** *The parity check code $C_p \subset \mathbb{F}_2^{n+1}$, in which its codewords are only the vectors with even parity.*

$$101 \mapsto 1010$$
$$111 \mapsto 1111$$
$$000 \mapsto 0000$$

Example 2.2 (The Repetition Code.)**.** *define the repetition code $C_r \subset \mathbb{F}_2^{n \cdot r}$, In which, for a fixed integer r, any bit of the original string is duplicated r times.*

$$1 \mapsto 1111$$
$$0 \mapsto 000$$

Having both examples in mind, we get a feeling of the behavior of the distance-rate trade-off. To summarize, using a simple construction, one could construct the codes $[r, 1, r]$ and $[r, r-1, 2]$. Each has a single perfect parameter, while the other decays to the worst.

2.1 Singleton Bound

Exercise 2.1. *Besides*

Besides being the first bound, Singleton bound demonstrates how one could get results by using relatively simple elementary arguments. It is also engaging to ask why the proof yields a bound that, empirically, seems far from being tight.

Theorem (Singleton Bound.)**.** *For any linear code with parameter* $[n, k, d]$*, the following inequality holds:*

$$k + d \leq n + 1$$

Proof. Since any two codewords of C differ by at least d coordinates, we know that by ignoring the first $d-1$ coordinate of any vector, we obtain a new code with one-to-one corresponding to the original code. In other words, we have found a new code with the same dimension embedded in $\mathbb{F}_2^{n-d+1}$. Combine the fact that dimension is, at most, the dimension of the container space, we get that:

$$\dim C = 2^k \leq 2^{n-d+1} \Rightarrow k + d \leq n + 1$$

□

It is also well known that the only binary codes that reach the bound are: $[n, 1, n]$, $[n, n-1, 2]$,$[n, n, 1]$ [AF22]. In particular, there are no good binary codes that obtain equality(And no binary code which get close to the equality exits). Let's review the polynomial code family [RS60], which is a code over none binary field that achieve the Singleton Bound.

Next, we will review Tanner's construction, that in addition to being a critical element to our proof, also serves as an example of how one can construct a code with arbitrary length and positive rate.

3 Tanner Code

The constructions require two main ingredients: a graph Γ, and for simplicity, we will restrict ourselves to a Δ regular graph, Yet notice that the following could be generalize straightforwardly for graphs with degree at most Δ. The second ingredients is a ;small' code C_0 at length equals the graph's regularity, namely $C_0 = [\Delta, \rho\Delta, \delta\Delta]$. We can think about any bit string at length Δ as an assignment over the edges of the graph. Furthermore, for every vertex $v \in \Gamma$, we will call the bit string, which is set on its edges, the local view of v. Then we can define, [Tan81]:

Definition 3.1. *Let* $C = \mathcal{T}(\Gamma, C_0)$ *be all the codewords which, for any vertex* $v \in \Gamma$*, the local view of* v *is a codeword of* C_0*. We say that* C *is a* ***Tanner code*** *of* Γ, C_0*. Notice that if* C_0 *is a binary linear code, So* C *is.*

Example 3.1. *Consider the Petersen graph* Γ*, which is a regular graph with degree* 3*. Let* C_0 *be the set of all words with even parity. It follows that* C_0 *contains all even-length binary strings of length* 3*:* 000*,* 110*,* 101*, and* 011*. However, the size of* $\mathcal{T}(\Gamma, C_0)$ *is significantly larger, as shown in Figure fig. 1. Specifically, any rotation of the inner and outer cycles simultaneously gives rise to another valid codeword, so any assignments that are not invariant under these rotations would produce five additional valid codewords.*

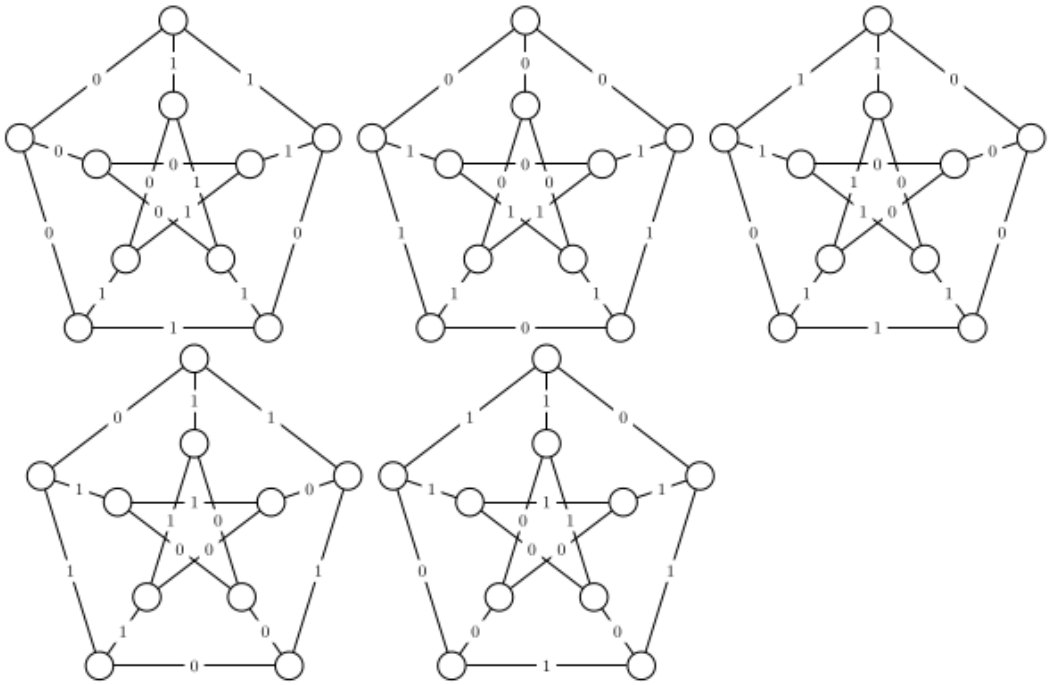

Figure 1: Peterson Graph.

Lemma 3.1. *Tanner codes have a rate of at least* $2\rho - 1$.

Proof. The dimension of the subspace is bounded by the dimension of the container minus the number of restrictions. So assuming non-degeneration of the small code restrictions, we have that any vertex count exactly $(1-\rho)\,\Delta$ restrictions. Hence,

$$\dim C \geq \frac{1}{2}n\Delta - (1-\rho)\,\Delta n = \frac{1}{2}n\Delta\,(2\rho - 1)$$

Clearly, any small code with rate $> \frac{1}{2}$ will yield a code with an asymptotically positive rate □

Based on Lemma 3.1, we can obtain a recipe for constructing codes with a almost non-vanishing rate for arbitrarily large lengths and dimensions. This recipe involves concatenating a series of Tanner codes over complete graphs. To be more precise, we can define a family of codes as follows:

$$\begin{aligned} C_{i+1} &= \mathcal{T}\left(K_{n(C_i)+1}, C_i\right) \\ C_0 &= \text{ Some simple } \Delta[1, \rho_0, \delta_0] \text{ code.} \end{aligned}$$

Where $n(C_i)$ represents the code length of the ith code. Repeating the process described above $\log^*_\Delta(n)$ times allows us to extend the initial code $\Delta[1, \rho_0]$ to $n[1, \sim 2\rho^{\log^*_\Delta(n)}]$. Interestingly, any family of finite groups generated by a constant-size generator set can define a family of codes by utilizing their Cayley graphs as a basis for Tanner codes.

Once we have seen that Tanner codes enable us to achieve rates, the next natural question to ask is about the distance of the codes. Achieving a linear distance requires a little bit more from the graphs, but to understand this idea better, let us return to the repetition code. For instance, the repetition code can be presented as a Tanner code over the cycle graph.

Example 3.2. *In this representation, each vertex checks if the bits on its edges are equal. A valid codeword is an assignment in which all the bits are equal, since otherwise, there would be an edge with no supporting vertex. An illustration of a legal assignment is provided in section 3.*

Recall that the distance of a linear code is the minimal weight of the non-zero codewords. Consider a codeword $c \in C$ and group the vertices by four sets V_i such that V_i is the set of vertices that see $i \in \{0,1\}^2$. Since $c \in C$, we have that $|V_{10}| = |V_{01}| = 0$. Additionally, any vertex in V_{00} is not connected to V_{11}, which gives us two possible cases: either all the vertices in V_{11} are isolated, or the graph is not connected. Hence, the distance of the code is equal to $\frac{1}{2}\sum |V_i| \cdot |i| = \frac{1}{2} 2 \cdot n = n$.

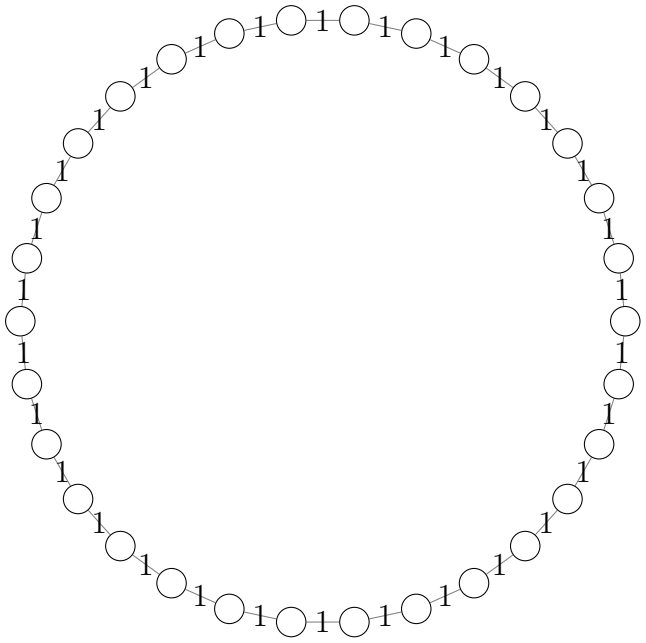

Figure 2: The 1^n assignment on the cycle graph. Any vertex compute parity $1+1=0$, therefore all the restrictions are satisfied and $1^n \in \mathcal{T}$ (cycle , parity).

It is worth mentioning that, in the literature, the repetition code is not usually given as an example of a Tanner code. However, this example will come up again later in the chapter on quantum codes, when we discuss the Toric code, its relation to the hyperproduct code, and how it can be seen as a hyperproduct of two cycle codes.

Furthermore, analyzing the repetition code gives a clue as to how, in certain cases, one might prove a lower bound on the code distance. We would like to say

that, if the weight of the code word is below the distance, then it must be that there is at least one vertex that has a non-trivial local view which is not a codeword in C_0. Put differently, we cannot spread a small weight codeword over $\{V_i\}$, defined above, without expanding into subsets corresponding to low $|i|$. Next, we are going to present the Expander codes, which are Tanner codes constructed from graphs with good algebraic expansion.

Part II
Quantum Error Correction Codes.

4 Introduction.

It is widely believed that quantum machines have a significant advantage over classical machines in a wide range of computational tasks [Gro96], [AK99]. Simple algorithms which can be interpreted as the quantum version of scanning all the options, reducing the running time by the square root of the classical magnitude. Nevertheless, Shor has demonstrated a polynomial-depth quantum circuit that solves the hidden abelian subgroup problem [Sho97], which is considered a breakthrough, as it made the computer science community believe that a quantum computer could offer an exponential advantage.

Despite a consensus on the superiority of the ideal quantum computing model, it is still uncertain whether it is possible to implement such a machine in a noisy environment. Still, simply pointing out the existence of noise is not sufficient to negate the feasibility of computation. Evidence of this is that classical computers also experience a certain rate of errors. Therefore, to fully comprehend the difficulty, let us compare two main factors that made it a challenging task.

First is the magnitude of the error rate; classical computers also have errors, and sometimes we witness system failures (e.g. the blue screen). The error rate of modern computers is so low that the probability of errors propagating stays negligible, even if the length of the computation is polynomial in the scale of what is considered a reasonable input size. It's worth mentioning that in exascale computing, when supercomputers perform around 10^{18} operations per second, it is difficult to overlook faults. In quantum computing, we become aware of their existence much earlier.

The second difference, which is a subtle point, is that quantum states are susceptible to additional types of errors. In addition to the possibility of bit-flip errors, a quantum state may also experience a change in phase. For example, consider the initial state $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, and suppose that due to noise the state is transformed into $\frac{1}{\sqrt{4}}\left(\sqrt{3}\,|0\rangle + |1\rangle\right)$. Classical circuits are oblivious to such faults, meaning that their operation would remain unchanged as no error has occurred. Quantum circuits, however, are usually affected and may fail. Furthermore, when designing a decoder for quantum error correction codes, one must ensure that the decoding process does not introduce bit-flip errors if a classical code is used to protect against phase flips.

5 Quantum Noise.

Definition 5.1 (Bit and pahse flip.)**.** *Consider a quantum state $|\psi\rangle$ encoded in the computation base. We will say that a bit-flip occurs in a scenario the operator Pauli X is applied on one of our state's qubits. The bit-flip event could be considered as exactly as the standard bit-flip error in the classical regime. Similarly, phase-flip occurs when the Pauli Z is applied on one of the qubits.*

However, even though quantum noise is so violent, it has been proven that any ideal circuit of polynomial depth can be transformed into a robust circuit at poly-logarithmic cost [AB99]. In other words, there is a threshold: if physicists provide qubits and a finite gate set with a noise rate below that threshold, then BQP, the class of polynomial-time ideal quantum computation, is feasible and can be computed on a realistic machine.

The basic idea in [AB99] was to show the existence of quantum error correction codes, which would enable logic operations to be performed in a way that prevents errors from propagating. This process involves separating the operation into two stages: the operation itself and an error correction stage. This comes with an additional cost in terms of space and time, but it can reduce the probability of the final state being faulty. The trade-off between the resources needed and the rate of decrease defines the threshold. If the balance is positive, then the process can be repeated in a recursive manner, and after log-log iterations, the failure probability will decay to zero. At the same time, the circuit will scale to a maximum of poly-logarithmic width and depth.

Let's return to the repetition code presented in Chapter 2. We would like to have an analog; a first and natural attempt might consider duplicating copies of the state. Unfortunately, copying a general state is not a linear operation and therefore cannot be done in the circuit model (or any other believed to be feasible). In particular, there is no circuit U which can simultaneously duplicate the states $|0\rangle, |1\rangle, |+\rangle, |-\rangle$.

To overcome the issue, Shor came up with the nine-qubit code [Sho95], which at first glance might seem a naive straightforward implementation of "duplication", but instead uses a clever insight about quantumness in general. Any operation can be seen as a linear (and even unitary) operation over a subspace embedded in a large enough dimension. The encoding is given as follows:

$$|\bar{0}\rangle = \frac{1}{2\sqrt{2}}\left(|000\rangle + |111\rangle\right)^{\otimes 3}$$
$$|\bar{1}\rangle = \frac{1}{2\sqrt{2}}\left(|000\rangle - |111\rangle\right)^{\otimes 3} \ .$$

For convenience, let us use the notation $|\mathbf{GHZ}^{\pm}\rangle = |0^m\rangle \pm |1^m\rangle$. We can also consider the Shor code over m^2 qubits, which is defined as above, such that any logical state contains m products over m qubits. Therefore, the state $|\bar{0}\rangle$ over m^2 qubits can be written as $|\mathbf{GHZ}^{+}\rangle^m$. We are now ready to prove a statement regarding the robustness.

Lemma 5.1. *The Shor code over 9 qubits enable to correct a single either bit or phase flip.*

It is evident that a single bit-flip error can be handled in the same way as in the conventional case. The decoder will check if any of the triples have the same value, and if not, it will correct it by majority. To create a decoder that can also correct a phase-flip error, we need the following statement. In this chapter, we denote the Hadamard gate over m qubits as H^m.

Claim 5.1. $H^m \left|\mathbf{GHZ}^{\pm}\right\rangle = \sum_{x\cdot\mathbf{1}=_2\pm} |x\rangle$

Proof.

$$H^m \left|\mathbf{GHZ}^{\pm}\right\rangle = H^m \left|0^m\right\rangle \pm H^m \left|1^m\right\rangle = \sum_{x\in\mathbb{F}_2^m} |x\rangle \pm \sum_{x\in\mathbb{F}_2^m} (-1)^{x\cdot\mathbf{1}} |x\rangle$$
$$= \sum_{x\in\mathbb{F}_2^m} \left(1 \pm (-1)^{x\cdot\mathbf{1}}\right) |x\rangle = \sum_{x\cdot\mathbf{1}=_2\pm} |x\rangle$$

□

Now it is clear how to correct a phase flip. One can apply the Hadamard transform and compute the parity of each triple. By the assumption that only a single phase flip may occur, either all the triples have the same parity or the faulted one has an opposite parity and needs to be corrected. Thus, we obtain an $[[9, 1, 3]]$ quantum error correction code. Asymptotically, this is an $[[m^2, 1, m]]$ code.

6 CSS Codes.

The Shor code is a specific case of the more general CSS (Calderbank-Shor-Steane) code [CS96]. A family composed by two binary codes C_X, C_Z such that $C_Z^{\perp} \subset C_X$.

Definition 6.1 (CSS Code)**.** *Let C_X, C_Z classical linear codes such that $C_Z^{\perp} \subset C_X$ define the $Q(C_X, C_Z)$ to be all the code words with following structure:*

$$|\mathbf{x}\rangle := |x + C_Z^{\perp}\rangle = \frac{1}{\sqrt{C_Z^{\perp}}} \sum_{z\in C_Z^{\perp}} |x + z\rangle$$

Clearly, the codewords are all the codewords in C_X which don't belong to $C_Z^{\perp}$ and therefore the dimension of the quantum code is $\dim Q(C_X, C_Z) = \dim C_X - \dim C_{Z^{\perp}} = \dim C_X + \dim C_Z - n$. Yet, it's not stems immediately how one can correct faults. Next, we are going to repeat the decoding process of the Shor code in the general setting of CSS codes.

Lemma 6.1. *Let C_X, C_Z classical codes such $Q(C_X, C_Z)$ is a CSS code. Let d_X be the minimal weigh of codeword in C_X which is not in $C_Z^{\perp}$, and define by the same way d_Z to be the minimal weight of codeword in C_Z which doesn't belong to $C_X^{\perp}$. Then the distance of $Q(C_X, C_Z)$ equals to $\min d_X, d_Z$. Moreover there is a decoder which correct any fault with weight at most $d/2$.*

Proof. First let us prove the following claim:

Claim 6.1. *Denote by $H^{\otimes n}$ the Hadamard gate over n qubits. Then for any code C it holds that: $H^n |C^\perp\rangle = |C\rangle$*

Proof.

$$\begin{aligned} H^n |C^\perp\rangle &= \frac{1}{\sqrt{|C_Z^\perp|}} \sum_{z \in C_Z^\perp} H^n |z\rangle = \frac{1}{\sqrt{2^n}} \frac{1}{\sqrt{|C_Z^\perp|}} \sum_{z \in C_Z^\perp} \sum_{y \in \mathbb{F}_2^n} (-1)^{\langle z,y \rangle} |y\rangle \\ &= \frac{1}{\sqrt{2^n}} \frac{1}{\sqrt{|C_Z^\perp|}} \sum_{z \in C_Z^\perp} \left(\sum_{y \in C_Z} |y\rangle + \text{ other terms } \right) \end{aligned} \tag{1}$$

Since the columns of matrix H_Z form a basis for the complementary space $C_Z^\perp$, and due to the dimensional theorem and the equivalence between the row rank and column rank of a matrix, we can deduce that $\dim \operatorname{rank} H_Z^\top + \dim \ker H_Z = n$, which implies that $|C_Z^\perp||C_Z| = 2^n$. Thus the norm of

$$\frac{1}{\sqrt{2^n}} \frac{1}{\sqrt{|C_Z^\perp|}} \sum_{z \in C_Z^\perp} \sum_{y \in C_Z} |y\rangle = \sqrt{\frac{|C_Z^\perp|}{2^n}} \sum_{y \in C_Z} |y\rangle \tag{2}$$

equals 1 and the summation over the vectors $y \notin C_Z$ in the inner closure in equation 1 must cancel. So we left only with a uniform superposition over the codewords in C_Z, Or in other words $H^n |C^\perp\rangle = |C\rangle$. □

Claim 6.1 states that, when considering CSS codes, pauli X operators can be seen as pauli Z operators in the rotated frame. That it,

$$H^n X^f |C_Z^\perp\rangle = \overbrace{H^n X^f H^n}^{Z^f} H^n |C_Z^\perp\rangle = Z^f |C\rangle$$

That insight hints a description to decoder for the quantum code. If one knows how to correct errors for each of the classical code C_X, C_Z than he can start by correct the bit flips in using the decoder of C_X, rotate the state by applying the Hadamard transform and then correct, what was before the transformation phase flips and now a bit flips by using the decoder of C_Z. Then in the end applying the Hadamard transform again for backing to the initial computation space. Indeed that decoder correct $\min\{d(C_X)/2, d(C_Z)/2\}$ errors. Yet more work is needed to show that this decoder also correct $d(Q(C_X, C_Z))/2$ errors.

Let us assume the existences of decoders of the classical codes C_X and C_Z, Denoted $D_X : \mathbb{F}_2^n \to C_x$ and D_Z. In particular for any $\xi \in \mathbb{F}_2^n$, $D_X(\xi) = \arg\min_{x \in C_X} |x + \xi|$.

Claim 6.2. *For any $x_0 \in C_X$ and $z_1 \neq z_2 \in C_z^\perp$, D_X correct $|x + z_1 + f\rangle$, $|x + z_2 + f\rangle$ into two different words in C_X.*

Proof. Suppose not, namely there exists $y \in C_X$ such that D_X correct $|x + z_1 + f\rangle$, $|x + z_2 + f\rangle$ into $|y\rangle$. Then we have that for both $i \in \{1, 2\}$ it holds that $d(x + z_i + f, y) \leq$

$d\left(C_Z^{\perp}/2\right)$ and therefore $d\left(x+z_1+f, x+z_2+f\right) \leq d\left(C_Z^{\perp}\right)$. But

$$\begin{aligned} & d\left(x+z_1+f, x+z_2+f\right)=\left|x+z_1+f+x+z_2+f\right| \\ =&\left|z_1+z_2\right|=d\left(z_1, z_2\right) \end{aligned}$$

contradiction for the assumption that $z_1, z_2 \in C_Z^{\perp}$. □

We are ready to show step by step the decoding process. Let $P=X^f Z^e$ be an error such that $e, f<d/2$ act on the state $|\mathbf{x}\rangle$. Denote by H_X, H_Z the parity check matrices of C_X, C_Z.Using the commute relation $[X^f, Z^e]=(-1)^{\langle e,f\rangle}$ we have that:

$$|\mathbf{x}\rangle \mapsto^P \frac{1}{\sqrt{|C_Z^{\perp}|}} \sum_{z \in C_Z^{\perp}} X^f Z^e |x+z\rangle=\frac{1}{\sqrt{|C_Z^{\perp}|}} \sum_{z \in C_Z^{\perp}}(-1)^{\langle e,f\rangle} Z^e |x+z+f\rangle$$

Now the decoder computes and stores the syndrome relative to the bits code using the parity check matrix H_X. And apply the inverse gate.

$$\begin{aligned} & \mapsto^{H_X} \frac{1}{\sqrt{|C_Z^{\perp}|}} \sum_{z \in C_Z^{\perp}}(-1)^{\langle e,f\rangle} Z^e |x+z+f\rangle \left|H_X(x+z+f)\right\rangle \\ = & \quad \frac{1}{\sqrt{|C_Z^{\perp}|}} \sum_{z \in C_Z^{\perp}}(-1)^{\langle e,f\rangle} Z^e |x+z+f\rangle \left|H_X f\right\rangle \\ & \mapsto^{X^f} \frac{1}{\sqrt{|C_Z^{\perp}|}} \sum_{z \in C_Z^{\perp}} Z^e |x+z\rangle \left|H_X f\right\rangle \end{aligned}$$

Then rotating into the phases base:

$$\begin{aligned} & \mapsto^{H^{\otimes n}} \frac{1}{\sqrt{|C_Z^{\perp}|}} \sum_{z \in C_Z^{\perp}} X^e H^{\otimes n} |x+z\rangle \\ & =\frac{1}{\sqrt{|C_Z^{\perp}|}} \sum_{z \in C_Z^{\perp}} X^e H^{\otimes n} X^x |z\rangle=\frac{1}{\sqrt{|C_Z^{\perp}|}} \sum_{z \in C_Z^{\perp}} X^e Z^x H^{\otimes n} |z\rangle \\ & =\frac{1}{\sqrt{|C_Z^{\perp}|}} X^e Z^x H^{\otimes n} \sum_{z \in C_Z^{\perp}} |z\rangle=\frac{1}{\sqrt{|C_Z|}} X^e Z^x \sum_{z \in C_Z} |z\rangle \\ & =\frac{1}{\sqrt{|C_Z|}}(-1)^{\langle x,e\rangle} Z^x \sum_{z \in C_Z} X^e |z\rangle=\frac{1}{\sqrt{|C_Z|}}(-1)^{\langle x,e\rangle} Z^x \sum_{z \in C_Z} |z+e\rangle \end{aligned}$$

So now we have back into the begging. Only now the phase flips are playing the role of bit flips relative to the code C_Z.

$$\begin{aligned} & =\frac{1}{\sqrt{|C_Z|}}(-1)^{\langle x,e\rangle} Z^x \sum_{z \in C_Z} |z+e\rangle \left|H_Z e\right\rangle \\ & \mapsto \frac{1}{\sqrt{|C_Z|}}(-1)^{\langle x,e\rangle} Z^x \sum_{z \in C_Z} |z\rangle \left|H_Z e\right\rangle \mapsto^{H^n} \frac{1}{\sqrt{|C_Z^{\perp}|}} X^x \sum_{z \in C_Z^{\perp}} |z\rangle \\ & =\frac{1}{\sqrt{|C_Z^{\perp}|}} \sum_{z \in C_Z^{\perp}} |z+x\rangle=|\mathbf{x}\rangle \end{aligned}$$

□

There is still one big difference between the classic repetition code and the Shor code. While each parity check of the Shor code examines a square root number of qubits, any check of the repetition code touches no more than a constant number of qubits; that is, any check just tests if any two adjacent bits are equal. That brings us to ask whether the Shors code is really the quantum analogy for the repetition code?

For getting an hint before formally presenting a quantum LDPC code, let's take another look on the general structure of the CSS codes. The decoding procedure the proof above teach us an additional point about CSS code, the task of finding a good code quantum code, could be reduce for finding a two classic binary linear codes which their parity check matrices ortogonal to each other. Furthermore, if one is willing to has an qLDPC code, then H_X and H_Z can't be parity check matrices of good classical code as any column of $H_z^\top$ is a codeword of C_X.

$$C_Z^\perp \subset C_X \Rightarrow H_X H_Z^\top = 0$$

And by being an LDPC code, the rows wights of H_Z is bounded by constant. Therefore there is a codeword $\in C_X$ which is also a row of H_Z that has a constant weight.

7 qLDPC Codes.

As exactly as in the classic case, qLDPC codes are codes in which any check act non trivially on at most a constant number of qubits, It was proved that using a good Quantum LDPC code one can achieve a fault tolerance threshold theorem at the cost of only constant overhead[1] [Got14]. We are now about to embark on a detailed review of the first quantum LDPC code [Den+02].

Recall that one way to present a code is by define the parity check matrix, Consider the $l \times l$ Tours, namely the Cayley graph of the group product $\mathbb{Z}_l \times \mathbb{Z}_l$. Associate any coordinate (bit/qubit) with an edge on the Tours. And consider the following two restrictions:

1. Each vertex requires form its local view, the bits lay on its supported edges, To has an even party. We will refer to this type of check as *cross check*.

2. Similarly, each face requires the same from its supported edges, but computes the parity in a different (specific) base. That it, the face first rotates the qubits by applying the Hadamard transform on them, and then computes their XOR. Finally, the qubits are rotated back to the computation base. We shall refer to this type of check as *face check*.

[1] under the assumption of having an efficient decoder.

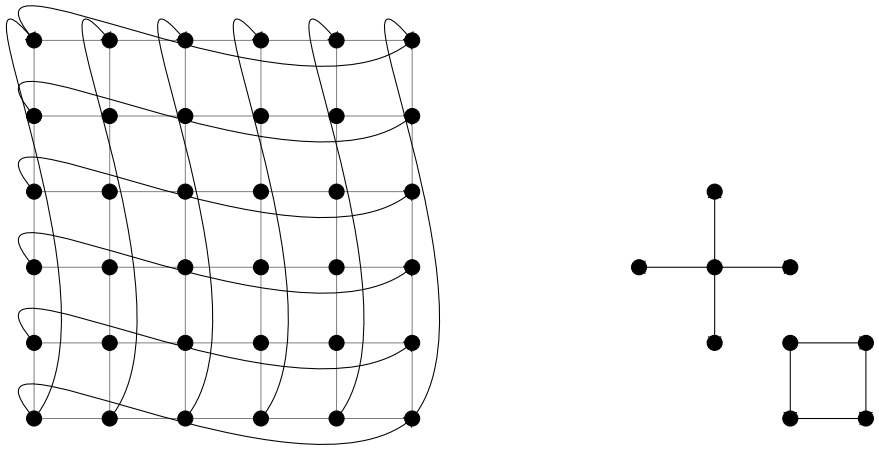

Figure 3: On the left is the Toric Graph. On the right are cross and face checks.

Claim 7.1. *The $l \times l$ Toric code is a CSS code, with dimension* 2 *and distance* $\Theta(l)$.

Proof. Consider a pair of cross and a face checks. If they are not intersect (share edges) then, obviously, they commute. So suppose they share an edge. Now there are a finite number of cases (4) in which a cross check can intersect a face, and for all of them we have they must to intersect in an exactly two edges. Therefore the crosses checks commute with all the faces checks.

Now, denote by C_X and C_Z the linear codes defined by the crosses and faces checks. Observes that C_X contain all the subgraph of the tours which have only evens degree, namely all the loops. The following claim will be used to show that all the low weight loops (squeezable loops) are in $C_Z^{\perp}$.

Claim 7.2. *Let c be an assignment of ones on a unite square, namely a closed loop at length* 4*, then $c \in C_Z^{\perp}$.*

Proof. Denote by c' a codeword of C_Z, therefore the parity sum induced by c' on any square equals zero, Particularly the induced parity on the square supporting c is zero. But that parity is exactly $c \cdot c'$. As it true for any $c' \in C_Z$ we obtain that $c \in C_Z^{\perp}$. □

Claim 7.3 (Veblen's theorem)**.** *The set of even subgraph is a linear space spanned by simple cycles (vertices degree equals exactly 2).*

Proof. Let $P \subset E$ be an even subgraph then it must to have a simple cycle denote it by P', now notice that P/P' is also an even subgraph. To see that consider a vertex v, As P' is simple cycle, substituting P' is either not effect the degree of v in P/P' or it decrees v degree by exactly 2 so $d_{P/P'}(v) = d_P(v) - 2$. In both cases $d_{P/P'}(v)$ is even. By repeating recursively until $P/P' = \{\emptyset\}$ we get a decomposition of P into a sum of simple cycles. □

Claim 7.4. *Associate with any vertex of the Torus a coordinate in $\mathbb{Z}_l \times \mathbb{Z}_l$. Consider a simple cycle P subset of the Torus. Denote P by the vertices composing it $v_0 v_1 .. v_k$ arranged in order. Consider a vertex $v_i \in P$ and denote $v = (x, y) \in \mathbb{Z}_l \times \mathbb{Z}_l$. Then there exists a vertex $u \in P$, $u \neq v_{i-1} v_{i+1}$ that shares one of the coordinates of v. Put differently, there exists z such that either $u = (z, y)$ or $u = (x, z)$.*

Proof. Assume without loss of generality that $v_{i-1} = (x, y-1)$. Now denote by f the projection on the second coordinate, $f(a, b) = b$. and observe that for any j the distance $f(v_{j+1}) - (v_j)$ satisfies:

$$f(v_{j+1}) - f(v_j) = \begin{cases} \pm(l-1) & v_j = (\cdot, l-1), v_{j+1} = (\cdot, 0) \\ \in \{\pm 1, 0\} & \text{else} \end{cases}$$

So if there is no $u \neq v_i$ such that $f(u) = f(v_i) = f(v_{i-1}) + 1$ then there must to be an edge $\{v_j, v_{j+1}\}$, $v_j = (\cdot, l-1)$, $v_{j+1} = (\cdot, 0)$ such P pass trough it. So

$$\begin{aligned} f(v_{i-1}) &\leq f(v_{i+1}) + |f(v_{i+2}) - f(v_{i+1})| + .. + -(l-1) \\ &\leq f(v_{i+1}) - |P| - (l-1) \end{aligned}$$

But $|P| \leq l$. □

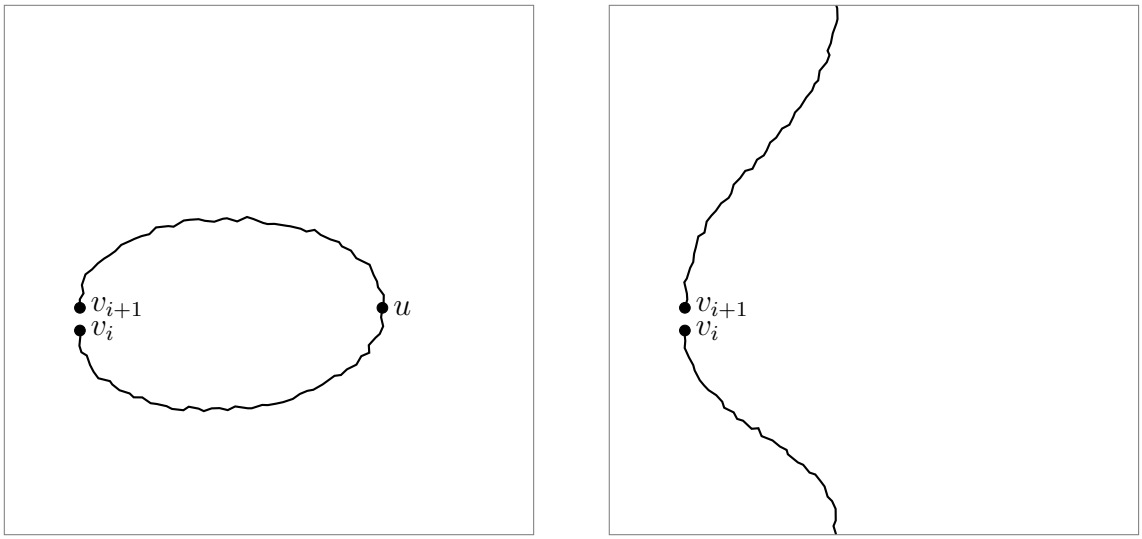

Figure 4: A simple cycle P that passes through a vertex u, with the same y-coordinate as v_i, on the left versus a cycle path for which there is no such u.

Definition 7.1. *Horizontal and Vertical diameters. Let P be defined again as exactly as in claim 7.4. We will say that the horizontal diameter of P is:*

$$\max_{v,u \in P} \min \{|v_x - u_x|, |v_x + l + u_x|\}$$

Similarly, we define the vertical diametr to be:

$$\max_{v,u \in P} \min \{|v_y - u_y|, |v_y + l + u_y|\}$$

Claim 7.5. *If P is a non empty simple cycle with vertical and horizontal diameters d_1, d_2, then it can either be a square or can be decomposed into two simple cycles P_1, P_2 with either the vertical diameter of both of them being strictly less than d_1 and their horizontal diameters being at most d_2, or the horizontal diameter of both of them being strictly less than d_2 and their vertical diameters being at most d_1.*

Proof. If P is not a square, it must have either a vertical or horizontal diameter greater than 1. We will assume, without loss of generality, that the vertical diameter is greater than 1. Let $v_0 v_1 .. v_k$ be the vertices that compose P, and let v_i, v_j be the vertices at maximal distance. We will pick the first vertex $v_{i'}$ from the left of v_i such that $v_{i',y} \neq v_{i,y}$. By claim 7.4, there must be at least one vertex v_l in P such that $v_{i,y} = v_{l,y}$ and is the closest in horizontal distance to v_i. We will denote by $P^{\uparrow}$ the path $v_i, v_{i+1} .. v_{l-1}, v_l$ and by $P^{\downarrow}$ the path $v_l, v_{l+1} .. v_{i-1}, v_i$. We will also denote by L the straight line $v_i, (v_{i,x}+1, v_{i,y}), (v_{i,x}+2, v_{i,y}), .. v_l$.

Observes that the vertical diameter will be lower by exactly 1 than the vertical diameter of P, and the horizontal distance between any point on L to other point on P will be lower than the horizontal distance of between the end of L (namely $\{v_i, v_l\}$), to the same point. So if $P^{\uparrow} \cup L$,$P^{\downarrow} \cup L$ are simple cycles then we get the desire. Suppose that $P^{\uparrow} \cup L$ is not a simple cycle. This means that the path $v_i, v_{i+1} .. v_{l-1}, v_l .. (v_{i,x}+2, v_{i,y}), (v_{i,x}+1, v_{i,y}) v_i$ must contain an inner loop. This loop cannot be supported on $P^{\uparrow}$ alone, because $P^{\uparrow} \subset P$ so the inner loop would have to be contained also in P, which is a contradiction. Therefore, the inner loop is also supported on L, so it follows that there is a vertex in L with degree > 2 in $P^{\uparrow} \cup L$, namely a vetex $u \neq v_i, v_l$ such that $u_y = v_{i,y}$, and also $u \in P$. But by the construction of L, $|u_x - v_{i,x}| < |v_{l,x} - v_{i,x}|$, which is a contradiction to the fact that v_l was chosen to minimize the horizontal distance. Thus, we have our claim. □

We have now established that $c \in C_X$ with $|c| < l$ can be decomposed into simple cycles, each of which has a weight less than l. Applying claim 7.5 recursively, we can further decompose each of these cycles into a sum of unit squares. However, claim 7.2 states that unit squares are in $C_Z^{\perp}$, so $c \in C^{\perp}$.

□

Note that the subtraction of any two l-length vertical cycles is a code word of C'_Z (one can be obtained by adding unit rectangles to the other). Therefore, we have only two non-trivial cosets in C_X / C'_Z. Hence, the dimension of the code is two.

References

[Ham50] R. W. Hamming. "Error detecting and error correcting codes". In: *The Bell System Technical Journal* 29.2 (1950), pp. 147–160. DOI: 10.1002/j.1538-7305.1950.tb00463.x.

[RS60] Irving S. Reed and Gustave Solomon. "Polynomial Codes Over Certain Finite Fields". In: *Journal of The Society for Industrial and Applied Mathematics* 8 (1960), pp. 300–304.

[Tan81] R. Tanner. "A recursive approach to low complexity codes". In: *IEEE Transactions on Information Theory* 27.5 (1981), pp. 533–547. DOI: 10.1109/TIT.1981.1056404.

[Sho95] Peter W. Shor. "Scheme for reducing decoherence in quantum computer memory". In: *Phys. Rev. A* 52 (4 Oct. 1995), R2493–R2496. DOI: 10.1103/PhysRevA.52.R2493. URL: https://link.aps.org/doi/10.1103/PhysRevA.52.R2493.

[CS96] A. R. Calderbank and Peter W. Shor. "Good quantum error-correcting codes exist". In: *Physical Review A* 54.2 (Aug. 1996), pp. 1098–1105. DOI: 10.1103/physreva.54.1098. URL: https://doi.org/10.1103%2Fphysreva.54.1098.

[Gro96] Lov K. Grover. *A fast quantum mechanical algorithm for database search*. 1996. arXiv: quant-ph/9605043 [quant-ph].

[Sho97] Peter W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1484–1509. DOI: 10.1137/s0097539795293172. URL: https://doi.org/10.1137%2Fs0097539795293172.

[AB99] Dorit Aharonov and Michael Ben-Or. *Fault-Tolerant Quantum Computation With Constant Error Rate*. 1999. arXiv: quant-ph/9906129 [quant-ph].

[AK99] Ashish Ahuja and Sanjiv Kapoor. *A Quantum Algorithm for finding the Maximum*. 1999. arXiv: quant-ph/9911082 [quant-ph].

[Den+02] Eric Dennis et al. "Topological quantum memory". In: *Journal of Mathematical Physics* 43.9 (Sept. 2002), pp. 4452–4505. DOI: 10.1063/1.1499754. URL: https://doi.org/10.1063%2F1.1499754.

[Got14] Daniel Gottesman. *Fault-Tolerant Quantum Computation with Constant Overhead*. 2014. arXiv: 1310.2984 [quant-ph].

[AF22] "Maximum distance separable (MDS) code". In: *The Error Correction Zoo*. Ed. by Victor V. Albert and Philippe Faist. 2022. URL: https://errorcorrectionzoo.org/c/mds.