CS294-242: Quantum Coding Theory Spring 2024

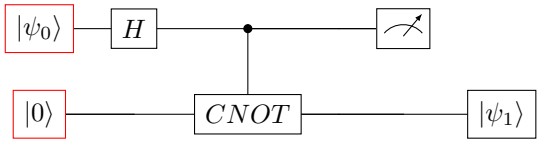
Lecture 25: Fault Tolerance

Instructor: John Wright Scribe: Theo Putterman

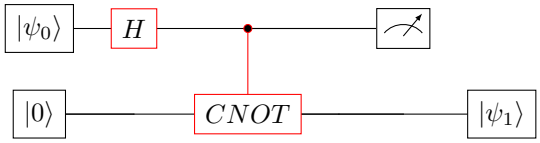
Suppose we have a circuit with a series of time steps. Each gate, state initialization, and measurement has a specific location. At each time step, we also define an empty location for qubits that have nothing done to them. Thus, at every time step every qubit has a location.

We have four types of locations

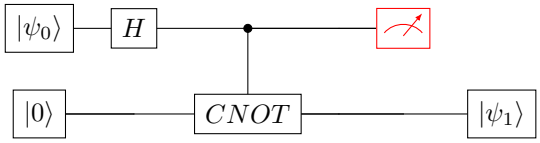
1. **State Preparation**



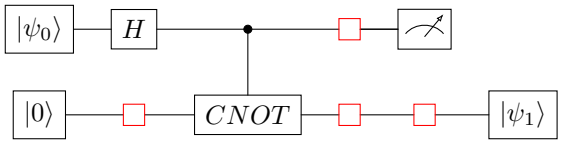
2. **Gates**



3. **Measurements**



4. **Wait**



Remark 25.1. A qubits life begins at preparation and ends at measurement.

Definition 25.2. *The **Independent Stochastic Noise Model** is a model in which every location L has a probability of error, p_L, and an error channel E_L.*

In this noise model, each error happens independent of all other locations. We say a location L is faulty with probability p_L, where P_L, P_I are independent for $I \neq L$. Thus, with probability p_L, E_L is applied to the qubit.

Remark 25.3. We would generally expect a wait location to be less error prone than a gate.

Which problems does this system not capture?

1. Consider a hadamard gate, where the device implements the hadamard gate approximately, with a small added rotation. This would not be captured by this model. Technically you could have an error probability close to 1 but we expect error probabilities to be ≈ 0.

2. Consider a fault at two locations, L_i, L_j. These error channels will be tensor multiplied, and thus independent, so we cannot handle correlated error channels.

3. Like error channels, we can't have correlated probability of errors.

Later we will consider a noise model that allows for correlated errors and error channels. Often, different p_L only depend on the type of location.

Our goal is to encode our circuit into a fault tolerant circuit.

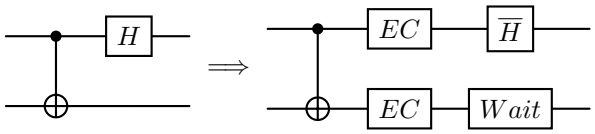
Remark 25.4. The natural thing is to encode each qubit in an error correcting code. In this construction, we take each qubit and separately encode it into an $[[n, 1, 2t + 1]]$ quantum error correcting code.

After encoding each qubit, we want to apply each of our operations on the encoded states. So, what we would like to do is to convert each location L into a gadget.

Definition 25.5. *A **gadget** is a small O(1) sized circuit that carries out an operation on a qubit and is made of physical locations*

So we have the following four points.

1. Circuit $\to$ fault tolerant circuit
2. Qubit $\to$ codeword
3. Location $\to$ gadget
4. Place an error correction between every consecutive pair of gadgets

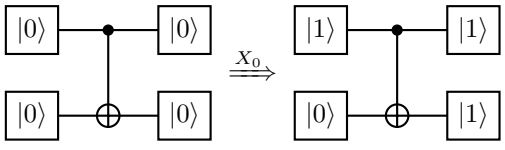


Each wire is an [[n, 1, 2t+1]] QECC, and each operation is replaced by a gadget that carries out that operation.

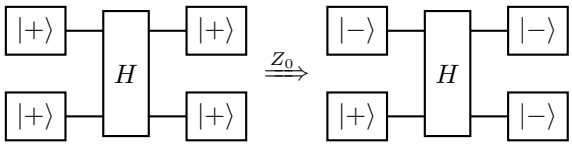
For example, CNOT becomes a gadget that carries out CNOT, and H becomes a gadget that carries out the hadamard. Between every pair of gadgets we place an error correcting step, and we keep wait locations the same.

Definition 25.6. *The biggest challenge that occurs with fault tolerant circuits is **Error Propogation***

Example 25.7. *Suppose we have a 2 qubit CNOT circuit. Both $|0\rangle$s should pass through it. If we have an error in this circuit, and the top $|0\rangle$ becomes a $|1\rangle$ (i.e. an X error). Now, we are controlling on a $|1\rangle$, so both qubits get flipped.*

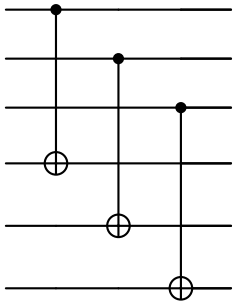


Example 25.8. *Suppose we have a $|+\rangle, |+\rangle$ into a hadamard. Now, the hadamard acts as the identity. Suppose we have a Z error on our bottom $|+\rangle$. Now, the output is $|+\rangle |-\rangle = |00\rangle - -|01\rangle + |10\rangle - |11\rangle \to |00\rangle - |01\rangle + |11\rangle - |10\rangle = |-\rangle |-\rangle$. Now, a Z error on the first qubit propogates to a Z error to both qubits.*



With error propogation like this, things get out of control and errors grow exponentially. Fault tolerant ircuits can fix this though.

Example 25.9. *Suppose we have three wires on top for an* $[[n, 1, d]]$ *qubit, and three wires below for an* $[[n, 1, d]]$ *qubit. CNOT from 1-4, 2-5, 3-6. Ifw e have an X error on qubit 1, then the X error gets propogated down to an x error on qubit 4. This can be corrected with a distance* 3 *code, since each X error is only distance* 1*.*



In general, as long as errors get propogated to different blocks, we should be able to correct our code. When this isn't satisfied, and our code has two errors on the same gadget.

Definition 25.10. *A transversal gate is a gate where we only have a gate between ith qubit of first block and ith qubit of second block.*

Transversal gates let us ensure that our errors don't propogate too much.

Remark 25.11. What we want to know is what logical actions we can perform on an encoded qubit in a transversal manner.

Fact 25.12. *If we encode our qubits with an* $[[n, k, d]]$ *stabilizer code, the logical paulis (e.g.* $X_1 Z_2 \dots Y_k$ *can be implemented transversally.*

Proof. For a stabilizer code, our logical operations are in $N(S)/S \approx Pauli_k$. So, for any k qubit pauli, there is an N qubit string of pauli matrices that will perform that logical pauli. □

In the $k = 1$ case, we can perform an X, Y, or Z operation by only performing local actions on the qubits in the codeword.

Example 25.13. *Consider the* $[[7, 1, 3]]$ *steene code. It has 6 stabilizers.*

1. $g_1 = I \otimes Z \otimes Z \otimes Z \otimes Z \otimes I \otimes I$
2. $g_2 = Z \otimes I \otimes Z \otimes Z \otimes I \otimes Z \otimes I$
3. $g_3 = Z \otimes Z \otimes I \otimes Z \otimes I \otimes I \otimes Z$
4. $g_4 = I \otimes X \otimes X \otimes X \otimes X \otimes I \otimes I$
5. $g_5 = X \otimes I \otimes X \otimes X \otimes I \otimes X \otimes I$
6. $g_6 = X \otimes X \otimes I \otimes X \otimes I \otimes I \otimes X$

Now, $\bar{Z} = Z \otimes Z \otimes Z \otimes Z \otimes Z \otimes Z \otimes Z$*, and* $\bar{X} = X \otimes X \otimes X \otimes X \otimes X \otimes X \otimes X$*. These two operations are clearly transversal, since every qubit is acted upon independently. So, one error in our code will not propogate*

Question 25.14. *What other gates can we apply in a transversal manner?*

Let $|\psi\rangle \in C$, where C is our codespace. Consider $U|\psi\rangle$. When is $U|\psi\rangle$ in the code? Consider $UM|\psi\rangle$, $M \in$ Stabilizer. Now, $UM|\psi\rangle = UMU^\dagger U|\psi\rangle$. So, we want $\{UMU^\dagger | M \in \text{Stabilizer}\}$. So, we want $UMU^\dagger$ to be in the original stabilizer.

Example 25.15. *Let $U = H^{\otimes 7}$. Now, $H : X \iff Z$.*

1. *$H^{\otimes 7} g_1 H^{\otimes 7} = g_4$, since the identities stay the same, and the Zs become Xs.*
2. *$H^{\otimes 7} g_2 H^{\otimes 7} = g_5$, by the same reasoning.*
3. *$H^{\otimes 7} g_3 H^{\otimes 7} = g_6$, by the same reasoning.*

and vice versa.

Next, $H^{\otimes 7} \bar{Z} H^{\otimes 7} = \bar{X}$, and $H^{\otimes 7} \bar{X} H^{\otimes 7} = \bar{Z}$. So, $H^{\otimes 7}$ swaps the role of $\bar{X}$ and $\bar{Z}$. Thus, $H^{\otimes 7}$ is a logical hadamard.

Example 25.16. *The classical S operator sends $Z \to Z$, and $X \to Y$. Let $U = S^{\otimes 7}$, $S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$. $S^{\otimes 7} g_1 S^{\otimes 7} = g_1$, since all the Zs are preserved. $S^{\otimes 7} g_4 S^{\otimes 7} = S^{\otimes 7} IXXXXII S^{\otimes 7} = IYYYYIII(i \cdot Z)(i \cdot Z)(i \cdot Z)(i \cdot Z)II = i^4 g_4 g_1$, since all the Zs are preserved. So, $S^{\otimes 7} \in$ Stabilizer. Now, $\bar{Z} \to^S \bar{Z}$, and $\bar{X} \to^S YYYYYYY = (iXZ)^{\otimes 7} = i^7 \bar{X}\bar{Z} = -i\bar{X}\bar{Z} = -\bar{Y}$. This is almost S but not quite. Rather, $S^{\otimes 7} = \bar{S}^\dagger$.*

Note that both H and $S^\dagger$ are in the clifford group, and they almost generate the entire clifford group. To generate the entire clifford group we also need a $CNOT$.

Question 25.17. *Can we transversally generate a $CNOT$ gate.*

Example 25.18. *Let $U = CNOT^{\otimes 7}$. Now, we can define the $CNOT$ gate by its actions on errors.*

1. *$XI \to XX$*
2. *$IX \to IX$*
3. *$ZI \to ZI$*
4. *$IZ \to ZZ$*

Consider two 7-qubit codewords. The generators of the overall codespace is $g_1 \otimes I^7, g_2 \otimes I^7 \dots g_6 \otimes I^7, I^7 \otimes g_1, I^7 \otimes g_2 \dots I^7 \otimes g_6$. Now, $U(g_1 \otimes I^7) = U(IZZZZII \otimes IIIIIII) = g_1 \otimes I^7$. On the other hand, $U(g_6 \otimes I^7) = XXIXIIX \otimes IIIIIII = XXIXIIX \otimes XXIXIIX = g_6 \otimes g_6$. $U(I^7 \otimes g_1) = g_1 \otimes g_1$. $U(I^7 \otimes g_6) = I^7 \otimes g_6$. So, every stabilizer gets mapped to another stabilizer by U.

Remark 25.19. This means U is performing a logical operation on the qubits, we just don't know which one.

To find out which logical operator U corresponds to, we want to know what it does to our four logical operators, $\bar{X} \otimes I \to \bar{X} \otimes \bar{X}, \bar{Z} \otimes I \to \bar{Z} \otimes I, I \otimes \bar{X} \to I \otimes \bar{X}, \bar{Z} \otimes \bar{Z}$. This is exactly our logical CNOT operator. So, $\overline{CNOT} = CNOT^{\otimes 7}$. So, the answer to our question is yes, and we can generate the entire clifford group with transversal operators.

Fact 25.20. *Unfortunately, we cannot implement the T gate, $\begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{\delta}} \end{bmatrix}$. So, we cannot implement a universal gate set.*