Part II: Fault-tolerance

Lecture I: Definition of fault tolerance

So far we have only considered error models where data qubits are affected by errors. But what if the error correction circuits themselves are also noisy?
This is the case in current (and most likely) future quantum hardware.

How do we run long computations when all parts of the circuits are noisy?

Classical hardware is so reliable that we don't need to worry about this issue.
Fault tolerant error correction

Like putting out a fire with a fire extinguisher that is also on fire!
Definition of fault tolerance

Def: circuit noise error model

Given a circuit, break it up into locations, where a location is a gate (1 qubit, 2 qubit, maybe 3 qubit), a measurement,
a state preparation (generally 102), or a storage/wait location. Assume that classical computations 'of modest size' are perfect and instantaneous.

For a location \( L \) assume that with prob. \( 1 - p \) the location functions as intended.
And with probability \( p \)
the location \( E \) is replaced by an unknown quantum channel \( E_e \). We usually assume that \( E_e \) maps qubits to qubits and that each error channel is independent. Commonly \( E_e \) just depends on the type of location.
We often assume a state prep $|0\rangle$. The state is passed through a gate, which is represented by a unitary operator $U$. The state is then subjected to a depolarizing channel with probability $p$, which is denoted by $E_{dep}^{(p)}$. The depolarizing channel is applied after the gate, and the final state is measured.
Sometimes we use different error probabilities for different types of location e.g. 2-qubit gates are usually more error-prone than 7-qubit gates.

This is by no means the most general error model! In a later lecture we will discuss extensions.
Fault-tolerance is a surprisingly slippery concept to define. The basic idea is that we encode the qubits of the circuit in a quantum error-correcting code and we replace each physical location with
a corresponding logical location. We want the logical locations to not spread errors ‘too much’. We also periodically do error correction to prevent the build-up of errors.
This usually looks something like

\[ \overline{u} \quad \therefore \]

physical circuit

\[ \overline{u} \quad \therefore \]

logical circuit

bar denotes logical location
Defn: FT QEC

Let $C$ be an [[n, k, d]] stabilizer code. Let $t = \lceil \frac{d-1}{2} \rceil$. An error correction protocol for $C$ is FT if:

1. For an input codeword with error of weight $s$, if $s_2$ faults occur during...
the protocol w/ \( s_1 + s_2 \leq t \) then perfectly decoding the output state gives \( |\Phi\rangle \).

(2) For \( s \leq t \) faults occurring during the protocol for an arbitrary input state the output state differs from a codeword by an error of weight \( \leq s \).
1. Ensures that correctable errors don't spread to uncorrectable errors during the course of the protocol.

To understand why 2) is necessary, let \( \frac{t}{n} < s < \frac{2t+1}{3} \) where \( n \in \mathbb{Z}^+ \), e consider a QEC protocol where \( r \) input errors and \( s \) errors during the protocol result
in an output with at most $r + s$ errors.

Now suppose we apply the protocol $j$ times

When $j > n$ the input state to EC will have $ns > t$ errors! Failure after linear number of steps.
But if (2) holds

Input $|\overline{\Phi}\rangle$

After EC output is $E_1 |\overline{\Phi}\rangle$

where $wt(E_1) \leq s$

After 2nd EC output is $E_2 |\overline{\Phi}\rangle$, but by (2) output is also $E'_2 |\overline{\emptyset}\rangle$ where $|\overline{\emptyset}\rangle$ is a codeword and $wt(E'_2) \leq s$

$E'_2 |\overline{\emptyset}\rangle = E_2 |\overline{\Phi}\rangle$
\[
(1\) \quad \overline{\phi} = E_2 + E_2 \uparrow \overline{\phi} \\
wt( E_2 + E_2 ) \leq 3s
\]
as \ wt(E_2) \leq 2s \quad \text{a} \quad wt(E_2') \leq s

By assumption \quad 3s < 2t+1

d code dist.

\[
\Rightarrow \quad (\overline{\psi}) = \overline{\phi}
\]

\[
wt( E_2 ) = wt( E_2' ) \leq s
\]

ie

\[
\rightarrow \quad \text{EC} \quad \downarrow s \quad \text{EC} \quad \downarrow s \quad \text{EC} \quad \downarrow s
\]
We can write similar defns for all location types e.g. for a logical gate if the input has $s_1$ errors and $s_2$ errors occur during the gate where $s_1 + s_2 \leq t$ then ideally decoding the output gives the same thing as ideally decoding the input.
after applying the gate with no errors.

**Upshot:** To construct a FT circuit we need to construct

1. FT error correction
2. FT state prep
3. FT measurement
4. FT gates

Lecture 3 + 4
Aside: the defn of fault-tolerance we just discussed is perhaps too stringent e.g. surface code error correction fails to satisfy this defn. However it is the right defn for proving threshold than w/ concatenated codes,
as we will see in Lecture 5.

Post script

For an 'operational' defn of fault tolerance see

arXiv.org/abs/1610.03507

For a discussion of the defn of fault tolerance see

https://youtu.be/FMXFNC1aF3k