Lecture 2:

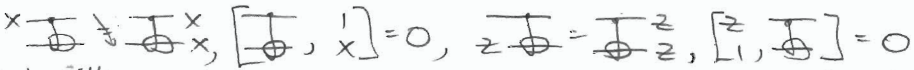
Stabilizer algebra on encoded states
Threshold against detected errors
"telecorrection"
Tolerable noise threshold heuristic argument
More fault-tolerance gadgets
Threshold proof

$T(k) = 2 + 2T(k-2)$

To generalize G_K, need

We need rules for
- applying Clifford
- measuring

To finish up from last time, I'm going to show how we can use stabilizer algebra to manipulate stabilizer codes and not just stabilizer states. Recall the four basic commutation rules:



To start with an example, the simplest fully quantum code is the 4-qubit error-detecting code, the stabilizer space of

```
XXXX
ZZZZ
```

With four qubits and two independent stabilizers, there remain two qubit degrees of freedom. What are they? So we need to find the encoded or logical operators acting within the codespace. These must commute with the stabilizers and be independent of them. In this case,

```
"(X1)_L"   X X 1 1
(Z1)_L     1 Z 1 Z        ⟶   encoded basis
(1X)_L     X 1 X 1             eg. (Z1)_L and (1Z)_L fix encoded |00⟩_L
(1Z)_L     1 1 Z Z
```

work. Arbitrarily, we may call them $(X1)_L, (Z1)_L, (1X)_L, (1Z)_L$. By fixing the names of these operators, we are choosing our basis for the code space. For example the encoded "$|00\rangle$" is the +1 eigenstate of $(Z1)_L$ and $(1Z)_L$.

What are the encoded Clifford gates?

1. encoded $H \otimes H$: apply H transversally, swap qbits 2 & 3
2. encoded CNOT:

Take two code blocks with the second encoded qbit in each fixed to $|0\rangle$. Altogether the code is

```
           XXXX                                    XXXXXXXX   ~XXXX....
           ZZZZ     I                              ZZZZ....
           11ZZ                                    11ZZ....                 code
                XXXX       transversal             ....XXXX                 space
       Is       ZZZZ       CNOTs                   ZZZZZZZZ     ....ZZZZ    fixed!
                11ZZ       1→5                     11ZZ11ZZ     ......ZZ
           --------        2→6                     --------
(X111)_L   XX11 1111       3→7                     XX11XX11     (X1X1)_L
(Z111)_L   1Z1Z 1111       4→8                     1Z1Z1111     (Z111)_L
(11X1)_L   1111 XX11                               ....XX11     (11X1)_L
(11Z1)_L   1111 1Z1Z                               .Z.Z.Z.Z     (Z1Z1)_L
```

We've applied an encoded CNOT gate! In other words, if $\mathcal{E}$ is an encoding circuit,

where / indicates a block or transversal operations thereon

In fact, transversal physical CNOT gates implement an encoded CNOT gate for any

CSS code: separate X & Z stabilizer generators

$$X(u) \equiv \bigotimes_{i=1}^{n} X^{u_i} \qquad \text{for } u \in C_1 \subseteq \{0,1\}^n$$

$$Z(v) \equiv \bigotimes_{i=1}^{n} Z^{v_i} \qquad v \in C_2$$

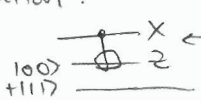
two codewords:

$$\begin{array}{ccc} X(u) & \otimes & 1^n \\ Z(v) & \otimes & 1^n \\ 1^n & \otimes & X(u) \\ 1^n & \otimes & Z(v) \\ \hline X_L & & 1 \\ Z_L & & 1 \\ 1 & & X_L \\ 1 & & Z_L \end{array} \quad \xrightarrow[i \to n+i]{\text{CNOTs}} \quad \begin{array}{ccc} X(u) & \otimes & X(u) \\ Z(v) & \otimes & 1^n \\ 1^n & \otimes & X(u) \\ Z(v) & \otimes & Z(v) \\ \hline X_L & & X_L \\ Z_L & & 1 \\ 1 & & X_L \\ Z_L & & Z_L \end{array}$$

Let me describe simulating measurement again simply by example. There are 3 cases:

measuring Q:
1. $Q \in S = \langle P_1, \dots, P_k \rangle$
   $\to$ outcome deterministic
2. $Q \notin S$ but commutes with $P_1, \dots, P_k$
   $\to$ logical measurement (depends on encoded state)
3. ★ Q anticommutes with P_1
   (ensure Q commutes with all σ_L, proceed as usual)

The third case is the only new one.
Consider teleportation:

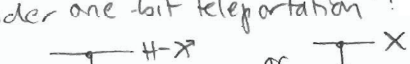


$\leftarrow$ operator measurement

initially: $\begin{array}{c} 1XX \\ 1ZZ \end{array} \xrightarrow{CNOT_{12}} \begin{array}{c} 1XX \\ ZZZ \end{array}$

$X_L: X11$
$Z_L: Z11$

Consider "one bit teleportation":



initial: $\dfrac{1Z}{\begin{array}{l} X_L\ X1 \\ Z_L\ Z1 \end{array}} \longrightarrow \dfrac{ZZ}{\begin{array}{l} XX \\ Z1 \sim 1Z \end{array}} \xrightarrow{mX_1} \dfrac{\pm X1}{\begin{array}{l} XX \sim \pm 1X \\ 1Z \end{array}}$

to correct X_L syndrome, apply $1Z$:

Alternatively, you can not apply the correction and just remember that in the $|-\rangle$ case, the basis is changed from $|0\rangle, |1\rangle$ to $|0\rangle, -|1\rangle$. In a physical qu computer with errors, we definitely won't apply the correction, because unnecessary gates mean unnecessary noise. In fact, we'll never apply any Pauli gate.

Up to corrections, the teleportation circuit is

How to teleport from a nontrivial stabilizer code?

Encoded into a CSS code with

$$X(C_1)$$
$$Z(C_2)$$
$$X_L = X^{\otimes n}$$
$$Z_L = Z^{\otimes n}$$

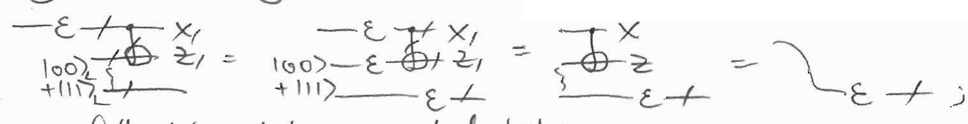
how do we implement the measurements?

transversally!

So to measure Z, measure each qubit destructively in the 0/1 basis. Each outcome will be random 50/50, but they will be correlated. The product of the measurements for every stabilizer will be 1 and the product of all the measurements will be the logical measurement result. Measurement satisfies

$$-\mathcal{E}\!\not{}\!-Z= \;=\; -Z=$$

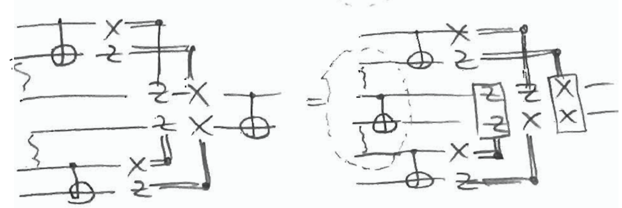
Putting the results together,



we've successfully teleported an encoded state.

Remark: any Clifford gate can be teleported into:

eg.



generalization.
cluster/graph states

Result: Threshold for detected errors!

detected/erasure/leakage
slightly broader

An error erases a qubit, and it tells us it has been erased; for example perhaps the atom carrying our qubit escaped from its trap and we can see that it is gone.

We can replace the atom, but the replacement qubit will be corr

Assume each gate fails indep. with prob. p.

1. Fix n-qubit CSS code X_L, Z_L transversal
2. Prepare perfect encoded ancilla for next gate
   → outputs will lie in code space (may have encoded errors)

   many ancillas in parallel → "factory"
3. Encoded teleport, by transversal $\oplus^{X}_{Z}$
4. Correct any possible errors using classical computer to determine X_L, Z_L meas. outcomes

If you get X_L or Z_L wrong, the correction will be wrong, giving a logical error on the output.

Claim: $\exists$ (random) CSS codes which can correct prob p detected errors for any $p < \frac{1}{2}$, except with error probability $e^{-\Omega(n)}$.

Idea: 2^{n-1} syndromes, $\binom{n}{pn} 4^{pn}$ typical errors to distinguish

$\approx 2^{(H(p)+2p)n}$

$p = 17\%$

classically, would be 2

$\Rightarrow$ Can compute to arbitrary reliability provided

$$\mathbb{P}(\text{detected } \overset{X}{\underset{Z}{\Phi}} \text{ error}) < \frac{1}{2}$$

Other error rates don't matter!

Caveat: Overhead

to achieve error ε, set $n = O(\log \frac{1}{\varepsilon})$

preparation overhead is $n^2 \quad e^{n^2} \sim e^{\log^2 \frac{1}{\varepsilon}} = \left(\frac{1}{\varepsilon}\right)^{\log \frac{1}{\varepsilon}}$

n^2: # gates/ancilla; e^{n^2}: # ancilla preps

superpolynomial in $\frac{1}{\varepsilon}$ ... this can be fixed easily..

44

---

10 minute break

(also non-Clifford gates...)

Main ideas: using teleportation for EC and computation combined (wouldn't work classically)

detected errors are easy, random CSS codes

Lecture 2, part 2: threshold for undetected errors

Problem: can't prepare large, perfect ancilla states except in detected errors model

can't even prepare large ancillas with nearly independent errors

-error during encoding will spread throughout the ancilla

need to bootstrap into larger and larger codes

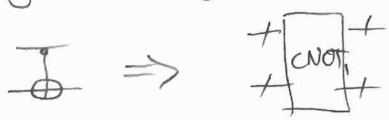
usually done by concatenating single constant-sized code.

(classically can add one bit at a time to the repetition code) $\to$, more difficult usually

Heuristic argument

The standard hand-waving argument goes as follows: Pick a constant-sized code, say Steane's 7-qubit, distance-3 code.

Compile each gate of the logical circuit we are trying to protect into an equivalent gadget.



Argue that this diagram commutes:

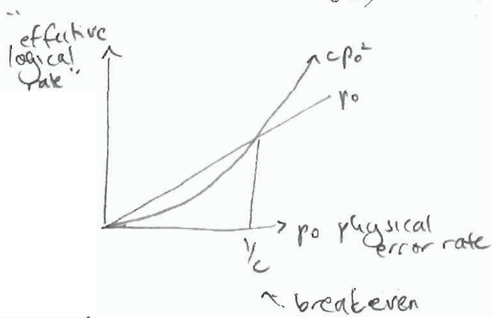
if there is at most one error inside the gadget. Declare failure otherwise.

$$\mathbb{P}[\text{"logical failure"}] \le c p_0^2$$

$\leftarrow p^{t+1}$ if code corrects t errors

$c \sim \binom{m}{2}$ if m possible failure locations

Plot this:



Intuitively, rinse and repeat.

k levels of concat.	error rate	overhead
0	p_0	1
1	cp_0^2	m
2	$c(cp_0^2)^2$	m^2
$\vdots$	$\vdots$	$\vdots$
k	$\frac{1}{c}(cp_0)^{2^k}$	m^k

Error rate drops rapidly provided $p_0 < \frac{1}{c}$ initially. Also, overhead is efficient

Set $k = \log\log\frac{1}{\varepsilon}$ $\quad m^k = \text{polylog}\frac{1}{\varepsilon}$

However, this won't give a threshold <u>proof</u>

To understand encoded CNOT gadget, need more parameters

Even if you could justify dropping parameters, effective error model will have dif. form

eg.

$-G_1-G_2-$

commuting decoder past G_2 will depend on input, which will depend on whether G_1 failed

failures become positively correlated btwn aboring encoded gates

Markovian $\longrightarrow$ non-Markovian noise

People run simulations to estimate thresholds, but these aren't proofs!

Fault-tolerant gadgets: want $t+1$ failures necessary to cause encoded failure

-want to minimize spread of errors

→ transversality is nice

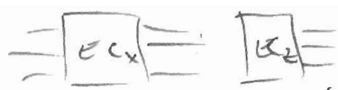
how to catch and correct errors?

-telecorrection

-Shor-type

-Steane-type

CSS code



separately

EC_X: can't measure codeword w/o destroying it

measure

has to "fail gracefully"

controlled failures

allow for concatenation

properties: any input

→ close to codespace

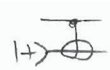
$p, p^2 \cdots p^t$

input w/ k errors

→ output has no logical error except w/ prob. p^{t+1-k}

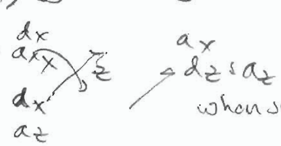
more rarely, leakage

1. perfect



no logical effect

2. errors



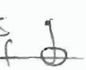
when successful

→ FT reduces to preparing <u>reliable</u> ancilla states

high fidelity not enough, want wt k errors to be $O(p_0^k)$ prob

Recursive verification:

CSS itself



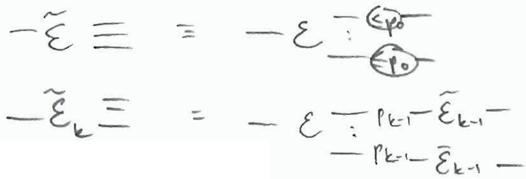
no effect! but copies X errors down

initially $\mathbb{P}[X \text{ error of wt } k] = O(p) \; \forall k$

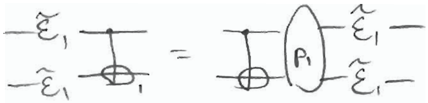
$p \;\; p \;\; p \;\; p \qquad p \;\; p^2 \;\; p^3 \;\; p^3$

$p \;\; p^2 \;\; p^2 \;\; p^2 \qquad p \;\; p^2 \;\; p^3 \;\; p^4$

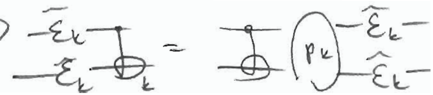
Threshold proof:



(*) argue



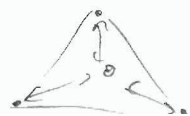
$\Rightarrow$



∴ remains to prove (*)

(*) is false. However it is close to true

$$\text{LHS} = \text{RHS} + \delta, \quad \delta = O(p^d)$$



$\Rightarrow$ constant threshold for discrete Pauli noise

10^{-3} highest estimates from simulation 1% - 6%