Algorithmic Fault Tolerance

Lecture 17
1 June 2026
Ph/CS 219

We've discussed Pauli-based computation (PBC): universal quantum computation via $|T\rangle$ prep and Pauli-product measurement (PPM). And we've discussed how to do PPM via surgery on either many surface code blocks or blocks of high-rate codes (the former with 2D connectivity, the latter with nonlocal connectivity). In surgery, we introduce ancillas and deform the code. For fault-tolerance, we consider qLDPC codes with distance d, and formulate the deformed code so that it, too, is qLDPC with $O(d)$ distance, and in addition the target PP is in the stabilizer of the deformed code. Therefore we can perform PPM by measuring the low-weight checks of the deformed code to determine the PPM as the parity of many check measurement outcomes.

We want this procedure to be FT, meaning that $\Omega(d)$ faults (qubit errors and measurement errors combined) are necessary for the PPM to fail. To gain sufficient confidence in the check measurement outcomes, we repeat the measurement of deformed code $O(d)$ times. Then prob of error in PPM is $\exp(-O(d))$.

Now we'll discuss a way to avoid this $O(d)$ blowup in the cycle time for FTQC. For this purpose, we'll make a distinction between the online part of the computation and the $|T\rangle$ factory. Preparing $|T\rangle$'s with $\exp(-O(d))$ probability of a logical error, we will need $O(d)$ syndrome extraction (SE) rounds. What we'll aim to speedup is the Clifford part of the computation, assuming that clean input $|T\rangle$ states are available. So our goal is really a space/time tradeoff. The Clifford computation is fast ($O(1)$ cycle time) and the $|T\rangle$ factory is slow ($O(d)$ preparation time). This means we'll need to pipeline the $|T\rangle$ prep, so there are always $|T\rangle$'s available when needed; that will increase the processor's spatial footprint. We'll describe how it works for the surface code, though similar ideas might apply to other CSS qLDPC codes that allow transversal Clifford group gates. In the surface code, $|T\rangle$ prep has spacetime cost $O(d^3) = O(d^2)$ qubits $\times$ $O(d)$ time. Conventionally we would say that a transversal CNOT gate also has spacetime cost $O(d^3)$, but we wish to reduce that to $O(d^2) = O(d^2)$ qubits $\times$ $O(1)$ time. This means we'll need fresh $|T\rangle$'s more often and must therefore enlarge the $|T\rangle$ factory.

For this to work, we'll need a paradigm shift concerning FTQC. We often formulate the goal of QEC to be protection of a quantum state, so that any decoded (Pauli) measurements sample from the same distribution of outcomes as if we had measured the ideal state instead. But that is more than we need to run one particular computation. In that case, all we care about is sampling from the ideal distribution for that one computation, which might work even if the protected state is far from ideal in other respects. This can make FTQC easier to achieve. This perspective is called "Algorithmic Fault Tolerance" (protecting a computation rather than a state). See:

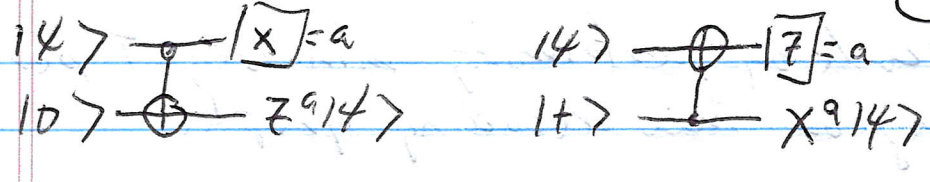
Zhou et al. 2024 arXiv: 2406.17653
Cain et al. 2025 arXiv: 2505.13587

In the surface code with 2D connectivity we are stuck with the old) slowdown. In lattice surgery in particular only O(1) measurement errors can cause a logical error if we measure the syndrome for only O(1) rounds. But we'll allow non-local connectivity as in neutral atom tweezer arrays, in which case gates that generate the Clifford group can be executed in depth 1.

How to perform logical H and S gates will be considered in a homework problem. What is especially valuable is the transversal CNOT gate (which works for any $k=1$ CSS code): $X_c \to X_c X_t$ and $Z_t \to Z_c Z_t$, performed at the physical level on pairs drawn from two code blocks, realizes $\bar{X}_c \to \bar{X}_c \bar{X}_t$ and $\bar{Z}_t \to \bar{Z}_c \bar{Z}_t$ at the logical level. We can bring the two blocks into "contact," perform the transversal operation, and then separate them again.

Strictly speaking, we won't be in the PBC model here. The goal will not be to measure a high-weight logical PP in a window with $O(1)$ temporal duration. We'll be doing Clifford + $|T\rangle$ prep. Though they are not really needed in the ideal computation, it will be helpful to introduce $|0\rangle$ and $|+\rangle$ prep as well as $|T\rangle$ prep. But the $|0\rangle$ and $|+\rangle$ prep (as well as X and Z measurement) will be part of our Clifford circuit — we do the $|0\rangle$ and $|+\rangle$ prep ourselves rather than taking it as given.

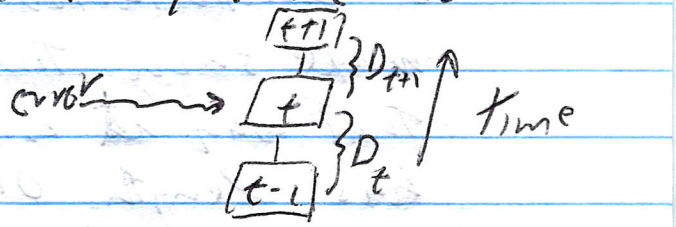
We might use 1-bit teleportation (1BT) to move logical information to a fresh block. Recall:



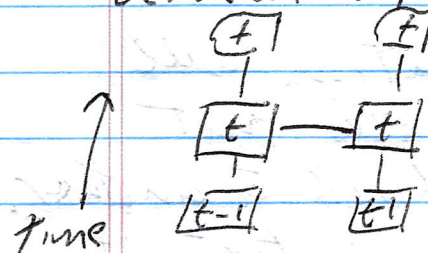
The measurement provides no info about $|+\rangle$. It is 50/50 random because measured observable anti-commutes with stabilizer (XI w.r.t. ZZ, ZI w.r.t. XX).

Recall that to protect a state in quantum memory, our decoder takes as input $O(d)$ rounds of SE. Why is that important?

There is a decoding graph in spacetime. Vertices are <u>detectors</u>. A detector fires if a particular syndrome bit changes from one time step to the next. The edges are errors (i.e. faults) which can be either qubit errors or measurement errors.

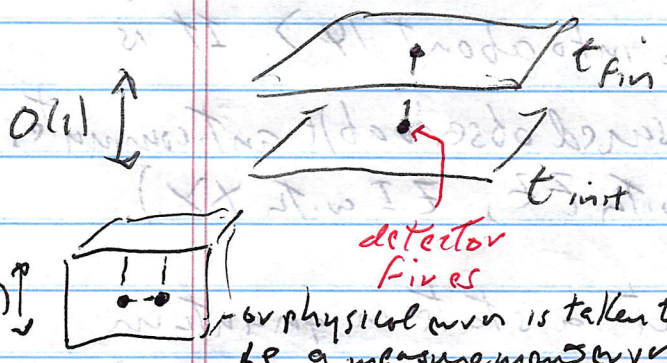


A measurement error in SE round t causes two detectors to fire. The syndrome bit flips twice, between rounds $t-1$ and t, then again between syndrome bits t and $t+1$.



A qubit error prior to SE round t also causes 2 detectors to fire — because it flips 2 syndrome bits in that one round.

We decode by performing min. weight perfect matching on this decoding graph.



A chain of $O(1)$ measurement errors can result in a isolated detector at an intermediate time, if we measure only $O(1)$ times. This detector gets paired with boundary of block or another detector far away, which can cause a logical error. To avoid this, $O(d)$ SE rounds to ensure that $O(d)$ faults (meas and qubit errors together) are needed for a logical error.

One way to say what we are trying to achieve is this: By focusing on protecting a computation instead of a state, we'll make such timelike chains of errors harmless. That way it will require a chain that has length $O(d)$ in the spacelike direction to cause failure. So the "fault distance" can be $O(d)$ even if ~~$O(d)$~~ vertical chains stretch across the temporal extent of our decoding window.

Our Clifford computation will include transversal CNOT gates that propagate errors from one block to another. This complicates decoding. In fact, if we perform $O(1)$ SE rounds between CNOT

layers, a single fault could spread to $2^{O(d)}$ blocks if we waited for $O(d)$ rounds before decoding, which seems unmanageable. For this reason, the conventional view has been to allow $O(d)$ SE rounds between CNOT layers, so we collect reliable syndrome data before allowing the damage to spread.

Another way to say why CNOT gates are problematic is that, if we allow blocks to interact, the decoding graph becomes a hypergraph; now a single fault can cause 3 or 4 detectors to fire. (Edges are pairs of vertices; hyperedges are sets of vertices that can contain more than two elements.) Matching on a hypergraph (finding an adequate hypothesis for the faults that caused detectors to fire) is a much harder computational problem than matching on a graph.

For example, between SE rounds $t-1$ and t an X error occurs which is then propagated by a CNOT from control to target block. In round t, 4 detectors are triggers in Z-type check operators, two in each block $\longrightarrow$ a 4-site hyperedge.

A more subtle example involves a measurement error, which causes detections in two consecutive time steps. An X measurement error in step t causes detectors to fire twice in the control block:

$$z_c^{(t-1)} \oplus z_c^{(t)} \quad \text{and} \quad z_c^{(t)} \oplus z_c^{(t+1)}$$

are both nontrivial. In addition the CNOT between steps t and $t+1$ modifies the detector on the target block, because $z_T^{(t)} \to z_c^{(t)} z_T^{(t)}$ and hence the detector becomes

$$z_c^{(t)} z_T^{(t)} z_T^{(t+1)},$$

which also fires because of the $z_c^{(t)}$ errors. This is a 3-site hyperedge.

We'll want to argue that, for the purpose of decoding a computation, the decoding hypergraph reduces to a graph, amenable to efficient MWPM decoding.

Now imagine decoding a particular Clifford + T computation, with input $|T\rangle$ (given), $|+\rangle|0\rangle$ (prepared by us) transversal Clifford gates, and X, Z measurements. We want our measurements to sample accurately from the distributions of outcomes in the ideal computation

In fact, though, there are some measured Pauli operators we don't need to simulate. Here is an example:

$|0\rangle$ —[X]=

The outcome is 50/50 random, we might as well flip a coin.

This happens because the "backpropagated" measured observable X anticommutes with the stabilizer Z of the initial state. (If, say, we are doing IBT, we might care about the outcome for updating the Pauli frame. But that matters only if that Pauli frame update is needed to interpret a later measurement.)

Another example: $|0\rangle$ —[Z]=

Here the outcome is deterministic, so we don't want to get it wrong. But suppose the measurement is destructive: all qubits measured in Z basis. Then those measurements are decoded to determine $\bar{Z}$. In that case $|0\rangle$ does not need to be accurately encoded because Z errors don't matter. If we do the prep starting with $|0\rangle^{\otimes w}$ and then do $O(1)$ rounds of SE, we'll have lots of uncertainty about outcomes of $X^{\otimes 4}$ syndrome measurements, but that is okay because Z errors won't change how $\bar{Z}$ is decoded. The blocks are not (might not even be entangled inside the block)

but the many $X^{\otimes 4}$ detections don't cause trouble.

Now consider, for example, X measurement on some logical block in the Clifford circuit, and backpropagate through the circuit until obtaining a Pauli operator acting on all preparations. If we obtain X acting on any $|0\rangle$ or Z acting on any $|+\rangle$, then the backprop Pauli anticommutes with stabilizer of initial state, which means outcome is 50/50 random. So we just assign a random bit to the outcome. No decoding is needed.

On the other hand, the backprop Pauli might have only Z or I acting on $|0\rangle$ or X or I acting on $|+\rangle$. In that case, though, we only need the $|0\rangle$ to give a reliable outcome for Z measurement. And for that purpose it is fine to prepare the state with only $O(1)$ rounds of syndrome extraction.

Now, if the outcome of the measurement of a particular block is 50/50 random, that does not mean we don't have to measure that block — even if its marginal distribution is uniform, it might have correlations with other blocks that we need to get right. So for each higher weight Pauli that is measured, we should consider whether or not

when backpropagated it commutes or anticommutes with stabilizer of $|0\rangle$ and $|+\rangle$ (that is, whether we find X acting on $|0\rangle$ or Z acting on $|+\rangle$). If it anticommutes, then no need to decode, just choose random outcome. If it commutes, then states prepped in Q11 rounds are okay. (The backprop may have X or Y on Z acting on $|T\rangle$, but that is okay because $|T\rangle$ has been prepared with reliable stabilizers.)

So our task is to decode the Pauli measurements we care about (called "reliable" in Cain 2025).

We notice that for this purpose, the decoding hypergraph becomes a graph to which MWPM can be applied. To build the decoding graph for a particular PP, we choose a standard logical representative for the Pauli product (e.g. a canonical choice of string operator for each $\bar{X}$ or $\bar{Z}$ in a surface code block, and then backpropate that operator through the Clifford circuit to find a Pauli $P(t)$ at each earlier time step. Now we consider the faults that can flip $P(t)$ (errors that anticommute with it) and include only those fault locations and the corresponding detectors triggered by those faults.

For example consider the X error described earlier that was propagated by a CNOT from control to target that fired 4 Z-check detectors at time t. An X error might hit a Z logical string operator flipping its value. If acting on these two blocks $P(t)$ is $Z_C \otimes I_T$ and the error flips the string, only the error on the control matters — hence two detectors. Same if $P(t) \sim I_C \otimes Z_T$. If the back prop $P(t)$ contains $Z_C \times Z_T$, then the error commutes with $P(t)$, and there are no detectors at all. In all three cases the detector becomes either an edge (2-vertices) or nothing.

Now consider the measurement error, which generated a 3-vertex hyperedge. Again we enumerate cases: ~~If $P(t) = Z_C \otimes I$~~, ~~there are two detectors~~

- $P(t+1) = Z_C \otimes Z_T \xrightarrow{\text{backprop}} P(t) = I_C \otimes Z_T$

  ~~[illegible]~~

  ~~[illegible]~~ No $(t-1, t)$ hyperedge because control block not in $P(t)$

  No $(t, t+1)$ hyperedge because XOR of two detectors is 0.

- $P(t+1) = I_C \otimes Z_T \xrightarrow{\text{backprop}} P(t) = Z_C \otimes Z_T$

  No $(t, t+1)$ detector on control block — it is not in $P(t)$ → 2 detectors → edge

- $P(t+1) = Z_C \otimes I_T \xrightarrow{\text{backprop}} P(t) = Z_C \otimes I_T$

  The target block is not in $P(t)$ or $P(t+1)$ → 2 detectors → edge

It always works, because P.P and errors propagate similarly.