Techniques for fault-tolerant quantum error correction

Ben Reichardt
UC Berkeley

Quantum fault-tolerance problem

- Fault-tolerant, larger $C$

- High tolerable noise
- Low overhead

$C$
Encoding for fault tolerance

- **Idea**: Encode ideal/logical circuit into quantum error-correcting code. Apply gates directly on the encoded data, each gate followed by error correction.

  - m-qubit, t-error correcting code
  
  $$[[m, 1, d=2t+1]]$$

  $$C_1 \leq \left( \frac{M}{t+1} \right) C_0^{t+1}$$

  level-1 CNOT failure rate

  physical failure rate

  Logical gate error rate

  \[1/c \approx \frac{1}{t} \]

  Concatenated encoding for arbitrary accuracy

  - **Idea**: Encode ideal/logical circuit into quantum error-correcting code. Apply gates directly on the encoded data, each gate followed by error correction.

    - m-qubit, t-error correcting code

    $$[[m, 1, d=2t+1]]$$

    $$C_k \leq \left( \frac{M}{t+1} \right) (C_{k-1})^{t+1}$$

    level-k CNOT failure rate

    level-(k-1) failure rate

  Logical gate error rate

  \[1/c^{t+1} \approx \frac{1}{t+1} \]
Threshold theorems

For a physical error rate $\varepsilon < \varepsilon_c$, an $N$-gate ideal quantum circuit can be reliably simulated with $N \poly(\log N)$ physical gates.

Examples:
- Independent probabilistic noise
  - $\varepsilon_c > 0$ [Aharonov & Ben-Or ‘97, Kitaev ‘97]
  - $\varepsilon_c > 2.7 \times 10^{-5}$ [Aliferis, Gottesman, Preskill ‘05]
  - $\varepsilon_c > 6 \times 10^{-6}$ with Pauli errors [R ‘05]
  - $\varepsilon_c \approx 10^{-4}$ (today)
  - $\varepsilon_c = 1/2$ for Bell measurement erasure errors (detected errors) [Knill ‘03]

Fault-tolerance threshold myths:
- Independent probabilistic noise.
- Nonlocal gates.
- Maximize the threshold regardless of the overhead.

Threshold theorems

For a physical error rate $\varepsilon < \varepsilon_c$, an $N$-gate ideal quantum circuit can be reliably simulated with $N \poly(\log N)$ physical gates.

Examples:
- Independent probabilistic noise
  - $\varepsilon_c > 0$ [Aharonov & Ben-Or ‘97, Kitaev ‘97]
  - $\varepsilon_c > 2.7 \times 10^{-5}$ [Aliferis, Gottesman, Preskill ‘05]
  - $\varepsilon_c > 6 \times 10^{-6}$ with Pauli errors [R ‘05]
  - $\varepsilon_c \approx 10^{-4}$ (today)
  - $\varepsilon_c = \frac{1}{2}$ for Bell measurement erasure errors (detected errors) [Knill ‘03]
- Non-Markovian local noise [Terhal/Burkard ‘04, Aliferis/Gottesman/Preskill ‘05]
- Correlated noise [Knill/Laflamme/Zurek ‘97]
- Local interactions
  - 2D grid (nearest neighbor), 1D line (next-nearest) [Gottesman ‘99]
  - with correlated noise [Aharonov, Kitaev, Preskill ‘05]
Outline

- **Idea** for improved ancilla verification for error correction: Differently prepare ancillas to verify against each other
  - Makes postselection unnecessary with 7-qubit Steane code [Aliferis]
  - Halves preparation complexity for 23-qubit Golay code ($1200 \rightarrow 600$ CNOT gates). Allows detailed combinatorial analysis to show high provable threshold ($10^{-4}$)

Outline

- Technical background
  - Error correction
  - Quantum ECCs
  - Stabilizer algebra

- Ancilla preparation and verification
  - Steane preparation and heuristic verification
    - for Steane 7-qubit, distance-3 code
    - for Bacon/Shor 9-qubit, distance-3 code
  - Strictly fault-tolerant verification
    - repeated purification
    - tweaked

- Rigorous noise threshold for 23-qubit, distance-7 Golay code
  - Technical setup
  - Combinatorial analysis
Steane-type error correction

Fact 1:
\[ |\psi\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \]
\[ |\psi\rangle \propto |+\rangle \]

Fact 2:
\[ a \otimes b \]

Def: CNOT

Steane-type error correction

Physical operations

Logical operations

Steane-type error correction
**Steane-type error correction**

Physical operations:
- Data: $|\psi\rangle_L$ and $|\psi\rangle_L$
- Ancilla: $|+\rangle_L$

Logical operations:
- $|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$
- $|\psi\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$

**Knill-type error correction**

Physical operations:
- Data: $|\psi\rangle_L$ and $|\psi\rangle_L$
- Ancilla: $|00\rangle_L + |11\rangle_L$

Logical operations:
- Apply correction
- $P_L|\psi\rangle_L$

**Teleportation**

- $|\psi\rangle$
- $|00\rangle$
- $|11\rangle$

**Steane-type error correction**

Physical operations:
- Data: $|\psi\rangle_L$ and $|\psi\rangle_L$
- Ancilla: $|+\rangle_L$

Logical operations:
- $|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$
- $|\psi\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$

**Knill-type correction + computation**

Physical operations:
- Data: $|\psi\rangle_L$ and $|\psi\rangle_L$
- Ancilla: $|00\rangle_L + |11\rangle_L$

Logical operations:
- Apply Knill-type correction
- Apply computation
- $U_L|\psi\rangle_L$

**Teleportation**

- $|\psi\rangle$
- $|00\rangle$
- $|11\rangle$
Steane-type error correction

Knill-type correction + computation

Error correction properties

- Arbitrary state is brought back into codespace (except with controlled errors: weight-k errors with probability $O(p^k)$).
- On states with controlled errors, no logical effect is applied (and errors remain controlled).

Teleportation

$$|\psi\rangle$$

$$|00\rangle_U + |11\rangle_U$$

$$U_L|\psi\rangle_U$$

Logical operations

Physical operations

Data ancilla

$mZ$

$mX$

$mX$
Remarks

- Computation can “typically” continue without waiting for error-correction measurements to complete (when correction information becomes available, propagate corrections through the circuit)
- High-fidelity ancillas do not suffice (need both high fidelity and uncorrelated errs)

⇒ Ancilla verification
  - Ancillas can’t be used until verified, so computation has to wait for verification measurements to complete

⇒ Ancilla factories
  - Prepare many ancillas in parallel and in advance, so a verified ancilla is always ready

⇒ High overhead

Quantum error-correcting codes

- [[n=4,k=2,d=2]] erasure code
  - used in Knill’s fault-tolerance scheme together with certain [[6,2,2]] code

- [[5,1,3]] code
  - not CSS — stabilizer includes, e.g., XZZX

- Steane [[7,1,3]] code

- Bacon/Shor [[9,1,3]] operator ECC

- [[15,1,3]] Reed-Muller code
  - allows for transverse (X+Z)/√2 application (for universality), but not self-dual

- Golay [[23,1,7]] code

CSS code: All stabilizers can be written as product of Xs or a product of Zs
CSS quantum stabilizer codes

- Classical codewords in the 0/1 basis
  ⇒ Correct bit flip $X$ errors

- Classical codewords in the $+i$- basis
  ⇒ Correct phase flip $Z$ errors

- E.g., Steane $[[7,1,3]]$ code corrects arbitrary error on one qubit
  Based on classical Hamming $[7,4,3]$ code

$$
C^\perp = \begin{pmatrix}
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1
\end{pmatrix}
\quad
C = \begin{pmatrix}
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1
\end{pmatrix}
$$

$$
X_L = X^{\otimes 7} \\
Z_L = Z^{\otimes 7}
$$

Steane $[[7,1,3]]$ quantum code

- Corrects arbitrary error on one qubit
  Based on classical Hamming $[7,4,3]$ code

- Simultaneous $+1$ eigenspace of 6 independent Pauli “stabilizer” elements

$$
\mathcal{H} = A \oplus B
$$

$$
S = \left\{ 0^7, 0001111, 0110011, 0111000, 1010101, 1011010, 1100110, 1101001 \right\}
$$

$$
|0_L\rangle = \frac{1}{\sqrt{8}} \sum_{x \in S} |x\rangle \\
|1_L\rangle = X^{\otimes 7} |0_L\rangle
$$

$$
H_L = H^{\otimes 7} \\
CNOT_L = CNOT^{\otimes 7}
$$
Stabilizer algebra

Def: S stabilizes $|\psi\rangle$ if $S|\psi\rangle = |\psi\rangle$

Rules:
- S, T stabilize $|\psi\rangle \Rightarrow ST$ stabilizes $|\psi\rangle$
- S stabilizes $|\psi\rangle \Rightarrow USU^\dagger$ stabilizes $U|\psi\rangle$

Def: Pauli group = tensor products of Pauli operators I, X, Y or Z (with phase $\pm 1$ or $\pm i$)
- note all Paulis have half eigenvalues +1, half -1; pairs of Paulis either commute or anticommute

Def: Stabilizer state on n qubits = intersection of +1 eigenspaces of n independent commuting Paulis

Example:

<table>
<thead>
<tr>
<th>Operation</th>
<th>State</th>
<th>Stabilizer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. prepare $</td>
<td>+\rangle$</td>
<td>$\frac{1}{\sqrt{2}}(</td>
</tr>
<tr>
<td>2. prepare $</td>
<td>1\rangle$</td>
<td>$\frac{1}{\sqrt{2}}(</td>
</tr>
<tr>
<td>3. CNOT$_{1,2}$</td>
<td>$\frac{1}{\sqrt{2}}(</td>
<td>01\rangle +</td>
</tr>
</tbody>
</table>

$X \otimes I \rightarrow X \otimes X$  
$I \otimes X \rightarrow I \otimes X$  
$I \otimes Z \rightarrow Z \otimes Z$

$Z \otimes I \rightarrow Z \otimes Z$  
$I \otimes X \rightarrow I \otimes X$  
$I \otimes Z \rightarrow Z \otimes Z$

Stabilizer algebra

Rule: S stabilizes $|\psi\rangle \Rightarrow USU^\dagger$ stabilizes $U|\psi\rangle$

$X \otimes I \rightarrow X \otimes X$  
$I \otimes X \rightarrow I \otimes X$  
$I \otimes Z \rightarrow Z \otimes Z$

$Z \otimes I \rightarrow Z \otimes Z$  
$I \otimes X \rightarrow I \otimes X$  
$I \otimes Z \rightarrow Z \otimes Z$

Def: Stabilizer state on n qubits = intersection of +1 eigenspaces of n independent commuting Paulis

Example:

<table>
<thead>
<tr>
<th>Operation</th>
<th>State</th>
<th>Stabilizer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. prepare $</td>
<td>+\rangle$</td>
<td>$\frac{1}{\sqrt{2}}(</td>
</tr>
<tr>
<td>2. prepare $</td>
<td>1\rangle$</td>
<td>$\frac{1}{\sqrt{2}}(</td>
</tr>
<tr>
<td>3. CNOT$_{1,2}$</td>
<td>$\frac{1}{\sqrt{2}}(</td>
<td>01\rangle +</td>
</tr>
</tbody>
</table>
**Stabilizer algebra**

- Rule: $S$ stabilizes $|\psi\rangle \Rightarrow USU^\dagger$ stabilizes $U|\psi\rangle$
  
  $\begin{align*}
  X \bullet &= \bullet \otimes X \otimes X \\
  Z \bullet &= \bullet \otimes Z \otimes Z
  \end{align*}$

- Example:
  
<table>
<thead>
<tr>
<th>Initial stabilizers</th>
<th>Final stabilizers</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>+\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>+\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>0\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>0\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>0\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>0\rangle$</td>
</tr>
</tbody>
</table>

Steane code $|0\rangle_L$

**Outline**

- Technical background
  - Error correction
  - Quantum ECCs
  - Stabilizer algebra

- Ancilla preparation and verification
  - Steane preparation and heuristic verification
    - for Steane 7-qubit, distance-3 code
    - for Bacon/Shor 9-qubit, distance-3 code
    - for higher-distance codes
  - Strictly fault-tolerant verification
    - repeated purification
    - tweaked

- Rigorous noise threshold for 23-qubit, distance-7 Golay code
  - Technical setup
  - Combinatorial analysis

**Idea:** Differently prepare ancillas to verify against each other

- No postselection for Steane code [Aliferis]
- Halves preparation complexity for 23-qubit Golay code
Steane encoded ancilla preparation

1. Using Gaussian elimination, and by rearranging qubits, put states $X$ (or $Z$) generators in standard form.

\[
\begin{align*}
\frac{1}{2} I & |A \\
\text{(or } A^T | I) \\
\end{align*}
\]

\[
\begin{array}{c}
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\end{array}
\]

\[
\Rightarrow
\begin{array}{c}
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\end{array}
\]

2. Starting with $|1^k 0^{n-k}\rangle$, use CNOT gates from first $k$ qubits into last $n-k$ qubits to generate each stabilizer.

\[
\begin{align*}
\text{Control qubits target qubits} \\
\text{target qubits control qubits} \\
\text{initial X stabilizers:} \\
\text{X X X X} \\
\text{X X X X} \\
\text{X X X X} \\
\text{X X X X} \\
\end{align*}
\]

\[
\Rightarrow
\begin{align*}
\text{X X X X} \\
\text{X X X X} \\
\text{X X X X} \\
\text{X X X X} \\
\end{align*}
\]

\[
\begin{align*}
\text{X X X X, X X X, X X} \\
\text{X X X, S X X, S X X} \\
\end{align*}
\]

2. Stabilizers are correctly generated automatically.
Steane encoded ancilla preparation

- Steane encoded ancilla preparation, and by rearranging points, put states \( X \) (or \( Z \)) generators in standard form.
- Steane encoding circuit:
  - Gives correlated errors e.g., weight-two \( X \) errors occur with 1st-order probability
  - \( Z \) errors are not correlated, so \( Z \) error verification is not required.

\[ \begin{array}{cccccc}
| & | & | & | & | \\
+ & + & + & + & + \\
| & | & | & | & | \\
0 & 0 & 0 & 0 & 0 \\
| & | & | & | & | \\
0 & 0 & 0 & 0 & 0 \\
| & | & | & | & | \\
0 & 0 & 0 & 0 & 0 \\
\end{array} \]

Steane heuristic verification

- Steane \( |0\rangle_L \) encoding circuit:
  - Gives correlated errors e.g., weight-two \( X \) errors occur with 1st-order probability
  - \( Z \) errors are not correlated, so \( Z \) error verification is not required.

\[ \begin{pmatrix} I & I & I & Z & Z & Z \\ I & I & I & I & I & I \\ Z & Z & Z & X & X & X \\ Z & Z & Z & X & X & X \\ \end{pmatrix} \]

- Verification against \( X \) errors is required for fault tolerance

- \( Z \) - \( ZZZ \) has no effect on \( Z \).
- \( ZZI \) has same effect as \( IIZ \), so all \( Z \) errors have reduced weight either 0 or 1.
Steane heuristic verification

- Purification: Prepare two ancillas, check one against the other. Postselect on no detected errors in second ancilla.

- In general: (but with a distance-3 code, this simplifies)

  - error weight: 0 1 2 3 4 ...
  - error order: 0 1 2 2 2 ...

- Steane finds, roughly, that one round of purification works well (according to simulations). However, this is not strictly fault-tolerant for codes of distance > 3.

  - Def: Fault-tolerant: Weight >1 errors are at most second-order events Suffices for threshold existence
  - Def: Strictly fault-tolerant: Weight-k errors are at most kth-order events, \( k \leq t+1 = (d+1)/2 \)

  - Required for \( p \rightarrow p^{t+1} \) effective error behavior

Def: Fault-tolerant: Weight >1 errors are at most second-order events

Def: Strictly fault-tolerant: Weight-k errors are at most kth-order events, \( k \leq t+1 = (d+1)/2 \)

Steane heuristic verification

- Purification: Prepare two ancillas, check one against the other. Postselect on no detected errors in second ancilla.

- In general: (but with a distance-3 code, this simplifies)

  - error weight: 0 1 2 3 4 ...
  - error order: 0 1 2 2 2 ...

- Steane finds, roughly, that one round of purification works well (according to simulations). However, this is not strictly fault-tolerant for codes of distance > 3.

  - Def: Fault-tolerant: Weight >1 errors are at most second-order events Suffices for threshold existence
  - Def: Strictly fault-tolerant: Weight-k errors are at most kth-order events, \( k \leq t+1 = (d+1)/2 \)

  - Required for \( p \rightarrow p^{t+1} \) effective error behavior

Encoding complexities

<table>
<thead>
<tr>
<th>Code type</th>
<th>n</th>
<th>k</th>
<th>d</th>
<th># encoded</th>
<th># rounds</th>
<th># gates</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hamming</td>
<td>7</td>
<td>1</td>
<td>3</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Golay</td>
<td>23</td>
<td>7</td>
<td>7</td>
<td>177</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BCH</td>
<td>31</td>
<td>11</td>
<td>15</td>
<td>122</td>
<td></td>
<td></td>
</tr>
<tr>
<td>QR</td>
<td>47</td>
<td>11</td>
<td>15</td>
<td>281</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BCH</td>
<td>63</td>
<td>27</td>
<td>27</td>
<td>350</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QR</td>
<td>79</td>
<td>13</td>
<td>15</td>
<td>799</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BCH</td>
<td>127</td>
<td>47</td>
<td>47</td>
<td>1802</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Encoding complexity can depend on code presentation.

[Steane, quant-ph/0207119]
Avoiding verification: Bacon/Shor 9-qubit code

- Shor’s code: Concatenate 3-qubit repetition code with its dual
  - Repetition code: 0 → 000, 1 → 111
    - Stabilizers ZI, ZZ, ZI.
    - Logical X is XXX, logical Z is ZII – IZI – IIZ.
    - Corrects one bit flip (X) error.

- Dual repetition code: |+⟩ → |+ + +⟩, |−⟩ → |− − −⟩
  - Stabilizers XIX, IXI, XIX.
  - Logical Z is ZZZ, logical X is XII – IXI – IIX.
  - Corrects one phase flip (Z) error.

- Concatenation: Stabilizer generators:
  - Corrects one X error in each block of three, and one Z error.

- Bacon: Remove code redundancies
  - Operator error-correcting code \( H = (A \otimes B) \oplus C \)

Bacon: Restore X/Z symmetry

Ike covered this...

- Shor’s code: Concatenate 3-qubit repetition code with its dual
- Preparing encoded ancilla \(|\_\_\_\rangle_L\):

\[
\begin{array}{cccccccc}
Z & Z & \cdots & Z & & Z & \cdots & Z \\
\cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdots & Z & \cdot & \cdot & \cdots & Z \\
\cdot & \cdot & \cdots & Z & \cdot & \cdot & \cdots & Z \\
\cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdot \\
\end{array}
\]

Thus \(|\_\_\_\rangle_L = (|000\rangle + |111\rangle)^\otimes 3\) and requires no Z verification. [Aliferis]

- Bacon: Restore X/Z symmetry
Golay code naïve verification

- **Purification:** Prepare two ancillas, check one against the other. Postselect on no detected errors in second ancilla.
- **In general, repeated purification:**

<table>
<thead>
<tr>
<th>Error weight</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 verifications</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1 verification</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2 verifications</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>3 verifications</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Error order with Z</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 verifications</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1 verification</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2 verifications</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3 verifications</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Def: Fault-tolerant: Weight > 1 errors are at most second-order events
Def: Strictly fault-tolerant: Weight-k errors are at most kth-order events, $k = t + 1 = (d+1)/2$

Golay code naïve verification

- For distance-seven code, generically need three rounds of verification against X errors, and two rounds of Z verification.
- **Repeated purification circuits:**

Circuit 2: First check X, then Z

Circuit 1: First check for Z, then X errors
Golay code naïve verification

- Repeated purification circuits:

  \[ \text{Circuit 3: } \times \times \times \times \]

  \[ \text{Circuit 4: One of many other variations} \]

Smarter verification for Steane code

- Observe: X errors are correlated, but not arbitrary.

- Assume at most one X error occurs during preparation. What are the possible errors on the output?
  - Arbitrary single-bit errors (of course)
  - But what else?

X stabilizers:

\[ \text{XIXIXIX} \]
\[ \text{IXXIXXX} \]
\[ \text{IIIXXXX} \]
Smarter verification for Steane code

Observe: X errors are correlated, but not arbitrary.

X stabilizers: \[
\begin{align*}
X_{\text{stabilizers}}: & \quad \text{XIXIXIX} \\
& \quad \text{IXXIIXX} \\
& \quad \text{IIIXXXX}
\end{align*}
\]

Smarter verification for Steane code

Observe: X errors are correlated, but not arbitrary.

X stabilizers: \[
\begin{align*}
X_{\text{stabilizers}}: & \quad \text{XIXIXIX} \\
& \quad \text{IXXIIXX} \\
& \quad \text{IIIXXXX}
\end{align*}
\]
Smarter verification for Steane code

Observe: X errors are correlated, but not arbitrary.

- X stabilizers:
  - XIXIXIX
  - IXXIIXX
  - IIIIXXX

Assume at most one X error occurs during preparation. What are the possible errors on the output?

- Arbitrary single-bit errors (of course)
- But what else?
  - $X_1X_3X_5X_7$
  - $X_1X_5X_7X_3$
  - $X_1X_5$ $X_7$
  - $X_3$

With one X error during preparation, what are the possible output errors?

- Arbitrary single-bit errors, and
  - $X_1X_7$
  - $X_2X_3$
  - $X_4X_5$

→ correct!

Conclusion: Applying CNOTs from a 123 ancilla into a 321 ancilla, correlated output errors from a single gate error can be distinguished, and corrected for.

- Arbitrary single-bit errors, and
  - $X_1X_3$
  - $X_2X_6$
  - $X_4X_7$

→ don’t correct!
With one X error during preparation, possible output errors are:
- Arbitrary single-bit errors, and
  \[ X_1 X_7 \]
  \[ X_2 X_3 \] → correct!
  \[ X_4 X_5 \] → don’t correct!

Conclusion:
Applying CNOTs from a 123 ancilla into a 321 ancilla, correlated output errors from a single gate error can be distinguished, and corrected for. Postselection on no detected errors is not necessary. [Aliferis]

Consequences:
- No need for ancilla to wait for measurement results before using it.
- Reduced overhead.
- Provable threshold increases, but ancilla reliability may decrease.


golay code preparation and verification

Stabilizers:

```
x . x . x . x . x . x . x . x . . . . .
xxxx . xx . x . . . . . . . . . . . .
xxxx . xx . x . . . . . . . . . . . .
xxxx . xx . x . . . . . . . . . . . .
xxxx . xx . x . . . . . . . . . . . .
x . x . x . x . x . . . . . . . . . .
xxxx . . . . . . . . x . . . . . . .
xxxx . . . . . . . . x . . . . . . .
xxxx . . . . . . . . x . . . . . . .
x . x . x . x . x . . . . . . . . . .
x . x . x . x . x . . . . . . . . . .
```


Golay code preparation and verification

Preparation circuit (shorthand):

\[
\begin{align*}
1.2..3..4567X & \quad \ldots \ldots \\
2345.67.1\ldots X & \quad \ldots \ldots \\
.2345.67.1\ldots X & \quad \ldots \ldots \\
..5671.23.4\ldots X & \quad \ldots \ldots \\
...7143.56.2\ldots X & \quad \ldots \ldots \\
3.7.2.156..4\ldots X & \quad \ldots \ldots \\
4562..1..73\ldots X & \quad \ldots \ldots \\
51.367..42\ldots \ldots X & \quad \ldots \ldots \\
.71.452..36\ldots \ldots X & \quad \ldots \ldots \\
6.1\ldots 43725\ldots \ldots X & \quad \ldots \ldots \\
.6.3\ldots 42715\ldots \ldots X & \quad \ldots \ldots
\end{align*}
\]

7 rounds

Verification by repeated postselection:

\[
\text{Circuit 4: One of many other variations, } \frac{1}{2} x x
\]
Golay code correlated errors

Possible output errors from single X failure:
Xs on
01234567 ~ Ø
0 234567 ~ 1
0 34567 ~ 12
0 4567 ~ 123
0 567 ~ 1234
0 67 ~ 12345
0 7 ~ 123456

If we reversed the rounds...
07654321 ~ Ø
0 654321 ~ 7
0 54321 ~ 67
0 4321 ~ 567
0 321 ~ 4567
0 21 ~ 34567
0 1 ~ 234567

Possible output errors from two X failures:
consecutive sequences \([a,b] = \{a,a+1,\ldots,b-1,b\}\) e.g. 2345

Golay code final preparation and encoding circuits

Round permutations:
\(A = 1243567\)
\(B = 6274531\)
\(A^r = 7653421\)

Possible output errors from two X failures:

Checking fault-tolerance reduces to checking following circuits:

Prepare two \(|0\) ancillas using round permutation \(A\). Verify once against \(X\) errors (measuring transversely in \(Z\) eigenbasis).

Prepare two \(|0\) ancillas using round permutation \(A^r\) (A reversed). Verify once against \(X\) errors.

Prepare two ancillas using round permutations \(B\) and \(B^r\). Verify against \(Z\) errors.

Verify once against \(Z\) errors (transverse \(X\) measurement).
Verify against \(X\) errors again.

Conclusion:
Reduces verification circuit complexity by half.
Reduces overhead esp. at high error rates.
Increases provable threshold (reduced combinatorial complexity allows much better computer-aided counting analysis).
But ancilla reliability may decrease.
Analysis

- Aharonov & Ben-Or threshold proof:
  - Idea: Maintain inductive invariant of (1-)goodness. (A good block "has at most one bad subblock.")
- Inefficient analysis:
  - \( p \rightarrow \left( \frac{n}{2} \right) p^2 \) not \( cp^3 \) for a distance-five code
  - No threshold for concatenated distance-three codes
- [R '05, Aliferis/Gottesman/Preskill '05] proofs apply too to distance-three codes
  - Idea: Maintain as inductive invariant recursive control over the probability distribution of errors in each block
- Gives rigorous (and fairly efficient) criterion for threshold

Combinatorial analysis
Conclusion

- Technical background
  - Error correction
  - Stabilizer algebra
  - Quantum ECCs
- Ancilla preparation and verification
  - Steane preparation and heuristic verification
    - for Steane 7-qubit, distance-3 code
    - for Bacon/Shor 9-qubit, distance-3 code
    - for higher-distance codes
  - Strictly fault-tolerant verification
    - repeated purification
    - tweaked
- Rigorous noise threshold for 23-qubit, distance-7 Golay code
  - Technical setup
  - Combinatorial analysis

**Idea:** Differently prepare ancillas to verify against each other
  - No postselection for Steane code [Aliferis]

**Result:** Threshold of $9.8 \times 10^{-5}$, or $>10^{-4}$ with 99.9% statistical confidence.

- Simulations haven’t been run to estimate actual improvement.
- Other effects, particularly locality, still need to be analyzed.
- Analyze schemes which aren’t strictly fault-tolerant.
- Consider schemes with no verification required.

- Other effects, particularly locality, still need to be analyzed.
- Consider schemes which aren’t strictly fault-tolerant.