Overview of fault-tolerant quantum computation

Why is building a quantum computer hard?

**Noise!**

**Shor's factoring algorithm**

factor a $K$-bit number using

$72K^3$ gates, (vs. $e^{K^{1/3}}$ classically)

on $5K$ qubits

$K=1024 \implies 10^{11}$ gates on 5000 qubits

$\implies$ need error $<10^{-11}$ per gate

(from environmental noise & control errors)

Realistic noise rates: 1% per gate? (maybe $10^{-5}$ or $10^{-4}$)
Table 1 | Current performance of various qubits

<table>
<thead>
<tr>
<th>Type of qubit</th>
<th>$T_2$</th>
<th>Benchmarking (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>One qubit</td>
</tr>
<tr>
<td>Infrared photon</td>
<td>0.1 ms</td>
<td>0.016</td>
</tr>
<tr>
<td>Trapped ion</td>
<td>15 s</td>
<td>0.48†</td>
</tr>
<tr>
<td>Trapped neutral atom</td>
<td>3 s</td>
<td>5</td>
</tr>
<tr>
<td>Liquid molecule nuclear spins</td>
<td>2 s</td>
<td>0.01†</td>
</tr>
<tr>
<td>$e^-$ spin in GaAs quantum dot</td>
<td>3 µs</td>
<td>5</td>
</tr>
<tr>
<td>$e^-$ spins bound to $^{31}P,^{28}Si$</td>
<td>0.6 s</td>
<td>5</td>
</tr>
<tr>
<td>$^{29}Si$ nuclear spins in $^{28}Si$</td>
<td>25 s</td>
<td>5</td>
</tr>
<tr>
<td>NV centre in diamond</td>
<td>2 ms</td>
<td>2</td>
</tr>
<tr>
<td>Superconducting circuit</td>
<td>4 µs</td>
<td>0.7†</td>
</tr>
</tbody>
</table>

[Ladd et al., Nature 2010]

Solution: Fault-tolerant quantum computation

![Ideal circuit diagram]

Intuition: Noise threshold

distance-3 code $\rightarrow$ quadratic reduction in error rate

"Effective logical error rate"
Concatenate the scheme for arbitrary reliability

"Threshold theorems": For various noise models,
- Tolerable noise rate is a constant $\geq 0$
- Size overhead is polylogarithmic or constant [Gottesman '13]

Ingredients:
- Noise model
  - stochastic?
  - Hamiltonian coupling to environment?
- Quantum error-correcting code
- Fault-tolerance scheme

Remark: If you know your noise model well, then you should try to deal with it at the hardware level; techniques for correcting general noise are more expensive.

Example: Knill's postselection-based scheme [Nature '05]
Remark: • Realistically, you’ll probably not want to have more than one or two levels of concatenation
  • The best scheme depends on the noise rate.
    ⇒ start with simple, high-threshold code & switch to more efficient, but lower-threshold code once the effective noise rate is low enough

Open problems:
• Develop fault-tolerance schemes that maximize tolerable noise rate with minimal overhead
  - especially schemes based on the surface code or large, efficient QECCs

Examples:
• Encoding step bottleneck: how to get data into the code

Control error cascade!
[Paetznick, Reichardt '11]
• Computing with more efficient codes

Standard scheme

Alternative

7 Logical qubits

15 Physical qubits

Harder to manipulate encoded data

[Paetznick, Reichardt '11]

Harrington, Reichardt '12

• More efficient, fault-tolerant, universal computation
Universal FT computation w/ transversal gates & error correction [Paetznick, Reichhardt '12]

Open:
• Can we construct inherently fault-tolerant systems?
  - "self-correcting" quantum codes
  - anyonic systems

Solutions needed!

★ 95 µs [Rigetti et al., 2012]

"This technology [is] a strong candidate for the immediate construction of"
Course Outline:
1. Quantum codes & stabilizer algebra
2. Fault-tolerance & threshold theorems
3. Surface code
Quantum codes and stabilizer algebra

Quantum Codes

\[ 10 \mapsto \text{encoded } 0 \]
\[ 11 \mapsto \text{encoded } 1 \]

Str. looking at only a few qubits of
\[ \alpha \text{ encoded } 0 \] + \[ \beta \text{ encoded } 1 \]
tells you nothing about \( \alpha, \beta \)

"distance d" means nothing is revealed by any subset of \( \frac{d-1}{2} \) qubits

Errors are digital

It seems like there are many different kinds of errors that can affect a quantum state.

In fact, infinitely many, since errors can be continuous. But look closer...

Pauli operators

\[ I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad \gamma = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \]
\[ X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad iXZ = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \]

Observe: • These form a basis, over \( \mathbb{C} \), for all 2x2 complex matrices
• Their k-fold tensor products form a basis for all \( 2^k \times 2^k \) complex matrices
\( \begin{align*}
\text{eq. } k &= 2 \\
&\quad I \otimes I, I \otimes X, I \otimes Y, I \otimes Z \\
&\quad ; \\
&\quad Z \otimes I, \ldots \ldots, Z \otimes Z \\
\textbf{Corollary:} \text{ Any error (any operation) on } k \text{ qubits}
&\text{ can be expanded as a linear combination of these Paulis.} \\
&\Rightarrow \text{ It is enough to protect against } X, Y, Z \text{ errors.}
\end{align*} \)

\[ 14\rangle = \text{codeword} \]

\[ R = \text{recovery procedure} \]

\[ R(X_j 14\rangle) = 14\rangle | \Phi_X \rangle \]

\[ R(Y_j 14\rangle) = 14\rangle | \Phi_Y \rangle \]

\[ R(Z_j 14\rangle) = 14\rangle | \Phi_Z \rangle \]

\[ R(14\rangle) = 14\rangle | \Phi \rangle \]

\[ E_j = \alpha I + \beta X + \gamma Y + \delta Z \]

\[ \Rightarrow \ R(E_j 14\rangle) = 14\rangle \otimes (\alpha | 1 \rangle \langle 0 | + \beta | \Phi_X \rangle \langle \Phi_X | + \gamma | \Phi_Y \rangle \langle \Phi_Y | + \delta | \Phi_Z \rangle \langle \Phi_Z |) \]

\[ \textbf{Dual bases} \]

\[ 14\rangle = (0) \]

\[ 14\rangle = \frac{1}{\sqrt{2}} (1) \]

\[ 10\rangle = (0) \]

\[ 10\rangle = \frac{1}{\sqrt{2}} (-1) \]

\[ \mathbf{Z} = (1, 0) \]

\[ \mathbf{X} = (0, 1) \]

\[ \textbf{The simplest quantum code} \]

\[ 10\rangle \mapsto 1000\rangle \quad \text{distance 3 against } X \text{ errors} \]

\[ 11\rangle \mapsto 1111\rangle \]

\[ 14\rangle \mapsto 1+++) \quad \left( \begin{array}{c}
10\rangle = \frac{1}{\sqrt{2}} (14\rangle + 1-) \\
10\rangle = \frac{1}{\sqrt{2}} (1++) + 1-) \end{array} \right) \mapsto 1+++7 + 1--7 \]

\[ 1-7 \mapsto 1---7 \quad \left( \begin{array}{c}
11\rangle = 14\rangle - 1-7 \\
11\rangle = 14\rangle - 1-7 \end{array} \right) \mapsto 1+++7 - 1--7 \]

\[ \text{Combine the codes} \quad \text{concatenate one on the other} \quad \text{to} \]

2014-2015 USC notes Page 2
Combine the codes — concatenate one on the other — to protect against both kinds of errors.

Parity checks and logical operators

\[ Z : |0\rangle \mapsto |0\rangle, \quad |1\rangle \mapsto |-1\rangle \]
\[ X : |+\rangle \mapsto |+\rangle, \quad |-\rangle \mapsto |-\rangle \]

\(Z \otimes Z\) measures parity of two qubits

\[
\begin{array}{c|cc}
+1 & 00 & 01 \\
00 & 10 & 11 \\
11 & 01 & 00 \\
\end{array}
\]

Repetition code is "stabilized by" \(Z \otimes Z \otimes I,\)
\(I \otimes Z \otimes Z\)

\[
(Z \otimes Z \otimes I)(|000\rangle + |111\rangle) = |000\rangle + |111\rangle
\]

Logical operators

\[ X \otimes X \otimes X : |000\rangle \leftrightarrow |111\rangle \]
\[ Z \otimes I \otimes I : |++\rangle \leftrightarrow |--\rangle \]

Similarly, \(X\) measures parity in the \(|+\rangle/|\rangle\) basis

\[ \Rightarrow \text{Stabilizers/parity checks of } 9\text{-qubit code} : \]

\[
\begin{array}{c}
Z \otimes Z \\
I \otimes Z \\
Z \otimes I \\
I \otimes I \\
x \otimes x \otimes x \otimes x \\
i \otimes i \otimes i \\
i \otimes i \otimes x \otimes x \\
i \otimes i \otimes x \otimes x
\end{array}
\]

(and everything in the abelian group they generate)

**Stabilizer Algebra** (Gottesman-Knill theorem)

Pauli operators
\[ I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad Y = iXZ \]
Stabilizer state = +1 eigenvalue eigenvector of a set of Pauli operator tensor products

Examples:

<table>
<thead>
<tr>
<th>State</th>
<th>Stabilizers</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>10\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>1+\rangle = \frac{1}{\sqrt{2}}(10\rangle + 11\rangle)$</td>
</tr>
<tr>
<td>$</td>
<td>11\rangle \otimes 1+\rangle$</td>
</tr>
<tr>
<td>$\frac{1}{\sqrt{2}}(100\rangle + 111\rangle)$</td>
<td>${\mathbb{I}, \mathbb{X} \otimes \mathbb{Z}, Z \otimes \mathbb{Z}, -Y \otimes Y}$</td>
</tr>
<tr>
<td>$\frac{1}{\sqrt{2}}(101\rangle - 110\rangle)$</td>
<td>${-\mathbb{X} \otimes \mathbb{X}, -Z \otimes Z}$</td>
</tr>
<tr>
<td>?</td>
<td>$\mathbb{X} \otimes \mathbb{X}, Z \otimes Z, 11\otimes X$</td>
</tr>
</tbody>
</table>

An $n$-qubit stabilizer state has $n$ independent, pairwise-commuting stabilizers. (They generate a $2^n$ element abelian group.)

Commutation relationships:

$XZ = -ZX$ anticommute

$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

$XZ \otimes Z$ commute

$1 \otimes Z \otimes Z$ even # of different Paulis

$\{\mathbb{Z} \otimes \mathbb{Z}, \mathbb{Z} \otimes \mathbb{Z}, \mathbb{Z} \otimes \mathbb{Z}, \mathbb{Z} \otimes \mathbb{Z}, \mathbb{Z} \otimes \mathbb{Z}, \mathbb{Z} \otimes \mathbb{Z}, \mathbb{Z} \otimes \mathbb{Z}\}$ anticommute

$\leq$ odd # of different Paulis
Manipulating stabilizer states

State | Stabilizers
--- | ---
$\mid 14\rangle$ | $P, Q, \ldots$
$\mid U14\rangle$ | $UPU^+, UQU^+, \ldots$

since $(UPU^+) \mid U14\rangle = U\mid P14\rangle = \mid U14\rangle$

**Def.**: A unitary is **Clifford** if it conjugates Pauli operators to Pauli operators.

**Examples**:

- **Hadamard**
  
  $H = \frac{1}{\sqrt{2}} (|1\rangle \langle 0| + |0\rangle \langle 1|)$

  $H \mid 0\rangle = \mid 1\rangle , \quad H \mid 1\rangle = \mid 0\rangle$

  $H^2 H = \mathbf{X} \quad H \otimes H = \mathbf{Z}$

- **CNOT**

  ![CNOT Diagram]

**Example**: To prepare $\frac{1}{\sqrt{2}} (\mid 1000\rangle + \mid 1111\rangle)$

<table>
<thead>
<tr>
<th>Initial stabilizers</th>
<th>$\overset{\text{CNOT}_{X1}^2}{\rightarrow}$</th>
<th>$\overset{\text{CNOT}_{Z1,3}^3}{\rightarrow}$</th>
<th>Final stabilizers</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X \mid 11$</td>
<td>$XX \mid 1$</td>
<td>$XXX$</td>
<td></td>
</tr>
<tr>
<td>$1 \mid 21$</td>
<td>$Z \mid 21$</td>
<td>$Z \mid 1$</td>
<td></td>
</tr>
<tr>
<td>$1 \mid 12$</td>
<td>$11 \mid 2$</td>
<td>$Z \mid 12 \sim Z \mid 22$</td>
<td></td>
</tr>
</tbody>
</table>

2014-2015 USC notes Page 5
Stabilizer codes

n qubits, \( m \leq n \) independent stabilizers

\[ \Rightarrow n-m \text{ encoded qubits (degrees of freedom)} \]

**Example:**

\[ n = 3, \ m = 2 \]

\[ 1 \mathbb{Z} \mathbb{Z} \]

\[ 1 \mathbb{X} \mathbb{X} \]

\[ \Rightarrow \text{one encoded qubit} \]

\[ 1^4 \otimes \frac{1}{2} (100 \uparrow + 111 \uparrow) \]

logical \( X : X \uparrow \uparrow \uparrow \)

logical \( Z : Z \uparrow \uparrow \uparrow \)

\( \text{commute with} \)

\( \text{anticommute with} \)

\( \text{stabilizers,} \)

\( \text{each other} \)

**Exercise:** Derive quantum teleportation using the stabilizer formalism.

\[ 1^4 \]

\[ 100 \uparrow + 111 \uparrow \} \]

\[ 1 \mathbb{H} \]

\[ 2 \]

\[ \mathbb{X} \]

**Examples of Stabilizer Codes**

- **Shor code** \([9, 1, 3]\)

\[ n = \text{# physical qubits} \]

\[ k = \text{# logical qubits} \]

\[ d = \text{distance} \]
- $[5, 1, 3]$ code
  - smallest qubit code w/ $d \geq 3$
  - not CSS

- $[4, 2, 2]$ error-detecting code
  \[
  \begin{array}{cccc}
  X & X & X & X \\
  \hline \\
  \downarrow & \downarrow & \downarrow & \downarrow \\
  X & X & X & X \\
  \end{array}
  \]

  Exercises:
  - Generalize to $[n, n-2, 2]$ for $n$ even
  - Give a $[3, 1, 2]$ code on qutrits
  
  Hint: Use $(0, 0, 0)$ and $(e^{\pi i/3}, 0)$

- Steane code $[7, 1, 3]$
  - self-dual CSS code

  Exercises:
  - Give a circuit that prepares $\text{10}$ encoded in this code.
  - Generalize to the family of Hadamard codes
    $[2^s - 1, 2^s - 1 - 2s, 3]$
\[ s = 3 : [7, 1, 3] \]
\[ s = 4 : [15, 7, 3] \]
\[ s = 5 : [31, 21, 3] \]

- Many other codes
  
  Concatenated, eg. \([7,1,3]\) on itself \(\rightarrow [49,1,9]\)
  
  Golay \([23, 1, 7]\) \(\rightarrow [21, 3, 5]\)

  BCH \([31, 11, 5]\)
  \([63, 27, 7]\)
  \([127, 29, 15]\)

  QR \([47, 1, 11]\)
  \([79, 1, 15]\)
  \([103, 1, 19]\)

  RM \([7, 1, 3]\), \([81, 1, 7]\), \([127, 1, 15]\)


and [Grassl, http://codetables.de/]
Simulating fault-tolerance schemes

Syndrome extraction and error correction for a stabilizer code only needs Clifford gates.

⇒ Can be efficiently simulated classically for stochastic Pauli noise models

Example: Depolarizing noise model

\[ \begin{array}{c}
\text{H} \\
\text{1-4X} \\
\text{1-4Y} \\
\text{1-4Z}
\end{array} \]

\[ \begin{array}{c}
\text{Y/15} \quad \text{1X} \\
\text{Y/15} \quad \text{Z}
\end{array} \]
We cannot simulate a universal set of quantum gates, nor Hamiltonian noise models.

**THE BACON-SHOR "SUBSYSTEM" CODE**

The 9-qubit Shor code is asymmetrical between $X$ and $Z$. Let’s fix this...

X X X X X X 1 1 1   X X X X X X 1 1 1
  1 1 1 X X X X X   1 1 1 X X X X X
  Z Z 1             Z Z 1 Z Z 1 Z Z 1
  1 Z Z             1 Z Z 1 Z Z 1 Z Z 1

Now they are symmetrical, except with 4 extra $Z$ stabilizers. Why do we need them? Get rid of them!

```
\begin{align*}
\text{Stabilizers:} & \\
     & X X X X X X 1 1 1 \\
     & 1 1 1 X X X X X \\
     & Z Z 1 Z Z 1 Z Z 1 \\
     & 1 Z Z 1 Z Z 1 Z Z 1 \\
\end{align*}
```
Observe: The code distance is \( d = 2 \),
but the first qubit is protected to distance 3!
(That is, any operator acting non-trivially on encoded qubit 1 has weight \( \geq 3 \).

Proof: The two \( X \) stabilizers allow for determining which block a \( Z \) error occurred on, but not where in the block. If you guess wrong, that just causes a logical error on encoded qubits 2 to 5. \( \square \)

Why is this useful?
• Fewer stabilizers \( \Rightarrow \) extracting syndromes is easier.
• Easier to prepare encoded states

e.g.,

Shor-encoded \( |10\rangle = \frac{1}{2}(10^3 0^3 0^3 + 11^3 1^3 0^3) + 11^3 0^3 1^3 + 0^3 1^3 1^3) \)

Bacon-Shor encoded \( |10\rangle : \) many choices, depending on qubits 2 to 5, but
encoded $10+++> = \left( \begin{array}{c} + \\ + \\ + \end{array} \right) \otimes^3 \left( \begin{array}{c} 1 \\ - \\ - \end{array} \right) \frac{1}{\sqrt{2}} \times \frac{1}{\sqrt{2}} \times \frac{1}{\sqrt{2}}$

-simpler, easier to prepare fault-tolerantly

- Easier to apply encoded Hadamard $H$
  - apply $H^{\otimes^3}$, then "transpose" the qubits

- Much easier to extract error syndromes

Measure

\[
\begin{array}{ccc}
\cdot & \cdot & \cdot \\
\times & \cdot & \cdot \\
\times & \cdot & \cdot \\
\times & \cdot & \cdot \\
\times & \cdot & \cdot \\
\end{array}
\]

\[10> \otimes 11> \text{ one-qubit ancilla is enough} \]

- errors can't spread

Remark: Why isn't noise such a problem for "CLASSICAL computation?"

SELF-CORRECTING CODES

Classical magnets: $H = \sum_{\text{edges}(i,j)} Z_i \otimes Z_j$

-1 if spins agree (00 or 11)
+1 if they disagree (01 or 10)

ID: 00000000000

or

\[
\begin{array}{cccccccccccc}
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\end{array}
\]

ground states
or

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>

ground states
repetition code
codewords

2 disagreements

2D:

disagreements all along the boundary

⇒ In 2D or higher, energy cost grows with # of errors

Energy penalty means errors will tend to shrink away.

But not in 1D:

\[
\begin{array}{ccc}
0 & 1 & 0 \\
0 & 1 & 0 \\
\end{array}
\]

\{ same energy! \}
Self-correcting quantum codes?

4D ✓ generalization of surface code

3D lattice ✓? - probably: see Bravyi & Haah

- the “compass model” (Bacon-Shor with Y⊗Y in the 3rd direction) is conjectured to self-correct

2D lattice ? maybe, but probably not

Bravyi & Terhal

Kay & Colbeck

- impossible for standard stabilizer codes,
  possible for a subsystem code (like Bacon-Shor)
Fault-tolerance schemes and threshold theorems

**Outline**

- Fault tolerance
  - what it is ✓
  - examples ✓
  - carefully tracking errors
- Threshold 1/2 for erasure errors ✓
- Other fault-tolerance schemes ✓
  - Steane-type error correction
- AGP threshold theorem ✓
  - malignant set counting
- Other threshold existence theorems ✓
- Overhead analysis of fault-tolerance schemes ✓

**Fault-tolerance schemes and threshold theorems**

Ben Reichardt

**Recall:**

Noise threshold intuition

- distance-3 code \(\rightarrow\) quadratic reduction in error rate

\[ \frac{c \cdot p^2}{p} \]
Concatenate the scheme for arbitrary reliability

**CNOT gate:** copies $X$ forward, $Z$ backward

\[
\begin{align*}
X & \quad = \quad X \\
X & \quad = \quad X \\
Z & \quad = \quad Z \\
Z & \quad = \quad Z
\end{align*}
\]

**FAULT TOLERANCE**

What we don’t do:

![Fault Tolerance Diagram]

We need to compute on the encoded data.
Error-correction procedures are themselves faulty, but applied periodically they hopefully keep noise under control.

**Fault-tolerant operations**

1. **CNOT gate**
   
   Start with a CSS code
   
   \[
   \begin{pmatrix}
   X \text{ stabilizers} \\
   Z \text{ stabilizers}
   \end{pmatrix}
   \]

   **Claim:** Transversal CNOTs between two code blocks implement logical CNOT.

   - and they don’t spread errors within either code block
   - 1 faulty gate → ≤ 1 error on each block

   **Why?**

   Block 1
   \[
   X \text{ stabilizers}
   \]
   \[
   X \text{ stabilizers}
   \]

   Block 2
   \[
   \text{same X stabilizers}
   \]
-so each block stays in the codespace

and logical $X$ operators are copied forward (as they should be)

logical $Z$ operators copied backward

2 Cat state preparation

$|0000\rangle + |1111\rangle$

This is not fault tolerant; errors can spread

$|000\rangle + |0110\rangle$

2014-2015 USC notes Page 4
one fault $\rightarrow$ 2 errors on output

**Solution:** Check for correlated errors.

1\>  
1\>  
1\>  
1\>  
1\>  
with the above $X11X$ error, this bit will flip to 1

Note that this will not catch $11XX$, but that can't happen w/ 1st-order probability

**Moral:** We only need to check for some correlated errors.

**Fault-tolerant state preparation:**

weight-$k$ errors should have probability $O(p^k)$.

(for $k$ up to obvious limit, eg., $k \leq 2$ for a distance-3 code)

? : Why do we only check for correlated $X$ errors?

Any two $Z$s are a stabilizer
for $100000+11111$.

3. **Syndrome extraction** (optional)

To correct errors, we need to know each stabilizer's sign, $\pm 1$.

Eg., repetition code

<table>
<thead>
<tr>
<th>code stabilizers</th>
<th>syndromes</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Z \equiv 1$</td>
<td>$-1$</td>
</tr>
<tr>
<td>$Z \equiv -1$</td>
<td>$1$</td>
</tr>
</tbody>
</table>
How do we measure these error syndromes, without collapsing the quantum state?

Example: Consider a code with a $\bar{Z}ZZZ$ stabilizer.

(a) Measuring each qubit & adding their parities gives the syndrome — but also collapses the state! It measures too much.

(b) Measure only the parity we want:

as desired, this copies $X$ errors down — but $Z$ errors are copied back!

(c) Fault-tolerant parity measurement
$$\begin{array}{c}
|+++angle \\
|---\rangle \\
\text{prepared fault-tolerantly}
\end{array}
\begin{array}{c}
\text{stabilizers} \\
\text{2 2 2 2} \\
\text{X X 1 1} \\
\text{1 X X 1} \\
\text{1 1 X X}
\end{array}

X \text{ errors are copied down, but two Xs are a stabilizer } \Rightarrow \text{ get the parity}

\textbf{Exercise}: \text{ Give a stabilizer circuit to prepare } 10\rangle \text{ encoded into the 7-qubit Steane code.}

\text{What correlated errors can the circuit create, i.e., single faults } \rightarrow \text{ errors of weight 2?}

\text{stabilizers:}
\begin{align*}
1 & 1 & 1 & X & X & X \\
1 & X & X & 1 & 1 & X \\
X & 1 & X & 1 & X & 1 \\
1 & 1 & 1 & \pm & 2 & 2 & 2 \\
1 & \pm & 2 & 2 & 1 & 1 & 2 \\
\pm & 1 & \pm & 2 & 1 & 2 & 1 \\
\pm & 2 & \pm & 2 & \pm & 2 & 2 & 2
\end{align*}
\begin{array}{c}
\text{code} \\
\text{stabilizers}
\end{array}

\text{code}
\begin{array}{c}
\text{logical Z}
\end{array}

\textbf{Threshold 1/2 for Erasure Errors}

\textbf{Theorem}: \text{ The threshold for erasure noise is } \frac{1}{2}.

[Knill, q-ph/0312190]

\textbf{Proof sketch:}

\textbf{Lemma}: \text{ For any } \varepsilon > 0, \text{ there exists a } (\text{CSS, stabilizer})\text{ QECC that with high probability corrects } \frac{1}{2} - \varepsilon
probability erasure errors.

Proof sketch:
Let $C$ be a uniformly random $n$-qubit stabilizer code (i.e., pick $n$-1 pairwise commuting random Pauli stabilizers).

\[
\text{Prob over choice of } C \text{ and the random error } E \left[ \text{code } C \text{ incorrectly decodes error } E \right] = 2^{-2(n)}.
\]

The actual error gives some syndrome $-\epsilon \xi + 1, -13^{n-1}$

\[
P[ \text{another Pauli error on the same qubits gives the same syndrome }] \leq \text{# of other Pauli errors on same qubits} = 4^{(\epsilon-\epsilon)n} 2^{n-1}\]

since any given nontrivial Pauli error's syndrome on a random stabilizer is $+1$ or $-1$ with equal prob. $\frac{1}{2}$

\[
2^{-2(n)} = P_{C,E}[C \text{ fails on } E] = \sum_{C} P(C) P_{E}[C \text{ fails on } E]
\]

so a good code exists.

\[\square\]

Computation by teleportation:
Computation by teleportation:

Standard teleportation:

![Teleportation Diagram]

欲：\[ 14\rangle \rightarrow U14\rangle \]

Fact: The Clifford group (generated by CNOT, H, T), and "P = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}

form a universal gate set.

- For \( U \in \text{Clifford} \), \( UZU^\dagger \) and \( UXU^\dagger \) are Paulis.
- For \( U = P \), \( \begin{pmatrix} P & P^+ \\ P^+ & Z \end{pmatrix} \)
  \[ PXP^+ = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \]

Putting it together:

We get universal quantum computation from:

- ability to apply Paulis \( X, Z \)
- preparation of states
  \((I \otimes H)(100\rangle + |111\rangle), (I \otimes T)(100\rangle + |111\rangle)\)
  \(\text{CNOT}_{2,3}(100\rangle + |111\rangle)^{\otimes 2}\)
  \((I \otimes \Phi)(100\rangle + |111\rangle)\)
- Bell basis measurements with adaptive classical control.

Fault-tolerance scheme: Do it at the encoded level...

- Prepare states \(E((I \otimes H)(100\rangle + |111\rangle))\), ...

If an error is detected, throw it away!

- Teleport to compute:

\[
\begin{align*}
E(14\rangle) & \quad \text{perfect} \\
E((I \otimes U)(100\rangle + |111\rangle) & \quad \text{perfect}
\end{align*}
\]

use the Bell measurements to determine the syndrome, decode the logical Bell measurement \(\Rightarrow E(\cup 14\rangle)\)

Question: Is this scheme efficient?
What is the overhead?

\(\Rightarrow\) Erasure error threshold is \(\geq \frac{1}{2}\).

Remark: This is tight. Why? (no-cloning) \(\square\)

Morals:
Morals:
• Detected errors are much nicer than undetected errors
• Large QECCs can be very efficient
• "Ancilla factories": preparing large encoded states is a key problem
  - for initialization, error correction, computation by teleportation
• Decoding efficiency is important
• The overhead is just as important as the noise threshold
  - overhead is highest just below the threshold but drops rapidly with lower noise rates

OTHER FAULT-TOLERANCE SCHEMES

Steane-style error correction

\[ \text{Observe: } \begin{array}{c} 1\rightarrow \text{qubit} \\ \text{has no effect} \end{array} \]
\[ \times 1\rightarrow = 1\rightarrow \]
\[ \frac{1}{\sqrt{2}}(1\rightarrow + 1\rightarrow) \]

\[ \therefore \text{on codewords for a CSS code,} \]
\[ \text{data} \ 1\rightarrow \]
\[ \text{ancilla} \ 1\rightarrow \]
\[ \text{has no logical effect} \]
\[ \text{but it copies } X \text{ errors from the data to the ancilla} \]
\[ \text{measure ancilla (in Z basis) to determine errors} \]

Since it uses transversal gates, this is fault tolerant for any CSS code, provided the encoded ancilla is prepared fault tolerant

(in an ancilla factory... typically you prepare lots of ancillas)
AGP Threshold Theorem & malignant set counting


- The easiest way of lower-bounding your fault-tolerance scheme's noise threshold.

\[ N \text{ error locations, fault tolerant, distance 3} \]
\[ \Rightarrow \text{effective error rate} \leq \binom{N}{2} p^2 \]
\[ \Rightarrow \text{tolerable noise threshold} \geq \frac{1}{\binom{N}{2}} \]

Better lower bound: "Malignant set counting"

not all pairs of locations can cause a logical error, e.g.,

\[ \text{no faults here can possibly cause a logical error} \]
Solve
\[ p = (\text{# malignant pairs}) \cdot p^2 + \binom{N}{2} p^3 \]
to determine the threshold.

For every set of 2 locations, see if \( XX, XY, \ldots \) have 2 errors
can lead to any logical error


<table>
<thead>
<tr>
<th>Code</th>
<th>FTEC</th>
<th>locs.</th>
<th>( \varepsilon_0 \times 10^{-4} )</th>
<th>( \varepsilon_0^{\text{MC}} \times 10^{-4} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steane ([7,1,3])</td>
<td>Steane</td>
<td>575</td>
<td>0.27</td>
<td></td>
</tr>
<tr>
<td>( C_{BS}^{(3)} [[9,1,3]]</td>
<td>Steane</td>
<td>297</td>
<td>1.21</td>
<td>1.21 ± 0.06</td>
</tr>
<tr>
<td>Knill</td>
<td>297</td>
<td>1.26</td>
<td>1.26 ± 0.05</td>
<td></td>
</tr>
<tr>
<td>( C_{BS}^{(5)} [[25,1,5]]</td>
<td>Steane</td>
<td>1,185</td>
<td>1.94</td>
<td>1.92 ± 0.02</td>
</tr>
<tr>
<td>Knill</td>
<td>1,185</td>
<td>2.07 ± 0.03</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Golay ([23,1,7]]</td>
<td>Steane</td>
<td>7,551</td>
<td></td>
<td>≈ 1</td>
</tr>
<tr>
<td>( C_{BS}^{(7)} [[49,1,7]]</td>
<td>Steane</td>
<td>2,681</td>
<td>1.74 ± 0.01</td>
<td></td>
</tr>
<tr>
<td>Knill</td>
<td>2,681</td>
<td>1.91 ± 0.01</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TABLE I:** Rigorous lower bounds on the accuracy threshold, \( \varepsilon_0 \), for adversarial stochastic noise with the concatenated Bacon-Shor code of varying block size and comparison with prior rigorous lower bounds using the concatenated Steane \([7,1,3]\) code [14] and Golay \([23,1,7]\) code [16]. The third column gives the number of locations in the CNOT extended rectangle [14]. The forth column gives exact lower bounds on \( \varepsilon_0 \); the results are obtained using a computer-assisted combinatorial analysis. The fifth column is the Monte-Carlo estimate for \( \varepsilon_0 \) with 1\( \sigma \) uncertainties. Bold fonts indicate the best results in each column.

**Remark:** Can also count malignant triples, etc.,
or can sample random sets to estimate
malignant set counts (especially useful at very low noise rates).

Highest proven threshold: $\sim 10^{-4}$ depolarizing noise per gate

**Threshold Existence Theorems**

1. **Leakage errors**

   $|0\rangle, |1\rangle, |2\rangle, |3\rangle, \ldots$

   Computational space leaks

   Teleportation eliminates leaks (leaks $\rightarrow$ erasure errors)

2. **Geometric locality constraints**

   [Gottesman
   
Problem: Swaps are not fault tolerant
  (get weight-2 errors in code block w/ 1st-order probability)

Solutions:
  - Allow next-nearest neighbor interactions
  
  \[
  \begin{align*}
  \text{A} & \quad \text{swap} \quad \text{swap} \\
  \text{Aux} & \quad - \quad - \\
  \text{B} & \quad \text{swap}
  \end{align*}
  \]

  - Almost - 1D architecture

\[
\ldots \quad \text{data qubits} \\
\quad \text{swap} \\
\ldots \quad \text{aux qubits}
\]

3 All-unitary control

This can be run inside the quantum computer, but then it is not fault tolerant

Trivial solution: Run the above circuit quantumly, but only apply the correction to qubit 1.
  (Then do it all again for qubit 2, etc.)

Moral: Classical control is very helpful,
allows tolerating much more noise.

4. Non-Markovian noise

$$\mathcal{H} = \mathcal{H}_S + \mathcal{H}_B + \mathcal{H}_{SB}$$

- system bath interaction

1. If SB only touches interacting data qubits

$$\mathcal{E} = \max \| H_{SB}(t) \| \cdot t_0$$

2. For noise coupling all data qubits, decaying in space

$$\mathcal{E}_c = \max \sum_i \| H_{iSB} \| \cdot t_0$$


Problem: Hamiltonian norms are not measurable, may be infinite (e.g., harmonic oscillator)


Challenges:

- Rigorous threshold lower bounds are far below simulation-based threshold estimates (especially for the surface code). Which is right?

- Rigorous thresholds for Hamiltonian noise are generally quadratically lower than for stochastic noise.

Is this an artifact of the proofs? Simulations can’t help.

OVERHEAD ANALYSIS OF FAULT-TOLERANCE SCHEMES

What is the overhead? What are the bottlenecks?

See, for example,

Comparing the Overhead of Topological and Concatenated Quantum Error Correction

http://arxiv.org/abs/1312.2316

Martin Suchara, Arvin Faruque, Ching-Yi Lai, Gerardo Paz, Frederic T. Chong, John Kubiatowicz

Table 3. Logical gate count for Shor’s algorithm factoring a 1024-bit number. A conservative estimate of parallelization factors shown.

<table>
<thead>
<tr>
<th>Gate</th>
<th>Occurences</th>
<th>Parallelism</th>
</tr>
</thead>
<tbody>
<tr>
<td>$CNOT$</td>
<td>$1.18 \times 10^9$</td>
<td>1</td>
</tr>
<tr>
<td>$H$</td>
<td>$3.36 \times 10^8$</td>
<td>1</td>
</tr>
<tr>
<td>$T$ or $T^\dagger$</td>
<td>$1.18 \times 10^9$</td>
<td>2.33</td>
</tr>
</tbody>
</table>

Fig. 13. The gate types used in a typical logical circuit and a typical fault-tolerant circuit that uses the Bacon-Shor and Surface codes all differ.
Table 4. The resources needed to factor a 1024-bit number with Shor’s algorithm. Results shown for the surface and Bacon-Shor codes on three technologies.

<table>
<thead>
<tr>
<th>Technology</th>
<th>Neutral</th>
<th>Supercond.</th>
<th>Ion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Atoms</td>
<td>Qubits</td>
<td>Traps</td>
</tr>
<tr>
<td>Gate error</td>
<td>$1 \times 10^{-3}$</td>
<td>$1 \times 10^{-5}$</td>
<td>$1 \times 10^{-9}$</td>
</tr>
<tr>
<td>Avg. gate time</td>
<td>19,000 ns</td>
<td>25 ns</td>
<td>32,000 ns</td>
</tr>
<tr>
<td>Execution time</td>
<td>2.62 years</td>
<td>10.81 hours</td>
<td>2.22 years</td>
</tr>
<tr>
<td>No. qubits</td>
<td>$5.29 \times 10^8$</td>
<td>$4.57 \times 10^7$</td>
<td>$1.44 \times 10^8$</td>
</tr>
<tr>
<td>No. gates</td>
<td>$1.02 \times 10^{21}$</td>
<td>$2.55 \times 10^{19}$</td>
<td>$5.10 \times 10^{19}$</td>
</tr>
<tr>
<td>Dominant gate</td>
<td>CNOT</td>
<td>CNOT</td>
<td>CNOT</td>
</tr>
<tr>
<td>Code distance</td>
<td>17</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Logical gate error</td>
<td>$4.99 \times 10^{-11}$</td>
<td>$2.95 \times 10^{-11}$</td>
<td>$4.92 \times 10^{-15}$</td>
</tr>
<tr>
<td>Logical gate time</td>
<td>$1.29 \times 10^2$ ns</td>
<td>$2.10 \times 10^2$ ns</td>
<td>$5.96 \times 10^5$ ns</td>
</tr>
<tr>
<td>No. qubits per logical</td>
<td>$3.73 \times 10^4$</td>
<td>$3.23 \times 10^3$</td>
<td>$1.16 \times 10^3$</td>
</tr>
<tr>
<td>No. gates per logical</td>
<td>$1.11 \times 10^5$</td>
<td>$9.60 \times 10^3$</td>
<td>$3.46 \times 10^3$</td>
</tr>
</tbody>
</table>

**Surface code** better at high error rates

**Bacon-Shor code** better at low error rates

Fig. 10. Properties of error correction in an abstract quantum technology with physical gate error varying between $1 \times 10^{-10}$ and $1 \times 10^{-2}$. Vertical lines indicate the error correction threshold of the Bacon-Shor and Surface error-correcting codes. The target error rate for a logical operation was chosen to be $1 \times 10^{-10}$.  

2014-2015 USC notes Page 18
Fig. 11. The required concatenation level and code distance of the Bacon Shor and surface codes increase with increasing gate error of the physical technology and decreasing desired logical gate error.
**The Surface Code**

Ben Reichardt

**Intuition:** (scale-invariant superpositions of) **String nets**

1. Start with a surface (2-manifold with boundary)

2. Draw a net on it (cycles or degree 4)
2. Codeword = uniform superposition overall such pictures

Discretization

$|1\rangle = \text{net edge}$
$|0\rangle = \text{none}$

This gives a quantum code!
A. Codewords   B. Protects against   C. Stabilizers
This gives a quantum code:

A. Codewords

B. Protects against errors

C. Stabilizers

\[ \text{encoded}\rangle = \begin{cases} |000\rangle & + \ldots \\ |100\rangle & = \begin{cases} |010\rangle & + \ldots \\ |110\rangle & = \begin{cases} \text{logical } X & \text{on first encoded qubit} \\ \text{meaning } \begin{cases} \times X & \text{along loop} \\ \otimes I & \text{elsewhere} \end{cases} \\ \text{switches first qubit} \\ |10\rangle & \leftrightarrow |11\rangle \end{cases} \end{cases} \]

Protects against errors:

Any region just looks like

\[ \therefore \text{you can't tell if there are an even (101) or odd (111) number of loops around a hole} \]
distance (minimum weight of an operator acting nontrivially on the codespace) = \min \left\{ \text{circumference of a hole}, \text{ distance between holes, or from hole to boundary} \right\}

\textbf{Stabilizers (parity checks satisfied by codewords)}

Rule 1: Even net degree at every vertex

Rule 2: All cycles have equal amplitude

To force $\alpha = \beta$, use the stabilizer this either creates a cycle with the same
A big cycle is created by multiplying the stabilizers for the tiles it encloses.

Observe: The stabilizers are local! (codespace = ground space of local Hamiltonian) — Important for physical implementation

- = code qubits
- = qubits used to measure
- = qubits used to measure

(Observe: Mathematically, the logical operators commute with the stabilizers; this is why they leave the code unchanged.)
Observe: Mathematically, the logical operators commute with the stabilizers; this is why they leave the code unchanged.
\[ P \times 14\rangle = X (P 14\rangle) = X 14\rangle \]

Codespace = ground-space of Hamiltonian
\[ \mathcal{H} = - \sum_{\text{vertices}} \frac{2}{r} - \sum_{\text{tiles}} \]
- all terms commute
- 4-local, and geometrically local in 2D

How to use this code?
1. How to correct errors
2. How to correct errors fault tolerantly?
3. How to compute on the encoded data fault tolerantly?

Errors and error correction

\[ X \text{ error} \]
\[ 2 \times X \text{ errors} \]

\[ X \text{ errors create strings—undetectable in the interior, but detectable at the endpoints} \]

\[ Z \text{ errors can’t be drawn as nets, but are completely symmetrical: chains of } Z \text{ errors (on the dual lattice) show up at their endpoints} \]
Error correction uses minimum-weight matching. For every vertex, consider its parity (should be even) many explanations:

Can correct $X$ and $Z$ errors separately. (I am ignoring some subtleties)

**Fault-tolerant error correction**

How do we explain?

Solution: Repeat "syndrome extraction," and run matching algorithm also in time!

Error chains can grow in space and time, with syndrome flips observed at the endpoints.

Fault-tolerant computation
Most interesting gate: CNOT (entangling)

\[ \begin{array}{c}
\text{a} \\
\text{b}
\end{array} \quad \begin{array}{c}
\text{a} \\
\text{a} \otimes \text{b}
\end{array} \]

**Method 1:** Transversal gates

**Observe:** Sum mod 2 of two valid net diagrams is another valid diagram.

\[ \begin{array}{c}
\text{T} \\
\oplus
\end{array} \quad \begin{array}{c}
\text{L}
\end{array} = \begin{array}{c}
\text{？}
\end{array} \]

**Corollary:** Transversal CNOTs implement encoded CNOT.

(a loop around the top hole will be copied around the bottom hole)

**Method 2:** Code deformation

**Idea:**

Code \xrightarrow{\text{slightly different code (different surface)}} \xrightarrow{\text{Original code but with manipulated codespace}}
Smooth and rough boundary conditions

smooth bdry

rough boundary
= dual of smooth bdry
— allows net lines to terminate

\[ X_1 \]

\[ X_2 \]
first bit (0 or 1) is XORed into 2nd bit
\[ \text{CNOT gate} \]

Typical braiding for CNOT gate, in time

2D Architecture for a quantum computer
Elaborations:

• Surface code on other lattices
• Complexity of min-wt matching: $O(n^3)$ by Edmond
• (non-fault tolerant) Syndrome extraction
• Codewords $1\bar{0}7$ and $1\bar{1}7$ in the computational basis
• Obtaining a universal gate set