How about quantum computing?

Bert de Jong
wadejong@lbl.gov
What makes quantum computing so exciting?

• Speedups over classical computing

• “Unbreakable” encryption protocols

• Quantum simulation

• Efficient optimization algorithms
Why is a computational chemist like me interested in QC?

Understanding → Control
The underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are thus completely known, and the difficulty is only that the exact application of these laws leads to equations much too complicated to be soluble.

Paul Dirac
Exaflop gives us only a factor of 10x ... we need a lot more
Quantum chemistry on quantum computers

Inaccessible, even at exascale! Quantum computer requires ~100 ideal qubits for solution

Nitrogenase enzyme

\[
N_2 + 3 H_2 \rightarrow 2 NH_3
\]

“FeMoco”

From Galli, University of Chicago

Photo-induced catalysis of water

Nature’s answer to Haber Process
Behold the power of quantum computers

• $2^n$ complex coefficients describe the state of a composite quantum system with $n$ qubits

• 100 qubits = $2^{100}$ states

• Quickly reaches number of particles in the universe
How do you get into quantum computing?

You need to learn some physics (quantum mechanics) if you want to do quantum computing.
Ingredients to make a quantum computer work

- Qubit
- Superposition
- Entanglement
- Interference

This is the one that makes it “Quantum.” The rest is just math.
You’ll need to know some linear algebra…

Qubit state is represented as a two-dimensional state space in $\mathbb{C}^2$ with orthonormal basis vectors

State $\rightarrow$ wave function $\rightarrow$ $|\psi\rangle = a |0\rangle + b |1\rangle \rightarrow a$ and $b$ are complex

$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ are computational basis

$|\psi\rangle = a |0\rangle + b |1\rangle = \begin{bmatrix} a \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ b \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix}$ with $|a|^2 + |b|^2 = 1$
Tensor products key for multiple qubits

• Notation for two qubits

\[ |0\rangle |0\rangle = |00\rangle \]

\[ |\psi\rangle = a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \]

• Tensor products

\[ |\psi\rangle = (a_1 |0\rangle + b_1 |1\rangle) \otimes (a_2 |0\rangle + b_2 |1\rangle) = \begin{bmatrix} a_1 a_2 \\ b_1 a_2 \\ a_1 b_2 \\ b_1 b_2 \end{bmatrix} \]
What’s the difference between a classical and quantum bit?

State = OFF
State = ON

State = a*OFF + b*ON

\[ |\psi\rangle = a |0\rangle + b |1\rangle \]
Qubits are represented on a Bloch Sphere

\[ |\psi\rangle = a|0\rangle + b|1\rangle \]

- **Coefficients** \(a\) and \(b\) are complex numbers
  
  0 with probability \(|a|^2\)
  
  1 with probability \(|b|^2\)

- **So, it’s not a probability on a number line**
Superposition, or being both in 0 and 1 at the same time...

Classical

Bits represents a *single value*, out of $2^N$ possible bit strings, e.g. $000 == 0$

Quantum

Bits represent an *ensemble* of all $2^N$ possible bit strings, from which you can sample, e.g. for 3 qubits:

$$|000\rangle$$
$$|001\rangle$$
$$|010\rangle$$
$$|011\rangle$$
$$|100\rangle$$
$$|101\rangle$$
$$|110\rangle$$
$$|111\rangle$$

*Increases “working memory” up to an exponential factor.*
Schrödinger’s Cat: Dead or Alive

You can only MEASURE either dead or alive, not both
Measuring a quantum bit

State = a*OFF + b*ON

Measure many times (sample)

You only measure either ON or OFF each time with probability equal to a and b squared

On average

\[ a = \frac{7000}{10000} \quad \text{and} \quad b = \frac{0.46}{10000} \]

State = OFF

State = ON

State = OFF
Operating on a qubit = Matrix-Vector operations

\[ R_x(\phi) = \begin{bmatrix} \cos(\frac{\phi}{2}) & -i \sin(\frac{\phi}{2}) \\ -i \sin(\frac{\phi}{2}) & \cos(\frac{\phi}{2}) \end{bmatrix} \]

\[ R_y(\phi) = \begin{bmatrix} \cos(\frac{\phi}{2}) & -\sin(\frac{\phi}{2}) \\ \sin(\frac{\phi}{2}) & \cos(\frac{\phi}{2}) \end{bmatrix} \]

\[ R_z(\phi) = \begin{bmatrix} e^{-i\frac{\phi}{2}} & 0 \\ 0 & e^{i\frac{\phi}{2}} \end{bmatrix} \]

Rotations around an axis

Pauli-X
\[ X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \]

Pauli-Y
\[ Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \]

Pauli-Z
\[ Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \]

Pauli matrices

\[ |\psi\rangle \alpha |0\rangle + |\psi\rangle \beta |1\rangle = \frac{1}{\sqrt{2}} |\psi\rangle 0 + \frac{1}{\sqrt{2}} |\psi\rangle 1 \]

Hadamard is special gate sauce

\[ \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \]
Entanglement, or making qubits interconnected

- Unifies multiple qubits into a single state
  
  Example (maximum entanglement):
  \[ |\psi\rangle = (|00\rangle + |11\rangle)/\sqrt{2} \]
  \[ \Rightarrow \text{measuring one qubit determines state of the other} \]

- A “physical” resource
  - Can be “added”, “removed”, used, and quantified (entanglement entropy)

- Allows “instantaneous” operation on all qubits
  - Popular: with superposition, “try all solutions in parallel”
  - Mathematically: off-diagonal elements in \(2^N\times2^N\) state matrix

\[\text{Increases information density up to an exponential factor.}\]
Einstein called it “spooky action at a distance”
Math of entanglement

• Not entangled means you can separate information of qubits

\[
\begin{bmatrix}
    a \\
    b \\
    c \\
    d
\end{bmatrix} = \begin{bmatrix}
    a_1 a_2 \\
    b_1 a_2 \\
    a_1 b_2 \\
    b_1 b_2
\end{bmatrix} \quad \Rightarrow \quad ad = bc = a_1 a_2 b_1 b_2
\]

• Effectively you can write the combined state as a tensor product of two Hilbert spaces

• If \( ad \neq bc \) we call the qubits entangled
How do we entangle two qubits?

Controlled-NOT (CNOT)

- Change state of second qubit is controlled by first qubit

\[
\begin{align*}
|00\rangle & \Rightarrow |00\rangle \\
|01\rangle & \Rightarrow |01\rangle \\
|10\rangle & \Rightarrow |11\rangle \\
|11\rangle & \Rightarrow |10\rangle \\
|AB\rangle & \Rightarrow |A(A\oplus B)\rangle \text{ or addition mod}(2)
\end{align*}
\]
Interference

- **Total probability over all bit strings sums to one**
  - Combined effect of superposition and entanglement
    \[ \Rightarrow \text{As one solution becomes more likely (larger amplitude), others have to become less likely (lower amplitude)}. \]

- **Amplify right solution, suppress others**
  - Physics: wave mechanics
  - Popular: music/orchestra
  - Mathematics: complex (\(\mathbb{C}\)) math

Example, Shor factorization: 3 and 5 fit a whole number of times in 15 \(\Rightarrow\) “standing waves”, others interfere destructively.

*Interference is how quantum algorithms are designed to work.*
More ”spooky action”, moving 2 bits with 1 qubit

• Moving 2 bits of information with 1 qubit only

\[ |\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \]

• Bob does a CNOT followed by Hadamard on Alice’s qubit

• Resulting state will be one of fours possible states
Quantum computing hardware technologies

Electrons: Spin up + Spin down

Majorana quasi-particle

Superconducting atoms

Solid state (spins)

D-Wave
### Strongest contenders ... at least right now

**Superconducting Qubits**  
(Transmon, flux, phase)  
- Qubit – Josephson junctions + capacitors  
- Information encoded by superconductor charge  
- Controlled by microwave  
- Dilution fridge required  
- Gates: rotations, CNOT, CZ  

**Trapped Ion Qubits/Qudits**  
(Hyperfine, optical)  
- Qubit – ion (Ca, Yb) trapped in vacuum  
- Information encoded in energy levels  
- Controlled by laser  
- Room temperature  
- Gates: Alltoall, Ising, phase shift  

<table>
<thead>
<tr>
<th>Technology</th>
<th>Qubit Description</th>
<th>Information Encoding</th>
<th>Control Method</th>
<th>Temperature</th>
<th>Gates Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Superconducting</td>
<td>Josephson junctions + capacitors</td>
<td>Superconductor charge</td>
<td>Microwave</td>
<td>Low</td>
<td>Rotations, CNOT, CZ</td>
</tr>
<tr>
<td>Trapped Ion</td>
<td>Ion (Ca, Yb) trapped in vacuum</td>
<td>Energy levels</td>
<td>Laser</td>
<td>Room</td>
<td>Alltoall, Ising, phase</td>
</tr>
</tbody>
</table>

- **T2:** ~100μs  
  Gate: ~10ns  
  T2: ~1s  
  Gate: ~μs

**Commercially viable technologies, fully explored**  
- Superconducting deemed as scalable  
- Ions deemed less noisy (T₂), room temp
What does a SC qubit computer look like?

IBM System One
What about the DWave annealer...

• In essence superconducting qubits
  – Adiabatic quantum computer
  – Thousands of bits

• Debate on quantumness still raging

• Good for very specific problems
  – Optimization
  – Graph problems
Many challenges with quantum hardware

- # of good qubits not yet enough for quantum supremacy/science
- Diverse technologies, each with its own instruction set
- Coherence (available compute time) very short (10s-100s of ops)
- Noise and errors still pretty large
For example, gate sets in superconducting chips

- Each chip has own native gate set
  - Single qubit, usually rotations, and Hadamard
  - Two-qubit, usually CNOT, CZ (Google), SWAP

- Each chip has a constrained topology
  - Ring, array, mesh, bow-tie

- Compilers needed to translate gate sets, do mapping
• **Right now quantum computing is still a *physics experiment***
  
  — Noise is everywhere
  
  — Measurement errors
Qubit errors due to relaxation and decoherence

\( T_1: \) relaxation, dampening

- Environment exchanges energy with the qubit, mixing the two states by stimulated emission or absorption
- Important during read-out

- Intuitively time to decay from \( |1\rangle \) to \( |0\rangle \)

\( T_1 > T_2 \)

\( \psi \) = \( \cos(\theta/2)|0\rangle + e^{i\phi}\sin(\theta/2)|1\rangle \)

These are not cut-off times, but “half-lives.”
Decay is **continuous.**

\( T_2: \) dephasing

- Environment creates loss of phase memory by smearing energy levels, changing phase velocity
- Important during “computation”, bounds circuit depth (number of consecutive gates)

- Intuitively time for \( \phi \) to get imprecise
How can we correct for quantum errors?

- **Quantum computing is** analog
  - Sensitive to noise: no projection to 0 or 1 as in digital
  - *All states are valid*: can not detect noisy results

- **Use group theory: algebra over logical qubits**
  - Use multiple qubits to represent states
  - Errors fall outside the group and can be detected
  - Stabilizer codes map errors back onto the group
  - Will require 1000s of qubits: not near-term

**Example (3-bit flip code):**

\[
\begin{align*}
|0\rangle & \rightarrow |0_L\rangle \equiv |000\rangle \\
|1\rangle & \rightarrow |1_L\rangle \equiv |111\rangle \\
|101\rangle & \rightarrow |111\rangle
\end{align*}
\]

States are continuous and all are valid

*Single bit-flip leads to detectable (and correctable) state:*
Reducing stochastic noise in quantum operations

Adding error on purpose

Converting non-stochastic to stochastic (randomized benchmarking)

Quick and dirty on correcting of measurement errors

One qubit measurement (IBMQX4):

|0> {'00000': 7904, '00001': 197, '00010': 85, '00011': 6}
|1> {'00000': 800, '00001': 7285, '00010': 11, '00011': 96}

Two qubit measurement

|00> {'00000': 7909, '00001': 191, '00010': 89, '00011': 3}
|01> {'00000': 707, '00001': 7382, '00010': 8, '00011': 95}
|10> {'00000': 585, '00001': 19, '00010': 7409, '00011': 179}
|11> {'00000': 66, '00001': 507, '00010': 686, '00011': 6933}

Correction with covariance matrices, disentangling confusion
We can also build error detection/correction into circuits.

Single error detection → Magic states

![Quantum circuit diagram]
All qubits are equal, but some qubits are more equal than others

IBMQ Tokyo Hardware

Circuit

Topology

Qubit mapping histogram for the 6-qubit circuit (258 mappings, +1 is the ideal output)
What does a quantum algorithm look like?

**Physically**

N input qubits

Sequence of physical manipulations of the N qubits

N output qubits

**Conceptually**

Probability distribution over $2^N$ binary classical states

Sequence of quantum gates

Seek to maximize probability of good solutions

Single result with probability (amplitude)$^2$

Sequence includes “state preparation” to get from the computational “null” state to the desired initial/input state.

Measurement includes gates to go from computational to measurement bases.
Two common algorithms for quantum simulations

Quantum Phase Estimation (QPE)

Prepare, evolve, FT and measure to find eigenvalue for eigenvector

Variational Quantum Eigensolver (VQE)

\[ H = \sum_{i} g_{i}^{\alpha} \langle \sigma_{i}^{\alpha} \rangle + \frac{1}{2} \sum_{i,j} g_{ij}^{\alpha \beta} \langle \sigma_{i}^{\alpha} \sigma_{j}^{\beta} \rangle + \ldots \]

Only prepare and measure, do the rest classically
Adiabatic quantum computing algorithm

Put quantum system in lowest-energy configuration in a way that's easy to do.

Evolve the quantum system in a way that keeps it in its lowest-energy configuration throughout.

Readout success of final state most probable for evolutions that are close to "adiabatic".

$H_0 = \sum_{i=1}^{n} \sigma_x^{(i)}$  
$|\psi_0\rangle = \frac{1}{2^{n/2}} \sum_{i=1}^{2^n} |i\rangle$

$H(s) = (1-s)H_0 + sH_1$

$H_1 = h_0 I + \sum_{i=1}^{n} h_i \sigma_z^{(i)} + \sum_{i,j=1}^{n} J_{ij} \sigma_z^{(i)} \otimes \sigma_z^{(j)}$
How do we program quantum computers?

**Circuit Model**
- Diagrams of wires (qubits) and gates (logical operations, applied in order)
- Write by hand or generated with science domain software (e.g. OpenFermion)
- Hard to generate optimally

**Unitary Linear Algebra**
- Matrices (operations) and vectors (state)
- Often more natural to science domain (e.g. coupling strengths)
- Hard to decompose: $2^N \times 2^N$ in size, with $N$ the number of qubits

\[
\begin{pmatrix}
1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 \\
1 & -1 & -1 & 1 \\
1 & 1 & -1 & -1
\end{pmatrix}
\begin{pmatrix}
a_1 b_1 \\
a_1 b_2 \\
a_2 b_1 \\
a_2 b_2
\end{pmatrix}
\]

Representations are equivalent, can go back and forth, and even mix.
A whole system software stack is needed

**Scientist**

Initial Quantum Algorithm

- High level interface
  - Arbitrary gates, qubit reset, feedback, measurement
  - Algorithm specified in any gate set

Compiled Quantum Algorithm

- Translate to processor
  - Arbitrary gates compiled into available gate set
  - Processor connectivity and timing constraints enforced

Pulses output by AWG

- Translation to hardware
  - Define pulse parameters (shape, phase, sequence)
  - Reset/feedback code applied by FPGAs

Courtesy of Irfan Siddiqi
An incomplete list of software tools

- **Frameworks from most chip providers**

<table>
<thead>
<tr>
<th>Provider</th>
<th>Framework</th>
<th>License</th>
<th>Cloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM</td>
<td>QisKit</td>
<td>Minor restrictions</td>
<td>IBM Q-Experience</td>
</tr>
<tr>
<td>Google</td>
<td>Cirq</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>Rigetti</td>
<td>Forest / PyQuil</td>
<td>Restrictive</td>
<td>Rigetti QCS (beta)</td>
</tr>
<tr>
<td>Microsoft</td>
<td>LiQUi&gt; / Q#</td>
<td>Minor restrictions</td>
<td></td>
</tr>
<tr>
<td>D-Wave</td>
<td>qbsolv</td>
<td>Minor restrictions</td>
<td>D-Wave Leap</td>
</tr>
</tbody>
</table>

- **Academia & startups target the above**
  - E.g. PyTKET (Cambridge Quantum), ProjectQ (ETH Zürich)
  - QuTiP (Academia, also RIKEN; [http://qutip.org](http://qutip.org))
And if you know Python, it’s not that scary...

```python
from qiskit import *
from qiskit.compiler import transpile, assemble

qr = QuantumRegister(3)
cr = ClassicalRegister(3)
circuit = QuantumCircuit(qr, cr)
circuit.x(qr[0])
circuit.cx(qr[0], qr[1])
circuit.measure(qr, cr)

qobj = assemble(transpile(circuit, backend=backend), shots=1024)
job = backend.run(qobj)
counts = job.result().get_counts()
print(counts)
```
How good is a quantum computer?: Let’s look at $\text{H}_2$

$\text{H}_2$ molecule on 2 qubits with minimal basis
Towards useful quantum computing for science

Hardware technology

- Increasing qubit count
- Increasing lifetimes
- Increasing fidelity and reducing errors

Scientific algorithms and software

- Reducing qubit count
- Decreasing operation counts
- Incorporating error resiliency
Acknowledgements

This work was supported by the DOE Office of Advanced Scientific Computing Research (ASCR) through the Quantum Algorithm Team and Quantum Testbed Pathfinder Program.

This research used computing resources of the Oak Ridge Leadership Computing Facility through the INCITE program and the National Energy Research Scientific Computing Center.
Study Resources

- Nielsen & Chuang “Quantum Computation and Quantum Information”
  - Complete, lots of material, better for physicists
- Nielsen’s “Quantum computing for the Determined”
  [Link](https://www.youtube.com/playlist?list=PL1826E60FD05B44E4)
- Rieffel & Polak, “A Gentle Introduction”
  - Targeted at computer scientists and mathematicians
- John’s Preskill’s lecture notes
  [Link](http://www.theory.caltech.edu/~preskill/ph219/ph219_2017)
- Todd Brun’s lecture notes (insightful)
  [Link](https://www-bcf.usc.edu/~tbrun/Course/)
- Interactive circuit simulator
  [Link](http://algassert.com/quirk)

Conferences: [Link](http://quantum.info/conf/2019.html)
Papers: [Link](https://arxiv.org)