Quantum Computing
The Why and How

Jonathan Baker
Argonne National Lab
7-29-2019
Outline for this Talk (Pt 1.)

1. Why Quantum Computing? What is the current state of quantum computing and what are current challenges? What are algorithms in this paradigm ideal for?

2. What is different between quantum and classical information? What makes a quantum computer that much different than a classical one?

3. How can we use a quantum computer to solve concrete problems? Can we do better than classical computers at the same tasks?
Why Quantum Computing?

- Fundamentally change what is computable (in a reasonable amount of time)
  - The only known means to potentially scale computation exponentially with the number of devices
  - We can do this by taking advantage of quantum mechanical phenomenon
- Solve currently intractable problems in chemistry, simulation, and optimization
- Moore’s Law is ending - quantum computing can act as a replacement in some scientific domains to help continue scaling applications
- Insights in classical computing
  - Many classical algorithms are “quantum-inspired”, e.g. in chemistry physics or cryptography
  - Challenges classical algorithms to compete with quantum algorithms
Current State of Quantum Computing: NISQ

- Noisy-Intermediate Scale Quantum
  - 10s to 100s of qubits
  - Moderate error rates
  - Limited connectivity
  - No error correction

IBM
50 Superconducting Qubits

Rigetti
20 Superconducting Qubits

Google
72 superconducting qubits
The Algorithms to Machines Gap

- Grover’s Algorithm (Database search)
- Shor’s Algorithm (Integer Factorization)

# Qubits Needed

# Qubits Buildable
The Algorithms to Machines Gap

<table>
<thead>
<tr>
<th>Year</th>
<th>#Qubits Needed</th>
<th># Qubits Buildable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1995</td>
<td>100,000</td>
<td>10</td>
</tr>
<tr>
<td>2000</td>
<td>1,000,000</td>
<td>100</td>
</tr>
<tr>
<td>2005</td>
<td>10,000,000</td>
<td>1,000</td>
</tr>
<tr>
<td>2010</td>
<td>100,000,000</td>
<td>10,000</td>
</tr>
<tr>
<td>2015</td>
<td>1,000,000,000</td>
<td>100,000</td>
</tr>
<tr>
<td>2020</td>
<td>10,000,000,000</td>
<td>1,000,000</td>
</tr>
<tr>
<td>2025</td>
<td>100,000,000,000</td>
<td>100,000,000</td>
</tr>
</tbody>
</table>

- Grover's Algorithm (Database search)
- Shor's Algorithm (Integer Factorization)

GAP!
The Algorithms to Machines Gap

- Grover’s Algorithm (Database search)
- Shor’s Algorithm (Integer Factorization)

# Qubits Needed vs. # Qubits Buildable

Q. Sim, Q Chem, QAOA
Closing the Gap: Software-Enabled Vertical Integration and Co-Design

Grover’s Algorithm (Database search)
Shor’s Algorithm (Integer Factorization)
Q. Sim, Q. Chem, QAOA

# Qubits Needed
# Qubits Buildable

GAP!
Result: Crossover by 2023!

Develop co-designed algorithms, SW, and HW to close the gap between algorithms and devices by 100-1000X, accelerating QC by 10-20 years.
Space-Time Product Limits

Error rates of quantum operations limit what we can accomplish.
Space-Time Product Limits

Error rates of quantum operations limit what we can accomplish

Gate Error $\sim 10^{-5}$
“Good” Quantum Algorithms

- Compact problem representation
  - Functions, small molecules, small graphs
- High complexity computation
- Compact solution
- Easily-verifiable solution
- Co-processing with classical supercomputers
- Can exploit a small number of quantum kernels
Introduction to the Basics
One Qubit

\[ |0\rangle \]

\[ |1\rangle \]
One Qubit

$|0\rangle$

$|1\rangle$

$|0\rangle + |1\rangle$
One Qubit

when we try to “see” the state, we can only see 0 or 1.
50% of the time we see 0.
50% of the time we see 1.
Identically prepared qubits can still behave randomly!

This randomness is inherent in nature, and not a limitation of our observation.
Multiple Qubits

\[ (1 + j) |0\rangle + |1\rangle \]

\[ |0\rangle + |1\rangle \]

\[ (-2 + j) |0\rangle + (-5j) |1\rangle \]
The Power of Quantum Information
Why simulating quantum systems becomes intractable quickly

$$a |000\rangle + b |001\rangle + c |010\rangle + d |011\rangle + e |100\rangle + f |101\rangle + g |110\rangle + h |111\rangle$$
The Power of Quantum Information

Why simulating quantum systems becomes intractable quickly

\[ a |000\rangle + b |001\rangle + c |010\rangle + d |011\rangle + e |100\rangle + f |101\rangle + g |110\rangle + h |111\rangle \]

- The state of \( n \) qubits is described by \( 2^n \) coefficients.
- Adding one qubit doubles the dimension.
- This is known as superposition.
The Power of Quantum Information

Why simulating quantum systems becomes intractable quickly

classical
probabilistic
bit
The Power of Quantum Information
Why simulating quantum systems becomes intractable quickly

\[(a0 + b1)\]

classical probabilistic bit
The Power of Quantum Information

Why simulating quantum systems becomes intractable quickly

\[(a_0 + b_1)(c_0 + d_1)\] classical
probabilistic
bit
The Power of Quantum Information
Why simulating quantum systems becomes intractable quickly

\[(a_0 + b_1) \cdot (c_0 + d_1) \cdot (e_0 + f_1)\]
The Power of Quantum Information

Why simulating quantum systems becomes intractable quickly

Quantum bit
(qubit)
The Power of Quantum Information
Why simulating quantum systems becomes intractable quickly

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

Quantum bit (qubit)
The Power of Quantum Information

Why simulating quantum systems becomes intractable quickly

\[ a \left| 0 \rightangle + b \left| 1 \rightangle \]

\[ a \left| 00 \rightangle + b \left| 01 \rightangle + c \left| 10 \rightangle + d \left| 11 \rightangle \]

Quantum bit (qubit)
The Power of Quantum Information
Why simulating quantum systems becomes intractable quickly

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

Quantum bit (qubit)

\[ a |00\rangle + b |01\rangle + c |10\rangle + d |11\rangle \]

\[ a |000\rangle + b |001\rangle + c |010\rangle + d |011\rangle + e |100\rangle + f |101\rangle + g |110\rangle + h |111\rangle \]
The Power of Quantum Information

Why simulating quantum systems becomes intractable quickly

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

Quantum bit (qubit)

\[ a |00\rangle + b |01\rangle + c |10\rangle + d |11\rangle \]

\[ a |000\rangle + b |001\rangle + c |010\rangle + d |011\rangle + e |100\rangle + f |101\rangle + g |110\rangle + h |111\rangle \]

The state of an n-qubit system cannot (in general) be written as the state of its individual components.
This is known as entanglement.
Quantum Information Processing
Using Vectors Matrices and Projections
Quantum Information Processing
Using Vectors Matrices and Projections

\[ a \ket{000} + b \ket{001} + c \ket{010} + d \ket{011} + e \ket{100} + f \ket{101} + g \ket{110} + h \ket{111} \]
Quantum Information Processing
Using Vectors, Matrices and Projections

\[ a \left| 000 \right\rangle + b \left| 001 \right\rangle + c \left| 010 \right\rangle + d \left| 011 \right\rangle + e \left| 100 \right\rangle + f \left| 101 \right\rangle + g \left| 110 \right\rangle + h \left| 111 \right\rangle \]
Quantum Information Processing
Using Vectors Matrices and Projections

\[ a |000\rangle + b |001\rangle + c |010\rangle + d |011\rangle + e |100\rangle + f |101\rangle + g |110\rangle + h |111\rangle \]

\[
\begin{pmatrix}
g_{00} & g_{01} & g_{02} & g_{03} & g_{04} & g_{05} & g_{06} & g_{07} \\
g_{10} & g_{11} & g_{12} & g_{13} & g_{14} & g_{15} & g_{16} & g_{17} \\
g_{20} & g_{21} & g_{22} & g_{23} & g_{24} & g_{25} & g_{26} & g_{27} \\
g_{30} & g_{31} & g_{32} & g_{33} & g_{34} & g_{35} & g_{36} & g_{37} \\
g_{40} & g_{41} & g_{42} & g_{43} & g_{44} & g_{45} & g_{46} & g_{47} \\
g_{50} & g_{51} & g_{52} & g_{53} & g_{54} & g_{55} & g_{56} & g_{57} \\
g_{60} & g_{61} & g_{62} & g_{63} & g_{64} & g_{65} & g_{66} & g_{67} \\
g_{70} & g_{71} & g_{72} & g_{73} & g_{74} & g_{75} & g_{76} & g_{77}
\end{pmatrix}
\begin{pmatrix}
a \\
b \\
c \\
d \\
e \\
f \\
g \\
h
\end{pmatrix}
\]
Quantum Information Processing
Using Vectors Matrices and Projections

\[ a \ket{000} + b \ket{001} + c \ket{010} + d \ket{011} + e \ket{100} + f \ket{101} + g \ket{110} + h \ket{111} \]
Quantum Information Processing
Using Vectors Matrices and Projections

\[ a \ket{000} + b \ket{001} + c \ket{010} + d \ket{011} + e \ket{100} + f \ket{101} + g \ket{110} + h \ket{111} \]

Important Result:
1-qubit & 2-qubit gates (i.e. local operations) are sufficient for universal computation [Barenco et al. 95].
Example Quantum Gates: The Hadamard Gate
Example Quantum Gates: The Hadamard Gate

\[ |0\rangle \xrightarrow{H} \]

H Creates Superposition!
Example Quantum Gates: The Hadamard Gate

\[ |0\rangle \xrightarrow{H} |\rangle \]

H Creates Superposition!

\[ |0\rangle + |1\rangle \]
Example Quantum Gates: The Hadamard Gate

\[ |0\rangle - \begin{array}{c} H \end{array} \]

\[ |0\rangle - \begin{array}{c} H \end{array} \]

\[ |0\rangle + |01\rangle + |10\rangle + |11\rangle \]
Example Quantum Gates: The Hadamard Gate

\[ |0\rangle - \boxed{H} - |0\rangle \]

\[ |0\rangle - \boxed{H} - |0\rangle \]

\[ |0\rangle - \boxed{H} - |0\rangle \]

\[ |000\rangle + |001\rangle + |010\rangle + |011\rangle + |100\rangle + |101\rangle + |110\rangle + |111\rangle \]
Example Quantum Gates: The Hadamard Gate

\[ |0\rangle \xrightarrow{H} |0\rangle + |1\rangle \xrightarrow{H} |0\rangle \]
Example Quantum Gates: The Hadamard Gate

\[ |1\rangle \xrightarrow{H} |0\rangle - |1\rangle \xrightarrow{H} |1\rangle \]
Example Quantum Gates: The Controlled-NOT Gate
The Quantum “If” (CNOT)
Example Quantum Gates: The Controlled-NOT Gate

The Quantum “If” (CNOT)

\[
\begin{align*}
|0\rangle & \quad \quad |0\rangle \\
\alpha |0\rangle + \beta |1\rangle & \quad \quad \quad \quad \alpha |0\rangle + \beta |1\rangle \\
|1\rangle & \quad \quad |1\rangle \\
\alpha |0\rangle + \beta |1\rangle & \quad \quad \beta |0\rangle + \alpha |1\rangle
\end{align*}
\]

A CNOT flips the target bit if the control bit is 1.
Quantum Algorithm: Bernstein-Vazirani

\[ x_{n-1} \quad \ldots \quad x_1 \quad x_0 \]
Quantum Algorithm: Bernstein-Vazirani

\[ x_{n-1} \quad \ldots \quad x_1 \quad x_0 \]

\[
\text{THE ORACLE}
\]

\[(s_{n-1} \quad \ldots \quad s_1 \quad s_0)\]
Quantum Algorithm: Bernstein-Vazirani

\[ x_{n-1} \quad \ldots \quad x_1 \quad x_0 \]

\[ \text{THE ORACLE} \]

\[ (s_{n-1} \quad \ldots \quad s_1 \quad s_0) \]

\[ x_{n-1}s_{n-1} \oplus \ldots \oplus x_1s_1 \oplus x_0s_0 \]
Quantum Algorithm: Bernstein-Vazirani

Classically, we need \( n \) tries.

Optimal classical strategy:

\[
\begin{align*}
X &= 1 0 \ldots 0 0 \ (2^{n-1}) \\
X &= 0 1 \ldots 0 0 \ (2^{n-2}) \\
\vdots & \quad \vdots \\
X &= 0 0 \ldots 1 0 \ (2) \\
X &= 0 0 \ldots 0 1 \ (1)
\end{align*}
\]

Quantumly, we need 1 try.
Quantum Algorithm: Bernstein-Vazirani
Classical Vs. Quantum Oracle

$x \rightarrow$ The Oracle $\rightarrow x \cdot s$

Classical Dot-Product Oracle
Quantum Algorithm: Bernstein-Vazirani

Classical Vs. Quantum Oracle

Classical Dot-Product Oracle

Quantum Dot-Product Oracle

Difference? Must be reversible!
Quantum Algorithm: Bernstein-Vazirani

Implementing the Oracle

The control pattern for the oracle depends on the hidden bitstring.
Quantum Algorithm: Bernstein-Vazirani
Implementing the Oracle

The control pattern for the oracle depends on the hidden bitstring.

\[ S = 0101 \]

\[
\begin{align*}
|x_0\rangle & \quad |x_1\rangle & \quad |x_2\rangle & \quad |x_3\rangle & \quad |t m p\rangle \\
& |x_3.s_3 \oplus x_2.s_2 \oplus x_1.s_1 \oplus x_0.s_0 \oplus t m p\rangle
\end{align*}
\]
Quantum Algorithm: Bernstein-Vazirani
Implementing the Oracle

The control pattern for the oracle depends on the hidden bitstring.

\[ S = 1111 \]

\[
\begin{align*}
\lvert x_0 \rangle & \rightarrow \bullet \\
\lvert x_1 \rangle & \rightarrow \bullet \\
\lvert x_2 \rangle & \\
\lvert x_3 \rangle & \\
\lvert \text{tmp} \rangle & \rightarrow \bullet \\
\end{align*}
\]

\[
\lvert x_3.s_3 \oplus x_2.s_2 \oplus x_1.s_1 \oplus x_0.s_0 \oplus \text{tmp} \rangle
\]
Quantum Algorithm: Bernstein-Vazirani
The Key Trick - Phase Kickback
Quantum Algorithm: Bernstein-Vazirani

The Key Trick - Phase Kickback

\[ (|0\rangle + |1\rangle)(|0\rangle - |1\rangle) = |00\rangle - |01\rangle + |10\rangle - |11\rangle \]
Quantum Algorithm: Bernstein-Vazirani

The Key Trick - Phase Kickback

\[ |0\rangle - |0\rangle - |1\rangle + |1\rangle = (|0\rangle - |1\rangle)(|0\rangle - |1\rangle) \]

Phase kickback!
Quantum Algorithm: Bernstein-Vazirani
The Key Trick - Phase Kickback

\[(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) = |00\rangle - |01\rangle + |10\rangle - |11\rangle\]

|0\rangle
\[H\]
|1\rangle
\[H\]
|1\rangle
\[H\]

Phase kickback!
Quantum Algorithm: Bernstein-Vazirani
Putting it all together
Quantum Algorithm: Bernstein-Vazirani
Putting it all together

\[
\begin{align*}
|q_0\rangle & \quad H \quad H \\
|q_1\rangle & \quad H \quad H \\
|q_2\rangle & \quad H \quad H \\
|q_3\rangle & \quad H \quad H \\
|\text{tmp}_0\rangle & \quad X \quad H
\end{align*}
\]
Quantum Algorithm: Bernstein-Vazirani

Putting it all together

\[ |q_0\rangle \quad |q_1\rangle \quad |q_2\rangle \quad |q_3\rangle \quad |\text{tmp}_0\rangle \]

\[ result_0 \quad result_1 \quad result_2 \quad result_3 \]
Quantum Algorithm: Bernstein-Vazirani

Putting it all together

Wherever there’s CNOT, phase kickback puts that control qubit in state $|1\rangle$. 
Quantum Algorithm: Bernstein-Vazirani
Why did it work?

1. Classical oracles can only be queried with a single number at a time. Quantum oracles can be queried in superposition.

2. We don’t just “try every answer simultaneously”. The problem had a structure that we could exploit using qubits: encode information in phases