

JANUARY 2018

- Agenda
 - Introductions and thanks to Microsoft
 - Quantum News
 - Food/Pizza
 - Quantum Computing refresher
- Presentations can be found at github.com/NYCQuantumComputing
- Twitter @NYCQuantum
- Looking for hosts, presenters, topics, suggestions



RECAP 2017

- March 2017 – Kickoff
- April 2017 – Grover search, IBM's Quantum Experience, math behind Grover
- May 2017 – Thanks to DWAVE for their technical presentation
- June 2017 – Thanks to Chris Monroe from IONQJ
- July 2017 – Quantum Entanglement
- September 2017- Bell's Inequality
- October 2017 – IBM presented QISKIT
- November 2017 – Nathan Weiбе from Microsoft
- December 2018 – Shor Discussion

EDX CLASS STARTING JANUARY 15, 2018

The screenshot shows the edX course page for "Quantum Information Science I, Part 1". The page features a large image of quantum states $|\hbar\rangle + |\psi\rangle$. The course title is "Quantum Information Science I, Part 1". A brief description states: "Want to learn about quantum bits, quantum logic gates, quantum algorithms, and quantum communications, and know some linear algebra but haven't yet learned much about quantum mechanics? This is the course for you!" The course is offered by MIT. The "Enroll Now" button is highlighted in green. To the right, there is a sidebar with course details:

Length:	5 weeks
Effort:	11 to 13 hours per week
Price:	FREE Add a Verified Certificate for \$49 USD
Institution:	MITx
Subject:	Computer Science
Level:	Intermediate
Language:	English
Video Transcripts:	English

Join Now
Started on January 15, 2018
Enroll Now
 I would like to receive email from Massachusetts Institute of Technology and learn about other offerings related to Quantum Information Science I, Part 1.

About this course
This course is part of a three-course series that provides an introduction to the theory and practice of quantum computation. We cover:

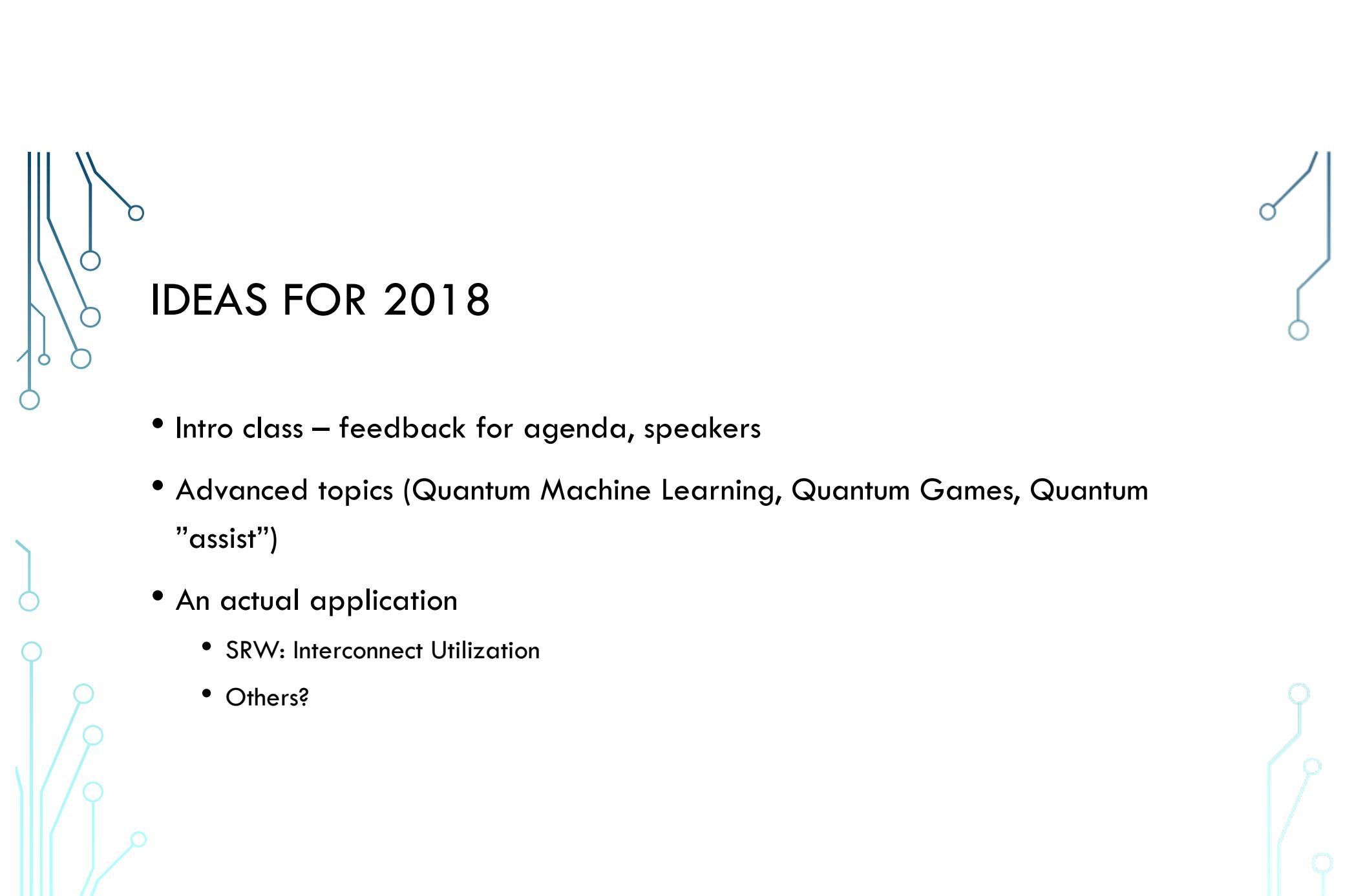
- the physics of information processing
- quantum logic
- quantum algorithms including Shor's factoring algorithm and Grover's search algorithm
- quantum error correction
- quantum communication and key distribution

This course will help you establish a foundation of knowledge for understanding what quantum computers can do, how they work, and how you can contribute to discovering new things and solving problems in quantum information science and engineering.
The three-course series comprises:

- 8.370.1x: Foundations of quantum and classical computing – quantum mechanics, reversible computation, and quantum measurement
- 8.370.2x: Simple quantum protocols and algorithms – teleportation and superdense coding, the

NEWS / INTERESTING

- “An Introduction to Quantum Computing, Without the Physics” Giacomo Nannicini, IBM
- “Complete 3-Qubit Grover search on a programmable quantum computer”, Nature Communications
- “Quantum Computing in the NISQ era and beyond” John Preskill



IDEAS FOR 2018

- Intro class – feedback for agenda, speakers
- Advanced topics (Quantum Machine Learning, Quantum Games, Quantum "assist")
- An actual application
 - SRW: Interconnect Utilization
 - Others?

QUANTUM COMPUTING REFRESHER

- Thanks to Emma Strubell for permission to use her slides!
 - Introduction to Quantum Algorithms
 - https://people.cs.umass.edu/~strubell/doc/quantum_tutorial.pdf
 - Slides
 - https://people.cs.umass.edu/~strubell/doc/quantum_presentation_1.pdf
 - https://people.cs.umass.edu/~strubell/doc/quantum_presentation_2.pdf

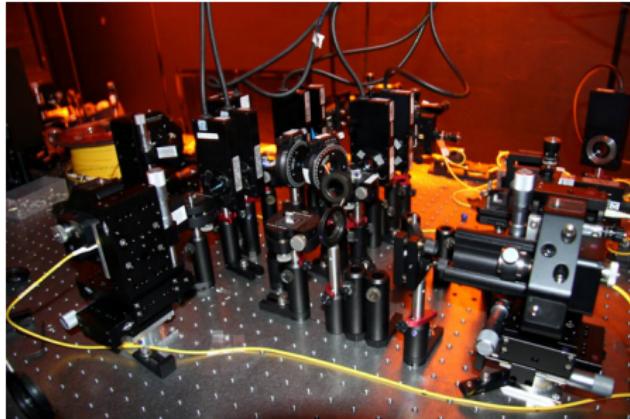
Introduction to Quantum Computing

Part I

Emma Strubell

http://cs.umaine.edu/~ema/quantum_tutorial.pdf

April 12, 2011



Overview

What is quantum computing?

- Background
- Caveats

Mathematical representation

- Fundamental differences
- Hilbert spaces and Dirac notation
- The qubit
- Quantum Registers
- Quantum logic gates
- Computational complexity

Outline

What is quantum computing?

- Background
- Caveats

Mathematical representation

- Fundamental differences
- Hilbert spaces and Dirac notation
- The qubit
- Quantum Registers
- Quantum logic gates
- Computational complexity

Origins of fame

- ▶ Quantum computer first proposed by Richard Feynman in 1981
 - ▶ Problem: efficiently simulating quantum systems inherently impossible on a classical computer
 - ▶ Solution: new machine “built of quantum mechanical elements which obey quantum mechanical laws”
- ▶ Daniel Simon demonstrates exponential speedup in 1994
 - ▶ nobody cares; algorithm too abstract
- ▶ Peter Shor demonstrates *exciting* exponential speedup in 1997
 - ▶ based on Simon’s algorithm
 - ▶ efficiently factors integers into primes
 - ▶ this breaks RSA



Outline

What is quantum computing?

- Background
- **Caveats**

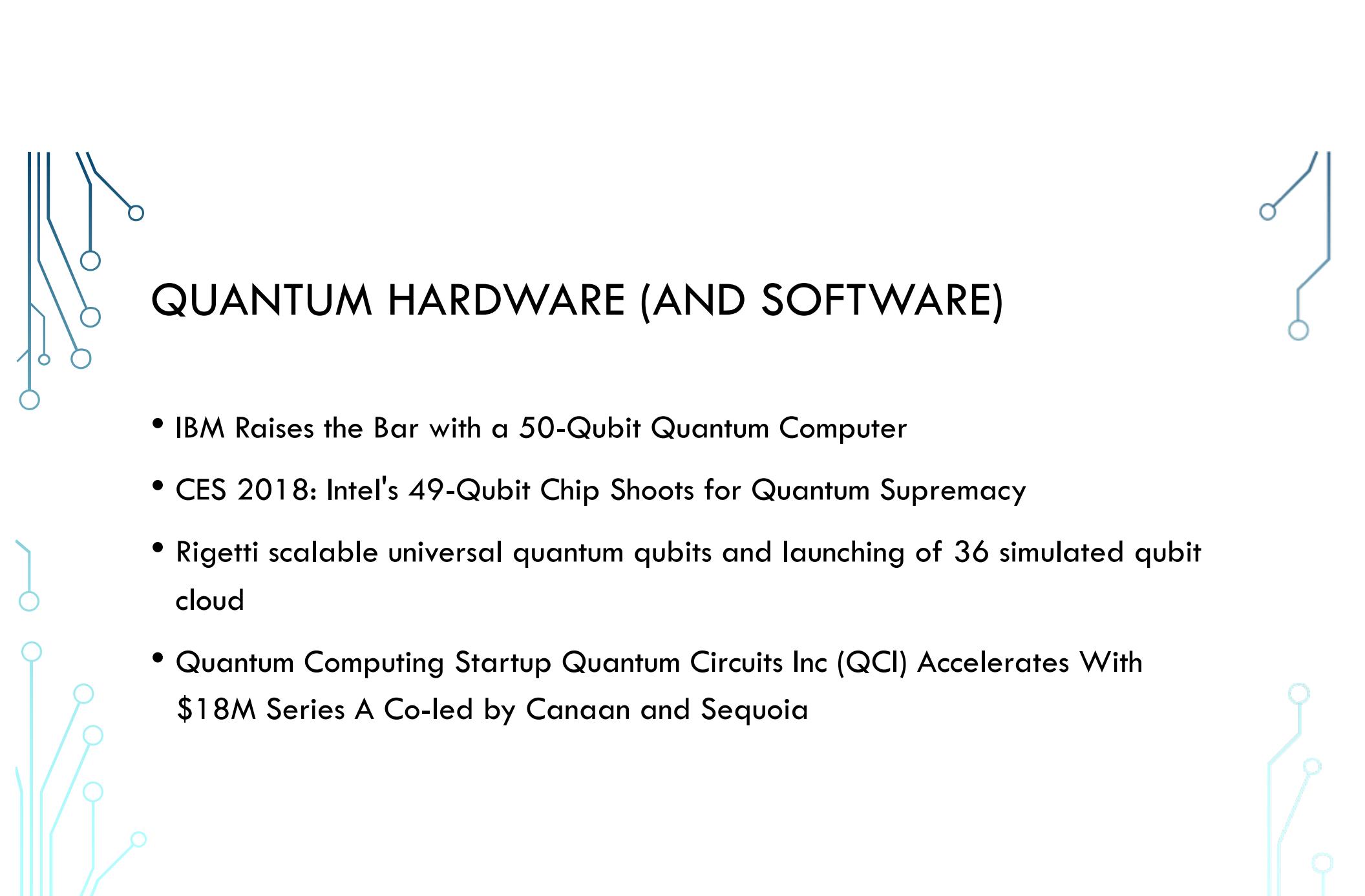
Mathematical representation

- Fundamental differences
- Hilbert spaces and Dirac notation
- The qubit
- Quantum Registers
- Quantum logic gates
- Computational complexity

Unfortunately, scalable QC's still don't exist

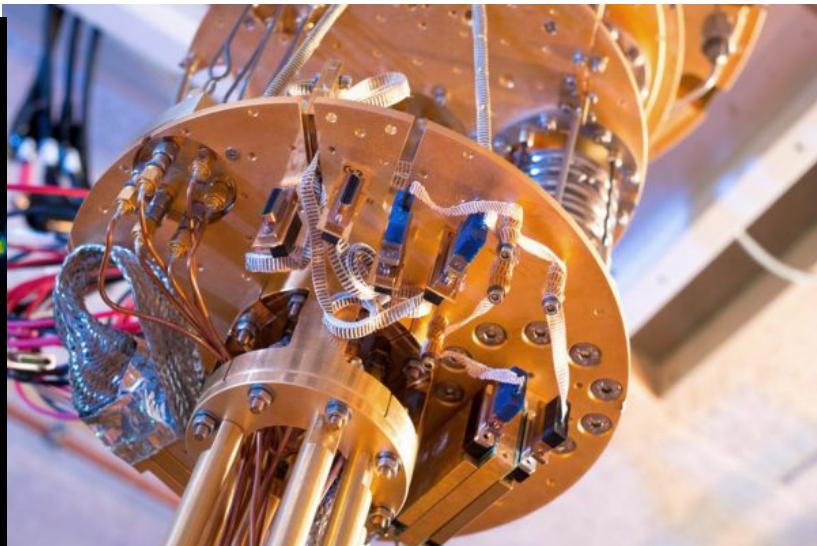
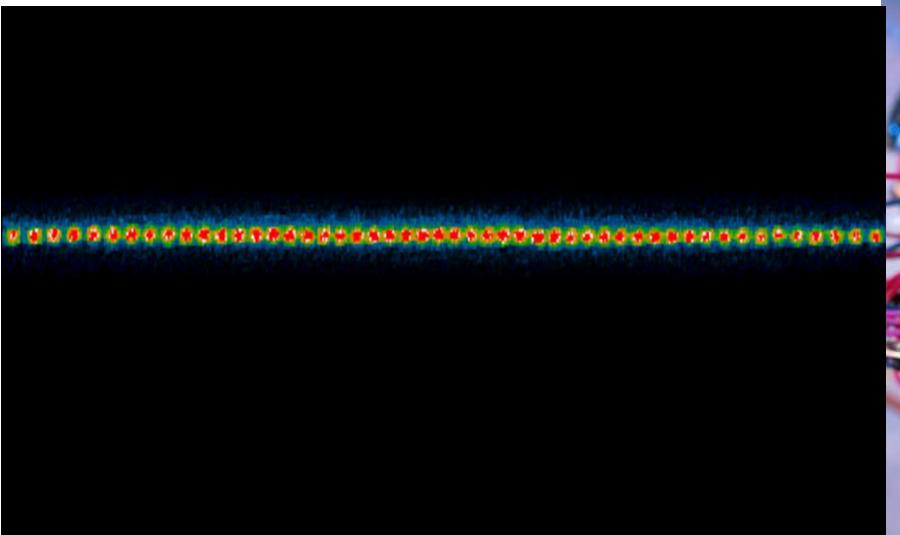
- ▶ As of 2009, quantum computers able to factor 15 into 5 and 3
- ▶ The problem is *decoherence*
 - ▶ Man-made quantum system wants to interact with surrounding systems
 - ▶ Sources of interference include electric and magnetic fields required to power machine itself





QUANTUM HARDWARE (AND SOFTWARE)

- IBM Raises the Bar with a 50-Qubit Quantum Computer
- CES 2018: Intel's 49-Qubit Chip Shoots for Quantum Supremacy
- Rigetti scalable universal quantum qubits and launching of 36 simulated qubit cloud
- Quantum Computing Startup Quantum Circuits Inc (QCI) Accelerates With \$18M Series A Co-led by Canaan and Sequoia



Comparison of Classical versus Quantum Computing

Classical Computer

- ◆ N particles
 - ⇒ 2^N unique states
- ◆ Computations are sequential
 - ⇒ Select 1 state
 - ⇒ Operate on it
 - ⇒ Put it back into memory
- ◆ Example: $N=3$
 - ⇒ 3 bits
 - ⇒ 8 states
 - ⇒ Work on one number at a time
 - ⇒ $x = 101$

Quantum Computer

- ◆ N particles
 - ⇒ 2^N unique states
- ◆ Computations occur in parallel
 - ⇒ States interact
 - ⇒ They are entangled
 - ⇒ Operate on all states at once
- ◆ Example: $N=3$
 - ⇒ 3 qubits
 - ⇒ 8 complex amplitudes
 - ⇒ Operator manipulates 8 at once
 - $x \xrightarrow{\frac{1}{\sqrt{2}} [a|000\rangle \ b|001\rangle \ c|010\rangle \ ...h|111\rangle]$

⇒ *Exponential Speedup*

Overview

What is quantum computing?

- Background
- Caveats

Mathematical representation

- Fundamental differences
- Hilbert spaces and Dirac notation
- The qubit
- Quantum Registers
- Quantum logic gates
- Computational complexity

Outline

What is quantum computing?

- Background
- Caveats

Mathematical representation

- Fundamental differences
- Hilbert spaces and Dirac notation
- The qubit
- Quantum Registers
- Quantum logic gates
- Computational complexity

Three main differences from classical computers

1 Superposition

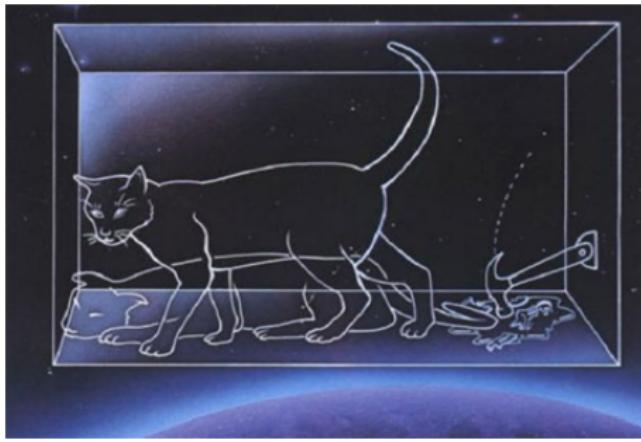
- ▶ quantum system exists in all possible states at all times

2 Probabilities

- ▶ fortunately, a probability can be associated with each of those states

3 Entanglement

- ▶ probabilities of different states can depend on each other
- ▶ quantum teleportation uses this property for cryptographic purposes



The Effect of Measurement

- ♦ If you *measure* the quantum system, the qubit (superposition of states) *collapses* to one of the basis states $|0\rangle$ and $|1\rangle$

$$|x\rangle = c_1|0\rangle + c_2|1\rangle \rightarrow \text{measure} \rightarrow \text{either } |0\rangle \text{ or } |1\rangle$$

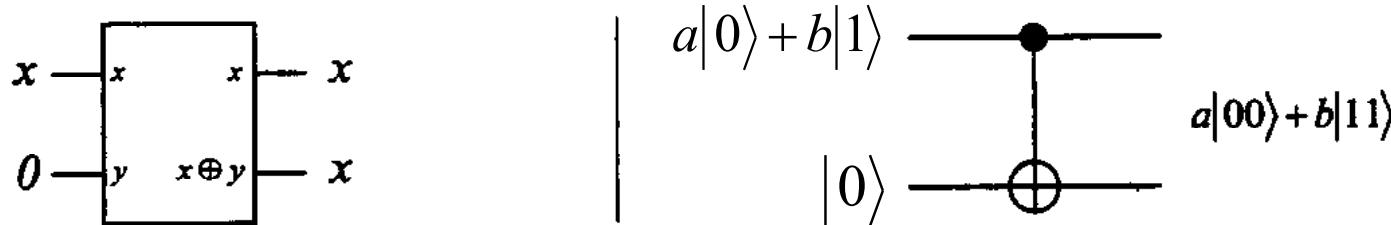
- ♦ c_1 and c_2 are called *probability amplitudes* because probability of getting $|0\rangle$ or $|1\rangle$ upon measurement depends on their squared amplitudes:

$$\text{Prob}(|0\rangle) = \|c_1\|^2 = a_1^2 + b_1^2 \quad \text{Prob}(|1\rangle) = \|c_2\|^2 = a_2^2 + b_2^2$$

- ♦ Since $\text{Prob}(|0\rangle) + \text{Prob}(|1\rangle) = 1$, $\|c_1\|^2 + \|c_2\|^2 = 1$

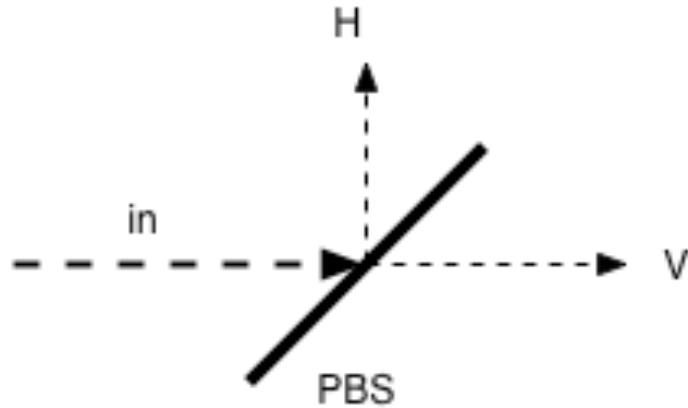
No Cloning Theorem

- ◆ There is no unitary transform that allows us to copy a qubit
 - ⇒ Proof: Suppose U is a copying matrix: $U|c0\rangle = |cc\rangle$ for all states $|c\rangle$
 - ⇒ Then,
$$\text{if } |c\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), U|c\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$
$$\text{but } U|c\rangle = |cc\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle) \neq \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$
- ◆ Illustration: Copy a bit using CNOT....yields an entangled state!



Classical and quantum circuits to copy an unknown bit

Schrödinger's Photons

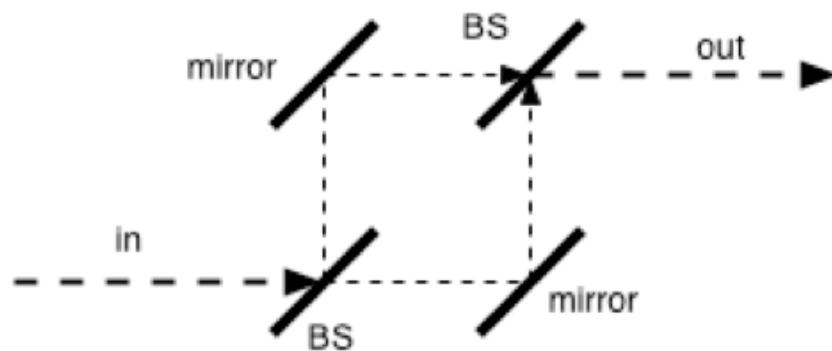


- **A polarizing beam splitter is a device which reflects all light of one polarization (say H) and transmits all light of the other polarization (say V).**
- **If light polarized 45° to H and V arrives, half of it is reflected and half transmitted.**
- **If a single photon at 45° arrives, it will be reflected or transmitted with 50/50 probability. We describe such a photon as a superposition of H and V: $(|H\rangle + |V\rangle)/\sqrt{2}$.**

Interference



However, it is not just a simple matter of a photon going one way or the other with equal probability. If we pass the two beams through a pair of beamsplitters, we find that the photon can recombine in such a way that the probability to go in one direction cancels out. This is interference.



Outline

What is quantum computing?

- Background
- Caveats

Mathematical representation

- Fundamental differences
- **Hilbert spaces and Dirac notation**
- The qubit
- Quantum Registers
- Quantum logic gates
- Computational complexity

Dirac notation

- ▶ Just another way of describing vectors:

$$\mathbf{v} = \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_n \end{bmatrix} = |\mathbf{v}\rangle$$

- ▶ and their duals:

$$\langle \mathbf{v}| = \overline{\mathbf{v}^T} = [\overline{v_0} \quad \overline{v_1} \quad \dots \quad \overline{v_n}]$$

- ▶ Convenient for describing vectors in the Hilbert space \mathbb{C}^n , the vector space of quantum mechanics

\mathbb{C}^n and the inner product

- ▶ A Hilbert space, for our (finite) purposes, is a vector space with an *inner product*, and a *norm* defined by that inner product. We use the following in \mathbb{C}^n :
 - ▶ The inner product assigns a scalar value to each pair of vectors:

$$\langle \mathbf{u} | \mathbf{v} \rangle = \overline{\mathbf{u}^T} \mathbf{v} = [\overline{u_0} \quad \overline{u_1} \quad \dots \quad \overline{u_n}] \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_n \end{bmatrix} = \overline{u_0} \cdot v_0 + \overline{u_1} \cdot v_1 + \dots + \overline{u_n} \cdot v_n$$

- ▶ The norm is the square root of the inner product of a vector with itself (i.e. Euclidean norm, ℓ^2 -norm, 2-norm over complex numbers):

$$\| |\mathbf{v} \rangle \| = \sqrt{\langle \mathbf{v} | \mathbf{v} \rangle}$$

- ▶ Geometrically, this norm gives the distance from the origin to the point $|\mathbf{v} \rangle$ that follows from the Pythagorean theorem.

Properties of the inner product

The inner product satisfies the three following properties:

Definition

- 1 $\langle \mathbf{v} | \mathbf{v} \rangle \geq 0$, with $\langle \mathbf{v} | \mathbf{v} \rangle = 0$ if and only if $|\mathbf{v}\rangle = \mathbf{0}$.
- 2 $\langle \mathbf{u} | \mathbf{v} \rangle = \overline{\langle \mathbf{v} | \mathbf{u} \rangle}$ for all $|\mathbf{u}\rangle, |\mathbf{v}\rangle$ in the vector space.
- 3 $\langle \mathbf{u} | \alpha_0 \mathbf{v} + \alpha_1 \mathbf{w} \rangle = \alpha_0 \langle \mathbf{u} | \mathbf{v} \rangle + \alpha_1 \langle \mathbf{u} | \mathbf{w} \rangle$.

More generally, the inner product of $|\mathbf{u}\rangle$ and $\sum_i \alpha_i |\mathbf{v}_i\rangle$ is equal to $\sum_i \alpha_i \langle \mathbf{u} | \mathbf{v}_i \rangle$ for all scalars α_i and vectors $|\mathbf{u}\rangle, |\mathbf{v}\rangle$ in the vector space (this is known as *linearity in the second argument*).

The outer product

- The *outer product* is the *tensor* or *Kronecker product* of a vector with the conjugate transpose of another. The result is not a scalar, but a matrix:

$$|\mathbf{v}\rangle \langle \mathbf{u}| = \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_n \end{bmatrix} \begin{bmatrix} \overline{u_0} & \overline{u_1} & \dots & \overline{u_m} \end{bmatrix} = \begin{bmatrix} v_0\overline{u_0} & v_0\overline{u_1} & \dots & v_0\overline{u_m} \\ v_1\overline{u_0} & v_1\overline{u_1} & \dots & v_1\overline{u_m} \\ \vdots & \vdots & \ddots & \vdots \\ v_n\overline{u_0} & v_n\overline{u_1} & \dots & v_n\overline{u_m} \end{bmatrix}$$

- Often used to describe a linear transformation between vector spaces.
- A linear transformation from a Hilbert space U to another Hilbert space V on a vector $|\mathbf{w}\rangle$ in U may be succinctly described in Dirac notation:

$$(|\mathbf{v}\rangle \langle \mathbf{u}|) |\mathbf{w}\rangle = |\mathbf{v}\rangle \langle \mathbf{u}| \mathbf{w} = \langle \mathbf{u} | \mathbf{w} \rangle |\mathbf{v}\rangle$$

Since $\langle \mathbf{u} | \mathbf{w} \rangle$ is a commutative, scalar value.

The tensor product

- ▶ Usually simplified from $|\mathbf{u}\rangle \otimes |\mathbf{v}\rangle$ to $|\mathbf{u}\rangle |\mathbf{v}\rangle$ or $|\mathbf{uv}\rangle$
- ▶ A vector tensored with itself n times is denoted $|\mathbf{v}\rangle^{\otimes n}$ or $|\mathbf{v}\rangle^n$
- ▶ Two column vectors $|\mathbf{u}\rangle$ and $|\mathbf{v}\rangle$ of lengths m and n yield a column vector of length $m \cdot n$ when tensored:

$$|\mathbf{u}\rangle |\mathbf{v}\rangle = |\mathbf{uv}\rangle = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_m \end{bmatrix} \otimes \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} u_0 \cdot v_0 \\ u_0 \cdot v_1 \\ \vdots \\ u_0 \cdot v_n \\ u_1 \cdot v_0 \\ u_1 \cdot v_1 \\ \vdots \\ u_{m-1} \cdot v_n \\ u_m \cdot v_0 \\ \vdots \\ u_m \cdot v_n \end{bmatrix}$$

Outline

What is quantum computing?

- Background
- Caveats

Mathematical representation

- Fundamental differences
- Hilbert spaces and Dirac notation
- **The qubit**
- Quantum Registers
- Quantum logic gates
- Computational complexity

\mathbb{C}^2 describes a single quantum bit (qubit)

- ▶ A classical bit may be represented as a base-2 number that takes either the value 1 or the value 0
- ▶ Qubits are also base-2 numbers, but in a superposition of the measurable values 1 and 0
- ▶ The state of a qubit at any given time represented as a two-dimensional *state space* in \mathbb{C}^2 with *orthonormal* basis vectors $|1\rangle$ and $|0\rangle$
- ▶ The superposition $|\psi\rangle$ of a qubit is represented as a linear combination of those basis vectors:

$$|\psi\rangle = a_0 |0\rangle + a_1 |1\rangle$$

Where a_0 is the complex scalar *amplitude* of measuring $|0\rangle$, and a_1 the amplitude of measuring the value $|1\rangle$.

Physical representation [edit]

Any two-level system can be used as a qubit. Multilevel systems can be used as well, if they possess two states that can be effectively decoupled from the rest (e.g., ground state and first excited state of a nonlinear oscillator). There are various proposals. Several physical implementations that approximate two-level systems to various degrees were successfully realized. Similarly to a classical bit where the state of a transistor in a processor, the magnetization of a surface in a [hard disk](#) and the presence of current in a cable can all be used to represent bits in the same computer, an eventual quantum computer is likely to use various combinations of qubits in its design. The following is an incomplete list of physical implementations of qubits, and the choices of basis are by convention only.

Physical support	Name	Information support	$ 0\rangle$	$ 1\rangle$
Photon	Polarization encoding	Polarization of light	Horizontal	Vertical
	Number of photons	Fock state	Vacuum	Single photon state
	Time-bin encoding	Time of arrival	Early	Late
Coherent state of light	Squeezed light	Quadrature	Amplitude-squeezed state	Phase-squeezed state
Electrons	Electronic spin	Spin	Up	Down
	Electron number	Charge	No electron	One electron
Nucleus	Nuclear spin addressed through NMR	Spin	Up	Down
Optical lattices	Atomic spin	Spin	Up	Down
Josephson junction	Superconducting charge qubit	Charge	Uncharged superconducting island ($Q=0$)	Charged superconducting island ($Q=2e$, one extra Cooper pair)
	Superconducting flux qubit	Current	Clockwise current	Counterclockwise current
	Superconducting phase qubit	Energy	Ground state	First excited state
Singly charged quantum dot pair	Electron localization	Charge	Electron on left dot	Electron on right dot
Quantum dot	Dot spin	Spin	Down	Up

Amplitudes, not probabilities

- ▶ Amplitudes may be thought of as “quantum probabilities” in that they represent the chance that a given quantum state will be observed when the superposition is collapsed
- ▶ Most fundamental difference between probabilities of states in classical probabilistic algorithms and amplitudes: amplitudes are complex
 - ▶ Complex numbers required to fully describe superposition of states, interference or entanglement in quantum systems.¹
 - ▶ As the probabilities of a classical system must sum to 1, so too the squares of the absolute values of the amplitudes of states in a quantum system must add up to 1

¹See <http://www.scottaaronson.com/democritus/lec9.html> for a great discussion by of why complex numbers and the 2-norm are used to describe quantum mechanical systems

Amplitudes and the normalization condition

- ▶ Just as the hardware underlying the bits of a classical computer may vary in voltage, quantum systems are not usually so perfectly behaved
- ▶ An assumption is made about quantum state vectors called the *normalization condition*: $|\psi\rangle$ is a unit vector.
 - ▶ $\| |\psi\rangle \| = \langle \psi | \psi \rangle = 1$
 - ▶ If $|0\rangle$ and $|1\rangle$ are orthonormal, then by orthogonality $\langle 0|1\rangle = \langle 1|0\rangle = 0$, and by normality $\langle 0|0\rangle = \langle 1|1\rangle = 1$
 - ▶ It follows that $|a_0|^2 + |a_1|^2 = 1$:

$$\begin{aligned} 1 &= \langle \psi | \psi \rangle \\ &= (\overline{a_0} \langle 0 | + \overline{a_1} \langle 1 |) \cdot (a_0 | 0 \rangle + a_1 | 1 \rangle) \\ &= |a_0|^2 \langle 0 | 0 \rangle + |a_1|^2 \langle 1 | 1 \rangle + \overline{a_1} a_0 \langle 1 | 0 \rangle + \overline{a_0} a_1 \langle 0 | 1 \rangle \\ &= |a_0|^2 + |a_1|^2 \end{aligned}$$

Why we use Dirac notation

The following is equivalent to the last slide:

$$\begin{aligned} 1 &= \langle \psi | \psi \rangle \\ &= (\overline{a_0} \langle 0 | + \overline{a_1} \langle 1 |) \cdot (a_0 | 0 \rangle + a_1 | 1 \rangle) \\ &= (\overline{a_0} [\overline{\psi_{00}} \quad \overline{\psi_{01}}] + \overline{a_1} [\overline{\psi_{10}} \quad \overline{\psi_{11}}]) \cdot \left(a_0 \begin{bmatrix} \psi_{00} \\ \psi_{01} \end{bmatrix} + a_1 \begin{bmatrix} \psi_{10} \\ \psi_{11} \end{bmatrix} \right) \\ &= [\overline{a_0 \psi_{00}} + \overline{a_1 \psi_{10}} \quad \overline{a_0 \psi_{01}} + \overline{a_1 \psi_{11}}] \cdot \begin{bmatrix} a_0 \psi_{00} + a_1 \psi_{10} \\ a_0 \psi_{01} + a_1 \psi_{11} \end{bmatrix} \\ &= \overline{a_0 \psi_{00}} a_0 \psi_{00} + \overline{a_1 \psi_{10}} a_0 \psi_{00} + \overline{a_0 \psi_{01}} a_1 \psi_{10} + \overline{a_1 \psi_{10}} a_1 \psi_{10} \\ &\quad + \overline{a_0 \psi_{01}} a_0 \psi_{01} + \overline{a_1 \psi_{11}} a_0 \psi_{01} + \overline{a_0 \psi_{01}} a_1 \psi_{11} + \overline{a_1 \psi_{11}} a_1 \psi_{11} \\ &= |a_0|^2 (|\psi_{00}|^2 + |\psi_{01}|^2) + |a_1|^2 (|\psi_{10}|^2 + |\psi_{11}|^2) \\ &\quad + \overline{a_1} a_0 (\overline{\psi_{10}} \psi_{00} + \overline{\psi_{11}} \psi_{01}) + \overline{a_0} a_1 (\overline{\psi_{00}} \psi_{10} + \overline{\psi_{01}} \psi_{11}) \\ &= |a_0|^2 + |a_1|^2 \end{aligned}$$

The computational basis

- ▶ $|0\rangle$ and $|1\rangle$ may be transformed into any two vectors that form an orthonormal basis in \mathbb{C}^2
- ▶ The most common basis used in quantum computing is called the *computational basis*:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

- ▶ The computational basis tends to be the most straightforward basis for computing and understanding quantum algorithms
- ▶ Assume I'm using the computational basis unless otherwise stated

Another basis

- ▶ Any other orthonormal basis could be used:

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

- ▶ Providing a slightly different but equivalent way of expressing of a qubit:

$$\begin{aligned} |\psi\rangle &= a_0 |0\rangle + a_1 |1\rangle \\ &= a_0 \frac{|+\rangle + |-\rangle}{\sqrt{2}} + a_1 \frac{|+\rangle - |-\rangle}{\sqrt{2}} \\ &= \frac{a_0 + a_1}{\sqrt{2}} |+\rangle + \frac{a_0 - a_1}{\sqrt{2}} |-\rangle \end{aligned}$$

- ▶ Here, instead of measuring the states $|0\rangle$ and $|1\rangle$ each with respective probabilities $|a_0|^2$ and $|a_1|^2$, the states $|+\rangle$ and $|-\rangle$ would be measured with probabilities $|a_0 + a_1|^2/2$ and $|a_0 - a_1|^2/2$.

Outline

What is quantum computing?

- Background
- Caveats

Mathematical representation

- Fundamental differences
- Hilbert spaces and Dirac notation
- The qubit
- **Quantum Registers**
- Quantum logic gates
- Computational complexity

Registers more useful than single qubits

- ▶ Each qubit in a quantum register is in a superposition of $|1\rangle$ and $|0\rangle$
- ▶ Consequently, a register of n qubits is in a superposition of all 2^n possible bit strings that could be represented using n bits
- ▶ The state space of a size- n quantum register is a linear combination of n basis vectors, each of length 2^n :

$$|\psi_n\rangle = \sum_{i=0}^{2^n-1} a_i |i\rangle$$

- ▶ A three-qubit register would thus have the following expansion:

$$\begin{aligned} |\psi_2\rangle &= a_0 |000\rangle + a_1 |001\rangle + a_2 |010\rangle + a_3 |011\rangle \\ &\quad + a_4 |100\rangle + a_5 |101\rangle + a_6 |110\rangle + a_7 |111\rangle \end{aligned}$$

Registers continued

- ▶ Each possible bit configuration in the quantum superposition is denoted by the tensor product of its counterpart qubits
- ▶ Consider $|101\rangle$, the bit string that represents the integer value 5:

$$\begin{aligned}|101\rangle &= |1\rangle \otimes |0\rangle \otimes |1\rangle \\&= \begin{bmatrix} 0 \\ 1 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\&= [0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0]^T\end{aligned}$$

- ▶ As with single qubits, the squared absolute value of the amplitude associated with a given bit string is the probability of observing that bit string, and the squares of the absolute values of the amplitudes of all 2^n possible bit configurations of an n -bit register sum to unity:

$$\sum_{i=0}^{2^n-1} |a_i|^2 = 1$$

Outline

What is quantum computing?

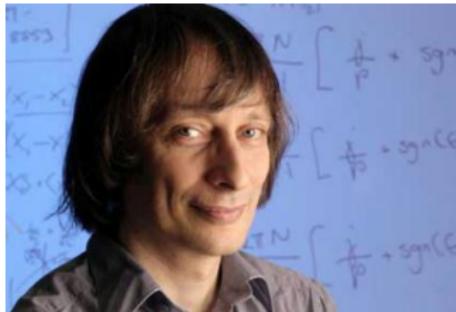
- Background
- Caveats

Mathematical representation

- Fundamental differences
- Hilbert spaces and Dirac notation
- The qubit
- Quantum Registers
- Quantum logic gates**
- Computational complexity

Evolving the system: quantum circuits and quantum gates

- ▶ One way of thinking about algorithm design and computation is via quantum Turing machines
- ▶ First described by David Deutsch in 1985, but both a quantum Turing machine's tape and its read-write head exist in superpositions of an exponential number states!
- ▶ Instead of using the Turing machine as a computational model, operations on a quantum computer most often described using quantum circuits (also introduced by Deutsch a few years later)
- ▶ Although circuits are computationally equivalent to Turing machines, they are usually much simpler to depict, manipulate and understand



Quantum gates represent unitary transformations

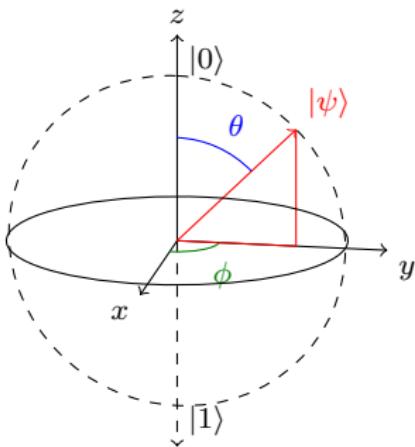
- ▶ Quantum gates are represented as transformation matrices, linear operators applied to a quantum register by tensoring the operator with the register
- ▶ All quantum linear operators must be *unitary*:
 - ▶ If a complex matrix U is unitary, then $U^{-1} = U^\dagger$, where U^\dagger is the conjugate transpose: $U^\dagger = \overline{U}^T$
 - ▶ It follows that $UU^\dagger = U^\dagger U = I$
 - ▶ Unitary operators preserve inner product:

$$\langle \mathbf{u} | U^\dagger U | \mathbf{v} \rangle = \langle \mathbf{u} | I | \mathbf{v} \rangle = \langle \mathbf{u} | \mathbf{v} \rangle$$

- ▶ The composition of two unitary operators is also unitary:

$$(UV)^\dagger = V^\dagger U^\dagger = V^{-1} U^{-1} = (UV)^{-1}$$

The Bloch sphere



- ▶ Unitary transformations performed on a qubit may be visualized as rotations and reflections about the x , y , and z axes of the *Bloch sphere*
- ▶ All linear combinations $a_0 |0\rangle + a_1 |1\rangle$ in \mathbb{C}^2 correspond to all the points (θ, ψ) on the surface of the unit sphere, where $a_0 = \cos(\theta/2)$ and $a_1 = e^{i\phi} \sin(\theta/2) = (\cos \phi + i \sin \phi) \sin \frac{\theta}{2}$

The Hadamard operator

$$\boxed{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \langle 0| + \frac{|0\rangle - |1\rangle}{\sqrt{2}} \langle 1|$$

- ▶ Often referred to as a “fair coin flip,” the Hadamard operator applied to a qubit with the value $|0\rangle$ or $|1\rangle$ will induce an equal superposition of the states $|0\rangle$ and $|1\rangle$:

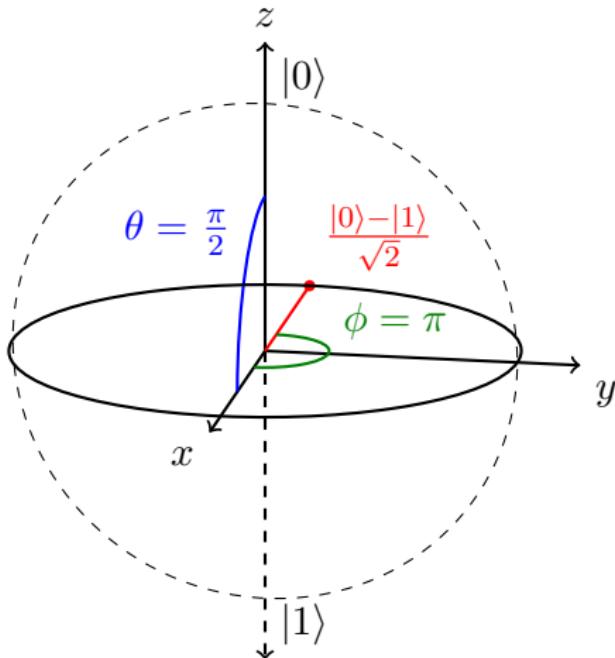
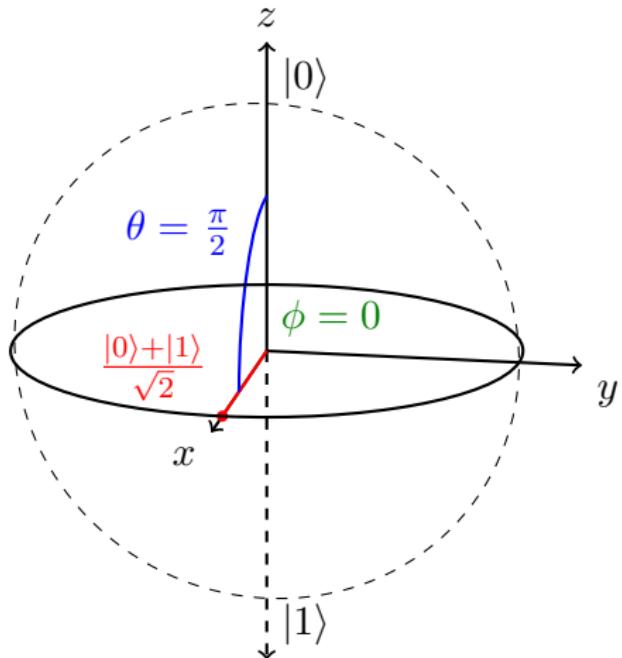
$$H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \langle 0|0\rangle + \frac{|0\rangle - |1\rangle}{\sqrt{2}} \langle 1|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

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

- ▶ Many quantum algorithms begin by applying the Hadamard operator to each qubit in a register initialized to $|0\rangle^n$, which puts the entire register into an equal superposition of states

Bloch sphere representation of the Hadamard operator

- Geometrically, the Hadamard operator performs a rotation of $\pi/2$ about the y axis followed by a rotation about the x axis by π radians on the Bloch sphere:



The Pauli gates

- ▶ The three Pauli gates, named after yet another Nobel laureate Wolfgang Pauli, are also important single-qubit gates for quantum computation
- ▶ The Pauli-X gate swaps the amplitudes of $|0\rangle$ and $|1\rangle$:

$$\text{---} \boxed{X} \text{---} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = |1\rangle\langle 0| + |0\rangle\langle 1|$$

- ▶ The Pauli-Y gate swaps the amplitudes of $|0\rangle$ and $|1\rangle$, multiplies each amplitude by i , and negates the amplitude of $|1\rangle$:

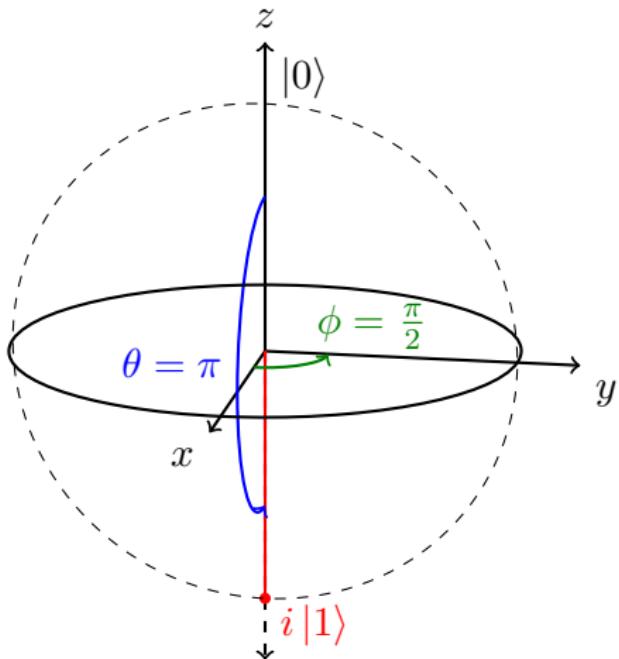
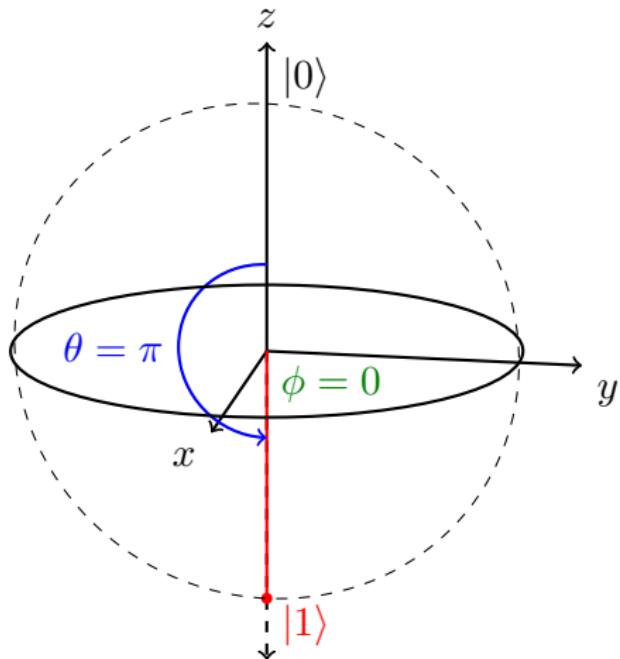
$$\text{---} \boxed{Y} \text{---} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = i|1\rangle\langle 0| - i|0\rangle\langle 1|$$

- ▶ And the Pauli-Z gate negates the amplitude of $|1\rangle$, leaving the amplitude of $|0\rangle$ the same:

$$\text{---} \boxed{Z} \text{---} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = |1\rangle\langle 0| - |0\rangle\langle 1|$$

Bloch sphere representation of Pauli-X and -Y gates

- The Pauli-X, -Y, and -Z gates correspond to rotations by π radians about the x , y , and z axes respectively on the Bloch sphere



Generalized phase shift

- ▶ The Pauli-Z gate, altering only the phase of the system, is a special case of the more general phase-shift gate, which does not modify the amplitude of $|0\rangle$ but changes the phase of $|1\rangle$ by a factor of $e^{i\theta}$ for any value of θ :

$$\text{---} \xrightarrow{R_\theta} \text{---} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix} = |1\rangle\langle 0| + e^{i\theta}|0\rangle\langle 1|$$

- ▶ The Pauli-Z gate is equivalent to the phase-shift gate with $\theta = \pi$.
- ▶ Wolfgang Pauli with friends Werner Heisenberg and Enrico Fermi:



More phase shift gates

- ▶ Another special case of the phase-shift gate where $\theta = \pi/2$ is known as simply the phase gate, denoted S , which changes the phase of $|1\rangle$ by a factor of i :

$$\text{---} \boxed{S} \text{---} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} = |1\rangle\langle 0| + i|0\rangle\langle 1|$$

- ▶ And the phase-shift gate where $\theta = \pi/4$ is referred to as the $\pi/8$ gate, or T :

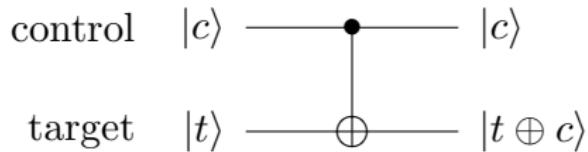
$$\text{---} \boxed{T} \text{---} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} = |1\rangle\langle 0| + e^{i\pi/4}|0\rangle\langle 1|$$

With the name $\pi/8$ coming from the fact that this transformation can also be written as a matrix with $\pi/8$ along the diagonal:

$$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} = e^{i\pi/8} \begin{bmatrix} e^{-i\pi/8} & 0 \\ 0 & e^{i\pi/8} \end{bmatrix}$$

Controlled operations: CNOT

- ▶ Quantum computing also makes use of *controlled operations*, multi-qubit operations that change the state of a qubit based on the values of other qubits
- ▶ The quantum controlled-NOT or CNOT gate swaps the amplitudes of the $|0\rangle$ and $|1\rangle$ basis states of a qubit, equivalent to application of the Pauli-X gate, only if the controlling qubit has the value $|1\rangle$:



```

"1. hadamard " MatrixForm[hadamard = (1 / (2^(1/2))) {{1, 1}, {1, -1}}]
MatrixForm[identity = {{1, 0}, {0, 1}}]
"2. CNOT operator " MatrixForm[cnot = {{1, 0, 0, 0}, {0, 1, 0, 0}, {0, 0, 0, 1}, {0, 0, 1, 0}}]
"3. Tensored Hadamard " MatrixForm[tensored = KroneckerProduct[hadamard, identity]]
cnothadamard2qubit = cnot.tensored
"4. Two qubit H/CNOT operator" MatrixForm[cnothadamard2qubit]
initialstate = {1, 0, 0, 0}
"5.A initial two qubit state is |00> " MatrixForm[initialstate]
"5.B result is a Bell pair "MatrixForm[cnothadamard2qubit.initialstate]
initialstate = {0, 1, 0, 0}
"6.A initial two qubit state is |01> " MatrixForm[initialstate]
"6.B result is a Bell pair "MatrixForm[cnothadamard2qubit.initialstate]
initialstate = {0, 0, 1, 0}
"7.A initial two qubit state is |10> " MatrixForm[initialstate]
"7.B result is a Bell pair "MatrixForm[cnothadamard2qubit.initialstate]
initialstate = {0, 0, 0, 1}
"8.A initial two qubit state is |11> " MatrixForm[initialstate]
"8.B result is a Bell pair "MatrixForm[cnothadamard2qubit.initialstate]

```

1. hadamard
$$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

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

2. CNOT operator
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

3. Tensored Hadamard

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

$$\left\{ \left\{ \frac{1}{\sqrt{2}}, 0, \frac{1}{\sqrt{2}}, 0 \right\}, \left\{ 0, \frac{1}{\sqrt{2}}, 0, \frac{1}{\sqrt{2}} \right\}, \left\{ 0, \frac{1}{\sqrt{2}}, 0, -\frac{1}{\sqrt{2}} \right\}, \left\{ \frac{1}{\sqrt{2}}, 0, -\frac{1}{\sqrt{2}}, 0 \right\} \right\}$$

4. Two qubit H/CNOT operator

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 \end{pmatrix}$$

$$\{1, 0, 0, 0\}$$

5.A initial two qubit state is $|00\rangle$

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

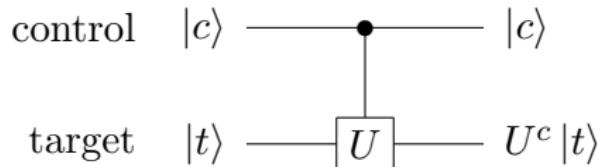
5.B result is a Bell pair

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix}$$

$$\{0, 1, 0, 0\}$$

Generalized controlled operations

- ▶ Controlled operations are not restricted to conditional application of the Pauli-X gate; Any unitary operation may be performed:



- ▶ Matrix representation:

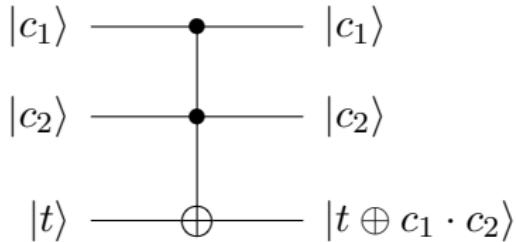
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & x_{00} & x_{10} \\ 0 & 0 & x_{01} & x_{11} \end{bmatrix}$$

- ▶ Dirac equivalent:

$$\begin{aligned} &|00\rangle\langle 00| + |01\rangle\langle 01| + x_{00}|10\rangle\langle 10| + x_{01}|10\rangle\langle 11| \\ &\quad + x_{10}|11\rangle\langle 10| + x_{11}|11\rangle\langle 11| \end{aligned}$$

Controlled operations: Toffoli

- ▶ In fact, controlled operations are possible with any number n control qubits and any unitary operator on k qubits
- ▶ The Toffoli gate is probably the best known of these gates
- ▶ Also known as the controlled-controlled-NOT gate, the Toffoli gate acts on three qubits: two control qubits and one target
- ▶ If both control qubits are set, then the amplitudes of the target qubit are flipped:



Toffoli continued

- ▶ The Toffoli gate was originally devised as a universal, reversible *classical* logic gate by Tommaso Toffoli
- ▶ It is especially interesting because depending on the input, the gate can perform logical AND, XOR, NOT and FANOUT operations...
- ▶ This makes it universal for classical computing!
- ▶ Quantum computing is reversible:
 - ▶ All evolution in a quantum system can be described by unitary matrices, all unitary transformations are invertible, and thus all quantum computation is reversible
- ▶ The Toffoli gate implies that quantum computation is at least as powerful as classical computation



Outline

What is quantum computing?

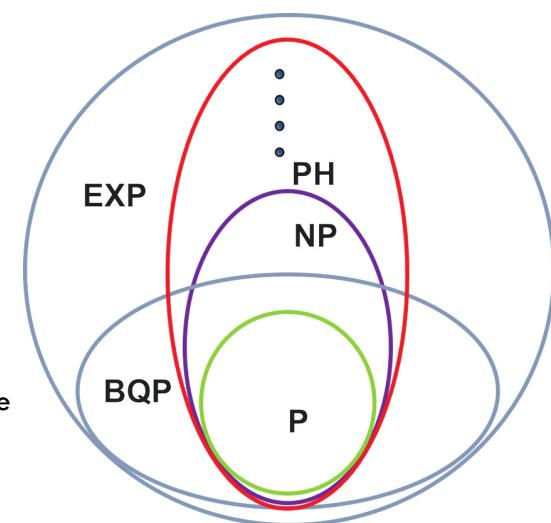
- Background
- Caveats

Mathematical representation

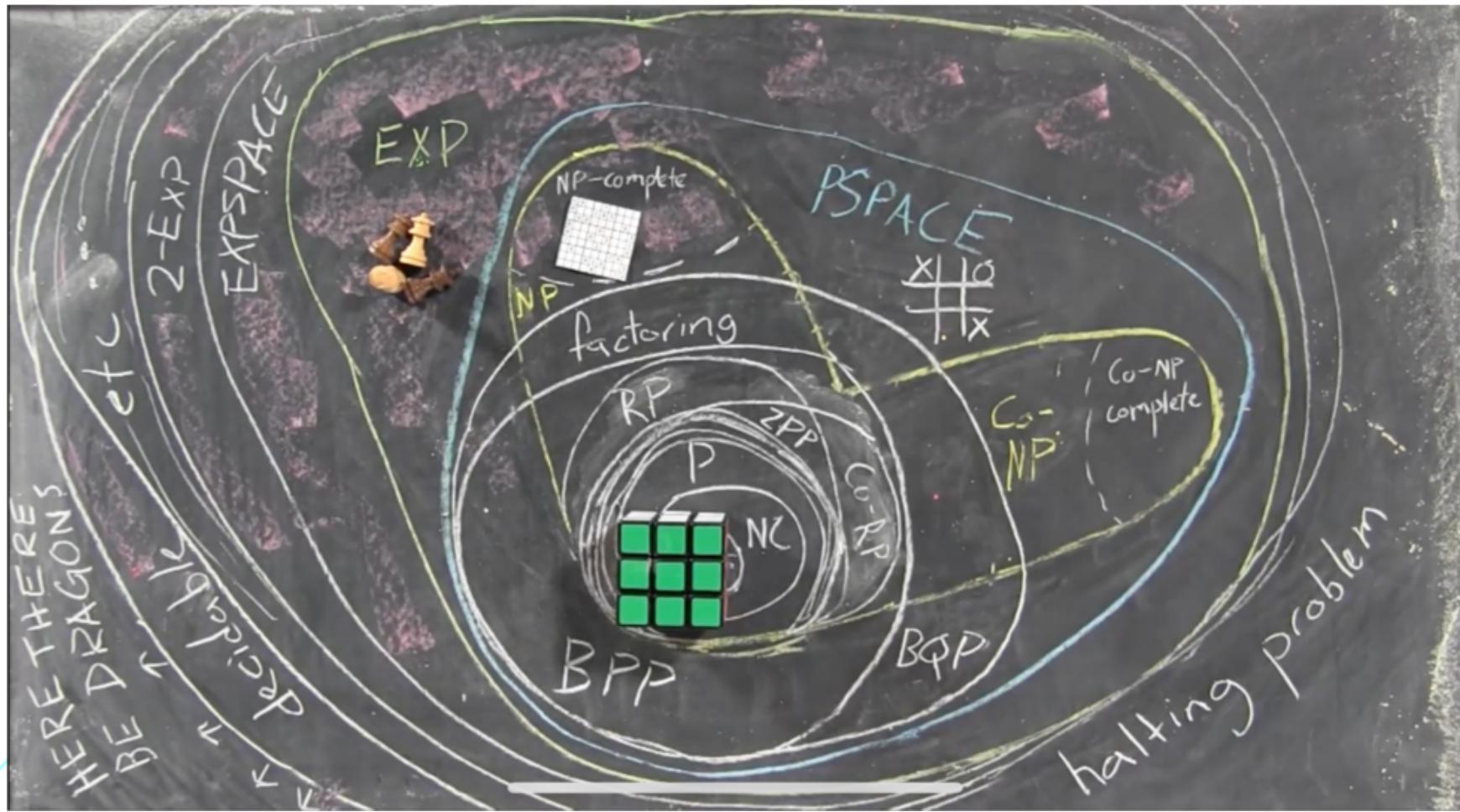
- Fundamental differences
- Hilbert spaces and Dirac notation
- The qubit
- Quantum Registers
- Quantum logic gates
- Computational complexity

AARONSON – “THE COMPLEXITY PETTING ZOO”

- Complexity theory analyzes the amount of resources time and/or space to solve a problem
 - Theoretical CS uses Turing machines as the basis
- P is the class of problems solved in polynomial time
 - "problem is a decision problem"
 - $f(k \text{ bits}) \rightarrow \{\text{True, False}\}$ that gets solved in $\text{TIME}(n^k)$
- PSPACE is P space where Time is unlimited
- EXP is the class of problems solved in $\text{TIME}(2^{n^k})$
 - Examples : Chess 10^{120} moves
- NP
 - EXP problems where a proof exists that can be checked in polynomial time
 - Favorite example : checkmate, factoring a 10000 digit number
- $P \neq NP$, NP-Hard, NP Complete, BQP (Another talk...)
- First 100 pages of Democritus



BIGGER COMPLEXITY ZOO



BIG O CHARTS VIA ERIC ROWELL

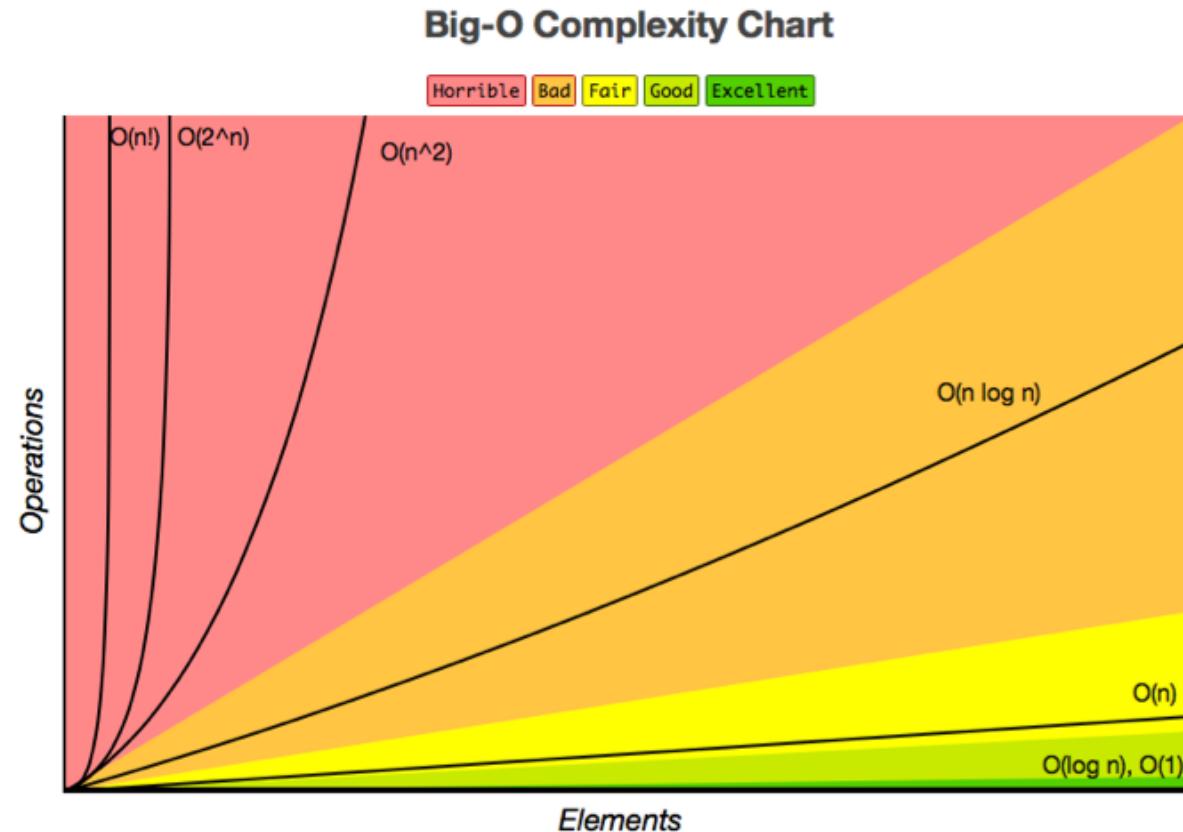
Common Data Structure Operations

Data Structure	Time Complexity								Space Complexity	
	Average				Worst					
	Access	Search	Insertion	Deletion	Access	Search	Insertion	Deletion		
Array	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	
Stack	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	
Queue	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	
Singly-Linked List	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	
Doubly-Linked List	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	
Skip List	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n \log(n))$	
Hash Table	N/A	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	N/A	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	
Binary Search Tree	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	
Cartesian Tree	N/A	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	N/A	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	
B-Tree	$\Theta(\log(n))$	$\Theta(n)$								
Red-Black Tree	$\Theta(\log(n))$	$\Theta(n)$								
Splay Tree	N/A	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	N/A	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(n)$	
AVL Tree	$\Theta(\log(n))$	$\Theta(n)$								
KD Tree	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	

Array Sorting Algorithms

Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst	
Quicksort	$\Omega(n \ log(n))$	$\Theta(n \ log(n))$	$\Theta(n^2)$	$\Theta(\log(n))$
Mergesort	$\Omega(n \ log(n))$	$\Theta(n \ log(n))$	$\Theta(n \ log(n))$	$\Theta(n)$
Timsort	$\Omega(n)$	$\Theta(n \ log(n))$	$\Theta(n \ log(n))$	$\Theta(n)$
Heapsort	$\Omega(n \ log(n))$	$\Theta(n \ log(n))$	$\Theta(n \ log(n))$	$\Theta(1)$
Bubble Sort	$\Omega(n)$	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(1)$
Insertion Sort	$\Omega(n)$	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(1)$
Selection Sort	$\Omega(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(1)$
Tree Sort	$\Omega(n \ log(n))$	$\Theta(n \ log(n))$	$\Theta(n^2)$	$\Theta(n)$
Shell Sort	$\Omega(n \ log(n))$	$\Theta(n(\log(n))^2)$	$\Theta(n(\log(n))^2)$	$\Theta(1)$
Bucket Sort	$\Omega(n+k)$	$\Theta(n+k)$	$\Theta(n^2)$	$\Theta(n)$
Radix Sort	$\Omega(nk)$	$\Theta(nk)$	$\Theta(nk)$	$\Theta(n+k)$
Counting Sort	$\Omega(n+k)$	$\Theta(n+k)$	$\Theta(n+k)$	$\Theta(k)$
Cubesort	$\Omega(n)$	$\Theta(n \ log(n))$	$\Theta(n \ log(n))$	$\Theta(n)$

BIG O CHARTS VIA ERIC ROWELL



Classical computational complexity: a review

- ▶ To understand the possible power of quantum computing, it helps to look at the computational power of quantum computers in relation to their classical counterparts
- ▶ Remember that problems in P are decision problems that can be solved in polynomial time by a deterministic Turing machine
- ▶ The equivalent class for space efficiency is referred to as PSPACE
- ▶ NP problems are those that require a nondeterministic Turing machine in order to be solved efficiently
- ▶ The class of NP-complete problems, abbreviated NPC, consists of the hardest problems in NP
 - ▶ Every problem in NP can be reduced to a problem in NPC
 - ▶ If one NPC problem was found to be in P, then all of the problems in NP would also be in P, proving $P = NP$
 - ▶ Most theoretical computer scientists believe that $P \neq NP$, but nobody has been successful in proving the conjecture either way.

Classical probabilistic complexity

- ▶ There is another important complexity class called BPP: Bounded-error Probabilistic Polynomial time
- ▶ BPP describes decision problems that can be solved in polynomial time by a *probabilistic* Turing machine
- ▶ Probabilistic Turing machines are those with direct access to some source of truly random input
- ▶ In BPP, the error of the solution is bounded in that the probability that the answer is correct must be at least two-thirds
- ▶ Although there are currently problems solvable in BPP that are not in P, the number of such problems has been decreasing since the introduction of BPP in the 1970's
- ▶ While it is not yet been proven whether $P \subset BPP$, it is conjectured that $P = BPP$



Quantum computational complexity

- ▶ Quantum computation introduces a number of new complexity classes to the polynomial hierarchy
- ▶ Probably the most studied complexity class is Bounded-error Quantum Polynomial time, or BQP
- ▶ BQP is the quantum extension of BPP: the class of decision problems solvable in polynomial time by an innately probabilistic quantum Turing machine, with the same error constraint as defined for BPP
- ▶ Unlike BPP, it is suspected that $P \subset BQP$, which would mean that quantum computers are capable of solving some problems in polynomial time that cannot be solved efficiently by a classical Turing machine!

A conjectured polynomial hierarchy

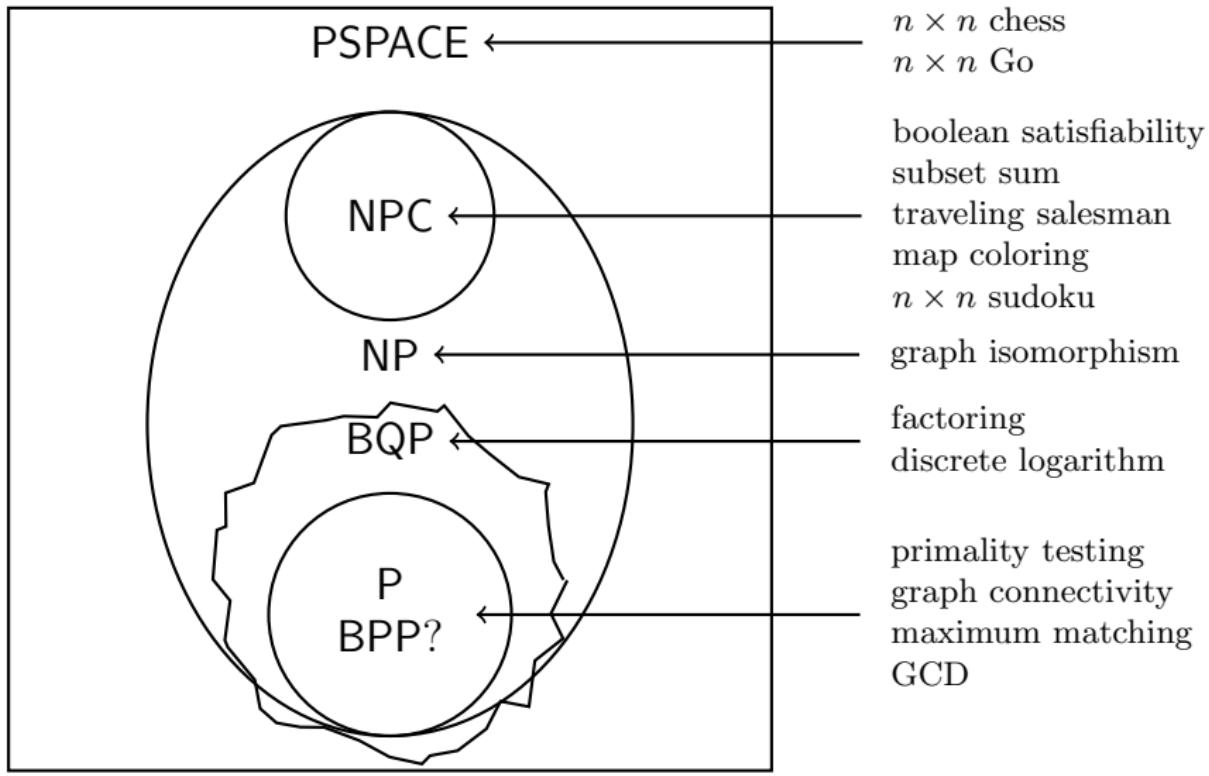
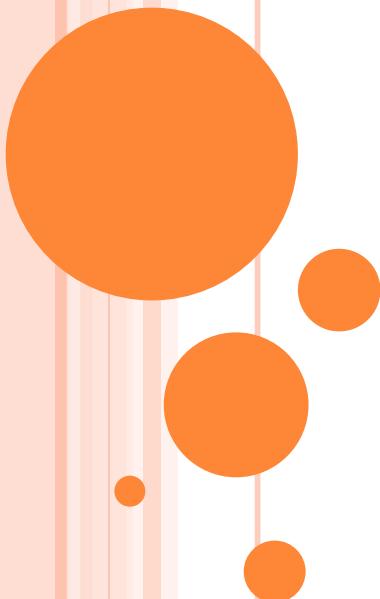


Image Credits

- ▶ Quantum Computer:
<http://www.wired.com/wiredscience/2010/01/quantum-computer-hydrogen-simulation/>
- ▶ Richard Feynman:
<http://www-scf.usc.edu/~kallos/feynman.htm>
- ▶ Peter Shor:
<http://www-math.mit.edu/~shor/>
- ▶ Cooling system for D-wave's quantum computer: <http://mail2web.com/blog/wp-content/uploads/2007/03/d-wave-quantum-computer-cryopump.png>
- ▶ Shrodinger's cat:
<http://confidentlysingle.com/2010/10/schrodingers-cat/>
- ▶ David Deutsch:
<http://datapeak.net/computerscientists.htm>
- ▶ Pauli & friends:
http://scienceblogs.com/startswithabang/2010/10/the_story_of_the_neutrino.php
- ▶ Tommaso Toffoli:
<http://pm1.bu.edu/~tt/>
- ▶ John T. Gill III:
<http://riddles.stanford.edu/gill/>

ENTANGLEMENT I

by
Robert Nemiroff



Physics X: About This Course

- Officially "Extraordinary Concepts in Physics"
- Being taught for credit at Michigan Tech
 - Light on math, heavy on concepts
 - Anyone anywhere is welcome
- No textbook required
 - Wikipedia, web links, and lectures only
 - Find all the lectures with Google at:
 - "Starship Asterisk" then "Physics X"
 - <http://bb.nightskylive.net/asterisk/viewforum.php?f=39>

ENTANGLEMENT: OVERVIEW

Two particles can be considered entangled when a quantum state of two particles are linked.

Can occur for particles that were

- created at the same place and time.
- collided at some place and time.

Typically involves

- spin
- momentum



ENTANGLEMENT: POSITRONIUM

Positronium

- electron and a positron orbiting
- decays quickly into two daughter photons

Daughter photons of Positronium have

- entangled spins (opposite)
- entangled momenta (opposite)



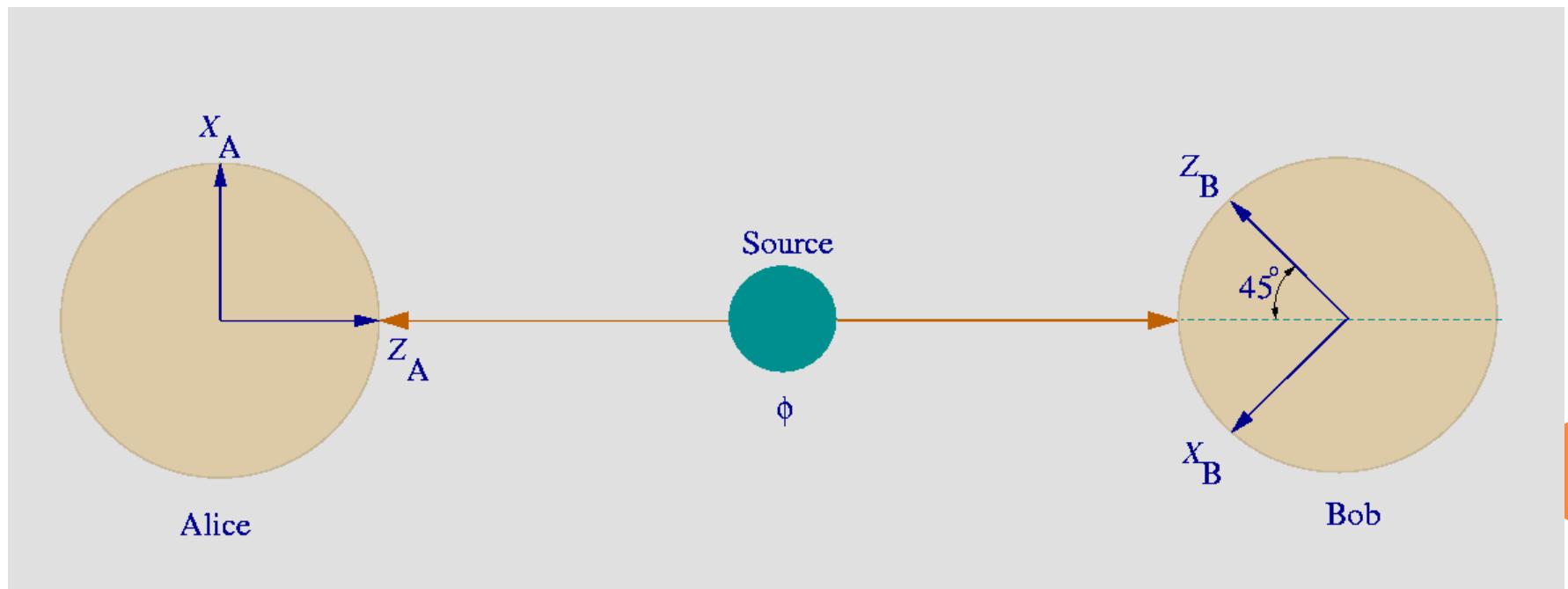
ENTANGLEMENT: EPR PARADOX

- 1935 paper by Einstein, Podolsky, and Rosen (EPR)
- Thought by EPR to show that
 - QM is incomplete
 - hidden variables must exist
 - otherwise QM "spooky action at a distance"
- origin of "entangled particles" idea
- Basis for Bell's Inequality test of QM versus HV
- QM statistics win



ENTANGLEMENT: EPR PARADOX

- Simple version of EPR experiment:
- A source decays into two particles (entangled)
 - one goes to Alice ("A"), the other to Bob ("B")
 - the total angular momentum is zero



ENTANGLEMENT: EPR PARADOX

If Alice measures the spin of her particle to be "spin up" on the z-axis, will Alice know that Bob will measure his entangled particle to be "spin down" on the z-axis before Bob knows it?

1. Yes, Alice is that smart.
2. No, that would require FTL communication.
3. Yes, but only in a statistical sense.
4. Depends on how much Alice is paying her psychic.



ENTANGLEMENT: EPR PARADOX

1. Yes, Alice is that smart.

In fact,

- If Alice and Bob both measure z axis spin
 - will ALWAYS get opposite z spin
- If Alice and Bob both measure x axis spin
 - will ALWAYS get opposite x spin



ENTANGLEMENT: EPR PARADOX

Does this, by itself, mean that Alice and Bob can communicate FTL?

1. Yes.
2. No.
3. Maybe so.



ENTANGLEMENT: EPR PARADOX

2. No.

It could be that the particles were created each having opposite spins. This could have been done with tennis balls -- this correlation requires only conservation of momentum, not FTL communication. That particles are created with attributes that are then frozen is part of the Hidden Variables (HV) interpretation of quantum mechanics.



ENTANGLEMENT: EPR PARADOX

Next, Alice measures her particle as "spin up" on z-axis and Bob measures his particle's spin about the x-axis. What spin will Bob measure?

1. Spin down, the opposite of Alice.
2. Spin up, the same as Alice.
3. Half the time spin up, half the time spin down.
4. Bob is getting dizzy.



ENTANGLEMENT: EPR PARADOX

3. Half the time spin up, half the time spin down.

Bob measurement is independent of Alice's measurement. Bob cannot tell from his measurements that Alice even exists.

Furthermore: Bob will always measure 50/50 no matter what or when Alice does her 50/50 measurements. Same with Alice.

Correlations between Alice and Bob can only be found retrospectively.



ENTANGLEMENT: LOCALITY

The principle of locality states that objects can only be affected by their immediate surroundings.

Entanglement appears to violate locality but not in such a way that allows distant observers to send information to each other faster than the speed of light.

