Intro to Quantum Computing

Lecture 1: Theory and the gate model

May 9, 2019

Olivia Di Matteo

Quantum Information Science Associate, TRIUMF

Overview

Purpose of these lectures

A non-sensational introduction to quantum computing, and setting realistic expectations for the near term; the technology is progressing rapidly, so now is the time to learn!

Purpose of these lectures

A non-sensational introduction to quantum computing, and setting realistic expectations for the near term; the technology is progressing rapidly, so now is the time to learn!

Conceptual topics

- Motivation behind quantum computing
- Different quantum computational paradigms
- Identifying suitable problems and understanding how they are mapped to quantum hardware
- Quantum advantage
- Major players, and successes and challenges of current-generation machines

Purpose of these lectures

Technical topics

- Qubits: gates, and measurement, entanglement
- Computation: gate model vs. annealing, key algorithms
- High-level overview of current-generation hardware
- Case studies of specific HEP applications
- (Bonus/optional) Getting started with IBM's Qiskit and D-Wave's Ocean Python libraries

Motivation

Physical limitations

'Classical' computers are made more powerful by

- making smaller transistors
- putting more transistors onto a single chip
- using multiple processors in parallel

Physical limitations

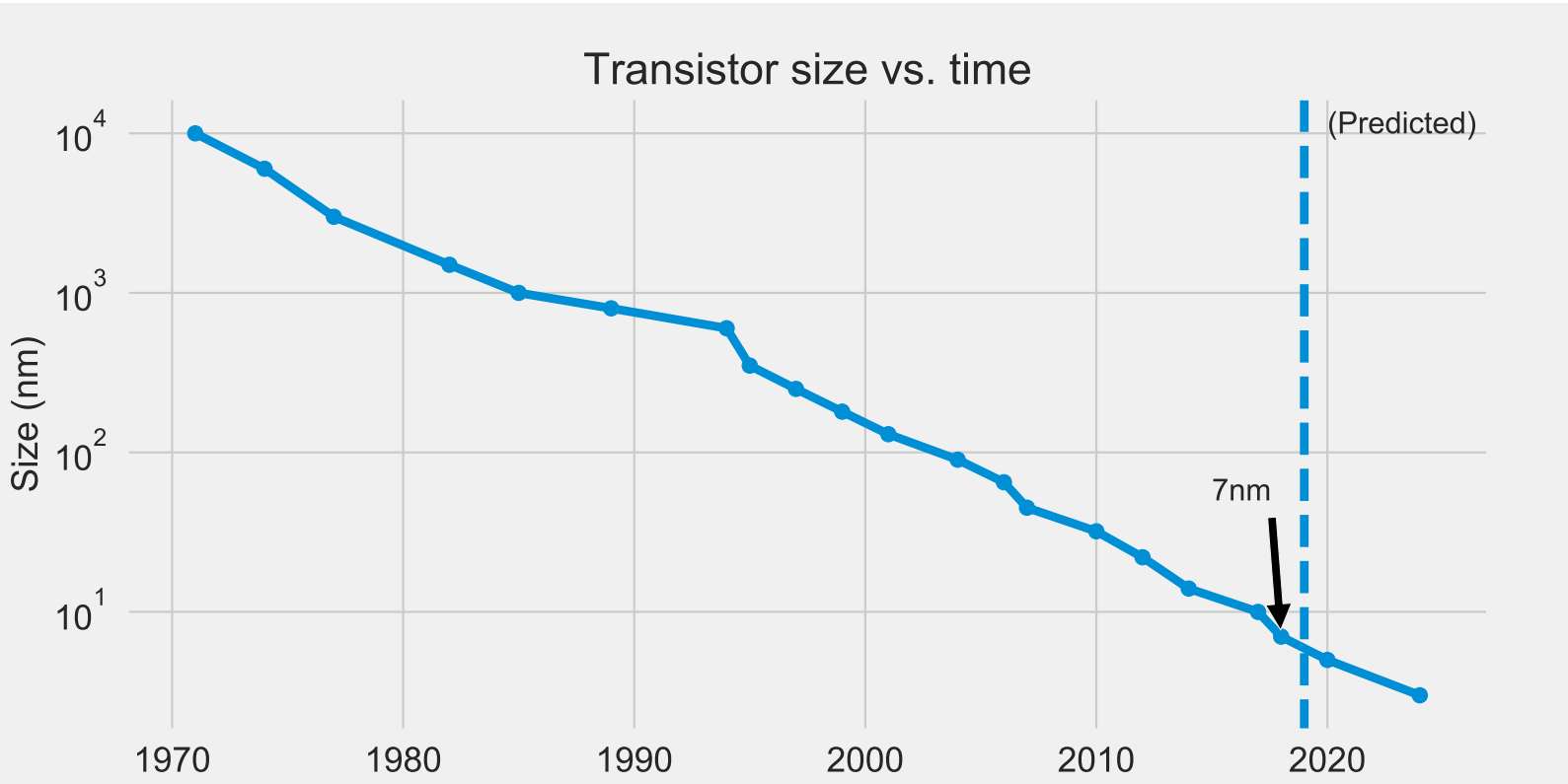
'Classical' computers are made more powerful by

- making smaller transistors
- putting more transistors onto a single chip
- using multiple processors in parallel

Moore's law suggests that every 18 months - 2 years the number of transistors on a chip will double, while the cost is halved.

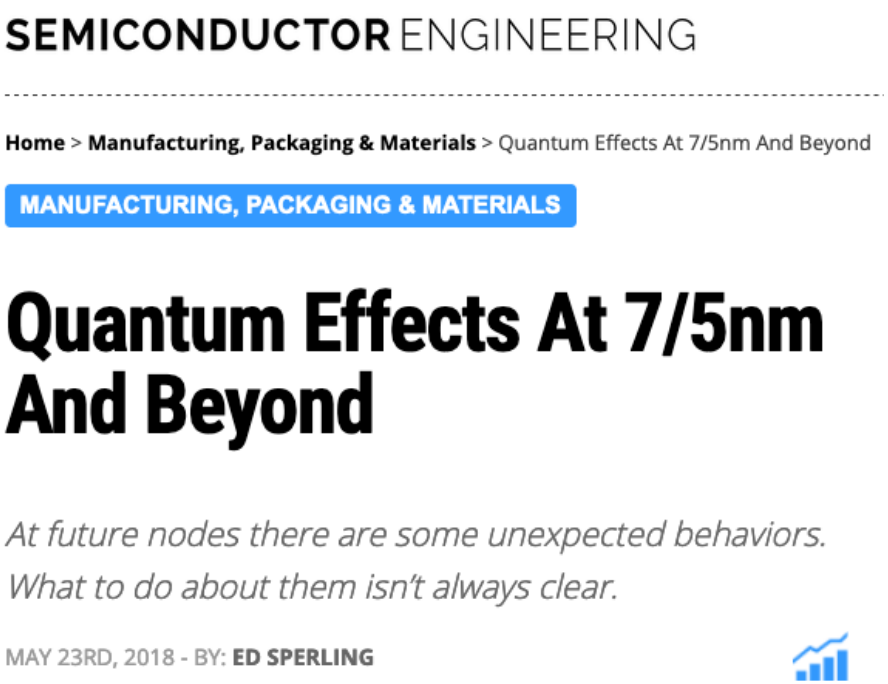
Why quantum computing?

But we are approaching the physical limits of how small a transistor we can put on a chip before quantum effects (tunneling) will become a problem.



Data source: `https://en.wikipedia.org/wiki/Semiconductor_device_fabrication`

Why quantum computing?



SEMICONDUCTOR ENGINEERING

Home > Manufacturing, Packaging & Materials > Quantum Effects At 7/5nm And Beyond

MANUFACTURING, PACKAGING & MATERIALS

Quantum Effects At 7/5nm And Beyond

At future nodes there are some unexpected behaviors. What to do about them isn't always clear.

MAY 23RD, 2018 - BY: **ED SPERLING**

Manufacturing of devices with 7nm chips began late in 2018.

Plans for 5nm to be released 2019-2020; design specs available as recently as April 2019.

Why quantum computing?

SEMICONDUCTOR ENGINEERING

Home > Manufacturing, Packaging & Materials > Quantum Effects At 7/5nm And Beyond

MANUFACTURING, PACKAGING & MATERIALS

Quantum Effects At 7/5nm And Beyond

At future nodes there are some unexpected behaviors. What to do about them isn't always clear.

MAY 23RD, 2018 - BY: **ED SPERLING**

Manufacturing of devices with 7nm chips began late in 2018.

Plans for 5nm to be released 2019-2020; design specs available as recently as April 2019.

Chips with transistors smaller than 7nm have required new and costly fabrication techniques; unclear whether anything smaller than 3nm is viable.

Computational limitations

Some problems will take an intractable amount of time to run on classical computers.

Parallelization can help, but we still cannot fully counteract the exponential complexity of some problems.

Computational limitations

Some problems will take an intractable amount of time to run on classical computers.

Parallelization can help, but we still cannot fully counteract the exponential complexity of some problems.

Sometimes that's a good thing:

- cryptographic infrastructure is built on such mathematically hard problems

Why quantum computing?

But usually it just prevents us from doing interesting things:

- solving complex optimization problems
- simulation of molecules and quantum systems
- searching large spaces
- machine learning with large amounts of data

What is quantum computing?

Quantum computation is the manipulation of the state of a physical quantum system in order to solve a problem.

What is quantum computing?

Quantum computation is the manipulation of the state of a physical quantum system in order to solve a problem.

Many quantum algorithms use 2-level quantum systems called *qubits* to solve problems more efficiently than the best-known classical algorithms:

- Quantum Fourier transform
- Shor's factoring algorithm
- Grover search algorithm
- Hamiltonian simulation
- Some linear algebra and machine learning algorithms

What's so good about qubits?

- The size of the mathematical space grows *exponentially* in the number of qubits

What's so good about qubits?

- The size of the mathematical space grows *exponentially* in the number of qubits
- We can make linear combinations of all possible qubit states, i.e. we can put them in *superposition*

What's so good about qubits?

- The size of the mathematical space grows *exponentially* in the number of qubits
- We can make linear combinations of all possible qubit states, i.e. we can put them in *superposition*
- We can *entangle* multiple qubits, and use these states as a resource in many algorithms

Single-qubit systems

From bits to qubit I: bits

All computation in our computers today is done with bits.

The state* of a bit is *either* 0 or 1.

A set of n bits is represented by n real numbers (0s and 1s).

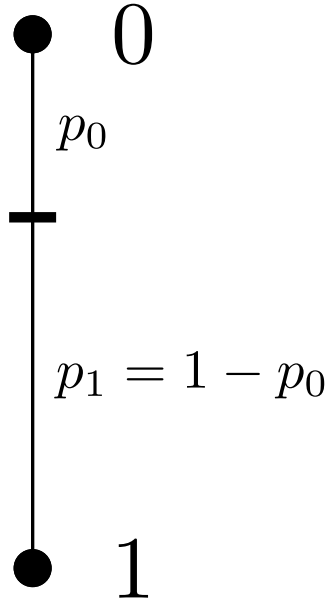
- 0
- 1

*Physically this is represented by some voltage and its state depends on whether or not that voltage is above/below a threshold value

From bits to qubit II: probabilistic bits

The value of a bit of information can be governed by a probability distribution.

For example, the outcome of a coin flip is a single bit of information, but before it is flipped it is probabilistic bit with both outcomes equally likely.



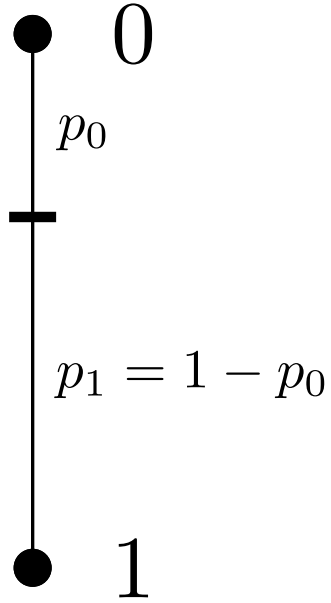
From bits to qubit II: probabilistic bits

We can assign a probabilistic bit a state based on its probability distribution:

$$\psi = \begin{pmatrix} p_0 \\ p_1 \end{pmatrix} \qquad (1)$$

where p_0 and p_1 are real numbers and we must have

$$p_0 + p_1 = 1. \qquad (2)$$



From bits to qubit II: probabilistic bits

We can transform one probability distribution to another using a stochastic matrix,

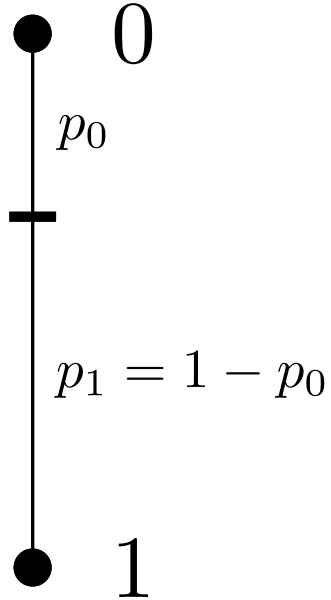
$$A = \begin{pmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{pmatrix} \qquad (3)$$

where all $a_{ij} \in [0, 1]$, and the sum of every row is 1.

Then

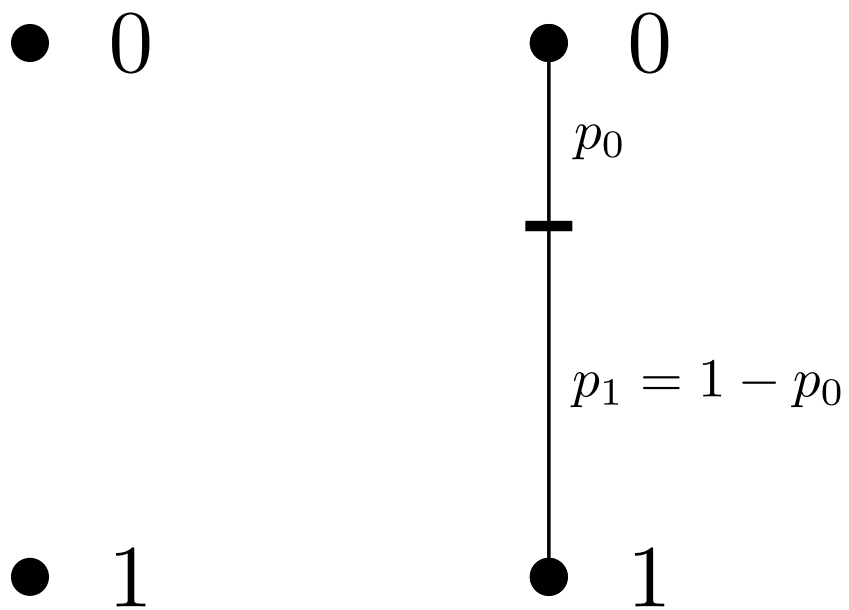
$$\begin{pmatrix} q_0 \\ q_1 \end{pmatrix} = A \begin{pmatrix} p_0 \\ p_1 \end{pmatrix} \qquad (4)$$

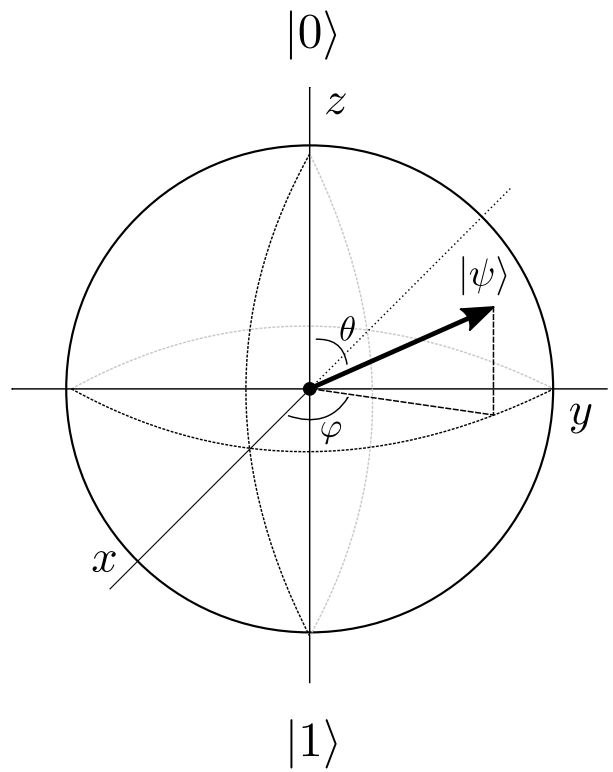
and $q_0 + q_1 = 1$.



From bits to qubit III: qubits

Extension of probabilistic bits to 2-level quantum systems:





Mathematical representation of qubits

Extension of probabilistic bits to 2-level quantum systems.

Instead of

$$\psi = \begin{pmatrix} p_0 \\ p_1 \end{pmatrix} \tag{5}$$

where $p_0, p_1 \in \mathbb{R}$ and $p_0 + p_1 = 1$, we have

$$|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \tag{6}$$

where $\alpha, \beta \in \mathbb{C}$, and $|\alpha|^2 + |\beta|^2 = 1$.

Mathematical representation of qubits

A qubit state is a unit vector that lives in a 2-dimensional complex vector space called *Hilbert space*.

Mathematical representation of qubits

A qubit state is a unit vector that lives in a 2-dimensional complex vector space called *Hilbert space*.

Qubit states are written as a linear combination, or *superposition*, of an *orthonormal basis* of the Hilbert space. The most commonly used one is the **computational basis**:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{7}$$

Then,

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad \alpha, \beta \in \mathbb{C}, \tag{8}$$

where unit length is ensured by having $|\alpha|^2 + |\beta|^2 = 1$.

Physical intuition: Spin 1/2 particles

This should look familiar from introductory QM: spin 1/2 particles are just like qubits!

$$|\uparrow\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |\downarrow\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{9}$$

where general spin states look like

$$|\psi\rangle = \alpha|\uparrow\rangle + \beta|\downarrow\rangle, \tag{10}$$

with $\alpha, \beta \in \mathbb{C}, |\alpha|^2 + |\beta|^2 = 1$.

Using qubits for computation

The amplitudes are the key here - when we measure the qubits at the end of our computation, they are what determine the outcome frequencies. We should make them meaningful to the problem.

But for this we need a way of:

1. putting qubits into superposition
2. manipulating the qubit to get the amplitudes that we want
3. combining multiple qubits
4. measuring the qubits to get the answer

Using qubits for computation

The amplitudes are the key here - when we measure the qubits at the end of our computation, they are what determine the outcome frequencies. We should make them meaningful to the problem.

But for this we need a way of:

1. putting qubits into superposition
2. manipulating the qubit to get the amplitudes that we want
3. combining multiple qubits
4. measuring the qubits to get the answer

How do we (mathematically) perform operations on qubits?

Unitary operations

Single-qubit states are manipulated by 2×2 unitary matrices, i.e. elements of $U(2)$. U is a unitary matrix if

$$U^\dagger U = U U^\dagger = \mathbb{1} \tag{11}$$

Unitary operations preserve the length of vectors, i.e. *maintain the normalization of qubit states*.

Unitary operations act *linearly* on superpositions:

$$U(\alpha|0\rangle + \beta|1\rangle) = \alpha U|0\rangle + \beta U|1\rangle \tag{12}$$

This is a key contributor to the power of quantum computing!

Creating a superposition: the Hadamard gate

Perhaps the most important[1] unitary in quantum computing is the Hadamard gate:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{13}$$

This gate creates and undoes a *uniform superposition*:

$$\begin{aligned} H|0\rangle &= \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) = |+\rangle \\ H|1\rangle &= \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) = |-\rangle \\ H|+\rangle &= |0\rangle \\ H|-\rangle &= |1\rangle \end{aligned}$$

[1] You'll see more how important it is when we talk about multi-qubit systems and quantum algorithms!

Single-qubit gates: Paulis

You are all familiar with the Pauli operators, X, Y, and Z:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{14}$$

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{15}$$

$$Y = iZX = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{16}$$

These are unitary, and can do some very useful things to qubit states.

Single-qubit gates: Paulis

X is called the **bit flip** operation:

$$X|0\rangle = |1\rangle, \tag{17}$$
$$X|1\rangle = |0\rangle \tag{18}$$

Z is called the **phase flip** operation:

$$Z|0\rangle = |0\rangle, \tag{19}$$
$$Z|1\rangle = -|1\rangle \tag{20}$$

Single-qubit gates: applying sequences of gates

We apply *products* of single-qubit operations to represent performing gates in sequence. For example,

$$\begin{aligned} XZH|0\rangle &= XZ\left(\frac{|0\rangle}{\sqrt{2}} + \frac{|1\rangle}{\sqrt{2}}\right) \\ &= X\left(\frac{|0\rangle}{\sqrt{2}} - \frac{|1\rangle}{\sqrt{2}}\right) \\ &= \frac{|1\rangle}{\sqrt{2}} - \frac{|0\rangle}{\sqrt{2}} \end{aligned}$$

Products are applied from right to left.

Single-qubit gates: rotations

X, Y, and Z are special cases of more general qubit rotations:

$$R_x(\theta) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{21}$$

$$R_y(\theta) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{22}$$

$$R_z(\theta) = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{pmatrix} \tag{23}$$

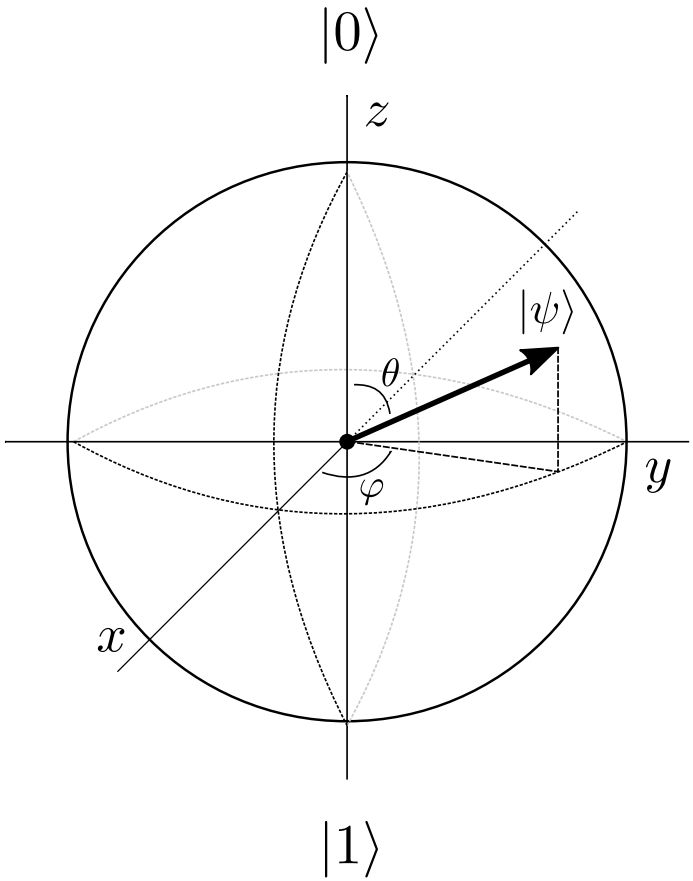
$X = R_x(\pi), \;\; Y = R_y(\pi), \;\; Z = R_z(\pi).$

R_x and R_y can put qubits into *non-uniform* superpositions.

... wait. Rotations? Rotations around *what*?

Visualizing a qubit: the Bloch sphere

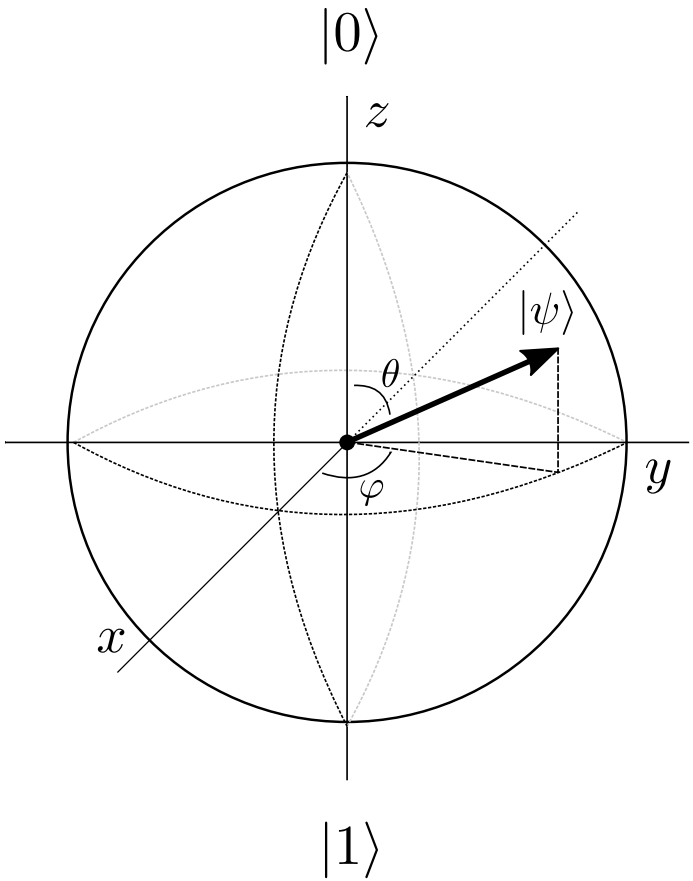
Single-qubit ket states can be represented on the surface of a sphere of radius 1 called the *Bloch sphere*.



Visualizing a qubit: the Bloch sphere

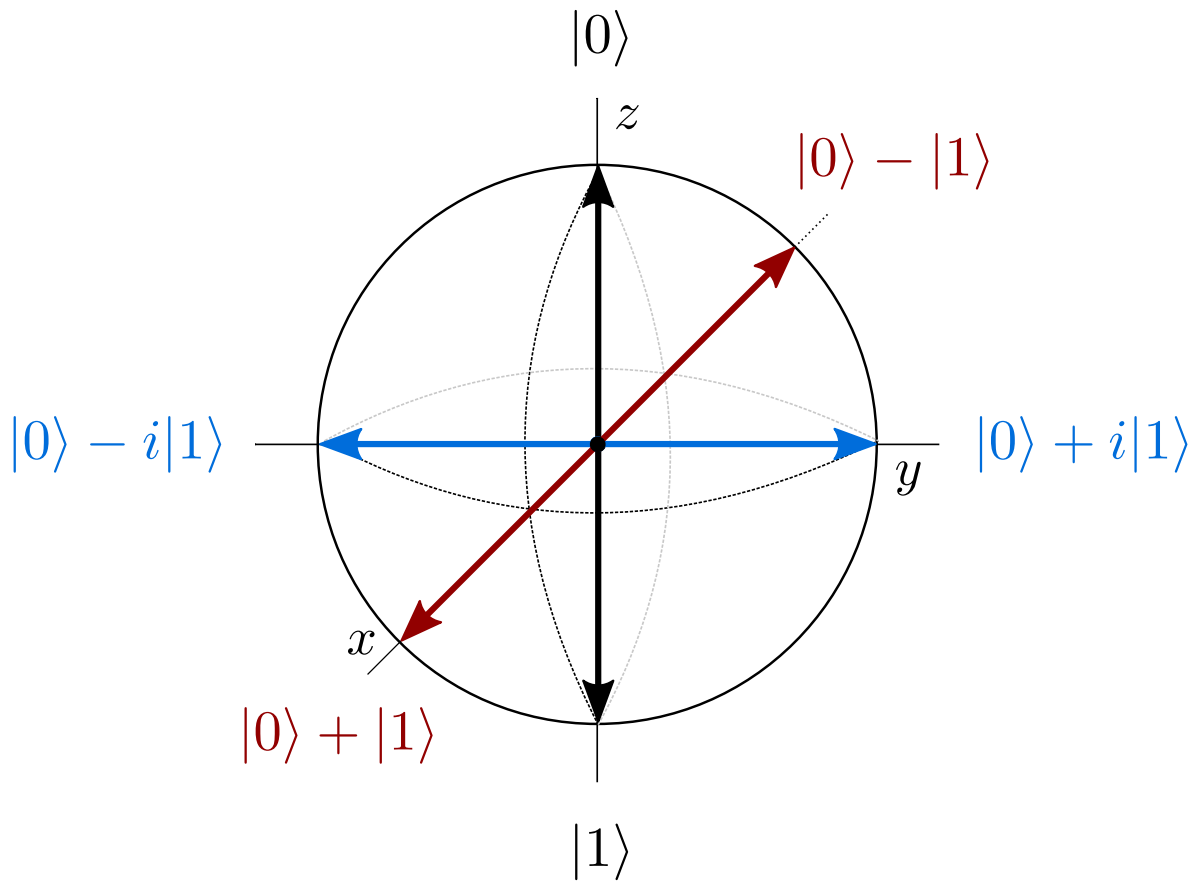
States can be plotted on the sphere using a parameterized form:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\varphi}\sin\left(\frac{\theta}{2}\right)|1\rangle \tag{24}$$



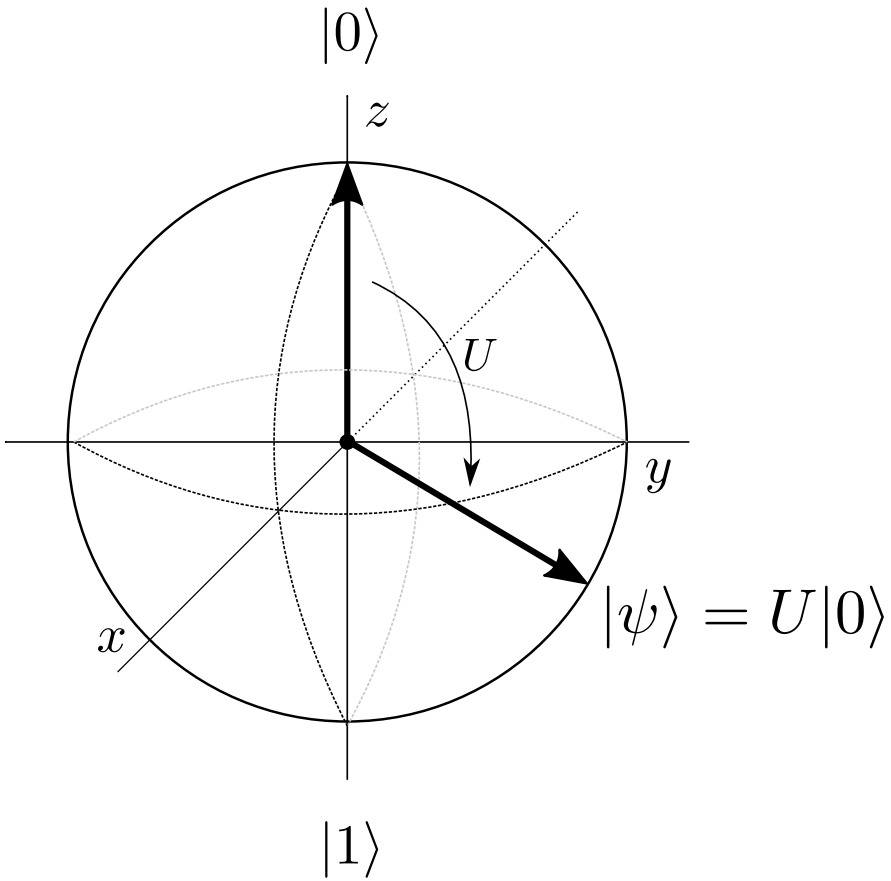
Visualizing a qubit: the Bloch sphere

The axis points correspond to the *eigenstates* of σ_x, σ_y, and σ_z.



Unitary operations

Unitary matrices rotate state vectors around the Bloch sphere



Let's watch some of them in action!

Single-qubit gates: complex phase gates

There are two more very important single-qubit gates:

Gate	Unitary	Action
Phase gate $S = \sqrt{Z}$	$\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$	$1/4$ turn around z-axis
T gate, $T = \sqrt{S}$	$\begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$	$1/8$ turn around z-axis

These gates change the *relative phase* between $|0\rangle$ and $|1\rangle$. They don't affect the magnitudes of the amplitudes, but we will see that they *do* affect the measurement statistics!

Measurement

We know now that *unitary operations* can be used to manipulate our qubits to perform a computation. But after we're done computing, how do we get the answer?

We need a mathematical formalism for measuring qubits.

This could be a whole lecture series on its own, so I will keep it simple and cover only projective measurements, which you are probably familiar with from QM.

Projective measurements

A Hermitian matrix Π is a *projector* if $\Pi^2 = \Pi$.

A *projective measurement* is a set of projectors $\mathcal{B} = \{\Pi_i\}_{i=0}^{M}$ where

$$\sum_{i=0}^{M} \Pi_i = \mathbb{1} \tag{25}$$

For every single-qubit ket state $|\psi\rangle$, the 2×2 matrix $|\psi\rangle \langle\psi|$ is a projector. If we take a set of orthonormal kets $\{|\psi_i\rangle\}$ (i.e. a basis), together they form a projective measurement.

For a single qubit,

$$|\psi_0\rangle \langle\psi_0| + |\psi_1\rangle \langle\psi_1| = \mathbb{1} \tag{26}$$

Projective measurements

When we make a projective measurement on a state $|\varphi\rangle$ in basis $\{|\psi_i\rangle\}$, the probability of obtaining outcome i is

$$\begin{aligned} \Pr(\text{outcome i}) &= \mathrm{Tr}(\Pi_i |\varphi\rangle \langle\varphi|) \\ &= |\langle\psi_i|\varphi\rangle|^2 \end{aligned}$$

If we observe outcome i, following the measurement the system will be left in state $|\psi_i\rangle$[2].

[2] Actually things are a bit more subtle than this; for a good overview of projective measurements, see `https://www.people.vcu.edu/~sgharibian/courses/CMSC491/notes/Lecture%203%20-%20Measurement.pdf`

Projective measurements

Example: measuring in the computational basis

Let $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$.

The computational basis $\{|0\rangle, |1\rangle\}$ is a projective measurement:

$$|0\rangle\langle 0| + |1\rangle\langle 1| = \mathbb{1} \tag{27}$$

Then if we measure $|\psi\rangle$,

$$\begin{aligned} \Pr(0) &= |\langle 0|\psi\rangle|^2 = |\alpha|^2 \\ \Pr(1) &= |\langle 1|\psi\rangle|^2 = |\beta|^2 \end{aligned}$$

Basis changes

We can measure in any orthonormal basis by applying a suitable unitary transformation to the computational basis vectors.

Example: measuring in the *Hadamard basis*:

$$\begin{aligned} |+\rangle &= H|0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \\ |-\rangle &= H|1\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right) \end{aligned}$$

You can check that $|+\rangle\langle+| + |-\rangle\langle-| = \mathbb{1}$ is indeed a valid projective measurement.

Basis changes

Then, for $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$,

$$\begin{aligned}
\Pr(+) &= |\langle +|\psi\rangle|^2 \\
&= \frac{1}{2}|\alpha\langle 0|0\rangle + \alpha\langle 0|1\rangle + \beta\langle 1|0\rangle + \beta\langle 1|1\rangle|^2 \\
&= \frac{1}{2}|\alpha + \beta|^2 \\
\Pr(-) &= \frac{1}{2}|\alpha - \beta|^2
\end{aligned}$$

Measuring in the Hadamard basis

Why would we want to measure in different bases?

Example

Consider $|+\rangle$ and $|-\rangle$.

If we measure in the computational basis, for both states we will get 0 with probability $1/2$ and also 1 with probability $1/2$. It's impossible to know which state we have!

But if we measure in the Hadamard basis, we will get either *only* $+$ or *only* $-$.

Phase factors

Let's consider another set of orthonormal states:

$$|y_+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle) \tag{28}$$

$$|y_-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle) \tag{29}$$

Measuring in the computational basis gives us:

$$\Pr(0) = \Pr(1) = \frac{1}{2} \tag{30}$$

Meausring in the Hadamard basis gives us:

$$\Pr(+) = \Pr(-) = \frac{1}{2} \tag{31}$$

Phase factors

These states are indistinguishable in both bases! This might seem strange, since

$$|y_+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle) \tag{32}$$

$$|y_-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle) \tag{33}$$

look so similar to

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{34}$$

$$|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{35}$$

The *relative* phase between $|0\rangle$ and $|1\rangle$ clearly has a large effect on the state even though the magnitudes of the amplitudes in other bases don't change.

Global vs. relative phase

Changing the **global phase**

$$|\psi\rangle \to e^{i\gamma}|\psi\rangle \tag{36}$$

does not change the measurement statistics.

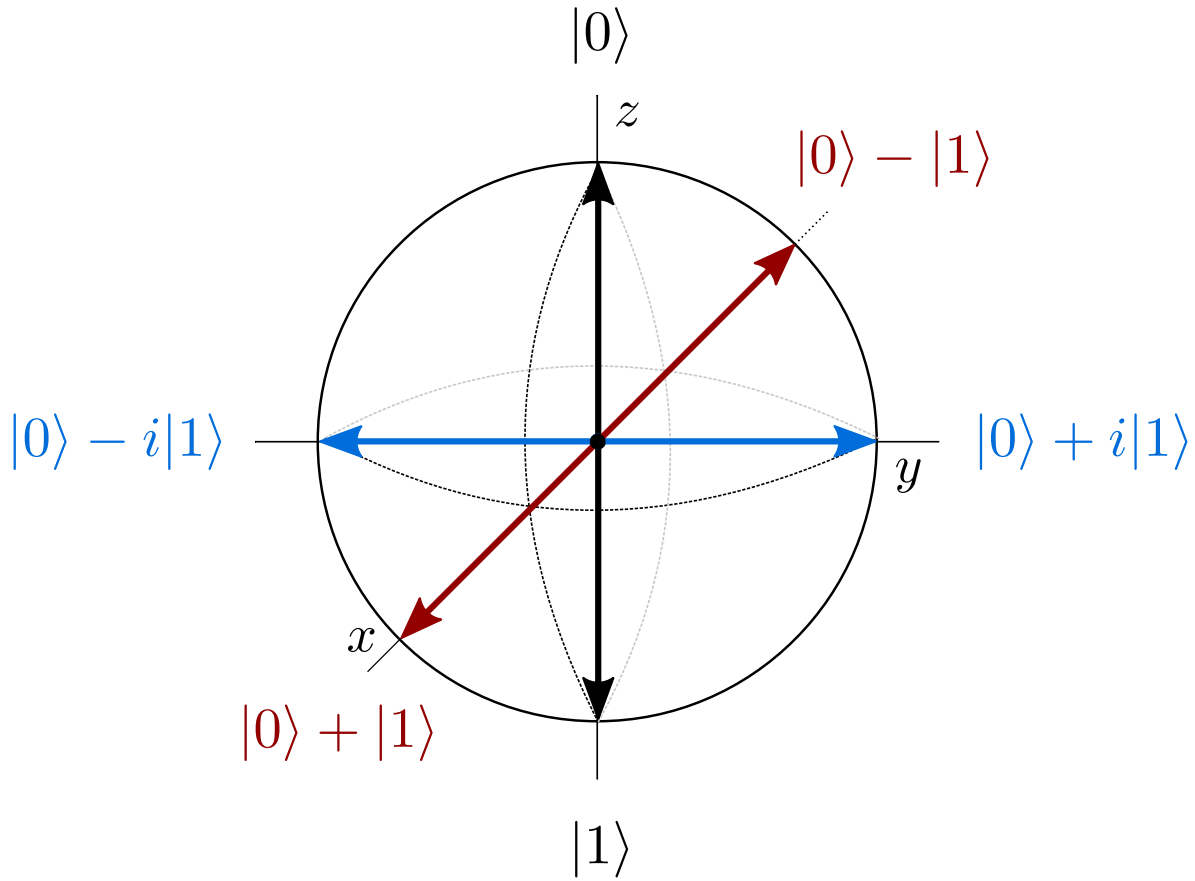
But changing the *relative phase*, i.e.

$$|\psi\rangle \to \alpha|0\rangle + e^{i\gamma}\beta|1\rangle \tag{37}$$

does change the measurement statistics, depending on which basis you are measuring in.

Global vs. relative phase

It is easier to see visually how these states are different - relative phases rotate us around the z axis of the Bloch sphere.



Multi-qubit systems

Hilbert spaces compose under the *tensor product*.

Example

Let

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}, \quad B = \begin{pmatrix} e & f \\ g & h \end{pmatrix}. \tag{38}$$

The tensor product of A and B, $A \otimes B$ is

$$A \otimes B = \begin{pmatrix} a\begin{pmatrix} e & f \\ g & h \end{pmatrix} & b\begin{pmatrix} e & f \\ g & h \end{pmatrix} \\ c\begin{pmatrix} e & f \\ g & h \end{pmatrix} & d\begin{pmatrix} e & f \\ g & h \end{pmatrix} \end{pmatrix} = \begin{pmatrix} ae & af & be & bf \\ ag & ah & bg & bh \\ ce & cf & de & df \\ cg & ch & dg & dh \end{pmatrix} \tag{39}$$

Qubit state vectors are also combined using the *tensor product*:

$$|01\rangle = |0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ 0 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \tag{40}$$

An n-qubit state is therefore a vector of length 2^n.

The tensor product is linear and distributive, so if we have

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad |\varphi\rangle = \gamma|0\rangle + \delta|1\rangle, \tag{41}$$

then they tensor together to form

$$\begin{aligned} |\psi\rangle \otimes |\varphi\rangle &= (\alpha|0\rangle + \beta|1\rangle) \otimes (\gamma|0\rangle + \delta|1\rangle) \\ &= \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma\ |10\rangle + \beta\delta|11\rangle \end{aligned}$$

The states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ are the computational basis vectors for 2 qubits; we can create arbitrary linear combinations of them as long as the normalization on the coefficients holds.

Single-qubit unitary operations also compose under tensor product.

For example, apply U_1 to qubit $|\psi\rangle$ and U_2 to qubit $|\varphi\rangle$:

$$(U_1 \otimes U_2)(|\psi\rangle \otimes |\varphi\rangle) = (U_1|\psi\rangle) \otimes (U_2|\varphi\rangle) \tag{42}$$

If an n-qubit ket is a vector with length 2^n, then a unitary acting on n qubits has dimension $2^n \times 2^n$.

Exercise: Suppose we have two qubits in the state $|01\rangle$. What state do we get when we apply H to the first qubit and S to the second qubit?

Exercise: Suppose we have two qubits in the state $|01\rangle$. What state do we get when we apply H to the first qubit and S to the second qubit?

Solution: We know:

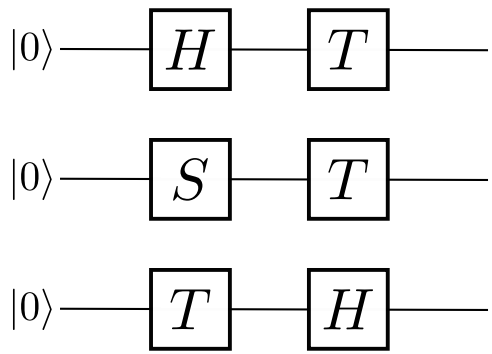
$$H|0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right), \quad S|1\rangle = i|1\rangle \tag{43}$$

So then

$$\begin{aligned}(H \otimes S)|01\rangle &= (H|0\rangle) \otimes (S|1\rangle) \\ &= \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \otimes i|1\rangle \\ &= \frac{i}{\sqrt{2}}\left(|01\rangle + |11\rangle\right)\end{aligned}$$

Quantum circuits

When we have multiple qubits, writing down matrices and vectors is tedious. Thankfully we have *quantum circuits*.



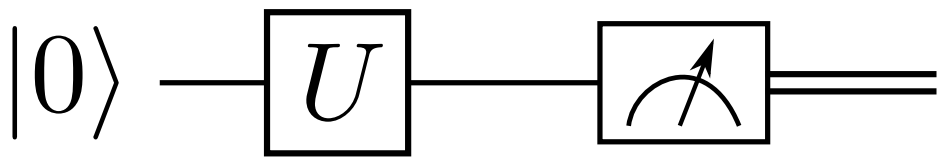
This circuit applies two 3-qubit unitaries,

$$U_1 = H \otimes S \otimes T \tag{44}$$

$$U_2 = T \otimes T \otimes H \tag{45}$$

The output state is $U_2 U_1 |000\rangle$; circuits are drawn in reverse order as the matrix products.

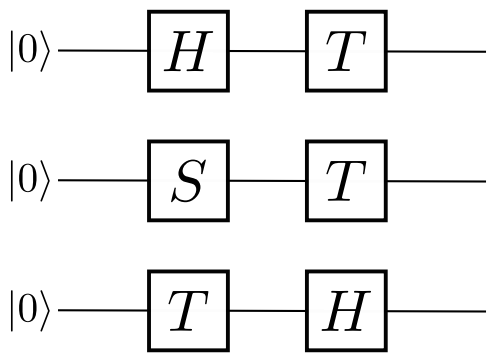
A measurement in a circuit is represented by a box with a dial:



The two wires coming out of it indicate a *classical bit* - the outcome of the measurement is not a qubit, it's either 0 or 1!

Quantum circuits

The *depth* of a quantum circuit is the largest number of circuit 'layers' that are applied to any qubit.



This circuit has depth 2.

Digression: simulating a quantum computer

You might be thinking: *"if everything is just linear algebra, why bother building a quantum computer at all?"*

For an n-qubit computation, we'd need to:

- Store 2^n complex numbers
- Store $2^n \times 2^n$ unitary matrices
- Multiply the two together repeatedly

This has **insane** time and memory requirements.

The best *full* quantum simulators running on a laptop can manage about 30 qubits with 16 GB RAM. The top supercomputers have managed ~ 50 qubits on circuits of depth ~ 40.

Exercise: Consider the 2-qubit state

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \tag{46}$$

Find

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad |\varphi\rangle = \gamma|0\rangle + \delta|1\rangle \tag{47}$$

such that

$$|\Psi\rangle = |\psi\rangle \otimes |\varphi\rangle \tag{48}$$

Exercise: Consider the 2-qubit state

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \tag{46}$$

Find

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad |\varphi\rangle = \gamma|0\rangle + \delta|1\rangle \tag{47}$$

such that

$$|\Psi\rangle = |\psi\rangle \otimes |\varphi\rangle \tag{48}$$

Solution: This is impossible (sorry!)

Entanglement

The state

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \tag{49}$$

is **entangled**.

We cannot describe the two qubits individually, we can only described their combined state.

Paraphrasing from John Preskill: *it's like you're reading a book, but instead of reading the pages sequentially, you have to read it all at the same time in order to understand it.*

Entanglement

Furthermore, the measurement outcomes of

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \tag{50}$$

are *perfectly correlated*.

For example, if I measure the first qubit and get 0, I'll get 0 for the second qubit as well!

Entanglement is not limited to two qubits. In principle we can entangle as many as we like:

$$|\Psi\rangle = |00\cdots0\rangle + |11\cdots1\rangle \tag{51}$$

A measurement outcome of 0 on qubit 1 means we'll get 0 on *all other qubits* too.

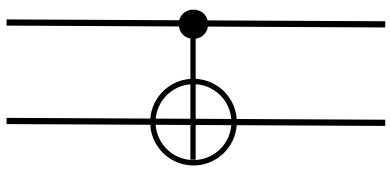
Entangling gates

How do we make an entangled state (in theory)? Previous 2-qubit operations we saw were expressed as tensor products of single-qubit ones.

There exist *entangling gates* that will turn a non-entangled, or separable, state into an entangled one. The most commonly used one is the controlled-NOT, or CNOT:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$\begin{aligned} \text{CNOT}|00\rangle &= |00\rangle \\ \text{CNOT}|01\rangle &= |01\rangle \\ \text{CNOT}|10\rangle &= |11\rangle \\ \text{CNOT}|11\rangle &= |10\rangle \end{aligned}$$



The first qubit is the *control* qubit - it controls whether or not an X (NOT) gate is applied to the second qubit.

Entangling gates: CNOT

Exercise: What happens when we apply a CNOT to qubits in state $|+\rangle \otimes |0\rangle$?

$$\begin{aligned}
\text{CNOT}\left[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle\right] &= \text{CNOT}\left[\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)\right] \\
&= \frac{1}{\sqrt{2}}(\text{CNOT}|00\rangle + \text{CNOT}|10\rangle) \\
&= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)
\end{aligned}$$

We've gone from a separable state to an entangled one!

Universal gate sets

Now that we've seen the CNOT gate, in principle *I don't need to introduce to you any additional single- or multi-qubit gates*.

That's a good thing - there are an infinite number of unitaries, and there's no way we can individually program each one into our quantum computing hardware.

Thankfully, the CNOT and a few single-qubit gates are all we need for universal quantum computation!

How many is "a few"?

Single-qubit universal gate set

If you have a quantum computer that can perform

$$\{H, \ T\} \tag{52}$$

then you can implement *any* other single-qubit unitary up to an arbitrary precision.

Universal gate sets

Single-qubit universal gate set

If you have a quantum computer that can perform

$$\{H,\ \ T\} \tag{52}$$

then you can implement *any* other single-qubit unitary up to an arbitrary precision.

Multi-qubit universal gate set

If you have a quantum computer that can perform

$$\{H,\ \ T,\ \ \mathrm{CNOT}\} \tag{53}$$

then you can implement *any* other multi-qubit unitary up to an arbitrary precision.

Universal gate sets

This is not the only set - many others exist, and your choice will likely depend on your hardware implementation:

- Single-qubit rotations and CNOT
- H, CNOT, Toffoli (controlled-CNOT, on 3 qubits)

[3] I work in this area, so come ask me about it!

Universal gate sets

This is not the only set - many others exist, and your choice will likely depend on your hardware implementation:

- Single-qubit rotations and CNOT
- H, CNOT, Toffoli (controlled-CNOT, on 3 qubits)

The process of decomposing arbitrary unitaries into these sequences is called *quantum circuit synthesis*[3] , or *quantum compilation*.

Your sequence might be very long, if you want to be precise (scales like $\log(1/\varepsilon)$ for precision ε), but it can always be done!

[3] I work in this area, so come ask me about it!

Quantum algorithms I

Classes of quantum algorithms

Quantum computers are are well-known for the 'big' algorithms, where they will achieve a notable speedup over classical computers:

- Grover's search algorithm
- Shor's factoring algorithm
- Hamiltonian simulation

We will discuss these in more detail tomorrow when we discuss hardware and applications.

Small quantum protocols

Today, I want to show you some 2-3 qubit protocols that produce results that are uniquely quantum:

- Superdense coding
- Quantum teleportation
- Deutsch's algorithm

First I will show you the theory behind it, then I will show you in Python that it actually works!

Superdense coding

Suppose Alice wants to send Bob two classical bits of information, say ‘1’ and ‘0’.

Q1: How many classical bits does she need to send to Bob to do this?

Superdense coding

Suppose Alice wants to send Bob two classical bits of information, say ‘1’ and ‘0’.

Q1: How many classical bits does she need to send to Bob to do this?
A1: 2.

Superdense coding

Suppose Alice wants to send Bob two classical bits of information, say ‘1’ and ‘0’.

Q1: How many classical bits does she need to send to Bob to do this?
A1: 2.

Q2: How many *qubits* does she need to send to Bob to do this?

Superdense coding

Suppose Alice wants to send Bob two classical bits of information, say ‘1’ and ‘0’.

Q1: How many classical bits does she need to send to Bob to do this?
A1: 2.

Q2: How many *qubits* does she need to send to Bob to do this?
A2: Only 1!

Superdense coding

Alice can send 2 bits with a single qubit if her and Bob share a pair of entangled qubits:

$$|\Phi\rangle_{AB} = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \tag{54}$$

Alice will perform some operations on her qubit depending on which bit she wants to send. Then she will send her qubit to Bob.

Once Bob receives Alice's qubit, he measure both qubits to determine the bits she wanted to send him.

Superdense coding

Alice and Bob start the protocol with this entangled state:

$$|\Phi\rangle_{AB} = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \tag{55}$$

Next, depending on her bits, Alice performs one of the following operations on her qubit:

$$00 \rightarrow I \tag{56}$$
$$01 \rightarrow X \tag{57}$$
$$10 \rightarrow Z \tag{58}$$
$$11 \rightarrow ZX \tag{59}$$

Superdense coding

What happened to the entangled state?

$$|\Phi\rangle_{AB} = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right)$$

It will transform to:

$$\begin{aligned}
00 \quad &\rightarrow_I \quad \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \\
01 \quad &\rightarrow_X \quad \frac{1}{\sqrt{2}}\left(|10\rangle + |01\rangle\right) \\
10 \quad &\rightarrow_Z \quad \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle\right) \\
11 \quad &\rightarrow_{ZX} \quad \frac{1}{\sqrt{2}}\left(-|10\rangle + |01\rangle\right)
\end{aligned}$$

Superdense coding

There is something special about these four states:

$$
\begin{aligned}
|\Phi_0\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \\
|\Phi_1\rangle &= \frac{1}{\sqrt{2}}\left(|10\rangle + |01\rangle\right) \\
|\Phi_2\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle\right) \\
|\Phi_3\rangle &= \frac{1}{\sqrt{2}}\left(-|10\rangle + |01\rangle\right)
\end{aligned}
$$

Any guesses?

Superdense coding

There is something special about these four states:

$$
\begin{aligned}
|\Phi_0\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \\
|\Phi_1\rangle &= \frac{1}{\sqrt{2}}\left(|10\rangle + |01\rangle\right) \\
|\Phi_2\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle\right) \\
|\Phi_3\rangle &= \frac{1}{\sqrt{2}}\left(-|10\rangle + |01\rangle\right)
\end{aligned}
$$

Any guesses?

Hint: Look at the inner products between the states.

This is a set of 4 orthonormal states - they form a basis in the 2-qubit Hilbert space. We call it the *Bell basis*.

$$
\begin{aligned}
|\Phi_0\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \\
|\Phi_1\rangle &= \frac{1}{\sqrt{2}}\left(|10\rangle + |01\rangle\right) \\
|\Phi_2\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle\right) \\
|\Phi_3\rangle &= \frac{1}{\sqrt{2}}\left(-|10\rangle + |01\rangle\right)
\end{aligned}
$$

Since this is an orthonormal basis, Bob can perform a measurement in it to determine with certainty which state he has, and correspondingly which bits Alice sent him.

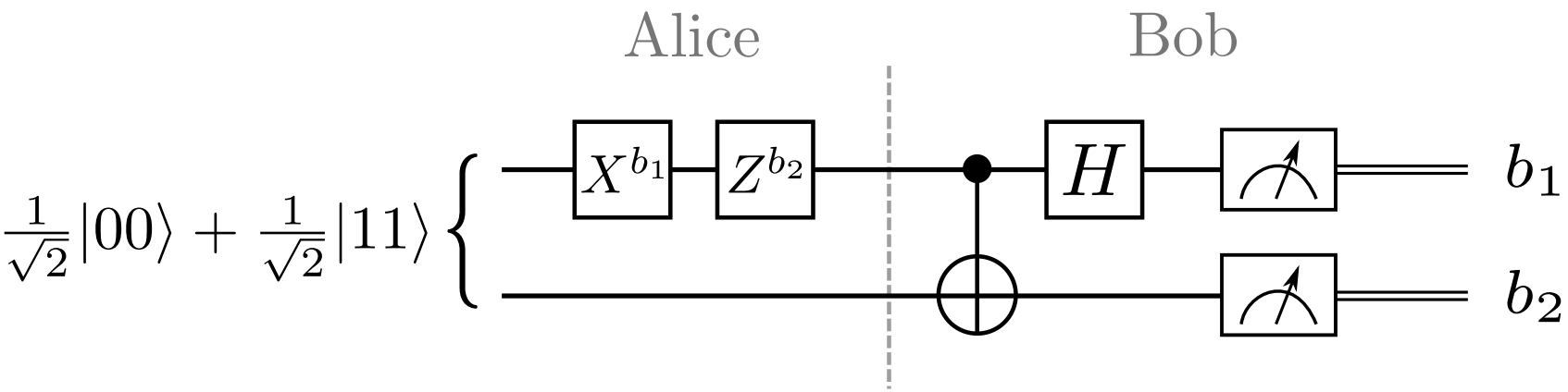
Superdense coding

Bob can measure directly in the Bell basis, or he can perform a basis transformation from the Bell basis back to the computational basis:

$$
\begin{aligned}
(H \otimes I)\mathrm{CNOT} \cdot \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) &= |00\rangle \\
(H \otimes I)\mathrm{CNOT} \cdot \frac{1}{\sqrt{2}}(|10\rangle + |01\rangle) &= |01\rangle \\
(H \otimes I)\mathrm{CNOT} \cdot \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) &= |10\rangle \\
(H \otimes I)\mathrm{CNOT} \cdot \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) &= |11\rangle
\end{aligned}
$$

Then he can measure in the computational basis to get the bits directly.

This is what it looks like as a circuit



Motivation: You are given access to an oracle that you know to be one of the following 4 functions:

Name	Action	Name	Action
f	$f(0) = 0$ $f(1) = 0$	g	$g(0) = 1$ $g(1) = 1$
φ	$\varphi(0) = 0$ $\varphi(1) = 1$	γ	$\gamma(0) = 1$ $\gamma(1) = 0$

Functions f and g are *constant* (same output no matter what the input), and φ and γ are balanced.

Oracle?

We often consider a quantum computer to behave like a black box, or an oracle - we don't care about the specifics of the circuit it implements, we just assume that it will perform some unitary U_f that will apply f and write the output in the state of some qubit:

$$U_f(|x\rangle|y\rangle) = |x\rangle|y \oplus f(x)\rangle \tag{60}$$

where $\oplus$ indicates addition modulo 2.

The *query complexity* of many quantum algorithms is expressed in terms of how many times we call the problem's oracle, i.e. apply U_f. For example, it linear in the number of qubits, polynomial, exponential, etc.

How many queries do you need to make to the oracle to determine if the function is constant or balanced? (i.e. either one of f/g, or one of φ/γ).

Name	Action	Name	Action
f	$f(0) = 0$ $f(1) = 0$	g	$g(0) = 1$ $g(1) = 1$
φ	$\varphi(0) = 0$ $\varphi(1) = 1$	γ	$\gamma(0) = 1$ $\gamma(1) = 0$

Classical solution: 2

We always need to query both inputs 0 and 1 to find out the nature of the function.

How many queries do you need to make to the oracle to determine if the function is constant or balanced? (i.e. either one of f/g, or one of φ/γ).

Name	Action	Name	Action
f	$f(0) = 0$ $f(1) = 0$	g	$g(0) = 1$ $g(1) = 1$
φ	$\varphi(0) = 0$ $\varphi(1) = 1$	γ	$\gamma(0) = 1$ $\gamma(1) = 0$

Quantum solution: 1
How???

Phase kickback

The secret lies in something called *phase kickback*.

What happens when we apply a CNOT to the following state?

$$|0\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right)$$

We get

$$\mathit{CNOT} \left(|0\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \right) = |0\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right)$$

The control qubit is in $|0\rangle$, so it doesn't have any effect on the target qubit.

Phase kickback

What happens when we apply a CNOT to this state instead?

$$|1\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right)$$

We get

$$\begin{aligned} \mathit{CNOT} \left(|1\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \right) &= |1\rangle \left(\frac{|1\rangle - |0\rangle}{\sqrt{2}} \right) \\ &= |1\rangle \left(- \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \right) \end{aligned}$$

It looks like we've changed the phase of the second qubit.

Phase kickback

But this is a *global* phase, and the math doesn't care which qubit it's attached to. We could equally well write

$$CNOT\left(|1\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)\right)=(-|1\rangle)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \tag{61}$$

Now it's as if the *target* qubit has done something to the *control* qubit!

We say that the phase has been "kicked back" from the second qubit to the first.

Phase kickback

We can write a general version of this effect:

$$CNOT\left(|b\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)\right) = (-1)^b|b\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \tag{62}$$

But what does this have to do with Deutsch's algorithm and figuring out if a function is constant or balanced?

Deutsch's algorithm

This is where our oracle comes in. Suppose we have a box that implements U_f which sends

$$U_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle \tag{63}$$

Setting $|y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ will allow us to 'extract' the value of $f(0) \oplus f(1)$ with just a single query.

Let's work through the math.

$$
\begin{aligned}
U_f|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) &= \frac{1}{\sqrt{2}}\left(U_f|x\rangle|0\rangle - U_f|x\rangle|1\rangle\right) \\
&= \frac{1}{\sqrt{2}}\left(|x\rangle|0\oplus f(x)\rangle - |x\rangle|1\oplus f(x)\rangle\right)
\end{aligned}
$$

If $f(x)=0$, we get

$$
U_f|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) = |x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \tag{64}
$$

If $f(x)=1$, we get

$$
U_f|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) = (-1)|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \tag{65}
$$

So just like the case of the CNOT where we wrote the general version

$$CNOT\left(|b\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)\right)=(-1)^{b}|b\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right), \tag{66}$$

we can write

$$U_f\left(|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)\right)=(-1)^{f(x)}|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right), \tag{67}$$

Essentially, before the CNOT was just playing the role of U_f for the specific function $f(0)=0, f(1)=1$.

Deutsch's algorithm

This doesn't look like much on its own - we want to get a *combination* of $f(0)$ and $f(1)$. How can we do this?

By setting $|x\rangle$ to be a superposition!

$$\begin{aligned} & U_f \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \\ = \quad & U_f \frac{|0\rangle}{\sqrt{2}} \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) + U_f \frac{|1\rangle}{\sqrt{2}} \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \\ = \quad & (-1)^{f(0)} \frac{|0\rangle}{\sqrt{2}} \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) + (-1)^{f(1)} \frac{|1\rangle}{\sqrt{2}} \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \end{aligned}$$

Let's pull out a phase factor of $(-1)^{f(0)}$, since global phase doesn't matter anyways.

$$
\begin{aligned}
&= \frac{|0\rangle}{\sqrt{2}}\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) + (-1)^{f(0)\oplus f(1)}\frac{|1\rangle}{\sqrt{2}}\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \\
&= \frac{|0\rangle + (-1)^{f(0)\oplus f(1)}|1\rangle}{\sqrt{2}}\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)
\end{aligned}
$$

Now let's look at how this state is different when f is a constant vs. a balanced function.

Deutsch's algorithm

If the function is constant, $f(0) \oplus f(1) = 0$ and the state is

$$U_f \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \quad (68)$$

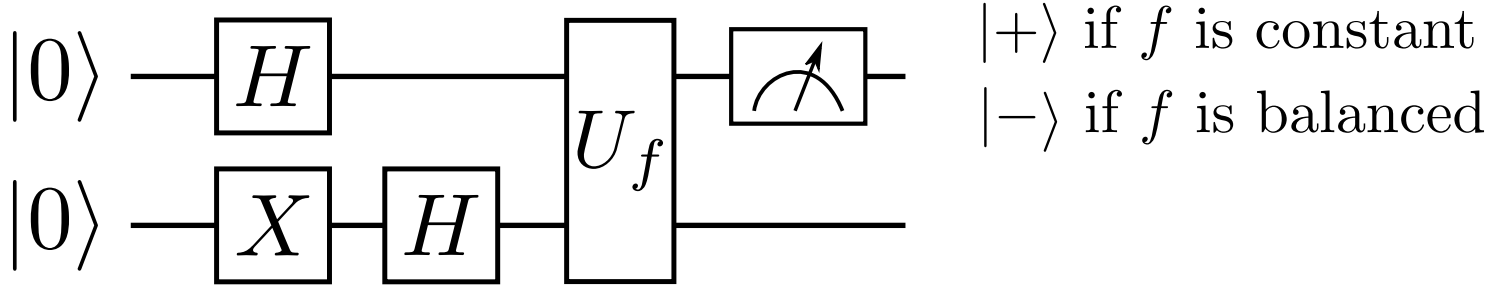
But if the function is balanced, $f(0) \oplus f(1) = 1$ and the state is

$$U_f \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) = \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \quad (69)$$

We can measure the first qubit in the *Hadamard basis* to determine exactly the value of $f(0) \oplus f(1)$!

Deutsch's algorithm

As a circuit, Deutsch's algorithm looks like this:

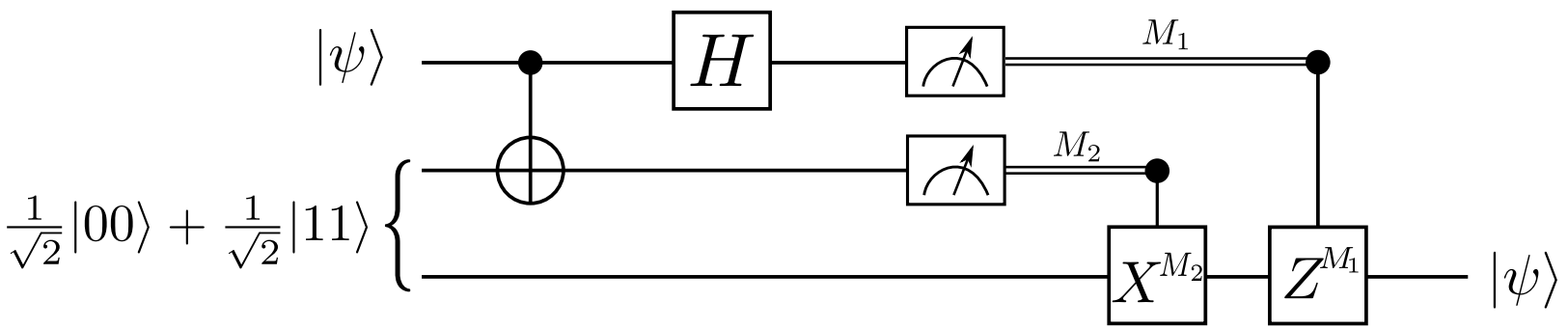


We called U_f just once, but were able to obtain information about the value of both $f(0)$ and $f(1)$!

Quantum teleportation

NOT related to Star Trek, despite the photos in media articles.

Alice has a qubit state she wants to send to Bob. She can do so if they share an entangled pair of qubits.



Wait a minute...

Why does she have to do this crazy thing? Why can't she just make a copy of her state and send it to him?

This is forbidden by the *no-cloning theorem*.

The no-cloning theorem

It is impossible to create a copying circuit that works for arbitrary quantum states.

We can prove this!

Proof of the no-cloning theorem

Suppose we want to clone a state $|\psi\rangle$. We want a unitary operation that sends

$$U(|\psi\rangle \otimes |s\rangle) \rightarrow |\psi\rangle \otimes |\psi\rangle \tag{70}$$

where $|s\rangle$ is some arbitrary state.

Let's suppose we find one. If our cloning machine is going to be universal, then we must also be able to clone some other state, $|\varphi\rangle$.

$$U(|\varphi\rangle \otimes |s\rangle) \rightarrow |\varphi\rangle \otimes |\varphi\rangle \tag{71}$$

Proof of the no-cloning theorem

We purportedly have:

$$U(|\psi\rangle \otimes |s\rangle) = |\psi\rangle \otimes |\psi\rangle \tag{72}$$

$$U(|\varphi\rangle \otimes |s\rangle) = |\varphi\rangle \otimes |\varphi\rangle \tag{73}$$

Take the inner product of the LHS of both equations:

$$(\langle\psi| \otimes \langle s|)U^\dagger U(|\varphi\rangle \otimes |s\rangle) = \langle\psi|\varphi\rangle\langle s|s\rangle = \langle\psi|\varphi\rangle \tag{74}$$

Proof of the no-cloning theorem

We purportedly have:

$$U(|\psi\rangle \otimes |s\rangle) = |\psi\rangle \otimes |\psi\rangle \tag{72}$$
$$U(|\varphi\rangle \otimes |s\rangle) = |\varphi\rangle \otimes |\varphi\rangle \tag{73}$$

Take the inner product of the LHS of both equations:

$$(\langle\psi| \otimes \langle s|)U^\dagger U(|\varphi\rangle \otimes |s\rangle) = \langle\psi|\varphi\rangle\langle s|s\rangle = \langle\psi|\varphi\rangle \tag{74}$$

Now take the inner product of the RHS of both equations:

$$(\langle\psi| \otimes \langle\psi|)(|\varphi\rangle \otimes |\varphi\rangle) = (\langle\psi|\varphi\rangle)^2 \tag{75}$$

Proof of the no-cloning theorem

We purportedly have:

$$U(|\psi\rangle \otimes |s\rangle) = |\psi\rangle \otimes |\psi\rangle \quad (72)$$

$$U(|\varphi\rangle \otimes |s\rangle) = |\varphi\rangle \otimes |\varphi\rangle \quad (73)$$

Take the inner product of the LHS of both equations:

$$(\langle\psi| \otimes \langle s|)U^\dagger U(|\varphi\rangle \otimes |s\rangle) = \langle\psi|\varphi\rangle\langle s|s\rangle = \langle\psi|\varphi\rangle \quad (74)$$

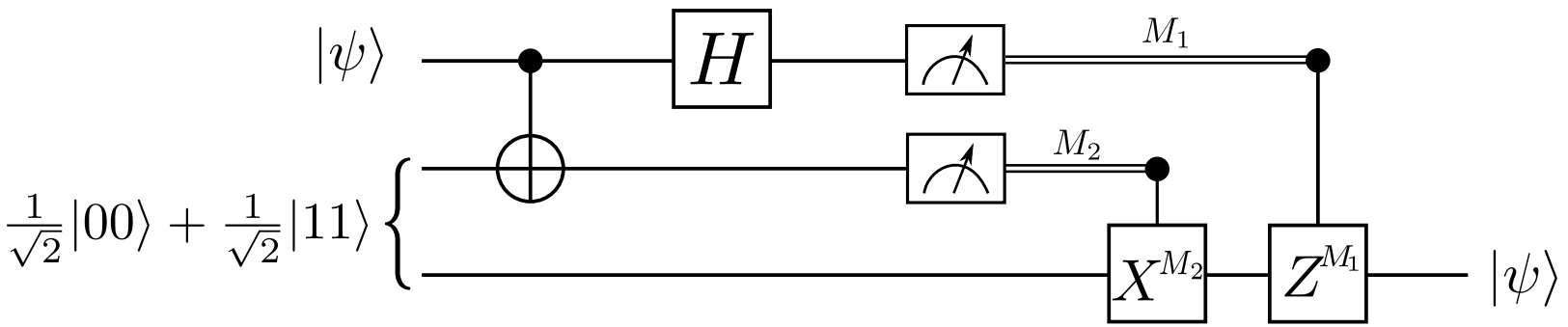
Now take the inner product of the RHS of both equations:

$$(\langle\psi| \otimes \langle\psi|)(|\varphi\rangle \otimes |\varphi\rangle) = (\langle\psi|\varphi\rangle)^2 \quad (75)$$

These two inner products must be equal; but the only numbers that square to themselves are 0 and 1! So either the two states are orthogonal, or are just the same state - they can't be arbitrary!

Back to quantum teleportation

So there is no general protocol for Alice to copy her qubit and send it to Bob; but teleportation allows her to transfer arbitrary *states* from her qubit to the qubit held by Bob.



How does this work?

Quantum teleportation

At the beginning, Alice has two qubits, one of which is part of a shared entangled state with Bob.

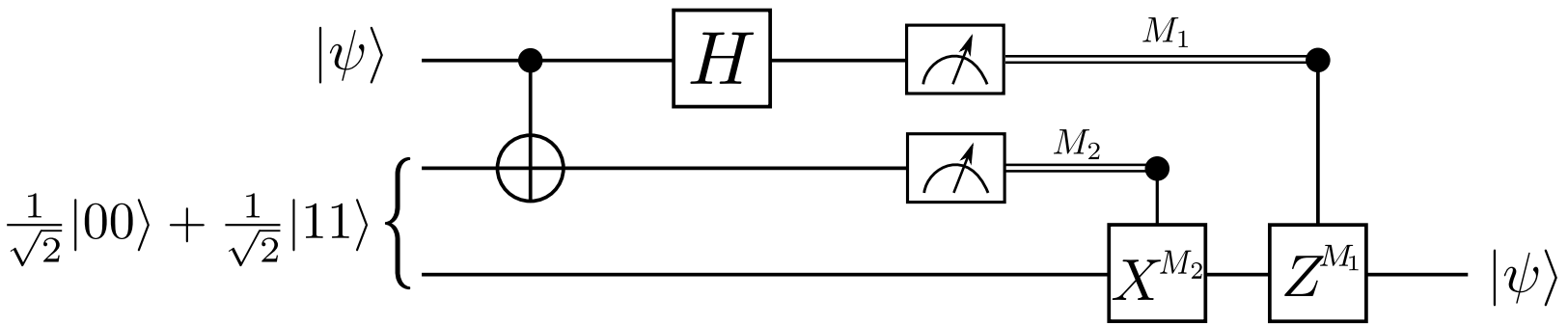
$$|\psi\rangle_A \otimes \frac{1}{\sqrt{2}} (|0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B) \tag{76}$$

Alice then entangles her first qubit with the one in the already-entangled state. She then applies a Hadamard to her first qubit, and measures both of her qubits.

Based on the results, Bob will perform a Pauli 'correction', and he will be left with Alice's original state.

I *really* encourage you to work through this one on your own.

Start by setting $|\psi\rangle_A = \alpha|0\rangle_A + \beta|1\rangle_A$, expand out the linear combination, work through the action of the gates, and then try and factor out Bob's qubit before performing the measurement.



There are some hints in the Jupyter notebook as well; I can show the solution next week.

Summary

Hopefully you have a basic understanding of:

- qubits and the common operations we apply on them
- entanglement and how it is used in quantum algorithms
- superdense coding, teleportation, and Deutsch's algorithm

Homework (can take up on Monday)

1. Work through the teleportation protocol.
2. Starting from two qubits in $|00\rangle$, design circuits that construct each of the four states in the Bell basis.
3. What is the result of $H \otimes H$ on $|00\rangle$.
   How about $H \otimes H \otimes H$ on $|000\rangle$?
   How does this generalize to $H^{\otimes n}$ acting on $|0\rangle^{\otimes n}$? Why might this state be useful?

Homework (can take up on Monday)

4. A CNOT is just one particular instance of a general controlled unitary gate. In a controlled-U gate, a unitary U is applied to the target qubit only if the control is 1 (for the CNOT, $U = X$). Working with single-qubit gates and CNOTs, design a circuit that implements a controlled-Z gate.
5. How do you implement a SWAP gate that swaps the state of two qubits? (Hint: work it out first for the computational basis states - the rest follows from linearity!)
6. A Toffoli gate is a controlled CNOT on 3 qubits (i.e. two controls and 1 target - X is applied only if *both* control qubits are 1). What is the unitary matrix for the Toffoli gate? Research: how can we implement the Toffoli using only single-qubit gates and CNOTs?