Week 5: Holveo's theorem and Introduction to Quantum Computation

COMS 4281 (Fall 2024)

Admin

1. Pset1 due Sunday, October 6, 11:59pm.
2. No practice worksheet/quiz this week.

Last time: the strangeness and power of entanglement

- Partial measurements, non-standard measurements on entangled states
- Heisenberg Uncertainty
- Quantum teleportation
- EPR Paradox and Bell's theorem

How much information do qubits store?

An n-qubit state $|\psi\rangle$ on the surface looks like it contains exponential amounts of information, because it is represented by a vector of dimension 2^n.

In quantum computing/quantum mechanics, we want to harness this to our advantage.

But as mentioned before, there's a tension between the exponentiality of quantum states, and the fact that this is hidden behind a **veil of measurement**.

Holevo's Theorem

If we have n qubits, how much classical information can we store? Can we use n qubits as a "quantum hard drive" to store many more than n classical bits?

No! Holevo's theorem states that, for the purposes of **information storage**, quantum bits are not much better than classical bits!

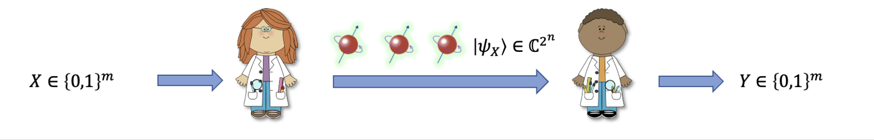
Holevo's Theorem

Alice has an m-bit string X that she wants to transmit to Bob. She wants to encode X into some n-qubit quantum state $|\psi_X\rangle$ such that Bob can perform some computation (unitaries + measurement) to try to decode X.

Bob only gets X with high probability if the number of qubits n is at least m.

Holevo's Theorem

Pictorially:



Superdense coding

Using preshared entanglement, Alice can save on the number of qubits she sends to convey X. This is achieved by a protocol known as **superdense coding**.

This allows Alice to convey m classical bits, while sending only $n = m/2$ qubits to Bob, **provided they use preshared entanglement**.

A simple equation describing superdense coding:

$$1\text{ebit} + 1\text{qbit} = 2\text{cbits}.$$

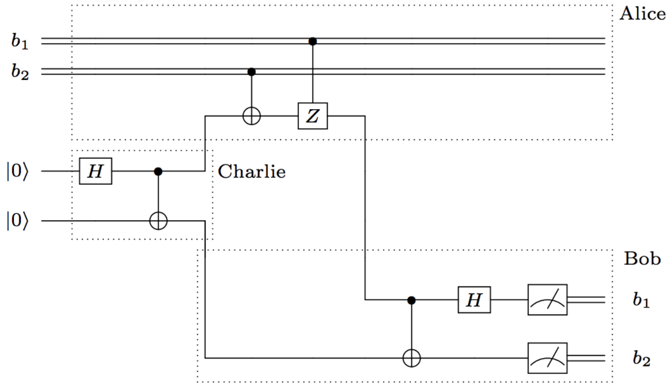
A simple equation describing superdense coding:

$$1\text{ebit} + 1\text{qbit} = 2\text{cbits}.$$

On the other hand, teleportation can be thought of as:

$$1\text{ebit} + 2\text{cbits} = 1\text{qbit}.$$

The protocol for superdense coding is very similar to teleportation. An EPR pair is shared (prepared by Charlie) is shared by Alice and Bob. Alice gets two bits (b_1, b_2), which determines which operations she applies to her qubit.

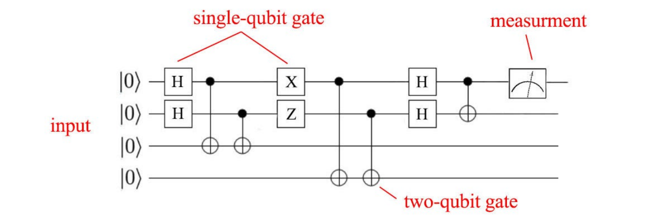


Quantum Computation

What is quantum computation?

It's whatever you can do with a quantum circuit, where (typically):

1. The input qubits start in the $|0\rangle$ state.
2. A sequence of single and two-qubit gates drawn from a **Gate Set**
3. Measurements (usually at the end)



Gate sets

What single- and two qubit gates are allowed in a quantum circuit?

Gate sets

What single- and two qubit gates are allowed in a quantum circuit?

It depends on the hardware!

Gate sets

What single- and two qubit gates are allowed in a quantum circuit?

It depends on the hardware!

A gate set $\mathcal{G}$ is called **universal** if, for any unitary U (which may act on many qubits), one can construct a circuit using gates from $\mathcal{G}$ to *approximate* U.

Notion of approximation

Definition: We say that unitary U ϵ-**approximates** V if for all quantum states $|\psi\rangle$,

$$\Big\| U|\psi\rangle - V|\psi\rangle \Big\| \leq \epsilon.$$

A circuit C without measurements corresponds to some unitary, so it is meaningful to say that a circuit C approximates a unitary matrix.

Universal gate sets

A **continuous** universal gate set: Rotations

$$\boxed{R_X(\theta)} = \begin{pmatrix} \cos\theta & -i\sin\theta \\ -i\sin\theta & \cos\theta \end{pmatrix} \qquad \boxed{R_Y(\theta)} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

$$\boxed{R_Z(\theta)} = \begin{pmatrix} e^{-i\theta} & 0 \\ 0 & e^{i\theta} \end{pmatrix}$$

over all $0 \leq \theta \leq 2\pi$ plus Phase shift

$$\boxed{P(\varphi)} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{pmatrix}$$

over all $0 \leq \varphi \leq 2\pi$ plus CNOT

Universal gate sets

A **discrete, finite** universal gate set:

$$\boxed{H} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \qquad \boxed{S} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \qquad \boxed{\text{CNOT}}$$

and

$$\boxed{T} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$$

Universal gate sets

Using a universal gate set $\mathcal{G}$ means, in principle we don't have to worry about what gates are used by a given quantum circuit C.

Universal gate sets

Using a universal gate set $\mathcal{G}$ means, in principle we don't have to worry about what gates are used by a given quantum circuit C.

That's because there is a way to **compile** the circuit C into another circuit C' that uses only gates from $\mathcal{G}$, and C *approximates* C'.

Universal gate sets

Using a universal gate set $\mathcal{G}$ means, in principle we don't have to worry about what gates are used by a given quantum circuit C.

That's because there is a way to **compile** the circuit C into another circuit C' that uses only gates from $\mathcal{G}$, and C *approximates* C'.

Ideally, we'd like the compiled circuit C' to be not much bigger than C.

Solovay-Kitaev Theorem

Let $\Gamma \subseteq \mathrm{SU}(2)$ (i.e. the set of single-qubit unitaries up to a global phase). Suppose

1. Γ generates a dense subgroup of $\mathrm{SU}(2)$.
2. Γ is closed under inverse.

Then for all $\epsilon > 0$ any unitary $U \in \mathrm{SU}(2)$ can be ϵ-approximated by a product of at most $O(\log(\frac{1}{\epsilon}))$ gates from Γ.

Universal gate sets

Corollary: Let $\Gamma \subseteq \mathrm{SU}(2)$ denote a set of single qubit unitaries satisfying conditions of Solovay-Kitaev theorem (an example is the set $\{H, T\}$). Then for all n, for all circuits C (allowed to use any single and two-qubit gates), for all $\epsilon > 0$

Universal gate sets

Corollary: Let $\Gamma \subseteq \mathrm{SU}(2)$ denote a set of single qubit unitaries satisfying conditions of Solovay-Kitaev theorem (an example is the set $\{H, T\}$). Then for all n, for all circuits C (allowed to use any single and two-qubit gates), for all $\epsilon > 0$

There exists a circuit C' consisting of gates from $\Gamma \cup \{CNOT\}$ **only** that

1. ϵ-approximates C
2. number of gates in C' is at most number of gates in C times $O(\log(1/\epsilon))$.

Quantum circuit synthesis

In quantum computing, we often want to come up with a quantum circuit C to (approximately) implement some unitary U. Hopefully, the circuit C is not too large! This is called **quantum circuit synthesis**.

Quantum circuit synthesis

In quantum computing, we often want to come up with a quantum circuit C to (approximately) implement some unitary U. Hopefully, the circuit C is not too large! This is called **quantum circuit synthesis**.

How many single- and two-qubit gates are needed to build a circuit C that approximates a given unitary U?

Quantum circuit synthesis

In quantum computing, we often want to come up with a quantum circuit C to (approximately) implement some unitary U. Hopefully, the circuit C is not too large! This is called **quantum circuit synthesis**.

How many single- and two-qubit gates are needed to build a circuit C that approximates a given unitary U?

In general, at least 4^n!

Our first quantum algorithm

Deutsch's problem

Given oracle access to a boolean function $f : \{0, 1\} \to \{0, 1\}$, decide whether $f(0) = f(1)$ or $f(0) \neq f(1)$.

Deutsch's problem

Given oracle access to a boolean function $f : \{0, 1\} \to \{0, 1\}$, decide whether $f(0) = f(1)$ or $f(0) \neq f(1)$.

Oracle access to a function f means that the computer can only access it as a **black-box**, i.e., query some inputs, and get the corresponding outputs. The computer cannot access how the black box is implemented.

Deutsch's problem

Given oracle access to a boolean function $f : \{0, 1\} \rightarrow \{0, 1\}$, decide whether $f(0) = f(1)$ or $f(0) \neq f(1)$.

Oracle access to a function f means that the computer can only access it as a **black-box**, i.e., query some inputs, and get the corresponding outputs. The computer cannot access how the black box is implemented.

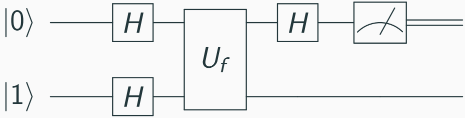
Claim: Any classical algorithm that solves the Deutsch problem must make 2 queries to f.

Quantum oracles

Quantum algorithms can access a black-box function f through a unitary U_f corresponding to the **reversible** version of f. For $f : \{0,1\} \to \{0,1\}$, this is a two-qubit unitary

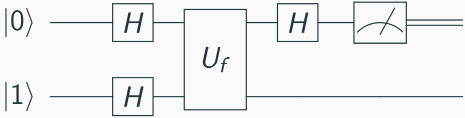
$$U_f \left| x, b \right\rangle = \left| x, b \oplus f(x) \right\rangle .$$

A quantum circuit that wants to access f will simply call U_f just like any other two-qubit gate.



This quantum algorithm solves the Deutsch problem with **one** call to U_f.

Deutsch's algorithm



This quantum algorithm solves the Deutsch problem with **one** call to U_f.

Let's do this on the board!

Deutsch's algorithm

The algorithm evaluates the function f in **superposition**. This seems to give a 2x speedup!

Is this cheating? Maybe the "quantum access" is just really making multiple classical queries under the hood?

Observation: the qubit storing the answer at the end corresponds to the *input wire* of the oracle U_f. We don't care about the output wire!

This is a common feature in many quantum algorithms with exponential speedup.

Simons Problem

Simons Problem

Problem: Given oracle access to $f : \{0,1\}^n \rightarrow \{0,1\}^n$ such that there exists a nonzero **secret string** $s \in \{0,1\}^n$ where for all $x, y \in \{0,1\}^n$

$$f(x) = f(y) \Leftrightarrow x \oplus y = s$$

find the secret string s.

Simons Problem

Example function f:

x	$f(x)$
000	101
001	010
010	000
011	110
100	000
101	110
110	101
111	010

What's the secret s?

Simons Problem

Problem: Given oracle access to $f : \{0,1\}^n \to \{0,1\}^n$ such that there exists a nonzero **secret string** $s \in \{0,1\}^n$ where for all $x, y \in \{0,1\}^n$

$$f(x) = f(y) \Leftrightarrow x \oplus y = s$$

find the secret string s.

Question: How many queries to f are needed to find the secret?

Classical algorithm to solve Simons Problem

1. Randomly sample $x_1, \ldots, x_K \in \{0,1\}^n$ for $K = 10\sqrt{2^n}$.
2. Check if there exists a pair $x_i \neq x_j$ where $f(x_i) = f(x_j)$. If so, then output $s = x_i \oplus x_j$.

Classical algorithm to solve Simons Problem

1. Randomly sample $x_1, \ldots, x_K \in \{0,1\}^n$ for $K = 10\sqrt{2^n}$.
2. Check if there exists a pair $x_i \neq x_j$ where $f(x_i) = f(x_j)$. If so, then output $s = x_i \oplus x_j$.

By the **birthday paradox**, this algorithm will find the secret with high probability. Requires $O(2^{n/2})$ queries to f.

(Show on board)

Classical algorithm to solve Simons Problem

1. Randomly sample $x_1, \ldots, x_K \in \{0,1\}^n$ for $K = 10\sqrt{2^n}$.
2. Check if there exists a pair $x_i \neq x_j$ where $f(x_i) = f(x_j)$. If so, then output $s = x_i \oplus x_j$.

By the **birthday paradox**, this algorithm will find the secret with high probability. Requires $O(2^{n/2})$ queries to f.

(Show on board)

$2^{n/2}$ queries are necessary for any classical algorithm!

Simons Algorithm

A quantum algorithm queries f by calling the $2n$-qubit unitary U_f that maps

$$|\underbrace{x}_{n \text{ qubits}}, \underbrace{z}_{n \text{ qubits}}\rangle \mapsto |x, z \oplus f(x)\rangle$$

Here, $\oplus$ denotes bitwise addition.

Simons Algorithm

Simons algorithm is a **classical-quantum** hybrid algorithm.

It uses the quantum computer as a *subroutine* to sample from a distribution many times, and uses **classical post-processing** to extract the secret.

Simons Algorithm

Simons algorithm is a **classical-quantum** hybrid algorithm.

It uses the quantum computer as a *subroutine* to sample from a distribution many times, and uses **classical post-processing** to extract the secret.

Simons subroutine: Quantum circuit queries U_f *once* and obtains a *uniformly random* string $y \in \{0,1\}^n$ where inner product of y and the secret s,

$$s \cdot y = s_1 y_1 \oplus s_2 y_2 \oplus \cdots \oplus s_n y_n$$

is equal to 0.

Simons algorithm, classical post-processing

Classical post-processing: Obtain $m = 100n$ samples $y^{(1)}, y^{(2)}, \dots, y^{(m)}$ such that

$$\begin{aligned} y^{(1)} \cdot s &= 0 \\ y^{(2)} \cdot s &= 0 \\ &\vdots \\ y^{(m)} \cdot s &= 0 \end{aligned}$$

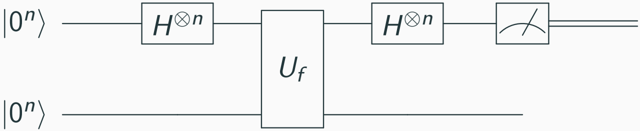
Simons algorithm, classical post-processing

Classical post-processing: Obtain $m = 100n$ samples $y^{(1)}, y^{(2)}, \ldots, y^{(m)}$ such that

$$\begin{aligned} y^{(1)} \cdot s &= 0 \\ y^{(2)} \cdot s &= 0 \\ &\vdots \\ y^{(m)} \cdot s &= 0 \end{aligned}$$

With high probability, can solve this system of linear equations using Gaussian elimination to get s.

Simons subroutine



$H^{\otimes n}$ means applying H to n qubits independently.

Hadamard math

We know that $H|0\rangle = |+\rangle$. What is $H^{\otimes n}|0\rangle^{\otimes n}$?

Hadamard math

We know that $H|0\rangle = |+\rangle$. What is $H^{\otimes n}|0\rangle^{\otimes n}$?

$$H^{\otimes n}|0\rangle^{\otimes n} = (H|0\rangle)^{\otimes n} = |+\rangle^{\otimes n} \ .$$

Hadamard math

We know that $H|0\rangle = |+\rangle$. What is $H^{\otimes n}|0\rangle^{\otimes n}$?

$$H^{\otimes n}|0\rangle^{\otimes n} = (H|0\rangle)^{\otimes n} = |+\rangle^{\otimes n} \ .$$

This in turn is

$$|+\rangle^{\otimes n} = \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x_1, \ldots, x_n\rangle$$

Hadamard math

Fix $x_1, \ldots, x_n \in \{0,1\}$. What is $H^{\otimes n} |x_1, \ldots, x_n\rangle$?

Hadamard math

Fix $x_1, \ldots, x_n \in \{0, 1\}$. What is $H^{\otimes n} |x_1, \ldots, x_n\rangle$?

This is

$$
\begin{aligned}
&(H|x_1\rangle) \otimes (H|x_2\rangle) \otimes \cdots \otimes (H|x_n\rangle) \\
&= \frac{1}{\sqrt{2^n}} \Big(|0\rangle + (-1)^{x_1} |1\rangle \Big) \otimes \Big(|0\rangle + (-1)^{x_2} |1\rangle \Big) \otimes \cdots \otimes \Big(|0\rangle + (-1)^{x_n} |1\rangle \Big)
\end{aligned}
$$

Hadamard math

Fix $x_1, \ldots, x_n \in \{0, 1\}$. What is $H^{\otimes n} |x_1, \ldots, x_n\rangle$?

This is

$$(H|x_1\rangle) \otimes (H|x_2\rangle) \otimes \cdots \otimes (H|x_n\rangle)$$
$$= \frac{1}{\sqrt{2^n}} \Big(|0\rangle + (-1)^{x_1} |1\rangle \Big) \otimes \Big(|0\rangle + (-1)^{x_2} |1\rangle \Big) \otimes \cdots \otimes \Big(|0\rangle + (-1)^{x_n} |1\rangle \Big)$$
$$= \frac{1}{\sqrt{2^n}} \sum_{y_1, y_2, \ldots, y_n \in \{0,1\}} (-1)^{x_1 y_1} |y_1\rangle \cdots (-1)^{x_n y_n} |y_n\rangle$$

Hadamard math

Fix $x_1, \ldots, x_n \in \{0,1\}$. What is $H^{\otimes n} |x_1, \ldots, x_n\rangle$?

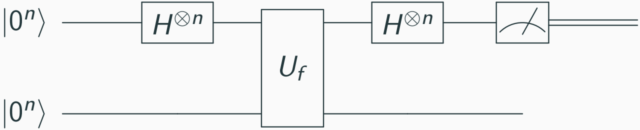
This is

$$
\begin{aligned}
&(H|x_1\rangle) \otimes (H|x_2\rangle) \otimes \cdots \otimes (H|x_n\rangle) \\
&= \frac{1}{\sqrt{2^n}} \Big(|0\rangle + (-1)^{x_1} |1\rangle \Big) \otimes \Big(|0\rangle + (-1)^{x_2} |1\rangle \Big) \otimes \cdots \otimes \Big(|0\rangle + (-1)^{x_n} |1\rangle \Big) \\
&= \frac{1}{\sqrt{2^n}} \sum_{y_1, y_2, \ldots, y_n \in \{0,1\}} (-1)^{x_1 y_1} |y_1\rangle \cdots (-1)^{x_n y_n} |y_n\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle
\end{aligned}
$$

where $x \cdot y$ denotes the inner product of the strings x and y modulo 2:

$$x \cdot y = x_1 y_1 + \cdots + x_n y_n \mod 2 .$$

Simons subroutine



Let's analyze this on the board.

Simons algorithm

- Makes $O(n)$ queries to U_f and solves the problem with high probability
- Once again, the valuable information is stored not in the answer register of U_f, but in the input register.

Simons algorithm

- Making crucial use of constructive/destructive interference!
- It's finding **global hidden structure** in the function.
- Is this speedup more convincing?

Simons algorithm

- Invented by Dan Simons in 1992, and was the first example of a problem that could be solved exponentially faster with a quantum algorithm compared to a classical randomized algorithm.
- This algorithm directly inspired Peter Shor to invent the famous factoring algorithm.
- Recently, Simons algorithm also has applications to breaking symmetric key cryptography.

Next time

Quantum Fourier Transform.