Quantum Algorithms

Seminar Template

Mingyu Lee
Seoul National University

red1108@snu.ac.kr
+82 010-2857-3320 (only text) / +01 530-760-8690

2026.04.03(FRI) 16:30 KST

Welcome

Welcome everyone! I'm so excited to have you in the course, and I hope these materials help you build both intuition and technical depth.

All lecture materials will be available on my personal homepage and on GitHub:

`https://github.com/red1108/lectures`

(You can also track my last-minute slide prep through the commit history.)

About Me

Hello, I am Mingyu Lee.

I am a fourth-year undergraduate from Seoul National University, currently an exchange student at UC Davis.

Personal homepage:
`https://red1108.github.io/`

LinkedIn:
`https://linkedin.com/in/red1108/`

Research Interests

I am studying quantum computation, quant trading, algorithms, and machine learning.

About This Course

Overview

- Major ideas in quantum algorithms
- Basic building blocks and frameworks
- Selected techniques with quantum complexity context
- Where quantum speedups may arise

Goals

- Build intuition for algorithms and complexity
- Learn several core quantum techniques
- Identify tasks that may benefit from quantum methods

About This Course

Plan

- 5–10 participants (Actually, 14)
- Basic background, around Grover search.
- 8 weeks
- Using latex slides, If I have time to prepare.
- Interactive!

Notes Schedule can be adjusted to fit the group

It already has: Fri 15:00, then Fri 16:00, and now Fri 16:30.
A very efficient optimization process.

Lecture Schedule

Date	Study content	Person in charge	Additional notes
4/3 Fri	**Quantum Complexity**	**Mingyu Lee**	**About oracle [Gri24]**
4/10 Fri	Quantum Amplitude Amplification	Mingyu Lee	Grover search quantum amplitude estimation
4/17 Fri	Quantum Min/Max Searching	Mingyu Lee	
4/24 Fri	Quantum Analog-Digital Conversion	Mingyu Lee	Swap test, k-NN
5/1 Fri	Quantum Walk	Mingyu Lee	
5/15 Fri	HHL	Mingyu Lee	recorded Zoom lecture
5/22 Fri	QSVT	Myeongjin Shin*	
5/29 Fri	Quadratic Speedup	Mingyu Lee	planted k-XOR community detection

* Special speaker

Survey time!

Outline

Outline

Back to the Grover Search

- Grover is one of the cleanest examples of a quadratic quantum speedup
- The task is simple: find a marked item in an unstructured space
- Classical: $O(N)$, Quantum: $O(\sqrt{N})$ (It's optimal!)
- Grover's algorithms assume oracle access to a Boolean function $f : \{0,1\}^n \to \{0,1\}$ that identifies the marked item

Back to the Grover Search

- Grover is one of the cleanest examples of a quadratic quantum speedup
- The task is simple: find a marked item in an unstructured space
- Classical: $O(N)$, Quantum: $O(\sqrt{N})$ (It's optimal!)
- Grover's algorithms assume oracle access to a Boolean function $f : \{0,1\}^n \to \{0,1\}$ that identifies the marked item

Back to the Grover Search

- Grover is one of the cleanest examples of a quadratic quantum speedup
- The task is simple: find a marked item in an unstructured space
- Classical: $O(N)$, Quantum: $O(\sqrt{N})$ (It's optimal!)
- Grover's algorithms assume oracle access to a Boolean function $f : \{0,1\}^n \to \{0,1\}$ that identifies the marked item

Back to the Grover Search

- Grover is one of the cleanest examples of a quadratic quantum speedup
- The task is simple: find a marked item in an unstructured space
- Classical: $O(N)$, Quantum: $O(\sqrt{N})$ (It's optimal!)
- Grover's algorithms assume oracle access to a Boolean function $f : \{0,1\}^n \to \{0,1\}$ that identifies the marked item

Back to the Grover Search

- Grover is one of the cleanest examples of a quadratic quantum speedup
- The task is simple: find a marked item in an unstructured space
- Classical: $O(N)$, Quantum: $O(\sqrt{N})$ (It's optimal!)
- Grover's algorithms assume oracle access to a Boolean function $f : \{0,1\}^n \to \{0,1\}$ that identifies the marked item

But does this Oracle access just magically build itself?
Did our ancestors hand it down to us?

Why Does the Oracle Feel Suspicious?

- We want to find an input x such that $f(x) = 1$
- In the quantum setting, this is phrased as oracle access to f
- That can sound like we are handing the quantum algorithm a mysterious black box
- Are we giving quantum extra power just by how we defined the model?

Why Does the Oracle Feel Suspicious?

- We want to find an input x such that $f(x) = 1$
- In the quantum setting, this is phrased as oracle access to f
- That can sound like we are handing the quantum algorithm a mysterious black box
- Are we giving quantum extra power just by how we defined the model?

Why Does the Oracle Feel Suspicious?

- We want to find an input x such that $f(x) = 1$
- In the quantum setting, this is phrased as oracle access to f
- That can sound like we are handing the quantum algorithm a mysterious black box
- Are we giving quantum extra power just by how we defined the model?

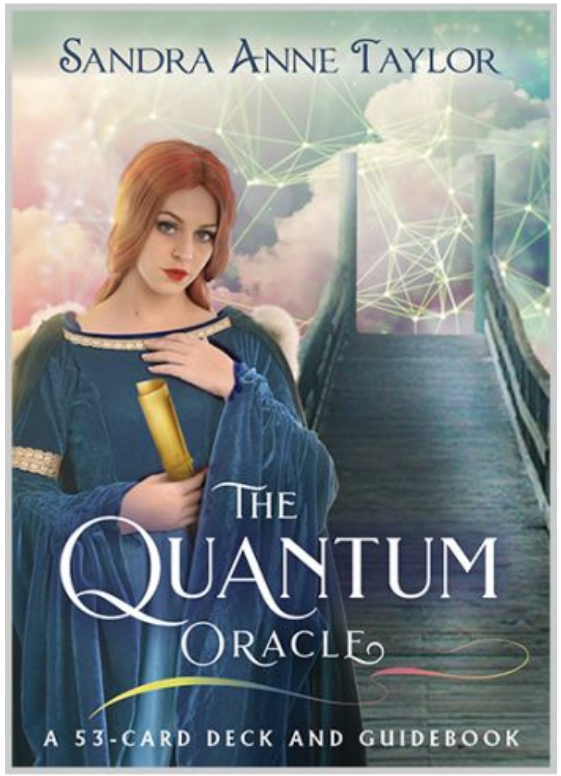


Image source: https://www.hayhouse.com/the-quantum-oracle-card-deck?srsltid=AfmBOoq5NoSrzFKPZUm3H9vLmz5C8wuV4hZ2

Why Does the Oracle Feel Suspicious?

- We want to find an input x such that $f(x) = 1$
- In the quantum setting, this is phrased as oracle access to f
- That can sound like we are handing the quantum algorithm a mysterious black box
- Are we giving quantum extra power just by how we defined the model?

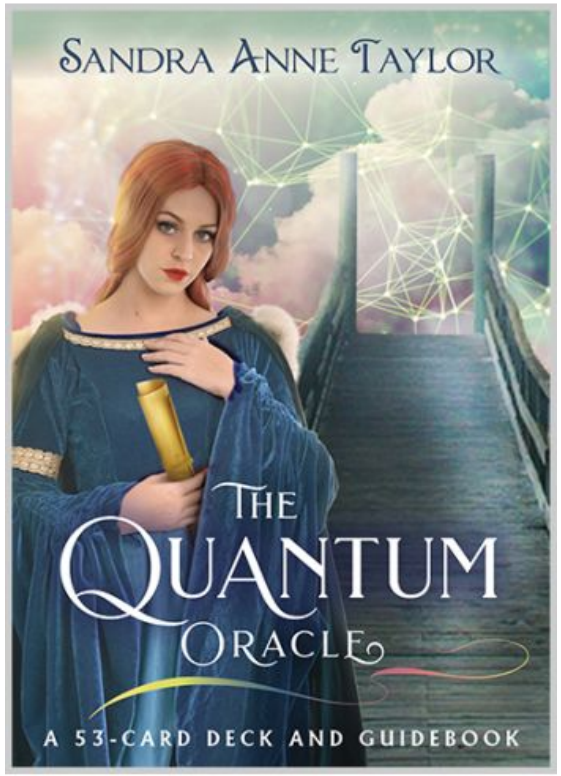
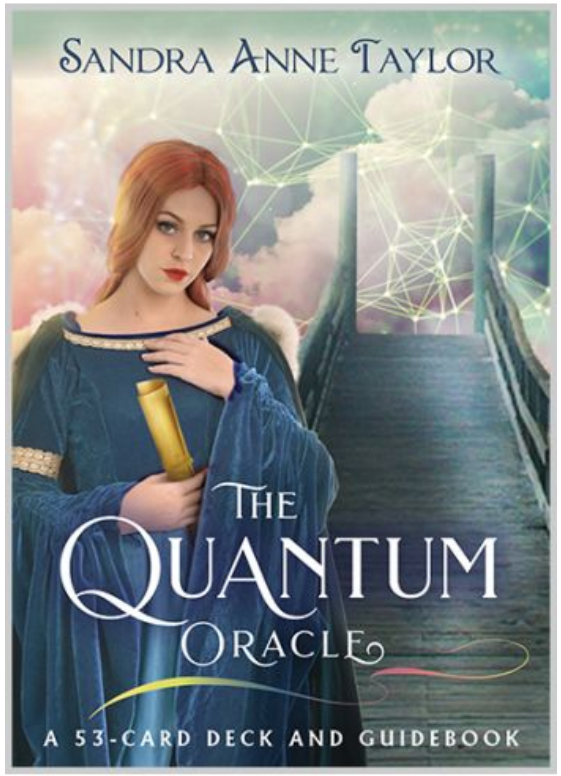


Image source: `https://www.hayhouse.com/the-quantum-oracle-card-deck?srsltid=AfmBOoq5NoSrzFKPZUm3H9vLmz5C8wuV4hZ2`

Why Does the Oracle Feel Suspicious?

- We want to find an input x such that $f(x) = 1$
- In the quantum setting, this is phrased as oracle access to f
- That can sound like we are handing the quantum algorithm a mysterious black box
- Are we giving quantum extra power just by how we defined the model?



But why do we rarely ask the same question on the classical side?

Image source: https://www.hayhouse.com/the-quantum-oracle-card-deck?srsltid=AfmBOoq5NoSrzFKPZUm3H9vLmz5C8wuV4hZ2

What the Classical Algorithm Already Assumes

- A classical $O(N)$ search just checks candidates one by one
- But to check a candidate x, we still need to evaluate $f(x)$
- In query complexity, that evaluation cost is packaged into a query and then abstracted away
- Classical search also assumes access to f; it just sounds less dramatic

What the Classical Algorithm Already Assumes

- A classical $O(N)$ search just checks candidates one by one
- But to check a candidate x, we still need to evaluate $f(x)$
- In query complexity, that evaluation cost is packaged into a query and then abstracted away
- Classical search also assumes access to f; it just sounds less dramatic

What the Classical Algorithm Already Assumes

- A classical $O(N)$ search just checks candidates one by one
- But to check a candidate x, we still need to evaluate $f(x)$
- In query complexity, that evaluation cost is packaged into a query and then abstracted away
- Classical search also assumes access to f; it just sounds less dramatic

What the Classical Algorithm Already Assumes

- A classical $O(N)$ search just checks candidates one by one
- But to check a candidate x, we still need to evaluate $f(x)$
- In query complexity, that evaluation cost is packaged into a query and then abstracted away
- Classical search also assumes access to f; it just sounds less dramatic

What the Classical Algorithm Already Assumes

- A classical $O(N)$ search just checks candidates one by one
- But to check a candidate x, we still need to evaluate $f(x)$
- In query complexity, that evaluation cost is packaged into a query and then abstracted away
- Classical search also assumes access to f; it just sounds less dramatic

The quantum oracle is the same hidden assumption, written in quantum language.

Why the Oracle Model Is Fair

- Suppose f comes from a practical problem and has a classical circuit description
- Then a quantum circuit can implement the standard oracle $B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$
- Intuition: implement the classical circuit reversibly and run it in superposition
- If there is no classical circuit for f, then the classical black-box model also stops making sense

Why the Oracle Model Is Fair

- Suppose f comes from a practical problem and has a classical circuit description
- Then a quantum circuit can implement the standard oracle $B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$
- Intuition: implement the classical circuit reversibly and run it in superposition
- If there is no classical circuit for f, then the classical black-box model also stops making sense

Why the Oracle Model Is Fair

- Suppose f comes from a practical problem and has a classical circuit description
- Then a quantum circuit can implement the standard oracle $B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$
- Intuition: implement the classical circuit reversibly and run it in superposition
- If there is no classical circuit for f, then the classical black-box model also stops making sense

Why the Oracle Model Is Fair

- Suppose f comes from a practical problem and has a classical circuit description
- Then a quantum circuit can implement the standard oracle $B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$
- Intuition: implement the classical circuit reversibly and run it in superposition
- If there is no classical circuit for f, then the classical black-box model also stops making sense

Why the Oracle Model Is Fair

- Suppose f comes from a practical problem and has a classical circuit description
- Then a quantum circuit can implement the standard oracle $B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$
- Intuition: implement the classical circuit reversibly and run it in superposition
- If there is no classical circuit for f, then the classical black-box model also stops making sense

So the oracle model is not giving quantum a special advantage by definition.

Why the Oracle Model Is Fair

- Suppose f comes from a practical problem and has a classical circuit description
- Then a quantum circuit can implement the standard oracle $B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$
- Intuition: implement the classical circuit reversibly and run it in superposition
- If there is no classical circuit for f, then the classical black-box model also stops making sense

So the oracle model is not giving quantum a special advantage by definition.

But, in some senses, your concern is valid....

Outline

Only Query Complexity Matters

Claim 1

Every classical computation can be simulated by a quantum computation.

- Classical algorithms are built from logical gates such as AND, OR, and NOT
- These can be embedded into reversible circuits
- So evaluating f is not a special burden placed only on the quantum side

Only Query Complexity Matters

Claim 1

Every classical computation can be simulated by a quantum computation.

- Classical algorithms are built from logical gates such as AND, OR, and NOT
- These can be embedded into reversible circuits
- So evaluating f is not a special burden placed only on the quantum side

Only Query Complexity Matters

Claim 1

Every classical computation can be simulated by a quantum computation.

- Classical algorithms are built from logical gates such as AND, OR, and NOT
- These can be embedded into reversible circuits
- So evaluating f is not a special burden placed only on the quantum side

Only Query Complexity Matters

Claim 1

Every classical computation can be simulated by a quantum computation.

- Classical algorithms are built from logical gates such as AND, OR, and NOT
- These can be embedded into reversible circuits
- So evaluating f is not a special burden placed only on the quantum side

The evaluation cost is abstracted away in both models;
That's why we only consider the query complexity.

Why the Oracle Model Is Fair

In classical computers, every computation is built from logical gates such as AND, OR, and NOT. These can be embedded into reversible circuits.

$$|a\rangle|b\rangle|0\rangle \xrightarrow{\text{Toffoli}} |a\rangle|b\rangle|a \wedge b\rangle$$

$$|a\rangle|b\rangle|0\rangle \xrightarrow{\text{Three X gates}} |\neg a\rangle|\neg b\rangle|1\rangle \xrightarrow{\text{Toffoli}} |\neg a\rangle|\neg b\rangle|a \vee b\rangle \xrightarrow{\text{Two X gates to uncompute}} |a\rangle|b\rangle|a \vee b\rangle$$

# of two-input gates is same. And, single qubit gates are (almost) free.

Why the Oracle Model Is Fair

In classical computers, every computation is built from logical gates such as AND, OR, and NOT. These can be embedded into reversible circuits.

$$|a\rangle|b\rangle|0\rangle \xrightarrow{\text{Toffoli}} |a\rangle|b\rangle|a \wedge b\rangle$$

$$|a\rangle|b\rangle|0\rangle \xrightarrow{\text{Three X gates}} |\neg a\rangle|\neg b\rangle|1\rangle \xrightarrow{\text{Toffoli}} |\neg a\rangle|\neg b\rangle|a \vee b\rangle \xrightarrow{\text{Two X gates to uncompute}} |a\rangle|b\rangle|a \vee b\rangle$$

# of two-input gates is same. And, single qubit gates are (almost) free.

Everyone happy?

Why the Oracle Model Is Fair

In classical computers, every computation is built from logical gates such as AND, OR, and NOT. These can be embedded into reversible circuits.

$$|a\rangle|b\rangle|0\rangle \xrightarrow{\text{Toffoli}} |a\rangle|b\rangle|a \wedge b\rangle$$

$$|a\rangle|b\rangle|0\rangle \xrightarrow{\text{Three X gates}} |\neg a\rangle|\neg b\rangle|1\rangle \xrightarrow{\text{Toffoli}} |\neg a\rangle|\neg b\rangle|a \vee b\rangle \xrightarrow{\text{Two X gates to uncompute}} |a\rangle|b\rangle|a \vee b\rangle$$

# of two-input gates is same. And, single qubit gates are (almost) free.

Everyone happy?

Hmm....

Is the Oracle Assumption Fair...?

In classical computers, we can do in-place computation, so we can evaluate f without extra space.

$$[a, b] \xrightarrow{\text{AND}} [a, a \wedge b]$$

$$[a, b] \xrightarrow{\text{OR}} [a, a \vee b]$$

But in quantum computers, we need to use an extra register to store the output of f to keep it reversible.

We need **extra space** to achieve same time complexity.

What Is Query Complexity?

- We want to learn some property of a function f
- But we only access f by querying one input at a time
- The main cost is not total gate count
- The main cost is: **how many times do we query f?**

What Is Query Complexity?

- We want to learn some property of a function f
- But we only access f by querying one input at a time
- The main cost is not total gate count
- The main cost is: **how many times do we query f?**

What Is Query Complexity?

- We want to learn some property of a function f
- But we only access f by querying one input at a time
- The main cost is not total gate count
- The main cost is: **how many times do we query f?**

What Is Query Complexity?

- We want to learn some property of a function f
- But we only access f by querying one input at a time
- The main cost is not total gate count
- The main cost is: **how many times do we query f?**

What Is Query Complexity?

- We want to learn some property of a function f
- But we only access f by querying one input at a time
- The main cost is not total gate count
- The main cost is: **how many times do we query f?**

$$\text{query complexity} = \#\text{ accesses to } f$$

Why This Model Is Useful

- The black-box model severely restricts what an algorithm can do
- That makes clean lower and upper bounds possible
- If quantum uses significantly fewer queries than classical,
- then we have concrete evidence of a quantum advantage

Why This Model Is Useful

- The black-box model severely restricts what an algorithm can do
- That makes clean lower and upper bounds possible
- If quantum uses significantly fewer queries than classical,
- then we have concrete evidence of a quantum advantage

Why This Model Is Useful

- The black-box model severely restricts what an algorithm can do
- That makes clean lower and upper bounds possible
- If quantum uses significantly fewer queries than classical,
- then we have concrete evidence of a quantum advantage

Why This Model Is Useful

- The black-box model severely restricts what an algorithm can do
- That makes clean lower and upper bounds possible
- If quantum uses significantly fewer queries than classical,
- then we have concrete evidence of a quantum advantage

Classical Query Model

- Classically, a query is straightforward
- We input $x \in \{0,1\}^n$ and receive $f(x)$
- For search, the classical algorithm simply keeps asking:

Classical Query Model

- Classically, a query is straightforward
- We input $x \in \{0,1\}^n$ and receive $f(x)$
- For search, the classical algorithm simply keeps asking:

Classical Query Model

- Classically, a query is straightforward
- We input $x \in \{0,1\}^n$ and receive $f(x)$
- For search, the classical algorithm simply keeps asking:

Classical Query Model

- Classically, a query is straightforward
- We input $x \in \{0,1\}^n$ and receive $f(x)$
- For search, the classical algorithm simply keeps asking:

$$f(x_1),\ f(x_2),\ f(x_3),\ \ldots$$

Quantum query algorithms should be compared against this same access model.

The Standard Quantum Oracle

- For a function $f : \{0,1\}^n \to \{0,1\}^m$, the standard oracle is

$$B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$$

- It computes the function value reversibly
- Input register stays intact
- Output register stores the answer

The Standard Quantum Oracle

- For a function $f : \{0,1\}^n \to \{0,1\}^m$, the standard oracle is

$$B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$$

- It computes the function value reversibly
- Input register stays intact
- Output register stores the answer

The Standard Quantum Oracle

- For a function $f : \{0,1\}^n \to \{0,1\}^m$, the standard oracle is

$$B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$$

- It computes the function value reversibly
- Input register stays intact
- Output register stores the answer

The Standard Quantum Oracle

- For a function $f : \{0,1\}^n \to \{0,1\}^m$, the standard oracle is

$$B_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$$

- It computes the function value reversibly
- Input register stays intact
- Output register stores the answer

The Oracle

- For Boolean functions $f : \{0,1\}^n \to \{0,1\}$, we also use

$$O_f|x\rangle = (-1)^{f(x)}|x\rangle$$

- The answer is written into the phase instead of an output register
- This looks different, but it is only marginally different from B_f
- In many algorithms, especially Fourier-style ones, the phase form is more natural

The Oracle

- For Boolean functions $f : \{0,1\}^n \to \{0,1\}$, we also use

$$O_f|x\rangle = (-1)^{f(x)}|x\rangle$$

- The answer is written into the phase instead of an output register
- This looks different, but it is only marginally different from B_f
- In many algorithms, especially Fourier-style ones, the phase form is more natural

The Oracle

- For Boolean functions $f : \{0,1\}^n \to \{0,1\}$, we also use

$$O_f|x\rangle = (-1)^{f(x)}|x\rangle$$

- The answer is written into the phase instead of an output register
- This looks different, but it is only marginally different from B_f
- In many algorithms, especially Fourier-style ones, the phase form is more natural

The Oracle

- For Boolean functions $f : \{0,1\}^n \to \{0,1\}$, we also use

$$O_f|x\rangle = (-1)^{f(x)}|x\rangle$$

- The answer is written into the phase instead of an output register
- This looks different, but it is only marginally different from B_f
- In many algorithms, especially Fourier-style ones, the phase form is more natural

Comparing Two Oracle Models

	Standard oracle B_f	**Phase oracle O_f**
Action	$\|x\rangle\|b\rangle \mapsto \|x\rangle\|b \oplus f(x)\rangle$	$\|x\rangle \mapsto (-1)^{f(x)}\|x\rangle$
Represent	output register	phase
Intuition	closer to classical computation	cleaner for interference and Fourier analysis
Main picture	reversibly computes $f(x)$	marks inputs by a sign change

In fact, B_f and O_f are **equivalent!**
They can simulate each other with only constant overhead.

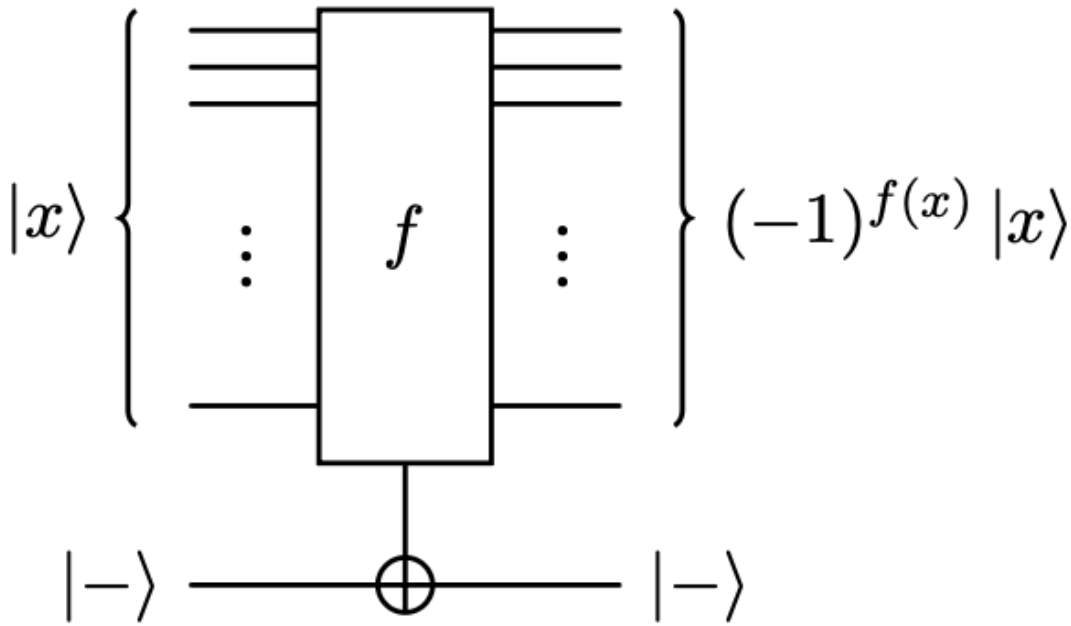
Two Oracles Are Equivalent

Theorem 2 (Equivalence of B_f and O_f)

For any Boolean function $f : \{0,1\}^n \to \{0,1\}$, the standard oracle B_f and the phase oracle O_f can simulate each other with only constant overhead.

Proof: easy and straightforward.

Two Oracles Are Equivalent: B_f to O_f



$$|x\rangle|-\rangle \xrightarrow{B_f} |x\rangle \frac{|f(x)\rangle - |1 \oplus f(x)\rangle}{\sqrt{2}} = (-1)^{f(x)}|x\rangle|-\rangle = \big(O_f|x\rangle\big)|-\rangle.$$

But we need one extra qubit initialized to $|-\rangle$ (constant overhead).

Two Oracles Are Equivalent: O_f to B_f

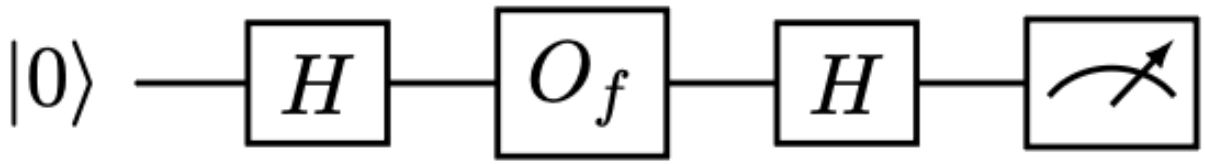
$$|x\rangle|b\rangle \xrightarrow{H \text{ on } b} |x\rangle \frac{|0\rangle + (-1)^b|1\rangle}{\sqrt{2}}$$

$$\xrightarrow{\text{controlled-}O_f} |x\rangle \frac{|0\rangle + (-1)^{b\oplus f(x)}|1\rangle}{\sqrt{2}} \xrightarrow{H \text{ on } b} |x\rangle|b \oplus f(x)\rangle.$$

So one query to a controlled-O_f oracle simulates B_f.

Outline

Deutsch's algorithm



$$|0\rangle \xrightarrow{H} \frac{|0\rangle + |1\rangle}{\sqrt{2}} \xrightarrow{O_f} \frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}}$$

$$= (-1)^{f(0)} \frac{|0\rangle + (-1)^{f(0)\oplus f(1)}|1\rangle}{\sqrt{2}} \xrightarrow{H} (-1)^{f(0)}|f(0) \oplus f(1)\rangle.$$

Deutsch-Jozsa problem

Definition 3 (Deutsch-Jozsa problem)

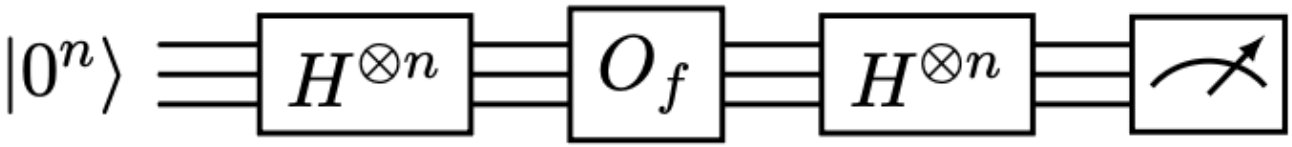
Let $f : \{0,1\}^n \to \{0,1\}$.

Promise: f is either

- **Constant:** $f(x) = f(y)$ for all $x, y \in \{0,1\}^n$
- **Balanced:** $|\{x \mid f(x) = 1\}| = 2^{n-1}$

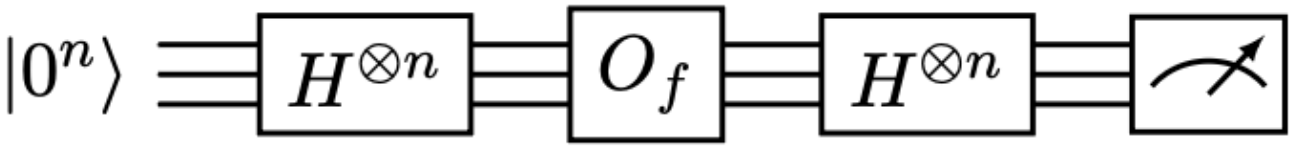
Goal: Determine whether f is constant or balanced.

Deutsch-Jozsa algorithm



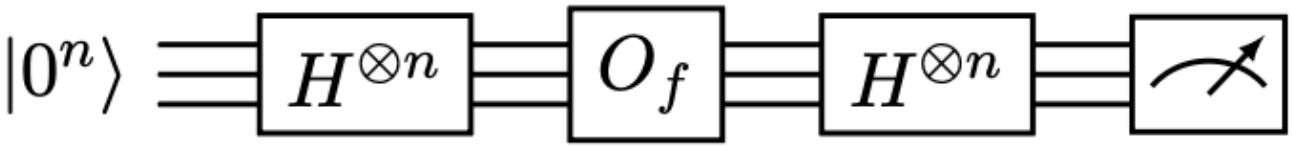
- The circuit is essentially the same as Deutsch's algorithm
- Apply $H^{\otimes n}$, then O_f, then $H^{\otimes n}$
- Quantum query cost: just **one** call to the oracle

Deutsch-Jozsa algorithm



- The circuit is essentially the same as Deutsch's algorithm
- Apply $H^{\otimes n}$, then O_f, then $H^{\otimes n}$
- Quantum query cost: just **one** call to the oracle

Deutsch-Jozsa algorithm



- The circuit is essentially the same as Deutsch's algorithm
- Apply $H^{\otimes n}$, then O_f, then $H^{\otimes n}$
- Quantum query cost: just **one** call to the oracle

Deutsch-Jozsa: State Evolution

$$|0^n\rangle \xrightarrow{H^{\otimes n}} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle$$

$$\xrightarrow{O_f} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle$$

$$\xrightarrow{H^{\otimes n}} \frac{1}{2^n} \sum_{x,y\in\{0,1\}^n} (-1)^{f(x)+x\cdot y}|y\rangle$$

Here $x \cdot y = \sum_{i=1}^{n} x_i y_i$, and

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle.$$

Deutsch-Jozsa: State Evolution

$$|0^n\rangle \xrightarrow{H^{\otimes n}} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle$$

$$\xrightarrow{O_f} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle$$

$$\xrightarrow{H^{\otimes n}} \frac{1}{2^n} \sum_{x,y\in\{0,1\}^n} (-1)^{f(x)+x\cdot y}|y\rangle$$

Here $x \cdot y = \sum_{i=1}^{n} x_i y_i$, and

$$\boxed{H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle.}$$

Deutsch-Jozsa: Measurement

- Look at the amplitude on $|0^n\rangle$

$$\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x)}$$

- If f is constant, this becomes $(-1)^{f(0^n)}$, so we measure $|0^n\rangle$ with probability 1
- If f is balanced, half the terms are $+1$ and half are -1, so the amplitude is 0

Deutsch-Jozsa: Measurement

- Look at the amplitude on $|0^n\rangle$

$$\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x)}$$

- If f is constant, this becomes $(-1)^{f(0^n)}$, so we measure $|0^n\rangle$ with probability 1
- If f is balanced, half the terms are $+1$ and half are -1, so the amplitude is 0

Deutsch-Jozsa: Measurement

- Look at the amplitude on $|0^n\rangle$

$$\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x)}$$

- If f is constant, this becomes $(-1)^{f(0^n)}$, so we measure $|0^n\rangle$ with probability 1
- If f is balanced, half the terms are $+1$ and half are -1, so the amplitude is 0

Deutsch-Jozsa: Measurement

- Look at the amplitude on $|0^n\rangle$

$$\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x)}$$

- If f is constant, this becomes $(-1)^{f(0^n)}$, so we measure $|0^n\rangle$ with probability 1
- If f is balanced, half the terms are $+1$ and half are -1, so the amplitude is 0

Measure $|0^n\rangle$: constant. Measure anything else: balanced.

Bernstein-Vazirani problem

Definition 4 (Bernstein-Vazirani problem)

Let $f : \{0,1\}^n \to \{0,1\}$.

Promise: f is linear:

$$f(x) = x \cdot s$$

for some secret string $s \in \{0,1\}^n$.

Goal: Find s.

Bernstein-Vazirani algorithm

- Classically, we need n queries
- Querying e_i reveals the ith bit of s
- Quantumly, the circuit is again the Deutsch-Jozsa circuit

Bernstein-Vazirani algorithm

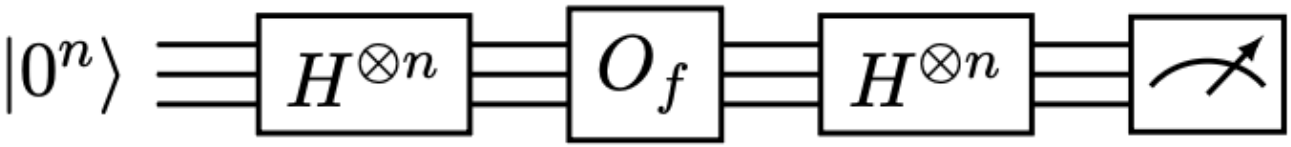
- Classically, we need n queries
- Querying e_i reveals the ith bit of s
- Quantumly, the circuit is again the Deutsch-Jozsa circuit

Bernstein-Vazirani algorithm

- Classically, we need n queries
- Querying e_i reveals the ith bit of s
- Quantumly, the circuit is again the Deutsch-Jozsa circuit

Bernstein-Vazirani algorithm

- Classically, we need n queries
- Querying e_i reveals the ith bit of s
- Quantumly, the circuit is again the Deutsch-Jozsa circuit



Same circuit, different promise, different conclusion.

Bernstein-Vazirani: Output

$$\frac{1}{2^n} \sum_{x,y \in \{0,1\}^n} (-1)^{f(x)+x\cdot y} |y\rangle = \frac{1}{2^n} \sum_{x,y \in \{0,1\}^n} (-1)^{x\cdot(s \oplus y)} |y\rangle = |s\rangle.$$

- If $y = s$, every phase is 1
- So the amplitude on $|s\rangle$ is exactly 1
- Therefore, the measurement returns $|s\rangle$ with probability 1

Bernstein-Vazirani: Output

$$\frac{1}{2^n} \sum_{x,y \in \{0,1\}^n} (-1)^{f(x)+x\cdot y} |y\rangle = \frac{1}{2^n} \sum_{x,y \in \{0,1\}^n} (-1)^{x\cdot(s \oplus y)} |y\rangle = |s\rangle.$$

- If $y = s$, every phase is 1
- So the amplitude on $|s\rangle$ is exactly 1
- Therefore, the measurement returns $|s\rangle$ with probability 1

Bernstein-Vazirani: Output

$$\frac{1}{2^n} \sum_{x,y \in \{0,1\}^n} (-1)^{f(x)+x\cdot y} |y\rangle = \frac{1}{2^n} \sum_{x,y \in \{0,1\}^n} (-1)^{x\cdot(s \oplus y)} |y\rangle = |s\rangle.$$

- If $y = s$, every phase is 1
- So the amplitude on $|s\rangle$ is exactly 1
- Therefore, the measurement returns $|s\rangle$ with probability 1

Bernstein-Vazirani: Output

$$\frac{1}{2^n} \sum_{x,y \in \{0,1\}^n} (-1)^{f(x)+x\cdot y} |y\rangle = \frac{1}{2^n} \sum_{x,y \in \{0,1\}^n} (-1)^{x\cdot(s \oplus y)} |y\rangle = |s\rangle.$$

- If $y = s$, every phase is 1
- So the amplitude on $|s\rangle$ is exactly 1
- Therefore, the measurement returns $|s\rangle$ with probability 1

Quantum: 1 query. Classical: n queries.

Survey time!

How are you doing so far? Any questions or comments on the material covered up to this point?

Outline

What Have We Seen So Far?

- Deutsch-Jozsa gives a dramatic quantum-classical separation
- Bernstein-Vazirani gives a clean 1 vs. n separation
- These examples make quantum query algorithms look extremely powerful

What Have We Seen So Far?

- Deutsch-Jozsa gives a dramatic quantum-classical separation
- Bernstein-Vazirani gives a clean 1 vs. n separation
- These examples make quantum query algorithms look extremely powerful

What Have We Seen So Far?

- Deutsch-Jozsa gives a dramatic quantum-classical separation
- Bernstein-Vazirani gives a clean 1 vs. n separation
- These examples make quantum query algorithms look extremely powerful

What Have We Seen So Far?

- Deutsch-Jozsa gives a dramatic quantum-classical separation
- Bernstein-Vazirani gives a clean 1 vs. n separation
- These examples make quantum query algorithms look extremely powerful

But there is a catch: both are promise problems.

Promise Problems vs Total Problems

- In a promise problem, the input is guaranteed to have special structure
- Our algorithm is only judged on inputs satisfying that promise
- In a total problem, the problem must be defined on all possible inputs
- Once we move to total functions, the landscape changes dramatically

Promise Problems vs Total Problems

- In a promise problem, the input is guaranteed to have special structure
- Our algorithm is only judged on inputs satisfying that promise
- In a total problem, the problem must be defined on all possible inputs
- Once we move to total functions, the landscape changes dramatically

Promise Problems vs Total Problems

- In a promise problem, the input is guaranteed to have special structure
- Our algorithm is only judged on inputs satisfying that promise
- In a total problem, the problem must be defined on all possible inputs
- Once we move to total functions, the landscape changes dramatically

Promise Problems vs Total Problems

- In a promise problem, the input is guaranteed to have special structure
- Our algorithm is only judged on inputs satisfying that promise
- In a total problem, the problem must be defined on all possible inputs
- Once we move to total functions, the landscape changes dramatically

A Limitation for Total Functions

Theorem 5 (Aaronson et al. [ABDKT20])

For a total function f,

$$D(f) \leq Q(f)^4.$$

Here, $D(f)$ denotes deterministic classical query complexity, and $Q(f)$ denotes quantum query complexity.

- So total functions cannot exhibit exponential quantum speedups
- At best, the separation is polynomial (at most quartic)
- The huge gaps we saw earlier rely on structure in the input promise

Back to the Grover Search

- Grover solves unstructured search
- Classical query complexity is $\Theta(2^n)$
- Quantum query complexity is $O(\sqrt{2^n})$
- So Grover gives a quadratic speedup

Back to the Grover Search

- Grover solves unstructured search
- Classical query complexity is $\Theta(2^n)$
- Quantum query complexity is $O(\sqrt{2^n})$
- So Grover gives a quadratic speedup

Back to the Grover Search

- Grover solves unstructured search
- Classical query complexity is $\Theta(2^n)$
- Quantum query complexity is $O(\sqrt{2^n})$
- So Grover gives a quadratic speedup

Back to the Grover Search

- Grover solves unstructured search
- Classical query complexity is $\Theta(2^n)$
- Quantum query complexity is $O(\sqrt{2^n})$
- So Grover gives a quadratic speedup

Back to the Grover Search

- Grover solves unstructured search
- Classical query complexity is $\Theta(2^n)$
- Quantum query complexity is $O(\sqrt{2^n})$
- So Grover gives a quadratic speedup

But if quartic is the maximum for total functions, is quadratic really the end of the story?

BBBV Lower Bound

- Theorem 3.3 still leaves open the possibility of something like $O(2^{n/4})$
- Unfortunately, that is not possible.

BBBV Lower Bound

- Theorem 3.3 still leaves open the possibility of something like $O(2^{n/4})$
- Unfortunately, that is not possible.

BBBV Lower Bound

- Theorem 3.3 still leaves open the possibility of something like $O(2^{n/4})$
- Unfortunately, that is not possible.

Theorem 6 (Bennett et al. [BBBV97])

The quantum query complexity of unstructured search is

$$\Omega(\sqrt{2^n}).$$

let's prove it!

BBBV Proof Roadmap

- Compare two worlds: no marked item vs exactly one marked item
- Find an input y that the algorithm barely touches
- Show that changing the oracle only on that hidden point barely changes the final state
- Conclude that a low-query algorithm cannot reliably distinguish the two worlds

BBBV Proof Roadmap

- Compare two worlds: no marked item vs exactly one marked item
- Find an input y that the algorithm barely touches
- Show that changing the oracle only on that hidden point barely changes the final state
- Conclude that a low-query algorithm cannot reliably distinguish the two worlds

BBBV Proof Roadmap

- Compare two worlds: no marked item vs exactly one marked item
- Find an input y that the algorithm barely touches
- Show that changing the oracle only on that hidden point barely changes the final state
- Conclude that a low-query algorithm cannot reliably distinguish the two worlds

BBBV Proof Roadmap

- Compare two worlds: no marked item vs exactly one marked item
- Find an input y that the algorithm barely touches
- Show that changing the oracle only on that hidden point barely changes the final state
- Conclude that a low-query algorithm cannot reliably distinguish the two worlds

BBBV Proof Roadmap

- Compare two worlds: no marked item vs exactly one marked item
- Find an input y that the algorithm barely touches
- Show that changing the oracle only on that hidden point barely changes the final state
- Conclude that a low-query algorithm cannot reliably distinguish the two worlds

$$U_T O_f U_{T-1} \cdots O_f U_1 O_f U_0 |0^n\rangle$$

A T-query algorithm is just a sequence of fixed unitaries alternating with oracle calls.

Step 1: Find a Point the Algorithm Barely Sees

If T is small, the algorithm cannot place large amplitude on every input.

$$|\psi_t\rangle := U_t U_{t-1} \cdots U_1 U_0 |0^n\rangle = \sum_{x \in \{0,1\}^n} \alpha_{x,t} |x\rangle$$

$$m_x := \sum_{t=0}^{T-1} |\alpha_{x,t}|^2$$

- m_x measures how much total amplitude the algorithm places on input x
- Since

$$\sum_x m_x = T,$$

there exists some y with

$$m_y \leq \frac{T}{2^n}$$

- So at least one input y is barely examined

Step 1: Find a Point the Algorithm Barely Sees

If T is small, the algorithm cannot place large amplitude on every input.

$$|\psi_t\rangle := U_t U_{t-1} \cdots U_1 U_0 |0^n\rangle = \sum_{x \in \{0,1\}^n} \alpha_{x,t} |x\rangle$$

$$m_x := \sum_{t=0}^{T-1} |\alpha_{x,t}|^2$$

- m_x measures how much total amplitude the algorithm places on input x
- Since

$$\sum_x m_x = T,$$

there exists some y with

$$m_y \leq \frac{T}{2^n}$$

- So at least one input y is barely examined

Step 1: Find a Point the Algorithm Barely Sees

If T is small, the algorithm cannot place large amplitude on every input.

$$|\psi_t\rangle := U_t U_{t-1} \cdots U_1 U_0 |0^n\rangle = \sum_{x \in \{0,1\}^n} \alpha_{x,t} |x\rangle$$

$$m_x := \sum_{t=0}^{T-1} |\alpha_{x,t}|^2$$

- m_x measures how much total amplitude the algorithm places on input x
- Since

$$\sum_x m_x = T,$$

there exists some y with

$$m_y \leq \frac{T}{2^n}$$

- So at least one input y is barely examined

Step 2: Hide the Marked Item at That Point

We now choose the hardest one-marked-item instance by marking exactly that neglected point y.

Let f be the function that is 1 only on y and 0 elsewhere.

- Define $O^{(t)}$ so that the first t queries are identity
- The remaining $T - t$ queries use the real oracle O_f
- This lets us move gradually from the zero function to the one-marked-item function

Step 2: Hide the Marked Item at That Point

We now choose the hardest one-marked-item instance by marking exactly that neglected point y.

Let f be the function that is 1 only on y and 0 elsewhere.

- Define $O^{(t)}$ so that the first t queries are identity
- The remaining $T - t$ queries use the real oracle O_f
- This lets us move gradually from the zero function to the one-marked-item function

Step 2: Hide the Marked Item at That Point

We now choose the hardest one-marked-item instance by marking exactly that neglected point y.

Let f be the function that is 1 only on y and 0 elsewhere.

- Define $O^{(t)}$ so that the first t queries are identity
- The remaining $T - t$ queries use the real oracle O_f
- This lets us move gradually from the zero function to the one-marked-item function

Step 2: Hide the Marked Item at That Point

We now choose the hardest one-marked-item instance by marking exactly that neglected point y.

Let f be the function that is 1 only on y and 0 elsewhere.

- Define $O^{(t)}$ so that the first t queries are identity
- The remaining $T - t$ queries use the real oracle O_f
- This lets us move gradually from the zero function to the one-marked-item function

$$|\phi^{(t)}\rangle := U_T O^{(t)} U_{T-1} \cdots O^{(t)} U_1 O^{(t)} U_0 |0^n\rangle$$

$|\phi^{(T)}\rangle$ is the final state for the all-zero function, and $|\phi^{(0)}\rangle$ is the final state for the function with one marked item.

Step 3: Each Hybrid Change Is Tiny

Changing one oracle call matters only through the amplitude already sitting on y.

$$\| |\phi^{(t+1)}\rangle - |\phi^{(t)}\rangle \| = \| |\psi_t\rangle - O_f|\psi_t\rangle \| = 2|\alpha_{y,t}|$$

- Each hybrid step changes the state only through the amplitude sitting on y
- But y was chosen so that the algorithm barely touches it
- Using Cauchy-Schwarz,

$$\sum_{t=0}^{T-1} |\alpha_{y,t}| \leq \frac{T}{\sqrt{2^n}}$$

- Therefore the total change across all hybrids is small:

$$\| |\phi^{(T)}\rangle - |\phi^{(0)}\rangle \| \leq \frac{2T}{\sqrt{2^n}}$$

Step 3: Each Hybrid Change Is Tiny

Changing one oracle call matters only through the amplitude already sitting on y.

$$\| |\phi^{(t+1)}\rangle - |\phi^{(t)}\rangle \| = \| |\psi_t\rangle - O_f|\psi_t\rangle \| = 2|\alpha_{y,t}|$$

- Each hybrid step changes the state only through the amplitude sitting on y
- But y was chosen so that the algorithm barely touches it
- Using Cauchy-Schwarz,

$$\sum_{t=0}^{T-1} |\alpha_{y,t}| \leq \frac{T}{\sqrt{2^n}}$$

- Therefore the total change across all hybrids is small:

$$\| |\phi^{(T)}\rangle - |\phi^{(0)}\rangle \| \leq \frac{2T}{\sqrt{2^n}}$$

Step 3: Each Hybrid Change Is Tiny

Changing one oracle call matters only through the amplitude already sitting on y.

$$\| |\phi^{(t+1)}\rangle - |\phi^{(t)}\rangle \| = \| |\psi_t\rangle - O_f|\psi_t\rangle \| = 2|\alpha_{y,t}|$$

- Each hybrid step changes the state only through the amplitude sitting on y
- But y was chosen so that the algorithm barely touches it
- Using Cauchy-Schwarz,

$$\sum_{t=0}^{T-1} |\alpha_{y,t}| \leq \frac{T}{\sqrt{2^n}}$$

- Therefore the total change across all hybrids is small:

$$\| |\phi^{(T)}\rangle - |\phi^{(0)}\rangle \| \leq \frac{2T}{\sqrt{2^n}}$$

Step 3: Each Hybrid Change Is Tiny

Changing one oracle call matters only through the amplitude already sitting on y.

$$\| |\phi^{(t+1)}\rangle - |\phi^{(t)}\rangle \| = \| |\psi_t\rangle - O_f|\psi_t\rangle \| = 2|\alpha_{y,t}|$$

- Each hybrid step changes the state only through the amplitude sitting on y
- But y was chosen so that the algorithm barely touches it
- Using Cauchy-Schwarz,

$$\sum_{t=0}^{T-1} |\alpha_{y,t}| \leq \frac{T}{\sqrt{2^n}}$$

- Therefore the total change across all hybrids is small:

$$\| |\phi^{(T)}\rangle - |\phi^{(0)}\rangle \| \leq \frac{2T}{\sqrt{2^n}}$$

Step 4: Close States Give Similar Outputs

If the final states are close, then no measurement can separate them well.

Observation 7

If two quantum states are close in Euclidean norm, then their measurement distributions are also close.

$$\mathrm{TV}(p, q) = \frac{1}{2}\|p - q\|_1$$

- Here $\mathrm{TV}(p, q)$ is total variation distance
- Small total variation means no measurement can reliably tell the two cases apart
- So if

$$\||\phi^{(T)}\rangle - |\phi^{(0)}\rangle\| \ll 1,$$

then the algorithm fails to distinguish "all 0" from "one marked item"

Step 4: Close States Give Similar Outputs

If the final states are close, then no measurement can separate them well.

Observation 7

If two quantum states are close in Euclidean norm, then their measurement distributions are also close.

$$\mathrm{TV}(p, q) = \frac{1}{2}\|p - q\|_1$$

- Here $\mathrm{TV}(p, q)$ is total variation distance
- Small total variation means no measurement can reliably tell the two cases apart
- So if

$$\||\phi^{(T)}\rangle - |\phi^{(0)}\rangle\| \ll 1,$$

then the algorithm fails to distinguish "all 0" from "one marked item"

Step 4: Close States Give Similar Outputs

If the final states are close, then no measurement can separate them well.

Observation 7

If two quantum states are close in Euclidean norm, then their measurement distributions are also close.

$$\mathrm{TV}(p, q) = \frac{1}{2}\|p - q\|_1$$

- Here $\mathrm{TV}(p, q)$ is total variation distance
- Small total variation means no measurement can reliably tell the two cases apart
- So if

$$\||\phi^{(T)}\rangle - |\phi^{(0)}\rangle\| \ll 1,$$

then the algorithm fails to distinguish "all 0" from "one marked item"

Why Distinguishing Matters

A search algorithm must behave differently on the two worlds we are comparing.

- World A: all outputs are 0000000000
- World B: there is exactly one marked item ...0001000000...
- A correct search algorithm must distinguish A from B.

Why Distinguishing Matters

A search algorithm must behave differently on the two worlds we are comparing.

- World A: all outputs are 0000000000
- World B: there is exactly one marked item ...0001000000...
- A correct search algorithm must distinguish A from B.

Why Distinguishing Matters

A search algorithm must behave differently on the two worlds we are comparing.

- World A: all outputs are 0000000000
- World B: there is exactly one marked item ...0001000000...
- A correct search algorithm must distinguish A from B.

Why Distinguishing Matters

A search algorithm must behave differently on the two worlds we are comparing.

- World A: all outputs are 0000000000
- World B: there is exactly one marked item ...0001000000...
- A correct search algorithm must distinguish A from B.

Why Distinguishing Matters

A search algorithm must behave differently on the two worlds we are comparing.

- World A: all outputs are 0000000000
- World B: there is exactly one marked item ...0001000000...
- A correct search algorithm must distinguish A from B.

Proposition 8

If we sample from p or q with equal prior probability, then any distinguisher succeeds with probability at most

$$\frac{1}{2} + \frac{\mathrm{TV}(p, q)}{2}.$$

Why This Proposition Is True

A distinguisher is just choosing a set S of outcomes on which it says "this came from p."

$$\Pr[\text{success}] = \frac{1}{2}p(S) + \frac{1}{2}q(S^c) = \frac{1}{2} + \frac{1}{2}\big(p(S) - q(S)\big)$$

- So the best distinguisher tries to maximize $p(S) - q(S)$
- Equivalently,

$$\mathrm{TV}(p, q) = \max_S \big(p(S) - q(S)\big)$$

- Therefore,

$$\Pr[\text{success}] \leq \frac{1}{2} + \frac{\mathrm{TV}(p, q)}{2}$$

Why This Proposition Is True

A distinguisher is just choosing a set S of outcomes on which it says "this came from p."

$$\Pr[\text{success}] = \frac{1}{2}p(S) + \frac{1}{2}q(S^c) = \frac{1}{2} + \frac{1}{2}\big(p(S) - q(S)\big)$$

- So the best distinguisher tries to maximize $p(S) - q(S)$
- Equivalently,

$$\mathrm{TV}(p, q) = \max_S \big(p(S) - q(S)\big)$$

- Therefore,

$$\Pr[\text{success}] \leq \frac{1}{2} + \frac{\mathrm{TV}(p, q)}{2}$$

Why This Proposition Is True

A distinguisher is just choosing a set S of outcomes on which it says "this came from p."

$$\Pr[\text{success}] = \frac{1}{2}p(S) + \frac{1}{2}q(S^c) = \frac{1}{2} + \frac{1}{2}\big(p(S) - q(S)\big)$$

- So the best distinguisher tries to maximize $p(S) - q(S)$
- Equivalently,
$$\mathrm{TV}(p, q) = \max_S \big(p(S) - q(S)\big)$$
- Therefore,
$$\Pr[\text{success}] \leq \frac{1}{2} + \frac{\mathrm{TV}(p, q)}{2}$$

BBBV Conclusion

Search is impossible if those two worlds remain almost indistinguishable.

$$\Pr[\text{distinguish correctly}] \leq \frac{1}{2} + \frac{\mathrm{TV}(p,q)}{2} \leq \frac{1}{2} + \frac{2T}{\sqrt{2^n}}$$

- To succeed with constant bias above $1/2$, we need

$$T = \Omega(2^{n/2})$$

- So unstructured search requires

$$\Omega(\sqrt{2^n})$$

quantum queries

BBBV Conclusion

Search is impossible if those two worlds remain almost indistinguishable.

$$\Pr[\text{distinguish correctly}] \leq \frac{1}{2} + \frac{\mathrm{TV}(p,q)}{2} \leq \frac{1}{2} + \frac{2T}{\sqrt{2^n}}$$

- To succeed with constant bias above $1/2$, we need

$$T = \Omega(2^{n/2})$$

- So unstructured search requires

$$\Omega(\sqrt{2^n})$$

quantum queries

BBBV Conclusion

Search is impossible if those two worlds remain almost indistinguishable.

$$\Pr[\text{distinguish correctly}] \leq \frac{1}{2} + \frac{\mathrm{TV}(p,q)}{2} \leq \frac{1}{2} + \frac{2T}{\sqrt{2^n}}$$

- To succeed with constant bias above $1/2$, we need

$$T = \Omega(2^{n/2})$$

- So unstructured search requires

$$\Omega(\sqrt{2^n})$$

quantum queries

That is exactly the scale achieved by Grover's algorithm.

Why Grover Matters

- Grover achieves $O(\sqrt{2^n})$ queries
- BBBV proves that $\Omega(\sqrt{2^n})$ queries are necessary
- So Grover is not just fast
- It is asymptotically optimal in the query model

Why Grover Matters

- Grover achieves $O(\sqrt{2^n})$ queries
- BBBV proves that $\Omega(\sqrt{2^n})$ queries are necessary
- So Grover is not just fast
- It is asymptotically optimal in the query model

Why Grover Matters

- Grover achieves $O(\sqrt{2^n})$ queries
- BBBV proves that $\Omega(\sqrt{2^n})$ queries are necessary
- So Grover is not just fast
- It is asymptotically optimal in the query model

Why Grover Matters

- Grover achieves $O(\sqrt{2^n})$ queries
- BBBV proves that $\Omega(\sqrt{2^n})$ queries are necessary
- So Grover is not just fast
- It is asymptotically optimal in the query model

Why Grover Matters

- Grover achieves $O(\sqrt{2^n})$ queries
- BBBV proves that $\Omega(\sqrt{2^n})$ queries are necessary
- So Grover is not just fast
- It is asymptotically optimal in the query model

That is why Grover is the central search algorithm, not just a nice example.

Quiz time!

Only one quiz...

Outline

References I

[ABDKT20] Scott Aaronson, Shalev Ben-David, Robin Kothari, and Avishay Tal.
Quantum implications of huang's sensitivity theorem, 2020.

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh V. Vazirani.
Strengths and weaknesses of quantum computing.
SIAM Journal on Computing, 26(5):1510–1523, 1997.

[Gri24] Daniel Grier.
Quantum complexity theory: Cse 291 / math 277a - fall 2024.
Course website and lecture notes, 2024.
Accessed 2026-04-02.

References II

[NC10] Michael A Nielsen and Isaac L Chuang.
Quantum computation and quantum information.
Cambridge university press, 2010.