A Classical Approach to Quantum Algorithms

Troy Lee
Centre for Quantum Technologies
Disclaimer

No knowledge of quantum mechanics is needed for this talk.
Disclaimer

No knowledge of quantum mechanics is needed for this talk.

The main result characterizes the complexity of (a model) of quantum algorithms as a classical optimization problem.
Disclaimer

No knowledge of quantum mechanics is needed for this talk.

The main result characterizes the complexity of (a model) of quantum algorithms as a classical optimization problem.

To design quantum algorithms, it suffices to find solutions to this optimization problem!
Disclaimer

No knowledge of quantum mechanics is needed for this talk.

The main result characterizes the complexity of (a model) of quantum algorithms as a classical optimization problem.

To design quantum algorithms, it suffices to find solutions to this optimization problem!

This paradigm has led to improved quantum algorithms for several problems.
That said...

Here is a two slide introduction to quantum computing.
That said...

Here is a two slide introduction to quantum computing.

Let’s think first about probabilistic computation.
That said...

Here is a two slide introduction to quantum computing.

Let’s think first about probabilistic computation.

The state of a randomized algorithm is described by a probability vector.

\[(p_1, \ldots, p_m) : \sum_i p_i = 1, p_i \geq 0\]
That said...

Here is a two slide introduction to quantum computing.

Let’s think first about probabilistic computation.

The state of a randomized algorithm is described by a probability vector.

\[(p_1, \ldots, p_m) : \sum_i p_i = 1, p_i \geq 0\]

Allowed transformations take probability vectors to probability vectors.
That said...

Here is a two slide introduction to quantum computing.

Let's think first about probabilistic computation.

The state of a randomized algorithm is described by a probability vector.

$$(p_1, \ldots, p_m) : \sum_i p_i = 1, p_i \geq 0$$

Allowed transformations take probability vectors to probability vectors. Stochastic matrices!
Quantum Analogue

The state of an algorithm is described by a vector of complex numbers.

\((\alpha_1, \ldots, \alpha_m) : \sum_i |\alpha_i|^2 = 1\)
Quantum Analogue

The state of an algorithm is described by a vector of complex numbers.

\[(\alpha_1, \ldots, \alpha_m) : \sum_i |\alpha_i|^2 = 1\]

Allowed operations send unit vectors to unit vectors.
Quantum Analogue

The state of an algorithm is described by a vector of complex numbers.

$$(\alpha_1, \ldots, \alpha_m) : \sum_i |\alpha_i|^2 = 1$$

Allowed operations send unit vectors to unit vectors. Unitary matrices!
Quantum Analogue

The state of an algorithm is described by a vector of complex numbers.

\((\alpha_1, \ldots, \alpha_m) : \sum_i |\alpha_i|^2 = 1\)

Allowed operations send unit vectors to unit vectors. Unitary matrices!

A unitary matrix \(U\) satisfies \(UU^* = I\)
Quantum Analogue

The state of an algorithm is described by a vector of complex numbers.

\[(\alpha_1, \ldots, \alpha_m) : \sum_i |\alpha_i|^2 = 1\]

Allowed operations send unit vectors to unit vectors. Unitary matrices!

A unitary matrix $U$ satisfies $UU^* = I$

When make a measurement, in state $i$ with probability $|\alpha_i|^2$
Interference
The main difference with probability distributions is interference!
Interference

The main difference with probability distributions is interference!

A simple example with two states

\[ |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \]
Interference
The main difference with probability distributions is interference!

A simple example with two states

\[ |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \]

\[
\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}
\]
Interference

The main difference with probability distributions is interference!

A simple example with two states

\[
|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}
\]

\[
\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}
\]

\[
\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \left( \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}
\]
Interference

The main difference with probability distributions is interference!

A simple example with two states

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

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

$$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \left( \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
Famous Quantum Algorithms

Shor’s algorithm (1994): Given an integer $n = p \times q$, a quantum computer can quickly find $p$ and $q$. 
Shor’s algorithm (1994): Given an integer \( n = p \times q \), a quantum computer can quickly find \( p \) and \( q \).

This would break the popular cryptography scheme RSA.
Famous Quantum Algorithms

Shor’s algorithm (1994): Given an integer $n = p \times q$, a quantum computer can quickly find $p$ and $q$.

This would break the popular cryptography scheme RSA.

Shor’s algorithm is exponentially faster than the best known classical algorithm.
Query Model

Most popular model for studying quantum algorithms.

Want to evaluate a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$

How many questions of the form $x_i = ?$ are needed to evaluate $f(x)$?

$D(f)$ is the number of queries needed by deterministic algorithm that always gives the right answer.

$Q(f)$ is the number of queries needed by quantum algorithm that always succeeds with prob. $\geq 2/3$
Quantum Algorithms

For other problems, quantum offers polynomial speedups.
Quantum Algorithms

For other problems, quantum offers polynomial speedups.

Grover’s algorithm: Can search a list of $n$ items for a desired item in time $\sqrt{n}$
Quantum Algorithms

For other problems, quantum offers polynomial speedups.

**Grover’s algorithm**: Can search a list of $n$ items for a desired item in time $\sqrt{n}$
Quantum Algorithms

For other problems, quantum offers polynomial speedups.

Grover’s algorithm: Can search a list of $n$ items for a desired item in time $\sqrt{n}$

Searching is one of the most common tasks performed by computers!
Quantum Algorithms

For other problems, quantum offers polynomial speedups.

**Ambainis’s algorithm:** Are two items in a list of $n$ items the same? Can do it with $n^{2/3}$ queries.
Quantum Algorithms

For other problems, quantum offers polynomial speedups.

Ambainis’s algorithm: Are two items in a list of $n$ items the same? Can do it with $n^{2/3}$ queries.
Quantum Algorithms

For other problems, quantum offers polynomial speedups.

Ambainis’s algorithm: Are two items in a list of $n$ items the same? Can do it with $n^{2/3}$ queries.

These problems are fundamental elements of an algorithmic and cryptographic toolkit!
Polynomial Speedups

Theorem (BBCMW ’98):
For any function $f : \{0, 1\}^n \to \{0, 1\}$ the quantum and deterministic query complexities are polynomially related.

\[ D(f) \leq Q(f)^6 \]

It is conjectured that they are always related by a square.
Certificate Complexity

A certificate is an assignment to a subset of the indices that determines the function’s value.

Example: Two balls of the same color certifies that there is a collision.

On the other hand, to certify that there is no collision, we need to see all the balls.
Certificate Complexity

Let $C_0(f)$ be the size of a largest 0-certificate and similarly for $C_1(f)$

Basic fact: any 0-certificate and 1-certificate must intersect.
Certificate Complexity

Let $C_0(f)$ be the size of a largest 0-certificate and similarly for $C_1(f)$.

Basic fact: any 0-certificate and 1-certificate must intersect.

We can use certificates to construct an algorithm.

$$D(f) \leq C_0(f)C_1(f)$$
Certificate Complexity

Let $C_0(f)$ be the size of a largest 0-certificate and similarly for $C_1(f)$

Basic fact: any 0-certificate and 1-certificate must intersect.

We can use certificates to construct an algorithm.

$$D(f) \leq C_0(f)C_1(f)$$

Idea: Take some 0-input $x$ consistent with the values seen. Query indices in a 0-certificate for $x$. This reduces the 1-certificate complexity of all remaining 1-inputs by at least one.
Alternative view

$$\max\{C_0(f), C_1(f)\} =$$

$$\min_{p_x} \max_{x \in \{0, 1\}^n} \sum_{i=1}^{n} p_x(i)$$

$$\sum_{i : x_i \neq y_i} p_x(i)p_y(i) \geq 1 \text{ for all } f(x) \neq f(y)$$

$$p_x(i) \in \{0, 1\}$$
Alternative view

\[
\max\{C_0(f), C_1(f)\} = \min_{p_x} \max_{x \in \{0,1\}^n} \sum_{i=1}^{n} p_x(i) \\
\sum_{i : x_i \neq y_i} p_x(i) p_y(i) \geq 1 \text{ for all } f(x) \neq f(y) \\
p_x(i) \in \{0, 1\}
\]

\[
p_x(i) = \begin{array}{cccccccc}
1 & 0 & 0 & 1 & 1 & 1 & 0 & 0
\end{array}
\]
General Adversary Bound

The general adversary bound was introduced as a lower bound technique for quantum query complexity \[\text{[Hoyer, L., Spalek '07]}\]

\[
\text{Adv}^\pm(f) = \min_{u_{x,i}} \max_{x \in \{0,1\}^n} \sum_{i=1}^{n} \|u_{x,i}\|^2 \\
\sum_{i : x_i \neq y_i} \langle u_{x,i}, u_{y,i} \rangle = 1 \\
\text{for all } f(x) \neq f(y)
\]
The general adversary bound was introduced as a lower bound technique for quantum query complexity [Hoyer, L., Spalek ’07]

$$\text{Adv}^\pm(f) = \min_{u_{x,i}} \max_{x \in \{0,1\}^n} \sum_{i=1}^n \|u_{x,i}\|^2$$

$$\sum_{i : x_i \neq y_i} \langle u_{x,i}, u_{y,i} \rangle = 1$$

for all $f(x) \neq f(y)$

Theorem: $Q(f) \geq \text{Adv}^\pm(f)/3$
In 2010, Reichardt showed that the general adversary bound is also an upper bound on quantum query complexity.
In 2010, Reichardt showed that the general adversary bound is also an upper bound on quantum query complexity.

In some follow-up work, L., Mittal, Reichardt, Spalek, Szegedy '11 showed the adversary bound captures the complexity of a much broader class of problems, known as state conversion problems.
A new approach to algorithms...

\[
\text{Adv}^\pm(f) = \min_{u_{x,i}} \max_{x \in \{0,1\}^n} \sum_{i=1}^n \|u_{x,i}\|^2 \\
\sum_{i : x_i \neq y_i} \langle u_{x,i}, u_{y,i} \rangle = 1
\]

for all \( f(x) \neq f(y) \)
A new approach to algorithms...

\[
\text{Adv}^\pm(f) = \min_{u_{x,i}} \max_{x \in \{0,1\}^n} \sum_{i=1}^n \|u_{x,i}\|^2 \\
\sum_{i : x_i \neq y_i} \langle u_{x,i}, u_{y,i} \rangle = 1 \\
\text{for all } f(x) \neq f(y)
\]

Not so easy!
Example: Grover

We want to compute the OR function: determine if the input is identically 0 or not.

**Negative inputs:** There is only one! $0^n$

- Let $u_{x,i} = 1$ for all $i$

**Positive inputs:** Know that $x_i = 1$ for some $i$

- Let $u_{x,i} = 1$ for this $i$
- $u_{x,j} = 0$ for $j \neq i$
Example: Grover

These vectors satisfy the constraint. What about the objective function?
Example: Grover

These vectors satisfy the constraint. What about the objective function?

Negative inputs: \[ \sum_{i=1}^{n} \| u_{x,i} \|^2 = n \]
Example: Grover

These vectors satisfy the constraint. What about the objective function?

Negative inputs: \[ \sum_{i=1}^{n} \| u_{x,i} \|^2 = n \]

Positive inputs: \[ \sum_{i=1}^{n} \| u_{x,i} \|^2 = 1 \]
Example: Grover

These vectors satisfy the constraint. What about the objective function?

Negative inputs: \[ \sum_{i=1}^{n} \| u_{x,i} \|^2 = n \]

Positive inputs: \[ \sum_{i=1}^{n} \| u_{x,i} \|^2 = 1 \]

Balance these out--multiply all positive vectors by \( n^{1/4} \) divide all negative vectors by \( n^{1/4} \)
Example: Grover

These vectors satisfy the constraint. What about the objective function?

Negative inputs: \[ \sum_{i=1}^{n} \|u_{x,i}\|^2 = n \]

Positive inputs: \[ \sum_{i=1}^{n} \|u_{x,i}\|^2 = 1 \]

Balance these out--multiply all positive vectors by \( n^{1/4} \)
divide all negative vectors by \( n^{1/4} \)

This gives a value of \( \sqrt{n} \) like Grover’s algorithm!
Learning Graphs

For more complicated functions, even designing feasible vector solutions can be quite difficult.
Learning Graphs

For more complicated functions, even designing feasible vector solutions can be quite difficult.

Enter learning graphs [Belovs ’11].
Learning Graphs

For more complicated functions, even designing feasible vector solutions can be quite difficult.

Enter learning graphs [Belovs ’11].

A learning graph imposes additional structure on the vectors in the solution. This structure ensures constraints are satisfied automatically.

Lets one focus on achieving a good objective value.
Learning Graphs

For more complicated functions, even designing feasible vector solutions can be quite difficult.

Enter learning graphs [Belovs ’11].

A learning graph imposes additional structure on the vectors in the solution. This structure ensures constraints are satisfied automatically.

Lets one focus on achieving a good objective value.

This has already led to many new algorithms.
A learning graph is a weighted graph, with vertices labeled by $S \subseteq [n]$. There can only be edges between vertices whose labels differ by one element.
Learning Graphs

A learning graph is a weighted graph, with vertices labeled by $S \subseteq [n]$. There can only be edges between vertices whose labels differ by one element.

For each positive input we must design a unit flow, starting from the root and ending at vertices labeled by one-certificate.
Learning graph for OR

A vertex for the empty set and all singleton sets.

Each edge has weight one.
Learning graph for OR

Unit flow on input

\[ x = 0 \cdots 010 \cdots 0 \text{ where } x_i = 1 \]
The weights and flows in learning graphs are used to construct vector solutions to the adversary bound.

Idea: vectors are constructed such that

$$\sum_{i: x_i \neq y_i} \langle u_{x,i}, u_{y,i} \rangle$$

is the flow through a cut separating root from sinks.
Complexity of Learning Graphs

The complexity of a learning graph is:

\[
\left( \sum_{e \in E} w_e \right)^{1/2} \left( \min_{p_y} \max_{y : f(y) = 1} \sum_{e \in E} \frac{p_y(e)^2}{w_e} \right)^{1/2}
\]

The weight of an edge \( w_e \) is like a conductance, so \( 1/w_e \) is a resistance.

The flow which minimizes the second term is the unit current from root to one-certificates.
Electrical Networks

The flow which achieves the minimum is exactly the unit current from the root to one-certificates for \( y \).
Electrical Networks

The flow which achieves the minimum is exactly the unit current from the root to one-certificates for $y$.

This minimum energy is also equal to the effective resistance between the root and the sinks.
Electrical Networks

The flow which achieves the minimum is exactly the unit current from the root to one-certificates for $y$.

This minimum energy is also equal to the effective resistance between the root and the sinks.

Really, all we have to specify in a learning graph are the weights.
Learning graph for OR

Unit flow on input

\[ x = 0 \cdots 010 \cdots 0 \text{ where } x_i = 1 \]
Learning graph for OR

Unit flow on input

\[ x = 0 \cdots 010\cdots 0 \] where \( x_i = 1 \)

Total weight is \( n \)
Learning graph for OR

Unit flow on input

\[ x = 0 \cdots 010 \cdots 0 \text{ where } x_i = 1 \]

Total weight is \( n \)

Energy of the flow is 1
Learning graph for OR

Unit flow on input

\[ x = 0 \cdots 010 \cdots 0 \] where \( x_i = 1 \)

Total weight is \( n \)

Energy of the flow is 1

Again we have complexity \( \sqrt{n} \)
Applications

Even with their additional structure, learning graphs can be hard to analyze.
Applications

Even with their additional structure, learning graphs can be hard to analyze. [L., Magniez, and Santha ’13] develop a framework for designing learning graphs for the problem of subgraph detection. For example, does the input graph contain a triangle?
Applications

Even with their additional structure, learning graphs can be hard to analyze. [L., Magniez, and Santha ’13] develop a framework for designing learning graphs for the problem of subgraph detection. For example, does the input graph contain a triangle?

In this framework, all edges have weight zero or one.
Applications

Even with their additional structure, learning graphs can be hard to analyze. [L., Magniez, and Santha ’13] develop a framework for designing learning graphs for the problem of subgraph detection. For example, does the input graph contain a triangle?

In this framework, all edges have weight zero or one.

The subsets of edges are the union of regular (unbalanced) bipartite graphs. Variables for set sizes and degree.
Triangle Finding

Making queries of the form: is there an edge between vertex $i$ and vertex $j$? Determine if a graph has a triangle.
Triangle Finding

Making queries of the form: is there an edge between vertex i and vertex j? Determine if a graph has a triangle.

Grover ’96: $n^{3/2}$
Triangle Finding

Making queries of the form: is there an edge between vertex i and vertex j? Determine if a graph has a triangle.

Grover ’96: $n^{3/2}$

Magniez, Santha, Szegedy ’05: $n^{1.3}$
Triangle Finding

Making queries of the form: is there an edge between vertex i and vertex j? Determine if a graph has a triangle.

Grover ’96: $n^{3/2}$

Magniez, Santha, Szegedy ’05: $n^{1.3}$

Belovs ’11: $n^{35/27} \approx n^{1.296}$
Triangle Finding

Making queries of the form: is there an edge between vertex i and vertex j? Determine if a graph has a triangle.

Grover ’96: \( n^{3/2} \)

Magniez, Santha, Szegedy ’05: \( n^{1.3} \)

Belovs ’11: \( n^{35/27} \approx n^{1.296} \)

L, Magniez, Santha: \( n^{9/7} \approx n^{1.286} \)
$A_1 : r_1 - 1$ vertices

$A_2 : r_2 - 1$ vertices

$A_1$ with 1 new vertex $\rightarrow r_1$ vertices

$A_2$ with 1 new vertex $\rightarrow r_2$ vertices

$A_3 : 1$ vertex

$A_1$ with $r_1$ vertices all connected to $A_2$

$A_2$ with $r_2$ vertices all connected to $A_1$

$A_3$ : 1 new edge

$A_1$ : $r_1$ vertices all connected to $A_2$

$A_2$ : $r_2$ vertices all connected to $A_1$
Non-Adaptive LGs

We use the most basic “nonadaptive” model of learning graphs.

This model does not depend on the underlying alphabet, just the structure of one-certificates.

For example, threshold 2 and element (non)distinctness have the same one-certificate structure.
Non-Adaptive LGs

We use the most basic “nonadaptive” model of learning graphs.

This model does not depend on the underlying alphabet, just the structure of one-certificates.

For example, threshold 2 and element (non)distinctness have the same one-certificate structure.

\[0010001000\]
\[7850435129\]
Triangle Optimality

Going to a non-boolean alphabet, our algorithms can find subgraphs with specified edge colors in a colored graph.

For example, we can solve the “triangle sum” problem, where edges are labeled by elements of a group and the question is if there are $(i, j), (j, k), (i, k)$ such that

$$e_{i,j} + e_{j,k} + e_{i,k} = 0$$

In a very recent result, Belovs and Rosmanis ’12 show a lower bound for the triangle sum problem of

$$\frac{n^{9/7}}{\sqrt{\log n}}$$
Conclusion

Now we can design quantum algorithms by studying a (relatively) simple optimization problem.

No knowledge of quantum effects is needed!

This paradigm has already led to several new algorithms, for example for telling if an input graph has a triangle.

We are looking for new ways to design vector systems...
Conclusion

Now we can design quantum algorithms by studying a (relatively) simple optimization problem.

No knowledge of quantum effects is needed!

This paradigm has already led to several new algorithms, for example for telling if an input graph has a triangle.

We are looking for new ways to design vector systems...

Thank You!