Quantum and Classical Query Complexities of Functions of Matrices

Changpeng Shao

Academy of Mathematics and Systems Science, Chinese Academy of Sciences
based on joint work with Ashley Montanaro (University of Bristol & Phasecraft)
arXiv:2311.06999

STOC 2024
24-28 June 2024

Quantum linear algebra

- Quantum linear algebra plays an important role in the exploration of quantum advantages.

Quantum linear algebra

- Quantum linear algebra plays an important role in the exploration of quantum advantages.
- Hamiltonian simulation: computing $e^{iAt}|\mathbf{b}\rangle$, where A is Hermitian. A fundamentally important problem in quantum computing.

Quantum linear algebra

- Quantum linear algebra plays an important role in the exploration of quantum advantages.
- Hamiltonian simulation: computing $e^{iAt}|\mathbf{b}\rangle$, where A is Hermitian. A fundamentally important problem in quantum computing.
- The HHL algorithm for linear systems: computing $A^{-1}|\mathbf{b}\rangle$. Greatly promoted the development of quantum linear algebra [Harrow-Hassidim-Lloyd, '08].

Quantum linear algebra

- Quantum linear algebra plays an important role in the exploration of quantum advantages.
- Hamiltonian simulation: computing $e^{iAt}|\mathbf{b}\rangle$, where A is Hermitian. A fundamentally important problem in quantum computing.
- The HHL algorithm for linear systems: computing $A^{-1}|\mathbf{b}\rangle$. Greatly promoted the development of quantum linear algebra [Harrow-Hassidim-Lloyd, '08].
- Has wide applications, such as
  - Quantum recommendation systems: computing $A_{\geq\delta}|i\rangle$, where $A_{\geq\delta}$ is the truncation by keeping singular values larger than δ [Kerenidis-Prakash, '16].
  - Solving linear differential equations: computing $e^{At}|\mathbf{b}\rangle$ [Berry, '10].

Quantum linear algebra

- Quantum linear algebra plays an important role in the exploration of quantum advantages.
- Hamiltonian simulation: computing $e^{iAt}|\mathbf{b}\rangle$, where A is Hermitian. A fundamentally important problem in quantum computing.
- The HHL algorithm for linear systems: computing $A^{-1}|\mathbf{b}\rangle$. Greatly promoted the development of quantum linear algebra [Harrow-Hassidim-Lloyd, '08].
- Has wide applications, such as
  - Quantum recommendation systems: computing $A_{\geq\delta}|i\rangle$, where $A_{\geq\delta}$ is the truncation by keeping singular values larger than δ [Kerenidis-Prakash, '16].
  - Solving linear differential equations: computing $e^{At}|\mathbf{b}\rangle$ [Berry, '10].
- All these problems can be described as functions of matrices: computing $f(A)|\mathbf{b}\rangle$.

Quantum linear algebra

- Quantum linear algebra plays an important role in the exploration of quantum advantages.
- Hamiltonian simulation: computing $e^{iAt}|\mathbf{b}\rangle$, where A is Hermitian. A fundamentally important problem in quantum computing.
- The HHL algorithm for linear systems: computing $A^{-1}|\mathbf{b}\rangle$. Greatly promoted the development of quantum linear algebra [Harrow-Hassidim-Lloyd, '08].
- Has wide applications, such as
  - Quantum recommendation systems: computing $A_{\geq\delta}|i\rangle$, where $A_{\geq\delta}$ is the truncation by keeping singular values larger than δ [Kerenidis-Prakash, '16].
  - Solving linear differential equations: computing $e^{At}|\mathbf{b}\rangle$ [Berry, '10].
- All these problems can be described as functions of matrices: computing $f(A)|\mathbf{b}\rangle$.
- Can be solved by a similar idea to HHL, but more efficiently by quantum singular value transform [Gilyén-Su-Low-Wiebe, '18].

Quantum singular value transform (QSVT)

Assume A is Hermitian and $\|A\| \leq 1$ for simplicity.

Quantum singular value transform (QSVT)

Assume A is Hermitian and $\|A\| \leq 1$ for simplicity.

Given a unitary of the form

$$U = \begin{pmatrix} A & * \\ * & * \end{pmatrix},$$

for any 1-bounded polynomial $f(x)$ of degree d, there is a quantum circuit that implements a unitary

$$\widetilde{U} = \begin{pmatrix} f(A) & * \\ * & * \end{pmatrix}.$$

Moreover, the circuit uses $U, U^\dagger$ and the controlled forms $O(d)$ times in total.

Quantum singular value transform (QSVT)

Assume A is Hermitian and $\|A\| \leq 1$ for simplicity.

Given a unitary of the form

$$U = \begin{pmatrix} A & * \\ * & * \end{pmatrix},$$

for any 1-bounded polynomial $f(x)$ of degree d, there is a quantum circuit that implements a unitary

$$\widetilde{U} = \begin{pmatrix} f(A) & * \\ * & * \end{pmatrix}.$$

Moreover, the circuit uses $U, U^\dagger$ and the controlled forms $O(d)$ times in total.

As a direct result, we can compute $f(A)|\mathbf{b}\rangle$.

Quantum singular value transform (QSVT)

Assume A is Hermitian and $\|A\| \leq 1$ for simplicity.

Given a unitary of the form

$$U = \begin{pmatrix} A & * \\ * & * \end{pmatrix},$$

for any 1-bounded polynomial $f(x)$ of degree d, there is a quantum circuit that implements a unitary

$$\widetilde{U} = \begin{pmatrix} f(A) & * \\ * & * \end{pmatrix}.$$

Moreover, the circuit uses $U, U^\dagger$ and the controlled forms $O(d)$ times in total.

As a direct result, we can compute $f(A)|\mathbf{b}\rangle$.

Still works if f is not a polynomial, just consider its polynomial approximation.

This talk

Motivations: For functions of matrices,

- QSVT is optimal in many cases, how about the general case?
- what is the quantum-classical separation?

This talk

Motivations: For functions of matrices,

- QSVT is optimal in many cases, how about the general case?
- what is the quantum-classical separation?

To resolve the above questions, we consider the following weaker problem

Problem (Approximate an entry of $f(A)$)

Let $f(x) : [-1,1] \to [-1,1]$ be a function, let A be sparse and Hermitian with $\|A\| \le 1$. Given two indices i, j and accuracy ε, compute $\langle i|f(A)|j\rangle \pm \varepsilon$.

This talk

Motivations: For functions of matrices,

- QSVT is optimal in many cases, how about the general case?
- what is the quantum-classical separation?

To resolve the above questions, we consider the following weaker problem

Problem (Approximate an entry of $f(A)$)

Let $f(x) : [-1,1] \to [-1,1]$ be a function, let A be sparse and Hermitian with $\|A\| \leq 1$. Given two indices i,j and accuracy ε, compute $\langle i|f(A)|j\rangle \pm \varepsilon$.

For a sparse matrix $A = (A_{i,j})$, we are given 2 oracles:

$$(i,j) \longrightarrow \boxed{\mathcal{O}_1} \longrightarrow p_{i,j}$$

$$(i,j) \longrightarrow \boxed{\mathcal{O}_2} \longrightarrow A_{i,j}$$

where p_{ij} is the index of the j-th nonzero entry in the i-th row. The query complexity is the minimal number of calls to the oracles to solve the problem.

An example: matrix powers A^d

Classical algorithms:

- Assume A is s-sparse, then by definition

$$(A^d)_{i,j} = \sum_{k_1} \sum_{k_2} \cdots \sum_{k_{d-1}} A_{i,k_1} A_{k_1,k_2} \cdots A_{k_{d-1},j}$$

The complexity is $O(s^{d-1})$

An example: matrix powers A^d

Classical algorithms:

- Assume A is s-sparse, then by definition

$$(A^d)_{i,j} = \sum_{k_1} \sum_{k_2} \cdots \sum_{k_{d-1}} A_{i,k_1} A_{k_1,k_2} \cdots A_{k_{d-1},j}$$

  The complexity is $O(s^{d-1})$

- For x^d, there is an approximation polynomial of degree $\Theta(\sqrt{d})$, so can be improved to $s^{O(\sqrt{d})}$ [Sachdeva & Vishnoi, 2014]

An example: matrix powers A^d

Classical algorithms:

- Assume A is s-sparse, then by definition

$$(A^d)_{i,j} = \sum_{k_1} \sum_{k_2} \cdots \sum_{k_{d-1}} A_{i,k_1} A_{k_1,k_2} \cdots A_{k_{d-1},j}$$

  The complexity is $O(s^{d-1})$
- For x^d, there is an approximation polynomial of degree $\Theta(\sqrt{d})$, so can be improved to $s^{O(\sqrt{d})}$ [Sachdeva & Vishnoi, 2014]
- Our result: $\widetilde{\Omega}((s/2)^{(\sqrt{d}-1)/6})$

An example: matrix powers A^d

Classical algorithms:

- Assume A is s-sparse, then by definition

$$(A^d)_{i,j} = \sum_{k_1}\sum_{k_2}\cdots\sum_{k_{d-1}} A_{i,k_1}A_{k_1,k_2}\cdots A_{k_{d-1},j}$$

  The complexity is $O(s^{d-1})$
- For x^d, there is an approximation polynomial of degree $\Theta(\sqrt{d})$, so can be improved to $s^{O(\sqrt{d})}$ [Sachdeva & Vishnoi, 2014]
- Our result: $\widetilde{\Omega}((s/2)^{(\sqrt{d}-1)/6})$

Quantum algorithms:

- Upper bound by QSVT $O(s\sqrt{d}/\varepsilon)$

An example: matrix powers A^d

Classical algorithms:

- Assume A is s-sparse, then by definition

$$(A^d)_{i,j} = \sum_{k_1} \sum_{k_2} \cdots \sum_{k_{d-1}} A_{i,k_1} A_{k_1,k_2} \cdots A_{k_{d-1},j}$$

  The complexity is $O(s^{d-1})$
- For x^d, there is an approximation polynomial of degree $\Theta(\sqrt{d})$, so can be improved to $s^{O(\sqrt{d})}$ [Sachdeva & Vishnoi, 2014]
- Our result: $\widetilde{\Omega}((s/2)^{(\sqrt{d}-1)/6})$

Quantum algorithms:

- Upper bound by QSVT $O(s\sqrt{d}/\varepsilon)$
- Our result (lower bound): $\Omega(\sqrt{d})$

General case (our results)

Assume f is continuous, A is sparse and Hermitian, then computing $f(A)_{i,j} \pm \varepsilon$ costs

	Quantum algorithm	Classical algorithm
Upper bound	$O(sd/\varepsilon)$	$O(s^{d-1})$
Lower bound	$\Omega(d)$	$\Omega((s/2)^{(d-1)/6})$

where $d = \widetilde{\deg}_\varepsilon(f)$ is the approximate degree:

$$\widetilde{\deg}_\varepsilon(f) = \min\{d : |f(x) - g(x)| \leq \varepsilon, \forall x \in [-1,1], \; g(x) \text{ is a polynomial of degree } d\}.$$

The quantum lower bound is similar to the famous polynomial method for Boolean functions [Beals, Buhrman, Cleve, Mosca, de Wolf, FOCS '98].

Key theorem in the proofs

Theorem (Key theorem)

Let $f : [-1,1] \to [-1,1]$ be continuous with odd and even parts f_{odd}, f_{even}, then

- *there is a symmetric tridiagonal matrix*

$$A = \begin{pmatrix} 0 & b_1 & & & \\ b_1 & 0 & b_2 & & \\ & b_2 & \ddots & \ddots & \\ & & \ddots & \ddots & b_{n-1} \\ & & & b_{n-1} & 0 \end{pmatrix}_{n\times n}$$

  satisfying $b_i \neq 0$, $\|A\| \leq 1$ and $f(A)_{1,n} = \varepsilon$, where $n = \widetilde{\deg}_\varepsilon(f_{\mathrm{odd}}) + O(1)$.
- *A similar result for f_{even}.*

Proof. linear semi-infinite programming + dual polynomial method + properties of tridiagonal matrices. ■

Lower bound's proof of quantum algorithms

Parity problem: Given $x_1, \ldots, x_n \in \{0, 1\}$, compute $x_1 \oplus \cdots \oplus x_n$, the quantum query complexity is $\Theta(n)$

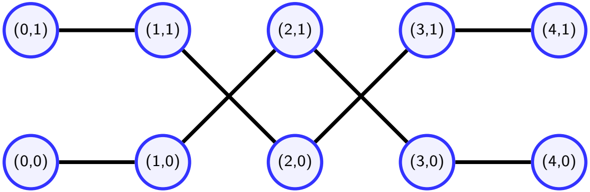
Lower bound's proof of quantum algorithms

Parity problem: Given $x_1, \ldots, x_n \in \{0, 1\}$, compute $x_1 \oplus \cdots \oplus x_n$, the quantum query complexity is $\Theta(n)$

We construct a weighted graph G:

- **vertices:** (i, t), where $i \in \{0, 1, \ldots, n\}, t \in \{0, 1\}$
- **edges:** an edge between $(i-1, t)$ and $(i, t \oplus x_i)$
- **weights:** to be determined

For example, $(x_1, x_2, x_3, x_4) = (0, 1, 1, 0)$, then G is



Lower bound's proof of quantum algorithms

Essentially, G consists of two paths

$$(0,0) - (1, x_1) - (2, x_1 \oplus x_2) - \cdots - (n, x_1 \oplus \cdots \oplus x_n)$$
$$(0,1) - (1, 1 \oplus x_1) - (2, 1 \oplus x_1 \oplus x_2) - \cdots - (n, 1 \oplus x_1 \oplus \cdots \oplus x_n)$$

Lower bound's proof of quantum algorithms

Essentially, G consists of two paths

$$(0,0) - (1, x_1) - (2, x_1 \oplus x_2) - \cdots - (n, x_1 \oplus \cdots \oplus x_n)$$
$$(0,1) - (1, 1 \oplus x_1) - (2, 1 \oplus x_1 \oplus x_2) - \cdots - (n, 1 \oplus x_1 \oplus \cdots \oplus x_n)$$

Let A be the adjacency matrix of G (essentially two symmetric tridiagonal matrices).

- **Case 1:** if $x_1 \oplus x_2 \oplus \cdots \oplus x_n = 0$, then $\langle 0,0|f(A)|n,1\rangle = 0$

Lower bound's proof of quantum algorithms

Essentially, G consists of two paths

$$(0,0) - (1, x_1) - (2, x_1 \oplus x_2) - \cdots - (n, x_1 \oplus \cdots \oplus x_n)$$
$$(0,1) - (1, 1 \oplus x_1) - (2, 1 \oplus x_1 \oplus x_2) - \cdots - (n, 1 \oplus x_1 \oplus \cdots \oplus x_n)$$

Let A be the adjacency matrix of G (essentially two symmetric tridiagonal matrices).

- **Case 1:** if $x_1 \oplus x_2 \oplus \cdots \oplus x_n = 0$, then $\langle 0,0|f(A)|n,1\rangle = 0$
- **Case 2:** if $x_1 \oplus x_2 \oplus \cdots \oplus x_n = 1$, then we hope to find appropriate weights such that $\langle 0,0|f(A)|n,1\rangle \geq \varepsilon$ for some ε. The weights are determined by our key theorem.

Lower bound's proof of quantum algorithms

Essentially, G consists of two paths

$$(0,0) - (1,x_1) - (2,x_1 \oplus x_2) - \cdots - (n,x_1 \oplus \cdots \oplus x_n)$$
$$(0,1) - (1,1 \oplus x_1) - (2,1 \oplus x_1 \oplus x_2) - \cdots - (n,1 \oplus x_1 \oplus \cdots \oplus x_n)$$

Let A be the adjacency matrix of G (essentially two symmetric tridiagonal matrices).

- **Case 1:** if $x_1 \oplus x_2 \oplus \cdots \oplus x_n = 0$, then $\langle 0,0|f(A)|n,1\rangle = 0$
- **Case 2:** if $x_1 \oplus x_2 \oplus \cdots \oplus x_n = 1$, then we hope to find appropriate weights such that $\langle 0,0|f(A)|n,1\rangle \geq \varepsilon$ for some ε. The weights are determined by our key theorem.

As a result, if we can estimate $\langle 0,0|f(A)|n,1\rangle$, we then can solve the parity problem, which is known hard.

Lower bound's proof of quantum algorithms

Essentially, G consists of two paths

$$(0,0) - (1, x_1) - (2, x_1 \oplus x_2) - \cdots - (n, x_1 \oplus \cdots \oplus x_n)$$
$$(0,1) - (1, 1 \oplus x_1) - (2, 1 \oplus x_1 \oplus x_2) - \cdots - (n, 1 \oplus x_1 \oplus \cdots \oplus x_n)$$

Let A be the adjacency matrix of G (essentially two symmetric tridiagonal matrices).

- **Case 1:** if $x_1 \oplus x_2 \oplus \cdots \oplus x_n = 0$, then $\langle 0,0|f(A)|n,1\rangle = 0$
- **Case 2:** if $x_1 \oplus x_2 \oplus \cdots \oplus x_n = 1$, then we hope to find appropriate weights such that $\langle 0,0|f(A)|n,1\rangle \geq \varepsilon$ for some ε. The weights are determined by our key theorem.

As a result, if we can estimate $\langle 0,0|f(A)|n,1\rangle$, we then can solve the parity problem, which is known hard. It is very important to ensure that $n = \widetilde{\deg}(f) + O(1)$ in the key theorem.

Lower bound's proof of quantum algorithms

Essentially, G consists of two paths

$$(0,0) - (1, x_1) - (2, x_1 \oplus x_2) - \cdots - (n, x_1 \oplus \cdots \oplus x_n)$$
$$(0,1) - (1, 1 \oplus x_1) - (2, 1 \oplus x_1 \oplus x_2) - \cdots - (n, 1 \oplus x_1 \oplus \cdots \oplus x_n)$$

Let A be the adjacency matrix of G (essentially two symmetric tridiagonal matrices).

- **Case 1:** if $x_1 \oplus x_2 \oplus \cdots \oplus x_n = 0$, then $\langle 0,0|f(A)|n,1\rangle = 0$
- **Case 2:** if $x_1 \oplus x_2 \oplus \cdots \oplus x_n = 1$, then we hope to find appropriate weights such that $\langle 0,0|f(A)|n,1\rangle \geq \varepsilon$ for some ε. The weights are determined by our key theorem.

As a result, if we can estimate $\langle 0,0|f(A)|n,1\rangle$, we then can solve the parity problem, which is known hard. It is very important to ensure that $n = \widetilde{\deg}(f) + O(1)$ in the key theorem. This idea is inspired by the no fast-forwarding theorem. [Berry, Ahokas, Cleve, Sanders, Comm. Math. Phys. '07]

Lower bound's proof of classical algorithms

Forrelation problem (Aaronson & Ambainis, 2015):

Given $g_1, g_2 : \{0,1\}^n \to \{\pm 1\}$, let $D_i = \text{diag}(g_i(x) : x \in \{0,1\}^n)$, $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, define

$$\begin{aligned} \Phi(g_1, g_2) &:= \langle 0^n | H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n} | 0^n \rangle \\ &= \frac{1}{2^{3n/2}} \sum_{x,y \in \{0,1\}^n} (-1)^{x \cdot y} g_1(x) g_2(y). \end{aligned}$$

The goal is to compute $\Phi(g_1, g_2) \pm 1/3$

Lower bound's proof of classical algorithms

Forrelation problem (Aaronson & Ambainis, 2015):

Given $g_1, g_2 : \{0,1\}^n \to \{\pm 1\}$, let $D_i = \text{diag}(g_i(x) : x \in \{0,1\}^n)$, $H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, define

$$\begin{aligned} \Phi(g_1, g_2) &:= \langle 0^n | H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n} | 0^n \rangle \\ &= \frac{1}{2^{3n/2}} \sum_{x,y \in \{0,1\}^n} (-1)^{x \cdot y} g_1(x) g_2(y). \end{aligned}$$

The goal is to compute $\Phi(g_1, g_2) \pm 1/3$

For this problem, the classical query complexity is lower bounded by $\Omega(\sqrt{2^n}/n)$, while the quantum query complexity is $O(1)$.

Feynman's clock construction

Let $U = U_{N-1} \cdots U_2 U_1$ be a unitary operator, define

$$A = \begin{pmatrix} 0 & b_1 U_1^\dagger & & & \\ b_1 U_1 & 0 & b_2 U_2^\dagger & & \\ & b_2 U_2 & \ddots & \ddots & \\ & & \ddots & \ddots & \end{pmatrix}$$

Feynman's clock construction

Let $U = U_{N-1} \cdots U_2 U_1$ be a unitary operator, define

$$A = \begin{pmatrix} 0 & b_1 U_1^\dagger & & & \\ b_1 U_1 & 0 & b_2 U_2^\dagger & & \\ & b_2 U_2 & \ddots & \ddots & \\ & & \ddots & \ddots & \end{pmatrix}$$

Let $|\psi_t\rangle := |t\rangle \otimes U_t \cdots U_1 |0\rangle$, then

$$A|\psi_t\rangle = b_{t-1}|\psi_{t-1}\rangle + b_{t+1}|\psi_{t+1}\rangle$$

In subspace $\{|\psi_t\rangle : t = 0, 1, \ldots, N-1\}$, A is a symmetric tridiagonal matrix.

Lower bound's proof of classical algorithms

In the Forrelation problem, we have $U = H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n}$. To ensure A is sparse in the clock construction, we decompose

$$H^{\otimes n} = (H \otimes I \otimes \cdots \otimes I)(I \otimes H \otimes \cdots \otimes I) \cdots (I \otimes I \otimes \cdots \otimes H)$$

Lower bound's proof of classical algorithms

In the Forrelation problem, we have $U = H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n}$. To ensure A is sparse in the clock construction, we decompose

$$H^{\otimes n} = (H \otimes I \otimes \cdots \otimes I)(I \otimes H \otimes \cdots \otimes I) \cdots (I \otimes I \otimes \cdots \otimes H)$$

Now $N = 3n + 2$,

$$\begin{aligned} |\psi_0\rangle &= |0\rangle \otimes |0\rangle \\ |\psi_{N-1}\rangle &= |N-1\rangle \otimes H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n} |0\rangle \end{aligned}$$

Lower bound's proof of classical algorithms

In the Forrelation problem, we have $U = H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n}$. To ensure A is sparse in the clock construction, we decompose

$$H^{\otimes n} = (H \otimes I \otimes \cdots \otimes I)(I \otimes H \otimes \cdots \otimes I) \cdots (I \otimes I \otimes \cdots \otimes H)$$

Now $N = 3n + 2$,

$$\begin{aligned} |\psi_0\rangle &= |0\rangle \otimes |0\rangle \\ |\psi_{N-1}\rangle &= |N-1\rangle \otimes H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n} |0\rangle \end{aligned}$$

Let $|\phi_{N-1}\rangle = |N-1\rangle \otimes |0\rangle$, then

$$\langle \phi_{N-1} | f(A) | \psi_0 \rangle = \langle \psi_{N-1} | f(A) | \psi_0 \rangle \cdot \Phi(g_1, g_2)$$

Lower bound's proof of classical algorithms

In the Forrelation problem, we have $U = H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n}$. To ensure A is sparse in the clock construction, we decompose

$$H^{\otimes n} = (H \otimes I \otimes \cdots \otimes I)(I \otimes H \otimes \cdots \otimes I) \cdots (I \otimes I \otimes \cdots \otimes H)$$

Now $N = 3n + 2$,

$$\begin{aligned} |\psi_0\rangle &= |0\rangle \otimes |0\rangle \\ |\psi_{N-1}\rangle &= |N-1\rangle \otimes H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n} |0\rangle \end{aligned}$$

Let $|\phi_{N-1}\rangle = |N-1\rangle \otimes |0\rangle$, then

$$\begin{array}{ccccc} \langle \phi_{N-1}|f(A)|\psi_0\rangle & = & \langle \psi_{N-1}|f(A)|\psi_0\rangle & \cdot & \Phi(g_1, g_2) \\ \downarrow & & \downarrow & & \downarrow \\ \text{an entry of } f(A) & & \text{Easy} & & \text{Hard} \end{array}$$

Lower bound's proof of classical algorithms

In the Forrelation problem, we have $U = H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n}$. To ensure A is sparse in the clock construction, we decompose

$$H^{\otimes n} = (H \otimes I \otimes \cdots \otimes I)(I \otimes H \otimes \cdots \otimes I) \cdots (I \otimes I \otimes \cdots \otimes H)$$

Now $N = 3n + 2$,

$$\begin{aligned} |\psi_0\rangle &= |0\rangle \otimes |0\rangle \\ |\psi_{N-1}\rangle &= |N-1\rangle \otimes H^{\otimes n} D_1 H^{\otimes n} D_2 H^{\otimes n} |0\rangle \end{aligned}$$

Let $|\phi_{N-1}\rangle = |N-1\rangle \otimes |0\rangle$, then

$$\begin{array}{ccccc} \langle \phi_{N-1} | f(A) | \psi_0 \rangle & = & \langle \psi_{N-1} | f(A) | \psi_0 \rangle & \cdot & \Phi(g_1, g_2) \\ \downarrow & & \downarrow & & \downarrow \\ \text{an entry of } f(A) & & \text{Easy} & & \text{Hard} \\ \downarrow & & & & \\ \text{Hard} & & & & \end{array}$$

Conclusion

- For functions of matrices, we proved

	Quantum algorithm	Classical algorithm
Upper bound	$O(sd/\varepsilon)$	$O(s^{d-1})$
Lower bound	$\Omega(d)$	$\Omega((s/2)^{(d-1)/6})$

- From the point of approximate degree,
  - quantum algorithm is optimal,
  - quantum-classical separation is exponential.

Conclusion

- For functions of matrices, we proved

	Quantum algorithm	Classical algorithm
Upper bound	$O(sd/\varepsilon)$	$O(s^{d-1})$
Lower bound	$\Omega(d)$	$\Omega((s/2)^{(d-1)/6})$

- From the point of approximate degree,
  - quantum algorithm is optimal,
  - quantum-classical separation is exponential.

Open questions:

- Lower bound in terms of ε in the quantum case? $\Omega(1/\varepsilon)$?
- Any applications similar to the famous polynomial method?

Conclusion

- For functions of matrices, we proved

	Quantum algorithm	Classical algorithm
Upper bound	$O(sd/\varepsilon)$	$O(s^{d-1})$
Lower bound	$\Omega(d)$	$\Omega((s/2)^{(d-1)/6})$

- From the point of approximate degree,
  - quantum algorithm is optimal,
  - quantum-classical separation is exponential.

Open questions:

- Lower bound in terms of ε in the quantum case? $\Omega(1/\varepsilon)$?
- Any applications similar to the famous polynomial method?

Thanks very much for your time!