0

# A Survey: Exponential algorithmic speedup by quantum walk

Yan-Tong Lin[1,*]

[1]*Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan*

(11 Jan 2021)

The authors construct an oracular problem (based on a random graph) that provide the first exponential quantum-classical algorithmic separation based on quantum walks. They show how to implement the continuous time quantum walk, provide physical intuition, and prove the time upper bound. Finally, the main contribution of the paper is the proof that no classical algorithm can solve the problem with high probability in subexponential time. In this survey, I will try to present the core values of the 24-page paper and sketch the proofs so that they are easier to understand.

## I. PRELIMINARIES

To understand this paper, one is expected to have some basic knowledge in the following fields.

- linear algebra
- calculus
- complexity theory
- graph theory
- quantum postulates

## II. INTRODUCTION AND PROBLEM DEFINITION

Simon's algorithm [1] provides an exponential separation between $\mathrm{BQP}^A$ and $\mathrm{BPP}^A$ with an oracle $A$ and it uses quantum Fourier transformation as its backbone. A previous work [2] shows an exponential gap between random walk and its quantum counterpart — quantum walk with graph $G_n$ (Figure 1). This paper, at that time, was the first to provide exponential separation between $\mathrm{BQP}^A$ and $\mathrm{BPP}^A$ with an oracle $A$ corresponding to graph $G'_n$ (Figure 2), using quantum walk. [1]

We define the problem as follow — given an oracle for the graph and the name of the ENTRANCE, find the name of the EXIT. The construction of the graph is shown by examples in Figure 1 and Figure 2. Despite that classical random walk requires steps exponential in $n$ to find the EXIT with high probability, one can easily construct a classical algorithm to solve the traversal problem using the fact that nodes in the middle are with degree 2.

For the rest of this survey, I will introduce quantum walk and show its efficiency in Section III and demonstrate the classical lower bound in Section III. Some details of the proofs will be skipped and you may find them in the original paper [4].

*0312fs3@gmail.com

[1] Please note that the relation of complexity classes relative to oracles provides intuitions and insights to the relation of original classes, but it cannot be used to show the relation itself.[3]

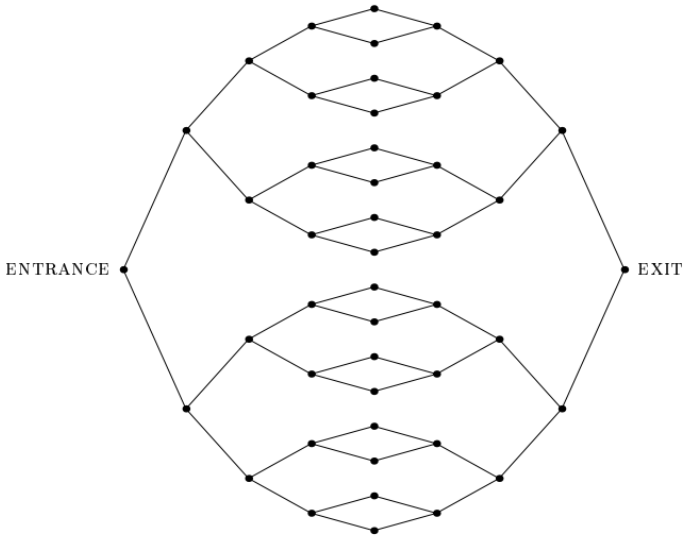


FIG. 1: The graph $G_4$.

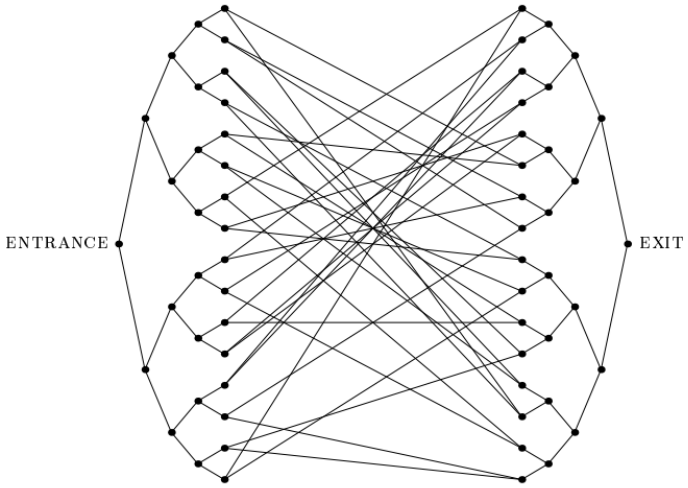


FIG. 2: A typical graph $G'_4$.

## III. QUANTUM ALGORITHM

### A. Quantum walk

#### *1. Classical random walk*

To know what is a quantum walk, one may better be familiar with its classical counter part.

In a continuous time classical random walk (which is also known as a Markov process), there is a fixed probability per unit time $\gamma$ of moving to an adjacent vertex. In other words, from any vertex, the probability of jumping to any connected vertex in a time $\epsilon$ is $\gamma\epsilon$ (in the limit $\epsilon \to 0$). If the graph has $N$ vertices, the random walk can be described by the $N \times N$ infinitesimal generator

matrix $K$ defined by

$$K_{aa'} = \begin{cases} \gamma & a \neq a',\ aa' \in G \\ 0 & a \neq a',\ aa' \notin G \\ -d(a)\gamma & a = a'. \end{cases} \tag{1}$$

If $p_a(t)$ is the probability of being at vertex $a$ at time $t$, then

$$\frac{\mathrm{d}p_a(t)}{\mathrm{d}t} = \sum_{a'} K_{aa'}\, p_{a'}(t)\,. \tag{2}$$

Note that because the columns of $K$ sum to zero, an initially normalized distribution remains normalized, i.e., $\sum_a p_a(t) = 1$ for all $t$.

#### *2. Quantum Analog*

Now consider quantum evolution in an $N$-dimensional Hilbert space spanned by states $|1\rangle, |2\rangle, \ldots, |N\rangle$ corresponding to the vertices of $G$. If the Hamiltonian is $H$, then the dynamics of the system are determined by the Schrödinger equation,

$$i\frac{\mathrm{d}}{\mathrm{d}t}\langle a|\psi(t)\rangle = \sum_{a'}\langle a|H|a'\rangle\langle a'|\psi(t)\rangle\,. \tag{3}$$

Its natural to consider a quantum evolution corresponding to a Hamiltonian equals to s the adjacency matrix of the graph times $\gamma$.

$$\langle a|H|a'\rangle = \begin{cases} \gamma & a \neq a',\ aa' \in G \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Because $H$ is Hermitian, probability is conserved, i.e., $\sum_a |\langle a|\psi(t)\rangle|^2 = 1$ for all $t$.

This will be the quantum walk we uses in this paper. [2] Now our goal is to simulate the unitary evolution $e^{-iHt}$ with $H$ given by (4).

### B. Implementing the quantum walk

#### *1. The oracle*

To show how to implement the quantum walk, we start by defining oracles corresponding to graphs rigorously.

To make an oracle, we require that the graph comes with additional structures so that the oracle is unitary. [3] In this paper we take consistent coloring — an edge coloring that no vertex is incident with two edges of the same color — as the additional structure. The coloring will be given in the Section IV and we will show that the coloring can be made up by classical algorithms so it cannot provide more information to solve the problem.

Assuming a consistent $k$-coloring of the graph is given. Let $n = \lceil \log N \rceil$ so it takes $n$ bits to list the vertices of the graph $G$. In the classical setting, the black box takes two inputs, a name $a$

[2] Please note that there can be many realizations of "quantum walk", as long as they make sense.
[3] It remains an open problem to construct efficient oracles for general graphs

given as a $2n$-bit [4] string and a color $c$. If the input name $a$ corresponds to a vertex that is incident with an edge of color $c$, then the output is the name of the vertex joined by that edge, otherwise the output is the special bit string $11\ldots1$, which is not the name of any vertex. For shorthand, we write $v_c(a) = a'$, where $a$ is the input name, $c$ is the color, and $a'$ is the output name. Note that $v_c(v_c(a)) = a$ for $v_c(a) \neq 11\ldots1$.

The unitary quantum oracle corresponding to this classical black box is described as follows. Let $a, b$ be $2n$-bit strings and let $c$ be a color. Then the action of the quantum black box $U$ associated with the graph $G$ is

$$U|a, b, c\rangle = |a, b \oplus v_c(a), c\rangle \tag{5}$$

Since we will never need to query $U$ with superposition of colors, it is sufficient to omit the color register and assume that we have access to the $k$ unitaries

$$U_c|a, b\rangle = |a, b \oplus v_c(a)\rangle\,, \tag{6}$$

one for each of the $k$ possible colors. In fact, since we want to check whether $v_c(a) = 11\ldots1$, it is also straightforward to extend the Hilbert space by a single qubit and perform each of the $k$ transformations

$$V_c|a, b, r\rangle = |a, b \oplus v_c(a), r \oplus f_c(a)\rangle\,, \tag{7}$$

where

$$f_c(a) = \begin{cases} 0 & v_c(a) \neq 11\ldots1 \\ 1 & v_c(a) = 11\ldots1\,. \end{cases} \tag{8}$$

The Hilbert space that $V_c$ acts on in (7) has dimension $2^{4n+1}$, which is much larger than the number of vertices in the graph. We want the quantum evolution $e^{-iHt}$ to take place in an $N$-dimensional subspace. To this end, we will show how to implement the quantum walk given the oracles $V_c$ in the sense that we will construct a Hamiltonian $H$ satisfying

$$H|a, 0, 0\rangle = \sum_{c:\, v_c(a)\in G} |v_c(a), 0, 0\rangle\,. \tag{9}$$

This means that if we start our evolution in the subspace of states of the form $|a, 0, 0\rangle$ where $a \in G$, then we will remain in this $N$-dimensional subspace.

### *2. Tools for simulating Hamiltonians*

In general, it is not easy to answer the question of whether a particular Hamiltonian can be simulated efficiently. However, there are some useful standard tools for simulating Hamiltonians. The following is a list of five such tools, four of which will be used in our construction. [5]

1. *Local terms.*
2. *Linear combination.*
3. *Commutation.*
4. *Unitary conjugation.*
5. *Tensor product.*

[4] we use $2n$ so that there is exponentially more names than vertices

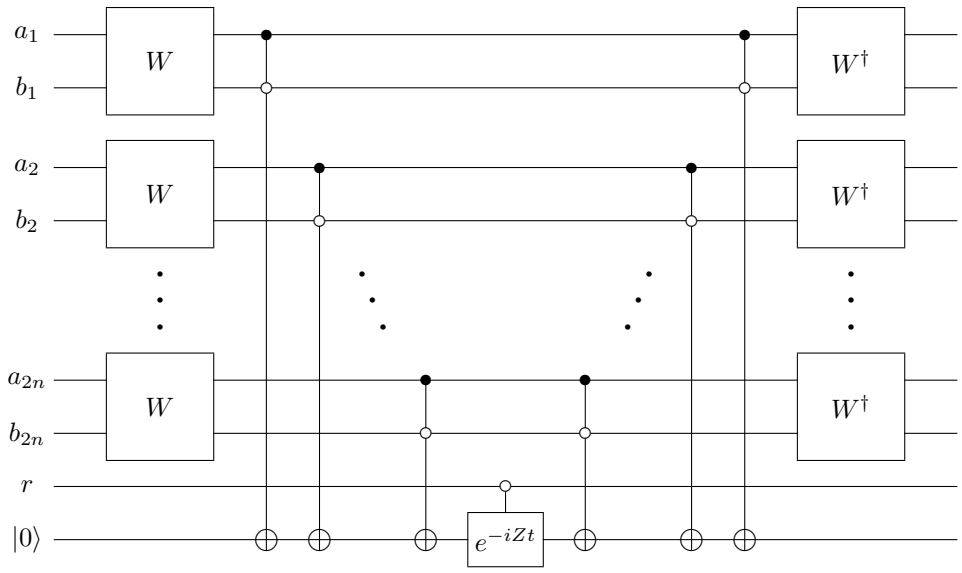


FIG. 3: A circuit for simulating $e^{-iTt}$.

### *3. T operator for swapping or deleting information*

We want to turn $V_c$ into $H$, so we define a Hermitian $T$ as follow,

$$T|a,b,0\rangle = |b,a,0\rangle \tag{10}$$
$$T|a,b,1\rangle = 0\,. \tag{11}$$

The operator $T$ may be written as

$$T = \left(\bigotimes_{l=1}^{2n} S^{(l,2n+l)}\right) \otimes |0\rangle\langle 0| \tag{12}$$

where the superscript indicates which two qubits $S$ acts on, and the projector onto $|0\rangle$ acts on the third register. Here $S$ is a Hermitian operator on two qubits satisfying $S|z_1 z_2\rangle = |z_2 z_1\rangle$.

Since the eigenvalues of $S$ are $\pm 1$, the eigenvalues of $T$ are $0, \pm 1$, and they are easy to compute. Thus $e^{-iTt}$ can be simulated with the circuit shown in Figure 3. In this figure, $W$ denotes a two-qubit unitary operator that diagonalizes $S$. The unique eigenvector of $S$ with eigenvalue $-1$ is $\frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$, so we take

$$W|00\rangle = |00\rangle \tag{13}$$
$$W\tfrac{1}{\sqrt{2}}(|01\rangle + |10\rangle) = |01\rangle \tag{14}$$
$$W\tfrac{1}{\sqrt{2}}(|01\rangle - |10\rangle) = |10\rangle \tag{15}$$
$$W|11\rangle = |11\rangle\,. \tag{16}$$

Applying $W^{\otimes 2n}$ diagonalizes $T$, and the Toffoli gates compute the argument of the eigenvalue in an ancilla register initially prepared in the state $|0\rangle$. After that, we apply the appropriate phase shift ($Z$ or $I$) by parity of the ancillary bit indicating the eigenvalue. Finally, we uncompute the ancillary bit and return to the original basis.

The targeted Hamiltonian $H$ equals to $\sum_c V_c^\dagger T V_c$ [5], which can be simulated using unitary conjugation and linear combination. [6]

### C. Physical intuition: propagation on a line

Before the actual proof, the authors first use standard physics techniques to give an intuition that the quantum walk propagates from the ENTRANCE to the EXIT in linear time. [7] They view the original problem as a walk on a finite line with a defect at the center, and show through examples that the defect and the boundaries do not significantly affect the walk. For this survey, I will only show the simplest case —- a walk on a infinite line without a defect.

#### *1. Column space*

The analysis of the walk on $G'_n$ is particularly simple because it can be reduced to a walk on a line with $2n+2$ vertices, one for each column of the original graph. Consider the $(2n+2)$-dimensional subspace spanned by the states

$$|\operatorname{col} j\rangle = \frac{1}{\sqrt{N_j}} \sum_{a \in \text{column } j} |a\rangle\,, \tag{17}$$

where

$$N_j = \begin{cases} 2^j & 0 \le j \le n \\ 2^{2n+1-j} & n+1 \le j \le 2n+1\,. \end{cases} \tag{18}$$

We refer to this subspace as the *column subspace*. It is obvious that applying $H$ to any state in the column subspace results in another state in this subspace, i.e. the column subspace is invariant under $H$. Thus, to understand the quantum walk starting from the ENTRANCE, we only need to understand how the Hamiltonian acts on the column subspace.

In this subspace, the non-zero matrix elements of $H$ are

$$\langle \operatorname{col} j|H|\operatorname{col}(j+1)\rangle = \begin{cases} 1 & 0 \le j \le n-1\,, \quad n+1 \le j \le 2n \\ \sqrt{2} & j = n \end{cases} \tag{19}$$

(and those deduced by Hermiticity of $H$), by setting $\gamma = 1/\sqrt{2}$ for simplicity.

#### *2. Propagator for the simplest case*

Consider a walk on a infinite line without a defect. The nonzero matrix elements of the Hamiltonian are

$$\langle j|H|j \pm 1\rangle = 1\,, \quad -\infty < j < \infty\,. \tag{20}$$

---

[5] The proof us relatively straight-forward and can be found in the original paper [4]

[6] $V_c^\dagger = V_c$

[7] I do not think this is worthy of a subsection. If you are not familiar with quantum physics, you may want to skip this part except for the introduction of the column space.

The eigenstates of this Hamiltonian are the momentum eigenstates $|p\rangle$ with components

$$\langle j|p\rangle = \frac{1}{\sqrt{2\pi}} e^{ipj}\,, \quad -\pi \le p \le \pi \tag{21}$$

having energies

$$E_p = 2\cos p\,. \tag{22}$$

The propagator function [8] to go from $j$ to $k$ in time $t$ is [9]

$$\begin{aligned} G(j,k,t) &= \langle k|e^{-iHt}|j\rangle && (23)\\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \mathrm{d}p\, e^{ip(k-j)-2it\cos p} && (24)\\ &= (-i)^{k-j} J_{k-j}(2t)\,, \quad -\infty < j,k < \infty && (25) \end{aligned}$$

where $J_\nu(\cdot)$ is a Bessel function of order $\nu$. By the well-known properties of the Bessel function, this shows that a state initially localized in a single column evolves as a left moving and a right moving wave packet, each propagating with speed 2. To see this, note that the Bessel function has the following asymptotic expansions for $\nu \gg 1$:

$$\begin{aligned} J_\nu(\nu\,\mathrm{sech}\,\xi) &\sim \frac{e^{-\nu(\xi-\tanh\xi)}}{\sqrt{2\pi\nu\tanh\xi}} && (26)\\ J_\nu(\nu+\xi\nu^{1/3}) &= (2/\nu)^{1/3}\,\mathrm{Ai}(-2^{1/3}\xi) + O(\nu^{-1}) && (27)\\ J_\nu(\nu\sec\xi) &= \sqrt{\frac{2}{\pi\nu\tan\xi}}\left\{\cos[\tfrac{\pi}{4} - \nu(\xi-\tan\xi)] + O(\nu^{-1})\right\}\,, \quad 0<\xi<\frac{\pi}{2} && (28) \end{aligned}$$

where $\mathrm{Ai}(\cdot)$ is an Airy function. These three relations show that for $|k-j| \gg 1$, $G(j,k,t)$ is exponentially small in $|k-j|$ for $t < 0.99\cdot|k-j|/2$, of order $|k-j|^{-1/3}$ for $t$ near $|k-j|/2$, and of order $|k-j|^{-1/2}$ for $t > 1.01\cdot|k-j|/2$. This gives that if the quantum state at $t=0$ is $|\,\mathrm{col}\,0\rangle$, then at a time of order $n/2$, there is an appreciable probability of being at $|\,\mathrm{col}(2n+1)\rangle$.

### D. Upper bound on the hitting time

Here comes the actual proof for the efficiency of the quantum algorithm. Since the proof is long, I will provide the sketch and describe the techniques used in the proof. For the details of the calculation, please turn to the original paper [4].

**Theorem 1** *For $n$ sufficiently large, running the quantum walk for a time chosen uniformly in $[0, \frac{n^4}{2\epsilon}]$ and then measuring in the computational basis yields a probability of finding the* EXIT *that is greater than $\frac{1}{2n}(1-\epsilon)$.*

**Proof sketch**

1. Consider the Hermitian $H$ on the column space
2. Bound the probability of success by the smallest spectral gap of $H$ (Lemma 2)

[8] https://en.wikipedia.org/wiki/Propagator
[9] Note: here use $|j\rangle = \int_p |p\rangle\langle p|j\rangle$

3. Show the smallest spectral gap is polynomially small (Lemma 3)

The Hermitian $H$ on the column space

$$\langle \mathrm{col}\, j|H|\, \mathrm{col}(j+1)\rangle = \begin{cases} 1 & 1 \le j \le n-1\,, \quad n+1 \le j \le 2n-1 \\ \sqrt{2} & j = n\,, \end{cases} \tag{29}$$

■

**Lemma 2** *Consider the quantum walk in $G'_{n-1}$ starting at the* ENTRANCE. *Let the walk run for a time $t$ chosen uniformly in $[0,\tau]$ and then measure in the computational basis. If $\tau \ge \frac{4n}{\epsilon\,\Delta E}$ for any constant $\epsilon > 0$, where $\Delta E$ is the magnitude of the smallest gap between any pair of eigenvalues of the Hamiltonian, then the probability of finding the* EXIT *is greater than $\frac{1}{2n}(1-\epsilon)$.*

**Proof sketch**

1. Write down the probability $\frac{1}{\tau}\int_0^\tau \mathrm{d}t\, |\langle \mathrm{col}\, 2n|e^{-iHt}|\, \mathrm{col}\, 1\rangle|^2$
2. Find simultaneous eigenstates of a reflection operator $R$ and $H$
3. Evaluate with the simultaneous eigenstates of $R$ and $H$

The reflector $R$ [10]

$$R|\, \mathrm{col}\, j\rangle = |\, \mathrm{col}(2n+1-j)\rangle\,. \tag{30}$$

$R$ commutes with $H$ on the column subspace, so we can find simultaneous eigenstates of $R$ and $H$.

$$\langle \mathrm{col}\, j|E\rangle = \begin{cases} \sin pj & 1 \le j \le n \\ \pm\sin(p(2n+1-j)) & n+1 \le j \le 2n\,, \end{cases} \tag{31}$$

The eigenvalue (to $H$) corresponding to the eigenstate $|E\rangle$ is $E = 2\cos p$, and the quantization condition comes from matching at vertices $n$ and $n+1$.

During step 3 of the proof, use Cauchy-Schwartz inequality and the fact that $\langle E|\, \mathrm{col}\, 1\rangle = \pm\langle E|\, \mathrm{col}\, 2n\rangle$.

■

**Lemma 3** *The smallest gap between any pair of eigenvalues of the Hamiltonian satisfies*

$$\Delta E > \frac{2\pi^2}{(1+\sqrt{2})n^3} + O(1/n^4)\,. \tag{32}$$

**Proof sketch**

1. Write down the quantization condition of eigenstates given by the defect.
2. Find the plotting to gain intuition of the solutions (Figure III D)
3. Calculate distance to $\frac{l\pi}{n}$ to bound $\Delta p$.
4. use $\Delta p$ to bind $\Delta E$

The constraint $\langle \mathrm{col}\, n|H|E\rangle = 2\cos p\, \langle \mathrm{col}\, n|E\rangle$ by definition can be simplified to

$$\frac{\sin((n+1)p)}{\sin np} = \pm\sqrt{2}\,. \tag{33}$$

■

[10] Note that $R^2 = 1$, so $R$ has eigenvalues $\pm 1$.

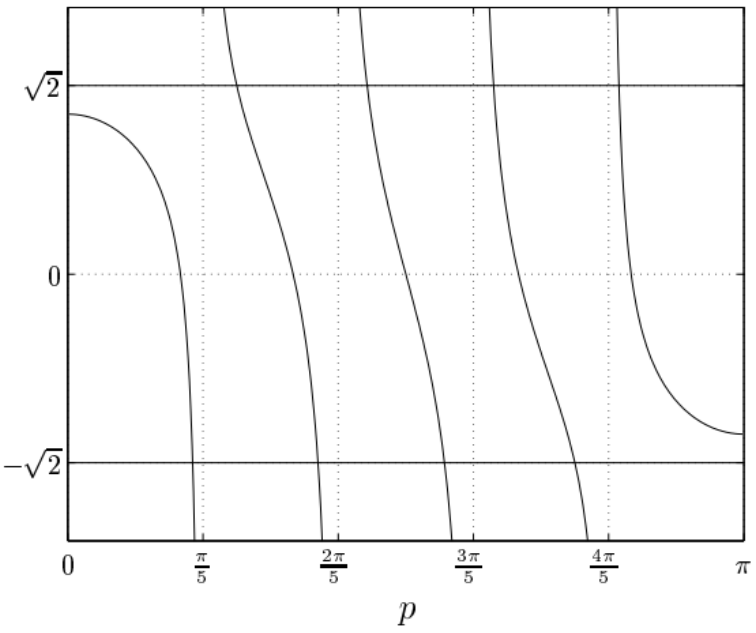


FIG. 4: Left hand side of (III D) for $n = 5$.

### E. An efficient quantum algorithm for traversing $G'_n$

To summarize, with theorem 1, we an efficient quantum algorithm for traversing $G'_n$. The computer is prepared in the state corresponding to the ENTRANCE, and the quantum walk is simulated using the construction described in Section III B. After running the walk for an appropriate $t = \text{poly}(n)$, the state of the computer is measured. The probability of finding the name of the EXIT can be $O(1/n)$. By repeating this process poly$(n)$ times, the success probability can be made arbitrarily close to 1.

## IV. CLASSICAL LOWER BOUND

Now we want to proof that no classical algorithm can find the EXIT with high probability in subexponential time. The authors achieve this by a sequence of games that the first game is equivelant to the original traversal problem, and each subsequent game is essentially as easy to win. Finally, they will show that the easiest game cannot be won in subexponential time.

### A. Turn the original problem to a game

**Game 1** *The oracle contains a random set of names for the vertices of the randomly chosen graph $G'_n$ such that each vertex has a distinct $2n$-bit string as its name and the* ENTRANCE *vertex has the name* 0*. In addition, the edges of the graph are randomly colored as described above. At each step, the algorithm sends a $2n$-bit string to the oracle, and if there exists a vertex with that name, the oracle returns the names of the neighbors of that vertex and the colors of the edges that join them. The algorithm wins if it ever sends the oracle the name of the* EXIT *vertex.*

### B. Restriction to known vertices

**Game 2** *The oracle contains a graph, set of vertex names, and edge coloring as described in Game 1. At each step, the algorithm sends the oracle the name of the* ENTRANCE *vertex or the name of a vertex it has previously been sent by the oracle. The oracle then returns the names of the neighbors of that vertex and the colors of the edges that join them. The algorithm wins it ever sends the oracle the name of the* EXIT *vertex.*

The win rate of the game is lower, but bounded with exponentially small term. This is because the chance that the original game algorithm can discover the name of a vertex that it is not told by the oracle is at most $t(2^{n+2}-2)/2^{2n}$, and unless this happens, the algorithms will have similar behavior.

### C. Removal of coloring

The authors make a consistent coloring that does not provide information for any classical quantum algorithm. At each vertex in an even numbered column, randomly color the incident edges $A, B, C$. At each vertex in an odd numbered column, randomly append $1, 2, 3$ to the colors of the incident edges. The edges are then colored with one of nine colors, $A1, A2, A3, B1, \ldots, C3$. Because of the structure of $G'_n$, any such coloring is consistent. Also, since the parity of columns is the only information required for a classical algorithm to make up the color of an edge, and this information is always available to an algorithm that can only traverse a connected subgraph, the coloring cannot provide more information to the algorithms.

**Game 3** *The oracle contains a graph and a set of vertex names as described in Game 1. At each step, the algorithm sends the oracle the name of the* ENTRANCE *vertex or the name of a vertex it has previously been sent by the oracle. The oracle then returns the names of the neighbors of that vertex. The algorithm wins it ever sends the oracle the name of the* EXIT *vertex.*

### D. Adding a new win condition — Finding a Cycle

**Game 4** *The oracle contains a graph and a set of vertex names as described in Game 1. At each step, the algorithm and the oracle interact as in Game 3. The algorithm wins it ever sends the oracle the name of the* EXIT *vertex, or if the subgraph it has seen contains a cycle.*

Game 4 is clearly easier to win than Game 3. And by this, we restrict the subgraph an algorithm sees must be a rooted binary tree

### E. Reformulation as Tree Embedding

For a rooted binary tree $T$, we define an embedding of $T$ into $G$ to be a function $\pi$ from the vertices of $T$ to the vertices of $G$ such that $\pi(\text{ROOT}) = \text{ENTRANCE}$ and for all vertices $u$ and $v$ that are neighbors in $T$, $\pi(u)$ and $\pi(v)$ are neighbors in $G$. We say that an embedding of $T$ is *proper* if $\pi(u) \neq \pi(v)$ for $u \neq v$. We say that a tree $T$ *exits* under an embedding $\pi$ if $\pi(v) = \text{EXIT}$ for some $v \in T$.

A random embedding of a tree is obtained by setting $\pi(\text{ROOT}) = \text{ENTRANCE}$ and then mapping the rest of $T$ into $G$ at random recursively.

**Game 5** *The algorithm outputs a rooted binary tree $T$ with $t$ vertices in which each internal vertex has two children. A random embedding $\pi$ is chosen. The algorithm wins if $\pi$ is an improper embedding of $T$ in $G'_n$ or if $T$ exits $G'_n$ under $\pi$.*

**Lemma 4** *For any algorithm $A$ for Game 4 that uses at most $t$ queries of the oracle, there exists an algorithm $A'$ for Game 5 that outputs a tree of at most $t$ vertices such that for all graphs $G$,*

$$\mathbb{P}_4^G(A) = \mathbb{P}_5^G(A')\,. \tag{34}$$

**Proof** Algorithm $A$ halts if it ever finds a cycle, exits, or uses $t$ steps. Algorithm $A'$ will generate a (random) tree by simulating $A$. Suppose that vertex $a$ in graph $G$ corresponds to vertex $a'$ in the tree that $A'$ is generating. If $A$ asks the oracle for the names of the neighbors of $a$, $A'$ generates two unused names $b'$ and $c'$ at random and uses them as the neighbors of $a'$. Now $b'$ and $c'$ correspond to $b$ and $c$, the neighbors of $a$ in $G$. Using the tree generated by $A'$ in Game 5 has the same behavior as using $A$ in Game 4. ■

Finally, we bound the probability that an algorithm wins Game 5:

**Lemma 5** *For rooted trees $T$ of at most $2^{n/6}$ vertices,*

$$\max_T \mathop{\mathrm{E}}_G \left[\mathbb{P}^G(T)\right] \leq 3 \cdot 2^{-n/6}. \tag{35}$$

**Proof sketch**

1. Define $\frac{n}{2}$-subtrees
2. Show it is hard to reach roots of $\frac{n}{2}$-subtrees on the right side
3. Show it is hard to form a cycle by considering the lists of subtrees passed corresponding to the paths in the tree $T$ that forms a cycle.

Part 3 is the core of this paper, one can find the full statement in the original paper [4]. It is the place that the difference between $G_n$ and $G'_n$ is used.
■

With the above results, we can have the final theorem.

**Theorem 6** *Any classical algorithm that makes at most $2^{n/6}$ queries to the oracle finds the* EXIT *with probability at most $4 \cdot 2^{-n/6}$.*

## V. CONCLUSION

- The authors show that $\mathrm{BQP}^A \neq \mathrm{BPP}^A$ for the oracle $A = U$ corresponding to graph $G'_n$.
- The separation is exponential and is not achieved by Fourier based method but with pure quantum inference effect.

## VI. DISCUSSION AND OPEN PROBLEMS

- The results can also be cast in terms of a graph reachability problem (a decision problem).
- The relation of complexity classes relative to oracles provides intuitions and insights to the relation of original classes, but it cannot be used to show the relation itself. (Relativization [3]) So it remains a challenge to determine BQP $\stackrel{?}{=}$ BPP.
- Can one find more other techniques for implementing quantum walks?
- Are there interesting problems that are considered classically hard can be solve efficiently with quantum walks?
  - To my knowledge, there is a machine-learning work aiming to predict speedup by quantum walk[6].
  - Can we give explicit characteristics (sufficient and necessary conditions)?

---


[1] D. Simon, *On the power of quantum computation*, Proc. 35th IEEE Symposium on the Foundations of Computer Science, 116 (1994).

[2] E. Farhi and S. Gutmann, *Quantum computation and decision trees*, Phys. Rev. A **58**, 915 (1998).

[3] Dana Moshkovitz, *Advanced Complexity Theory Lecture 2: Relativization*, https://ocw.mit.edu/courses/mathematics/18-405j-advanced-complexity-theory-spring-2016/lecture-notes/MIT18/_405JS16_Relativ.pdf.

[4] Childs, Andrew M., et al, *Exponential algorithmic speedup by a quantum walk*, Proceedings of the thirty-fifth annual ACM symposium on Theory of computing. 2003.

[5] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, (Cambridge University Press, Cambridge, 2000).

[6] Melnikov, Alexey A., Leonid E. Fedichkin, and Alexander Alodjants, *Predicting quantum advantage by quantum walk with convolutional neural networks*, New Journal of Physics 21.12 (2019): 125002.