# Quantum Query Complexity of Boolean Functions with Small On-Sets

Andris Ambainis[*] Kazuo Iwama[†] Masaki Nakanishi[‡] Harumichi Nishimura[§]
Rudy Raymond[¶] Seiichiro Tani[‖**] Shigeru Yamashita[††]

**Abstract**

The main objective of this paper is to show that the quantum query complexity $Q(f)$ of an $N$-bit Boolean function $f$ is bounded by a function of a simple and natural parameter, i.e., $M = |\{x \mid f(x) = 1\}|$ or the size of $f$'s *on-set*. We prove that: (i) For $poly(N) \leq M \leq 2^{N^d}$ for some constant $0 < d < 1$, the upper bound of $Q(f)$ is $O(\sqrt{N \log M / \log N})$. This bound is tight, namely there is a Boolean function $f$ such that $Q(f) = \Omega(\sqrt{N \log M / \log N})$. (ii) For the same range of $M$, the (also tight) lower bound of $Q(f)$ is $\Omega(\sqrt{N})$. (iii) The average value of $Q(f)$ is bounded from above and below by $Q(f) = O(\log M + \sqrt{N})$ and $Q(f) = \Omega(\log M / \log N + \sqrt{N})$, respectively. The first bound gives a simple way of bounding the quantum query complexity of testing some graph properties. In particular, it is proved that the quantum query complexity of planarity testing for a graph with $n$ vertices is $\Theta(N^{3/4})$ where $N = \frac{n(n-1)}{2}$.

## 1 Introduction

Query complexities for Boolean functions are one of the most fundamental and popular topics in quantum computation. It is well known that a quadratic speed-up, i.e., $\Omega(N)$ classically to $O(\sqrt{N})$ quantumly, is possible for several Boolean functions including OR, AND, AND-OR trees [18, 19, 17, 7]. On the other hand, we can obtain only a constant-factor speed-up (i.e., $\Omega(N)$ are needed both classically and quantumly) for other Boolean functions such as PARITY [10], and this is also the case for almost all Boolean functions with $N$ variables [4, 13]. Thus our knowledge about the quantum query complexity for Boolean functions is relatively good for these typical cases, but much less is known for the others, especially for quantitative properties based on nontrivial parameters. In this paper, we show that the *size of the on-set* of a Boolean function $f$ plays a key role for this purpose, i.e., to non-trivially bound the $f$'s quantum query complexity.

Obviously this line of research started with the Grover's quantum search algorithm [18], which can be directly used to compute the Boolean OR function of $N$ variables in the same $O(\sqrt{N})$ queries. Since then, a sequence of results have extensively appeared in the literature, showing that similar speed-ups are possible for many other, more general Boolean functions. For example, if a Boolean function is given by a constant-depth balanced AND-OR trees (OR is by a single-depth

[*] Institute of Mathematics and Computer Science, University of Latvia, Latvia. `ambainis@lu.lv`.

[†] School of Informatics, Kyoto University. Kyoto, Japan. `iwama@kuis.kyoto-u.ac.jp`.

[‡] Graduate School of Information Science, NAIST, Nara, Japan. `m-naka@is.naist.ac.jp`

[§] School of Science, Osaka Prefecture University. Osaka, Japan. `hnishimura@mi.s.osakafu-u.ac.jp`.

[¶] Tokyo Research Laboratory, IBM Japan. Kanagawa, Japan. `raymond@jp.ibm.com`.

[‖] NTT Communication Science Laboratories, NTT Corporation. Atsugi, Japan. `tani@theory.brl.ntt.co.jp`.

[**] Quantum Computation and Information Project, SORST, JST, Tokyo, Japan.

[††] Graduate School of Information Science, NAIST, Nara, Japan. `ger@is.naist.ac.jp`

tree), it can be computed in $O(\sqrt{N})$ queries [19]. This was recently extended to any AND-OR tree with $O(N^{\frac{1}{2}+o(1)})$ queries by using the quantum walk technique [7]. Another famous example is monotone Boolean functions describing monotone graph properties (if a graph has $n$ vertices then the corresponding Boolean function has $N = n(n-1)/2$ variables, one for each possible edge). In the quantum setting, it is known that $O(N^{13/20})$ queries suffice to decide if the given graph $G$ includes a triangle [21, 22], $O(N^{3/4})$ queries if $G$ includes a star [11] (with zero error), and $O(N^{3/4})$ queries if $G$ is connected [16]. Classical query complexities for those functions are all $\Omega(N)$.

Note that each of these Boolean functions, for which quantum query complexities are significantly smaller than classical query complexities, has a certain kind of "structure". In other words, researchers have been working on the question of what kind of structures help for efficient quantum computation. Our question in this paper is quite different; namely we ask if there is a "non-structural" parameter that greatly affects the quantum complexity of Boolean functions.

**Our contribution:** Let $\mathcal{F}_M$ be a family of $N$-variable Boolean functions $f$ whose on-set is of size $M$, namely, $f$ has output 1 (true) for $M$ 0/1 assignments among the total $2^N$ ones. Then we can show that for *any* Boolean function $f$ in $\mathcal{F}_{poly(N)}$, its query complexity is $\Theta(\sqrt{N})$ and the complexity gradually increases as $M$ grows up to $2^{N^d}$ for some constant $d$ ($0 < d < 1$). More in detail, let $Q(f)$ be the (true) query complexity of $f$. Then we investigate the upper bound $C(\mathcal{F}_M)$, the lower bound $c(\mathcal{F}_M)$, and the average value, $\tilde{C}(\mathcal{F}_M)$, of $Q(f)$ over all functions $f$ in $\mathcal{F}_M$. Our results are as follows: (i) For $poly(N) \leq M \leq 2^{N^d}$ for some constant $0 < d < 1$, $C(\mathcal{F}_M) = \Theta(\sqrt{N \log M / \log N})$. This means that for any function in $\mathcal{F}_M$, its query complexity is $O(\sqrt{N \log M / \log N})$ and there exists a function in $\mathcal{F}_M$ such that its complexity is $\Omega(\sqrt{N \log M / \log N})$. (ii) For the same range of $M$, $c(\mathcal{F}_M) = \Theta(\sqrt{N})$, meaning that for any function in $\mathcal{F}_M$ its complexity is $\Omega(\sqrt{N})$ and there exists a function such that its complexity is $O(\sqrt{N})$. Thus our results are tight for both $C(\mathcal{F}_M)$ and $c(\mathcal{F}_M)$. Unfortunately, there is a $\log N$ factor gap in the evaluation of $\tilde{C}(\mathcal{F}_M)$, namely (iii) $\tilde{C}(\mathcal{F}_M) = O(\log M + \sqrt{N})$ and $\tilde{C}(\mathcal{F}_M) = \Omega(\log M / \log N + \sqrt{N})$.

A direct application of our upper bound result reduces bounding the query complexity of graph property testing to counting all graphs with a given property: For the family $\mathcal{F}$ of all graphs with a given property, $O(\sqrt{N \log |\mathcal{F}| / \log N})$ queries suffice to test if a given graph has the property. An interesting special case is that $O(N^{3/4})$ queries can decide if a given graph $G$ is isomorphic to an arbitrary fixed graph $G'$. The bound is optimal in the worst case over all $G'$. Another interesting case is to test planarity, one of the most fundamental graph properties. We show the tight bound of $\Theta(N^{3/4})$ for the quantum query complexity of planarity testing. The upper bound is by just bounding the number of planar graphs with the fact that they are sparse. The interesting part is its lower bound. Our proof is based on the quantum adversary method [5], which requires us to find carefully two graphs which are almost the same but have different answers. We also prove that the lower bound of the classical query complexity is $\Omega(N)$, thus adding the new nontrivial property into the class of graph properties for which there is a significant gap between the quantum and classical query complexities.

**Related works:** A large literature exists for the quantum query complexity of Boolean functions. Other than OR and AND-OR trees, the complexity of the threshold function [10] was tightly characterized in the early stages. Element distinctness was also tightly shown to be $\Theta(N^{2/3})$ while the upper bound [6] and lower bound [3] of its complexity were obtained after exhaustive work of many researchers, which gave many technical contributions in (not for only quantum) complexity theory. For the monotone graph properties, Dürr et al. [16] showed the tight complexity $\Theta(N^{3/4})$ of connectivity. The quantum query complexities of total functions are polynomially related to the classical equivalents [10], and the maximum gap is conjectured to be quadratic.

The two major lower bound methods of quantum query complexity are polynomial methods [10] and adversary methods [5] (see [20] for its excellent survey.) Our lower bound of planarity testing is inspired by the application of adversary methods to connectivity in [16], which uses one-cycle vs. two-cycles as the two graphs with different answers. A similar choice of graphs is also used in [24] to get the lower bounds of several graph problems such as bipartiteness.

There have been few studies on the complexity of "non-structural" Boolean functions. All Boolean functions have quantum query complexity at most $N/2 + O(\sqrt{N})$ [13] while almost all functions have quantum query complexity at least $N/4 + \Omega(\sqrt{N})$ [4, 15].

## 2 Preliminaries

We assume the oracle (or black-box) model in the quantum setting (e.g., [10]). In this model, an input (i.e., a problem instance) is given as an oracle. For any input $x = (x_1, \ldots, x_N) \in \{0,1\}^N$, a unitary operator $O$, corresponding to a single query to an oracle, maps $|i\rangle|b\rangle|w\rangle$ to $|i\rangle|b \oplus x_i\rangle|w\rangle$ for each $i \in [N] = \{1, 2, \ldots, N\}$ and $b \in \{0,1\}$, where $w$ denotes workspace. A *quantum computation* of the oracle model is a sequence of unitary transformations $U_0 \to O \to U_1 \to O \to \cdots \to O \to U_t$, where $U_j$ may be any unitary transformation that does not depend on the input. The above computation sequence involves $t$ oracle calls, which is our measure of the complexity: The *quantum query complexity* $Q(P)$ of a problem $P$ whose input is given as an $N$-bit string is defined to be the number of quantum queries needed to solve $P$ with bounded-error, i.e., with success probability at least $1/2 + c$ where $c$ is some constant.

In this paper, our problem $P$ is to evaluate the value (0 or 1) of a Boolean function $f(x_1, \ldots, x_N)$ over $N$ variables, assuming that the truth table of $f$ is known. The *on-set* of $f$ is the set of assignments $(x_1, \ldots, x_N)$ satisfying $f(x_1, \ldots, x_N) = 1$. We denote the family of all functions whose on-set is of size $M$ by $\mathcal{F}_M$.

Our algorithms in this paper use the algorithm in [9] for the oracle identification problem defined as follows. Notice that there are $2^N$ different oracles with length $N$.

**Definition 1 (Oracle Identification Problem (OIP) [8, 9])** *Given an oracle $x$ and a set $S$ of $M$ oracle candidates out of $2^N$ ones, determine which oracle in $S$ is identical to $x$ with the promise that $x$ is a member of $S$.*

Improving the previous result in [8], Ambainis et al. [9] showed the following upper bound for the quantum query complexity of OIP when $M$ is not so large, which is asymptotically optimal.

**Theorem 1 (Optimal bound of OIP [9])** *OIP can be quantumly solved with a constant success probability by making $O(\sqrt{N\frac{\log M}{\log N}})$ queries to the given oracle if poly$(N) \leq M \leq 2^{N^d}$ for some constant $d$ $(0 < d < 1)$.*

## 3 Worst-Case Analysis

In this section, we study both upper and lower bounds for the quantum query complexity of Boolean functions in $\mathcal{F}_M$. First, we show the upper bound.

**Theorem 2 (Upper Bound)** *Any function $f \in \mathcal{F}_M$ has quantum query complexity $O(\sqrt{N\frac{\log M}{\log N}})$ if poly$(N) \leq M \leq 2^{N^d}$ for some constant $d$ $(0 < d < 1)$.*

*Proof* Recall that OIP is the problem that we are requested to find a hidden oracle, with the promise that it is a member of oracle candidate set $S$. To use this for evaluation of the Boolean function $f$, let $S$ be the on-set of $f$, which can be constructed from the known truth table of $f$. Note that $|S| = M$ since $f \in \mathcal{F}_M$. We then invoke the OIP algorithm of Theorem 1 to find the hidden oracle with $O(\sqrt{N \frac{\log M}{\log N}})$ queries, assuming the promise that the current oracle $x$ is in $S$ (actually, the promise does not hold if $f(x) = 0$). Let $z \in \{0,1\}^N$ be the string obtained by the OIP algorithm.

If $f(x) = 1$, the promise of the above OIP is indeed satisfied; $z$ is equal to $x$ with high probability.

If $f(x) = 0$, the promise does not hold; the OIP algorithm outputs some answer $z \in S$ such that $z \neq x$. To recognize this case, it suffices to check whether $z$ is equal to $x$ by using Grover search [18] with $O(\sqrt{N})(\in O(\sqrt{N \frac{\log M}{\log N}}))$ queries. This completes the proof. ■

In fact, this upper bound is optimal for the threshold function with threshold $\Theta(\log M/\log N)$, which has a query complexity of $\Omega(\sqrt{N \frac{\log M}{\log N}})$ due to the results in [10]. The following corollary is immediate.

**Corollary 1** *For $M \in poly(N)$, any function $f \in \mathcal{F}_M$ has quantum query complexity $O(\sqrt{N})$.*

This corollary together with the next lower bound theorem implies that if $M$ is in *poly*$(N)$, then any function $f \in \mathcal{F}_M$ has essentially the same complexity up to a constant factor as the OR function.

**Theorem 3 (Lower Bound)** *If $M \leq 2^{\frac{N}{2+\epsilon}}$ for any positive constant $\epsilon$, any $f \in \mathcal{F}_M$ has quantum query complexity $\Omega(\sqrt{N})$.*

*Proof* We use the sensitivity argument. Recall that the sensitivity $s_x(f)$ of a Boolean function $f$ on $x \in \{0,1\}^N$ is the number of variables $x_i$ such that $f(x) \neq f(x^i)$, where $x^i$ is the string obtained from $x$ by flipping the value of $x_i$. The sensitivity $s(f)$ of $f$ is the maximum of $s_x(f)$ over all $x$. The results of Beals et al. [10] implies $Q(f) = \Omega(\sqrt{s(f)})$. By the definition of $s(f)$ and the result by Beals et al., we can see that $Q(f) = \Omega(\sqrt{|Z|})$, where $Z$ is the set of 0-*points*, elements whose values of $f$ is 0, "around" an arbitrarily chosen element in the on-set (1-*point*). Here, "around" means the Hamming distance is 1. Therefore, if there is a 1-point around which there are $\Omega(N)$ 0-points, $Q(f) = \Omega(\sqrt{N})$.

To prove by contradiction, we assume that, around every 1-point, there are $o(N)$ 0-points, i.e., there are $(N - o(N))$ 1-points. Suppose that $(0, 0, \ldots, 0)$ is a 1-point (otherwise, we can give a similar argument using some 1-point). Set $S_0 = \{(0, 0, \ldots, 0)\}$. Define $S_k$ inductively to be the set of all 1 points around all points in $S_{k-1}$, whose Hamming weight is $k$. By assumption, the number of 1-points around every point in $S_{k-1}$ is $N - o(N) = N(1-\alpha)$ for any small $\alpha = o(1)$. For each point $x$ in $S_{k-1}$, there exist at most $(k-1)$ 1-points around $x$ in $S_{k-2}$. Thus, for each point $x$ in $S_{k-1}$, there exist at least $(N(1-\alpha) - (k-1))$ 1-points around $x$ in $S_k$. Similarly, for each point $x$ in $S_k$, there exist at most $k$ 1-points around $x$ in $S_{k-1}$. Thus, $|S_k| \geq |S_{k-1}|(N(1-\alpha) - (k-1))/k$. From this inductive inequality and $|S_0| = 1$, we have $|S_k| \geq (N(1-\alpha))(N(1-\alpha)-1)(N(1-\alpha)-2)\cdots(N(1-\alpha) - (k-1))/k!$. The number of inputs $x$ such that $f(x) = 1$ and the Hamming weight of $x$ is at most $k$ is $T(k) = |S_0| + \cdots + |S_k|$. We will show $T(k) > M$ for some $k \leq N/2$, a contradiction, as follows. $T(k) > |S_k| \geq (N(1-\alpha))(N(1-\alpha)-1)\cdots(N(1-\alpha)-(k-1))/k! > \left(\frac{N(1-\alpha)}{k}\right)^k$. For $k = \frac{N}{2+\epsilon}$, we obtain $T(k) > 2^{\frac{N}{2+\epsilon}} \geq M$. ■

The above lower bound is tight, since it is easy to construct a Boolean function for any $M \leq 2^{\frac{N}{2+\epsilon}}$ such that its query complexity is $O(\sqrt{N})$. Thus we have shown that there are Boolean functions, $f_1$ and $f_2$, which are "easiest" and "hardest" in class $\mathcal{F}_M$, such that $Q(f_1) = \Theta(\sqrt{N})$ and $Q(f_2) = \Theta\left(\sqrt{N\frac{\log M}{\log N}}\right)$.

## 4 Average-Case Analysis

This section considers upper and lower bounds for the quantum query complexity for almost all functions in $\mathcal{F}_M$. They are essentially $O(\log M)$ and $\Omega(\log M/\log N)$, thus having a $\log N$ factor gap. Note that the bounds hold for the entire range of $M$.

**Theorem 4 (Upper Bound)** *Almost all Boolean functions in $\mathcal{F}_M$ have quantum query complexity $O(\log M + \sqrt{N})$.*

*Proof* It suffices to show the statement for $poly(N) < M < 2^{\frac{N}{3}}$ since the case of $M \in poly(N)$ is obtained by Corollary 1, and the case of $M \geq 2^{\frac{N}{3}}$ leads to the trivial bound. We can make the following claim (proof will be given later):

**Claim.** *If we generate a random Boolean function $f$ whose on-set $S_f$ has size $M$, then, for almost all cases, any two of $M$ elements in $S_f$ differ from each other in the first $k$ bits, where $k = 3\log M$ (which is smaller than $N$).*

Now the following algorithm works by using the claim. Below we will use $k$ as $k = 3\log M$. First, we identify the first $k$ bits of the current input $x$ by making $k$ classical queries. Then we can decide that $f(x) = 0$ regardless of the remaining bits, if the $k$-bit string is different from the first-$k$-bit string of any element in the on-set. Otherwise, $f(x)$ can have value 1, depending on the remaining $N - k$ bits. For the latter case, the claim implies that, for almost all functions, there is only one possible way of assigning 0/1 to the remaining $N - k$ bits that determines $f(x) = 1$. Thus, we just check whether the remaining bits are subject to such one possibility or not by using Grover search. In total, the query complexity is $O(k + \sqrt{N}) = O(\log M + \sqrt{N})$.

What remains is to show the above claim. The number of all functions whose on-set has size $M$ is $\binom{2^N}{M}$. Among such functions, we count the number of our desired functions, i.e., the functions such that any two inputs in the on-set differ from each other in the first $k$ bits. We first consider the number of possible assignments to the first $k$ bits of $M$ inputs. The number of possibilities is $\binom{2^k}{M}$ since we need to choose different $M$ $k$-bit strings among the $2^k$ possibilities. We then choose the remaining $(N-k)$ bits arbitrarily, i.e., $2^{N-k}$ possibilities for each of the $M$ assignments to the first $k$ bits. Thus, the number of possibilities of assigning the remaining $(N - k)$ bits for all $M$ inputs is $(2^{(N-k)})^M = 2^{M(N-k)}$. In conclusion, the number of our desired functions is $\binom{2^k}{M}2^{M(N-k)}$. The ratio of our desired functions is $\frac{\binom{2^k}{M}2^{M(N-k)}}{\binom{2^N}{M}}$. We can show that, by using $k = 3\log M$, this ratio is larger than $\exp(-\frac{M^2-M}{2(M^3-M+1)})$. Hence, the ratio approaches 1 as $N$ (and hence $M$) goes to infinity. ■

**Theorem 5 (Lower Bound)** *Almost all Boolean functions in $\mathcal{F}_M$ have quantum query complexity $\Omega(\log M/\log N + \sqrt{N})$.*

*Proof* For almost all $f \in \mathcal{F}_M$, we prove that $Q(f) = \Omega(\log M/\log N)$ when $M > N^{\sqrt{N}}$ since by Theorem 3 the lower bound $\Omega(\sqrt{N})$ holds for $M \leq N^{\sqrt{N}}$. Moreover, we assume that $M \leq 2^{N-1}$

without loss of generality. We shall show the lower bound for the *unbounded-error* setting, where the success probability suffices to be at least $1/2+\epsilon$, for any positive $\epsilon$ (which does not need to be a constant as in the bounded-error setting). Obviously, any lower bound in this setting also holds in the bounded-error setting.

The lower bound of unbounded-error query complexity of a Boolean function $f$ is characterized by the minimum degree of its *sign-representing polynomial* $p$, a real-valued polynomial with properties that $p(x)$ is positive whenever $f(x)=0$ and $p(x)$ is negative whenever $f(x)=1$, for all $N$-bit strings $x$. There are two nice properties of sign-representing polynomials, which are useful for our proof. First, as shown in [23] (also, implicitly in [12]), the unbounded-error quantum query complexity of $f$ is exactly half of the minimum degree of its sign-representing polynomial. Secondly, the number of Boolean functions of $N$ variables whose minimum degrees of sign-representing polynomials are at most $d$, denoted as $T(N,d)$, is also known to be at most $2\sum_{k=0}^{D-1}\binom{2^N-1}{k}$, for $D=\sum_{i=0}^{d}\binom{N}{i}$ as proved in [1].

Hence to show that almost all Boolean functions in $\mathcal{F}_M$ have unbounded-error quantum query complexity $\Omega(\log M/\log N)$ , it suffices to show that $T(N,\frac{\log M}{2\log N})$ is small compared to $\binom{2^N}{M}$, i.e., the size of $\mathcal{F}_M$. Notice that in this case $D=\sum_{i=0}^{\log M/(2\log N)}\binom{N}{i}\leq N^{\frac{\log M}{2\log N}}=\sqrt{M}$, and therefore,

$$\frac{T\left(N,\log M/(2\log N)\right)}{\binom{2^N}{M}}\leq\frac{2\sum_{k=0}^{D-1}\binom{2^N-1}{k}}{\binom{2^N}{M}}\leq\frac{2\sum_{k=0}^{\sqrt{M}-1}\binom{2^N}{k}}{\binom{2^N}{M}}\leq\frac{2\sqrt{M}\binom{2^N}{\sqrt{M}}}{\binom{2^N}{M}}. \tag{1}$$

Moreover, the right-hand side of (1) is bounded by

$$\frac{2\sqrt{M}\binom{2^N}{\sqrt{M}}}{\binom{2^N}{M}}=2\sqrt{M}\frac{M\cdots(\sqrt{M}+1)}{(2^N-\sqrt{M})\cdots(2^N-M+1)}\leq\frac{2\sqrt{M}}{2^N-M+1}. \tag{2}$$

By the assumption that $M\leq 2^{N-1}$, the right-hand side of (2) goes to 0 as $N$ goes to infinity, which completes the proof. ■

# 5 Applications

As application of Theorem 2, we consider the problem of graph property testing, i.e., the problem of testing, for a given graph $G$ as an oracle, if $G$ has a certain property. More precisely, an $n$-vertex graph is given as $n(n-1)/2$ Boolean variables, $x_i$ for $i\in\{1,\ldots,n(n-1)/2\}$, representing the existence of the $i$th possible edge $e_i$, i.e., $x_i=1$ if and only if $e_i$ exists. In this setting, graph property testing is just to evaluate a Boolean function $f$ depending on the $n(n-1)/2$ variables such that $f(x_1,\ldots,x_{n(n-1)/2})=1$ if and only if the graph has a certain property. An interpretation of graph property testing according to Theorem 2 is to decide if $G$ is a member of $\mathcal{F}$ for the family $\mathcal{F}$ of all graphs with certain properties. Thus, Theorem 2 directly gives the next theorem with $M=|\mathcal{F}|$ and $N=n(n-1)/2$.

**Theorem 6** *For graph family $\mathcal{F}$ defined as in the above, graph property testing can be solved with $O(\sqrt{n^2\log|\mathcal{F}|/\log n})$ quantum queries, if poly$(n)\leq|\mathcal{F}|\leq 2^{n^{2d}}$ for some constant $d$ $(0<d<1)$.*

An interesting special case of the problem is graph isomorphism testing: the problem of deciding if a given graph $G$ is isomorphic to an arbitrary fixed graph $G'$.

**Corollary 2** *Graph isomorphism testing can be solved with $O(n^{1.5})$ quantum queries.*

*Proof* The number of graphs isomorphic to $G'$ is at most the number of permutations over the vertex set, which is $n! = 2^{O(n \log n)}$. ■

This upper bound is optimal in the worst case, since the lower bound $\Omega(n^{1.5})$ of connectivity testing problem in [16] is essentially that of deciding whether a given graph is isomorphic to one cycle or two cycles. Another interesting special case is *planarity testing*, the problem of testing if a given graph is planar.

**Corollary 3** *Planarity testing can be solved with $O(n^{1.5})$ quantum queries.*

*Proof* Since the number of edges of any planar graph is at most $3n - 6 < 3n$ [14], the number of planar graphs is at most $\binom{n(n-1)/2}{3n} < n^{2 \cdot 3n} = 2^{6n \log n}$. ■

For verifying optimality, we prove a lower bound of $\Omega(n^{1.5})$ for this problem. We also prove a tight classical lower bound of $\Omega(n^2)$.

To prove the lower bounds, we use the next two theorems.

**Theorem 7 (Quantum adversary method [5], reformulated by [2])** *Let $\mathcal{A} \subseteq F^{-1}(0)$ and $\mathcal{B} \subseteq F^{-1}(1)$ be sets of inputs to function $F$. Let $R(A, B) \geq 0$ be a real-valued function, and for $A \in \mathcal{A}$, $B \in \mathcal{B}$, and location $i$, let$\theta(A, i) = \frac{\sum_{B^* \in \mathcal{B}\ :\ A(i) \neq B^*(i)} R(A, B^*)}{\sum_{B^* \in \mathcal{B}} R(A, B^*)}$, $\theta(B, i) = \frac{\sum_{A^* \in \mathcal{A}\ :\ A^*(i) \neq B(i)} R(A^*, B)}{\sum_{A^* \in \mathcal{A}} R(A^*, B)}$, where $A(i)$ and $B(i)$ denotes the value of the $i$th variable for $A$ and $B$, respectively, the denominators are all nonzero. Then the number of quantum queries needed to evaluate $F$ with probability at least $9/10$ is $\Omega(1/\upsilon_{\text{geom}})$, where*

$$\upsilon_{\text{geom}} = \max_{\substack{A \in \mathcal{A},\ B \in \mathcal{B},\ i\ :\\ R(A,B) > 0,\ A(i) \neq B(i)}} \sqrt{\theta(A, i)\, \theta(B, i)}.$$

**Theorem 8 (Classical adversary method [2])** *Let $\mathcal{A}, \mathcal{B}, R, \theta$ be the same as in Theorem 7. Then the number of randomized queries needed to evaluate $F$ with probability at least $9/10$ is $\Omega(1/\upsilon_{\min})$, where*

$$\upsilon_{\min} = \max_{\substack{A \in \mathcal{A},\ B \in \mathcal{B},\ i\ :\\ R(A,B) > 0,\ A(i) \neq B(i)}} \min\{\theta(A, i), \theta(B, i)\}.$$

Now we are ready to present our lower bound.

**Theorem 9** *Solving planarity testing needs $\Omega(n^{1.5})$ queries in the quantum setting, and $\Omega(n^2)$ queries in the classical setting.*

*Proof*

Before giving our formal proof, we briefly describe our proof idea. To apply the adversary method, we need to take two sets, $\mathcal{A}$ and $\mathcal{B}$, of non-planar graphs and planar graphs, respectively, such that for "many" pairs $(A, B) \in \mathcal{A} \times \mathcal{B}$, $A$ and $B$ are hard to distinguish. We define $\mathcal{A}$ as a set of subdivisions of the complete graph $K_5$. Focusing on one cycle included in each $A \in \mathcal{A}$, we define $\mathcal{B}$ be a set of planar graphs $B$ obtained by dividing the cycle to two cycles. We then give a relation between graphs $A \in \mathcal{A}$ and graphs $B \in \mathcal{B}$. The relation is seemingly similar to that defined in [16] for proving the quantum lower bound of connectivity. However, their relation is not enough to keep "many" pairs in our case. Different from the case of [16], we place a careful restriction on the way of transforming $A$ to $B$ as well as $B$ to $A$.

Our formal proof is as follows. To get lower bounds by using Theorems 7 and 8, we will define sets $\mathcal{A}$ and $\mathcal{B}$, and function $R : \mathcal{A} \times \mathcal{B} \rightarrow \{0, 1\}$. Let $a, b, c, d$ be any four vertices of complete graph

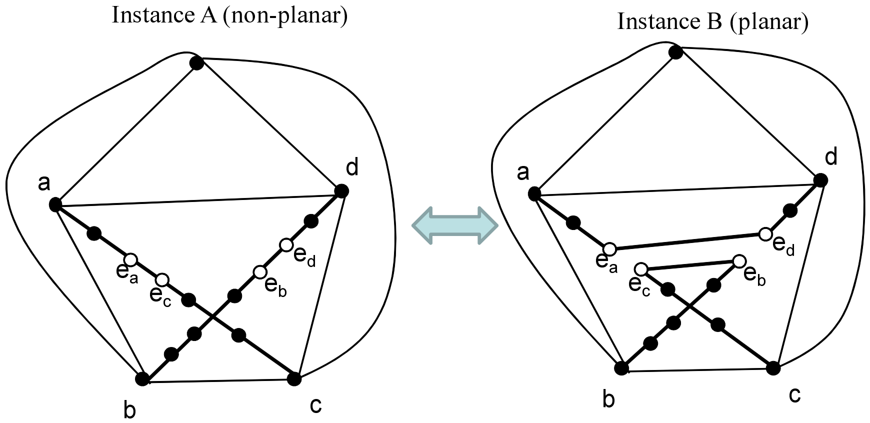

Figure 1: Instance $A \in \mathcal{A}$ and instance $B \in \mathcal{B}$

$K_5$ with five vertices. Let $\mathcal{A}$ be the set of graphs obtained by replacing edges $(a, c)$ and $(b, d)$ of the $K_5$ with path $P_{ac}$ between $a$ and $c$ and path $P_{bd}$ between $b$ and $d$, respectively, on which there are $n - 5$ vertices except $a, b, c, d$, and each one of which is at most three times longer than the other. Every graph in $\mathcal{A}$ is not planar and has $n$ vertices, since it is a subdivision of $K_5$, i.e., it becomes $K_5$ by contracting all but one edges on each of $P_{ac}$ and $P_{bd}$. An example of an instance in $\mathcal{A}$ is shown at the left side in Fig. 1. Let $\mathcal{B}$ be the set of graphs obtained by replacing $(a, c)$ and $(b, d)$ of the $K_5$ with path $P_{ad}$ between $a$ and $d$ and path $P_{bc}$ between $b$ and $c$, respectively, on which there are $n - 5$ vertices except $a, b, c, d$, and one of which is at most three times longer than the other. It is easy to see that every graph in $\mathcal{B}$ is planar and has $n$ vertices. An example of an instance in $\mathcal{B}$ is shown at the right side in Fig. 1.

Now, we define $B \in \mathcal{B}$ such that $R(A, B) = 1$ for every $A \in \mathcal{A}$. For every graph $A \in \mathcal{A}$, let $(e_a, e_c)$ be any edge on $P_{ac}$, where $e_a$ is assumed to be closer to $a$ on the path, and let $(e_b, e_d)$ be any edge on $P_{bd}$, where $e_b$ is assumed to be closer to $b$ on the path (see the left graph in Fig. 1). If we replace $(e_a, e_c)$ and $(e_b, e_d)$ with $(e_a, e_d)$ and $(e_b, e_c)$, the resulting graph has paths $P_{ad}$ and $P_{bc}$ instead of $P_{ac}$ and $P_{bd}$ (see the right graph in Fig. 1). We can guarantee that each of $P_{ad}$ and $P_{bc}$ is at most three times longer than the other by imposing some restriction on the choice of $(e_b, e_d)$ for each $(e_a, e_c)$, which will be proved later; the resulting graph is a member of $\mathcal{B}$. Similarly, we define $A \in \mathcal{A}$ such that $R(A, B) = 1$ for every $B \in \mathcal{B}$. For every graph $B \in \mathcal{B}$, let $(e_a, e_d)$ be any edge on $P_{ad}$, where $e_a$ is assumed to be closer to $a$ on the path, and let $(e_b, e_c)$ be any edge on $P_{bc}$, where $e_b$ is assumed to be closer to $b$ on the path. If we replace $(e_a, e_d)$ and $(e_b, e_c)$ with $(e_a, e_c)$ and $(e_b, e_d)$, the resulting graph has paths $P_{ac}$ and $P_{bd}$ instead of $P_{ad}$ and $P_{bc}$. Since we can guarantee that each of $P_{ac}$ and $P_{bd}$ is at most three times longer than the other by imposing a similar restriction (shown later) on the choice of $(e_b, e_c)$ for each $(e_a, e_d)$, the resulting graph is a member of $\mathcal{A}$.

We here show the restriction on the choice of $(e_b, e_d)$ for each $(e_a, e_c)$ when relating $A$ to $B$ (a similar restriction works when relating $B$ to $A$). Without loss of generality, $P_{ac}$ is shorter than or equal to $P_{bd}$ (otherwise, we just switch $P_{ac}$ and $P_{bd}$). Let the length of paths $P_{ac}$ and $P_{bd}$ be $cL$ and $(1 - c)L$, respectively, for $1/4 \leq c \leq 1/2$, where $L = n - 3$. Notice that the sum of the lengths of $P_{ac}$ and $P_{bd}$ is always $(n - 3)$. If the subpath between $a$ and $e_a$ of $P_{ac}$ is of length $k \in \{0, \ldots, cL - 1\}$, we choose $(e_b, e_d)$ such that the subpath of $P_{bd}$ between $e_b$ and $d$ has the length of at least $\max\{L/4 - k, 1\}$ and at most $\min\{3L/4 - k, (1 - c)L\}$. Then after replacing $(e_a, e_c)$ and $(e_b, e_d)$ with $(e_a, e_d)$ and $(e_b, e_c)$, the lengths of $P_{ad}$ and $P_{bc}$ are each at least $L/4$ and at most $3L/4$. This edge replacement is always possible in many ways, since there are many edges $(e_b, e_d)$ that satisfies the condition, as proved below. More precisely, there are at least $L/4$ choices of $(e_b, e_d)$:

$\Delta \equiv \min\{3L/4-k,(1-c)L\} - \max\{L/4-k,1\} \geq L/4$. If $\min\{3L/4-k,(1-c)L\} = 3/4L-k$,

$$\Delta = 3/4L - k - \max\{L/4-k,1\} = \min\{L/2, 3L/4-k-1\},$$

which is at least $\min\{L/2, 3L/4-cL\} \geq L/4$.
If $\min\{3L/4-k,(1-c)L\} = (1-c)L$,

$$\Delta = (1-c)L - \max\{L/4-k,1\} = \min\{(3/4-c)L+k, (1-c)L-1\},$$

which is at least $\min\{(3/4-c)L, (1-c)L-1\} \geq L/4$.

This means that, for each $(e_a, e_c)$, there are $\Omega(L)$ choices of $(e_b, e_d)$. Since there are $cL$ choices of $(e_a, e_c)$, $\sum_{B^*\in\mathcal{B}} R(A,B^*) = \Omega(cL\cdot L) = \Omega(n^2)$, implying $\Theta(n^2)$. Similarly, $\sum_{A^*\in\mathcal{A}} R(A^*,B) = \Theta(n^2)$.

For any fixed $A \in \mathcal{A}$, $\sum_{B^*\in\mathcal{B}\,:\,A(i)=1,B^*(i)=0} R(A,B^*)$ attains the maximum value of $\Theta(n)$, when $i$ is the index of an edge on the shorter path of $P_{ac}$ and $P_{bd}$. For any fixed $B \in \mathcal{B}$, $\sum_{A^*\in\mathcal{A}\,:\,A^*(i)=1,B(i)=0} R(A^*,B)$ is a positive constant $\Theta(1)$ for all $i$ such that there exists at least one $A^*$ satisfying that $A^*(i) = 1, B(i) = 0$ and $R(A^*,B) = 1$. This is because, to flip $x_i$, we need to pick up a pair of edges which are adjacent to the $i$th possible edge, and replace the edge pair with another pair of edges including the $i$th possible edge. Thus, for every $A$ and $B$ such that $R(A,B) = 1$, $\max_{i:A(i)=1,B(i)=0} \sqrt{\theta(A,i)\theta(B,i)} = \Theta(\sqrt{(n/n^2)(1/n^2)}) = \Theta(1/n^{1.5})$. Similarly, $\max_{i:A(i)=0,B(i)=1} \sqrt{\theta(A,i)\theta(B,i)} = \Theta(1/n^{1.5})$. The quantum lower bound follows from Theorem 7.

The classical lower bound can be obtained by a similar argument. For any $A$ and $B$ such that $R(A,B) = 1$, $\max_{i:A(i)=1,B(i)=0} \min\{\theta(A,i),\theta(B,i)\} = \Theta(1/n^2)$, and $\max_{i:A(i)=0,B(i)=1} \min\{\theta(A,i),\theta(B,i)\} = \Theta(1/n^2)$, since the smallest value of $\theta(A,i)$ and $\theta(B,i)$ is $\Theta(1/n^2)$. The classical lower bound follows from Theorem 8. ■

## References

[1] M. Anthony. Classification by polynomial surfaces. *Discrete Applied Mathematics* 61:91–103, 1995.

[2] S. Aaronson: Lower bounds for local search by quantum arguments. *SIAM J. Comput.* 35(4):804-824, 2006.

[3] S. Aaronson and Y. Shi. Quantum lower bounds for the collision and the element distinctness problems. *J. ACM* 51(4): 595-605, 2004.

[4] A. Ambainis. A note on quantum black-box complexity of almost all Boolean functions. *Inf. Process. Lett.* 71(1):5–7, 1999.

[5] A. Ambainis. Quantum lower bounds by quantum arguments, *J. Comput. Sys. Sci.* 64:750–767, 2002.

[6] A. Ambainis. Quantum walk algorithm for element distinctness. *SIAM J. Comput.* 37(1): 210-239, 2007.

[7] A. Ambainis, A. M. Childs, B. W. Reichardt, R. Špalek, and S. Zhang. Any AND-OR formula of size $N$ can be evaluated in time $N^{1/2+o(1)}$ on a quantum computer. In *Proc. 48th FOCS*, pages 363–372, 2007.

[8] A. Ambainis, K. Iwama, A. Kawachi, H. Masuda, R. H. Putra, and S. Yamashita. Quantum identification of Boolean oracles. In *Proc. 21st STACS, Lecture Notes in Comput. Sci.* 2996:105–116, 2004.

[9] A. Ambainis, K. Iwama, A. Kawachi, R. Raymond, and S. Yamashita. Improved algorithms for quantum identification of Boolean oracles. *Theor. Comput. Sci.* 378(1):41–53, 2007.

[10] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *J. ACM* 48(4):778–797, 2001.

[11] H. Buhrman, R. Cleve, R. de Wolf, and C. Zalka. Bounds for small-error and zero-error quantum algorithms. In *Proc. 40th FOCS*, pages 358–368, 1999.

[12] H. Buhrman, N. Vereschagin, and R. de Wolf. On computation and communication with small bias. In *Proc. 22nd CCC*, pages 24–32, 2007.

[13] W. van Dam. Quantum oracle interrogation: getting all information for almost half the price. In *Proc. 39th FOCS*, pages 362–367, 1998.

[14] R. Diestel. *Graph Theory.* Graduate Texts in Mathematics. Springer, 2nd edition, 2000.

[15] R. O'Donnell and R. A. Servedio. Extremal properties of polynomial threshold functions. In *Proc. 18th CCC*, pages 3–12, 2003.

[16] C. Dürr, M. Heiligman, P. Høyer, and M. Mhalla. Quantum query complexity of some graph problems. *SIAM J. Comput.* 35(6): 1310-1328, 2006.

[17] E. Farhi, J. Goldstone, and S. Gutmann. A quantum algorithm for the Hamiltonian NAND tree. quant-ph/0702144, 2007.

[18] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proc. 28th STOC*, pages 212–219, 1996.

[19] P. Høyer, M. Mosca, and R. de Wolf. Quantum search on bounded-error inputs. In *Proc. 30th ICALP, Lecture Notes in Comput. Sci.* 2719: 291–299, 2003.

[20] P. Høyer, R. Špalek. Lower bounds on quantum query complexity. *Bulletin of the EATCS* 87: 78-103, 2005.

[21] F. Magniez, M. Santha, and M. Szegedy. Quantum algorithms for the triangle problem. In *Proc. ACM-SIAM SODA*, pages 1109–1117, 2005.

[22] F. Magniez, A. Nayak, J. Roland, and M. Santha. Search via quantum walk. In *Proc. 39th STOC*, pages 575-584, 2007.

[23] A. Montanaro, H. Nishimura, and R. Raymond. Unbounded-error quantum query complexity. To appear in *Proc. 19th ISAAC*, 2008.

[24] S. Zhang. On the power of Ambainis lower bounds. *Theor. Comput. Sci.* 339(2-3): 241-256, 2005.