Generalized Deutsch-Jozsa problem and the optimal quantum algorithm

Daowen Qiu[1,2,*] and Shenggen Zheng[1]

[1]*Institute of Computer Science Theory, School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China*

[2]*Instituto de Telecomunicações, Departamento de Matemática, Instituto Superior Técnico, Avenida Rovisco Pais 1049-001, Lisbon, Portugal*

(Received 4 February 2018; published 20 June 2018)

The Deutsch-Jozsa algorithm is one of the first examples of a quantum algorithm that is exponentially faster than any possible deterministic classical algorithm. It was proposed by Deutsch and Jozsa in 1992 with improvements by Cleve, Ekert, Macchiavello, and Mosca in 1998. The Deutsch-Jozsa problem is a promise problem and we can equivalently describe it as a partial function $\mathrm{DJ}_n^0 : \{0,1\}^n \rightarrow \{0,1\}$ defined as $\mathrm{DJ}_n^0(x) = 1$ for $|x| = n/2$, $\mathrm{DJ}_n^0(x) = 0$ for $|x| = 0,n$, and it is undefined for the rest of the cases, where n is even, and $|x|$ is the Hamming weight of x. The optimal quantum algorithm needs only one query to compute DJ_n^0 but the classical deterministic algorithm requires $2^{n-1} + 1$ queries to compute it in the worse case. In this article, we generalize the Deutsch-Jozsa problem as $\mathrm{DJ}_n^k(x) = 1$ for $|x| = n/2$, $\mathrm{DJ}_n^k(x) = 0$ for $|x|$ in the set $\{0,1,\ldots,k,n-k,n-k+1,\ldots,n\}$, and it is undefined for the rest of the cases, where $0 \leqslant k < n/2$. In particular, we give and prove an optimal exact quantum query algorithm with complexity $k+1$ for computing the generalized Deutsch-Jozsa problem DJ_n^k. It is clear that the case of $k = 0$ is in accordance with the Deutsch-Jozsa problem. Also, we give a method for finding the approximate and exact degrees of symmetric partial Boolean functions.

DOI: 10.1103/PhysRevA.97.062331

I. INTRODUCTION

Quantum computing models can be divided into *bounded-error* and *exact* versions in terms of their outputs. A bounded-error model means that the mistake probability for any output cannot be beyond an error value given *a priori*, and an exact model requires its outputs be fully correct always, without any error allowed. *Exact quantum computing models* have been studied in the frameworks of *quantum finite automata* [6,24] and particularly *quantum query models* (for example, [2,5,11,12,19,28,31]).

The quantum query models are the quantum analog to the classical Boolean decision tree models, so they are also called *quantum decision tree* models and are at least as powerful as the classical decision tree models [12]. The implementation procedure of a quantum decision tree model is exactly a *quantum query algorithm*, and it can be roughly described as follows: It starts with a fixed starting state $|\psi_s\rangle$ of a Hilbert $\mathcal{H}$ and will perform the sequence of operations $U_0,O_x,U_1,\ldots,O_x,U_t$, where U_i's are unitary operators that do not depend on the input x but the query O_x does. This leads to the final state $|\psi_f\rangle = U_t O_x U_{t-1} \ldots U_1 O_x U_0 |\psi_s\rangle$. The result is obtained by measuring the final state $|\psi_f\rangle$.

A quantum query algorithm $\mathcal{A}$ *exactly computes* a Boolean function f if its output equals $f(x)$ with probability 1, for all input x. $\mathcal{A}$ *computes with bounded error* f if its output equals $f(x)$ with probability at least $\frac{2}{3}$, for all input x. The *exact quantum query complexity* denoted by $Q_E(f)$ is the minimum number of queries used by any quantum algorithm which computes $f(x)$ exactly for all input x.

The exact quantum query algorithms for computing total Boolean functions have been extensively studied [2,5,15,20,22,27,30,31,34]; one quantum speedup was $Q_E(f) = O[D(f)^{0.8675\ldots}]$ by Ambainis [2], and recently Ambainis *et al.* [3] presented a better separation with a superquadratic gap between its quantum and deterministic query complexities, where $D(f)$ denotes the minimum number of queries used by any classical deterministic query algorithm.

However, for computing partial Boolean functions, there can be an exponential separation between exact quantum and classical deterministic query complexity, and one result was the well-known Deutsch-Jozsa algorithm proposed by Deutsch and Jozsa in 1992 [19] with improvements by Cleve, Ekert, Macchiavello, and Mosca in 1998 [15]. The Deutsch-Jozsa problem [19] can be described as a partial Boolean function $\mathrm{DJ}_n^0 : \{0,1\}^n \rightarrow \{0,1\}$ defined as follows: n is even, and $\mathrm{DJ}_n^0(x) = 1$ for $|x| = \frac{n}{2}$ and $\mathrm{DJ}_n^0(x) = 0$ for $|x| = 0$ or n, and the other cases are undefined, where $|x|$ is the Hamming weight of x. The Deutsch-Jozsa problem has attracted a lot of research and discussion (for example, [7,16,26,31]), and the physical realization was implemented in [33]. Recently, Montanaro, Jozsa, and Mitchison [31] generalized the Deutsch-Jozsa problem to another partial Boolean function, say $\mathrm{DJ}_n^1 : \{0,1\}^n \rightarrow \{0,1\}$ defined as DJ_n^0 except for $\mathrm{DJ}_n^1(x) = 0$ for $|x| = 0,1,n-1,n$. Also, Montanaro *et al.* [31] designed an exact quantum two-query algorithm to compute it by using an analytical method. However, the optimality of the exact quantum two-query algorithm has not been demonstrated.

Ambainis [1] showed that almost all total Boolean functions have high approximate degree, so we are also interested in partial Boolean functions with a lower degree. Indeed, partial Boolean functions have also been called promise problems [21,23], and both symmetric Boolean functions and partial

[*]issqdw@mail.sysu.edu.cn

Boolean functions have had important applications in cryptography (for example, [18,21,23]).

The remainder of this article is organized as follows. In Sec. II we give and recall some definitions, notions, and results concerning quantum query algorithms and complexity that are useful in the article, and we also present the main result that we will demonstrate. After that, in order to demonstrate the lower bound of our quantum algorithm, in Sec. III we study the representation of symmetric partial Boolean functions with multilinear polynomials, and give a method for finding the approximate (and exact) degree of symmetric partial Boolean functions. Then in Sec. IV we investigate the exact quantum and classical deterministic query complexity of the generalized Deutsch-Jozsa problem, that is, the function DJ_n^k, and we present an optimal exact quantum $(k+1)$-query algorithm to compute DJ_n^k, but its classical deterministic query complexity is $n/2+k+1$. Section V summarizes the main results and mentions some problems for further study. Some verifications are put in Appendixes A and B.

II. PRELIMINARIES

Let f be a Boolean function from $D \subseteq \{0,1\}^n$ to $\{0,1\}$. If $D = \{0,1\}^n$, then f is called a total Boolean function. Otherwise, f is called a partial Boolean function or a promise problem [21,23] and D is referred to as the domain of definition or promised set.

A (partial) Boolean function f is called symmetric if $f(x)$ only depends on the Hamming weight of x, i.e., $|x|$. Some characteristics of the symmetric Boolean functions were given in, for example, [18]. Some common symmetric functions over $\{0,1\}^n$ are listed as follows.

$\mathrm{OR}_n(x) = 1$ if and only if $|x| \geqslant 1$;
$\mathrm{AND}_n(x) = 1$ if and only if $|x| = 1$;
$\mathrm{PARITY}_n(x) = 1$ if and only if $|x|$ is odd;
$\mathrm{MAJ}_n(x) = 1$ if and only if $|x| > n/2$;
$\mathrm{EXACT}_n^k(x) = 1$ if and only if $|x| = k$, where $0 \leqslant k \leqslant n$;
$\mathrm{THRESHOLD}_n^k(x) = 1$ if and only if $|x| \geqslant k$, where $0 \leqslant k \leqslant n$.

Definition 1. Let $f : \{0,1\}^n \to \{0,1\}$ be a partial Boolean function, and let $D \subseteq \{0,1\}^n$ be its domain of definition. If for any $x \in D$ and for any $y \in \{0,1\}^n$ with $|x| = |y|$, it holds that $y \in D$ and $f(x) = f(y)$, then f is called a *symmetric partial Boolean function*. When $D = \{0,1\}^n$, f is a symmetric function.

Clearly, if $f : \{0,1\}^n \to \{0,1\}$ is a symmetric partial function, then its domain of definition has the version $\{x : |x| = k_1,k_2,\ldots,k_l\}$ for some $0 \leqslant k_i \leqslant n$ with $i = 1,2,\ldots,l$.

Isomorphism is useful in the study of query complexity, and two partial functions f and g over $\{0,1\}^n$ are *isomorphic* if they are equal up to negations and permutations of the input variables, and negation of the output variable.

Fact 1. For any two partial functions f,g over $\{0,1\}^n$, if they are isomorphic, then they have the same (exact) quantum query complexity.

Proof. Let $g(x) = (\neg)f(\pi((\neg)x_1,(\neg)x_2,\ldots,(\neg)x_n))$ where π is a permutation. Suppose that there is a t-queries quantum algorithm $\mathcal{A}$ that computes $f(x)$, and let $\mathcal{A}(x)$ represent the output (0 or 1) for input x. Now for any x in the domain of definition of g, we consider the following t-queries quantum algorithm $\mathcal{A}'$:

$$\mathcal{A}'(x) = (\neg)\mathcal{A}U_1U_0(x), \tag{1}$$

where $U_0(x) = ((\neg)x_1,(\neg)x_2,\ldots,(\neg)x_n)$ and $U_1(x) = \pi(x)$. It is clear that $\mathcal{A}'$ computes exactly function g. ■

Remark 1. Given a symmetric partial function $f : \{0,1\}^n \to \{0,1\}$, with the domain D of definition, it can be equivalently described by a vector $(b_0,b_1,\ldots,b_n) \in \{0,1,*\}^{n+1}$, where $f(x) = b_{|x|}$, i.e., b_k is the value of $f(x)$ when $|x| = k$, and $f(x)$ is "undefined" for $b_{|x|} = *$. In the interest of simplicity, sometimes we will use the vector to denote a symmetric partial function in this article.

Concerning the n-bit symmetric partial functions, it is clear that the following functions are isomorphic to each other:
$(b_0,b_1,\ldots,b_n)$;
$(b_n,b_{n-1},\ldots,b_0)$;
$(\bar{b}_0,\bar{b}_1,\ldots,\bar{b}_n)$;
$(\bar{b}_n,\bar{b}_{n-1},\ldots,\bar{b}_0)$.

We need to introduce some complexity measures for symmetric partial functions.

Definition 2. Let f be a partial function with a domain of definition $D \subseteq \{0,1\}^n$. For $0 \leqslant \varepsilon < 1/2$, we say a real multilinear polynomial p approximates f with error ε if
(1) $|p(x) - f(x)| \leqslant \varepsilon$ for all $x \in D$;
(2) $0 \leqslant p(x) \leqslant 1$ for all $x \in \{0,1\}^n$.

The approximate degree of f with error ε, denoted by $\widetilde{\deg}_\varepsilon(f)$, is the minimum degree among all real multilinear polynomials that approximate f with error ε.

Clearly, if $\varepsilon = 0$, then $\widetilde{\deg}_0(f)$ is the *exact degree* of f. Furthermore, if $D = \{0,1\}^n$, i.e., f is a total function, then the exact degree of f is exactly the degree of f as usual [12], denoted by $\deg(f)$. In the interest of simplicity, sometimes we just identity $\widetilde{\deg}_0(f)$ with $\deg(f)$ for any partial Boolean function f, since no confusion leads.

Let input $x = x_1 \ldots x_n \in \{0,1\}^n$ for some fixed n. We will consider a Hilbert space $\mathcal{H}$ with basis states $|i,j\rangle$ for $i \in \{0,1,\ldots,n\}$ and $j \in \{1,\ldots,m\}$ (where m can be chosen arbitrarily). A query O_x to an input $x \in \{0,1\}^n$ will be formulated as the following unitary transformation:
$O_x|0,j\rangle = |0,j\rangle$;
$O_x|i,j\rangle = (-1)^{x_i}|i,j\rangle$ for $i \in \{1,2,\ldots,n\}$.

A quantum query algorithm $\mathcal{A}$ which uses t queries for an input x consists of a sequence of unitary operators $U_0,O_x,U_1,\ldots,O_x,U_t$, where U_i's do not depend on the input x and the query O_x does. The algorithm will start in a fixed starting state $|\psi_s\rangle$ of $\mathcal{H}$ and will perform the above sequence of operations. This leads to the final state

$$|\psi_f\rangle = U_tO_xU_{t-1}\ldots U_1O_xU_0|\psi_s\rangle. \tag{2}$$

The final state is then measured with a measurement $\{M_0,M_1\}$. For an input $x \in \{0,1\}^n$, we denote $\mathcal{A}(x)$ the output of the quantum query algorithm $\mathcal{A}$. Obviously, $Pr[\mathcal{A}(x) = 0] = \|M_0|\psi_f\rangle\|^2$ and $Pr[\mathcal{A}(x) = 1] = \|M_1|\psi_f\rangle\|^2 = 1 - Pr[\mathcal{A}(x) = 0]$. We say that the quantum query algorithm $\mathcal{A}$ computes f within an error ε if for every input $x \in \{0,1\}^n$ it holds that $Pr[\mathcal{A}(x) = f(x)] \geqslant 1 - \varepsilon$. If $\varepsilon = 0$, we says that the quantum algorithm is exact. For more details on quantum query complexity, we may refer to [2,11,12,31].

Quantum query models are one of the most important computing models in quantum computing. In this complexity model [12], an algorithm is charged for "queries" to the input bits, while any intermediate computation is considered as free. For many functions one can obtain large quantum speedups in the case where algorithms are allowed a constant small probability of error (bounded error). As the most famous example, Grover's algorithm [25] computes the n-bit OR function with $O(\sqrt{n})$ queries in the bounded-error mode, while any classical algorithm needs $\Omega(n)$ queries. The model of *exact quantum query*, where the algorithms must output the correct answer with certainty for every possible input, seems to be more intriguing [10,15,19]. It is much more difficult to come up with exact quantum algorithms that outperform classical deterministic algorithms.

In the exact quantum query complexity, it was recognized that the best quantum speedup for computing total functions was by a factor of 2 for many years [22]. In a breakthrough result, Ambainis has presented the first example of a Boolean function $f:\{0,1\}^n \to \{0,1\}$ for which exact quantum algorithms have superlinear advantage over classical deterministic algorithms [2]. Based on the results in [5,31], Ambainis *et al.* [4] have verified that exact quantum algorithms have a certain advantage for most Boolean functions.

Ambainis *et al.* [5] have developed optimal exact quantum algorithms for computing functions EXACT_n^k and THRESHOLD_n^k, which are to determine whether an n-bit string has Hamming weight exactly k and to determine whether an n-bit string has Hamming weight at least k. The complexity is as follows:

$Q_E(\text{EXACT}_n^k) = \max(k,n-k)$;

$Q_E(\text{THRESHOLD}_n^k) = \max(k,n-k+1)$.

If f is allowed to be a partial function, the Deutsch-Jozsa algorithm [19] proved that there can be an exponential separation between exact quantum and classical deterministic query complexity. Some generalizations [17,26,31,36] of the Deutsch-Jozsa problem were also investigated, and we will indicate them carefully if there exist relations to our results.

A general generalization of Deutsch-Jozsa problem is the following symmetric partial function:

$$\text{DJ}_n^k(x) = \begin{cases} 1, & \text{if } |x| = n/2, \\ 0, & \text{if } |x| \leqslant k \text{ or } |x| \geqslant n-k, \end{cases} \tag{3}$$

where n is even and $0 \leqslant k < n/2$.

Clearly, when $k=0$, it is the Deutsch-Jozsa problem, and when $k=1$, it equals the problem given by Montanaro *et al.* [31].

III. DEGREE OF POLYNOMIALS FOR SYMMETRIC PARTIAL FUNCTIONS

First we study the exact degree of symmetric partial functions. We can use the method of symmetrization [32] to prove the following lemma.

Lemma 1. For any symmetric partial function f over $\{0,1\}^n$ with domain of definition D, suppose $\widetilde{\deg}_0(f) \leqslant d$. Then there exists a real multilinear polynomial q representing f and q that can be written as

$$q(x) = c_0 + c_1 V_1 + c_2 V_2 + \cdots + c_d V_d, \tag{4}$$

where $c_i \in \mathbf{R}$, $V_i = \Sigma_{j_1 j_2 \ldots j_i \in \{1,2,\ldots,n\}^i} x_{j_1} x_{j_2} \ldots x_{j_i}$ where any $j_1 j_2 \ldots j_i \in \{1,2,\ldots,n\}^i$ is without repeated number, $1 \leqslant i \leqslant d$, for example, $V_1 = x_1 + \cdots + x_n$, $V_2 = x_1x_2 + x_1x_3 + \cdots + x_{n-1}x_n$, etc.

Proof. Let p be a multilinear polynomial representing f and let $\deg(p) = \widetilde{\deg}_0(f) = d$. If π is some permutation and $x = (x_1,\ldots,x_n)$, then $\pi(x) = (x_{\pi(1)},\ldots,x_{\pi(n)})$. Let S_n be the set of all $n!$ permutations. For any $x \in \{0,1\}^n$, the symmetrization of p is

$$p^{\text{sym}}(x) = \frac{\sum_{\pi \in S_n} p(\pi(x))}{n!}. \tag{5}$$

Clearly, $0 \leqslant p(x) \leqslant 1$ implies $0 \leqslant p^{\text{sym}}(x) \leqslant 1$ for $x \in \{0,1\}^n$. Since f is symmetric, $x \in D$ implies $\pi(x) \in D$. For all $x \in D$, we have $f(\pi(x)) = f(x)$. Since p represents f, for any $x \in D$, we have $p(\pi(x)) = f(\pi(x)) = f(x) = p(x)$. Therefore, for any $x \in D$,

$$p^{\text{sym}}(x) = \frac{\sum_{\pi \in S_n} p(\pi(x))}{n!} = \frac{\sum_{\pi \in S_n} p(x)}{n!} = p(x) = f(x). \tag{6}$$

So p^{sym} can represent f. Let the multilinear polynomial $q = p^{\text{sym}}$. According to Minsky and Papert's result [32] (also Lemma 2 in [12]), q can be written as

$$q(x) = c_0 + c_1 V_1 + c_2 V_2 + \cdots + c_d V_d. \tag{7}$$

Therefore, the lemma has been proved. ■

Example 1. Let us give an example to find out $\widetilde{\deg}_0(f)$ for $f = \text{DJ}_n^0$, which is the Deutsch-Jozsa problem. To prove that $\widetilde{\deg}_0(\text{DJ}_n^0) \leqslant 2$, we assume that there is a multilinear polynomial $q(x) = c_0 + c_1 V_1 + c_2 V_2$ representing DJ_n^0. For $|x| = 0$, we have $q(x) = c_0 = f(x) = 0$. For $|x| = n$, we have $q(x) = \binom{n}{0}c_0 + \binom{n}{1}c_1 + \binom{n}{2}c_2 = \binom{n}{1}c_1 + \binom{n}{2}c_2 = 0$. For $|x| = \frac{n}{2}$, we have $q(x) = \binom{n/2}{0}c_0 + \binom{n/2}{1}c_1 + \binom{n/2}{2}c_2 = \binom{n/2}{1}c_1 + \binom{n/2}{2}c_2 = 1$. Therefore, we need to find out the solution of the following linear system of equations:

$$\begin{aligned} c_0 &= 0, \\ \binom{n}{0}c_0 + \binom{n}{1}c_1 + \binom{n}{2}c_2 &= 0, \\ \binom{n/2}{0}c_0 + \binom{n/2}{1}c_1 + \binom{n/2}{2}c_2 &= 1. \end{aligned} \tag{8}$$

It is easy to obtain that $c_0 = 0$, $c_1 = \frac{4(n-1)}{n^2}$, $c_2 = -\frac{8}{n^2}$, and $q(x) = \frac{4(n-1)}{n^2}V_1 - \frac{8}{n^2}V_2$ representing DJ_n^0. Therefore, $\widetilde{\deg}_0(\text{DJ}_n^0) \leqslant 2$.

Suppose that $\widetilde{\deg}_0(\text{DJ}_n^0) \leqslant 1$. Then there exists a multilinear polynomial $q(x) = c_0 + c_1 V_1$ representing DJ_n^0. We need to get the solution for the following linear group of equations:

$$\begin{aligned} c_0 &= 0, \\ \binom{n}{0}c_0 + \binom{n}{1}c_1 &= 0, \\ \binom{n/2}{0}c_0 + \binom{n/2}{1}c_1 &= 1. \end{aligned} \tag{9}$$

It is easy to deduce that there is no solution. Therefore, $\widetilde{\deg}_0(\text{DJ}_n^0) > 1$, and consequently $\widetilde{\deg}_0(\text{DJ}_n^0) = 2$. The example ends.

For any total function f, with the next lemma it has been proved [8] (or see [12]) that $Q_E(f) \geqslant \frac{1}{2}\deg(f)$.

Lemma 2. [8,12] Let $\mathcal{A}$ be a quantum query algorithm that makes t queries. Then there exist complex-valued n-variate multilinear polynomials α_i of degree at most t, such that the final state of $\mathcal{A}$ is

$$\sum_{i\in\{0,1\}^m} \alpha_i(x)|i\rangle \tag{10}$$

for every input $x \in \{0,1\}^n$.

For the case of bounded error, we have the following result.

Lemma 3. For any partial Boolean function f, $Q_\varepsilon(f) \geqslant \frac{1}{2}\widetilde{\deg}_\varepsilon(f)$, where $Q_\varepsilon(f)$ denotes the quantum query complexity for f with bounded-error ε.

Proof. Consider a $Q_\varepsilon(f)$-query quantum algorithm for f with error ε. Let S be the set of basis states corresponding to a one-output. Consider the polynomial $p(x) = \sum_{i\in S}|\alpha_i(x)|^2$, which is the probability that the algorithm outputs 1. If $x \in D$ and $f(x) = 1$, then $p(x) \geqslant 1-\varepsilon$. If $x \in D$ and $f(x) = 0$, then $p(x) \leqslant \varepsilon$. Therefore, $|p(x) - f(x)| \leqslant \varepsilon$ for all $x \in D$. Since the algorithm procedure to get the last state for any input x is the implementation of a sequence of unitary operators, it is clear that $0 \leqslant p(x) \leqslant 1$ for all $x \in \{0,1\}^n$. So polynomial $p(x)$ approximates f with error ε. According to Lemma 2, the α_i are polynomials of degree no more than $Q_\varepsilon(f)$, therefore $p(x)$ is a polynomial of degree no more than $2Q_\varepsilon(f)$. Consequently, we have

$$\widetilde{\deg}_\varepsilon(f) \leqslant \deg(p) \leqslant 2Q_\varepsilon(f), \tag{11}$$

and the lemma has been proved. ■

In particular, when $\varepsilon = 0$ we have the following special case.

Lemma 4. For any partial Boolean function f, $Q_E(f) \geqslant \frac{1}{2}\widetilde{\deg}_0(f)$.

We have proved that $\widetilde{\deg}_0(\mathrm{DJ}_n^0) = 2$. According to the above lemma, $Q_E(\mathrm{DJ}_n^0) \geqslant \frac{1}{2}\widetilde{\deg}_0(\mathrm{DJ}_n^0) = 1$. It is known that $Q_E(\mathrm{DJ}_n^0) \leqslant 1$ [19]. Therefore, we can use the above lemma to conclude $Q_E(\mathrm{DJ}_n^0) = 1$.

Now we deal with the case of approximating representation.

Lemma 5. For any symmetric partial Boolean function f over $\{0,1\}^n$ with domain of definition D, suppose $\widetilde{\deg}_\varepsilon(f) = d$. Then there exists a real multilinear polynomial q approximates f with error ε and q can be written as

$$q(x) = c_0 + c_1V_1 + c_2V_2 + \cdots + c_dV_d, \tag{12}$$

where $c_i \in \mathbb{R}$, $V_1 = x_1 + \cdots + x_n$, $V_2 = x_1x_2 + x_1x_3 + \cdots + x_{n-1}x_n$, etc.

Proof. The proof is similar to that of Lemma 1. For the readability, we outline it again. Let p be a multilinear polynomial with degree d that approximates f with error ε. The symmetrization of p is

$$p^{\mathrm{sym}}(x) = \frac{\sum_{\pi\in S_n} p(\pi(x))}{n!}. \tag{13}$$

If $x \in D$, then $|p(x) - f(x)| \leqslant \varepsilon$. Since f is symmetric, we have $|p^{\mathrm{sym}}(x) - f(x)| = |p(x) - f(x)| \leqslant \varepsilon$. Since $0 \leqslant p(\pi(x)) \leqslant 1$ for all $x \in \{0,1\}^n$, we have $0 \leqslant p^{\mathrm{sym}}(x) \leqslant 1$ for all $x \in \{0,1\}^n$. According to Minsky and Papert's result [32] (also Lemma 2 in [12]), p^{sym} can be written as

$$p^{\mathrm{sym}}(x) = c_0 + c_1V_1 + c_2V_2 + \cdots + c_dV_d. \tag{14}$$

Therefore, p^{sym} is the polynomial required. ■

It is important to determine the approximate degree of symmetric partial functions. The following lemma shows whether or not a symmetric partial function has degree d. The following lemma shows whether or not a symmetric partial function has approximate degree at most d.

Lemma 6. For any symmetric partial function f over $\{0,1\}^n$ with domain of definition D, and for fixed d and $0 \leqslant \varepsilon < 1/2$, there is an algorithm to discover whether or not there exists

$$q(x) = c_0 + c_1V_1 + c_2V_2 + \cdots + c_dV_d = \sum_{k=0}^{d} c_kV_k \tag{15}$$

approximating f with error ε.

Proof. Suppose that f is fully described by the vector $(b_0,b_1,\ldots,b_n) \in \{0,1,*\}^{n+1}$, where $f(x) = b_i$ for $|x| = i$. For input x, $V_k = \binom{|x|}{k}$. If there exists a polynomial q with degree d approximating f with error ε, then for $0 \leqslant i \leqslant n$, $q(x)$ satisfies the following inequalities and equalities:

(1) $0 \leqslant q(x) = \sum_{k=0}^{d} c_k\binom{i}{k} \leqslant \varepsilon$ if $b_i = 0$;

(2) $1-\varepsilon \leqslant q(x) = \sum_{k=0}^{d} c_k\binom{i}{k} \leqslant 1$ if $b_i = 1$;

(3) $0 \leqslant q(x) = \sum_{k=0}^{d} c_k\binom{i}{k} \leqslant 1$ if $b_i = *$.

Therefore, it suffices to verify whether the polyhedra has solution or not. It is easy to transfer the above polyhedra to the normal form, i.e., $S = \{c|Ac \leqslant h\}$, where the matrix $A \in \mathbb{R}^{2(n+1)\times(d+1)}$ and the vector $h \in \mathbb{R}^{2(n+1)}$. We now consider the following linear programming problem [9,29]:

$$\text{LP: Max } Z, \tag{16}$$

$$\text{s.t. } Ac + eZ \leqslant h, \tag{17}$$

$$Z \leqslant 0, \tag{18}$$

where $e \in \mathbb{R}^{2(n+1)}$ and $e^T = (1,1,\ldots,1)$. It is clear that $S \neq \emptyset$ if and only if the maximal value $Z^* = 0$. By using the method of liner programming and Karmarkar's algorithm [29], we can further transform the above linear programming problem into a standard form of linear programming problem and then obtain that the time complexity for finding a solution $c \in S$ or returning no solution is $O(d^2n^{5.5}\log^2 n)$ [35], and clearly its upper bound is $O(n^{7.5}\log^2 n)$ since $d \leqslant n$. ■

According to Lemma 5, determining $\widetilde{\deg}_\varepsilon(f)$ is equivalent to finding out the minimal d such that $q(x) = \sum_{k=0}^{d} c_kV_k$ approximates f with error ε.

Theorem 1. For any symmetric partial Boolean function f over $\{0,1\}^n$ with domain of definition D, and for the fixed $0 \leqslant \varepsilon < 1/2$, there exists an algorithm to find out $\widetilde{\deg}_\varepsilon(f)$ with an upper bound $O(n^{7.5}\log^3 n)$ on the time complexity.

Proof. This is Algorithm 1, and we analyze the process in detail. Let $\mathbf{b} = (b_0,b_1,\ldots,b_n)$ be the vector describing f. Let subroutine $\mathrm{S}(n,\mathbf{b},\varepsilon,d) = 1(0)$ if there does (not) exist polynomial q with degree d approximating f with error ε. The subroutine $\mathrm{S}(n,\mathbf{b},\varepsilon,d)$ can be done with the Karmarkar's algorithm according to Lemma 6. We give a binary search algorithm (Algorithm 1) to find out $\widetilde{\deg}_\varepsilon(f)$ as follows:

Algorithm 1 Algorithm for finding out $\widetilde{\deg}_\varepsilon(f)$

```
Procedure DEGREE (integer $n$, array b, real $\varepsilon$)    ▷ $\mathbf{b}\in\{0,1,*\}^{n+1}$
    integer $l := 0, r := n$;
    while $l \leqslant r$
        $d = \lfloor (l+r)/2 \rfloor$;
        if $S(n,\mathbf{b},\varepsilon,d) = 0$ then $l = d+1$;
        else $r = d-1$;
        end if
    end while
return $l$;
end procedure
```

In each iteration of the "while" loop, it holds that $S(n,\mathbf{b},\varepsilon,r+1) = 1$ and $S(n,\mathbf{b},\varepsilon,l-1) = 0$. We have $\widetilde{\deg}_\varepsilon(f) \leqslant r+1$ and $\widetilde{\deg}_\varepsilon(f) > l-1$. When the "while" loop is finished, we have that $l = r+1$ and $\widetilde{\deg}_\varepsilon(f) \leqslant r+1 = l$. Therefore, $\widetilde{\deg}_\varepsilon(f) = l$. The upper bound on time complexity is $O(\log n)O(n^{7.5}\log^2 n) = O(n^{7.5}\log^3 n)$ in terms of Lemma 6. ■

IV. GENERALIZED DEUTSCH-JOZSA PROBLEM

In this section we consider a generalized Deutsch-Jozsa problem DJ_n^k that was described by Eq. (3), that is, the problem of distinguishing between the inputs of Hamming weight $n/2$ and Hamming weights in the set $\{0,1,\ldots,k,n-k,n-k+1,\ldots,n\}$ for all even n with $0 \leqslant k < n/2$.

A. Exact quantum algorithm

By combining the exact quantum query algorithms for the functions EXACT and THRESHOLD by Ambainis *et al.* [5], in this subsection we give an exact quantum query algorithm for computing DJ_n^k.

Theorem 2. For even n, the exact quantum query complexity of DJ_n^k satisfies

$$Q_E\big(DJ_n^k\big) \leqslant k+1. \tag{19}$$

Proof. We will give an exact quantum algorithm using $k+1$ queries for DJ_n^k. One of the important subroutines that we will use in this paper is as follows:

Input: $x = x_1, x_2, \ldots, x_m$.

Output: If the output is (0,0) then $|x| \neq m/2$. Otherwise, it will output (i,j) such that $x_i \neq x_j$.

We call this subroutine Xquery. Let $x \in \{0,1\}^m$. If $\mathrm{Xquery}(m,x) = (0,0)$, then $|x| \neq m/2$. If $\mathrm{Xquery}(m,x) = (i,j)$, then $x_i \neq x_j$.

Indeed, according to [5] by Ambainis *et al.*, the subroutine Xquery can be implemented in one exact quantum query algorithm, and we put the details concerning the subroutine Xquery in Appendix A.

Based on the subroutine Xquery, now we give an algorithm (Algorithm 2) for DJ_n^k. It is clear that Algorithm 2 uses at most $k+1$ queries. ■

B. Lower bound of exact quantum query complexity

The purpose of this subsection is to prove that the exact quantum query complexity of DJ_n^k is no less than $k+1$, i.e., $Q_E(\mathrm{DJ}_n^k) \geqslant k+1$.

Theorem 3. For even n, the exact quantum query complexity of DJ_n^k satisfies

$$Q_E\big(DJ_n^k\big) \geqslant k+1. \tag{20}$$

Proof. We will prove that $\widetilde{\deg}_0(\mathrm{DJ}_n^k) \geqslant 2k+2$. Let us consider a simple case $k=1$ and $n \geqslant 6$ first. Suppose that $\widetilde{\deg}_0(\mathrm{DJ}_n^1) \leqslant 3$, according to Lemma 1, there exists a multilinear polynomial $q(x) = \sum_{i=0}^3 c_i V_i$ representing DJ_n^1. For $|x| = 0$, we have $q(x) = c_0 = f(x) = 0$. For $|x| = 1$, we have $q(x) = c_0 + \binom{1}{1}c_1 = f(x) = 0$ and therefore $c_1 = 0$. For $|x| = n, n-1, n/2$, we have the following equations:

$$\begin{aligned}\binom{n}{2}c_2 + \binom{n}{3}c_3 &= 0,\\ \binom{n-1}{2}c_2 + \binom{n-1}{3}c_3 &= 0,\\ \binom{n/2}{2}c_2 + \binom{n/2}{3}c_3 &= 1.\end{aligned} \tag{21}$$

Algorithm 2 Algorithm for DJ_n^k

```
procedure DJ (integer $n$, integer $k$, array $x$)    ▷ $x\in\{0,1\}^n$
    integer $l$:=1
    while $l \leqslant k$ do
        Output ← Xquery($n$,$x$)
        if Output=(0,0) then return 0
        end if
        if Output=(i, j) then
            $x \leftarrow x \setminus \{x_i, x_j\}$
            $l \leftarrow l+1$
            $n \leftarrow n-2$
        end if
    end while
    Output ← Xquery($n$,$x$)
    if Output=(0,0) then return 0
    end if
    if Output=(i, j) then return 1
    end if
end procedure
```

Let us consider the determinant

$$\begin{vmatrix} (n2) & (n3) \\ \binom{n-1}{2} & \binom{n-1}{3} \end{vmatrix} = \begin{vmatrix} \binom{n}{n-2} & \binom{n}{n-3} \\ \binom{n-1}{n-3} & \binom{n-1}{n-4} \end{vmatrix} \tag{22}$$

$$= \frac{1}{n}\begin{vmatrix} \binom{n}{n-2} & \binom{n}{n-3} \\ n\binom{n-1}{n-3} & n\binom{n-1}{n-4} \end{vmatrix} \tag{23}$$

$$= \frac{1}{n}\begin{vmatrix} \binom{n}{n-2} & \binom{n}{n-3} \\ (n-2)\binom{n}{n-2} & (n-3)\binom{n}{n-3} \end{vmatrix} \tag{24}$$

$$= \frac{1}{n}\begin{vmatrix} \binom{n}{n-2} & \binom{n}{n-3} \\ \binom{n}{n-2} & 0 \end{vmatrix} \neq 0. \tag{25}$$

Therefore, in order to satisfy the first two equations, we have $c_2 = c_3 = 0$. The last equation will not hold, which means that such q does not exist. Thus, $\widetilde{\deg}_0(\mathrm{DJ}_n^1) \geqslant 4$. According the Lemma 4, we have $Q_E(\mathrm{DJ}_n^1) \geqslant \frac{1}{2}\widetilde{\deg}_0(\mathrm{DJ}_n^1) \geqslant 2$. That is to say, the algorithm in Theorem 2 for DJ_n^1 is optimal. The algorithm in [31] for DJ_n^1 is also optimal.

Now we consider for the general case. Suppose that $\widetilde{\deg}_0(\mathrm{DJ}_n^k) \leqslant 2k+1$, according to Lemma 1, there exists a multilinear polynomial $q(x) = \sum_{i=0}^{2k+1} c_i V_i$ representing DJ_n^k. For $0 \leqslant |x| \leqslant k$, $f(x) = 0$. Therefore, we have $c_0 = c_1 = \cdots = c_k = 0$. For $|x| = n, n-1, \ldots, n-k$, we have the following equations:

$$\begin{aligned}
&\binom{n}{k+1}c_{k+1} + \binom{n}{k+2}c_{k+2} + \cdots + \binom{n}{2k+1}c_{2k+1} = 0,\\
&\binom{n-1}{k+1}c_{k+1} + \binom{n-1}{k+2}c_{k+2} + \cdots + \binom{n-1}{2k+1}c_{2k+1} = 0,\\
&\cdots\\
&\binom{n-k}{k+1}c_{k+1} + \binom{n-k}{k+2}c_{k+2} + \cdots + \binom{n-k}{2k+1}c_{2k+1} = 0.
\end{aligned} \tag{26}$$

Let us consider the determinant (see Appendix B for the detailed proof):

$$\begin{vmatrix}
\binom{n}{k+1} & \binom{n}{k+2} & \cdots & \binom{n}{2k+1}\\
\binom{n-1}{k+1} & \binom{n-1}{k+2} & \cdots & \binom{n-1}{2k+1}\\
\vdots & \vdots & \ddots & \vdots\\
\binom{n-k}{k+1} & \binom{n-k}{k+2} & \cdots & \binom{n-k}{2k+1}
\end{vmatrix}
= (-1)^{\frac{k(k+5)}{2}} \cdot \frac{\prod_{i=k+1}^{2k+1}\binom{n}{i}}{\prod_{i=1}^{k}\binom{n}{i}} \neq 0. \tag{27}$$

Therefore, we have $c_{k+1} = \cdots = c_{2k+1} = 0$. Then for $|x| = n/2$, $f(x) = q(x) = 0$, which is a contradiction. Therefore, $\widetilde{\deg}_0(\mathrm{DJ}_n^k) \geqslant 2k+2$ and $Q_E(\mathrm{DJ}_n^k) \geqslant \frac{1}{2}\widetilde{\deg}_0(\mathrm{DJ}_n^k) \geqslant k+1$. ■

C. Exact classical query complexity

Theorem 4 For even n, the classical deterministic query complexity of DJ_n^k satisfies

$$D(DJ_n^k) = n/2 + k + 1. \tag{28}$$

Proof. If the first $n/2$ queries return $x_i = 1$ and the next k queries return $x_i = 0$, then we will need to make another query as well. Therefore, $D(\mathrm{DJ}_n^k) \geqslant n/2 + k + 1$.

Now suppose that we have made $n/2 + k + 1$ queries. If no more than k queries return $x_i = 0$, then there are more than $n/2 + 1$ queries returning $x_i = 1$ and $\mathrm{DJ}_n^k(x) = 0$. If no more than k queries return $x_i = 1$, then there are more than $n/2 + 1$ queries returning $x_i = 0$ and $\mathrm{DJ}_n^k(x) = 0$. If there are more than k queries returning $x_i = 0$ and also more than k queries returning $x_i = 1$, then it must be balanced and $\mathrm{DJ}_n^k(x) = 1$. Therefore, $D(\mathrm{DJ}_n^k) \leqslant n/2 + k + 1$ and the theorem has been proved. ■

Remark 2. Again, we make some comparisons to the previous results. When $k = 0$, this is the Deutsch-Jozsa problem; when $k = 1$, this problem was considered by Montanaro *et al.* [31] and an exact quantum two-query algorithm was given to solve it, but the optimality was not verified. Also, the method in [31] is different (the unitary operator in their query algorithm was derived from distinguishing two orthogonal subsets of states).

So far, according to Theorems 2–4, our main result has been proved and is described as follows.

Theorem 5. For even n, the exact quantum query complexity of DJ_n^k satisfies

$$Q_E(DJ_n^k) = k + 1. \tag{29}$$

However, the classical deterministic query complexity for DJ_n^k is

$$D(DJ_n^k) = n/2 + k + 1. \tag{30}$$

Remark 3. When $k = 1$, Montanaro *et al.* [31] designed an exact quantum two-query algorithm to compute it, and their method is to derive a unitary operator from solving a system of equations, but the optimality with two-query was not verified. Our result also implies that the algorithm by Montanaro *et al.* [31] is optimal.

V. CONCLUSIONS

Symmetric Boolean functions are important in cryptography and property testing[13,14,18] due to their characters (for example, their outputs only depend on the Hamming weights of their inputs). Symmetric partial Boolean functions are symmetric but their domains of definition are allowed to be partial, so they also belong to the promise problems [21,23], for example, the Deutsch-Jozsa problem as a typical promise problem can be represented by an symmetric partial Boolean function. In this article, we have proved the exact quantum query complexity for a generalized Deutsch-Jozsa problem, and presented a method for determining the degree of symmetric partial Boolean functions. This method may be useful for further studying the quantum query complexity of symmetric partial Boolean functions.

ACKNOWLEDGMENTS

The authors are grateful to the two anonymous referees for their important suggestions that helped us improve the quality of the manuscript. Also, the first author thanks G. Xu, Z. Wu, X. Zhou, and S. Du for checking the time complexity of determining the approximate degree of symmetric partial Boolean functions, and we thank L. Li for discussion on query complexity. This work was supported in part by the National Natural Science Foundation of China (Grants No. 61572532, No. 61272058, No. 61602532), the Natural Science Foundation of Guangdong Province of China (Grant No. 2017B030311011), the Fundamental Research Funds for the Central Universities of China (Grant No. 17lgjc24), and D.Q. was also funded by FCT Project No. UID/EEA/50008/2013.

APPENDIX A: SUBROUTINE FOR XQUERY

The subroutine will use basis state $|0,0\rangle$, $|i,0\rangle$, and $|i,j\rangle$ with $1 \leqslant i < j \leqslant m$.

(1) The subroutine Xquery begins in the state $|0,0\rangle$ and then a unitary mapping U_1 is applied on it:

$$U_1|0,0\rangle = \sum_{i=1}^{m} \frac{1}{\sqrt{m}}|i,0\rangle. \tag{A1}$$

(2) The subroutine Xquery then performs the query

$$\sum_{i=1}^{m} \frac{1}{\sqrt{m}}|i,0\rangle \to \sum_{i=1}^{m} \frac{1}{\sqrt{m}}(-1)^{x_i}|i,0\rangle. \tag{A2}$$

(3) The subroutine Xquery performs a unitary mapping U_2 to the current state such that

$$U_2|i,0\rangle = \sum_{j>i} \frac{1}{\sqrt{m}}|i,j\rangle - \sum_{j<i} \frac{1}{\sqrt{m}}|j,i\rangle + \frac{1}{\sqrt{m}}|0,0\rangle \tag{A3}$$

and the resulting quantum state will be

$$U_2 \sum_{i=1}^{m} \frac{1}{\sqrt{m}}(-1)^{x_i}|i,0\rangle = \frac{1}{m}\sum_{i=1}^{m}(-1)^{x_i}|0,0\rangle + \frac{1}{m}\sum_{1\leqslant i<j}[(-1)^{x_i} - (-1)^{x_j}]|i,j\rangle. \tag{A4}$$

(4) The subroutine Xquery measures the resulting state in the standard basis. If the outcome is $|0,0\rangle$, then $\sum_{i=1}^{m}(-1)^{x_i} \neq 0$ and $|x| \neq m/2$. Otherwise, suppose that we get the state $|i,j\rangle$. Then we have $x_i \neq x_j$ and the subroutine outputs (i,j).

APPENDIX B: PROOF OF EQUALITY (27)

Proof. We define $\binom{p}{l} = 0$ if $p < l$ and also $\binom{p}{l} = 0$ if $l < 0$. For any integers p and l, it is easy to see that $\binom{p}{l} = \binom{p}{p-l}$. Now we prove that for any integers p and l,

$$(p+1)\binom{p}{l} = (l+1)\binom{p+1}{l+1}. \tag{B1}$$

There are several cases as follows.

Case 1: $p < l$. In this case, $\binom{p}{l} = 0$ and $\binom{p+1}{l+1}$, the equality holds.

Case 2: $p \geqslant l \geqslant 0$. In this case, $(p+1)\binom{p}{l} = (p+1)\frac{p!}{l!(p-l)!} = (l+1)\frac{(p+1)!}{(l+1)![(p+1)-(l+1)]!} = (l+1)\binom{p+1}{l+1}$.

Case 3: $p \geqslant l$ and $l < -1$. In this case, $\binom{p+1}{l+1} = 0$ and $\binom{p}{l} = 0$, the equality holds.

Case 4: $p \geqslant l$ and $l = -1$. In this case, $\binom{p}{l} = 0$ and $(l+1)\binom{p+1}{l+1} = 0$, the equality holds.

Therefore, the equality holds.

Now we are ready to prove equality (27):

$$\begin{vmatrix} \binom{n}{k+1} & \binom{n}{k+2} & \cdots & \binom{n}{2k} & \binom{n}{2k+1} \\ \binom{n-1}{k+1} & \binom{n-1}{k+2} & \cdots & \binom{n-1}{2k} & \binom{n-1}{2k+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \binom{n-k+1}{k+1} & \binom{n-k+1}{k+2} & \cdots & \binom{n-k+1}{2k} & \binom{n-k+1}{2k+1} \\ \binom{n-k}{k+1} & \binom{n-k}{k+2} & \cdots & \binom{n-k}{2k} & \binom{n-k}{2k+1} \end{vmatrix} = \begin{vmatrix} \binom{n}{n-k-1} & \binom{n}{n-k-2} & \cdots & \binom{n}{n-2k} & \binom{n}{n-2k-1} \\ \binom{n-1}{n-k-2} & \binom{n-1}{n-k-3} & \cdots & \binom{n-1}{n-2k-1} & \binom{n-1}{n-2k-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \binom{n-k+1}{n-2k} & \binom{n-k+1}{n-2k-1} & \cdots & \binom{n-k+1}{n-3k-1} & \binom{n-k+1}{n-3k} \\ \binom{n-k}{n-2k-1} & \binom{n-k}{n-2k-2} & \cdots & \binom{n-k}{n-3k} & \binom{n-k}{n-3k-1} \end{vmatrix}$$

$$= \frac{1}{(n-k+1)} \times \begin{vmatrix} \binom{n}{n-k-1} & \binom{n}{n-k-2} & \cdots & \binom{n}{n-2k} & \binom{n}{n-2k-1} \\ \binom{n-1}{n-k-2} & \binom{n-1}{n-k-3} & \cdots & \binom{n-1}{n-2k-1} & \binom{n-1}{n-2k-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \binom{n-k+1}{n-2k} & \binom{n-k+1}{n-2k-1} & \cdots & \binom{n-k+1}{n-3k-1} & \binom{n-k+1}{n-3k} \\ (n-k+1)\binom{n-k}{n-2k-1} & (n-k+1)\binom{n-k}{n-2k-2} & \cdots & (n-k+1)\binom{n-k}{n-3k} & (n-k+1)\binom{n-k}{n-3k-1} \end{vmatrix}$$

$$
= \frac{1}{(n-k+1)} \times \begin{vmatrix} \binom{n}{n-k-1} & \binom{n}{n-k-2} & \cdots & \binom{n}{n-2k} & \binom{n}{n-2k-1} \\ \binom{n-1}{n-k-2} & \binom{n-1}{n-k-3} & \cdots & \binom{n-1}{n-2k-1} & \binom{n-1}{n-2k-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \binom{n-k+1}{n-2k} & \binom{n-k+1}{n-2k-1} & \cdots & \binom{n-k+1}{n-3k-1} & \binom{n-k+1}{n-3k} \\ (n-2k-1)\binom{n-k+1}{n-2k} & (n-2k-2)\binom{n-k+1}{n-2k-1} & \cdots & (n-3k)\binom{n-k+1}{n-3k-1} & (n-3k-1)\binom{n-k+1}{n-3k} \end{vmatrix}
$$

$$
\xlongequal{r_{k+1}-(n-3k-1)r_k} \frac{1}{(n-k+1)} \times \begin{vmatrix} \binom{n}{n-k-1} & \binom{n}{n-k-2} & \cdots & \binom{n}{n-2k} & \binom{n}{n-2k-1} \\ \binom{n-1}{n-k-2} & \binom{n-1}{n-k-3} & \cdots & \binom{n-1}{n-2k-1} & \binom{n-1}{n-2k-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \binom{n-k+1}{n-2k} & \binom{n-k+1}{n-2k-1} & \cdots & \binom{n-k+1}{n-3k-1} & \binom{n-k+1}{n-3k} \\ k\binom{n-k+1}{n-2k} & (k-1)\binom{n-k+1}{n-2k-1} & \cdots & \binom{n-k+1}{n-3k-1} & 0 \end{vmatrix} \tag{B2}
$$

$$
\xlongequal{\cdots} \frac{1}{(n-k+1)(n-k+2)\cdots n} \times \begin{vmatrix} \binom{n}{n-k-1} & \binom{n}{n-k-2} & \cdots & \binom{n}{n-2k} & \binom{n}{n-2k-1} \\ k\binom{n}{n-k-1} & (k-1)\binom{n}{n-k-2} & \cdots & \binom{n}{n-2k} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ k\binom{n-k+2}{n-2k+1} & (k-1)\binom{n-k+2}{n-2k} & \cdots & \binom{n-k+2}{n-3k} & 0 \\ k\binom{n-k+1}{n-2k} & (k-1)\binom{n-k+1}{n-2k-1} & \cdots & \binom{n-k+1}{n-3k-1} & 0 \end{vmatrix}
$$

$$
= \frac{k(k-1)\cdots 1}{(n-k+1)(n-k+2)\cdots n} \times \begin{vmatrix} \frac{1}{k}\binom{n}{n-k-1} & \frac{1}{k-1}\binom{n}{n-k-2} & \cdots & \frac{1}{1}\binom{n}{n-2k} & \binom{n}{n-2k-1} \\ \binom{n}{n-k-1} & \binom{n}{n-k-2} & \cdots & \binom{n}{n-2k} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \binom{n-k+2}{n-2k+1} & \binom{n-k+2}{n-2k} & \cdots & \binom{n-k+2}{n-3k} & 0 \\ \binom{n-k+1}{n-2k} & \binom{n-k+1}{n-2k-1} & \cdots & \binom{n-k+1}{n-3k-1} & 0 \end{vmatrix}
$$

$$
= \frac{1}{\binom{n}{k}} \times \begin{vmatrix} \frac{1}{k}\binom{n}{n-k-1} & \frac{1}{k-1}\binom{n}{n-k-2} & \cdots & \frac{1}{1}\binom{n}{n-2k} & \binom{n}{n-2k-1} \\ \binom{n}{n-k-1} & \binom{n}{n-k-2} & \cdots & \binom{n}{n-2k} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \binom{n-k+2}{n-2k+1} & \binom{n-k+2}{n-2k} & \cdots & \binom{n-k+2}{n-3k} & 0 \\ \binom{n-k+1}{n-2k} & \binom{n-k+1}{n-2k-1} & \cdots & \binom{n-k+1}{n-3k-1} & 0 \end{vmatrix}
$$

$$
= (-1)^{k+2} \cdot \frac{\binom{n}{n-2k-1}}{\binom{n}{k}} \times \begin{vmatrix} \binom{n}{n-k-1} & \binom{n}{n-k-2} & \cdots & \binom{n}{n-2k} \\ \vdots & \vdots & \ddots & \vdots \\ \binom{n-k+2}{n-2k+1} & \binom{n-k+2}{n-2k} & \cdots & \binom{n-k+2}{n-3k} \\ \binom{n-k+1}{n-2k} & \binom{n-k+1}{n-2k-1} & \cdots & \binom{n-k+1}{n-3k-1} \end{vmatrix}
$$

$$
\xlongequal{\cdots} (-1)^{k+2}(-1)^{k+1}\cdots(-1)^{3} \times \frac{\binom{n}{n-2k-1}\binom{n}{n-2k}\cdots\binom{n}{n-k-2}}{\binom{n}{k}\binom{n}{k-1}\cdots\binom{n}{1}} \left|\binom{n}{n-k-1}\right|
$$

$$
= (-1)^{\frac{k(k+5)}{2}} \cdot \frac{\prod_{i=k+1}^{2k+1}\binom{n}{i}}{\prod_{i=1}^{k}\binom{n}{i}} \neq 0.
$$

■

[1] A. Ambainis, Inf. Process. Lett. **71**, 5 (1999).
[2] A. Ambainis, SIAM J. Comput. **45**, 617 (2016).
[3] A. Ambainis, K. Balodis, A. Belovs, T. Lee, M. Santha, and J. Smotrovs, in *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing* (ACM, New York, 2016), p. 800.
[4] A. Ambainis, J. Gruska, and S. Zheng, Quantum Inf. Comput. **15**, 0435 (2015).
[5] A. Ambainis, A. Iraids, and J. Smotrovs, arXiv:1302.1235.
[6] A. Ambainis and A. Yakaryilmaz, Inf. Process. Lett. **112**, 289 (2012).
[7] D. Bera, Quantum Inf. Process. **14**, 1777 (2015).
[8] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf, J. ACM **48**, 778 (2001).
[9] S. Boyd and L. Vandenberghe, *Convex Optimization* (Cambridge University Press, Cambridge, UK, 2004).
[10] G. Brassard and P. Høyer, in *Proceedings of the Fifth Israeli Symposium on Theory of Computing and Systems*, Ramat-Gan, 1997 (IEEE, Los Alamitos, CA, 1997).
[11] H. Barnum, M. Saks, and M. Szegedy, in *Proceedings 18th IEEE Annual Conference on Computational Complexity* (IEEE, Aarhus, Denmark, 2003), p. 179.
[12] H. Buhrman and R. de Wolf, Theor. Comput. Sci. **288**, 21 (2002).
[13] E. Blais, A. Weinstein, and Y. Yoshida, SIAM J. Comput. **44**, 411 (2015).
[14] A. Childs and W. van Dam, Rev. Mod. Phys. **82**, 1 (2010).
[15] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, in Proc. R. Soc. London, Ser. A **454A**, 339 (1998).
[16] D. Collins, K. W. Kim, and W. C. Holton, Phys. Rev. A **58**, R1633 (1998).
[17] D. P. Chi, J. Kim, and S. Lee, arXiv:quant-ph/0005059.
[18] A. Canteaut and M. Videau, IEEE T. Inform. Theory **51**, 2791 (2005).
[19] D. Deutsch and R. Jozsa, Proc. R. Soc. London, Ser. A **439**, 553 (1992).
[20] A. Dubrovska and T. Mischenko-Slatenkova, arXiv:quant-ph/0607022.
[21] S. Even, A. L. Selman, and Y. Yacobi, Inform. Control **61**, 159 (1984).
[22] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, Phys. Rev. Lett. **81**, 5442 (1998).
[23] O. Goldreich, *Lecture Notes in Computer Science* (Springer, Berlin, 2006), Vol. 3895.
[24] J. Gruska, D. W. Qiu, and S. G. Zheng, Int. J. Found. Comput. Sci. **26**, 381 (2015).
[25] L. K. Grover, in *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing* (ACM, New York, 1996), p. 212.
[26] J. Gruska, D. W. Qiu, and S. G. Zheng, Math. Struct. Comput. Sci. **27**, 311 (2017).
[27] T. Hayes, S. Kutin, and D. van Melkebeek, Algorithmica **34**, 480 (2002).
[28] P. Høyer and R. Špalek, Bull. Eur. Assoc. Theor. Comput. Sci. **87**, 78 (2005).
[29] N. Karmarkar, Combinatorica **4**, 373 (1984).
[30] G. Midrijānis, arXiv:quant-ph/0403168.
[31] A. Montanaro, R. Jozsa, and G. Mitchison, Algorithmica **419**, 775 (2015).
[32] M. Minsky and S. Papert, *Perceptrons: An Introduction to Computational Geometry* (MIT Press, Cambridge, UK, 1988).
[33] B. Perez-Garcia, M. McLaren, S. K. Goyal, R. I. Hernandez-Aranda, A. Forbes, and T. Konrad, Phys. Lett. A **380**, 1925 (2016).
[34] T. Mischenko-Slatenkova, A. Vasilieva, I. Kucevalovs, and R. Freivalds, *Lecture Notes in Computer Science* (Springer, Cham, 2015), Vol. 9118.
[35] G. L. Xu, Z. G. Wu, X. Zhou, and D. W. Qiu, *Determining the Approximate Degree of Symmetric Partial Boolean Functions*, technical report, Sun Yat-sen University, 2018.
[36] S. Zheng and D. W. Qiu, *Lecture Notes in Computer Science* (Springer, Cham, 2014), Vol. 8808.