entropy

MDPI

Article

Partial Boolean Functions with Exact Quantum Query Complexity One

Guoliang Xu [1,2] and Daowen Qiu [1,2,*]

[1] Institute of Quantum Computing and Computer Theory, School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China; xu1guo2liang@foxmail.com
[2] Guangdong Key Laboratory of Information Security Technology, Sun Yat-sen University, Guangzhou 510006, China
* Correspondence: issqdw@mail.sysu.edu.cn

Abstract: We provide two sufficient and necessary conditions to characterize any n-bit partial Boolean function with exact quantum query complexity 1. Using the first characterization, we present all n-bit partial Boolean functions that depend on n bits and can be computed exactly by a 1-query quantum algorithm. Due to the second characterization, we construct a function F that maps any n-bit partial Boolean function to some integer, and if an n-bit partial Boolean function f depends on k bits and can be computed exactly by a 1-query quantum algorithm, then $F(f)$ is non-positive. In addition, we show that the number of all n-bit partial Boolean functions that depend on k bits and can be computed exactly by a 1-query quantum algorithm is not bigger than an upper bound depending on n and k. Most importantly, the upper bound is far less than the number of all n-bit partial Boolean functions for all efficiently big n.

Keywords: quantum computation; quantum query complexity; quantum query algorithm; partial Boolean function

Citation: Xu, G.; Qiu, D. Partial Boolean Functions with Exact Quantum Query Complexity One. *Entropy* **2021**, *23*, 189. https://doi.org/10.3390/e23020189

Academic Editor: Vitaly Kocharovsky
Received: 9 November 2020
Accepted: 28 January 2021
Published: 3 February 2021

1. Introduction

In the field of theoretical computer science, computational complexity aims to measure "how much" computation is necessary and sufficient to finish some certain computational tasks. In classical computation, a simplest model of computation is the decision tree (For more details, we can refer to the survey paper [1]). Correspondingly, the quantum query model (quantum black box model, or quantum decision tree model) is a generalization of the decision tree model in quantum computation [1–5]. Most of famous quantum algorithms are captured by the quantum query model [6], such as Shor's factoring algorithm [7], Grover's unstructured search algorithm [8], and so on [9–11]. The quantum query model can be investigated in the exact setting and the bounded-error setting [1]. Given an input $x \in D \subseteq \{0,1\}^n$ that can only be accessed through a black box by querying some bit x_i of the input, the quantum query model computes an n-bit partial Boolean function $f : D \to \{0,1\}$ exactly (or with bounded-error) [1]. An exact quantum algorithm must always output the correct function value for all legal inputs [1]. If a quantum algorithm outputs the function value with a probability greater than a constant ($>\frac{1}{2}$) for all legal inputs, then the quantum algorithm is said to compute the function with bounded error. In the quantum query model, we care the quantum query complexity that is the decision tree complexity for the quantum model [1–3]. Roughly speaking, the exact (or bounded-error) quantum query complexity of a Boolean function denotes the number of queries of an optimal quantum decision tree that computes the Boolean function exactly (or with bounded-error) [1].

In quantum computation, the hope is to find out many problems whose computational complexity in quantum computer is less than the computational complexity in classical computer, i.e., finding out many problems that have the quantum advantage. For a function

f, quantum advantage can be investigated by comparing the exact quantum query complexity $Q_E(f)$ and the deterministic decision tree complexity $D(f)$ [1], where $D(f)$ denotes the minimum number of queries used by any classical deterministic algorithm. Over the past decade, there have been many results on the quantum query model [12–20]. In particular, Ambainis et al. [14] proved that exact quantum algorithms have advantage for almost all Boolean functions in 2015. For total Boolean functions (i.e., partial Boolean functions with $D = \{0,1\}^n$), the first known quantum speed-up was $Q_E(f) = O(D(f)^{0.8675\ldots})$ by Ambainis [13], and then Ambainis et al. [21] presented a better separation with a quadratic gap between the exact quantum query complexity and the deterministic decision tree complexity, up to polylogarithmic factors.

For any partial Boolean function, the best separation is still achieved by Deutsch-Jozsa algorithm [10,17]. And, some main related results are as follows. In 2007, Montanaro [22] considered a problem of exact oracle identification with a single quantum query. In 2015, Montanaro et al. [15] investigated all small total Boolean functions up to four bits and symmetric total Boolean functions up to six bits. In 2016, Qiu et al. [17,23] generalized Deutsch-Jozsa problem and gave its optimal exact quantum query algorithm, and in particular, Qiu et al. [23,24] presented all symmetric partial Boolean functions (it is a special class of partial Boolean functions) with exact quantum query complexity 1, and proved that any symmetric partial Boolean function f can be computed exactly by a 1-query quantum algorithm if and only if f can be computed by the Deutsch-Jozsa algorithm [10]. In the same year, Aaronson et al. [25] showed an equivalence between 1-query quantum algorithms and bounded quadratic polynomials in the bounded-error setting. Also in the same year, Grillo et al. [26] investigated partial Boolean functions which are computed exactly with t queries. However, the result in [26] (i.e., Theorem 5) for any numerical procedure is difficult to use as an analytic tool. In 2017, Arunachalam et al. [27] proved a characterization of t-query quantum algorithms in terms of the unit ball of a space of degree-$(2t)$ polynomials. Using the same method as the proof of Theorem 11 given by Qiu et al. [23,24], Chen et al. [28] showed that the total Boolean functions with exact quantum query complexity 1 are the one-bit function $f(x) = x_1$ and the two-bit function $x_1 \oplus x_2$, and Mukherjee et al. [29] noticed that this result of [28] is the same as a result by Montanaro et al. [15].

As above, the exact 1-query quantum model for all partial Boolean functions is expected to be investigated further. On one hand, similar to Refs. [23–27], establishing an equivalence between quantum query algorithms and some other theories will inspire more new algorithms and related results on the complexity theory. On the other hand, with the development of quantum computer, the problems solved by 1-query quantum algorithms may be the first to be used widely in the future, as the 1-query quantum algorithm costs the least unitary operators. Specifically, we investigate the following two problems.

- The partial Boolean functions can be regarded as a generalization of the total Boolean functions. Actually, Deutsch's algorithm [9] computes a two-bit partial (also total) Boolean function using one query. Both the extension of Deutsch's problem (computed by Deutsch-Jozsa algorithm [10]) and a generalized Deutsch-Jozsa problem in Ref. [17] are described by even n-bit partial (not total) Boolean functions. Naturally, what is the characterization of partial Boolean functions with exact quantum query complexity 1?
- In the field of quantum computation, it is a fundamental and interesting subject to evaluate the computational power of the 1-query quantum model, and is also critical for discovering quantum advantage. Specifically, the number of partial Boolean functions with exact quantum query complexity 1 shows the power and advantage of the exact 1-query quantum model. So, how many partial Boolean functions can be computed exactly by 1-query quantum algorithms?

The rest of the paper is organized as follows. In Section 2, we introduce some basis notations and the related knowledge. Then, we state and prove the first characterization and a related result in Section 3. Next, we state and prove the second characterization and two related results in Section 4. Finally, the conclusion is presented in Section 5.

For the sake of brevity and readability, all proofs of lemmas in this paper are showed in Appendices A–D.

2. Preliminaries

In this section, we introduce some basic notations and recall some basic knowledge of partial Boolean functions and the exact quantum query model. For the details, we can refer to Refs. [1,6,20,23,30,31].

As usual, notations $\mathbb{N}$, $\mathbb{R}$, and $\mathbb{C}$ denote the sets of natural numbers, real numbers, and complex numbers, respectively. In particular, we will always use the notation D (or promised set) to denote a subset of $\{0,1\}^n$. For any input $x = x_1x_2\cdots x_n \in D$, the Hamming weight (number of 1s) of x is denoted by $|x|$. Given a real number set S, the notation $\max S$ denotes the maximum in S and the notation $\min S$ denotes the minimum in S. For any finite set S, the notation $|S|$ denotes the number of elements in S. For a complex matrix A, A^T is the transpose of the matrix A, and $A^\dagger = (A^T)^*$ is the conjugate transpose of the matrix A. Obviously, $A^\dagger = A^T$ for any real matrix A. Furthermore, the notation $|a\rangle$ is usually used to denote a quantum state which is a unit vector in a Hilbert space and labeled by the notation a. In particular, $\langle a| = (|a\rangle)^\dagger$ is a row vector.

In this paper, we mainly concern partial functions $f : D \to \mathbb{C}$, $f : D \to \mathbb{R}$ and $f : D \to \{0,1\}$. In general, these functions can be given by a 2^n-dimensional vector $(f(0), f(1), \cdots, f(x), \cdots, f(2^n-1))^T$ whose entry $f(x)$ is denoted by $*$ for any undefined input $x \in \{0,1\}^n \backslash D$. For example, the Boolean function f computed by Deutsch's algorithm [9] can be given by $(f(00), f(01), f(10), f(11)) = (1,0,0,1)$. Sometimes, we also use a two-tuple $(\{x : f(x) = 0\}, \{y : f(y) = 1\})$ to give a certain partial Boolean function $f : D \to \{0,1\}$. For example, the even n-bit partial Boolean function f computed by Deutsch-Jozsa algorithm [10] can be given by $(\{x \in \{0,1\}^n : |x| = 0, n\}, \{y \in \{0,1\}^n : |y| = \frac{n}{2}\})$.

In order to represent these functions, we need to use two monomials $X_S = \prod_{i\in S} x_i$ and $(-1)^{S\cdot x} = \prod_{i\in S}(-1)^{x_i}$ [1,20,30]. In particular, $X_\emptyset = (-1)^{\emptyset\cdot x} = 1$. And, the set $\{X_S : S \subseteq \{1,2,\cdots,n\}\}$ is usually called the polynomial basis and the set $\{(-1)^{S\cdot x} : S \subseteq \{1,2,\cdots,n\}\}$ is usually called Fourier basis [1,20,30]. If a function $p : \mathbb{R}^n \to \mathbb{C}$ can be written as $\Sigma_S \alpha_S X_S$ for some complex numbers α_S, then the function p is called a multilinear polynomial [1]. Meanwhile, the degree of the multilinear polynomial p is defined by $\deg(p) = \max\{|S| : \alpha_S \neq 0\}$. For any partial function $f : D \to \mathbb{C}$, a multilinear polynomial $p(x)$ represents f if and only if $p(x) = f(x)$ for all $x \in D$ [1,32]. Unlike total functions $f : \{0,1\}^n \to \mathbb{C}$, the multilinear representation of a partial (not total) function $f : D \to \mathbb{C}$ is usually not unique. Naturally, the degree of a partial (or total) function $f : D \to \mathbb{C}$ can be defined by $\deg(f) = \min\{\deg(p) : p \text{ represents } f\}$.

In the quantum query model, for every input $x \in D$, the quantum black box O_x can be described as a unitary operator which is defined by

$$O_x|i,j\rangle = \begin{cases} (-1)^{x_i}|i,j\rangle, & \text{if } i \in \{1,2,\cdots,n\}, \\ |0,j\rangle, & \text{if } i = 0. \end{cases} \tag{1}$$

Here, the integer number $i \in \{0,1,2,\cdots,n\}$ is the query-part and the label j is the auxiliary space. As a result, a t-query quantum algorithm can be determined by an initial state $|\psi_0\rangle$ and a sequence of unitary transformations $U_0, O_x, U_1, O_x, \cdots, O_x, U_t$ followed by a measurement, where $t+1$ unitary operators $U_0, U_1, \cdots, U_t$ are independent of the input [1,6].

3. The First Characterization

This section introduces and proves the first characterization and a related result.

3.1. A Characterization by the Linear System of Equations

Inspired by proofs of Theorem 8 in Ref. [23], the first characterization is presented in the following. In fact, the characterization can also be got by Theorem 5 in [26].

Theorem 1. *An n-bit non-constant partial Boolean function $f : D \to \{0,1\}$ can be computed exactly by a 1-query quantum algorithm, if and only if there exist at least one non-negative solution $\mathbf{z} = (z_0, z_1, z_2, \cdots, z_n)^T$ of equations $z_0+z_1+z_2+\cdots+z_n=1$ and $\mathbf{F}^T(a \oplus b)\mathbf{z} = 0$ for all $a \in \{x : f(x) = 0\}$ and $b \in \{x : f(x) = 1\}$ where the function vector $\mathbf{F}(x) = (1, (-1)^{x_1}, (-1)^{x_2}, \cdots, (-1)^{x_n})^T$ for any $x = x_1 x_2 \cdots x_n \in \{0,1\}^n$.*

Proof. $\Rightarrow$). Since the algorithm is exact, the quantum state $U_1 O_a U_0|\psi_0\rangle$ for all $a \in \{x : f(x) = 0\}$ must be orthogonal to the quantum state $U_1 O_b U_0|\psi_0\rangle$ for all $b \in \{x : f(x) = 1\}$. Meanwhile, since the unitary operator U_1 preserves the inner product of any two complex vectors, the quantum state $O_a U_0|\psi_0\rangle$ for all $a \in \{x : f(x) = 0\}$ must be orthogonal to the quantum state $O_b U_0|\psi_0\rangle$ for all $b \in \{x : f(x) = 1\}$. For any quantum state $U_0|\psi_0\rangle = \sum_{i,j} \alpha_{i,j}|i,j\rangle$, note that

$$O_a U_0|\psi_0\rangle = \sum_j \alpha_{0,j}|0,j\rangle + \sum_{i,j} \alpha_{i,j}(-1)^{a_i}|i,j\rangle \tag{2}$$

and the inner product

$$\begin{aligned}
&(O_a U_0|\psi_0\rangle)^\dagger O_b U_0|\psi_0\rangle = \sum_j |\alpha_{0,j}|^2 + \sum_{i,j}(-1)^{a_i \oplus b_i}|\alpha_{i,j}|^2 \\
=&(\sum_j |\alpha_{0,j}|^2, \sum_j |\alpha_{1,j}|^2, \cdots, \sum_j |\alpha_{n,j}|^2)(1, (-1)^{a_1 \oplus b_1}, \cdots, (-1)^{a_n \oplus b_n})^T \\
=&(z_0, z_1, \cdots, z_n)\mathbf{F}(a \oplus b) = \mathbf{z}^T \mathbf{F}(a \oplus b)
\end{aligned} \tag{3}$$

where the notation $z_i = \sum_j |\alpha_{i,j}|^2$ for all $i \in \{0,1,2,\cdots,n\}$ is introduced in the last equality. Thus, there exists at least one non-negative solution $\mathbf{z} = (z_0, z_1, z_2, \cdots, z_n)^T$ of equations $z_0+z_1+z_2+\cdots+z_n=1$ and $\mathbf{z}^T\mathbf{F}(a \oplus b) = 0$ for all $a \in \{x : f(x) = 0\}$ and $b \in \{x : f(x) = 1\}$.

$\Leftarrow$). For a non-negative solution $\mathbf{z} = (z_0, z_1, z_2, \cdots, z_n)^T$ of equations $z_0+z_1+z_2+\cdots+z_n=1$ and $\mathbf{z}^T\mathbf{F}(a \oplus b) = 0$ for all $a \in \{x : f(x) = 0\}$ and $b \in \{x : f(x) = 1\}$, if we set $\sum_j |\alpha_{i,j}|^2 = z_i$ for all $i \in \{0,1,2,\cdots,n\}$ in the state $U_0|\psi_0\rangle = \sum_{i,j} \alpha_{i,j}|i,j\rangle$ of an undetermined 1-query quantum algorithm, then the inner product

$$(U_1 O_a U_0|\psi_0\rangle)^\dagger U_1 O_b U_0|\psi_0\rangle = 0 \tag{4}$$

for all $a \in \{x : f(x) = 0\}$ and $b \in \{x : f(x) = 1\}$. By Gram-Schmidt orthogonalization, we can find an orthonormal basis $\{U_1 O_a U_0|\psi_0\rangle : f(a) = 0, a \in D\}$ and an orthonormal basis $\{U_1 O_b U_0|\psi_0\rangle : f(b) = 1, b \in D\}$, respectively. By the measurement consisting of the two orthonormal bases, the 1-query quantum algorithm (with the state $U_0|\psi_0\rangle = \sum_{i,j} \alpha_{i,j}|i,j\rangle$) computes f exactly. Thus, Theorem 1 has been proved. □

Discussions on Theorem 1

Theorem 1 transforms the problem of deciding the partial Boolean function with exact quantum query complexity 1 into the problem of solving a linear system of equations. Specifically, the number of variables in the linear system is $n + 1$ and the number of equations is $1 + |\{a \oplus b : f(a) = 0, f(b) = 1\}| \leq 2^n$.

Considering the definition of the exact quantum query complexity, the task of proving $Q_E(f) = k$ must be finished by presenting an exact k-query quantum algorithm. For a small n, presenting an optimal exact quantum query algorithm is still quite hard in some cases. For $k = 1$, Theorem 1 transforms this task to the problem of solving a linear system which is a computable task. For any non-constant n-bit partial Boolean function $f : D \to \{0,1\}$, if $|D|$ is not big, then this task can be done efficiently in a classical computer. This is the most important contribution of Theorem 1.

In fact, Theorem 1 is also a practical tool in some cases. In the worst case, the number of equations in the linear system is $1 + |\{a \oplus b : f(a) = 0, f(b) = 1\}|$ which is an exponential

number. Note that the number of variables in the linear system is $n+1$. So, the following two results can be got.

(1) Given any n-bit partial Boolean function f, if some equations (By basic linear algebra, $n+1$ equations are enough in the best case) lead to an empty (non-negative real) solution of the linear system, then by Theorem 1 these equations are enough to prove that the exact quantum query complexity of f is bigger than 1. Thus, for the best case, other $|\{a \oplus b : f(a) = 0, f(b) = 1\}| - n$ equations can be ignored and things become quite easy.

(2) If the exact quantum query complexity of an n-bit partial Boolean function $f : D \to \{0,1\}$ is bigger than 1, then the exact quantum query complexity of any n-bit partial Boolean function g defined by

$$\begin{cases} g(x) = f(x), & \forall x \in D, \\ g(x) \in \{0,1,*\}, & \text{otherwise} \end{cases} \tag{5}$$

is also bigger than 1. The number of partial Boolean functions in this form is $3^{2^n-|D|}$ which is also an exponential number.

3.2. Partial Boolean Functions Depending on All Bits

This subsection considers a special class of all partial Boolean functions using Theorem 1.

First, we introduce the following definition [14,31] (A background of this definition is introduced briefly in Appendix A).

Definition 1. *[14,31]. An n-bit partial Boolean function $f : D \to \{0,1\}$ is said to depend on k ($\leq n$) bits, if k is the minimum number of variables in all multilinear polynomials representing f.*

By Definition 1, the two-bit total Boolean function computed by Deutsch's algorithm [9] depends on two bits, and, for even n, the n-bit partial Boolean function computed by Deutsch-Jozsa algorithm [10] depends on $\frac{n}{2}+1$ bits.

For partial Boolean functions depending on all bits, the result is stated in the following.

Theorem 2. *For any n-bit partial Boolean function $f : D \to \{0,1\}$ depending on all n bits, f can be computed exactly by a 1-query quantum algorithm, if and only if $f(x) = x_1$ or $1 \oplus x_1$ with $D = \{0,1\}$, or $f(x) = x_1 \oplus x_2$ or $1 \oplus x_1 \oplus x_2$ with $D \in \{E \subseteq \{0,1\}^2 : |E| \in \{3,4\}\}$.* □

Proof. $\Rightarrow$). For any n-bit partial Boolean function $f : D \to \{0,1\}$ and $k \in \{1,2,\cdots,n\}$, any multilinear polynomial representation of f can be written as

$$f(x) = x_k q_1(x_1, x_2, \cdots, x_{k-1}, x_{k+1}, \cdots, x_n) + q_2(x_1, x_2, \cdots, x_{k-1}, x_{k+1}, \cdots, x_n) \tag{6}$$

where $q_1(x_1, x_2, \cdots, x_{k-1}, x_{k+1}, \cdots, x_n)$ and $q_2(x_1, x_2, \cdots, x_{k-1}, x_{k+1}, \cdots, x_n)$ are two multilinear polynomials on variables $x_1, x_2, \cdots, x_{k-1}, x_{k+1}, \cdots, x_n$ (i.e., not on the variable x_k). Let the input $X_k^{\{k\}}$ be the same as the input X_k except for the k-th bit being flipped. With an argument, if f depends on n bits (By Definition 1, this means that the number of variables in all multilinear polynomials representing f is at least n), then there must exist at least n input pairs $(X_1, X_1^{\{1\}})$, $(X_2, X_2^{\{2\}})$, $\cdots$, $(X_n, X_n^{\{n\}})$ such that $1 \oplus f(X_k) = f(X_k^{\{k\}}) \in \{0,1\}$ for all $k \in \{1,2,\cdots,n\}$ (In fact, for a certain k, if $f(X) = f(X^{\{k\}})$ always hold for any $X \in D$, then the polynomial $q_1(x_1, x_2, \cdots, x_{k-1}, x_{k+1}, \cdots, x_n) = 0$ and $f(x) = q_2(x_1, x_2, \cdots, x_{k-1}, x_{k+1}, \cdots, x_n)$ depends on at most $n-1$ bits. This result contradicts the assumption that f depends on n bits).

By Theorem 1, there exists a non-negative vector $\mathbf{z} = (z_0, z_1, z_2, \cdots, z_n)^T$ such that the equations $z_0+z_1+z_2+\cdots+z_n=1$ and

$$\mathbf{z}^T\mathbf{F}(X_k \oplus X_k^{\{k\}}) = 0 = z_0 + \sum_{i\neq k} z_i - z_k, k \in \{1,2,\cdots,n\} \tag{7}$$

hold. Combining with $z_0 + z_1 + z_2 + \cdots + z_n = 1 = z_0 + \sum_{i\neq k} z_i + z_k$, we have $2z_k = 1$ for all $k \in \{1,2,\cdots,n\}$ which implies that $n = 1$ with $\mathbf{z} = (\frac{1}{2}, \frac{1}{2})^T$ or $n = 2$ with $\mathbf{z} = (0, \frac{1}{2}, \frac{1}{2})^T$.

The case $n = 1$ is trivial, and f can be given by $(f(0), f(1)) = (0,1)$ or $(1,0)$. For the case $n = 2$, the unique non-negative solution $\mathbf{z} = (0, \frac{1}{2}, \frac{1}{2})^T$ implies that

$$\frac{1}{2}(-1)^{a_1\oplus b_1} + \frac{1}{2}(-1)^{a_2\oplus b_2} = 0 \tag{8}$$

for all $a \in \{x : f(x) = 0\}$ and $b \in \{x : f(x) = 1\}$. Then, $a_1 \oplus a_2 \neq b_1 \oplus b_2$ for all $a \in \{x : f(x) = 0\}$ and $b \in \{x : f(x) = 1\}$. This result implies that $f(x) = x_1 \oplus x_2$ or $1 \oplus x_1 \oplus x_2$. Meanwhile, since $f : D \to \{0,1\}$ is a two-bit partial Boolean function depending on two bits, $|D| \in \{3,4\}$.

$\Leftarrow$). This direction is trivial. Thus, Theorem 2 has been proved. □

Discussions on Theorem 2

Since there are many partial Boolean functions with exact quantum query complexity 1, it is necessary to divide them into some classes. In 2015, Montanaro et al. [15] investigated all small total Boolean functions up to four bits and symmetric total Boolean functions up to six bits. In 2016, Qiu et al. [23,24] studied all symmetric partial Boolean functions and remained others open. By Definition 1, it is natural to divide all n-bit partial Boolean functions into $n + 1$ classes.

For all n-bit partial Boolean functions depending on 1 and 2 bits, the problem is trivial. For n-bit partial Boolean functions depending on n bits, the result is put into Theorem 2. By a trivial (not direct) argument, the statement that a partial Boolean function f depends on all n bits is consistent with previous definition. This is a key hint in the proof. Intuitively, this implication seems obvious. However, since the implication does not hold for $\frac{n}{2}$, we remark that a proof is necessary.

Theorem 2 clarifies all partial Boolean functions that depend on all bits and can be computed exactly by a 1-query quantum algorithm. Observing the unique multilinear polynomial of any total Boolean function, any n-bit total Boolean function depending on k ($\leq n$) bits can be identified with a k-bit total Boolean function depending on k bits. Therefore, Theorem 2 generalizes the result on total Boolean functions [15] to partial Boolean functions. Surprisingly, the number of all n-bit partial Boolean functions depending on n bits is quite big. This fact is implied by the following lemma.

Lemma 1. *Let $N(n)$ $(n \geq 1)$ denote the number of all n-bit partial Boolean functions depending on n bits. Then, $N(n) \geq 2 \times 3^{2^n-n-1}$.*

In contrast, the number of all n-bit total Boolean functions (investigated by Montanaro et al. [15]) is 2^{2^n} and the number of all n-bit symmetric partial Boolean functions (investigated by Qiu et al. [23,24]) is 3^{n+1}.

As a result, many n-bit partial Boolean functions depending on $k \in \{3,\cdots,n-1\}$ bits is sitll unclear. This motivate us to investigate partial Boolean functions further.

4. The Second Characterization

This section introduces and proves the second characterization and two related results.

4.1. A Characterization by the Sum-of-Squares Representation

Following the discussions of Lemma 7 and Theorem 17 in [1], if $f : D \to \{0,1\}$ can be computed by an exact 1-query quantum algorithm, then there exist degree-1 SOS complex representations of f and $1-f$. Thus, this subsection introduces and proves a characterization using the SOS representation.

First, the following lemma follows the discussions of Lemma 7 and Theorem 17 in [1]. This lemma is proved and also used in our recent paper [33].

Lemma 2. *[1]. If there exists an exact 1-query quantum algorithm computing an n-bit partial Boolean function $f : D \to \{0,1\}$, then there must exist degree-1 SOS complex (multilinear polynomials) representations of f and $1-f$.*

In order to give the second characterization, we introduce the following definition. This definition is also used in [33]. More related definitions can be seen in Refs. [34–39].

Definition 2. *[34–39]. Let the function vector $\mathbf{F}(x) = (1,(-1)^{x_1},(-1)^{x_2},\cdots,(-1)^{x_n})^T$. For an n-bit partial Boolean function $f : D \to \{0,1\}$, if there exist two real $(1+n)$-dimensional column vector sets $\{\mathbf{a}_1,\mathbf{a}_2,\cdots,\mathbf{a}_p\}$ and $\{\mathbf{a}_{p+1},\mathbf{a}_{p+2},\cdots,\mathbf{a}_{p+q}\}$ such that the equations*

$$\begin{cases} f(x) = \sum_{l=1}^{p} |\mathbf{a}_l^T \mathbf{F}(x)|^2, & x \in D, \\ 1-f(x) = \sum_{l=p+1}^{p+q} |\mathbf{a}_l^T \mathbf{F}(x)|^2, & x \in D, \\ \sum_{l=p+1}^{p+q} |\mathbf{a}_l^T \mathbf{F}(x)|^2 = 1 - \sum_{l=1}^{p} |\mathbf{a}_l^T \mathbf{F}(x)|^2, & x \in \{0,1\}^n, \end{cases} \tag{9}$$

hold, then the $(p+q)\times(1+n)$ complex coefficient matrix $[\alpha_f]$ in the form of

$$[\alpha_f] = \begin{bmatrix} \mathbf{a}_1^T \\ \vdots \\ \mathbf{a}_p^T \\ \mathbf{a}_{p+1}^T \\ \vdots \\ \mathbf{a}_{p+q}^T \end{bmatrix} \tag{10}$$

is called a degree-1 SOS complex representation matrix of f and $1-f$. Here, every row vector $\mathbf{a}_p^T$ is the coefficient vector of the degree-1 Fourier polynomial $\mathbf{a}_l^T\mathbf{F}(x)$ in Equation (9).

As we know, for the state $U_1O_xU_0|\psi_0\rangle = \sum_{i,j}(\sum_S \alpha_{i,j}^S(-1)^{S\cdot x})|i,j\rangle$ in a 1-query quantum algorithm, the amplitude $\sum_S \alpha_{i,j}^S(-1)^{S\cdot x}$ of any basis state $|i,j\rangle$ is a polynomial of degree ≤ 1. Therefore, $\sum_S \alpha_{i,j}^S(-1)^{S\cdot x} = (\alpha_{i,j}^{\emptyset},\alpha_{i,j}^{\{1\}},\alpha_{i,j}^{\{2\}},\cdots,\alpha_{i,j}^{\{n\}})(1,(-1)^{x_1},(-1)^{x_2},\cdots,(-1)^{x_n})^T$. Then, the coefficient matrix $\left[\alpha_{i,j}^S\right]$ (here, the number pair i,j is the row index and the number set S the column index. In a 1-query quantum algorithm, the number pair i,j traverses all basis states and the set S traverses $\emptyset,\{1\},\{2\},\cdots,\{n\}$) with $n+1$ columns in the form of

$$[\mathbf{a}_{\emptyset},\mathbf{a}_{\{1\}},\mathbf{a}_{\{2\}},\cdots,\mathbf{a}_{\{n\}}] \tag{11}$$

can be used to represent the state $U_1O_xU_0|\psi_0\rangle$. Without loss of generality, assume that the basis states in a 1-query quantum algorithm are the computational basis $\{|0\rangle,|1\rangle,|2\rangle,\cdots\}$. Next, the equation $U_1O_xU_0|\psi_0\rangle = \left[\alpha_{i,j}^S\right]\mathbf{F}(x)$ holds (here, $\mathbf{F}(x) = (1,(-1)^{x_1},(-1)^{x_2},\cdots,(-1)^{x_n})^T$). In order to distinguish the coefficient matrix of

the state $O_xU_0|\psi_0\rangle$ from the coefficient matrix of the state $U_1O_xU_0|\psi_0\rangle$, the notation $\left[\beta_{i,j}^S\right]$ denotes the coefficient matrix $U_1^{-1}\left[\alpha_{i,j}^S\right]$ of the state $O_xU_0|\psi_0\rangle$.

By Definition 2 and the coefficient matrix, the second characterization is presented in the following.

Theorem 3. *Any n-bit non-constant partial Boolean function $f : D \to \{0,1\}$ can be computed exactly by a 1-query quantum algorithm, if and only if there exists a degree-1 SOS complex representation matrix $[\alpha_f]$ of f and $1-f$ such that*

$$[\alpha_f]^\dagger[\alpha_f] = diag(u_0, u_1, u_2, \cdots, u_n). \tag{12}$$

Proof. $\Rightarrow$). On one hand, the coefficient matrices of the states $U_1O_xU_0|\psi_0\rangle$ and $O_xU_0|\psi_0\rangle$ are in the form of

$$\left[\alpha_{i,j}^S\right] = [\mathbf{a}_\emptyset, \mathbf{a}_{\{1\}}, \mathbf{a}_{\{2\}}, \cdots, \mathbf{a}_{\{n\}}] \tag{13}$$

and

$$\left[\beta_{i,j}^S\right] = U_1^{-1}\left[\alpha_{i,j}^S\right], \tag{14}$$

respectively. On the other hand, for any quantum state

$$U_0|\psi_0\rangle = \sum_j \alpha_{0,j}^\emptyset|0,j\rangle + \sum_{i\neq 0}\sum_j \alpha_{i,j}^\emptyset|i,j\rangle, \tag{15}$$

the state

$$O_xU_0|\psi_0\rangle = \sum_j \alpha_{0,j}^\emptyset|0,j\rangle + \sum_{i\neq 0}\sum_j \alpha_{i,j}^\emptyset(-1)^{x_i}|i,j\rangle. \tag{16}$$

Thus, the coefficient matrix of the state $O_xU_0|\psi_0\rangle$ is in the form of a block diagonal matrix

$$\left[\beta_{i,j}^S\right] = \mathrm{diag}(B_0, B_1, \cdots, B_n) \tag{17}$$

where the i-th block matrix

$$B_i = \begin{bmatrix} \alpha_{i,0}^\emptyset \\ \alpha_{i,1}^\emptyset \\ \vdots \\ \alpha_{i,j}^\emptyset \\ \vdots \end{bmatrix} \tag{18}$$

for every fixed $i \in \{0,1,2,\cdots,n\}$. Thus, $U_1^{-1}\left[\alpha_{i,j}^S\right] = \left[\beta_{i,j}^S\right] = \mathrm{diag}(B_0, B_1, \cdots, B_n)$.

According to Lemma 2 and Definition 2, the coefficient matrix $\left[\alpha_{i,j}^S\right]$ of the state $U_1O_xU_0|\psi_0\rangle$ is an SOS complex representation matrix $[\alpha_f]$ of the partial Boolean function f and $1-f$. Meanwhile, since all columns of any block-diagonal matrix $\mathrm{diag}(B_0, B_1, \cdots, B_n)$ are pairwise orthogonal and the unitary operator U_1^{-1} preserves the inner product of any two complex vectors, all columns of the matrix $[\alpha_f]$ are also pairwise orthogonal. Thus, Equation (12) holds.

$\Leftarrow$). For a degree-1 SOS complex representation matrix $[\alpha_f]$ of f and $1-f$ satisfying $[\alpha_f]^\dagger[\alpha_f] = \mathrm{diag}(u_0, u_1, u_2, \cdots, u_n)$, all columns of the matrix $[\alpha_f] = [\mathbf{a}_\emptyset, \mathbf{a}_{\{1\}}, \mathbf{a}_{\{2\}}, \cdots, \mathbf{a}_{\{n\}}]$ are pairwise orthogonal. Note that we can always get a sequence of proper vectors $B_0, B_1, \cdots, B_n$ satisfying $||B_0||=||\mathbf{a}_\emptyset|| = \sqrt{u_0}$ and $||B_i||=||\mathbf{a}_{\{i\}}|| = \sqrt{u_i}$ for all $i \in \{1,2,\cdots,n\}$ (for example, $B_i = (\sqrt{u_i}, 0, 0, \cdots)^T$). Since all columns of $[\alpha_f]$ and $\mathrm{diag}(B_0, B_1, \cdots, B_n)$ are pairwise orthogonal, there always exists a unitary operator U_1^{-1} such that $U_1^{-1}[\alpha_f]$=$\mathrm{diag}(B_0, B_1, \cdots, B_n)$.

As a result, the three states $U_1O_xU_0|\psi_0\rangle$, $O_xU_0|\psi_0\rangle$ and $U_0|\psi_0\rangle$ of an exact 1-query quantum algorithm computing f can be determined by $[\alpha_f]$, $\mathrm{diag}(B_0, B_1, \cdots, B_n)$ and

$$\begin{bmatrix} B_0 \\ B_1 \\ \vdots \\ B_n \end{bmatrix}, \tag{19}$$

respectively. Thus, Theorem 3 has been proved. □

Discussions on Theorem 3

In order to use Theorem 3, we need to find a pair of SOS real representations of f and $1-f$ first, and then transform it into a proper SOS complex representation matrix. Since it is feasible to get a pair of SOS real representations for very small (partial) Boolean functions [34–39], Theorem 3 can be tested on very small partial Boolean functions.

To some extent, Theorem 3 provides a different style for the characterization of partial Boolean function with exact quantum query complexity 1. On one hand, similar to Ref. [25] (showed an equivalence between 1-query quantum algorithms and bounded quadratic polynomials in the bounded-error setting) and [27] (proved a characterization of t-query quantum algorithms in terms of the unit ball of a space of degree-$(2t)$ polynomials), Theorem 3 shows an equivalence between the sum-of-squares polynomial representations and the exact 1-query quantum algorithm. In fact, Theorem 3 transforms the problem of proving $Q_E(f) = 1$ to the problem of solving a system of multivariate quadratic equations which is a difficult problem in practical applications. On the other hand, combining Theorem 3 with Theorem 1, we can see that the problem of solving the system of multivariate quadratic equations in Theorem 3 can be reduced to the problem of solving the linear system of equations in Theorem 1. This result is a quantum-inspired result which is an interesting application of the quantum theory.

4.2. Partial Boolean Functions Depending on k Bits

This subsection considers partial Boolean functions depending on k bits.

Inspired by Theorem 3, we get the following result.

Theorem 4. *Let the function vector $\mathbf{P}(x) = (1, x_1, x_2, \cdots, x_n)^T$ where $x = x_1x_2\cdots x_n \in \{0,1\}^n$. For any n-bit non-constant partial Boolean function $f : D \to \{0,1\}$, if f depends on k bits and can be computed exactly by a 1-query quantum algorithm, then*

$$rank((\cdots, \mathbf{P}(x), \cdots)_{f(x)=0}), rank((\cdots, \mathbf{P}(x), \cdots)_{f(x)=1}) \in \{1, 2, \cdots, n\} \tag{20}$$

and

$$rank((\cdots, \mathbf{P}(x), \cdots)_{f(x)=0}) + rank((\cdots, \mathbf{P}(x), \cdots)_{f(x)=1}) - (2n + 2 - k) \leq 0 \tag{21}$$

where all columns $\mathbf{P}(x)$ in the matrix $(\cdots, \mathbf{P}(x), \cdots)_{f(x)=b}$ traverse all legal inputs $x \in D$ satisfying $f(x) = b$ for $b \in \{0,1\}$.

In order to prove Theorem 4, the following lemma is necessary.

Lemma 3. *If an n-bit partial Boolean function $f : D \to \{0,1\}$ depends on k ($\leq n$) bits and there exists a degree-1 SOS complex representation of f, then there exist at least k non-zero columns $\mathbf{a}_{\{i_1\}}, \mathbf{a}_{\{i_2\}}, \cdots, \mathbf{a}_{\{i_k\}}$ in the matrix $[\alpha_f]$ where $i_1, i_2, \cdots, i_k \in \{1, 2, \cdots, n\}$.*

Then, Theorem 4 can be proved in the following.

Proof. Recall that all columns of the matrix $(\cdots, \mathbf{P}(x), \cdots)_{f(x)=b}$ traverse all legal inputs $x \in D$ satisfying $f(x) = b$ where the vector function $\mathbf{P}(x) = (1, x_1, x_2, \cdots, x_n)^T$. Note that the size of the matrix $(\cdots, \mathbf{P}(x), \cdots)_{f(x)=b}$ is $(n+1) \times |\{x : f(x) = b\}|$.

On one hand, for a non-constant n-bit partial Boolean function f, if there exists a degree-1 SOS complex representation, then there exists a sequence of $(n+1)$-dimensional non-zero vectors $\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_p$ satisfying

$$f(x) = \sum_{l=1}^{p} |\mathbf{a}_l^T \mathbf{P}(x)|^2, \forall x \in D. \tag{22}$$

Considering Equation (22) for x such that $f(x) = 0$ forces $\mathbf{a}_l^T \mathbf{P}(x) = 0$. Thus, $\mathbf{a}_l \perp \mathbf{P}(x)$ for all $\mathbf{P}(x)$ with $f(x) = 0$. Then, $\mathbf{a}_l$ for any l is in the orthogonal complement of the space spanned by all $\mathbf{P}(x)$ with $f(x) = 0$, we know the existence of the sequence (i.e., non-zero vectors $\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_p$) requires $1 \leq (n+1) - \text{rank}((\cdots, \mathbf{P}(x), \cdots)_{f(x)=0}) \leq n$. Similarly, considering Equation (22) for x such that $f(x) = 1$, we can get $1 \leq (n+1) - \text{rank}((\cdots, \mathbf{P}(x), \cdots)_{f(x)=1}) \leq n$. Thus, Equation (20) holds.

On the other hand, by Theorem 3, for an n-bit partial Boolean function f with exact quantum query complexity 1, there exists an SOS complex representations matrix $[\mathbf{a}_\emptyset, \mathbf{a}_{\{1\}}, \mathbf{a}_{\{2\}}, \cdots, \mathbf{a}_{\{n\}}] = [\alpha_f]$ of f and $1 - f$ such that $U_1^{-1}[\alpha_f] = \text{diag}(B_0, B_1, \cdots, B_n)$. By Equation (10), we can see that

$$\text{rank}([\alpha_f]) \leq \text{rank}([\mathbf{a}_1, \cdots, \mathbf{a}_p]) + \text{rank}([\mathbf{a}_{p+1}, \cdots, \mathbf{a}_{p+q}]). \tag{23}$$

Moreover,

$$\text{rank}([\mathbf{a}_1, \cdots, \mathbf{a}_p]) \leq (1+n) - \text{rank}((\cdots, \mathbf{P}(x), \cdots)_{f(x)=0}) \tag{24}$$

and

$$\text{rank}([\mathbf{a}_{p+1}, \cdots, \mathbf{a}_{p+q}]) \leq (1+n) - \text{rank}((\cdots, \mathbf{P}(x), \cdots)_{f(x)=1}) \tag{25}$$

can be obtained by considering Equation (22) for x such that $f(x) = 0$ and $f(x) = 1$, respectively. Using the property (i.e., preserving Euclidean norm and the rank) of the unitary matrix and Lemma 3,

$$\begin{aligned}
k \leq & |\{i \in \{0,1,\cdots,n\} : \mathbf{a}_{\{i\}} \neq 0\}| \quad \text{(Lemma 3)} \\
= & |\{i \in \{0,1,\cdots,n\} : \|B_i\| \neq 0\}| \quad \text{(The unitary operator } U_1 \text{ preserves Euclidean norm)} \\
= & \text{rank}(\text{diag}(B_0, B_1, \cdots, B_n)) \quad \text{(The characterization of the block diagonal matrix)} \\
= & \text{rank}([\mathbf{a}_\emptyset, \mathbf{a}_{\{1\}}, \mathbf{a}_{\{2\}}, \cdots, \mathbf{a}_{\{n\}}]) \quad \text{(The unitary operator } U_1 \text{ preserves the rank)} \\
= & \text{rank}([\alpha_f]) \leq 2(1+n) - \sum_{b=0}^{1} \text{rank}((\cdots, \mathbf{P}(x), \cdots)_{f(x)=b}). \quad \text{(Equations (23)–(25))}
\end{aligned} \tag{26}$$

Thus, Theorem 4 has been proved. □

Discussions on Theorem 4

The inverse direction of Theorem 4 is not always hold. For example, a three-bit partial Boolean function f given by the two-tuple $(\{x \in \{0,1\}^3 : |x| = 0\}, \{y \in \{0,1\}^3 : |y| = 1\})$. Here, it is not difficult to know that $\text{rank}((\cdots, \mathbf{P}(x), \cdots)_{f(x)=0}) = 1$ and $\text{rank}((\cdots, \mathbf{P}(x), \cdots)_{f(x)=1}) = 3$. However, using Theorem 10 of Ref. [23], $Q_E(f) \geq 2$.

Theorem 4 gives a necessary condition on the case that an n-bit partial Boolean function depends on k bits and can be computed exactly by a 1-query quantum algorithm. That is, the function $F(f) := \text{rank}((\cdots, \mathbf{P}(x), \cdots)_{f(x)=0}) + \text{rank}((\cdots, \mathbf{P}(x), \cdots)_{f(x)=1}) - (2n+2-k)$ must be negative.

Compared with Theorem 1, Theorem 4 provides an approximate method on deciding the exact quantum query complexity of a partial Boolean function. Most importantly, the method in Theorem 4 is more efficient than Theorem 1. In order to construct the linear system in Theorem 1, we should observe at most $|\{a : f(a) = 0\}||\{b : f(b) = 1\}|$ equations. In contrast, we only observe at most $|\{a : f(a) = 0\}| + |\{b : f(b) = 1\}|$ inputs in Theorem 4. From this point, Theorem 4 is more efficient than Theorem 1. Note that the

result in Theorem 4 is one-side exact. That is, if the exact quantum query complexity of a partial Boolean function f is bigger than 1 by Theorem 4, then the result is exact.

4.3. Estimating the Number of Partial Boolean Functions Depending on k Bits

In this subsection, let us evaluate the number $N_1(n,k)$ of n-bit partial Boolean functions which depend on k bits and can be computed exactly by a 1-query quantum algorithm.

As a preparation, the following lemma is necessary.

Lemma 4. *Let the vector function* $\mathbf{P}(X_k) = (1, X_{k,1}, X_{k,2}, \cdots, X_{k,n})^T$ *for a string* $X_k = X_{k,1}X_{k,2}\cdots X_{k,n} \in \{0,1\}^n$. *If* $n \geq 2$, *for any* $j \in \{1,2,\cdots,n+1\}$ *different linearly independent vectors* $\mathbf{P}(X_1)$, $\mathbf{P}(X_2)$, $\cdots$, $\mathbf{P}(X_j)$, *there exist at most* $T_j \leq 2^{j-1} - j$ *other different vectors* $\mathbf{P}(X_{j+1})$, $\mathbf{P}(X_{j+2})$, $\cdots$, $\mathbf{P}(X_{j+T_j})$ *satisfying rank*$([\mathbf{P}(X_1), \mathbf{P}(X_2), \cdots, \mathbf{P}(X_{j+T_j})]) = j$.

By Theorem 4, the fifth result is the following.

Theorem 5. *Let* $N_1(n,k)$ *be the number of all n-bit partial Boolean functions which depend on k bits and can be computed exactly by a 1-query quantum algorithm. If* $n \geq 3$ *and* $k \geq 2$, *then* $N_1(n,k) \leq n^2 2^{2^{n-1}(1+2^{2-k})+2n^2}$.

Proof. Recall that all columns in the matrix $(\cdots, \mathbf{P}(x), \cdots)_{f(x)=b}$ traverse all legal inputs $x \in D$ satisfying $f(x) = b$ where the function vector $\mathbf{P}(x) = (1, x_1, x_2, \cdots, x_n)^T$. Let r_0 denote the rank of $(\cdots, \mathbf{P}(x), \cdots)_{f(x)=0}$ and r_1 the rank of $(\cdots, \mathbf{P}(x), \cdots)_{f(x)=1}$. According to Theorem 4, $N_1(n,k)$ is not bigger than the number of all n-bit partial Boolean functions satisfying $r_0 + r_1 \leq 2n + 2 - k$ and $1 \leq r_0, r_1 \leq n$. For every fixed (r_0, r_1), an n-bit partial Boolean function can be determined using the following two steps.

In the first step, we choose r_0 linearly independent vectors

$$\mathbf{P}(X_1), \mathbf{P}(X_2), \cdots, \mathbf{P}(X_{r_0}) \tag{27}$$

and r_1 linearly independent vectors

$$\mathbf{P}(Y_1), \mathbf{P}(Y_2), \cdots, \mathbf{P}(Y_{r_1}) \tag{28}$$

for some $X_1, X_2, \cdots, X_{r_0}, Y_1, Y_2, \cdots, Y_{r_1} \in \{0,1\}^n$, respectively. Here, r_0 linearly independent vectors $\mathbf{P}(X_1)$, $\mathbf{P}(X_2)$, $\cdots$, $\mathbf{P}(X_{r_0})$ correspond to r_0 different inputs with function value 0, and r_1 linearly independent vectors $\mathbf{P}(Y_1)$, $\mathbf{P}(Y_2)$, $\cdots$, $\mathbf{P}(Y_{r_1})$ correspond to r_1 different inputs with function value 1. For every fixed (r_0, r_1), the number of different selections (i.e., $\{\mathbf{P}(X_1), \mathbf{P}(X_2), \cdots, \mathbf{P}(X_{r_0})\}$ and $\{\mathbf{P}(Y_1), \mathbf{P}(Y_2), \cdots, \mathbf{P}(Y_{r_1})\}$) is not bigger than

$$\binom{2^n}{r_0}\binom{2^n - r_0}{r_1} \leq 2^{n(r_0+r_1)} \tag{29}$$

for all $n \geq 3$ and $k \geq 2$.

In the second step, we add some other vectors $\mathbf{P}(X_{r_0+1}), \cdots$ and $\mathbf{P}(Y_{r_1+1}), \cdots$ to the sets $\{\mathbf{P}(X_1), \mathbf{P}(X_2), \cdots, \mathbf{P}(X_{r_0})\}$ and $\{\mathbf{P}(Y_1), \mathbf{P}(Y_2), \cdots, \mathbf{P}(Y_{r_1})\}$, respectively. Here, every newly added vector in the set $\{\mathbf{P}(X_{r_0+1}), \cdots\}$ should be represented linearly by the determined r_0 linearly independent vectors $\mathbf{P}(X_1), \mathbf{P}(X_2), \cdots, \mathbf{P}(X_{r_0})$ and every newly added vector in the set $\{\mathbf{P}(Y_{r_1+1}), \cdots\}$ should be represented linearly by the determined r_1 linearly independent vectors $\mathbf{P}(Y_1), \mathbf{P}(Y_2), \cdots, \mathbf{P}(Y_{r_1})$. After that, an n-bit partial Boolean function f is determined as follows. $f(x) = 0$ for x in the set $\{X_1, X_2, \cdots, X_{r_0}, \cdots\}$, $f(x) = 1$ for x in the set $\{Y_1, Y_2, \cdots, Y_{r_1}, \cdots\}$, and it is undefined for the rest cases. Using Equation (29) and Lemma 4, for every fixed (r_0, r_1), there are at most

$$2^{n(r_0+r_1)} 2^{(2^{r_0-1}-r_0)} 2^{(2^{r_1-1}-r_1)} < 2^{2n^2} 2^{(2^{n-1}+2^{n+1-k})} = 2^{2^{n-1}(1+2^{2-k})+2n^2} \tag{30}$$

partial Boolean functions with a pair of fixed r_0 and r_1. Note that the number of different (r_0, r_1) is not bigger than n^2. Thus, Theorem 5 has been proved. □

Discussions on Theorem 5

The most important contribution of Theorem 5 is to give an estimate on the number of partial Boolean functions with exact quantum 1-query complexity. This is the first non-trivial upper bound on this problem. In contrast, 3^{2^n} is the number of all n-bit partial Boolean functions, as each n-bit partial Boolean function corresponds to a string $f(0)f(1)\cdots f(2^n-1) \in \{0,1,*\}^{2^n}$. In fact, the exact quantum query complexity of any n-bit partial Boolean function is in the set $\{0,1,2,\cdots,n\}$, which implies that $\max\{N_j(n,k) : j,k \in \{0,1,2,\cdots,n\}\} \geq \frac{3^{2^n}}{(n+1)^2}$. Here, the notation $N_j(n,k)$ is used to denote the number of n-bit partial Boolean functions which depend on k bits and can be computed exactly by a j-query quantum algorithm. Thus, all n-bit partial Boolean functions with exact quantum query complexity 1 only make up a very tiny proportion of all n-bit partial Boolean functions.

Furthermore, corresponding to Fact 1 in Ref. [23], the following Fact 2 is also applicable to all partial Boolean functions, as a common 1-query quantum algorithm computes the two partial Boolean functions.

Fact 2. For any two partial Boolean functions f and g satisfying $\{x : g(x) = 0\} \subseteq \{x : f(x) = 0\}$ and $\{y : g(y) = 1\} \subseteq \{y : f(y) = 1\}$, if f can be computed exactly by a 1-query quantum algorithm, then g can also be computed exactly by this 1-query quantum algorithm.

As we know, the partial Boolean function computed by Deutsch-Jozsa algorithm can be written as

$$f(x) = \begin{cases} 0, & \forall |x| = \frac{n}{2}, \\ 1, & |x| = 0, n. \end{cases} \tag{31}$$

By Fact 2, the exact quantum query complexity of any partial Boolean function g defined by

$$g(x) = \begin{cases} 0 \text{ or } *, & \forall |x| = \frac{n}{2}, \\ 1 \text{ or } *, & |x| = 0, n. \end{cases} \tag{32}$$

is also 1. The number of functions in this form is $3 \times (2^{\binom{n}{\frac{n}{2}}} - 1)$. Thus, for an even integer n, $3 \times (2^{\binom{n}{\frac{n}{2}}} - 1)$ is a trivial lower bound on the number of n-bit partial Boolean functions with exact quantum query complexity 1. In general, given an n-bit partial Boolean function with exact quantum query complexity 1, we can find out trivially at least $(2^{|\{a:f(a)=0\}|} - 1) \times (2^{|\{b:f(b)=0\}|} - 1)$ different partial Boolean functions with exact quantum query complexity 1. By Stirling's approximation

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n, \tag{33}$$

we have

$$\binom{n}{\frac{n}{2}} = \frac{n!}{\left(\frac{n}{2}!\right)^2} \approx \frac{\sqrt{2\pi n}(\frac{n}{e})^n}{\left(\sqrt{2\pi\frac{n}{2}}\left(\frac{\frac{n}{2}}{e}\right)^{\frac{n}{2}}\right)^2} = \frac{\sqrt{2\pi n}}{\pi n}2^n. \tag{34}$$

Thus, the trivial lower bound

$$3 \times (2^{\binom{n}{\frac{n}{2}}} - 1) \approx 3 \times 2^{\frac{\sqrt{2\pi n}}{\pi n}2^n}. \tag{35}$$

Finally, the gap between the upper bound and the lower bound comes from the following two aspects. The first aspect is that the upper bound is obtained from a necessary

condition (i.e., Theorem 4) and an approximate counting argument. The second aspect is that there exist many unknown partial Boolean functions with exact quantum query complexity 1.

5. Conclusions

Motivated by this issue of exact 1-query quantum model [9,10,15,22–24], in this paper, we have investigated the power and advantage of the exact 1-query quantum model for partial Boolean functions. Specifically, we have contributed two sufficient and necessary conditions for characterizing n-bit partial Boolean functions with exact quantum query complexity 1, and one necessary condition for characterizing n-bit partial Boolean functions that depend on k ($k \leq n$) bits and can be computed exactly by a 1-query quantum algorithm. Using these characterizations, we have clarified all n-bit partial Boolean functions that depend on n bits and can be computed exactly by a 1-query quantum algorithm (in fact, $n \leq 2$ in this case, i.e. Theorem 2). Also, we have proved that the number of all n-bit partial Boolean functions with exact quantum query complexity 1 is quite small. As a result, the following two problems are worthy of further consideration.

- Find all (or some) non-trivial n-bit partial Boolean functions with exact quantum query complexity 1. This is an interesting problem for the following two aspects. On one hand, the upper bound (given by Theorem 5) of the actual number of partial Boolean functions in this class is quite big. On the other hand, known non-trivial n-bit partial Boolean functions in this class are still fairly rare [9,10,15,22–24].
- How many n-bit partial Boolean functions can be computed exactly (or with bounded-error) by k-query quantum algorithms for all $k \in \{2, 3, \cdots, n\}$? The solution of this problem is a quantitative evaluation of the advantage of the k-query quantum model. In contrast, the result of Ambainis et al. [14] is a qualitative evaluation of the advantage of the quantum query model.

Author Contributions: Conceptualization, G.X. and D.Q.; formal analysis, G.X. and D.Q.; investigation, G.X. and D.Q.; writing–original draft preparation, G.X. and D.Q.; visualization, G.X. and D.Q.; supervision, D.Q.; funding acquisition, D.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by the National Natural Science Foundation of China (Nos. 61572532, 61876195), the Natural Science Foundation of Guangdong Province of China (No. 2017B030311011).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the anonymous referees for important suggestions that help us improve the quality of the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. The Background of Definition 1

First, the following definition is used widely for total Boolean functions (i.e., $D = \{0,1\}^n$) [14,28,31].

Definition 3. *[31]. Given an n-bit partial Boolean function $f : \{0,1\}^n \to \{0,1\}$, we say that f depends on the k-th ($k \in \{1, 2, \cdots, n\}$) bit if there exists a pair of inputs $X_k, X_k^{\{k\}} \in \{0,1\}^n$ such that $1 \oplus f(X_k) = f(X_k^{\{k\}}) \in \{0,1\}$. Here, $X_k^{\{k\}}$ is the same as X_k except for the k-th bit being flipped.*

Based on Definition 3, the statements "f depends on k bits" and "f depends on all bits (or n bits)" is also used widely for total Boolean functions (i.e., $D = \{0,1\}^n$). Clearly, if

a total Boolean function $f:\{0,1\}^n \to \{0,1\}$ depends on k bits, then we can always find out a sequence $i_1, i_2, \cdots, i_k$ such that f depends on the i_1-bit, the i_2-bit, $\cdots$, and the i_k-bit. Meanwhile, the unique multilinear polynomial representation of f is on the k variables (i.e., $x_{i_1}, x_{i_2}, \cdots, x_{i_k}$). However, for partial Boolean functions (i.e., $D \neq \{0,1\}^n$), things become different. On one hand, the even n-bit partial Boolean function g computed by Deutsch-Jozsa algorithm [10] does not depend on any bit (using Definition 3). On the other hand, the partial Boolean function g is not a constant function. Thus, Definition 3 is not proper for many partial Boolean functions, while the statements "f depends on k bits" and "f depends on all bits (or n bits)" should also be reconsidered. This point motivates us to use Definition 1 in this paper. Specifically, in the case of total Boolean functions, Definition 1 is consistent with Definition 3.

Appendix B. Proof of Lemma 1

Proof. Recall that $N(n)$ $(n \geq 1)$ denotes the number of all n-bit partial Boolean functions depending on n bits. For any k-bit $(k \geq 1)$ partial Boolean function $f: D \to \{0,1\}$ $(D \subseteq \{0,1\}^k)$ depending on k bits (Note that D is not an empty set for $k \geq 1$), we construct at least 3^{2^k-1} $(k+1)$-bit partial Boolean function g depending on $k+1$ bits by f. In fact, for any fixed $y = y_1 y_2 \cdots y_k \in D \subseteq \{0,1\}^k$, any $(k+1)$-bit partial Boolean function g defined by

$$\begin{cases} g(x0) = f(x), & \forall x \in D, \\ g(x0) = *, & \forall x \in \{0,1\}^k \backslash D, \\ g(x1) = 1 \oplus f(y), & x = y, \\ g(x1) \in \{0,1,*\}, & \forall x \in \{0,1\}^k \backslash \{y\} \end{cases} \tag{A1}$$

is a $(k+1)$-bit partial Boolean function depending on $(k+1)$ bits. For any $k \geq 1$, the last line in Equation (A1) implies that

$$\frac{N(k+1)}{N(k)} \geq 3^{2^k-1}. \tag{A2}$$

Since $N(1) = 2$ (i.e., $(f(0), f(1)) = (0,1)$ and $(f(0), f(1)) = (1,0)$), we have

$$N(n) \geq 2 \prod_{k=1}^{n-1} 3^{2^k-1} = 2 \times 3^{2^n-n-1}. \tag{A3}$$

The lemma has been proved. □

Appendix C. Proof of Lemma 3

Proof. Assume that there exist at most $k-1$ non-zero vectors in all vectors $\mathbf{a}_\emptyset$, $\mathbf{a}_{\{1\}}$, $\mathbf{a}_{\{2\}}$, $\cdots$, $\mathbf{a}_{\{n\}}$. Then, there exist at least $n-k+1$ zero vectors $\mathbf{a}_{\{i_k\}}$, $\mathbf{a}_{\{i_{k+1}\}}$, $\cdots$, $\mathbf{a}_{\{i_n\}}$ where $i_k, i_{k+1}, \cdots, i_n \in \{1,2,\cdots,n\}$. Note that all entries $a^l_{\{i_r\}}$ in $\mathbf{a}_{\{i_r\}}$ are zeros for all $l \in \{1,2,\cdots,p+q\}$ and $r \in \{k, k+1, \cdots, n\}$. Therefore,

$$\begin{aligned} f(x) =& |f_1(x)|^2 + \cdots + |f_p(x)|^2 \\ =& \sum_{l=1}^{p} \left| a^l_\emptyset + \sum_{i \in \{1,2,\cdots,n\}} a^l_{\{i\}} (-1)^{x_i} \right|^2 \\ =& \sum_{l=1}^{p} \left| a^l_\emptyset + \sum_{r \in \{1,2,\cdots,k-1\}} a^l_{\{i_r\}} (-1)^{x_{i_r}} \right|^2. \end{aligned} \tag{A4}$$

Obviously, the number of variables in this representation of f is at most $k-1$. Since f depends on k bits, this is a contradiction. Thus, the lemma has been proved. □

Appendix D. Proof of Lemma 4

Proof. For every $k \geq j+1$ and every fixed input $X_r = X_{r,1}X_{r,2}\cdots X_{r,n} \in \{0,1\}^n$ where $r \in \{1,2,\cdots,j,k\}$, let $\mathbf{P}(X_k) = s_1\mathbf{P}(X_1) + s_2\mathbf{P}(X_2) + \cdots + s_j\mathbf{P}(X_j)$ which is a linear system of equations on j undetermined variables s_1, s_2, $\cdots$, s_j. Note that this linear system of equations

$$\begin{cases} \sum_{r=1}^{j} s_r = 1, \\ \sum_{r=1}^{j} s_r X_{r,i} = X_{k,i}, \quad i \in \{1,2,\cdots,n\}. \end{cases} \tag{A5}$$

consists of $n+1$ equations. Since vectors $\mathbf{P}(X_1)$, $\mathbf{P}(X_2)$, $\cdots$, $\mathbf{P}(X_j)$ is a base, the rank of the matrix $(\mathbf{P}(X_1), \mathbf{P}(X_2), \cdots, \mathbf{P}(X_j))$ is j. In other word, for all rows of the matrix $(\mathbf{P}(X_1), \mathbf{P}(X_2), \cdots, \mathbf{P}(X_j))$, we can find out a base which consists of j row vectors. After that, the augmented matrix of the linear system Equation (A5)

$$\begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ X_{1,1} & X_{2,1} & \cdots & X_{j,1} & X_{k,1} \\ X_{1,2} & X_{2,2} & \cdots & X_{j,2} & X_{k,2} \\ \vdots & \vdots & & \vdots & \vdots \\ X_{1,n} & X_{2,n} & \cdots & X_{j,n} & X_{k,n} \end{bmatrix} \tag{A6}$$

can be transformed into

$$\begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ X_{1,i_1} & X_{2,i_1} & \cdots & X_{j,i_1} & X_{k,i_1} \\ X_{1,i_2} & X_{2,i_2} & \cdots & X_{j,i_2} & X_{k,i_2} \\ \vdots & \vdots & & \vdots & \vdots \\ X_{1,i_{j-1}} & X_{2,i_{j-1}} & \cdots & X_{j,i_{j-1}} & X_{k,i_{j-1}} \\ 0 & 0 & \cdots & 0 & X'_{k,i_j} \\ 0 & 0 & \cdots & 0 & X'_{k,i_{j+1}} \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & X'_{k,i_n} \end{bmatrix} \tag{A7}$$

where $\{i_1, i_2, \cdots, i_n\} = \{1,2,\cdots,n\}$. According to the solution theory of linear system of equations, if any of X'_{k,i_j}, $X'_{k,i_{j+1}}$, $\cdots$, and X'_{k,i_n} is non-zero, then there exists no solution of the linear system Equation (A5) and the vector $\mathbf{P}(X_k)$ is not what we want. Otherwise, we can always get a solution

$$\begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_j \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ X_{1,i_1} & X_{2,i_1} & \cdots & X_{j,i_1} \\ X_{1,i_2} & X_{2,i_2} & \cdots & X_{j,i_2} \\ \vdots & \vdots & & \vdots \\ X_{1,i_{j-1}} & X_{2,i_{j-1}} & \cdots & X_{j,i_{j-1}} \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ X_{k,i_1} \\ X_{k,i_2} \\ \vdots \\ X_{k,i_{j-1}} \end{bmatrix}. \tag{A8}$$

As a result, for every $X_{k,i_1}X_{k,i_2}\cdots X_{k,i_{j-1}} \in \{0,1\}^{j-1}$, we either can get a unique string $X_k = X_{k,1}X_{k,2}\cdots X_{k,n} \in \{0,1\}^n$ satisfying $X'_{k,i_j} = X'_{k,i_{j+1}} = \cdots = X'_{k,i_n} = 0$ in Equation (A7) or can not get a string $X_{r,1}X_{r,2}\cdots X_{r,n} \in \{0,1\}^n$ satisfying $X'_{k,i_j} = X'_{k,i_{j+1}} = \cdots = X'_{k,i_n} = 0$ in Equation (A7). The lemma has been proved. □

References

1. Buhrman, H.; de Wolf, R. Complexity measures and decision tree complexity: A survey. *Theor. Comput. Sci.* **2002**, *288*, 21–43. [CrossRef]
2. Beals, R.; Buhrman, H.; Cleve, R.; Mosca, M.; de Wolf, R. Quantum lower bounds by polynomials. *J. ACM* **2001**, *48*, 778–797. [CrossRef]
3. Ambainis, A. Quantum lower bounds by quantum arguments. *J. Comput. Syst. Sci.* **2002**, *64*, 750–767. [CrossRef]
4. Childs, A.M.; Landahl, A.J.; Parrilo, P.A. Quantum algorithms for the ordered search problem via semidefinite programming. *Phys. Rev. A* **2007**, *75*, 032335. [CrossRef]
5. Hyer, P.; Ŝpalek, R. Lower Bounds on Quantum Query Complexity. *Bull. Eur. Assoc. Theor. Comput. Sci.* **2005**, *87*, 78–103.
6. Nielson, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information*, 10th ed.; Cambridge University Press: Cambridge, MA, USA, 2012.
7. Shor, P.W. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, USA, 20–22 November 1994; Goldwasser, S., Ed.; IEEE Computer Society Press: Los Alamitos, CA, USA, 1994.
8. Grover, L.K. A fast quantum mechanical algorithm for database search. In Proceedings of the 28th Annual ACM Symposium on the Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; ACM: New York, NY, USA, 1996.
9. Deutsch, D. Quantum theory, the Church-Turing Principle and the universal quantum computer. *Proc. R. Soc. Lond. Ser. A* **1985**, *400*, 97–117.
10. Deutsch, D.; Jozsa, R. Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond. Ser. A* **1992**, *439*, 553–558.
11. Harrow, A.W.; Hassidim, A.; Lloyd, S. Quantum Algorithm for Linear Systems of Equations. *Phys. Rev. Lett.* **2009**, *103*, 150502. [CrossRef]
12. Ambainis, A.; Iraids, J.; Smotrovs, J. Exact quantum query complexity of EXACT and THRESHOLD. In Proceedings of the 8th Conference on the Theory of Quantum Computation, Communication and Cryptography, Guelph, ON, Canada, 21–23 May 2013; Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik: Dagstuhl, Germany, 2013.
13. Ambainis, A. Superlinear advantage for exact quantum algorithms. In Proceedings of the 45th Annual ACM Symposium on Theory of Computing, Palo Alto, CA, USA, 1–4 June 2013; ACM: New York, NY, USA, 2013.
14. Ambainis, A.; Gruska J.; Zheng, S.G. Exact quantum algorithms have advantage for almost all Boolean functions. *Quantum Inf. Comput.* **2015**, *15*, 435–452.
15. Montanaro, A.; Jozsa, R.; Mitchison, G. On exact quantum query complexity. *Algorithmica* **2015**, *71*, 775–796. [CrossRef]
16. Ambainis, A.; Iraids, J.; Nagaj, D. Exact Quantum Query Complexity of EXACTnk,l. In *SOFSEM 2017: Theory and Practice of Computer Science, Proceedings of the 43rd International Conference on Current Trends in Theory and Practice of Computer Science, Limerick, Ireland, 16–20 January 2017*; Steffen, B., Baier, C., Eds.; Springer: Cham, Switzerland, 2017.
17. Qiu, D.W.; Zheng, S.G. Generalized Deutsch-Jozsa problem and the optimal quantum algorithm. *Phys. Rev. A* **2018**, *97*, 062331. [CrossRef]
18. He, X.Y.; Sun, X.M.; Yang, G.; Yuan, P. Exact Quantum Query Complexity of Weight Decision Problems via Chebyshev Polynomials. Available online: https://arxiv.org/abs/1801.05717. (accessed on 22 December 2020).
19. Kaniewski, J.; Lee, T.; de Wolf, R. Query Complexity in Expectation. In Proceedings of the 42nd International Colloquium on Automata, Languages and Programming, Kyoto, Japan, 6–10 July 2015; Springer: Berlin, Germany, 2016.
20. Montanaro, A.; Nishimura, H.; Raymond, R. Unbounded error quantum query complexity. *Theor. Comput. Sci.* **2011**, *412*, 4619–4628. [CrossRef]
21. Ambainis, A.; Balodis, K.; Belovs, A.; Lee, T.; Santha, M.; Smotrovs, J. Separations in query complexity based on pointer functions. In Proceedings of the 48th ACM Symposium on Theory of Computing, Cambridge, MA, USA, 19–21 June 2016; pp. 800–813. Available online: https://arxiv.org/abs/1506.04719 (accessed on 22 December 2020).
22. Montanaro, A. Structure, Randomness and Complexity in Quantum Computation. Available online: https://people.maths.bris.ac.uk/csxam/papers/thesis.pdf (accessed on 22 December 2020).
23. Qiu, D.W.; Zheng, S.G. Characterizations of Symmetrically Partial Boolean Functions with Exact Quantum Query Complexity. Available online: https://arxiv.org/abs/1603.06505 (accessed on 22 December 2020).
24. Qiu, D.W.; Zheng, S.G. Revisiting Deutsch-Jozsa Algorithm. *Inform. Comput.* **2020**, *275*, 104605. [CrossRef]
25. Aaronson, S.; Ambainis, A.; Iraids, J.; Kokainis, M.; Smotrovs, J. Polynomials, Quantum Query Complexity, and Grothendieck's Inequality. In Proceedings of the 31st Conference on Computational Complexity, Tokyo, Japan, 29 May–1 June 2016; Raz, R., Ed.; Schloss Dagstuhl: Wadern, Germany, 2016.
26. Grillo, S.A.; Marquezino, F.L. Quantum query as a state decomposition. *Theor. Comput. Sci.* **2018**, 736, 62–75. Available online: https://arxiv.org/abs/1602.07716 (accessed on 22 December 2020).
27. Arunachalam, S.; Briet, J.; Palazuelos, C. Quantum Query Algorithms Are Completely Bounded Forms. *SIAM J. Comput.* **2019**, *48*, 903–925. Available online: https://arXiv:1711.07285 (accessed on 22 December 2020).
28. Chen, W.J.; Ye, Z.K.; Li, L.Z. Characterization of exact one-query quantum algorithms. *Phys. Rev. A* **2020**, *101*, 022325. [CrossRef]
29. Mukherjee, C.S.; Maitra, S. Classical-Quantum Separations in Certain Classes of Boolean Functions-Analysis Using the Parity Decision Trees. Available online: https://arxiv.org/abs/2004.12942 (accessed on 22 December 2020).
30. De Wolf, R. Nondeterministic quantum query and communication complexities. *SIAM J. Comput.* **2003**, *32*, 681–699. [CrossRef]

31. Simon, H.U. A tight $\omega(\log \log n)$-bound on the time for parallel RAM's to compute non-degenerated boolean functions. *Inf. Control.* **1982**, *55*, 102–107. [CrossRef]
32. Nisan, N.; Szegedy, M. On the degree of Boolean functions as real polynomials. *Comput. Complex.* **1994**, *4*, 301–313. [CrossRef]
33. Xu, G.L., Qiu, D.W. From the sum-of-squares representation of a Boolean function to an optimal exact quantum query algorithm. *Quantum Inf. Process* **2021**, *20*, 33. [CrossRef]
34. Lee, T.; Prakash, A.; de Wolf, R.; Yuen, H. On the sum-of-squares degree of symmetric quadratic functions. In Proceedings of the 31st Conference on Computational Complexity, Tokyo, Japan, 29 May–1 June 2016; Raz, R., Ed.; Schloss Dagstuhl: Wadern, Germany, 2016.
35. Powers, V.; Wörmann, T. An algorithm for sums of squares of real polynomials. *J. Pure. Appl. Algebra* **1998**, *127*, 99–104. [CrossRef]
36. Lasserre, J.B. A sum of squares approximation of nonnegative polynomials. *SIAM J. Optimiz.* **2006**, *16*, 751–765. [CrossRef]
37. Lasserre, J.B. Sufficient conditions for a real polynomial to be a sum of squares. *Arch. Math.* **2006**, *89*, 390–398. [CrossRef]
38. Papachristodoulou, A.; Anderson, J.; Valmorbida, G.; Prajna, S.; Seiler, P.; Parrilo, P. SOSTOOLS Version 3.00 Sum of Squares Optimization Toolbox for MATLAB. Available online: https://arxiv.org/abs/1310.4716 (accessed on 22 December 2020).
39. Blekherman, G.; Gouveia, J.; Pfeiffer, J. Sums of squares on the hypercube. *Math. Z.* **2016**, *284*, 41–54. [CrossRef]