Quantum algorithms for the ordered search problem via semidefinite programming

Andrew M. Childs,[1,*] Andrew J. Landahl,[2,†] and Pablo A. Parrilo[3,‡]

[1]*Institute for Quantum Information, California Institute of Technology, Pasadena, California 91125, USA*

[2]*Center for Advanced Studies, Department of Physics and Astronomy, University of New Mexico, Albuquerque, New Mexico 87131, USA*

[3]*Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA*

(Received 24 August 2006; published 26 March 2007)

One of the most basic computational problems is the task of finding a desired item in an ordered list of N items. While the best classical algorithm for this problem uses $\log_2 N$ queries to the list, a quantum computer can solve the problem using a constant factor fewer queries. However, the precise value of this constant is unknown. By characterizing a class of quantum query algorithms for the ordered search problem in terms of a semidefinite program, we find quantum algorithms for small instances of the ordered search problem. Extending these algorithms to arbitrarily large instances using recursion, we show that there is an exact quantum ordered search algorithm using $4\log_{605} N \approx 0.433\log_2 N$ queries, which improves upon the previously best known exact algorithm.

DOI: 10.1103/PhysRevA.75.032335 PACS number(s): 03.67.Lx

I. INTRODUCTION

The *ordered search problem* (OSP) is the problem of finding the first occurrence of a target item in an ordered list of N items subject to the promise that the target item is somewhere in the list. Equivalently, we can remove the promise by viewing the OSP as the problem of finding the earliest insertion point for a target item in a sorted list of $N-1$ items. The OSP is ubiquitous in computation, not only in its own right, but also as a subroutine in algorithms for related problems, such as sorting.

To characterize the computational difficulty of the ordered search problem, we are interested in knowing how many times the list must be queried to find the location of the target item. The minimal number of queries required to solve the problem in the worst case is known as its *query complexity*. Using information-theoretic arguments, one can prove that any deterministic classical algorithm for the OSP requires $\lceil \log_2 N \rceil$ queries. This lower bound is achieved by the well-known binary search algorithm [1].

Quantum computers can solve the ordered search problem using a number of queries that is smaller by a constant factor than the number of queries used in the binary search algorithm. The best known lower bound, proved by Høyer, Neerbek, and Shi, shows that any quantum algorithm for the OSP that is *exact* (i.e., succeeds with unit probability after a fixed number of queries) requires at least $(\ln N - 1)/\pi \approx 0.221\log_2 N$ queries [2]. In other words, at most a constant factor speedup is possible. The best published exact quantum OSP algorithm, obtained by Farhi *et al.*, uses $3\lceil \log_{52} N \rceil \approx 0.526\log_2 N$ queries, showing that a constant-factor speedup is indeed possible [3]. However, there remains a gap between the constants in these lower and upper bounds. Since the OSP is such a basic problem, it is desirable to establish the precise value of the constant factor speedup for the best possible quantum algorithm: this constant is a fundamental piece of information about the computational power of quantum mechanics.

In this article, we study the query complexity of the ordered search problem by exploiting a connection between quantum query problems and convex optimization. Specifically, we show that the existence of an algorithm for the OSP that is *translation invariant* (in the sense of [3]) is equivalent to the existence of a solution for a certain semidefinite program (SDP). By solving this semidefinite program numerically, we show that there is an exact quantum query algorithm to search a list of size $N=605$ using four queries.

Since the size of the semidefinite program increases as we increase N, we cannot directly perform a numerical search for a quantum ordered search algorithm for arbitrarily large problem instances. However, by applying the four-query algorithm recursively, we see that there is an exact algorithm for a list of size N using $4\log_{605} N \approx 0.433\log_2 N$ queries. Thus, our result narrows the gap between the best known algorithm and the lower bound of [2]. In particular, this shows that the quantum query complexity of the OSP is strictly less than $\log_2 \sqrt{N}$, which one might have naively guessed was the query complexity of ordered search by analogy with the *unordered search problem*, whose quantum query complexity is $\Theta(\sqrt{N})$ [4,5].

In addition to providing a way of searching for algorithms, the semidefinite programming approach has the advantage that a solution to the dual SDP provides a certificate of the nonexistence of an algorithm. Thus we are able to provide some evidence (although not a proof) that $N=605$ is the largest size of a list that can be searched with $k=4$ queries by showing that no algorithm exists for $N=606$.

The remainder of the article is organized as follows. In Sec. II, we describe the class of translation-invariant algorithms that we focus on and summarize known results about such algorithms. In Sec. III, we show how these algorithms can be characterized as the solutions of a semidefinite program. Finally, in Sec. IV, we present the results obtained by solving this semidefinite program and conclude with a brief discussion.

*Electronic address: amchilds@caltech.edu
†Electronic address: alandahl@unm.edu
‡Electronic address: parrilo@mit.edu

II. TRANSLATION-INVARIANT QUANTUM ALGORITHMS FOR THE OSP

A. Query models for the OSP

In the standard query model for the ordered search problem, sometimes known as the *comparison model*, a query to the xth position of the list indicates whether the target item occurs before or after (or at) that position. If the target item is at position $j \in \{0,1,\ldots,N-1\}$, then its location is encoded in the function $f_j:\{0,1,\ldots,N-1\}\to\{\pm 1\}$ defined as

$$f_j(x) := \begin{cases} -1, & x < j, \\ +1, & x \geqslant j. \end{cases} \tag{1}$$

When searching an explicit list with no information about its structure other than the fact that it is ordered, this function captures essentially all the information that is available from examining a given position in the list. The ordered search problem is to determine j using as few queries to f_j as possible.

The ordered search problem has a kind of symmetry: for $j \in \{0,1,\ldots,N-2\}$, if we change the target item from j to $j+1$, then we find

$$f_{j+1}(x) = \begin{cases} -1, & x = 0, \\ f_j(x-1), & 1 \leqslant x < N. \end{cases} \tag{2}$$

Unfortunately, we must handle what happens at the boundary (namely, at $x=0$) as a special case. However, as observed in [3], we can remedy the situation by extending f_j to the function $g_j:\mathbb{Z}/2N\to\{\pm 1\}$ defined as

$$g_j(x) := \begin{cases} f_j(x), & 0 \leqslant x < N, \\ -f_j(x-N), & N \leqslant x < 2N, \end{cases} \tag{3}$$

for $j \in \{0,1,\ldots,N-1\}$, and

$$g_j(x) := -g_{j-N}(x), \tag{4}$$

for $j \in \{N,N+1,\ldots,2N-1\}$, where all arithmetic is done in $\mathbb{Z}/2N$—i.e., modulo $2N$. The advantage of using this modified function is that the symmetry expressed in (2) now appears without special boundary conditions as a *translation equivariance* in the group $\mathbb{Z}/2N$: namely, as

$$g_{j+\ell}(x) = g_j(x-\ell), \quad \forall\, j,x,\ell \in \mathbb{Z}/2N. \tag{5}$$

Although the functions g_j are defined for all $j \in \mathbb{Z}/2N$, it is sufficient to consider the problem of determining j with the promise that $j \in \{0,1,\ldots,N-1\}$. (Indeed, with the quantum phase oracle for g_j that we define in the next section, it will turn out that j is indistinguishable from $j+N$.) For this problem, the functions f_j and g_j are equivalent, in the sense that any algorithm using one type of query can be mapped onto an algorithm using the same number of the other type of query. A single query to f_j can be simulated by simply querying g_j on the original value of $x \in \{0,1,\ldots,N-1\}$. On the other hand, one query to g_j can be simulated by a query to f_j pre- and post-processed according to (3). Thus, there is no loss of generality in using the modified function g_j instead of the original function f_j: the query complexity of the OSP is the same using either type of query.

B. Quantum query algorithms

In the quantum mechanical version of the query model, access to the query function is provided by a unitary transformation. Specifically, we will use the *phase oracle* for g_j, a linear operator G_j defined by the following action on the computational basis states $\{|x\rangle : x \in \mathbb{Z}/2N\}$:

$$G_j|x\rangle := g_j(x)|x\rangle. \tag{6}$$

A k-query quantum algorithm is specified by an initial quantum state $|\psi_0\rangle$ and a sequence of (j-independent) unitary operators $U_1,U_2,\ldots,U_k$. The algorithm begins with the quantum computer in the state $|\psi_0\rangle$, and query transformations and the operations U_t are applied alternately, giving the final quantum state

$$|\phi_j\rangle := U_k G_j U_{k-1} \cdots U_1 G_j |\psi_0\rangle. \tag{7}$$

We say the algorithm is *exact* if $\langle\phi_j|\phi_{j'}\rangle = \delta_{j,j'}$ for all $j,j' \in \{0,1,\ldots,N-1\}$, since in this case there is some measurement that can determine the result j mod N with certainty. (Note that the final unitary U_k has no effect on whether the algorithm is exact, but it is convenient to include for the purposes of the following discussion.) For each value of N, our goal is to find choices of $|\psi_0\rangle$ and $U_1,U_2,\ldots,U_k$ for k as small as possible so that the resulting quantum algorithm is exact.

The search for a good quantum algorithm for the OSP can be considerably simplified by exploiting the translation equivariance (5) of the function g_j [3]. This equivariance manifests itself as a symmetry of the query operators. In terms of the translation operator T defined by

$$T|x\rangle := |x+1\rangle \quad \forall\, x \in \mathbb{Z}/2N, \tag{8}$$

where addition is again performed in $\mathbb{Z}/2N$, we have

$$TG_jT^{-1} = G_{j+1} \quad \forall\, j \in \mathbb{Z}/2N. \tag{9}$$

Thus, it is natural to choose the quantum algorithm to have the translation-invariant initial state

$$|\psi_0\rangle = \frac{1}{\sqrt{2N}} \sum_{x=0}^{2N-1} |x\rangle, \tag{10}$$

satisfying $T|\psi_0\rangle = |\psi_0\rangle$, and translation-invariant unitary operations U_t—i.e., unitary operators satisfying

$$TU_tT^{-1} = U_t \tag{11}$$

for $t \in \{1,2,\ldots,k\}$. Of course, while (9) holds for all $j \in \mathbb{Z}/2N$, we are promised that $j \in \{0,1,\ldots,N-1\}$. Correspondingly, we can require the N possible orthogonal final states to label the location of the marked item as follows:

$$|\phi_j\rangle := \begin{cases} \dfrac{1}{\sqrt{2}}(|j\rangle + |j+N\rangle), & k \text{ even}, \\[2ex] \dfrac{1}{\sqrt{2}}(|j\rangle - |j+N\rangle), & k \text{ odd}, \end{cases} \tag{12}$$

where the separation into k even and odd is done for reasons explained in [3]. Overall, we refer to an algorithm with the initial state (10), unitary operations satisfying (11), and the final states (12) as an *exact, translation-invariant algorithm* (in the sense of [3]).

An advantage of a translation-invariant algorithm for the OSP is that, if it can find the target item when $j=0$, then it can find the target item for all values of j. Using (9), we have $T^{-j}G_jT^j=G_0$. Thus

$$|\phi_j\rangle = (T^jU_kT^{-j})G_j \cdots (T^jU_1T^{-j})G_j(T^j|\psi_0\rangle) \tag{13}$$

$$=T^jU_k(T^{-j}G_jT^j)U_{k-1}\cdots U_1(T^{-j}G_jT^j)|\psi_0\rangle \tag{14}$$

$$=T^jU_kG_0U_{k-1}\cdots U_1G_0|\psi_0\rangle \tag{15}$$

$$=T^j|\phi_0\rangle. \tag{16}$$

In other words, if we find an algorithm whose final state in the case $j=0$ is given by (12), then the final state will also be given by (12) for $j\in\{1,2,\ldots,N-1\}$.

C. Characterizing algorithms by polynomials

Another advantage of translation-invariant quantum algorithms for the OSP is that they have a convenient characterization in terms of Laurent polynomials. A *Laurent polynomial* is a function $Q:\mathbb{C}\to\mathbb{C}$ that can be written as

$$Q(z)=\sum_{i=-D}^{D} q_iz^i \tag{17}$$

for some non-negative integer D, where each $q_i\in\mathbb{C}$. We call D the *degree* of $Q(z)$. We say $Q(z)$ is *non-negative* if, on the unit circle $|z|=1$, $Q(z)$ is real-valued and satisfies $Q(z)\geqslant 0$. Note that for $|z|=1$, $z^*=z^{-1}$, so $Q(z)$ is real valued on the unit circle if and only if $q_i=q_{-i}^*$ for all $i\in\{0,1,\ldots,D\}$. If $Q(z)=Q(z^{-1})$ for all $z\in\mathbb{C}$—i.e., if $q_i=q_{-i}$ for all $i\in\{1,2,\ldots,D\}$—we say $Q(z)$ is *symmetric*. Thus, $Q(z)$ is non-negative and symmetric if and only if $q_i=q_{-i}\in\mathbb{R}$ for all $z\in\{0,1,\ldots,D\}$. An example of a non-negative, symmetric Laurent polynomial that is relevant to the ordered search problem is the *Hermite kernel* of degree $N-1$,

$$H_N(z) := \sum_{i=-(N-1)}^{N-1}\left(1-\frac{|i|}{N}\right)z^i \tag{18}$$

$$=\frac{1}{N}\left(\frac{z^{-N}-1}{z^{-1}-1}\right)\left(\frac{z^N-1}{z-1}\right). \tag{19}$$

The following result of Farhi *et al.* characterizes exact translation-invariant algorithms for the ordered search problem in terms of Laurent polynomials.

Theorem 1 ([3]). There exists an exact, translation-invariant, k-query quantum algorithm for the N-element OSP if and only if there exist non-negative, symmetric Laurent polynomials $Q_0(z),\ldots,Q_k(z)$ of degree $N-1$ such that

$$Q_0(z)=H_N(z), \tag{20}$$

$$Q_t(z)=Q_{t-1}(z) \text{ at } z^N=(-1)^t, \forall t\in\{1,2,\ldots,k\}, \tag{21}$$

$$Q_k(z)=1, \tag{22}$$

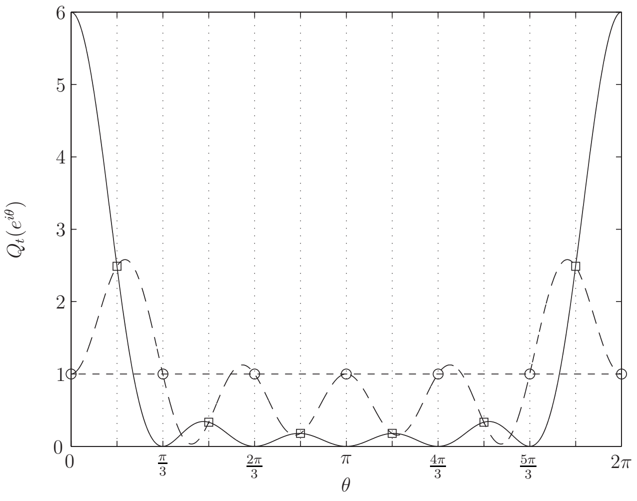

FIG. 1. $Q_t(e^{i\theta})$ as a function of θ for $k=2$ and $N=6$. The solid, long-dashed, and short-dashed lines represent $t=0$, 1, and 2, respectively. The intersections at roots of 1 and -1 are indicated by circles and squares, respectively.

$$\frac{1}{2\pi}\int_0^{2\pi} Q_t(e^{i\theta})d\theta=1,\quad \forall\, t\in\{0,1,\ldots,k\}. \tag{23}$$

Each polynomial $Q_t(z)$ in this theorem represents the quantum state of the algorithm after t queries. Indeed, if we write

$$Q_t(z)=\sum_{i=-(N-1)}^{N-1} q_i^{(t)}z^i, \tag{24}$$

then

$$q_i^{(t)}=2\sum_{m=1}^{N-i}\langle\psi_t|N-m\rangle\langle N-m-i|\psi_t\rangle, \tag{25}$$

where

$$|\psi_t\rangle := U_tG_0U_{t-1}\ldots U_1G_0|\psi_0\rangle \tag{26}$$

is the state of the quantum computer after t queries when the target item is $j=0$ [3]. Given polynomials satisfying (20)–(23), one can reconstruct all of the unitary operators U_t for the algorithm using (25).

Figure 1 shows the (unique) solution to (20)–(23) for $k=2$ and $N=6$ [3]. In general, the polynomial $Q_0(z)$ (the Hermite kernel) characterizes complete ignorance of the target location at the beginning of the algorithm, and subsequent polynomials become flatter and flatter until the final polynomial $Q_k(z)=1$ is reached, corresponding to exact knowledge of the target location. Because each query can only change the quantum state in a restricted way, successive polynomials must agree at certain roots of ± 1. Also, each polynomial must be non-negative and suitably normalized.

With $k=2$, there is a unique choice for the polynomial $Q_1(z)$, which might or might not be non-negative depending upon the value of N. For $N\leqslant 6$, this polynomial is non-negative (showing that an ordered list of size $N \leqslant 6$ can be searched in two quantum queries), whereas for $N \geqslant 7$, it is not [3].

The best ordered search algorithm discovered by Farhi *et al.* was found by considering $k=3$ queries. For fixed values of the degree $N-1$, they numerically searched for polynomials $Q_1(z)$ and $Q_2(z)$ satisfying the constraints (20)–(23) of theorem 1. The largest value of N for which they found a solution was $N=52$. Applying this 52-item ordered search algorithm recursively gives an algorithm for instances with N arbitrarily large. Specifically, one divides the list into 52 sublists and applies the algorithm to the largest (rightmost) item of each sublist, finding the sublist that contains the target in 3 queries. This process repeats, with every 3 queries dividing the problem size by 52, leading to a query complexity of $3\lceil \log_{52} N \rceil$. (Note that although the base algorithm in this recursion is translation-invariant, the scalable algorithm generated in this way is not.)

In general, recursion can be used to turn small base cases into scalable algorithms, so improved quantum algorithms for the OSP can be found by discovering improved base cases. Subsequent work by Brookes, Jacokes, and Landahl sought such algorithms using a conjugate gradient descent search for the polynomials $Q_t(z)$ [6]. This method is guaranteed to work (for a small enough step size) because the space of polynomials satisfying (20)–(23) is convex. The best solutions found by this method were $N=56$ for $k=3$ and $N=550$ for $k=4$, implying a $(4\log_{550} N \approx 0.439 \log_2 N)$-query recursive algorithm. Unfortunately, conjugate gradient descent (or any approach based on local optimization) can never prove that finite-instance algorithms do not exist for a given number of queries, k. It could always be the case that lack of progress by a solver is indicative of inadequacies of the solver (e.g., the step size is too large, etc.). In the next section, we recharacterize exact translation-invariant quantum OSP algorithms in a way that allows either their existence or nonexistence (whichever the case may be) to be proved efficiently.

III. SEMIDEFINITE PROGRAM FOR TRANSLATION-INVARIANT QUANTUM ALGORITHMS FOR THE OSP

A. Formulation of the SDP

In this section, we show that the problem of finding Laurent polynomials satisfying the conditions of theorem 1 can be viewed as an instance of a particular kind of convex optimization problem: namely, a semidefinite program [7]. The basic idea is to use the spectral factorization of non-negative Laurent polynomials to rewrite Eqs. (20)–(23) as linear constraints on positive semidefinite matrices.

The spectral factorization of non-negative Laurent polynomials follows from the Fejér-Riesz theorem.

Theorem 2 ([8,9]). Let $Q(z)$ be a Laurent polynomial of degree D. Then $Q(z)$ is non-negative if and only if there exists a polynomial $P(z)=\sum_{i=0}^{D} p_i z^i$ of degree D such that $Q(z)=P(z)P(1/z^*)^*$.

Let Tr_i denote the trace along the ith superdiagonal [or $(-i)$th subdiagonal, for $i<0$]—i.e., for an $N\times N$ matrix X,

$$\mathrm{Tr}_i X = \begin{cases} \displaystyle\sum_{\ell=1}^{N-i} X_{\ell,\ell+i}, & i \geqslant 0, \\ \displaystyle\sum_{\ell=1}^{N+i} X_{\ell-i,\ell}, & i < 0. \end{cases} \tag{27}$$

The Fejér-Riesz theorem can be used to express non-negative Laurent polynomials in terms of positive semidefinite matrices, as shown by the following lemma.

Lemma 1. Let $Q(z)=\sum_{i=-(N-1)}^{N-1} q_i z^i$ be a Laurent polynomial of degree $N-1$. Then $Q(z)$ is non-negative if and only if there exists an $N\times N$ Hermitian, positive semidefinite matrix Q such that $q_i=\mathrm{Tr}_i\, Q$.

Proof. The "if" direction follows from the representation

$$Q(z) = [1 \cdots z^{-(N-1)}]\, Q \begin{bmatrix} 1 \\ \vdots \\ z^{N-1} \end{bmatrix}. \tag{28}$$

This $Q(z)$ is real on $|z|=1$ since $Q=Q^\dagger$; it is non-negative there because Q is positive semidefinite.

The converse follows from the spectral factorization of $Q(z)$. Let $Q(z)=P(z)P(1/z^*)^*$, let $\mathbf{p} := [p_0 \cdots p_{N-1}]^T$, and let $\mathbf{z} := [1 \cdots z^{N-1}]^T$. Then $P(z)=\mathbf{p}^T\mathbf{z}$ and $Q(z)=\mathbf{z}^\dagger \mathbf{p}^* \mathbf{p}^T \mathbf{z}$ on $|z|=1$. We choose $Q := \mathbf{p}^*\mathbf{p}^T$, which by construction is Hermitian and positive semidefinite. Furthermore, since $Q(z)$ on $|z|=1$ determines the coefficients q_i, we have $q_i=\mathrm{Tr}_i\, Q$. ■

Because the Laurent polynomials in theorem 1 are not only non-negative but also *symmetric*, we can restrict the associated matrices to be real symmetric, as the following lemma shows.

Lemma 2. If $Q(z)$ is a non-negative, symmetric Laurent polynomial, then the matrix Q in lemma 1 can be chosen to be real and symmetric without loss of generality.

Proof. Let Q be a Hermitian, positive semidefinite matrix such that $Q(z)=\mathbf{z}^\dagger Q \mathbf{z}$ on $|z|=1$, where $\mathbf{z}$ is defined as in the proof of lemma 1. Then the symmetry $Q(z)=Q(z^{-1})$ implies that $Q(z)=\mathbf{z}^\dagger Q^T \mathbf{z}$ on $|z|=1$, and by averaging these two expressions, we have $Q(z)=\mathbf{z}^\dagger \tilde{Q} \mathbf{z}$ on $|z|=1$, where $\tilde{Q} := (Q+Q^T)/2$ is real and symmetric. ■

Using lemma 2, we can recast the conditions (20)–(23) of theorem 1 as the following *semidefinite program*.

Semidefinite program $[S(k,N)]$. Find real symmetric positive semidefinite $N\times N$ matrices $Q_0, Q_1, \ldots, Q_k$ satisfying

$$Q_0 = E/N, \tag{29}$$

$$\mathcal{T}_t Q_t = \mathcal{T}_t Q_{t-1}, \quad \forall\, t \in \{1,2,\ldots,k\}, \tag{30}$$

$$Q_k = I/N, \tag{31}$$

$$\mathrm{Tr} Q_t = 1, \quad \forall\, t \in \{0,1,\ldots,k\}, \tag{32}$$

where E is the $N\times N$ matrix in which every element is 1 and $\mathcal{T}_t : \mathcal{S}^N \to \mathbb{R}^{N-1}$ is a linear operator (on the space $\mathcal{S}^N$ of real symmetric $N\times N$ matrices) that computes signed traces along the (off-) diagonals: namely,

$$(\mathcal{T}_t X)_i := \mathrm{Tr}_i X + (-1)^t \mathrm{Tr}_{i-N} X \tag{33}$$

for $i \in \{1,2,\ldots,N-1\}$.

The existence of an exact, translation-invariant quantum algorithm for the OSP is equivalent to the existence of a solution to this semidefinite program, which can be seen as follows.

Theorem 3. There exists an exact, translation invariant, k-query quantum algorithm for the N-element OSP if and only if $S(k,N)$ has a solution.

Proof. Given $Q_0, Q_1, \ldots, Q_k$ satisfying $S(k,N)$, let $Q_j(z) := [1 \cdots z^{-(N-1)}] Q_j [1 \cdots z^{N-1}]^T$. Then the symmetry of each matrix Q_j implies that each $Q_j(z)$ is a non-negative, symmetric Laurent polynomial and conditions (29)–(32) imply conditions (20)–(23), respectively.

Conversely, suppose $Q_0(z), Q_1(z), \ldots, Q_k(z)$ are non-negative, symmetric Laurent polynomials of degree $N-1$ satisfying (20)–(23). Let $Q_0 := E/N$, let $Q_k := I/N$, and let $Q_1, Q_2, \ldots, Q_{k-1}$ be positive semidefinite matrices obtained from $Q_1(z), Q_2(z), \ldots, Q_{k-1}(z)$ according to lemma 2. Then Eqs. (21) and (23) imply Eqs. (30) and (32), respectively. ■

This reformulation of the problem has the advantage that semidefinite programs are a well-studied class of convex optimization problems. In fact, semidefinite programming feasibility problems can be solved (modulo some minor technicalities) in polynomial time [7,10]. Furthermore, there are several widely available software packages for solving semidefinite programs [11–13].

Note that by "solving" a semidefinite program, we mean not only finding a solution if one exists, but also generating an *infeasibility certificate* (namely, a solution to the dual semidefinite program) if no solution exists. Thus, by solving $S(k,N)$ for various values of k and N, not only can we extract algorithms from feasible solutions, but we can also generate lower bounds for the quantum query complexity of the OSP (assuming we restrict our attention to exact, translation invariant algorithms). In other words, this approach unifies algorithm design and lower bound analysis into a single method.

B. Improved formulation by symmetry reduction

In moving from the polynomial to the semidefinite programming formulation, we have increased the number of real parameters specifying an exact, translation-invariant quantum OSP algorithm from $(N-1)(k-1)$ to $N(N+1)(k-1)/2$. As benefits, we have put the problem in a numerically tractable form and we are now able to prove the nonexistence as well as existence of algorithms. But the increase in parameters is nevertheless undesirable.

Fortunately, in our case we can reduce the size of the parameter set roughly by half by exploiting symmetry. In particular, in terms of the $N \times N$ counterdiagonal matrix (the *counteridentity matrix*)

$$J := \begin{bmatrix} 0 & 0 & \cdots & 1 \\ \vdots & \vdots & \cdot^{\cdot^{\cdot}} & \vdots \\ 0 & 1 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \end{bmatrix}, \tag{34}$$

we have the following lemma.

Lemma 3. If $Q_0, Q_1, \ldots, Q_k$ is a solution to $S(k,N)$, then so is $JQ_0J, JQ_1J, \ldots, JQ_kJ$.

Proof. The matrices JQ_tJ are positive semidefinite since J is unitary. Clearly, $JQ_0J = Q_0$ and $JQ_kJ = Q_k$, so Eqs. (29) and (31) are satisfied. Since $\mathrm{Tr}_i JQ_tJ = \mathrm{Tr}_{-i} Q_t$ by the definition of J and since $\mathrm{Tr}_{-i} Q_t = \mathrm{Tr}_i Q_t$ because each Q_t is a symmetric matrix, Eq. (31) is satisfied. Finally, (32) is satisfied since $J^2 = I$. ■

Thus, by convexity, if $Q_0, Q_1, \ldots, Q_k$ is a solution to $S(k,N)$, then so is $\frac{1}{2}(Q_0 + JQ_0J), \frac{1}{2}(Q_1 + JQ_1J), \ldots, \frac{1}{2}(Q_k + JQ_kJ)$. In other words, we can assume that the matrices Q_t commute with J without loss of generality.

We can use group representation theory to harness this symmetry. Note that $\{I, J\}$ is an N-dimensional (reducible) representation of the group $\mathbb{Z}/2$. Since J has $\lceil N/2 \rceil$ eigenvalues equal to -1 and the rest equal to $+1$, this representation can be diagonalized into $\lceil N/2 \rceil$ copies of the sign representation and $\lfloor N/2 \rfloor$ copies of the trivial representation by some unitary matrix U. The set of matrices that commute with all matrices in this representation (the representation of the *commutant subalgebra* of $\mathbb{Z}/2$) are therefore block diagonalizable (by the same matrix U) into two blocks, with one block having size $\lfloor N/2 \rfloor$ and the other having size $\lceil N/2 \rceil$. When N is even,

$$U = \frac{1}{\sqrt{2}} \begin{bmatrix} I & I \\ J & -J \end{bmatrix}, \tag{35}$$

and when N is odd,

$$U = \frac{1}{\sqrt{2}} \begin{bmatrix} I & 0 & I \\ 0 & \sqrt{2} & 0 \\ J & 0 & -J \end{bmatrix}, \tag{36}$$

where I and J are the $\lfloor N/2 \rfloor$ by $\lfloor N/2 \rfloor$ identity and counteridentity matrices, respectively.

Now, since we can choose the matrices Q_t to commute with J without loss of generality, we can block diagonalize them into twice as many matrices, each of which has about one quarter the number of elements. For example, for N even, $Q = JQJ$ implies that Q has the form

$$Q = \begin{bmatrix} A & B \\ JBJ & JAJ \end{bmatrix}, \tag{37}$$

where $A = A^T$ and $B = JB^TJ$. Thus we have

$$U^\dagger Q U = \frac{1}{2} \begin{bmatrix} I & I \\ J & -J \end{bmatrix}^T \begin{bmatrix} A & B \\ JBJ & JAJ \end{bmatrix} \begin{bmatrix} I & I \\ J & -J \end{bmatrix} \tag{38}$$

$$= \begin{bmatrix} A + BJ & 0 \\ 0 & A - BJ \end{bmatrix}, \tag{39}$$

so that Q is positive semidefinite if and only if $A \pm BJ$ are both positive semidefinite. The net effect of this symmetry reduction is to cut the number of real parameters in $S(k,N)$ to $N(N/2+1)(k-1)/2$ (for N even) or $(N+1)^2(k-1)/4$ (for N odd)—i.e., roughly by half.

TABLE I. Ordered list sizes N^* that are searchable by a k-query exact, translation-invariant quantum algorithm such that no such algorithm exists for a list of size N^*+1.

k	N^*
2	6
3	56
4	605
5	>5000

IV. RESULTS AND DISCUSSION

We solved the semidefinite program $S(k,N)$ for various values of k and N using the numerical solvers SEDUMI [11], SDPT3 [12], and SDPA [13]. These solvers use general-purpose primal-dual interior-point methods that eventually become limited by machine memory. (Although there are algorithms for solving SDP's that are not based on interior-point methods, we did not attempt to use such algorithms.)

The time required to solve $S(k,N)$ was substantially reduced by exploiting the symmetry described in Sec. III B. In addition, it is helpful that the constraints are fairly sparse. Nevertheless, we are ultimately limited by the fact that the maximum size of a list that can be searched increases exponentially with the number of queries, so that we can only consider fairly small values of k.

For each $k \leqslant 4$, we found the smallest value N^* such that $S(k,N^*)$ has a solution but $S(k,N^*+1)$ does not. On a dual 2.2-GHz Pentium 4 computer with 4 GB of memory, solving the SDP for a single value of N near N^* took a few seconds with $k=3$ and about 20 min with $k=4$. Although we were able to find solutions to $S(5,N)$ for some values of N (taking a few weeks for values of N in the thousands), we ran out of machine memory before we could find an infeasibility certificate for any value of N. A summary of the values N^* we obtained is presented in Table I.

By recursion, the $k=4$, $N^*=605$ query algorithm yields a scalable algorithm whose query complexity is

$$4 \log_{605} N \approx 0.433 \log_2 N. \tag{40}$$

This result also implies improvements to other algorithms; for example, it implies a quantum sorting algorithm whose query complexity is $4N \log_{605} N$.

As mentioned in the Introduction, infeasibility of $S(k,N^*+1)$ does not necessarily imply that N^* is the largest size of a list that can be searched with a k-query exact, translation invariant algorithm. However, it seems reasonable to conjecture that this might be the case. Indeed, for $k=2$ and 3, we know that the values of N^* in Table I are optimal. For those smaller problems, we were able to numerically solve the larger SDP developed by Barnum, Saks, and Szegedy to characterize general quantum query algorithms [14]. (To solve large enough instances of that SDP, it was also crucial for us to exploit the symmetry of the problem.) Those results show that $N=6$ and $N=56$ are the largest sizes of lists that

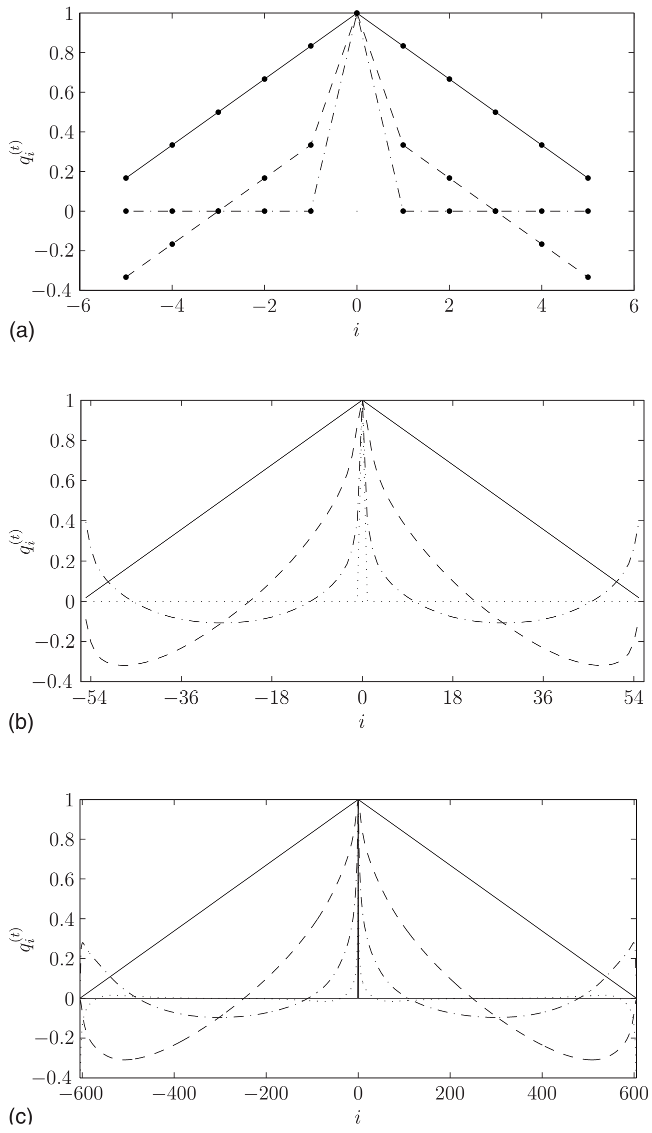

FIG. 2. Laurent polynomial coefficients $q_i^{(t)}$ as a function of i for exact, translation-invariant OSP algorithms. (a), (b), and (c) show $k=2$, 3, and 4, with $N=6$, 56, and 605, respectively. The coefficients for $t=0,1,2,3,4$ are indicated by solid, dashed, dashed-dotted, dotted, and dashed-dotted-dotted lines respectively.

can be searched with $k=2$ and $k=3$ queries, respectively, even when the assumption of translation invariance is removed.

Whether the $\frac{1}{\pi} \ln N$ lower bound on the query complexity of the OSP can be saturated remains open. However, the structure of the algorithms we obtained suggests the possibility of a well-behaved analytic solution, and it would be interesting to understand the behavior of the solution in the limit of large N. Figure 2 shows the coefficients of the polynomials $Q_t(z)$ associated with the optimal feasible solutions Q_t for $k=2,3,4$. Note the similarity of the coefficients for different values of N.

Not only have we found a particular quantum algorithm for the ordered search problem, but we have also demonstrated the usefulness of semidefinite programming as a numerical technique for discovering quantum query algorithms. Indeed, the connection between quantum query complexity and convex optimization is not unique to the ordered search problem: as mentioned above, arbitrary quantum query problems can be characterized in terms of semidefinite programs [14]. Thus, semidefinite programming appears to be a powerful tool for studying quantum query complexity.

After this work was completed, we learned that Ben-Or and Hassidim have developed an approach to quantum algorithms for ordered search based on adaptive learning [15]. Their resulting algorithm is not exact, but rather is zero error, with a stochastic running time (sometimes referred to as a Las Vegas algorithm). The expected running time of their algorithm is $0.32 \log_2 N$.

ACKNOWLEDGMENTS

A.M.C. is supported in part by the National Science Foundation under Contract No. PHY-0456720. A.J.L. is supported in part by Army Research Office Contract No. W911NF-04-1-0242. P.A.P. is supported in part by AFOSR MURI subaward No. 2003-07688-1 and the Singapore-MIT Alliance. While previously at the MIT Center for Theoretical Physics, A.M.C. and A.J.L. received support from the Department of Energy under cooperative research agreement No. DE-FC02-94ER40818 and from the Cambridge-MIT Institute, A.M.C. received support from the Fannie and John Hertz Foundation and from the National Security Agency and Advanced Research and Development Activity under Army Research Office Contract No. DAAD19-01-1-0656, and A.J.L. received support from Hewlett Packard and MIT.

[1] D. E. Knuth, *Sorting and Searching, Vol. 3 of The Art of Computer Programming*, 2nd ed. (Addison-Wesley, Upper Saddle River, NJ, 1998).

[2] P. Høyer, J. Neerbek, and Y. Shi, Algorithmica **34**, 429 (2002).

[3] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, e-print quant-ph/9901059.

[4] L. K. Grover, in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, edited by Gary L. Miller (ACM, New York, 1996), pp. 212–219.

[5] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, SIAM J. Comput. **26**, 1510 (1997).

[6] E. M. Brookes, M. B. Jacokes, and A. J. Landahl (unpublished).

[7] L. Vandenberghe and S. Boyd, SIAM Rev. **38**, 49 (1996).

[8] L. Fejér, J. Reine Angew. Math. **146**, 52 (1915).

[9] F. Riesz, J. Reine Angew. Math. **146**, 83 (1915).

[10] L. Vandenberghe and S. Boyd, *Convex Optimization* (Cambridge University Press, Cambridge, England, 2004).

[11] J. Sturm, computer code SEDUMI version 1.05, 2001, available from http://sedumi.mcmaster.ca

[12] K. C. Toh, R. H. Tütüncü, and M. J. Todd, computer code SDPT3 version 3.02, 2002, available from http://www.math.nus.edu.sg/~mattohkc/sdpt3.html

[13] K. Fujisawa, M. Kojima, K. Nakata, and M. Yamashita, computer code SDPA version 6.2.0, 2004, available from http://grid.r.dendai.ac.jp/sdpa

[14] H. Barnum, M. Saks, and M. Szegedy, in *Proceedings of the 18th IEEE International Conference on Computational Complexity*, edited by Danielle C. Martin (IEEE, Los Alamitos, California, 2003), pp. 179–193.

[15] M. Ben-Or and A. Hassidim (unpublished).