Quantum approach to the unique sink orientation problem

Dave Bacon*
Google, Seattle, Washington 98103, USA
(Received 6 July 2016; published 18 July 2017)

We consider quantum algorithms for the unique sink orientation problem on cubes. This problem is widely considered to be of intermediate computational complexity. This is because there is no known polynomial algorithm (classical or quantum) for the problem and yet it arises as part of a series of problems for which it being intractable would imply complexity-theoretic collapses. We give a reduction which proves that if one can efficiently evaluate the kth power of the unique sink orientation outmap, then there exists a polynomial time quantum algorithm for the unique sink orientation problem on cubes.

DOI: 10.1103/PhysRevA.96.012323

I. INTRODUCTION

In this paper, we are concerned with finding an efficient quantum algorithm for a problem that admits no known classical polynomial time algorithm despite considerable effort: the unique sink orientation problem on cubes. In this problem, one is given a directed graph on a hypercube such that every face (subcube) of the hypercube admits a unique sink vertex, and the goal is to find the global unique sink of the entire cube. Access to information about an instance of this problem is via an oracle which takes as inputs one of the vertices of the cube and outputs a list of the directions of the outgoing edges of that vertex. The best (classical) algorithm for this problem queries the oracle $O((1.467\ldots)^n)$ times, where n is the dimension of the hypercube [1], or, if the directed graph has no cycles, $O(\exp(n^{1/2}))$ times [2,3]. We are not able to obtain a polynomial time quantum algorithm for this problem, but we are able to show that if one could efficiently calculate the kth power of the oracle in this problem (defined below), then there exists an efficient quantum algorithm. Our path to this is through period finding, the key ingredient to Shor's efficient quantum algorithm for factoring [4,5]. The main difference in our failure versus Shor's success is that in Shor's algorithm, there are efficient algorithms to calculate the kth power of a number mod N for k exponentially large (modular exponentiation), whereas for the unique sink orientation problem on cubes, we do not yet have such a procedure. Our reduction opens an approach towards obtaining an efficient quantum algorithm for the unique sink orientation problem on cubes.

II. BACKGROUND AND MOTIVATION

The unique sink orientation problem on cubes (hereafter abbreviated as the USO problem) arises as a fundamental problem in a variety of optimization problems. The original application of the USO problem came from the observation that a polynomial time algorithm for the USO problem would yield a polynomial time algorithm for a class of linear complementarity problems that has no known polynomial time algorithm (those arising from P matrices) [6]. Another application is to linear programming. Recall that a numerical problem is strongly polynomial if, assuming unit cost for arithmetic on the involved numerical quantities, the algorithm takes a polynomial amount of time in the number of numerical constants. While there are weakly polynomial time algorithms for linear programming [7], there is no known strongly polynomial time algorithm. A polynomial time algorithm for the USO problem, however, would imply a strongly polynomial time algorithm for linear programming [8]. A variety of problems in the theory of games would also admit polynomial time solution [9–11] if there is a polynomial time algorithm for the USO problem. Finally, finding the smallest ball enclosing a set of balls would also admit a polynomial time solution if there was a polynomial time algorithm for the USO problem [12].

The USO problem is a promise problem: we are promised that the faces of the hypercube graph all have a unique sink. This promise is itself not known to be polynomial time verifiable [13]. Therefore (a decision) version of this problem does not fit nicely into a discussion of computational complexity. However, there is strong evidence that the problems which motivate investing USO are not computationally intractable. For example, the P-matrix linear complementarity problem is not NP-hard unless NP=co-NP (co-NP represents the complement of NP) [14]. Because of this, the USO problem is widely considered as a candidate for either having a polynomial time algorithm or being of intermediate computational complexity, similar to the status of the factoring problem.

A variety of classical algorithms exists for the USO problem. Most of these are known to have instances in which they take an exponentially long time. For example, the algorithm of randomly following one outgoing edge of a vertex, i.e., the Random Edge algorithm, can be shown to take $\Omega(\exp(n^{1/3}))$ time [15]. Similarly, the algorithm of choosing a random facet, and then recursing, can also be shown to take $\Omega(\exp(n^{1/2}))$ time [3]. These results all give instances upon which the given algorithms fail to work in polynomial time. In a similar vein, we will show below that a naive classical version of our quantum algorithm that uses the standard oracle to solve the USO problem requires exponential time.

Despite the USO problem being of intermediate computational complexity, the only quantum computation work for this

*dabacon@google.com

problem has been on the problem of recognizing whether or not an orientation is a unique sink orientation [16]. We note in passing that a naive application of Grover's algorithm [17] to the general USO problem yields an $O(\sqrt{2^n}) = O((1.414\ldots)^n)$ quantum algorithm, which just marginally beats the best-known classical algorithm which is $O((1.467\ldots)^n)$ [1].

III. UNIQUE SINK ORIENTATION PROBLEM

Here we introduce the USO problem and present some prior results that will be useful. An excellent introduction to this problem is provided by Schurr [18]. We begin with an information description of the problem and then proceed to a more formal specification which is useful for stating basic useful properties of the USO problem.

An instance of the USO problem is an orientation of a Boolean (combinatorial) hypercube, which has special properties. Recall that the Boolean hypercube is the graph in which vertices of the graph are Boolean strings and there is an edge between two vertices if these vertices differ in only one character of the string. Thus 0010 is a vertex and 0011 is one of its neighbors, while 1011 is not one of its neighbors. An orientation is a specification of a direction for each of the edges in the graph. There are 2^{2^n} orientations. In the USO problem, however, a constraint is imposed on the orientation. Recall that a vertex in a graph with directed edges is a sink if all the edges adjacent to a vertex are incoming. In the USO problem, the orientation is required to satisfy the condition that if one restricts one's attention to a subcube, only one of these vertices is a sink under this restriction. By restricting to a subcube, we mean considering a $k \leqslant n$ hypercube and only examining the orientation of edges between the vertices of this subcube.

Next we introduce the notation for the combinatorial hypercube that we will use in this paper. This notation is the same as that used by the literature about USOs and is used here to help provide a bridge into that literature and also to make the proofs that appear in Sec. IV cleaner. Let $[n] = \{1,2,\ldots,n\}$. We will label the vertices of the n-dimensional hypercube by the subsets of $[n]$. That is, vertices correspond to the power set of $[n]$: $V = 2^{[n]} := \{v \subset [n]\}$. If one prefers to think about the hypercube vertices as Boolean strings, then one can consider a subset of $[n]$ as corresponding to the elements in an n bit string that are 1, while those not in the subset are 0. The symmetric difference of two sets u and v is the set of elements that are in u and v but not both, $u \oplus v := (u \cup v) \setminus (u \cap v)$. Given this definition, the edges of the n-dimensional hypercube are then vertices whose symmetric difference has size 1: $E = \{(u,v)|u,v \subseteq [n], |u \oplus v| = 1\}$. This is the version of the condition that an edge in a hypercube exists between Boolean strings if they differ in exactly one position.

More generally, for sets $u \subseteq v$, define the interval of these sets as $[u : v] := \{w|u \subseteq w \subseteq v\}$. Then the cube spanned by these sets, $\mathcal{C}^{[u:v]}$, is defined as the graph with vertex set

$$V(\mathcal{C}^{[u:v]}) := [u : v] \tag{1}$$

and edge set

$$E(\mathcal{C}^{[u:v]}) := \{(s,t)|s,t \in [u : v], |s \oplus t| = 1\}. \tag{2}$$

A useful shorthand is $\mathcal{C}^u = \mathcal{C}^{[\varnothing:u]}$, in which case the n-dimensional hypercube is $\mathcal{C}^{[n]}$. Intervals are useful for defining

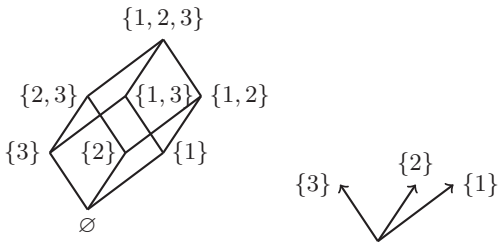

FIG. 1. Labeling of the vertices of a hypercube for $n = 3$ on the left and the different directions on the right.

subcubes. The cube $\mathcal{C}^{[u,v]}$ is a subcube of the cube $\mathcal{C}^{[u':v']}$ iff $[u,v] \subseteq [u' : v']$. Subcubes of a cube are often called faces and a face of dimension that is one less than the cube is called a facet. If one prefers to work over the Boolean string representation of a hypercube, then the vertices of a hypercube are those where a fixed set of character positions are fixed, and other are allowed to vary. The elements that are in v but not in u of a subcube $\mathcal{C}^{[u,v]}$ are exactly the locations where the characters are allowed to vary.

Edges of the n-dimensional hypercube are labeled in a natural way by their direction label. That is, for the edge $e = (u,v)$, there is a unique λ such that $\{\lambda\} = u \oplus v$. The set of all labels of a cube is called the carrier,

$$\text{carr}\mathcal{C}^{[u:v]} := v \setminus u. \tag{3}$$

For a given direction λ, we can split the cube into two subcubes, called the λ facets. In particular for the cube $\mathcal{C}^{[n]}$, the lower λ facet is $\mathcal{C}^{[n]\setminus\{\lambda\}}$ and the upper λ facet is $\mathcal{C}^{[\{\lambda\}:[n]]}$. Less formally, the condition is that when one considers only the orientations of the edges between the vertices in a subcube, this orientation has a unique sink vertex. See Fig. 1 for this labeling scheme and direction in the case of $n = 3$.

An orientation ϕ of a graph $G = (V,E)$ is a map from the edges E to the vertices V such that $\phi((v,w)) = v$ or $\phi((v,w)) = w$ for all $(v,w) \in E$. This maps edges to their corresponding sink vertex. Thus if $(v,w) \in E$ and $\phi((v,w)) = v$, then the orientation has the edge E directed from w to v. In this case, w is the source and v is the sink. Given this notion for a particular vertex, we can partition edges that contain the vertex into those for which the vertex is the sink (incoming edges) and those for which it is the source (outgoing edges).

Given an orientation ϕ of a cube $\mathcal{C}$, we define the outmap s of ϕ as the map that assigns to every vertex the labels of the outgoing edges from that vertex:

$$s : V(\mathcal{C}) \to 2^{\text{carr}\mathcal{C}}. \tag{4}$$

We can now define a USO of a cube $\mathcal{C}$. A USO of a cube $\mathcal{C}$ is an orientation ϕ such that every subcube has a unique sink. That is, an orientation ϕ with outmap s is a USO iff

$$\forall u,v C^{[u,v]} \tag{5}$$

is a unique sink orientation, where a subcube $C^{[u,v]}$ is a unique sink orientation if there exists a unique vertex w such that the outmap does not point along any of the outgoing directions in the carrier space for a subcube, $\text{carr}\mathcal{C}^{[u:v]}$,

$$\exists \text{ unique } w \text{ such that } \forall \lambda \in \text{carr}\mathcal{C}^{[u:v]}, s(w) \notin \lambda. \tag{6}$$

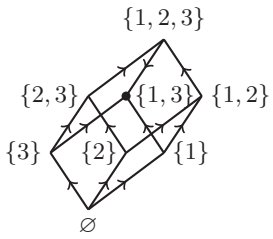

FIG. 2. Example of a USO. The unique sink is vertex {1,3} and the unique source is φ. Note, for instance, that this is a USO since all of its faces considered alone have a unique sink vertex. For example, the $\mathcal{C}^{[\varnothing:\{1,2\}]}$ cube has the unique sink of {1,2}.

See Fig. 2 for a picture of a USO for $n = 3$.

The central problem we would like to solve can now be introduced.

Problem 1. USO. Given a USO ϕ on a cube $\mathcal{C}^{[n]}$ and a subcube $\mathcal{C}^{[u:v]}$, determine by querying the outmap s of ϕ whether the unique sink of $\mathcal{C}^{[n]}$ lies in $\mathcal{C}^{[u:v]}$.

Note that this is a decision problem having a yes or no answer for each ϕ and subcube $\mathcal{C}^{[u:v]}$. However, it can be used to efficiently solve the search version of the problem, finding the unique sink vertex of the entire cube. To do this, one simply uses the decision version on facets of the cube, determining whether the unique sink is in one of two facets, and then recursively applying this procedure to the facet with the unique sink.

IV. PROPERTIES OF USO's

Here we recall certain properties of USOs on cubes that we will use. We present short proofs of these results for completeness.

If ϕ is an orientation with corresponding outmap s for a cube $\mathcal{C}$, then the orientation ϕ', in which for every $\lambda \in \Lambda \subseteq \text{carr}\mathcal{C}$ the λ edge of ϕ are reversed, has an outmap of $s'(v) = \Lambda \oplus s(v)$. If the original orientation ϕ is a USO, then, according to the following Lemma, the orientation ϕ' is also a USO.

Lemma 1. [18] Given a unique sink outmap s of a cube $\mathcal{C}^{[n]}$, and a set of directions, $\Lambda \subseteq \text{carr}\mathcal{C}^{[n]}$, the map defined by $s'(v) = \Lambda \oplus s(v)$ is the outmap of a USO.

Proof. We will first show that this is true for $\Lambda = \{\lambda\}$. We need to show that the new outmap s' corresponds to an orientation ϕ' that has the USO property. Let $\mathcal{C}_u$ and $\mathcal{C}_l$ denote the upper and lower λ facets of $\mathcal{C}^{[n]}$ respectively. Consider any subcube $\mathcal{C}$ of $\mathcal{C}^{[n]}$. If $\mathcal{C}$ is entirely within a λ facet, then the orientation obtained by reversing all edges along the λ direction does not change the orientation on any edges of $\mathcal{C}$. In this case, since ϕ has a unique sink on $\mathcal{C}$, so does ϕ' on $\mathcal{C}$. On the other hand, if $\mathcal{C}$ spans λ facets, then we can partition $\mathcal{C}$ into subcubes $\mathcal{D}_u = \mathcal{C} \cup \mathcal{C}_u$ and $\mathcal{D}_d = \mathcal{C} \cup \mathcal{C}_d$. Over the orientation ϕ suppose that these two subcubes have unique sinks o_u and o_d. Assume without loss of generality that the unique sink of ϕ over $\mathcal{C}$ is o_u. Flipping all of the edges along λ means that o_u is no longer a unique sink (since its λ edge now points away from it). However, o_d must now be a unique sink of $\mathcal{C}$ since prior to flipping the λ edge the only thing keeping o_d from being a unique sink of $\mathcal{C}$ was the λ edge. All other vertices cannot be unique sinks of $\mathcal{C}$ since they are not unique sinks in $\mathcal{D}_u$ or $\mathcal{D}_d$ and none of the edges in those subcubes are flipped. Hence, ϕ' is the outmap of a USO for $\Lambda = \{\lambda\}$. For the general case of $\Lambda = \{\lambda_1, \ldots, \lambda_k\}$, note that it follows from the application of the just proven case k times since $s'(v) = \Lambda \oplus s(v) = \{\lambda_1\} \oplus \cdots \oplus \{\lambda_k\} \oplus s(v)$.

The use of the above lemma is that it implies an important property of all outmaps of a USO of a cube.

Lemma 2. [19] The outmap of a USO of a cube $\mathcal{C}$ is a bijection.

Proof. Given a USO ϕ of a cube $\mathcal{C}$, suppose that there exist two vertices, $u \neq v$, which each have the same image under the outmap s, $s(v) = s(u) = t$. Then consider the orientation ϕ' obtained by flipping all of the edges along the t direction. This has outmap $s'(v) = t \oplus v$. Via Lemma 1, this new orientation is a USO. However, $s'(u) = s'(v) = \varnothing$, which is a contradiction. Hence there are not two vertices which have the same image under the outmap s.

V. QUANTUM APPROACH

In the last section, we have seen that the outmap of a USO is a bijection from vertices to the power set of the carrier space of the cube. Vertices are labeled by elements of $2^{[n]}$ and elements of the carrier space are also labeled by $2^{[n]}$. Viewed in this manner, we can map the carrier space back to the vertices, and hence we can view the outmap as a permutation on $2^{[n]}$. Given this view, we can then define the kth power of this map,

$$s^k := \underbrace{s \circ s \circ \cdots \circ s}_{k \text{ times}}, \tag{7}$$

i.e., the permutation s applied k times.

Consider the sequence

$$\varnothing, s(\varnothing), s^2(\varnothing), \ldots. \tag{8}$$

Because s is a bijection, this sequence is periodic with a period, at most, 2^n. That is, there exists a minimal $l > 0$ such that $s^k(\varnothing) = s^{k+l}(\varnothing)$.

Suppose that we could determine the minimal $l \neq 0$ such that $s^k(\varnothing) = s^{k+l}(\varnothing)$. Then, $s^l(\varnothing) = \varnothing$ and hence $s(s^{l-1}(\varnothing)) = \varnothing$, or, in other words, $s^{l-1}(\varnothing)$ is the unique sink of the cube. In other words, determining the period of s, and then evaluating s raised to that power minus one, yields the unique sink. The problem of finding the period of a function is one in which quantum computers offer an exponential advantage over classical computers, and this is the basis of Shor's algorithm for factoring [4,5].

In particular, this leads us to our main result. Let ϕ be a USO with outmap s. Let $\mathcal{H} = \mathbb{C}_2^n \otimes \mathbb{C}_2^n \otimes \mathbb{C}_2^n$ and label the basis of these states by $|k,u,v\rangle$ with $k \in \{0, \ldots, 2^n - 1\}$ and $u, v \in 2^{[n]}$. We say that a unitary U calculates the kth power of an outmap if it acts on this basis as

$$U|k,u,v\rangle = |k,u,v \oplus s^k(u)\rangle. \tag{9}$$

Theorem 1. Given a unitary U which can calculate the kth power of the outmap permutation of a USO, there exists a polynomial sized quantum algorithm that queries this oracle $O(1)$ times and identifies the unique sink with constant probability with a circuit of size $O(\text{poly}(n))$.

Proof. Straightforward application of Shor's period finding algorithm to the given oracle [4,5], followed by using the oracle once to calculate $s^{l-1}(\varnothing)$, provides the proof.

VI. EXPONENTIALLY LONG PERIODS

In the previous section, we showed that if one can efficiently determine the period of an outmap s of a USO of a cube and one can evaluate the outmap at this period minus one, then one can efficiently solve the USO problem. A question which arises is whether or not this is an efficient classical algorithm. A naive classical use of this period finding method would be to try to calculate $s(\varnothing), s^2(\varnothing), \ldots$ until the sequence repeats. Here we show that this approach fails because there are USOs where this sequence is exponentially long.

We will explicitly show an example of such a USO. Given two USOs of the same dimension n, ϕ_1, and ϕ_2, one can produce a new USO of dimension $n+1$ by using ϕ_1 as a lower $n+1$ face and ϕ_2 as the upper $n+1$ face and then directing all of the $n+1$ edges either from the lower to the upper face, or vice versa. Label these two cases $\phi_1 \uparrow \phi_2$ and $\phi_1 \downarrow \phi_2$, respectively. It is clear that the new orientation constructed in this manner is a USO since every face either lies entirely in ϕ_1 or ϕ_2 (and so is a USO), or the face spans ϕ_1 and ϕ_2 and hence each component in the upper and lower $n+1$ face has a unique sink so that the global USO exists and is uniquely the one in which the new $n+1$ orientation points.

Let ψ_1 be the one-dimensional orientation in which the single edge points from $\varnothing$ to $\{1\}$. Further, define the uniform orientation $u_{n,a}$ as the n-dimensional orientation whose outmap is $s(v) = v \oplus a$. Recursively define the orientation

$$\psi_{n+1} = \psi_n \downarrow u_{n-1,\varnothing}. \tag{10}$$

This orientation has ψ_n as its lower $n+1$ facet, the uniform orientation towards $\varnothing$ in its upper $n+1$ facet, and $n+1$ edges from the upper to the lower $n+1$ facet.

Lemma 3. Define $P(\phi)$ as the period of the outmap of ϕ, starting at $\varnothing$. Then,

$$P(\psi_{n+1}) = 2P(\psi_n), \tag{11}$$

and hence since $P(\psi_1) = 2$, $P(\psi_n) = 2^n$.

Proof. Let s_n denote the outmap of ψ_n. The outmap of ψ_{n+1} can be written as

$$s_{n+1}(v) = \begin{cases} \{n+1\} \cup s_n(v) & \text{if } n+1 \notin v \\ v \setminus \{n+1\} & \text{otherwise.} \end{cases} \tag{12}$$

From this it follows that the sequence $s_{n+1}(\varnothing)$, $s^2_{n+1}(\varnothing)$, $s^3_{n+1}(\varnothing)$, $s^4_{n+1}(\varnothing), \ldots$ is equal to $s_n(\varnothing) \cup \{n+1\}$, $s_n(\varnothing)$, $s^2_n(\varnothing) \cup \{n+1\}$, $s^2_n(\varnothing), \ldots$. Since s_n does not act on the $n+1$ element, the claim follows directly.

Because there exist USOs such as ϕ_n that have exponentially long periods, the naive classical algorithm for finding the period fails. Note that this does not mean that a classical approach based upon identifying the period will fail, only that the naive algorithm would take exponentially long. Perhaps USOs have structure which would allow for an efficient classical algorithm based upon finding the period of the outmap.

VII. THE MISSING PIECE

We have shown that the ability to efficiently calculate the kth power of a USO's outmap would lead to an efficient quantum algorithm for the problem and that the same is not necessarily true for a naive classical algorithm. In Shor's algorithm, the step we are replacing with the kth power of a USO is the evaluation of $r^x \bmod N$. This can be done by modular exponentiation (using the trick of repeated squaring). We have been unable to find an efficient way to calculate this kth power. In general, it would seem that such a procedure would need to rely on properties of USOs. Note that in general there is no procedure that efficiently calculates the power of an arbitrary unitary gate. This follows from [20] where it was shown that evaluating $\exp(-iHt)$ scales at least linearly in t, though this does not preclude such powering for known sets of unitaries (see also [21,22]). There are large classes of USOs which have structure because they arise from different algorithmic problems [23]. For example, those arising in the P-matrix linear complementarity problem are known to be a restricted class that is rather small relative to all USOs, and yet efficiently solving the USO arising from this problem would constitute a breakthrough. Focusing on these instances is suggested as an important future direction.

VIII. CONCLUSION

We have shown that the ability to efficiently calculate the kth power of a USO's outmap would imply that there is an efficient quantum algorithm for the USO problem. This approach uses a key component that is thought to be important for quantum speedups [24], i.e., period finding. A variety of open problems remain, even beyond the obvious of trying to efficiently power the outmap. One important question is whether there are classes of USOs for which there is a quantum speedup. For example, USOs that are decomposable are known to have query complexity $\Theta(n)$ [18,25]; is there a constant query quantum algorithm for these cases? Another interesting question is whether the query complexity of the USO is polynomial in an information theoretic sense.

[1] T. Szabó and E. Welzl, Unique sink orientations of cubes, in *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*, edited by B. Werner (IEEE, Piscataway, NJ, 2001), pp. 547–555.

[2] B. Gärtner, Combinatorial linear programming: Geometry can help, in *Randomization and Approximation Techniques in Computer Science*, edited by M. Luby, J. D. P. Rolim, and M. Serna (Springer, New York, 1998), pp. 82–96.

[3] B. Gärtner, The random-facet simplex algorithm on combinatorial cubes, Random Struct. Alg. **20**, 353 (2002).

[4] P. W. Shor, Algorithms for quantum computation: Discrete logarithms and factoring, in *35th Annual Symposium on*

Foundations of Computer Science, edited by S. Goldwasser (IEEE, Piscataway, NJ, 1994), pp. 124–134.

[5] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Rev. **41**, 303 (1999).

[6] R. W. Cottle, J.-S. Pang, and R. E. Stone, *The Linear Complementarity Problem*, Vol. 60 (SIAM, Philadelphia, PA, 1992).

[7] L. G. Khachiyan, Polynomial algorithms in linear programming, USSR Comput. Math. Math. Phys. **20**, 53 (1980).

[8] B. Gärtner and I. Schurr, Linear programming and unique sink orientations, in *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm*, edited by C. Stein (Society for Industrial and Applied Mathematics, Philadelphia, PA, 2006), pp. 749–757.

[9] B. Gärtner and L. Rüst, Simple stochastic games and p-matrix generalized linear complementarity problems, in *Fundamentals of Computation Theory*, edited by M. Liśkiewicz and R. Reischuk (Springer, New York, 2005), pp. 209–220.

[10] O. Svensson, Mean payoff games and linear complementarity, in *Proceedings of the ESSLLI*, edited by P. Egre and L. A. i Alemany (Citeseer, Aix-en-Provence, France, 2005).

[11] S. Vorobyov, Cyclic games and linear programming, Discrete Appl. Math. **156**, 2195 (2008).

[12] K. Fischer and B. Gärtner, The smallest enclosing ball of balls: Combinatorial structure and algorithms, Int. J. Comput. Geometry Applic. **14**, 341 (2004).

[13] B. Gärtner and A. Thomas, The complexity of recognizing unique sink orientations, in *32nd International Symposium on Theoretical Aspects of Computer Science*, edited by E. W. Mayr and N. Ollinger (Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Wadern, 2015), pp. 341–353.

[14] N. Megiddo, *A Note on the Complexity of P-matrix LCP and Computing an Equilibrium* (IBM Thomas J. Watson Research Division, San Jose, CA, 1988).

[15] J. Matousek and T. Szabó, Random edge can be exponential on abstract cubes, in *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science 2004*, edited by E. Upfal (IEEE, Piscataway, NJ, 2004), pp. 92–100.

[16] V. Bosshard, Classical and quantum algorithms for USO recognition, Master's thesis, ETH Zürich, 2015.

[17] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the Twenty-eighth Annual ACM symposium on Theory of Computing*, edited by G. Miller (ACM, New York, NY, 1996), pp. 212–219.

[18] I. A. Schurr, Unique sink orientations of cubes, Ph.D. thesis, ETH Zürich, 2004.

[19] K. W. Hoke, Completely unimodal numberings of a simple polytope, Discrete Appl. Math. **20**, 69 (1988).

[20] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, Efficient quantum algorithms for simulating sparse Hamiltonians, Commun. Math. Phys. **270**, 359 (2006).

[21] M. Soleimanifar and V. Karimipour, No-go theorem for iterations of unknown quantum gates, Phys. Rev. A **93**, 012344 (2016).

[22] Y. Atia and D. Aharonov, Fast-forwarding of Hamiltonians and exponentially precise measurements, arXiv:1610.09619 [quant-ph].

[23] J. Foniok, B. Gärtner, L. Klaus, and M. Sprecher, Counting unique-sink orientations, Discrete Appl. Math. **163**, 155 (2014).

[24] A. M. Childs and W. van Dam, Quantum algorithms for algebraic problems, Rev. Mod. Phys. **82**, 1 (2010).

[25] I. Schurr and T. Szabó, Finding the sink takes some time, in *Algorithms—ESA 2002*, edited by R. H. Möhring and R. Raman (Springer, New York, 2002), pp. 833–844.