Quantum Walk Search and Applications in Linear Algebra

Sebastian Dörn
Inst. für Theoretische Informatik
Universität Ulm
89069 Ulm, Germany

Thomas Thierauf
Fak. Elektronik und Informatik
HTW Aalen
73430 Aalen, Germany

{sebastian.doern,thomas.thierauf}@uni-ulm.de

Abstract

In this paper we present an application of the quantum walk search schema by Magniez et al. [MNRS07] for finding more than one solution of a search problem. We apply our quantum walk to matrix multiplication, thereby improving a result by Buhrman and Špalek [BS06]. Furthermore we give tight quantum query complexity bounds of some important linear algebra problems, like the determinant, rank, matrix inverse and the matrix power problem.

1 Introduction

Quantum walks are the quantum counterpart of Markov chains and random walks. A discrete quantum walk is a way of formulating local quantum dynamics on a graph. The walk takes discrete steps between neighbouring vertices and is a sequence of unitary transformations.

The research on quantum walks is concentrated on the discovery of properties of quantum walks and their algorithmic applications. Aharonov et al. [AAKV01] introduced quantum walks on graphs. They showed how fast quantum walks spread and proved lower bounds on the possible speed up by quantum walks for general graphs. Kempe et al. [KSW03] presented a quantum search algorithm based on a quantum walk in a hypercube. Ambainis [Amb04] constructed a fundamental quantum walk algorithm for the element distinctness problem. This was the first quantum walk algorithm that went beyond the capability of Grover search. Magniez et al. [MSS05] have used Ambainis algorithm for finding a triangle in a graph. Szegedy [Sze04] generalized the element distinctness algorithm of Ambainis to an arbitrary graph by using Markov chains. He showed that for a class of Markov chains, quantum walk algorithms are quadratically faster than the corresponding classical algorithms. Magniez and Nayak [MN05] used Szegedy's result for quantize a classical Markov chain for testing the commutativity of a black box group. Buhrman and Špalek [BS06] constructed a quantum algorithms for matrix multiplication and its verification. Magniez et al. [MNRS07] developed a new scheme for quantum search, based

on any ergodic Markov chain. Their work generalizes previous results by Ambainis [Amb04] and Szegedy [Sze04]. They extend the class of possible Markov chains and improve the quantum complexity. Dörn and Thierauf [DT07] presented the first application of this new quantum random walk technique for testing the associativity of a multiplication table.

The quantum walk search algorithm by Magniez et al. [MNRS07] finds only one solution of a search problem. In many practical applications we are interested in more solutions. In the first part of our paper we extend the quantum walk search schema of Magniez et al. [MNRS07] by presenting a quantum walk algorithm which finds more solutions. We analyse the quantum complexity of this quantum walk search for finding k solutions. Then we apply our result to matrix multiplication and improve an algorithm by Buhrman and Špalek [BS06].

In the second part of our paper, we present quantum query complexity bounds of some important linear algebra problems. We prove $\Omega(n^2)$ quantum query bounds for the verification of the determinant, rank, matrix inverse and the matrix power problem. Therewith follows that with quantum computation we can't speed up these linear algebra problems in the query model.

The paper is organized as follows: In Section 3 we present the quantum walk search schema for finding more than one solutions of a search problem. In Section 4 we give quantum algorithms for matrix multiplication. In Section 5 we compute tight quantum query complexity bounds of some important linear algebra problems.

2 Preliminaries

Notations. We denote with $[n]$ the set $\{1, 2, \ldots, n\}$. Let A be a matrix of dimension $n \times m$. For a subset $R \subseteq [n]$, let $A|_{R,*}$ the $|R| \times m$ sub-matrix of A restricted to the rows from R. Analogously, for every $S \subseteq [m]$, let $A|_{*,S}$ the $n \times |S|$ sub-matrix of A restricted to the columns from S.

Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs. The *graph categorical product* $G = (V, E) = G_1 \times G_2$ of G_1, G_2 is defined as follows: $V = V_1 \times V_2$, and $((g_1, g_2), (g_1', g_2')) \in E$ iff $(g_1, g_1') \in E_1$ and $(g_2, g_2') \in E_2$.

Quantum Query Model. In the query model, the input $x_1, \ldots, x_N$ is contained in a black box or oracle and can be accessed by queries to the black box. As a query we give i as input to the black box and the black box outputs x_i. The goal is to compute a Boolean function $f : \{0,1\}^N \to \{0,1\}$ on the input bits $x = (x_1, \ldots, x_N)$ minimizing the number of queries. The classical version of this model is known as decision tree.

The quantum query model was explicitly introduced by Beals et al. [BBCMW01]. In this model we pay for accessing the oracle, but unlike the classical case, we use the power of quantum parallelism to make queries in superposition. The state of the computation is represented by $|i, b, z\rangle$, where i is the query register, b is the answer register, and z is the working register.

A quantum computation with T queries is a sequence of unitary transfor-

mations

$$U_0 \to O_x \to U_1 \to O_x \to \ldots \to U_{T-1} \to O_x \to U_T,$$

where each U_j is a unitary transformation that does not depend on the input x, and O_x are query (oracle) transformations. The oracle transformation O_x can be defined as $O_x : |i, b, z\rangle \to |i, b \oplus x_i, z\rangle$.

The computation consists of the following three steps:

1. Go into the initial state $|0\rangle$.
2. Apply the transformation $U_T O_x \cdots O_x U_0$.
3. Measure the final state.

The result of the computation is the rightmost bit of the state obtained by the measurement.

The quantum computation determines f with bounded error, if for every x, the probability that the result of the computation equals $f(x_1, \ldots, x_N)$ is at least $1-\epsilon$, for some fixed $\epsilon < 1/2$. In the query model of computation each query adds one to the query complexity of an algorithm, but all other computations are free. The time complexity of the algorithm is usually measured in terms of the total circuit size for the unitary operations U_i.

Quantum Search. A search problem is a subset $S \subseteq [N]$ of the search space $[N]$. With S we associate its characteristic function $f_S : [N] \to \{0, 1\}$ with $f_S(x) = 1$ if $x \in S$, and 0 otherwise. Any $x \in S$ is called a solution to the search problem. Let $k = |S|$ be the number of solutions of S. It is a well known fact in quantum computing (see [Gro96, BBHT98]), that for $k > 0$, the expected quantum query complexity for finding one solution of S is $O(\sqrt{N/k})$, and for finding all solutions, it is $O(\sqrt{kN})$. Furthermore, whether $k > 0$ can be decided in $O(\sqrt{N})$ quantum queries to f_S.

3 Searching with Quantum Walks

An important tool for quantum searching are quantum walks. Quantum walks are the quantum counterpart of Markov chains and random walks. The quantum walk search provide a promising source for new quantum algorithms, like element distinctness algorithm [Amb04], triangle finding [MSS05], testing group commutativity [MN05], matrix verification [BS06] and testing associativity [DT07].

Let $P = (p_{xy})$ be the transition matrix of an ergodic symmetric Markov chain on the state space X. Let $M \subseteq X$ be a set of marked states. Assume that the search algorithms use a data structure D that associates some data $D(x)$ with every state $x \in X$. From $D(x)$, we would like to determine if $x \in M$. When operating on D, we consider the following three types of cost:

- *Setup cost s*: The worst case cost to compute $D(x)$, for $x \in X$.
- *Update cost u*: The worst case cost for transition from x to y, and update $D(x)$ to $D(y)$.

- *Checking cost c*: The worst case cost for checking if $x \in M$ by using $D(x)$.

Magniez et al. [MNRS07] developed a new scheme for quantum search, based on any ergodic Markov chain. Their work generalizes previous results by Ambainis [Amb04] and Szegedy [Sze04].

Theorem 3.1 [MNRS07] *Let $\delta > 0$ be the eigenvalue gap of a ergodic Markov chain P and let $\frac{|M|}{|X|} \geq \epsilon$. Then there is a quantum algorithm that determines if M is empty or finds an element of M with cost*

$$s + \frac{1}{\sqrt{\epsilon}} \left(\frac{1}{\sqrt{\delta}} u + c \right).$$

The main idea of this quantum walk search is the application of the quantum phase estimation [CEMM98] to the quantum walk operator in order to implement an approximate reflection operator. This operator is then used in an amplitude amplification scheme [BHMT00].

In most applications the quantum walk takes place on the Johnson graph $J(n, r)$, which is defined as follows: the vertices are subsets of $\{1, \ldots, n\}$ of size r, and two vertices are connected iff they differ in exactly one number. It is well known, that the spectral gap δ of $J(n, r)$ is $\Theta(1/r)$ for $1 \leq r \leq \frac{n}{2}$ (see e.g. [Knu03]).

The quantum walk search algorithm of Theorem 3.1 finds only one solution of a search problem. In many practical applications we are interested in more solutions. In this section we apply the above quantum walk search schema to find more solutions of a search problem in a Johnson graph.

Theorem 3.2 *Let $S \subseteq [n]$ be a search problem and let P be a random walk on the Johnson graph $J(n, r)$, where $r = o(n)$. Let M be the class of all r-subsets that contain a solution of S. Then there is a quantum algorithm that finds up to k of the solutions with cost*

$$\begin{cases} O\left(s \cdot k + \sqrt{k \cdot \frac{n}{r}} \left(\sqrt{r} u + c\right)\right), & \frac{kr}{n} \leq 1, \\ O\left(s \cdot k + \sqrt{k \cdot \frac{n}{r}} \log n \left(\sqrt{r} u + c\right)\right), & \frac{kr}{n} > 1. \end{cases}$$

Proof. Suppose our search problem contains at most k different solution. We use the quantum walk search schema of Theorem 3.1. The result of this quantum search is an element of the marked states M. The marked state contains a solution x of our search problem. We store this element x in a list. Now we use the quantum walk search schema again for finding another solution. For this task, we modify the oracle in the amplitude amplification, such that a state of the Johnson graph is marked, if it contains a solution which is not yet in the list. We repeat this quantum walk search step $k - 1$ times. The result of this procedure is a list with k different solutions. In case that there are only $l < k$ solutions, possibly $l = 0$, then the algorithm will detect this after the $(l + 1)$-th iteration and output the l solutions found.

Now we compute the quantum cost of this search algorithm. In the $(k - i + 1)$-th iteration of our algorithm, the search problem contains i different solutions. Let M_i be the set of marked states of the Johnson graph with

state space X, when there are i different solutions. Let furthermore $\epsilon_i := \frac{|M_i|}{|X|}$. By Theorem 3.1, the cost for finding k solutions is

$$s \cdot k + \sum_{i=0}^{k-1} \frac{1}{\sqrt{\epsilon_{k-i}}} \left(\frac{1}{\sqrt{\delta}} u + c \right) = s \cdot k + \Delta_\epsilon \left(\frac{1}{\sqrt{\delta}} u + c \right)$$

where

$$\Delta_\epsilon := \frac{1}{\sqrt{\epsilon_1}} + \ldots + \frac{1}{\sqrt{\epsilon_k}}.$$

The eigenvalue gap δ of the Johnson graph is $\Theta(1/r)$ for $1 \leq r \leq \frac{n}{2}$, therefore the cost is bounded by

$$s \cdot k + \Delta_\epsilon \left(\sqrt{r} u + c \right).$$

Now we compute the value of Δ_ϵ. The number of states in the Johnson graph $J(n, r)$ is $|X| = \binom{n}{r}$. If there is one solution, then there are $\binom{n-1}{r-1}$ marked vertices. Suppose there are i solutions of the search problem, then there are

$$|M_i| = \binom{n}{r} - \binom{n-i}{r}$$

marked vertices in the Johnson graph. Then we obtain

$$\epsilon_i = \frac{|M_i|}{|X|} = 1 - \frac{\binom{n-i}{r}}{\binom{n}{r}} \geq 1 - \left(\frac{n-i}{n} \right)^r.$$

Since $(1 - \frac{x}{n})^n \leq e^{-x}$, we get

$$\epsilon_i \geq 1 - e^{-\frac{ir}{n}}.$$

Now we get an upper bound for Δ_ϵ

$$\Delta_\epsilon \;\leq\; \sum_{i=1}^{k} \frac{1}{\sqrt{1 - e^{-\frac{ir}{n}}}} \;\leq\; \int_1^{k+1} \frac{1}{\sqrt{1 - e^{-\frac{ir}{n}}}}\, di \;=\; \frac{2n}{r} \operatorname{arctanh} \sqrt{1 - e^{-\frac{ir}{n}}} \Big|_1^{k+1}.$$

If $\frac{kr}{n} \leq 1$, then $\operatorname{arctanh} \sqrt{1 - e^{-\frac{kr}{n}}} \in O\left(\sqrt{\frac{kr}{n}} \right)$, by using the definition of arctanh and a simple analysis estimation. Therefore it holds

$$\Delta_\epsilon \leq O\left(\frac{n}{r} \cdot \sqrt{\frac{kr}{n}} \right) = O\left(\sqrt{\frac{kn}{r}} \right).$$

Otherwise, if $\frac{kr}{n} > 1$, then $\operatorname{arctanh} \sqrt{1 - e^{-\frac{kr)}{n}}} \in O\left(\sqrt{\frac{kr}{n}} \log\left(\frac{kr}{n} \right) \right)$, then

$$\Delta_\epsilon \leq O\left(\sqrt{\frac{kn}{r}} \log n \right).$$

□

Now we use the idea of the proof of Theorem 3.2 and combine it with the bound on the number of marked state in the graph categorical product of two Johnson graphs shown in [BS06].

Theorem 3.3 *Let $S \subseteq [n] \times [n]$ be a search problem and let P be a random walk on $J(n,r) \times J(n,r)$. Let M be the class of all $(r \times r)$-subsets that contain a solution of S. Then there is a quantum algorithm that finds up to k of the solutions with $r \leq n^{2/3}/\min(k,\sqrt{n})^{1/3}$ and cost*

$$\begin{cases} O\left(s \cdot k + \frac{n}{r}\sqrt{k}\left(\sqrt{r}u + c\right)\right), & k \leq \sqrt{n}, \\ O\left(s \cdot k + \frac{n^{3/4}}{r}k\left(\sqrt{r}u + c\right)\right), & k > \sqrt{n}. \end{cases}$$

Proof. The eigenvalue gap δ of $J(n,r) \times J(n,r)$ is $\Theta(1/r)$, for $1 \leq r \leq \frac{n}{2}$ (see [BS06]). Therefore the cost for finds up to k of the solutions is bounded by

$$s \cdot k + \Delta_\epsilon \left(\sqrt{r}u + c\right),$$

where $\Delta_\epsilon := \frac{1}{\sqrt{\epsilon_1}} + \ldots + \frac{1}{\sqrt{\epsilon_k}}$. Now we use the estimation of the value of ϵ_i by [BS06], which holds also for the general search problem in the graph categorical product of two Johnson graphs $J(n,r)$. Therefore

$$\epsilon_i = \Omega\left(\frac{r^2}{n^2}q_i\right),$$

where $q_i \geq \min(i, \sqrt{n})$. Then we get

$$\Delta_\epsilon \leq \sum_{i=1}^{\sqrt{n}} \frac{n}{r\sqrt{i}} + \sum_{i=\sqrt{n}+1}^{k} \frac{n}{rn^{\frac{1}{4}}} \leq \begin{cases} \frac{n}{r}\sqrt{k}, & k \leq \sqrt{n}, \\ \frac{n^{3/4}}{r}k, & k > \sqrt{n}. \end{cases}$$

□

4 Matrix Multiplication

In the matrix multiplication problem we have given two $n \times n$ matrices A and B over any integral domain. The task is to compute the product $C = AB$. The fastest known classical algorithm for computing the product of two matrices works in time $O(n^{2.376})$, see Coppersmith and Winograd [CW90]. Buhrman and Špalek [BS06] presented a quantum algorithm which is faster when the number of nonzero elements of the product matrix is $o(n^{0.876})$. The worst case quantum query complexity of their algorithm is $n^{5/3}w \log n$, where w is the number of nonzero entries of the matrix $C = AB$. The expected quantum time complexity of their algorithm is

$$\begin{cases} O(n^{5/3}w^{2/3} \log n), & 1 \leq w \leq \sqrt{n} \\ O(n^{3/2}w \log n), & \sqrt{n} \leq w \leq n \\ O(n^2\sqrt{w} \log n), & n \leq w \leq n^2. \end{cases}$$

We present a quantum algorithm that uses Theorem 3.3, which improves the worst case quantum query complexity of [BS06]. For $1 \leq w \leq n$, the worst case complexity of our algorithm is even better than the expected time complexity of [BS06] by a logarithmic factor. We formulate our theorem in terms of the query complexity. By additionally multiplying with random verctors (see [BS06]) one can achieve the same time complexity as the query complexity.

Theorem 4.1 *There is a quantum algorithm for the matrix multiplication problem with query complexity of*

$$\begin{cases} O(n^{5/3}w^{2/3}), & 1 \le w \le \sqrt{n}, \\ O(n^{3/2}w), & \sqrt{n} < w \le n^2, \end{cases}$$

where w is the number of nonzero entries of the product matrix C.

Proof. Given two $n \times n$ matrices A and B, we want to compute the matrix $C = AB$. At the beginning we set C to the zero-matrix. Our algorithm consists of two main steps. In the first step we search for all wrong entries in the matrix C. In the second step we recompute all wrong entries.

For the first step, we apply Theorem 3.3 for finding all nonzero entries in C. Let R, S be two subsets of $[n]$ of size r. We will determine r later. The database is the $r \times r$ matrix $D(R,S) = A|_{R,*} \cdot B|_{*,S}$. The quantum walk takes place on the graph categorical product of two Johnson graphs $J = J(n,r) \times J(n,r)$. The marked vertices of J correspond to pairs (R,S) with $A|_{R,*} \cdot B|_{*,S} \neq C|_{R,S}$. In every step of the walk, we exchange one row and one column of R and S. The setup query cost for the database is $O(rn)$ and the update query cost is $O(n)$. For checking if a vertex is marked, we use Grover search for finding an entry (i,j) with $(A|_{R,*} \cdot B|_{*,S} - C|_{R,S})_{i,j} \neq 0$. Therefore the checking cost is $O(r)$.

Suppose $w \le \sqrt{n}$, then the quantum query complexity of this step is

$$O\left(rnw + \frac{n}{r}\sqrt{w}\left(\sqrt{r}n + r\right)\right).$$

Let $r = \frac{n^{2/3}}{w^{1/3}}$, then we satisfy the condition $r \le n^{2/3}/\min(w, \sqrt{n})^{1/3}$ of Theorem 3.3, since $\min(w, \sqrt{n}) \le w$. Since we do not know the number w of nonzero entries, we search in ascending order for $1, 2, 4, \ldots, 2^{\log w - 1}$ nonzero entries. Then the total query complexity for $w \le \sqrt{n}$ of this iteration is

$$O\left(\sum_{i=0}^{\log w - 1} n^{5/3} \cdot 2^{2i/3}\right) = O(n^{5/3}w^{2/3}).$$

Otherwise, if $w > \sqrt{n}$, then the quantum query complexity is

$$O\left(rnw + \frac{n^{3/4}}{r}w\left(\sqrt{r}n + r\right)\right) = O\left(rnw + \frac{n^{7/4}}{\sqrt{r}}w\right) = O(n^{3/2}w)$$

for $r = \sqrt{n}$. The value of r satisfy the condition of Theorem 3.3. In the second step of our algorithm we recompute all wrong entries of C, this can be done in $O(nw)$ queries. □

5 Quantum Query Complexity of some Linear Algebra Problems

In this section we present tight bounded error quantum query complexity bounds of some important linear algebra problems.

5.1 Matrix Power Verification

Now we determine the quantum query complexity of the *matrix power* resp. *matrix power element problem*. In the matrix power problem we have given two $n \times n$ matrices A and B and an integer m, decide whether $A^m = B$. In the matrix power element problem we have given a $n \times n$ matrix A and integers i, j, a, m, decide if $(A^m)_{i,j} = a$. We show that the quantum query complexity of this problem is $\Theta(n^2)$. For this task, we define the following problem for n variables $x_1, \ldots, x_n \in \{0, 1\}$ and $0 \leq a \leq n$:

$$\textsc{Exact}_n(x_1, \ldots, x_n, a) := \begin{cases} 1, \text{ if } \sum_{i=1}^{n} x_i = a, \\ 0, \text{ otherwise.} \end{cases}$$

Using the quantum adversary lower bound method [Amb02], we have

Lemma 5.1 *The quantum query complexity of* $\textsc{Exact}_n$ *is* $\Theta(n)$.

We show in the following how to reduce $\textsc{Exact}_{n^2}$ to matrix power, in fact, power of 3.

Theorem 5.2 *The quantum query complexity of the matrix power and the matrix power element problem is* $\Theta(n^2)$. *This already holds for powers of 3.*

Proof. Given $x_1, \ldots, x_{n^2} \in \{0, 1\}$ and $0 \leq a \leq n^2$ as input for $\textsc{Exact}_{n^2}$, we construct a directed graph G as follows. G has $2n + 2$ nodes. With nodes $1, \ldots, 2n$ we construct a bipartite graph with nodes $1, \ldots, n$ on the left side and nodes $n + 1, \ldots, 2n$ on the right side. For the edges, we consider the variables x_k. Index k can be uniquely written as $k = (i - 1)n + j$, for $1 \leq i, j \leq n$. Edge $(i, n + j)$ is present in G iff $x_k = 1$. For the remaining two nodes, let $s = 2n + 1$ and $t = 2n + 2$. Add edges from s to all the nodes $1, \ldots, n$ and edges from all nodes $n + 1, \ldots, 2n$ to t. This completes the construction of graph G.

Observe that all paths from s to t in G have length 3 and each such path uniquely corresponds to a variable x_k with value 1. Moreover, there are no further paths of length 3 in G. Let A be the adjacency matrix of G. We conclude that the entry (s, t) of A^3 is the number of paths from s to t in G, and all other entries are 0. Hence we have

$$\textsc{Exact}_{n^2}(x_1, \ldots, x_{n^2}, a) = 1 \iff (A^3)_{s,t} = a \iff A^3 = B,$$

where matrix B has (s, t) entry a and 0 elsewhere. □

5.2 Determinant, Rank and Inverse Verification

Next, we consider the *determinant* and *inverse problem*. In the determinant problem we have given a $n \times n$ matrix A, decide whether $\det(\mathrm{A}) = 0$. In the inverse problem we have given a regular $n \times n$ matrix A and integers i, j, a. One has to decide whether $(A^{-1})_{i,j} = a$.

Theorem 5.3 *The quantum query complexity of the determinant and the inverse problem is* $\Theta(n^2)$.

Proof. We slightly modify the standard reduction from matrix power element to the determinant. Let $A = (a_{i,j})$ be a $n \times n$ matrix and a be given. By Theorem 5.2, we may assume that A is a 0-1-matrix and we consider the problem whether $(A^3)_{1,n} = a$. We will construct a matrix B such that $\left(A^3\right)_{1,n} = \det(B)$.

Interpret A as representing a directed bipartite graph on $2n$ nodes. That is, the nodes are arranged in two columns of n nodes each. In both columns, nodes are numbered from 1 to n. If $a_{i,j} = 1$ then we put an edge from node i in the first column to node j in the second column.

Now, take 3 copies of this graph, put them in a sequence and identify each second column of nodes with the first column of the next graph in the sequence. We have 4 columns of n nodes each so far. Now we add a 5-th column of n nodes as well and connect it by horizontal edges with the 4-th column.

Call the resulting graph G'. Graph G' has $N = 5n$ nodes, and the entry at position $(1, n)$ in A^3 is the number of paths in G' from node 1 in the first column to node n in the last column. Call these two nodes s and t, respectively.

Next, we add an edge from t to s and put self-loops at all nodes except s and t. Call the resulting graph G and let B be the adjacency matrix of G. From combinatorial matrix theory we know that the determinant of B is the signed sum of cycle covers of G. Any cycle cover of G consists of one cycle of length 5 which goes from s to t via the columns in G' and then back to s. The remaining cycles of the cover are self-loops. Therefore each cycle cover corresponds to one path from s to t in G'. The sign of the cycle cover is $(-1)^{N+k}$, where k is the number of cycles in the cover. We have $k = N - 4$. Therefore the sign is 1. We conclude that $\det(B) = \left(A^3\right)_{1,n}$.

Note that the size of B is linear in the size of A. Therefore the lower bound for matrix powering carries over to the determinant.

In order to get a reduction to the matrix inverse problem, we modify graph G from above and add self-loops to nodes s and t. Let H be the resulting graph and $C = (c_{i,j})$ be the adjacency matrix of H. The identity permutation is an additional cycle cover of H compared to G. Therefore we have $\det(C) = \left(A^3\right)_{1,n} + 1$. If C has no inverse, then we have $\det(C) = 0$ and consequently $\left(A^3\right)_{1,n} = -1$. In the following, assume that C has an inverse.

Note that $c_{i,i} = 1$ since all nodes have self-loops. Furthermore, with the convention $s = 1$ and $t = N$ we have $c_{N,1} = 1$ because of the edge from t to s. Note that except for position $(N, 1)$ matrix C is an upper triangular matrix. We consider the Laplace expansion of $\det(C)$. Let $C_{i,j}$ denote the matrix obtained from C by deleting row i and column j. We consider the expansion for the first column:

$$\begin{aligned}\det(C) &= c_{1,1} \det(C_{1,1}) + (-1)^{N+1} \det(C_{N,1}) \\ &= 1 + (-1)^{N+1} \det(C_{N,1}),\end{aligned}$$

because $\det(C_{1,1}) = 1$. Let c be the entry at position $(N, 1)$ in C^{-1}. By Cramers rule we have $c = \det(C_{N,1}) / \det(C)$. Hence we get

$$\det(C) = 1 + (-1)^{N+1} c \cdot \det(C).$$

We replace $\det(C)$ by $\left(A^3\right)_{1,n} + 1$ and obtain

$$\left(A^3\right)_{1,n} ((-1)^{N+1} - c) = c.$$

It follows that $((-1)^{N+1} - c)$ is non-zero and we have

$$\left(A^3\right)_{1,n} = \frac{c}{(-1)^{N+1} - c}.$$

The size of C is linear in the size of A. Therefore the lower bound for matrix powering carries over to the inverse. □

In the *rank problem* we have given an $n \times n$ matrix A and integer k. One has to decide whether $\text{rank}(A) = k$. For $k = n$ we have $\text{rank}(A) = n \iff \det(A) \neq 0$. Therefore the quantum query complexity of the rank problem follows from the determinant problem.

Corollary 5.4 *The quantum query complexity of the rank problem is* $\Theta(n^2)$.

Open problems

From the $\Omega(n^2)$ quantum query lower bound for the determinant it is tempting to conjecture the same bound for the perfect matching problem. There is a quantum only lower bound of $\Omega(n^{1.5})$ (see [Zha04]).

Buhrman and Špalek [BS06] presented an $O(n^{5/3})$ quantum time algorithm for the verification problem. The known lower bound is $\Omega(n^{3/2})$. It would be interesting to close this gap.

Acknowledgments

We thank Thanh Minh Hoang for helpful discussions and the referees of the paper for valuable hints that simplified some computations.

References

[Amb02] A. Ambainis, *Quantum Lower Bounds by Quantum Arguments*, Journal of Computer and System Sciences 64: pages 750-767, 2002.

[AAKV01] A. Aharonov, A. Ambainis, J. Kempe, U. Vazirani, *Quantum walks on graphs*, Proceedings of STOC'01: pages 50-59, 2001.

[Amb03] A. Ambainis, *Quantum walks and their algorithmic applications*, International Journal of Quantum Information 1: pages 507-518, 2003.

[Amb04] A. Ambainis, *Quantum walk algorithm for element distinctness*, Proceedings of FOCS'04: pages 22-31, 2004.

[BBCMW01] R. Beals, H. Buhrman, R. Cleve, M. Mosca, R. de Wolf, *Quantum lower bounds by polynomials*, Journal of ACM 48: pages 778-797, 2001.

[BBHT98] M. Boyer, G. Brassard, P. Høyer, A. Tapp, *Tight bounds on quantum searching*, Fortschritte Der Physik 46(4-5): pages 493-505, 1998.

[BHMT00] G. Brassard, P. Hóyer, M. Mosca, A. Tapp, *Quantum amplitude amplification and estimation*, In Quantum Computation and Quantum Information: A Millennium Volume, AMS Contemporary Mathematics Series, 2000.

[BS06] H. Buhrman, R. Špalek, *Quantum Verification of Matrix Products*, Proceedings of SODA'06: pages 880-889, 2006.

[CEMM98] R. Cleve, A. Ekert, C. Macchiavello, M. Mosca, *Quantum algorithms revisited*, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 454(1969): pages 339-354, 1998.

[CW90] D. Coppersmith, S. Winograd, *Matrix multiplication via arithmetic progressions*, Journal of symbolic computation 9: pages 251-280, 1990.

[DT07] S. Dörn, T. Thierauf, *The Quantum Query Complexity of Algebraic Properties*, Proceedings of FCT'07, 2007.

[Gro96] L. Grover, *A fast mechanical algorithm for database search*, Proceedings of STOC'96: pages 212-219, 1996.

[Knu03] D.E. Knuth, *Combinatorial matrices*, Selected Papers on Discrete Mathematics, volume 106 of CSLI Lecture Notes, Stanford University, 2003.

[KSW03] J. Kempe, N. Shenvi, K.B. Whaley, *Quantum Random-Walk Search Algorithm*, Physical Review Letters A, Vol. 67 (5), 2003.

[MN05] F. Magniez, A. Nayak, *Quantum complexity of testing group commutativity*, Proceedings of ICALP'05: pages 1312-1324, 2005.

[MNRS07] F. Magniez, A. Nayak, J. Roland, M. Santha, *Search via Quantum Walk*, Proceedings of STOC'07, 2007.

[MSS05] F. Magniez, M. Santha, M. Szegedy, *Quantum Algorithms for the Triangle Problem*, Proceedings of SODA'05: pages 1109-1117, 2005.

[NC03] M.A. Nielsen, I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, 2003.

[Sze04] M. Szegedy, *Quantum speed-up of Markov chain based algorithms*, Proceedings of FOCS'04: pages 32-41, 2004.

[Zha04] S. Zhang, *On the power of Ambainis's lower bounds*, Proceedings of ICALP'04, Lecture Notes in Computer Science 3142: pages 1238-1250, 2004.