Quantum Algorithms for Graph Traversals and Related Problems

Sebastian Dörn

Institut für Theoretische Informatik, Universität Ulm, 89069 Ulm, Germany
sebastian.doern@uni-ulm.de

Abstract. We study the complexity of algorithms for graph traversal problems on quantum computers. More precisely, we look at eulerian tours, hamiltonian tours, travelling salesman problem and project scheduling. We present quantum algorithms and quantum lower bounds for these problems. Our results improve the best classical algorithms for the corresponding problems. In particular, we prove that the quantum algorithms for the eulerian tour and the project scheduling problem are optimal in the query model.

1 Introduction

Quantum computation has the potential to demonstrate that for some problems quantum computation is more efficient than classical computation. The goal of quantum computing is to determine when quantum computers provide a speed-up over classical computers. Today, two main complexity measures for quantum algorithms have been studied: the quantum query and the quantum time complexity. The quantum query complexity of a quantum algorithm $\mathcal{A}$ is the number of quantum queries to the input, and the quantum time complexity of $\mathcal{A}$ is the number of basic quantum operations made by $\mathcal{A}$.

In this paper we are interested in graph traversal problems. The study of the quantum complexity for graph problems is an active topic in quantum computing. Dürr, Heiligman, Høyer and Mhalla [DHHM04] presented optimal quantum query algorithms for the minimum spanning tree, graph connectivity, strong graph connectivity and the single source shortest path problem. Magniez, Santha and Szegedy [MSS05] constructed a quantum query algorithm for finding a triangle in a graph. Polynomial-time quantum algorithms are given by Ambainis and Špalek [AS06] for the maximum matching and the network flow problem. Dörn [Doe07] presented quantum algorithms for several independent set problems.

In this paper, we present quantum algorithms and quantum query lower bounds for graph traversals and related problems. Our input is a directed or undirected graph with n vertices and m edges. We consider two query models for graphs: the adjacency matrix and the adjacency list model. In section 3 we study the quantum query and the quantum time complexity of the *eulerian graph problem*. In the eulerian graph problem we have to decide if a graph G has an

eulerian cycle, this is a closed walk that contains every edge of G exactly once. We compute the quantum query complexity of the eulerian graph problem in the adjacency matrix and the list model. Futhermore, we show that our lower and upper bounds are tight in both graph representation models.

In section 4 we consider the *hamiltonian cycle problem*, an important **NP**-complete graph problem. A hamiltonian cycle of a graph G is a cycle which contains all the vertices of G exactly once. Berzina et al. [BDFLS04] proved, that the hamiltonian cycle problem requires $\Omega(n^{1.5})$ quantum queries to the adjacency matrix. We show an $O(n^{2n/(n+1)})$ quantum query upper bound for this problem, by using a recent quantum walk technique.

In section 5 we study the *travelling salesman problem* in graphs with maximal degree three, four and five. Eppstein [Epp03] constructed algorithms for the travelling salesman problem on graphs with bounded degree three and four which are faster than $O(2^n)$. We analyse the quantum time complexity of these algorithms. We show that with a quantum computer we can solve the travelling salesman problem on graphs with maximal degree three, four and five quadratically faster than in the classical case.

In section 6 we consider a project scheduling problem. A digraph model can be used to schedule projects consisting of several interrelated tasks. Some of these tasks can be executed simultaneously, but some tasks cannot begin until certain others are completed. The goal is to compute the minimal project completion time. We present an optimal quantum query algorithm for computing the earliest completion time for every vertex of the network.

2 Preliminaries

2.1 Graph Theory

We denote by $[n]$ the set $\{1, 2, \ldots, n\}$. Let $G = (V, E)$ be a undirected or directed graph, with $V = V(G)$ and $E = E(G)$ we denote the set of vertices and edges of G. Let $n = |V|$ be the number of vertices and $m = |E|$ the number of edges of G. We denote by (u, v) a directed edge in G from vertex u to vertex v; the vertex u is called *adjacent* to vertex v in G. The number of vertices adjacent to v is called the *out-degree* of v and denote by $d^+_G(v)$. The *in-degree* of a vertex v is the number of edges directed to v, denoted by $d^-_G(v)$. A *cycle* of G is a sequence $(v_1, \ldots, v_k, v_1)$ where $k \geq 3$ and $v_1, \ldots, v_k$ are distinct vertices of G such that $(v_i, v_{i+1}) \in E$ for $i \in [k-1]$ and $(v_k, v_1) \in E$.

We consider the following two models for accessing information in digraphs:

- *Adjacency matrix model:* Given is the adjacency matrix $A \in \{0,1\}^{n \times n}$ of G with $A_{i,j} = 1$ iff $(i, j) \in E$. Weighted graphs are encoded by a weight matrix, where $A_{i,j}$ is the weight of edge (i, j) and for convenience we set $A_{i,j} = \infty$ if $(i, j) \notin E$.
- *Adjacency list model:* Given are the out-degrees $d^+_G(1), \ldots, d^+_G(n)$ of the vertices and for every $i \in V$ an array with its neighbours $f_i : [d^+_G(i)] \to [n]$. The value $f_i(j)$ is the j-th neighbour of i. Weighted graphs are encoded by a

sequence of functions $f_i : [d_G^+(i)] \to [n] \times \mathbb{N}$, such that if $f_i(j) = (i', w)$ then there is an edge (i, i') with weight w and i' is the j-th neighbour of i.

In undirected graphs, we replace the directed edge (u, v) by an undirected edge $\{u, v\}$, and the out-degree $d_G^+(i)$ through the degree $d_G(i)$ of the every $i \in V$.

2.2 Quantum Computing

For the basic notation on quantum computing, we refer the reader to the textbook by Nielsen and Chuang [NC03]. For the quantum algorithms included in this paper we use the following two complexity measures:

- The *quantum query complexity* of a graph algorithm $\mathcal{A}$ is the number of queries to the adjacency matrix or to the adjacency list of the graph made by $\mathcal{A}$.
- The *quantum time complexity* of a graph algorithm $\mathcal{A}$ is the number of basic quantum operations made by $\mathcal{A}$.

Now we give three tools for the construction of our quantum algorithms.

Quantum Search. A search problem is a subset $P \subseteq [N]$ of the search space $[N]$. With P we associate its characteristic function $f_P : [N] \to \{0, 1\}$ with $f_P(x) = 1$ if $x \in P$, and 0 otherwise. Any $x \in P$ is called a solution to the search problem. Let $k = |P|$ be the number of solutions of P.

Theorem 1. [Gro96,BBHT98] *For $k > 0$, the expected quantum query complexity for finding one solution of P is $O(\sqrt{N/k})$, and for finding all solutions, it is $O(\sqrt{k \cdot N})$. Futhermore, whether $k > 0$ can be decided in $O(\sqrt{N})$ quantum queries to f_P.*

Theorem 2. [DH96] *There is a quantum algorithm for finding the maximum element in a set of N real numbers with query complexity of $O(\sqrt{N})$.*

Amplitude Amplification. Let $\mathcal{A}$ be an algorithm for a problem with small success probability at least ϵ. Classically, we need $\Theta(1/\epsilon)$ repetitions of $\mathcal{A}$ to increase its success probability from ϵ to a constant, for example 2/3. The corresponding technique in the quantum case is called amplitude amplification.

Theorem 3. [BHMT00] *Let $\mathcal{A}$ be a quantum algorithm with one-sided error and success probability at least ϵ. Then there is a quantum algorithm $\mathcal{B}$ that solves $\mathcal{A}$ with success probability 2/3 by $O(\frac{1}{\sqrt{\epsilon}})$ invocations of $\mathcal{A}$.*

Quantum Walk. Quantum walks are the quantum counterpart of Markov chains and random walks. The quantum walk search provide a promising source for new quantum algorithms, see [Amb04], [MSS05], [MN05] and [BS06].

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]. They extend the class of possible Markov chains and improve the query complexity as follows.

Theorem 4. [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*

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

In the most practical application ([Amb04], [MSS05]) 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 $1/r$.

Remark 1. Our quantum algorithms output an incorrect answer with a constant probability p. If we want to reduce the error probability to less than ϵ, we repeat each quantum subroutine l times, where $p^l \leq \epsilon$. It follows, that we have to repeat each quantum subroutine $l = O(\log n)$ times, to make the probability of a correct answer greater than $1 - 1/n$. This increases the running time of all our algorithms by a logarithmic factor. Furthermore, the running time of Grover search is bigger that its query complexity by another logarithmic factor.

3 Eulerian Graph Problem

In this section we consider the eulerian graph problem. Given an undirected graph G, decide if G has a closed walk that contains every edge of G once. We denote such a walk as eulerian tour. It is a well known fact in graph theory, that a graph G is eulerian iff the degree of every vertex in G is even.

Theorem 5. *The quantum query complexity of the eulerian graph problem is $O(\sqrt{n})$ in the adjacency list and $O(n^{1.5})$ in the adjacency matrix model.*

Proof. In the adjacency list model, the degree of every vertex is given. We search an odd number in the degree list. If there is a vertex in G with odd degree, then the graph is not eulerian. This simple quantum search can be done in $O(\sqrt{n})$ quantum queries to the degree list.

In the adjacency matrix model, we search a vertex with odd degree (if there is one). We use Grover search in combination with a classical algorithms for computing the parity. Total the quantum query complexity of the eulerian graph problem is $O(n^{1.5})$ in the adjacency matrix model. ∎

By using Remark 1, we obtain the quantum time complexity of the eulerian graph problem:

Corollary 1. *There is a quantum algorithm for the eulerian graph problem with time complexity of $O(\sqrt{n}\log^2 n)$ in the adjacency list and $O(n^{1.5}\log^2 n)$ in the adjacency matrix model.*

Now we show that our upper bounds are tight in the matrix and list model.

Theorem 6. *The eulerian graph problem requires $\Omega(\sqrt{n})$ quantum queries to the adjacency list and $\Omega(n^{1.5})$ quantum queries to the adjacency matrix.*

Proof. In the adjacency matrix model, we reduce OR of n parities of length n to the eulerian tour problem. We define

$$z := (x_{1,1} \oplus \ldots \oplus x_{1,n}) \vee \ldots \vee (x_{n,1} \oplus \ldots \oplus x_{n,n}).$$

It is a well known fact, that the computation of z requires $\Omega(n^{1.5})$ quantum queries [Amb02]. Then it is $z = 0$ iff the graph G with adjacency matrix $A = (x_{i,j})$ has an eulerian tour. In the adjacency list model, the $\Omega(\sqrt{n})$ lower bound follows by a simple reduction from the Grover search. ∎

Since the upper and the lower bound match, we have determined the precise quantum query complexity of the eulerian tour problem.

4 Hamiltonian Circuit Problem

In the hamiltonian circuit problem we have given a directed graph G, one has to decide if G has a cycle which contains all the vertices of G exactly once. A hamiltonian graph is one containing a hamiltonian cycle. The hamiltonian cycle problem is analogous to the eulerian graph problem, but a simple characterization of a hamiltonian graph does not exist. The hamiltonian circuit problem is a well known **NP**-complete problem.

There is a quantum query lower bound of $\Omega(n^{1.5})$ for the hamiltonian cycle problem in the matrix model, proved by Berzina et al. [BDFLS04]. We show an upper bound for this problem by using the quantum walk search technique.

Theorem 7. *The quantum query complexity of the hamiltonian cycle problem is $O(n^{2n/(n+1)})$ in the adjacency matrix model.*

Proof. We use Theorem 4. To do so, we construct a Markov chain and a database for checking if a vertex of the chain is marked.

Let $G = (V, E)$ be a directed input graph with n vertices. Let A be a subset of $[n] \times [n]$ of size $r > n$. We will determine r later. The database is the edge-induced subgraph $G[A] := (V, E \cap A)$. Our quantum walk take place on the Johnson graph $J(n^2, r)$. The marked vertices M of $J(n^2, r)$ correspond to subsets of $[n] \times [n]$ with size r, where $G[A]$ contains a hamiltonian cycle in G for all $A \in M$. In every step of the walk, we exchange one element of A.

We determine the quantum query costs for setup, update and checking. The setup cost for the database is $O(r)$, the update cost is $O(1)$, and the checking cost is zero. The spectral gap of the walk on $J(n^2, r)$ is $\delta = O(1/r)$ for $1 \leq r \leq \frac{n^2}{2}$, see e.g. [BS06]. If there is a hamiltonian cycle in G, then there are at least $\binom{n^2-n}{r-n}$ marked sets, since a hamilonian cycle contains n edges. Therefore it holds

$$\epsilon \geq \frac{|M|}{|X|} \geq \frac{\binom{n^2-n}{r-n}}{\binom{n^2}{r}} \geq \Omega\left(\left(\frac{r}{n^2}\right)^n\right).$$

Then the quantum query complexity of the hamiltonian cycle problem is

$$O(r + \left(\frac{n^2}{r}\right)^{n/2} \cdot \sqrt{r}) = O(n^{2n/(n+1)})$$

iff $r = n^{2n/(n+1)}$. ■

5 Travelling Salesman Problem

In this section we consider the travelling salesman problem (TSP) in graphs with maximal degree three, four and five. Given a weighted graph G with bounded degree, compute a hamiltonian cycle in G with minimum total edge weight. There is a simple algorithm by Held and Karp [HK62] for computing a travelling salesman tour with running time $O(2^n)$. Today, this is the fastest known algorithm for the TSP.

5.1 Bounded Degree Three Graphs

Eppstein [Epp03] constructed an algorithm for the TSP on graphs with bounded degree three and running time $O(2^{n/3})$. The general idea of this algorithm is the following (see Figure 1): Let G be a directed weighted graph with maximum degree three. Let F be the set of edges that must be used in the travelling salesman tour, denoted as the *forced edge*. In every step of the algorithm, we choose an edge (t, v) or (t, y) which are adjacent to a forced edge (s, t). If we add (t, v) to F, we delete (t, y) from G, and add the two edges (x, y) and (y, z) to F. Therewith the number of forced edges is increased by three. The subproblem in which we add (t, y) to F is symmetric.

This procedure is the main subroutine of the Eppstein algorithm. It is not difficult to see, that we can transform this deterministic algorithm with running time of $O(2^{n/3})$ in a probabilistic polynomial time algorithm with success probability of $1/2^{n/3}$.

From this classical algorithm we obtain a quantum algorithm by the following two modifications: We use Grover search for finding the edges of the graph, and we apply the quantum amplitude amplification [BHMT00] in oder to get an algorithm which computes a travelling salesman tour with constant success probability. Then we obtain the following result:

Theorem 8. *There is a quantum time algorithm for the TSP on graphs with bounded degree three and running time of* $O(2^{n/6})$.

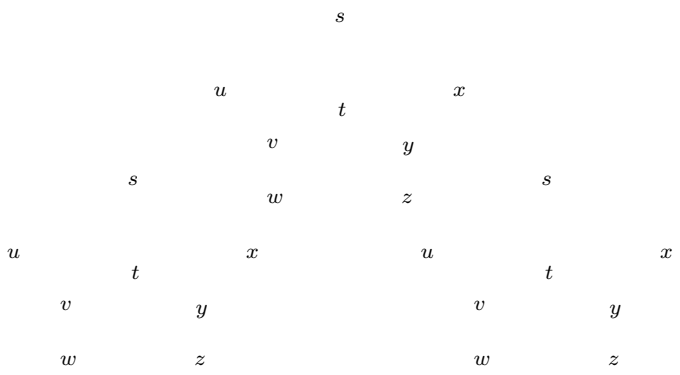

Fig. 1. Travelling salesman tour in graphs with maximal degree three

5.2 Bounded Degree Four Graphs

Now we use an idea of Eppstein [Epp03] to compute the quantum time complexity for finding a travelling salesman tour in graphs with maximal degee four. In classical computation, the fastest algorithm for this problem has running time $O(1.890^n)$, see [Epp03].

Theorem 9. *There is a quantum time algorithm for the TSP on graphs with bounded degree four and running time of* $O((27/4)^{n/6}) = O(1.375^n)$.

Proof. Let k be the number of degree four vertices in the graph G with maximum degree four. The algorithm consists of the following steps: For each degree four vertex v with adjacent edges a, b, c and d, let a be the incoming edge of the tour. We choose randomly among the three possible partitions $\{a, b\}$, $\{a, c\}$ and $\{a, d\}$. We divide the vertex v into two vertices, and connect the two vertices by a forced edge. The new graph has maximum degree three, and therefore we can apply the quantum algorithm of Theorem 8.

Each such divide preserves the travelling salesman tour, if the two edges of the tour do not belong to the same set of the partition. This happens with probability 2/3. We apply the quantum amplitude amplification, and after $\sqrt{(3/2)^k}$ invocation the algorithm finds the correct solution. In total, the quantum time complexity of the algorithm is

$$O\left(1.5^{k/2} \cdot 2^{n/6}\right) = O((27/4)^{n/6}) = O(1.375)^n. \quad \blacksquare$$

5.3 Bounded Degree Five Graphs

In this subsection we consider the travelling salesman problem in graphs with maximal degee five. There is no classical algorithms with running time faster than $O(2^n)$ for this problem. We present a quantum algorithm for the TSP on graphs with maximal degree five and running time $O(1.5874^n)$. We use the same strategy as for bounded degree four graphs.

Theorem 10. *There is a quantum time algorithm for the TSP on graphs with bounded degree five and running time of* $O(1.5874^n)$.

Proof. We use the proof of Theorem 9. Here we choose randomly among four possible partitions, and divide a vertex with degree five into two vertices, which we connect by a forced edge. Then the new graph has maximum degree four, and we can apply Theorem 9. In total, the quantum time complexity of the TSP algorithm for bounded degree five is

$$O\left((4/3)^{k/2} \cdot (27/4)^{n/6}\right) = O(1.5874^n). \quad \blacksquare$$

6 Project Scheduling

A digraph model can be used to schedule projects consisting of several interrelated tasks. Some of these tasks can be executed simultaneously, but some tasks cannot begin until certain others are completed. The goal is to compute the minimal project completion time. One way to represent scheduling projects is to use a digraph model, which is called AOA network.

Definition 1. *An AOA network* $N = (G, c)$ *is a digraph* $G = (V, E)$ *with an edge weight* $c : V \times V \to \mathbb{R}^+$. *Each edge in the digraph represents a task of the project, the direction of the edge is the direction of progress in the project. Each vertex in the AOA network represents an event that signifies the completion of one or more activities and the beginning of a new one. An activity A called predecessor of activity B, if B cannot begin until A is completed.*

We compute the quantum query complexity of the project scheduling problem: Given an AOA network $N = (G, c)$, compute the earliest completion time for every vertex in G. The AOA network must be an acyclic digraph, otherwise none of the tasks corresponding to the edges on the cycle could ever begin.

We are interested on the earliest time point at which each event can occur. Let $ET(i)$ denote the earliest time point in which the event corresponding to vertex i can occur. A vertex j is called *immediate predecessor* of a vertex i if there is an edge from j to i. Let $P(i)$ be the set of all immediate predecessors of vertex i.

Lemma 1. *It holds* $E(1) = 0$ *and* $E(i) = \max_{j \in P(i)}\{ET(j) + c(j, i)\}$.

The earliest time $ET(i)$ for every event i to occur is the length of the longest directed path in the network from vertex 1 to vertex i.

Theorem 11. *The quantum query complexity of the project scheduling problem is* $O(n^{1.5})$ *in the adjacency matrix model and* $O(\sqrt{nm})$ *in the adjacency list model.*

Proof. We use Lemma 1 to compute the earliest time $ET(i)$ for every vertex $i \in \{2, 3, \ldots, n\}$ in order, since the vertices are numbered in a topological way. We use the quantum algorithm by Dürr and Hoyer [DH96] (see Theorem 2) to compute the maximum of $ET(j)+c(j,i)$ for all immediate predecessors of vertex i. The quantum query complexity for this step is $O(\sqrt{n})$ in the adjacency matrix model and $O(\sqrt{d_G^-(i)})$ in the adjacency list model. The total number of quantum queries to the adjacency matrix is $O(n^{1.5})$ and

$$\sum_{i=1}^{n} \sqrt{d_G^-(i)} \leq \sqrt{n} \sqrt{\sum_{i=1}^{n} d_G^-(i)} = O(\sqrt{nm})$$

in the adjacency list model. ∎

Theorem 12. *The quantum time complexity of the* Project Scheduling *algorithm is $O(n^{1.5} \log^2 n)$ in the adjacency matrix model and $O(\sqrt{nm} \log^2 n)$ in the adjacency list model.*

Theorem 13. *The project scheduling problem requires $\Omega(n^{1.5})$ quantum queries to the adjacency matrix and $\Omega(\sqrt{nm})$ quantum queries to the adjacency list.*

Proof. The proof is a reduction from maximum finding. Let k be an integer and M be a matrix with n rows, k columns and with $N = kn$ positive entries. The quantum query lower bound for finding the maximum value in every row is $\Omega(\sqrt{nN})$, see [DHHM04].

We construct a weighted graph $G = (V, E)$, where the set of vertices is $V = \{s, v_1, \ldots, v_k, u_1, \ldots, u_n, t\}$. The edges (s, v_i) and (u_j, t) have the weight 0 for all $i \in [k]$ and $j \in [n]$. The edges (v_i, u_j) get the weight M_{ji}. The graph G has $n + k + 2$ vertices and $m = kn + k + n$ edges.

The earliest time $ET(v_i)$ is zero for all vertices v_i, and the earliest time $ET(u_j)$ is the maximal weighted edge of (v, u_i) with $v \in \{v_1, \ldots, v_k\}$. Then the project scheduling problem requires $\Omega(\sqrt{nm})$ quantum queries to the adjacency list. Setting $m = n^2$, the quantum query lower bound for the adjacency matrix follows. ∎

Conclusion and Open Problems

In this paper we presented quantum algorithms and lower bounds for graph traversal problems. We constructed optimal quantum query algorithms for the eulerian graph and the project scheduling problem. We showed, that the travelling salesman problem in graphs with maximal degree three, four and five can be solved quadratic faster with quantum computing.

Some questions remain open: Is there are a quantum time algorithm for TSP with running time $O(c^n)$ for some $c < 2$? There is a simple algorithm by Held and Karp [HK62] for computing a travelling salesman tour in a graph with running time $O(2^n)$. This algorithm was published in 1962, and it yields

the best complexity that is known today. An other interesting problem is the improvement of the lower or upper bound for the quantum query complexity of the hamiltonian cycle problem.

References

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

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

[AS06] A. Ambainis, R. Špalek, *Quantum Algorithms for Matching and Network Flows*, Proceedings of STACS'06, 2006.

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

[BDFLS04] A. Berzina, A. Dubrovsky, R. Freivalds, L. Lace, O. Scegulnaja, *Quantum Query Complexity for Some Graph Problems*, Proceedings of SOFSEM'04: pages 140-150, 2004.

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

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

[DHHM04] C. Dürr, M. Heiligman, P. Høyer, M. Mhalla, *Quantum query complexity of some graph problems*, Proceedings of ICALP'04: pages 481-493, 2004.

[DH96] C. Dürr, P. Høyer, *A quantum algorithm for finding the minimum*, Technical Report arXiv:quant-ph/9607014, 1996.

[Doe07] S. Dörn, *Quantum Complexity Bounds of Independent Set Problems*, Proceedings of SOFSEM'07 SRF, 2007.

[Epp03] D. Eppstein, *The traveling salesman problem for cubic graphs*, Lecture Notes in Computer Science 2748: pages 307-318, Springer, 2003.

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

[GY99] J. Gross, J. Yellen, *Graph Theory and its Applications*, CRC Press, London 1999.

[HK62] M. Held, R.M. Karp, *A dynamic programming approach to sequencing problems* Journal of SIAM 10: pages 196-210, 1962.

[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, and 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.