The Sturm-Liouville eigenvalue problem and NP-complete problems in the quantum setting with queries

A. Papageorgiou[1] and H. Woźniakowski[2]

[1,2]Department of Computer Science, Columbia University, New York, USA

[2]Institute of Applied Mathematics and Mechanics, University of Warsaw, Poland

May 16, 2006

Abstract

We show how a number of NP-complete as well as NP-hard problems can be reduced to the Sturm-Liouville eigenvalue problem in the quantum setting with queries. We consider *power* queries which are derived from the propagator of a system evolving with a Hamiltonian obtained from the discretization of the Sturm-Liouville operator. We use results of our earlier paper concering the complexity of the Sturm-Liouville eigenvalue problem. We show that the number of power queries as well the number of qubits needed to solve the problems studied in this paper is a low degree polynomial. The implementation of power queries by a polynomial number of elementary quantum gates is an open issue. If this problem is solved positively for the power queries used for the Sturm-Liouville eigenvalue problem then a quantum computer would be a very powerful computation device allowing us to solve NP-complete problems in polynomial time.

PACS numbers: 03.67.Lx, 02.60.-x
Keywords: Complexity, quantum algorithms, appoximation, NP complete problems

1 Introduction

An important question in quantum computing is whether NP-complete problems can be solved in polynomial time, see [3, 4] and papers cited there. We address this question by studying the quantum setting with queries. This paper is based on results in our previous paper [20], where we studied the classical and quantum complexity of the Sturm-Liouville eigenvalue problem. We consider two types of queries: *bit* and *power* queries, see [17] for general information about queries and quantum computation. Here we only mention that bit queries are used in Grover's search algorithm [9]. They allow us to obtain the values of Boolean functions [3], and the approximate values of real functions [10]. Moreover, we know that bit queries cannot be used to solve NP-complete problems in polynomial time [3].

Power queries are used in the well-known phase estimation algorithm, see [17], which plays a central role in Shor's factorization algorithm [21]. In a recent paper [20] we dealt with power queries in the study of the quantum complexity of the Sturm-Liouville eigenvalue problem.

In this paper, we show how to reduce NP-complete problems to the Sturm-Liouville eigenvalue problem whose complexity in the classical and quantum settings has been studied in [20]. Obviously, it would be enough to show this reduction for one NP-complete problem. We choose to present this reduction for several problems to show how the number of power queries and qubits depends on the particular NP-complete problem. In particular, that is why we consider satisfiability and the traveling salesman problem, as well as their NP-hard versions. The reductions presented in this paper can be summarized in the following diagram.

$$\begin{array}{ccccccccc} & & \mathbf{SAT} & \Longrightarrow & \mathbf{BOOL} & \Longrightarrow & \mathbf{INT} & \Longrightarrow & \mathbf{SLE} \\ \mathbf{TSP} & \Longrightarrow & \mathbf{MIN} & \Longrightarrow & \mathbf{BOOL} & \Longrightarrow & \mathbf{INT} & \Longrightarrow & \mathbf{SLE} \\ & & \mathbf{GRO} & \Longrightarrow & \mathbf{BOOL} & \Longrightarrow & \mathbf{INT} & \Longrightarrow & \mathbf{SLE} \end{array}$$

Here, **SAT** stands for the satisfiability problem, **TSP** for the traveling salesman problem, **MIN** for the minimization problem of choosing the smallest number out of N real numbers, **GRO** for Grover's problem, **BOOL** for the Boolean mean problem, **INT** for the integration problem, and finally **SLE** for the Sturm-Liouville eigenvalue problem.

These reductions mean, in particular, that the satisfiability problem is reduced to the Boolean mean problem for a specific Boolean function which is reduced to the integration problem for a specific integrand, which in turn is reduced to the Sturm-Liouville eigenvalue problem for a specific function, and finally the last problem is solved by the quantum algorithm using power queries.

The Sturm-Liouville problem is defined in the next section. For the moment we mention that we want to approximate the smallest eigenvalue of a specific differential operator, and this smallest eigenvalue is given in a variational form as the minimum of specific integrals. We use a formula relating the Sturm-Liouville eigenvalue problem to a weighted integration problem, see [20]. Many computational problems including the discrete problems mentioned above can be recasted as this weighted integration problem. Thus, we can solve them using the algorithms of [20] for solving the Sturm-Liouville eigenvalue problem. These algorithms use of order $\varepsilon^{-1/3}$ bit queries or $\log \varepsilon^{-1}$ power queries and compute an ε-approximation of the smallest eigenvalue with probability $\frac{3}{4}$. The bounds on bit and power queries are sharp up to multiplicative constants, see [5, 20]. Hence, exponentially fewer power queries than bit queries are needed to solve the Sturm-Liouville eigenvalue problem. As we shall see, the same is true for the problems studied in this paper.

In the quantum setting with bit queries, we do not obtain surprising results. The polynomial number of bit queries, $\varepsilon^{-1/3}$, implies that the solution of NP-complete problems by modifications of the algorithm for the Sturm-Liouville eigenvalue problem will require exponentially many queries in terms of the NP problem size.

The situation is quite different if we consider power queries. The logarithmic number of power queries, $\log \varepsilon^{-1}$, implies that NP-complete problems can be solved by modifications of the algorithm for the Sturm-Liouville eigenvalue problem and the number of power queries is polynomial in the problem size.

More specifically, the satisfiability problem for Boolean functions with n variables can be solved with probability $1-\delta$ using of order n log δ^{-1} power queries and n qubits. Furthermore, a truth assignment to a non-zero Boolean function with n variables can be computed with probability $1-\delta$ using of order $n^2(\log\,\delta^{-1}+\log\,n)$ power queries and n qubits.

The traveling salesman problem with m cities can be solved with probability $1-\delta$ using of order

$$m\log\,m(\log\,\delta^{-1}+\log\,m+\log\,d_{\max})(\log\,m+\log\,d_{\max})$$

power queries and $m\log\,m$ qubits, where $d_{\max}$ denotes the maximal distance between cities. Furthermore, an optimal route for the traveling salesman problem can be computed with probability $1-\delta$ using of order

$$m^2\log^2 m(\log\,\delta^{-1}+\log\,m)+m\log\,m(\log\,\delta^{-1}+\log\,m+\log\,d_{\max})(\log\,m+\log\,d_{\max})$$

power queries and $m\log\,m$ qubits.

Finally, Grover's problem for Boolean functions with n variables can be solved with probability $1-\delta$ using of order $n\log\,\delta^{-1}$ power queries and n qubits. It is well known that Grover's problem requires of order $2^{n/2}$ bit queries. Therefore, it is evident that exponentially fewer power than bit queries are required for this problem.

We stress that we only show how many power queries are needed to solve a particular problem. We use power queries which are of the form controlled- W^{p_j} for a $k\times k$ unitary matrix W and some exponents p_j. In our case, the matrix W is given by

$$W\;=\;\exp\left(\tfrac{1}{2}\,\mathrm{i}\,M_q\right)\qquad\text{with}\;\;\mathrm{i}=\sqrt{-1}. \tag{1}$$

Here, the matrix M_q has a particularly simple form since it is a $k\times k$ real symmetric tridiagonal matrix,

$$M_q\;=\;(k+1)^2\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}+\begin{bmatrix} q(\frac{1}{k+1}) & & & & \\ & q(\frac{2}{k+1}) & & & \\ & & \ddots & & \\ & & & q(\frac{k-1}{k+1}) & \\ & & & & q(\frac{k}{k+1}) \end{bmatrix},$$

with a function $q:[0,1]\to[0,1]$ that is two times continuously differentiable and bounded by one up to the second derivative. This matrix corresponds to a classical approximation of the Sturm-Liouville operator.

Contrary to the situation in Shor's algorithm where powers of a unitary operator can be implemented efficiently, the quantum implementation of the power queries for the $k\times k$ matrix W of the form (1) by a number of known elementary quantum gates which is polylog in k is an open issue. If it turns out that the implementation cost of such power queries is disproportionally large compared to that of bit queries, then the positive results on the number of power queries are of only theoretical interest. If, on the other hand, power queries for W of the form (1) and a function q satisfying conditions that we discuss later in this paper, can be implemented efficiently, i.e., at cost which is polylog in k, then we will have a very powerful computational device allowing us to solve NP-complete problems in polynomial time.

2 Sturm-Liouville and Integration

We briefly recall the problem and some of the results from [20]. We consider the following class of functions

$$\mathbf{Q} \;=\; \left\{ q : [0,1] \to [0,1] \;\middle|\; q \in C^2([0,1]) \;\text{ and }\; \max_{i=0,1,2} \|q^{(i)}\|_\infty \le 1 \right\},$$

where $C^2([0,1])$ stands for the class of twice continuously differentiable functions, and $\|q\|_\infty = \max_{x\in[0,1]} |q(x)|$.

We studied the approximate computation of the Sturm-Liouville smallest eigenvalue $\lambda(q)$ which is defined in the variational form by

$$\lambda(q) \;=\; \min_{0\neq u \in H_0^1} \frac{\int_0^1 \left[(u'(x))^2 + q(x)u^2(x)\right]\, dx}{\int_0^1 u^2(x)\, dx}\,, \tag{2}$$

where H_0^1 is the Sobolev space of absolutely continuous functions for which $u' \in L_2([0,1])$ and $u(0) = u(1) = 0$, see [2, 7, 22]. Combining results from from [7, 15, 23] we have the formula that relates the Sturm-Liouville smallest eigenvalue problem to integration,

$$\int_0^1 \left(q(x) - \tfrac{1}{2}\right) \sin^2(\pi x)\, dx \;=\; \tfrac{1}{2}\left(\lambda(q) - \pi^2 - \tfrac{1}{2}\right) \;+\; O\left(\|q - \tfrac{1}{2}\|_\infty^2\right). \tag{3}$$

We analyzed the quantum setting with bit and power queries in [20]. For bit queries, we showed that $\lambda(q)$ can be computed with error η and probability $\frac{3}{4}$ using $\Theta(\eta^{-1/3})$ bit queries, and this bound is sharp modulo a multiplicative constant. It is easy to check that from this result follows that NP-complete problems of size n can be solved with an exponential in n number of bit queries. Therefore, from now on, we restrict ourselves to the quantum setting with power queries for matrices W of the form (1).

In [20] we presented a quantum algorithm ϕ based on phase estimation applied to the discretized matrix of the Sturm-Liouville problem. The initial state was an approximate eigenvector, as proposed by Abrams and Lloyd [1], and computed by the algorithm of the Jaksch and Papageorgiou [14]. The algorithm ϕ computes $\lambda(q,\eta)$ such that

$$|\lambda(q) - \lambda(q,\eta)| \;\le\; \eta \qquad \text{with probability } \tfrac{3}{4} \quad \forall\, q \in \mathbf{Q}$$

using of order $\log \eta^{-1}$ power queries, $\log^2 \eta^{-1}$ additional quantum operations, $O(1)$ function values of q and classical operations, as well as $\log \eta^{-1}$ qubits.

It is well known that we can increase the probability of success to, say, $1-\delta$ by repeating the algorithm ϕ of order $\log \delta^{-1}$ times and then taking the median as the final approximation. The algorithm ϕ with repetitions computes $\lambda(q,\eta,\delta)$ which is an η-approximation of $\lambda(q)$ with probability $1-\delta$, i.e.,

$$|\lambda(q) - \lambda(q,\eta,\delta)| \;\le\; \eta \qquad \text{with probability } \; 1-\delta \quad \forall\, q \in \mathbf{Q}. \tag{4}$$

The resulting algorithm with repetitions ϕ uses of order

- $\log \delta^{-1} \log \eta^{-1}$ power queries,
- $\log \delta^{-1} \log^2 \eta^{-1}$ additional quantum operations,
- $O(1)$ function values of q and classical operations, and
- $\log \eta^{-1}$ qubits.

In the next sections we show how to modify the algorithm ϕ to solve a number of continuous and discrete problems. In what follows, we restrict ourselves and mention only the number of power queries and the number of qubits of these modifications because they are the most important characteristic of the cost of a quantum algorithm. The rest of the cost characteristics can be easily derived from the corresponding components of the cost of the algorithm ϕ with repetitions.

We start with integration. Knowing how to approximate $\lambda(q)$ we can approximate the integral in (3) modulo the second term which is of order $\|q - \frac{1}{2}\|_\infty^2$. We provide the details in the next section.

3 Integration

Consider the (weighted) integration problem

$$I(f) := \int_0^1 f(x) \sin^2(\pi x)\, dx$$

for functions f from the class

$$\mathbb{F}_M = \left\{ f \in C^2([0,1]) : \max_{i=0,1,2} \|f^{(i)}\|_\infty \le M \right\}.$$

Here M is a positive number. We want to compute an ε-approximation of $I(f)$ with probability $1-\delta$ on a quantum computer with power queries. Since $|I(f)| \le M$ we assume that $\varepsilon < M$ since otherwise 0 is an ε-approximation, and the problem is trivial. Without loss of generality, we also assume that $\varepsilon < 1$.

Observe that $f \in \mathbb{F}_M$ implies that the function

$$q_{f,c}(x) = \tfrac{1}{2} + cf(x) \qquad \forall\, x \in [0,1],$$

belongs to $\mathbf{Q}$, defined in the previous section, for $c \in (0, (2M)^{-1}]$. In this case, the formula (3) states

$$I(f) = \frac{1}{2c}\left(\lambda(q_{f,c}) - \pi^2 - \tfrac{1}{2}\right) + O(cM^2) \qquad \forall\, f \in \mathbb{F}_M.$$

Define

$$c = \frac{\varepsilon}{M^2 \log \varepsilon^{-1}} \quad \text{and} \quad \eta = c\varepsilon = \frac{\varepsilon^2}{M^2 \log \varepsilon^{-1}}.$$

Let $\lambda(q_{f,c},\eta,\delta)$ be an η-approximation of $\lambda(q_{f,c})$ with probability $1-\delta$ computed by the algorithm ϕ with repetitions of the previous section. Knowing $\lambda(q_{f,c},\eta,\delta)$ we compute on a classical computer

$$A^{\mathrm{Int}}(f,\varepsilon,\delta) \,=\, \frac{1}{2c}\left(\lambda(q_{f,c},\eta,\delta)-\pi^2-\tfrac{1}{2}\right).$$

Then

$$|I(f)-A^{\mathrm{Int}}(f,\varepsilon,\delta)| \,\le\, \frac{1}{2c}\,\eta \,+\, O(cM^2) \,=\, \tfrac{1}{2}\,\varepsilon(1+o(1)) \quad \text{with probability} \;\; 1-\delta.$$

Hence, for small ε, $A^{\mathrm{Int}}(f,\varepsilon,\delta)$ is an ε-approximation of $I(f)$ with probability $1-\delta$. In this way we can solve the integration problem and we summarize this result in the following theorem.

Theorem 3.1. *We compute an ε-approximation with probability $1-\delta$ for the integration problem for the class $\mathbb{F}_M$ by the quantum algorithm A^{Int} using of order*

- $\log\delta^{-1}\,(\log\,M\,+\,\log\,\varepsilon^{-1})$ *power queries and*
- $\log\,M\,+\,\log\,\varepsilon^{-1}$ *qubits.*

4 Preliminaries

We now present some preliminaries that will be used as technical tools to translate the integration problem of the previous section to the NP-complete and NP-hard problems discussed in this paper.

Take a function $h \in C^2([0,1])$ with $h^{(i)}(0)=h^{(i)}(1)=0$ for $i=0,1,2$, and for which the integral $\int_0^1 h(x)\,dx$ is positive. Examples of such functions include $h(x)=(x(1-x))^\alpha$ with $\alpha>2$ or $h(x)=x^3(1-x)^3g(x)$ for a positive $g\in C^2([0,1])$. We extend the domain of h by defining $H(x)=h(x)$ for $x\in[0,1]$ and $H(x)=0$ otherwise. Due to the boundary conditions imposed on h, we have $H\in C^2(\mathbb{R})$.

For a positive (large) integer N, we subdivide the interval $[\frac{1}{4},\frac{3}{4}]$ by introducing the points

$$x_j \,=\, \frac{1}{4}+\frac{1}{2}\,\frac{j}{N} \qquad \text{for} \;\; j=0,1,\dots,N.$$

For $j=0,1,\dots,N-1$, we define the functions

$$h_j(x) \,=\, \frac{1}{4N^2}\,H\left(2N(x-x_j)\right) \qquad \text{for} \;\; x\in[0,1].$$

Observe that h_j vanishes outside the open interval (x_j,x_{j+1}), and $\|h_j\|_\infty=\|h\|_\infty/(4N^2)$, $\|h_j'\|_\infty=\|h'\|_\infty/(2N)$, and $\|h_j''\|_\infty=\|h''\|_\infty$. Hence, if we set

$$M \,:=\, \max\left(\frac{\|h\|_\infty}{4N^2},\frac{\|h'\|_\infty}{2N},\|h''\|_\infty\right),$$

which is equal to $\|h''\|_\infty$ for large N, then

$$h_j \in \mathbb{F}_M \qquad \text{for } \ j=0,1,\dots,N-1.$$

Observe finally that

$$\int_0^1 h_j(x)\,dx \;=\; \int_{x_j}^{x_{j+1}} h_j(x)\,dx \;=\; \frac{1}{8N^3}\,\mathrm{Int}(h) \qquad \text{for } \ j=0,1,\dots,N-1, \tag{5}$$

where

$$\mathrm{Int}(h) \;:=\; \int_0^1 h(x)\,dx.$$

5 Boolean mean

Consider the class $\mathbb{B}_n$ of all Boolean functions of n variables mapping $\{0,1\}^n$ into $\{0,1\}$. We can equivalently assume that the domain of such Boolean functions is $\{0,1,\dots,N-1\}$ with $N=2^n$. Hence, $\mathbb{B}\in\mathbb{B}_n$ means that

$$\mathbb{B}:\{0,1,\dots,N-1\} \;\to\; \{0,1\}.$$

We want to approximate the mean

$$\mathrm{S}_N(\mathbb{B}) \;=\; \frac{1}{N}\sum_{j=0}^{N-1}\mathbb{B}(j)$$

by a quantum algorithm with power queries.

We now show how this problem can be reduced to the integration problem of Section 3. Using the notation of Section 4, we define the function $f_{\mathbb{B}}:[0,1]\to\mathbb{R}$ by

$$f_{\mathbb{B}}(x) \;=\; \begin{cases} h_j(x)\,\mathbb{B}(j)\,\left(2\sin^2(\pi x)\right)^{-1} & \text{if } x\in[x_j,x_{j+1}] \text{ with } j=0,1,\dots,N-1,\\ 0 & \text{if } x\in[0,\frac14]\cup[\frac34,1]. \end{cases}$$

Observe that due to the fact that h_j vanishes up to the second derivatives at x_j and x_{j+1}, and the fact that $2\sin^2(\pi x)\ge 1$ for $x\in[\frac14,\frac34]$, we conclude that $f_{\mathbb{B}}\in C^2([0,1])$. Furthermore

$$\begin{aligned} \|f_{\mathbb{B}}\|_\infty &\le \|h\|_\infty/(4N^2),\\ \|f'_{\mathbb{B}}\|_\infty &\le \|h'\|_\infty/(2N)(1+O(N^{-1})), and\\ \|f''_{\mathbb{B}}\|_\infty &\le \|h''\|_\infty(1+O(N^{-1})). \end{aligned}$$

Hence, if we set $M=\|h''\|_\infty(1+O(N^{-1}))$ then $f_{\mathbb{B}}\in\mathbb{F}_M$. Observe that, due to (5), the integration problem for $f_{\mathbb{B}}$ takes now the form

$$I(f_{\mathbb{B}}) \;=\; \sum_{j=0}^{N-1}\tfrac12\mathbb{B}(j)\int_{x_j}^{x_{j+1}} h_j(x)\,dx \;=\; \frac{\mathrm{Int}(h)}{16N^2}\,\frac{1}{N}\sum_{j=0}^{N-1}\mathbb{B}(j) \;=\; \frac{\mathrm{Int}(h)}{16N^2}\,\mathrm{S}_N(\mathbb{B}).$$

Let $\eta = \text{Int}(h)\varepsilon/(16N^2)$. We now use the quantum algorithm $A^{\text{Int}}(f_{\mathbb{B}}, \eta, \delta)$ from Section 3 which, for small ε or large N, computes an η-approximation of $I(f_{\mathbb{B}})$ with probability $1 - \delta$. Knowing $A(f_{\mathbb{B}}, \eta, \delta)$ we compute on a classical computer

$$A_n^{\text{Bool}}(\mathbb{B}, \varepsilon, \delta) \,=\, \frac{16N^2}{\text{Int}(h)}\, A^{\text{Int}}(f_{\mathbb{B}}, \eta, \delta).$$

Then

$$|\text{S}_N(\mathbb{B}) - A_n^{\text{Bool}}(\mathbb{B}, \varepsilon, \delta)| \,=\, \frac{16N^2}{\text{Int}(h)}\, |I(f_{\mathbb{B}}) - A^{\text{Int}}(f_{\mathbb{B}}, \eta, \delta)| \,\leq\, \frac{16N^2}{\text{Int}(h)}\eta \,=\, \varepsilon,$$

and this holds with probability $1 - \delta$. We summarize this result in the following theorem.

Theorem 5.1. *We compute an ε-approximation with probability $1 - \delta$ for the Boolean mean problem for the class $\mathbb{B}_n$ by the quantum algorithm A_n^{Bool} using of order*

- $\log \delta^{-1}\ (\log \varepsilon^{-1} + n)$ *power queries and*
- $\log \varepsilon^{-1} + n$ *qubits.*

It is known that the amplitude amplification algorithm of Brassard, Høyer, Mosca and Tapp [6] computes an ε-approximation of $\text{S}_N(\mathbb{B})$ with probability $8/\pi^2 = 0.81\dots$ using of order $\min(N, \varepsilon^{-1})$ bit queries. Furthermore, this number of queries is order-minimal as proven by Nayak and Wu [16]. We stress that the basic part of the quantum algorithm A_n^{Bool} is the phase estimation algorithm which uses power queries. For $\varepsilon \geq N^{-1}$ this algorithm uses of order $\log \varepsilon^{-1}$ power queries to compute an ε-approximation with probability $8/\pi^2$. Hence, the nunber of queries has an exponential improvement in its the dependence on ε^{-1}.

The amplitude amplification algorithm of [6] has been used as a basic tool for solving many continuous problems such as real mean, multivariate integration, path integration and multivariate approximation in the quantum setting. These problems have been defined over many classical spaces such as L_p, Sobolev and Korobov spaces. For a number of these continuous problems, the bit query complexity and the quantum speedups over the worst case and randomized settings have been established based on the optimality of the quantum summation algorithm, see [10, 11, 12, 13, 18, 19, 24]. The use of power queries yields an exponential improvement in the number of queries. This can be achieved simply by using the exponentially better power query bound for the Boolean mean.

6 Satisfiability

The satisfiability problem, SAT for short, is a well known NP-complete problem in the Turing machine model of computation [8]. This means that all NP-complete problems can be reduced to SAT in polynomial time, and if the conjecture P≠NP is true then there are no algorithms solving SAT in polynomial time with respect to the length of the Boolean function expressed in conjunctive normal form. SAT can be stated as a decision or as a computational problem and we deal with both of them in this section.

As in the previous section, consider the class $\mathbb{B}_n$ of all Boolean functions of n variables with the domain $\{0,1,\dots,N-1\}$, where $N=2^n$. The two variants of SAT problem are defined as

- $\mathbf{SAT}_1$: for $\mathbb{B} \in \mathbb{B}_n$ given in the conjunctive normal form, verify if there exists an index j such that $\mathbb{B}(j)=1$.
- $\mathbf{SAT}_2$: for a non-zero $\mathbb{B} \in \mathbb{B}_n$ given in the conjunctive normal form, compute an index j such that $\mathbb{B}(j)=1$.

We now show that either problem can be solved with probability $1-\delta$ by using a number of power queries which is polynomial in n and $\log\,\delta^{-1}$.

We begin with SAT_1 and use the notation of Section 5. Observe that for any $\mathbb{B} \in \mathbb{B}_n$, the mean $\mathrm{S}_N(\mathbb{B})$ is a multiple of N^{-1}, i.e., $\mathrm{S}_N(\mathbb{B}) = k/N$ for some $k \in \{0,1,\dots,N\}$. If we have a real number x such that $|\mathrm{S}_N(\mathbb{B}) - x| < \frac{1}{2N}$ then $|k - N\,x| < \frac{1}{2}$ which implies that

$$k \;=\; \lfloor N\,x + \tfrac{1}{2} \rfloor \quad \text{and} \quad \mathrm{S}_N(\mathbb{B}) \;=\; \frac{\lfloor N\,x + \frac{1}{2} \rfloor}{N}.$$

Obviously, $k>0$ iff there exists an index j for which $\mathbb{B}(j)=1$.

For $\eta < \frac{1}{2N}$, we conclude that from an η-approximation of $\mathrm{S}_N(\mathbb{B})$, with probability $1-\delta$, we can compute the exact value of $\mathrm{S}(\mathbb{B})$ with probability $1-\delta$. We know that, for small η, $A^{\mathrm{Bool}}(\mathbb{B},\eta,\delta)$ computes an η-approximation of $\mathrm{S}_N(\mathbb{B})$ with probability $1-\delta$. Letting $\eta = 1/(3N)$ we compute $A_n^{\mathrm{Bool}}(\mathbb{B},1/(3N),\delta)$, which is an $1/(3N)$-approximation of $\mathrm{S}_N(\mathbb{B})$. From this we can compute the exact value of $\mathrm{S}_N(\mathbb{B})$ with probability $1-\delta$. This means that

$$A_n^{\mathrm{SAT}_1}(\mathbb{B},\delta) \;=\; \begin{cases} \text{YES} & \text{if } \lfloor N\,A^{\mathrm{Bool}}(\mathbb{B},1/(3N),\delta) + \frac{1}{2} \rfloor \;>\; 0, \\ \text{NO} & \text{if } \lfloor N\,A^{\mathrm{Bool}}(\mathbb{B},1/(3N),\delta) + \frac{1}{2} \rfloor \;=\; 0 \end{cases}$$

solves the satisfiability problem. We stress that to compute $A_n^{\mathrm{SAT}}(\mathbb{B},\delta)$ we run the quantum algorithm $A_n^{\mathrm{Bool}}(\mathbb{B},1/(3N),\delta)$ and the rest is computed on a classical computer. Since we know how many power queries and qubits are used by A_n^{Bool} we obtain the following theorem.

Theorem 6.1. *The satisfiability problem SAT_1 for the class $\mathbb{B}_n$ is solved with probability $1-\delta$ by the quantum algorithm $A_n^{\mathrm{SAT}_1}$ which uses of order*

- *$n\,\log\delta^{-1}$ power queries and*
- *n qubits.*

We turn to the SAT_2 problem. That is, for a non-zero $\mathbb{B}$ from $\mathbb{B}_n$ we want to compute an index $j \in \{0,1,\dots,N-1\}$ for which $\mathbb{B}(j)=1$.

We will use bisection on the domain of $\mathbb{B}$. Every bisection step will shrink the cardinality of the domain by 2. Using the quantum algorithm $A_k^{\mathrm{SAT}_1}$ with $k=n,n-1,\dots,0$, we will know whether an index of the true assignment belongs to the the decreased domain. In this way, after n steps we identify an index j for which $\mathbb{B}(j)=1$. Since $A_k^{\mathrm{SAT}_1}$ is a probabilistic algorithm and we use it n times, we need the success probability of this algorithm to be $1-\delta_1$, where

$$(1-\delta_1)^n \;=\; 1-\delta.$$

For small δ, we obviously have $\delta_1 = \delta/n(1+o(1))$.

More precisely, let

$$D_k \;=\; \{0,1,\dots,2^k-1\} \qquad \text{for } \; k=0,1,\dots,n.$$

We set $j_n = 0$, and perform the following steps for the Boolean function $\mathbb{B}$ from $\mathbb{B}_n$.

For $k = n-1, n-2, \dots, 1, 0$ do:

- define $f_k : D_k \to \{0,1\}$ by $f_k(j) = \mathbb{B}(j + j_{k+1})$ for $j \in D_k$,
- run the quantum algorithm $A_k^{\mathrm{SAT}_1}(f_k, \delta_1)$ and compute on a classical computer

$$j_k \;=\; \begin{cases} j_{k+1} + 2^k & \text{if } A_k^{\mathrm{SAT}_1}(f_k,\delta_1) \;=\; \text{NO},\\ j_{k+1} & \text{if } A_k^{\mathrm{SAT}_1}(f_k,\delta_1) \;=\; \text{YES}. \end{cases}$$

Finally we set

$$A_n^{\mathrm{SAT}_2}(\mathbb{B},\delta) \;=\; j_0.$$

We claim that the algorithm $A_n^{\mathrm{SAT}_2}$ solves the satisfiability problem SAT_2, i.e., for the index j_0 we have $\mathbb{B}(j_0) = 1$.

Indeed, first note that $j_k \;\le\; 2^k + 2^{k+1} + \cdots + 2^{n-1} \;\le\; 2^n - 1 = N-1$ and $j + j_{k+1} \le 2^k - 1 + j_{k+1} \le N-1$ for $j \in D_k$. Therefore, the Boolean functions f_k are well defined.

For the first step, $k = n-1$, we have $f_{n-1} \equiv \mathbb{B}$ on the first half, D_{n-1}, of the domain D_n. We check whether f_{n-1} is zero. This holds with probability $1-\delta_1$ iff $A_{n-1}^{\mathrm{SAT}_1}(f_{n-1},\delta_1) = \text{NO}$. If $f_{n-1} \equiv 0$ then $\mathbb{B}$ is non-zero on the complement of D_{n-1} and an index j for which $\mathbb{B}(j) = 1$ is at least equal to 2^{n-1}. That is why we define $j_{n-1} = 2^{n-1}$ in this case. If, however, f_{n-1} is non-zero over D_{n-1} then we are looking for an index j with $\mathbb{B}(j) = 1$ in the set D_{n-1}, and we set $j_{n-1} = j_n = 0$. In this way, after the first step we can restrict the search of an index to the domain of cardinality 2^{n-1}. For the second step it is enough to work with the domain D_{n-2} and use the proper shift j_{n-1} in the definition of the Boolean function f_{n-2}. After n steps we identify a proper index $j = j_0$ for which $\mathbb{B}(j) = 1$. In fact, it is easy to see that we will find the smallest index j for which $\mathbb{B}(j) = 1$

Since the quantum algorithm $A_k^{\mathrm{SAT}_1}(f_k,\delta_1)$ works with probability $1-\delta_1$ and we repeat n times the algorithm, the probability of success is at least $(1-\delta_1)^n$. By the definition of δ_1, this is equal to $1-\delta$. This proves the following theorem.

Theorem 6.2. *The satisfiability problem SAT_2 for the class $\mathbb{B}_n$ is solved with probability $1-\delta$ by the quantum algorithm $A_n^{\mathrm{SAT}_2}$ which uses of order*

- $n^2\,(\log\delta^{-1} + \log n)$ *power queries and*
- n *qubits.*

7 Grover's problem

Grover's problem can be defined as the satisfiability problem SAT_2 for the class $\bar{\mathbb{B}}_n$ of Boolean functions of n variables for which we know a priori that there exists exactly one index $j = j_{\mathbb{B}}$ for which $\mathbb{B}(j) = 1$. Obviously, the algorithm $A_n^{\mathrm{SAT}_2}$ solves Grover's problem although the a priori knowledge about the uniqueness of the index $j_{\mathbb{B}}$ is not used. We now show that using this a priori knowledge it is possible to find a more efficient quantum algorithm than $A_n^{\mathrm{SAT}_2}$.

Define the weighted Boolean mean

$$W_N(\mathbb{B}) \;=\; \frac{1}{N}\sum_{j=0}^{N-1} j\,\mathbb{B}(j) \qquad \text{for } \; \mathbb{B} \in \bar{\mathbb{B}}_n.$$

Clearly, $W_N(\mathbb{B}) = j_{\mathbb{B}}/N$. Hence, it is enough to compute the exact value of $W_N(\mathbb{B})$ and then $j_{\mathbb{B}} = N\,W_N(\mathbb{B})$. This can be achieved by switching to the integration problem, as we did in Section 5, for the function $g_{\mathbb{B}} : [0,1] \to \mathbb{R}$ defined by

$$g_{\mathbb{B}}(x) \;=\; \begin{cases} j\,h_j(x)\,\mathbb{B}(j)\,\big(2\,N\,\sin^2(\pi x)\big)^{-1} & \text{if } x \in [x_j, x_{j+1}] \text{ with } j = 0,1,\dots,N-1, \\ 0 & \text{if } x \in [0,\frac{1}{4}] \cup [\frac{3}{4},1]. \end{cases}$$

As in Section 5, we conclude that $g_{\mathbb{B}} \in \mathbb{F}_M$ for $M = \|h''\|_\infty(1+O(N^{-1}))$, and

$$I(g_{\mathbb{B}}) \;=\; \frac{j_{\mathbb{B}}}{2N}\int_{x_{j_{\mathbb{B}}}}^{x_{j_{\mathbb{B}}+1}} h_j(x)\,dx \;=\; \frac{\mathrm{Int}(h)}{16N^3}\,W_N(\mathbb{B}).$$

Hence,

$$j_{\mathbb{B}} \;=\; \frac{16N^4}{\mathrm{Int}(h)}\,I(g_{\mathbb{B}}).$$

In Section 3, we defined the algorithm A^{Int} such that, for small ε, $A^{\mathrm{Int}}(f,\varepsilon,\delta)$ is an ε-approximation of $I(f)$ with probability $1-\delta$.

Let $\varepsilon = \mathrm{Int}(h)/(48N^4)$. Define the quantum algorithm

$$A_n^{\mathrm{Grover}}(\mathbb{B},\delta) \;=\; \left\lfloor \frac{16N^4}{\mathrm{Int}(h)}\,A^{\mathrm{Int}}(g_{\mathbb{B}},\varepsilon,\delta) \,+\, \frac{1}{2}\right\rfloor.$$

Then

$$\left| j_{\mathbb{B}} - \frac{16N^4}{\mathrm{Int}(h)}\,A^{\mathrm{Int}}(g_{\mathbb{B}},\varepsilon,\delta)\right| \;=\; \frac{16N^4}{\mathrm{Int}(h)}\,\big|I(g_{\mathbb{B}}) - A^{\mathrm{Int}}(g_{\mathbb{B}},\varepsilon,\delta)\big| \;\le\; \frac{16N^4}{\mathrm{Int}(h)}\,\varepsilon \;=\; \frac{1}{3},$$

and this holds with probability $1-\delta$. Hence,

$$j_{\mathbb{B}} \;=\; A_n^{\mathrm{Grover}}(\mathbb{B},\delta) \;\text{ with probability } 1-\delta.$$

This and Theorem 3.1 yield the following theorem.

Theorem 7.1. *Grover's problem for the class* $\mathbb{B}_n$ *is solved with probability* $1-\delta$ *by the quantum algorithm* A_n^{Grover} *which uses of order*

- $n\,\log\delta^{-1}$ *power queries and*
- n *qubits.*

8 Minimization

In this section we consider a real number minimization problem which we will use to solve the traveling salesman problem in the quantum setting with power queries.

For positive $N = 2^n$ and M, define the set

$$\mathbb{X}_{n,M} \;=\; \{\, x = [x_0, x_1, \dots, x_{N-1}] \;:\; x_j \in \mathbb{R} \text{ and } |x_j| \;\le\; M \text{ for } \; j = 0, 1, \dots, N-1 \,\}.$$

Let

$$\mathrm{Min}(x) \;=\; \min_{j=0,1,\dots,N-1} x_j.$$

The minimization problem is defined as:

- $\mathbf{MIN}_1$: compute $A(x)$ which is an ε-approximation of $\mathrm{Min}(x)$ with probability $1-\delta$, i.e., $|\mathrm{Min}(x) - A(x)| \;\le\; \varepsilon$ holds with probability $1-\delta$ for all $x \in \mathbb{X}_{n,M}$.
- $\mathbf{MIN}_2$: compute an index $j = j(x)$ for which $|\mathrm{Min}(x) - x_j| \le \varepsilon$ with probability $1-\delta$ for all $x \in \mathbb{X}_{n,M}$.

Clearly, using a classical computer we must use each x_j at least once and that is why the worst case and randomized complexities are proportional to N, i.e., they are exponential in n. We now show how to solve this problem in the quantum setting using a number of power queries which is polynomial in $n, \log M, \log \delta^{-1}$ and $\log \varepsilon^{-1}$.

We begin with the minimization problem MIN_1. For a real number y, define the Boolean function $f_y : \{0, 1, \dots, N-1\} \to \{0, 1\}$ by

$$f_y(j) \;=\; \begin{cases} 1 & \text{if } x_j \;\le\; y, \\ 0 & \text{if } x_j \;>\; y. \end{cases}$$

Then let

$$\mathrm{S}_N(f_y) \;=\; \frac{1}{N} \sum_{j=0}^{N-1} f_y(j).$$

Note that

$$\mathrm{S}_N(f_y) > 0 \quad \text{iff} \quad y \;\ge\; \mathrm{Min}(x).$$

Clearly, the condition $\mathrm{S}_N(f_y) > 0$ is equivalent to the SAT_1 problem for the Boolean function f_y and can be solved by the quantum algorithm $A_n^{\mathrm{SAT}_1}$ of Section 6.

Initially, we know that $\mathrm{Min}(x) \in [-M, M]$. That is why we set $a_0 = -M$, $b_0 = M$ and $y_0 = 0$, and use the bisection algorithm for the interval $[-M, M]$ with k^* steps,

$$k^* \;=\; \left\lceil \log_2 \frac{M}{\varepsilon} \right\rceil.$$

We also choose δ_1 such that $(1-\delta_1)^{k^*} = 1 - \delta$. For small δ, we have $\delta_1 = \delta/k^*(1 + o(1))$. More precisely, we perform the following steps.

For $k = 1, 2, \dots, k^*$ do:

- run the quantum algorithm $A^{\mathrm{SAT}_1}(f_{y_{k-1}}, \delta_1)$,
- compute on a classical computer

$$a_k \;=\; \begin{cases} a_{k-1} & \text{if } A_n^{\mathrm{SAT}_1}(f_{y_{k-1}}, \delta_1) = \mathrm{YES}, \\ y_{k-1} & \text{if } A_n^{\mathrm{SAT}_1}(f_{y_{k-1}}, \delta_1) = \mathrm{NO}, \end{cases}$$

$$b_k \;=\; \begin{cases} y_{k-1} & \text{if } A_n^{\mathrm{SAT}_1}(f_{y_{k-1}}, \delta_1) = \mathrm{YES}, \\ b_{k-1} & \text{if } A_n^{\mathrm{SAT}_1}(f_{y_{k-1}}, \delta_1) = \mathrm{NO}, \end{cases}$$

- and $y_k \;=\; \frac{1}{2}(a_k + b_k)$.

Finally set

$$A_{n,M}^{\mathrm{Min}_1}(x, \varepsilon, \delta) \;=\; y_{k^*}.$$

After k^* steps we have the interval $[a_{k^*}, b_{k^*}]$ of length $2M/2^{k^*} \;\le\; 2\varepsilon$, and $\mathrm{Min}(x) \in [a_{k^*}, b_{k^*}]$. Therefore $|\mathrm{Min}(x) - y_{k^*}| \;\le\; \varepsilon$ is an approximation of $\mathrm{Min}(x)$. Note that this algorithm works with probability $(1-\delta_1)^{k^*} \;=\; 1-\delta$. Knowing the requirements of the algorithm $A_n^{\mathrm{SAT}_1}$ we obtain the following theorem.

Theorem 8.1. *We compute an ε-approximation with probability $1-\delta$ for the minimization problem MIN$_1$ for the class $\mathbb{X}_{n,M}$ by the quantum algorithm $A_{n,M}^{\mathrm{Min}_1}$ using of order*

- $n\;(\log\, M \;+\; \log\, \varepsilon^{-1})\,(\log \delta^{-1} \;+\; \log\,(\log\, M \;+\; \log\, \varepsilon^{-1}))$ *power queries and*
- n *qubits.*

We now turn to the minimization problem MIN$_2$. It is easy to see that this problem can be solved by combining the quantum algorithms developed so far. To explain the main idea of the quantum algorithm for the minimization problem MIN$_2$ we ignore for a moment the fact that all the quantum algorithms of the previous sections work probabilistically. Knowing $y = y_{k^*}$ by the $A_n^{\mathrm{Min}_1}$ algorithm, we apply the A_n^{Bool} algorithm of Section 5 and compute the exact value of $\mathrm{S}_N(f_y)$. If $\mathrm{S}_N(f_y) \neq 0$ then f_y is a non-zero Boolean function. If $S(f_y) = 0$ then $y < \mathrm{Min}(x)$ and since y is an ε-approximation to $\mathrm{Min}(x)$, we have $\mathrm{Min}(x) - \varepsilon \le y < \mathrm{Min}(x)$. In this case we have $\mathrm{Min}(x) \le y + \varepsilon$ and $0 \le y + \varepsilon - \mathrm{Min}(x) \le \varepsilon$. Hence, $y + \varepsilon$ is also an ε-approximation to $\mathrm{Min}(x)$ and $S(f_{y+\varepsilon}) \neq 0$. Thus we modify $y := y + \varepsilon$ if $S(f_y) = 0$. Then $S(f_y) \neq 0$ in either case, and y is still an ε-approximation to $\mathrm{Min}(x)$. Knowing that f_y is a non-zero Boolean function, we now run the $A_n^{\mathrm{SAT}_2}$ algorithm and compute an index j for which $f_y(j) = 1$, or equivalently, $x_j \le y$. Hence, $\mathrm{Min}(x) \le x_j \le y \le \mathrm{Min}(x) + \varepsilon$ and x_j is an ε-approximation that solves the minimization problem MIN$_2$. We now formalize the idea of this algorithm. Let $(1-\delta_1)^3 = 1 - \delta$. Hence, for small δ we have $\delta_1 = 3\delta(1 + o(1))$. We perform the following steps:

- run the quantum algorithm $A_{n,M}^{\mathrm{Min}_1}(x, \varepsilon, \delta_1)$ to obtain y,
- run the quantum algorithm $A_n^{\mathrm{Bool}}(f_y, 1/(3N), \delta_1)$ to obtain z,
- if $z = 0$ then set $y := y + \varepsilon$,

- run the quantum algorithm $A_n^{\mathrm{SAT}_2}(f_y,\delta_1)$ to obtain j.

Finally set,

$$A_{n,M}^{\mathrm{Min}_2}(x,\varepsilon,\delta) \;=\; j.$$

Then the index j solves the minimization problem MIN_2 and this algorithm works with probability $(1-\delta_1)^3 = 1-\delta$. Counting the number of power queries and qubits of all parts of the algorithm we obtain the theorem.

Theorem 8.2. *We compute an ε-approximation with probability $1-\delta$ for the minimization problem MIN_2 for the class $\mathbb{X}_{n,M}$ by the quantum algorithm $A_{n,M}^{\mathrm{Min}_2}$ using of order*

- $n^2\,(\log\,\delta^{-1} + \log\,n) \quad + \quad n\,(\log\,M \,+\, \log\,\varepsilon^{-1})\,(\log\delta^{-1} \,+\, \log\,(\log\,M \,+\, \log\,\varepsilon^{-1}))$ *power queries and*

- *n qubits*

.

9 Traveling salesman

The traveling salesman problem, TSP for short, is a well known NP-complete problem that deals with the shortest tour between m cities, see [8]. Let $D = [d(j,k)]_{j,k=1}^m$ denote the $m\times m$ matrix with $d(j,k)$ being the distance between the city j and the city k. We assume that $d(j,k)$ are positive integers for $j\neq k$ and $d(j,j)=0$. The TSP can be studied as a decision or as a computational problem and we consider the three variants:

- $\mathbf{TSP}_1$: for a given integer B verify if there is a permutation $\pi = [\pi(1),\pi(2),\dots,\pi(m)]$ of indices $\{1,2,\dots,m\}$ for which

  $$d(\pi) \;:=\; \sum_{j=1}^m d(\pi(j),\pi(j+1)) \;\leq\; B$$

  with $\pi(m+1) = \pi(1)$.

- $\mathbf{TSP}_2$: compute

  $$\mathrm{Min}(D) \;=\; \min_\pi d(\pi).$$

- $\mathbf{TSP}_3$: compute a permutation π^* such that

  $$d(\pi^*) \;=\; \mathrm{Min}(D).$$

Observe that if we can solve TSP_2 then it is just enough to check whether $\mathrm{Min}(D)\leq B$. Similarly, if we can solve TSP_3 then it is enough to compute $d(\pi^*)$ to solve TSP_2.

We now show that TSP is a special case of the minimization problem studied in the previous section. Indeed, consider the set P_m of all $m!$ possible permutations, and let $g:\{0,1,\dots,m!-1\}\to P_m$ be an injective mapping. We take

$$n \;:=\; \lceil \log\,m!\rceil \;=\; m\log\,m\,(1+o(1)).$$

Let $N = 2^n$. For $j = m!, m!+1, \ldots, N-1$ we extend the function g by setting $g(j) = g(m!-1)$. Then $g : \{0, 1, \ldots, N-1\} \to P_m$ and

$$g(j) = \pi_j = [\pi_j(1), \pi_j(2), \ldots, \pi_j(m)] \qquad \text{for } j \in \{0, 1, \ldots, N-1\}.$$

Defining

$$x_j = \sum_{k=1}^{m} d(\pi_j(k), \pi_j(k+1)) \qquad \text{with } \pi_j(m+1) = \pi_j(1),$$

we see that

$$\min_{j=0,1,\ldots,N-1} x_j = \mathrm{Min}(D).$$

Note that x_j can be computed using a number of bits which is polynomial in m and the maximal distance between cities,

$$d_{\max} = \max_{i,j=1,2,\ldots,m} d(i,j).$$

Furthermore, $x_j \in [0, M]$ for $j \in \{0, 1, \ldots, N-1\}$ with

$$M \geq m\, d_{\max}.$$

To run the algorithms $A_{n,M}^{\mathrm{Min}_1}$ and $A_{n,M}^{\mathrm{Min}_2}$ for the solutions of the minimization problem we need to know an upper bound on $m\, d_{\max}$. This can be achieved as follows. For $k = 0, 1, \ldots,$ we run the quantum algorithm $A_n^{\mathrm{SAT}_1}(1 - f_{2^k}, \delta_1)$ of Section 6 to check, with probability $1 - \delta_1$, whether the Boolean function $1 - f_{2^k}$ defined in Section 8 is zero. If so, then $f_{2^k} \equiv 1$ meaning that for all $x_j \leq 2^k$, and we can take $M = 2^k$. Hence, we perform as many steps as necessary until the first occurrence of $A_n^{\mathrm{SAT}_1}(1 - f_{2^k}, \delta_1) = \mathrm{NO}$. This will happen after at most $p := \lceil \log_2 m + \log_2 d_{\max} \rceil$ steps. Then we define $M = 2^p$ and $M \in [m\, d_{\max}, 2m\, d_{\max}]$. We choose δ_1 such that $(1-\delta_1)^p = (1-\delta)^{1/2}$ to conclude that we obtain M with probability $(1-\delta)^{1/2}$. For small δ, we have $\delta_1 = \delta/(2p)(1 + o(1))$.

Knowing M, we can solve TSP by solving the minimization problem of Section 8. Since all x_j are integers, it is enough to compute an ε-approximation y of $\mathrm{Min}(x)$ with $\varepsilon < \frac{1}{2}$ to conclude that $\lfloor y + \frac{1}{2} \rfloor$ is equal to $\mathrm{Min}(x)$. The number y can be computed with probability $(1-\delta)^{1/2} = 1 - \delta_2$ by the quantum algorithm $A_{n,M}^{\mathrm{Min}_1}(x, \varepsilon, \delta_2)$ with ε, say, $\frac{1}{3}$. In the same way, we can find a shortest path represented by a permutation π^* by first computing the index j by the algorithm $A_{n,M}^{\mathrm{Min}_2}$ of Section 8 for which $|\mathrm{Min}(x) - x_j| \leq \frac{1}{3}$. Again, since x_j is an integer, $x_j = \mathrm{Min}(x)$ and $\pi^* = g(j)$ is a needed permutation. Hence the following algorithms

$$\begin{aligned}
A_m^{\mathrm{TSP}_1}(D, \delta) &= \begin{cases} \mathrm{YES} & \text{if } \lfloor A_{n,M}^{\mathrm{Min}_1}(x, \frac{1}{3}, \delta_2) + \frac{1}{2} \rfloor \leq B, \\ \mathrm{NO} & \text{if otherwise}, \end{cases} \\
A_m^{\mathrm{TSP}_2}(D, \delta) &= x_{A_{n,M}^{\mathrm{Min}_2}(x, \frac{1}{3}, \delta_2)} = d(\pi_{A_{n,M}^{\mathrm{Min}_2}(x, \frac{1}{3}, \delta_2)}) \\
A_m^{\mathrm{TSP}_3}(D, \delta) &= g\left(A_{n,M}^{\mathrm{Min}_2}(x, \tfrac{1}{3}, \delta_2)\right)
\end{aligned}$$

solve the traveling salesman problems TSP_j for $j = 1, 2, 3$, respectively, with probability $1 - \delta_2 = (1-\delta)^{1/2}$. We also know the proper M with probability $(1-\delta)^{1/2}$. Therefore, the whole algorithm works with probability $1 - \delta$. This proves the following theorem.

Theorem 9.1. *The traveling salesman problems TSP_j are solved with probability $1-\delta$ by the quantum algorithms $A_m^{\mathrm{TSP}_j}$ using of order*

- *for $j = 1$,*
  - $m \log m\,(\log \delta^{-1} + \log m + \log d_{\max})\,(\log m + \log d_{\max})$ *power queries and*
  - $m \log m$ *qubits,*
- *for $j = 2, 3$,*
  - $$\begin{aligned} m^2 \log^2 m \quad & \big(\log \delta^{-1} + \log m\big) \\ + \quad & m \log m \big(\log \delta^{-1} + \log m + \log d_{\max}\big)\,(\log m + \log d_{\max}) \end{aligned}$$

    power queries and
  - $m \log m$ *qubits.*

Acknowledgments

This research has been supported in part by the National Science Foundation, the Defense Advanced Research Projects Agency, and the Air Force Research Laboratory.

We are grateful for valuable comments and suggestions how to improve the presentation of the paper from Stefan Heinrich, Marek Kwas, Klaus Meer Joseph F. Traub and Mihalis Yannakakis.

References

[1] Abrams, D. S. and Lloyd, S. (1999), *Quantum Algorithm Providing Exponential Speed Increase for Finding Eigenvalues and Eigenvectors* Phys. Rev. Lett., 83, 5162–5165.

[2] Babuska, I. and Osborn, J. (1991), *Eigenvalue Problems*, in Handbook of Numerical Analysis, Vol. II, P. G. Ciarlet and J. L. Lions, eds., North-Holland, Amsterdam, 641–787.

[3] Bennet, C. H., Bernstein, E., Brassard, G. and Vazirani, U. (1997), *Strengths and weaknesses of quantum computing* SIAM J. Computing, 26(5), 1510–1523.

[4] Bernstein, E., and Vazirani, U. (1997), *Quantum complexity theory*, SIAM J. Computing, 26(5), 1411–1473.

[5] Bessen, A. J. (2005), *A lower bound for phase estimation on a quantum computer*, Physical Review A, 71(4). Also http://arXiv.org/quant-ph/0412008.

[6] Brassard, G., Høyer, P., Mosca, M., and Tapp, A. (2002), *Quantum Amplitude Amplification and Estimation* in Contemporary Mathematics, Vol. 305, Am. Math. Soc., 53–74. Also http://arXiv.org/quant-ph/0005055.

[7] Courant, C. and Hilbert, D. (1989), Methods of Mathematical Physics, Vol. I, Wiley Classics Library, Willey-Interscience, New York.

[8] Garey, M. R. and Johnson, D. S. (1979), Computers and Intractability, A Guide to the Theory of NP-Completeness, W. H. Freeman and Company, New York.

[9] Grover, L. (1997), *Quantum mechanics helps in searching for a needle in a haystack*, Phys. Rev. Lett., 79(2), 325–328. Also http://arXiv.org/quant-ph/9706033.

[10] Heinrich, S. (2002), *Quantum Summation with an Application to Integration*, J. Complexity, 18(1), 1–50. Also http://arXiv.org/quant-ph/0105116.

[11] Heinrich, S. (2003), *Quantum integration in Sobolev spaces*, J. Complexity, 19, 19–42.

[12] Heinrich, S. (2004), *Quantum Approximation I. Embeddings of Finite Dimensional L_p Spaces,* J. Complexity, 20(1), 5–26. Also http://arXiv.org/quant-ph/0305030.

[13] Heinrich, S. (2004), *Quantum Approximation II. Sobolev Embeddings,* J. Complexity, 20(1), 27–45. Also http://arXiv.org/quant-ph/0305031.

[14] Jaksch, P. and Papageorgiou, A. (2003), *Eigenvector approximation leading to exponential speedup of quantum eigenvalue calculation*, Phys. Rev. Lett., 91, 257902. Also http://arXiv.org/quant-ph/0308016.

[15] Keller, H. B. (1968), Numerical methods for two-point boundary-value problems, Waltham, Mass., Blaisdell.

[16] Nayak, A. and Wu, F. (1999), *The quantum query complexity of approximating the median and related statistics*, STOC, 384-393. See also LANL preprint quant-ph/9804066.

[17] Nielsen, M.A. and Chuang, I.L. (2000), Quantum Computation and Quantum Information, Cambridge University Press, Cambridge, UK.

[18] Novak, E. (2001), *Quantum complexity of integration,* J. Complexity, 17, 2–16. Also http://arXiv.org/quant-ph/0008124.

[19] Novak, E., Sloan, I. H. and Woźniakowski, H. (2004), *Tractability of approximation for weighted Korobov spaces on classical and quantum computers*, Journal of Foundations of Computational Mathematics, 4(2), 121–156. Also http://arXiv.org/quant-ph/0206023.

[20] Papageorgiou, A. and Woźniakowski, H. (2005) *Classical and quantum complexity of the Sturm-Liouville eigenvalue problem,* Quantum Information Processing, 4, 87–127. Also http://arXiv.org/quant-ph/0502054.

[21] Shor, P. W. (1997), *Polynomial-time algorithms for prime factorization and discrete logarithm on a quantum computer*, SIAM J. Comput., 26(5), 1484–1509.

[22] Strang, G. and Fix, G. J. (1973), An Analysis of the Finite Element Method, Prentice-Hall, Englewood Cliffs, NJ.

[23] Titschmarsh, E. C. (1958), Eigenfunction Expansions Associated with Second-Order Differential Equations, Part B, Oxford University Press, Oxford, UK.

[24] Traub, J. F. and Woźniakowski, H. (2002), *Path integration on a quantum computer,* Quantum Information Processing, 1, 365–388, 2002. Also http://arXiv.org/quant-ph/0109113.