Can Quantum Search Accelerate Evolutionary Algorithms?

Daniel Johannsen
Max Planck Institute for Informatics
Department of Algorithms and Complexity
Saarbrücken, Germany
daniel.johannsen@mpi-inf.mpg.de

Piyush P Kurur*
Dept of Comp. Sci. and Engg
Indian Institute of Technology Kanpur
Kanpur UP, India 208016
ppk@cse.iitk.ac.in

Johannes Lengler
Department of Mathematics
Saarland University
Saarbrücken, Germany
johannes.lengler@math.uni-sb.de

ABSTRACT

In this article, we formulate for the first time the notion of a quantum evolutionary algorithm. In fact we define a quantum analogue for any elitist (1+1) randomized search heuristic. The quantum evolutionary algorithm, which we call *(1+1) quantum evolutionary algorithm* (QEA), is the quantum version of the classical (1+1) evolutionary algorithm (EA), and runs only on a quantum computer. It uses Grover search [13] to accelerate the search for improved offsprings.

To understand the speedup of the (1+1) QEA over the (1+1) EA, we study the three well known pseudo-Boolean optimization problems OneMax, LeadingOnes, and Discrepancy. We show that although there is a speedup in the case of OneMax and LeadingOnes in the quantum setting, the speedup is less than quadratic. For Discrepancy, we show that the speedup is at best constant.

The reason for this inconsistency is due to the difference in the probability of making a successful mutation. On the one hand, if the probability of making a successful mutation is large then quantum acceleration does not help much. On the other hand, if the probabilities of making a successful mutation is small then quantum enhancement indeed helps.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

*Work done on a visit to the Max Planck Institute for Informatics (MPII) funded by MPII and Research I project (NRNM/CS/20030163)

General Terms

Theory, Algorithms

Keywords

Running time analysis, Quantum Algorithm, Theory

1. INTRODUCTION

One of the most prominent computational problems which a quantum algorithm solves more efficiently than classical algorithms is searching in an unordered database. In his seminal work [13], Grover gave an algorithm which can search in an unordered data base of N elements in time proportional to $\sqrt{N}$, whereas any classical algorithm requires time proportional to N.

Following this result, algorithms based on Grover's search have drawn much attention in the last decade. Moreover, there are many problems for which specialized algorithms have been designed, such as searching [13, 7], Element Distinctness [19], Minimum-Finding [12] and many others (e.g., [10, 5, 23]).

Grover's search is known to be optimal [4, 22] when the underlying search space has no structure. Grover's search can be thought of as evaluating the boolean function OR on N bits. For evaluating certain boolean functions like XOR, quantum algorithms give no advantage over classical ones — both have a query complexity[1] of $\Theta(N)$ [2, 3].

Optimization problems, which is the topic of interest of this paper, have also received much attention in the quantum setting. Using Grover's algorithm, Dürr, Heiligman, Høyer, and Mhalla [11] have shown that it is possible to find the global optimum of a black-box optimization problem on the search space $\{0,1\}^n$ in an expected number of $O(2^{n/2})$ queries. Moreover, a matching lower bound of $\Omega(2^{n/2})$ for all possible quantum algorithms exists [22].

In addition, if there is enough structure in the search space, better bounds can be shown. For example on general

[1]In the standard literature on the theory of evolutionary algorithms the term *optimization time* is used instead of the term *query complexity*. Here we use the latter term since in the context of our investigations it seems more intuitive to us.

	RLS / RLS* / (1+1) EA / (1+1) EA*	QLS / (1+1) QEA	QLS* / (1+1) QEA*
OneMax	$\Theta(n \log n)$	$\Theta(n)$	$\Theta(n)$
LeadingOnes	$\Theta(n^2)$	$\Theta(n^{3/2})$	$\Theta(n^2)$
Disc	$\Theta(\sqrt{n})$	$\Theta(\sqrt{n})$	$\Theta(\sqrt{n})$

Table 1: A comparison of the expected optimization times (query complexities) between RSHs and QSHs on the pseudo-Boolean objective functions OneMax, LeadingOnes, and Disc.

graph-based search spaces, Magniez, Nayak, Roland, and Santha [15, Theorem 3] have shown that if the Markov chain associated with the random walk on the space is ergodic, significant improvement in the expected query complexity is possible provided that the spectral gap is large. Furthermore, if the underlying *quantum random walk* is symmetric, superior problem-specific quantum algorithms are available [21, 16].

In this article, we consider quantum versions of elitist (1+1) *randomized search heuristics* (RSHs), that is, heuristics that successively generate candidate solutions according to some distribution depending only on the current candidate solution and select the candidate solution if and only if there is an improvement. Since the Markov processes underlying these algorithms are not ergodic and far from symmetric, the setting of quantum random walks as in [21, 16, 15] does not apply.

An elitist evolutionary algorithm for an optimization problem can never move from a solution of higher objective value to a solution of smaller objective value. However all quantum operations, other than measurements, are required to be reversible. Thus, in order to simulate the behavior of an elitist (1+1) RSH, it is necessary to perform a measurement after every elitist selection step, basing the further decisions of the algorithm on the outcome of this measurement.

Given a finite search space $\mathcal{S}$, typically encoded as n-bit strings, and an objective function f from $\mathcal{S}$ to $\mathbb{R}$, we want to compute an optimum (i.e., either a maximum or a minimum) of f. Such optimization problems are called pseudo-Boolean optimization problem. The elitist (1+1) RSHs we consider work in the following way. They start with a candidate solution $\mathbf{x}_0$ and repeatedly improve the solution by performing the following two steps:

(1) generate a new solution $\mathbf{y}$ according to a distribution $p_{\mathbf{x}}(\mathbf{y})$ depending on the current solution $\mathbf{x}$;

(2) if the new solution $\mathbf{y}$ is better then retain it, otherwise discard it.

Thus, elitist (1+1) RSHs only differ in the nature of the distribution $p_{\mathbf{x}}$. For example, Randomized Local Search (RLS) selects an index i at random and flips the bit x_i to get the new candidate solution whereas the (1+1) Evolutionary Algorithm (EA) flips each bit x_i with probability $1/n$. We will indicate algorithms that retain solutions of equal fitness by * (e.g. RLS*). In the conclusion we discuss the differences.

The main idea of the paper is to use *quantum probability amplification*, which is a reformulation of Grover's search [8], to speed up the generation phase, i.e., step (1). Instead of picking a new candidate solution directly from the distribution, which is what is done classically, we amplify the probability of getting a better solution to say a constant $1/2$ using quantum probability amplification (see Section 2). If $p_{\mathbf{x}} = \sum_{f(\mathbf{y})>f(\mathbf{x})} p_{\mathbf{x}}(\mathbf{y})$ is the probability to obtain a better solution (assuming a maximization problem) from a candidate solution $\mathbf{x}$ in the classical setting, then in order to do so the quantum probability amplification requires only $\Theta\big(1/\sqrt{p_{\mathbf{x}}}\big)$ queries to f in expectation as opposed to $1/p_{\mathbf{x}}$ in the classical setting. We call this quantum variant of RSH a *Quantum Search Heuristic* (QSH). In particular, we call the quantum variants of RLS and EAs *Quantum Local Search* (QLS), and *Quantum Evolutionary Algorithms* (QEAs). These RSHs can only run on a quantum computer.

The quantum local search which we define here is a restricted version of the quantum algorithm by Aaronson [1] which first chooses $\Theta(n^{1/3}2^{2n/3})$ search points uniformly at random and then uses Grover search to determine the optimal initial search point among them. The algorithm of Aaronson then proceeds exactly like ours. However, our algorithm does not attempt to optimize on the starting point, because (i) the runtime of such an optimization would dominate the runtimes of our algorithms by orders of magnitude and (ii) the classical RSHs we compare with do not attempt to do so either.

There are two other streams of work which sound similar to our work but are in fact not at all related. Our results on QSHs and QEAs are significantly different from that of Quantum-Inspired Evolutionary Algorithms (QIEAs) as introduced in [14]. QIEAs are classical algorithms where the mutation and selection steps, though classical, are inspired from quantum operations. However our mutation process is genuinely quantum and cannot be implemented on a classical computer. On the other hand, our algorithms are not attempts to apply genetic programming techniques to better design quantum algorithms unlike for example the work of Spector *et. al.* [20] where the "code" of an ordinary quantum algorithm is optimized by an evolutionary algorithm. To the best of our knowledge, the (1+1) QEA investigated here is the first attempt to generalize evolutionary algorithms to the quantum setting.

The general bound of $\Theta(2^{n/2})$ for expected query complexity in optimizing an arbitrary pseudo-boolean function in the black-box model also applies to QSHs. However, we may ask whether QSHs also experience a quadratic speedup over ordinary RSHs. In order to answer this question, we follow the approaches of [6] and [9] and study the behavior of QLS and the (1+1) QEA on specific pseudo-Boolean optimization problems.

In particular, we study the query complexity of these QSHs to maximize the objective function LeadingOnes and to minimize the objective functions OneMax and Disc. In all three cases, the speedup over the their classical counterparts are not quadratic. Moreover, it differs for each of the problems. As can be seen in Table 1, the speedup is by a

factor of $\Theta(\log n)$ for OneMax and by $\Theta(\sqrt{n})$ for LeadingOnes, while there is no asymptotic speedup for the function Disc.

We now give a broad reason for the lack of speed up in certain cases. The quantum acceleration does not differ form its classical counter part in the statistical nature of the candidate solutions picked on its way to the optimal solution. It speeds up by reducing the expected time required for a successful mutation. For LeadingOnes, it is rather hard to find the next search point, so there is a substantial speedup. On the other hand, for Disc it is very easy to find a better search point: the expected time for improving the fitness function is constant, and so there is no any asymptotic speedup. On OneMax the performance improvement, though present, is less that that of LeadingOnes.

Summing up, we see that quantum search may speed up evolutionary algorithms in some cases. It may give a quadratic speedup at most, and there are problems which are substantially accelerated by quantum search. However, it depends on the specific problem how much is really gained, and for some problems there is no improvement in the expected running time at all. In Lemma 9, we give a precise statement that enables us to analyze the benefits of quantum search purely in non-quantum terms.

2. PROBABILITY AMPLIFICATION

In this section we describe the Quantum search algorithm and its reformulation quantum probability amplification in a form that is suitable for our purpose. As before, let $\mathcal{S}$ denote the set $\{0,1\}^n$ of all n-bit strings. Let $\mathcal{S}_0$ be a subset of $\mathcal{S}$ for which we are given a membership oracle, i.e. we are given a oracle M from $\mathcal{S}$ to $\{0,1\}$ such that $\mathcal{S}_0 = \{\mathbf{x}|M(\mathbf{x}) = 1\}$. Our task is to *search* for a string $\mathbf{x}_0$ in $\mathcal{S}_0$ using queries to M. In this setting, we are interested in minimizing the number of queries made to M.

In an important breakthrough, Grover [13] gave a quantum algorithm to search for such an element $\mathbf{x}_0 \in \mathcal{S}_0$ that makes only $\sqrt{|\mathcal{S}|/|\mathcal{S}_0|}$ queries to the oracle M. One needs to, however, make the oracle M work for quantum states. The standard approach, which we describe briefly for completeness, is to consider the membership oracle as unitary operator U_M on the n-qubit Hilbert space $\mathcal{H} = \mathbb{C}^{2^{\otimes^n}}$ defined as

$$U_M |\mathbf{x}\rangle = (-1)^{M(\mathbf{x})} |\mathbf{x}\rangle .$$

One application of this unitary operator U_M is considered as a single query to the membership oracle.

Let N denote the cardinality of the search space $\mathcal{S}$, and let the cardinality of the set $\mathcal{S}_0$ be M. During initialization, Grover's quantum search algorithm prepares the uniform superposition $|\psi_0\rangle = 1/\sqrt{N}\sum_{\mathbf{x}\in\mathcal{S}} |\mathbf{x}\rangle$. The algorithm iteratively applies the Grover step, a unitary operator which we denote by G, to $|\psi_0\rangle$. Let $|\psi_t\rangle$ denote the state after t applications of G, i.e. $|\psi_t\rangle = G^t |\psi_0\rangle$. If we choose some appropriate t in $O(\sqrt{N/M})$ then on measuring the state $|\psi_t\rangle$ we obtain an element $\mathbf{x} \in \mathcal{S}_0$ with constant probability. More precisely, if we write the state as $|\psi_t\rangle = \sum_{\mathbf{x}\in\mathcal{S}} \alpha_x(t) |\mathbf{x}\rangle$, then for $t = O(\sqrt{N/M})$ we have $\sum_{\mathbf{x}\in\mathcal{S}_0} |\alpha_{\mathbf{x}}(t)|^2$ is a constant (say 1/2). The exact form of the Grover step G is not relevant (for details see the text book of Nielsen and Chuang [18, Chapter 6]) but the crucial point is that G can be constructed using one application of the unitary operator U_M. Hence the Grover search makes $\sqrt{N/M}$ queries to the oracle.

Grover search starts with the uniform superposition as a priori there is no specific reason to prefer one bit string over the other. Instead if we start the search algorithm with the state $|\psi_0\rangle = \sum \alpha_{\mathbf{x}} |x\rangle$, then running time will be $\sqrt{1/p}$ where $p = \sum_{\mathbf{x}\in\mathcal{S}_0} |\alpha_{\mathbf{x}}|^2$ is the probability of picking $\mathbf{x} \in \mathcal{S}_0$ had we measured the initial state $|\psi_0\rangle$ directly. This reformulation due to Brassard *et al* [8] is often called the *quantum probability amplification* or *quantum amplitude amplification* as a quantum algorithm is able to amplify the probability by making just $\sqrt{1/p}$ queries as opposed to $1/p$ required by a classical algorithm.

Grover's search algorithm, however, comes with a caveat. One needs to stop the Grover iteration after $\Theta(\sqrt{N/M})$ steps, for otherwise the probability of getting a favorable $\mathbf{x}_0 \in \mathcal{S}_0$ actually deteriorates. Thus it appears as if without knowing the count $|\mathcal{S}_0|$, or in the case of probability amplification, the probability p of sampling an $x \in \mathcal{S}_0$ under the given distribution, one cannot use Grover search. However, using phase estimation, Brassard *et al* [8] gave a way to overcome this difficulty with essentially no change in the overall running time. From now on, by quantum probability amplification we mean this generalized version where we do not need to know the probabilities.

Let $|\psi\rangle$ be any state in the Hilbert space of n-qubits and let $|\varphi\rangle = G|\psi\rangle$. Let $\mathcal{D}_\psi$ and $\mathcal{D}_\varphi$ be the probability distributions obtained on $\mathcal{S}$ on measuring $|\psi\rangle$ and $|\varphi\rangle$, respectively. An important property of the Grover operation G is that the conditional probabilities of obtaining $\mathbf{x} \in \mathcal{S}_0$ given $\mathcal{S}_0$ is the same with respect to either of the distribution $\mathcal{D}_\psi$ and $\mathcal{D}_\varphi$ (see the analysis of Grover's search in Section 6.1.3 of Nielsen and Chuang's book [18]). Thus if we start the Grover search on the state $|\psi_0\rangle$ and perform the Grover search, although the probability of obtaining $\mathbf{x} \in \mathcal{S}_0$ improves, the conditional probabilities of obtaining an element in $\mathcal{S}_0$ given the event that we have obtained an element in $\mathcal{S}_0$ remains the same as in the beginning of the algorithm.

Given a classical sampling algorithm A, consider the quantum algorithm that uses probability amplification starting with the initial state obtained by sampling an $\mathbf{x} \in \mathcal{S}$ using the (quantum version of the) sampling algorithm A followed by measurement and repeats the process until it succeeds in getting an $\mathbf{x}_0 \in \mathcal{S}_0$. Since the probability has been amplified to a constant we would repeat this process at most a constant time. Further since the conditional probability of getting a particular $\mathbf{x}_0 \in \mathcal{S}_0$ does not changes after every Grover step, the quantum algorithm will produce a sample $\mathbf{x}_0 \in \mathcal{S}_0$ with probability $\text{Prob}[\mathbf{x}_0|\mathcal{S}_0]$. Hence we have the following reformulation of probability amplification which is more suitable for our analysis.

Theorem 1 (Probability Amplification).
Let $\mathcal{S}$ be a finite search space, $\mathcal{S}_0$ be any subset of $\mathcal{S}$ for which there is a membership oracle M, and A a sampling procedure that produces a distribution $\mathcal{D}_A$ on $\mathcal{S}$. Let p be the probability $\text{Prob}_{\mathcal{D}_A}[\mathbf{x} \in \mathcal{S}_0]$ of obtaining a element in $\mathcal{S}_0$ on running A. Then there exists a quantum algorithm that makes on expectation $\Theta(p^{-1/2})$ queries to the membership oracle M and samples an element $\mathbf{x}_0$ in $\mathcal{S}_0$ with a distribution $\mathcal{D}_\psi$ on $\mathcal{S}_0$ given by

$$\text{Prob}_{\mathcal{D}_\psi}[\mathbf{x} = \mathbf{x}_0] = \text{Prob}_{\mathcal{D}_A}[\mathbf{x} = \mathbf{x}_0|\mathbf{x} \in \mathcal{S}_0] .$$

The above quantum algorithm uses probability amplifica-

tion starting with the initial state obtained by sampling an $\mathbf{x} \in \mathcal{S}$ using the (quantum version of the) sampling algorithm A followed by measurement and repeats the process until it succeeds in getting an $\mathbf{x}_0 \in \mathcal{S}_0$. Since the probability has been amplified to a constant we would repeat this process at most a constant time each of which costs $\sqrt{1/p}$ queries. Further since the conditional probability of getting a particular $\mathbf{x}_0 \in \mathcal{S}_9$ does not changes, we have the above result.

3. THE QUANTUM SEARCH HEURISTIC

In this section we study quantum versions of known elitist (1+1) Randomized Search Heuristics (short RSHs). Given any RSH like Random Local Search (RLS) or the (1+1) Evolutionary Algorithm (EA), we use quantum probability amplification at each of its mutation and selection steps. We call such a version a quantum version of the search heuristic.

Let $\mathcal{S}$ be the search space and let f be a function from the search space $\mathcal{S}$ to $\mathbb{R}$ that we want to maximize. Randomized search heuristics like random local search (RLS) and evolutionary algorithms can be formalized by defining what is known as its *mutation operator*.

DEFINITION 2 (MUTATION OPERATOR MUT).
Let $\mathcal{S}$ be a finite search space. A mutation operator MUT *over $\mathcal{S}$ is a function from $\mathcal{S}$ to the space of probability distributions on $\mathcal{S}$.*

The mutation operator MUT is essentially the search strategy of the corresponding RSH. With a slight abuse of notation we write $\text{MUT}(\mathbf{x})$ to denote a sample picked from $\mathcal{S}$ according to the distribution $\text{MUT}(\mathbf{x})$.

ALGORITHM 1 (RSH).
The elitist (1+1) randomized search heuristic (RSH) *over the finite search space $\mathcal{S}$ with mutation operator* MUT *that maximizes the objective function $f : \mathcal{S} \to \mathbb{R}$ is the following iterative algorithm:*

1. *Start with $\mathbf{x}_0 \in \mathcal{S}$ uniformly at random.*
2. *For each $t \geq 0$ iteratively assume that $\mathbf{x}_t$ had been picked*
   (a) *Pick $\mathbf{y}_t \in \mathcal{S}$ according to the distribution* $\text{MUT}(\mathbf{x}_t)$.
   (b) *Set $\mathbf{x}_{t+1} = \mathbf{y}_t$ if $f(\mathbf{y}_t) > f(\mathbf{x}_t)$. Otherwise, discard $\mathbf{y}_t$ and repeat (a).*

One can define a randomized search heuristic for minimizing f by changing step 2b of Algorithm 1: set $\mathbf{x}_{t+1} = \mathbf{y}_t$ if $f(\mathbf{y_t}) < f(\mathbf{x}_t)$.

It follows from our definition of RSH that all the candidate solutions $\mathbf{x}_t$ are distinct. Furthermore, for maximization problem, the sequence of reals $\{f(\mathbf{x}_t)\}_{t=0}^{\infty}$ form an increasing sequence.

Having defined the RSH associated with a mutation operator MUT, we now define its quantum version. As before, we assume (without loss of generality) that our task is to maximize f. We can think of the Step 2a of Algorithm 1 as follows: Generate a distribution on $\mathcal{S}$ according to $\text{MUT}(\mathbf{x}_t)$ and sample a candidate solution $\mathbf{y}_t$ out of it. In the quantum version, we do not pick a $\mathbf{y}_t$ directly from the distribution $\text{MUT}(\mathbf{x}_t)$. Instead we amplify the probability of getting a favorable $\mathbf{y}_t$ using quantum probability amplification and then measure to obtain $\mathbf{y}_t$. For this we need a membership oracle and the objective function gives us just that. Given f, we define for each $\mathbf{x} \in \mathcal{S}$, a membership oracle $M_{f,\mathbf{x}}$ as follows:

$$M_{f,\mathbf{x}}(\mathbf{y}) = \begin{cases} 1 & \text{if } f(\mathbf{y}) > f(\mathbf{x}), \\ 0 & \text{otherwise.} \end{cases}$$

We now define elitist (1+1) Quantum Search Heuristics (QSH) associated with a mutation MUT.

ALGORITHM 2 (QSH).
The elitist (1+1) quantum search heuristic (QSH) *over the finite search space $\mathcal{S}$ with mutation operator* MUT *that maximizes the objective function $f : \mathcal{S} \to \mathbb{R}$ is the following iterative algorithm:*

1. *Start with $\mathbf{x}_0 \in \mathcal{S}$ uniformly at random.*
2. *For each $t \geq 0$ iteratively assume that $\mathbf{x}_t$ had been picked. Sample $\mathbf{x}_{t+1}$ according to sampling procedure for Theorem 1 with search space space $\mathcal{S}$, membership oracle $M_{f,\mathbf{x}_t}$, and sampling procedure* $\text{MUT}(\mathbf{x}_t)$.

Instead of the rather strong condition "$f(\mathbf{y}_t) > f(\mathbf{x}_t)$" for accepting a new search point in Step 2.(b), often the weaker condition "$f(\mathbf{y}_t) \geq f(\mathbf{x}_t)$ with $\mathbf{y}_t \neq \mathbf{x}_t$" is chosen. This results in variants of the two above algorithms which we call RSH* and QSH*.

ALGORITHM 3 (RSH* AND QSH*).
By RSH*, we denote a RSH where in Step 2.(b) the condition "$f(\mathbf{y}_t) > f(\mathbf{x}_t)$" is replaced by "$f(\mathbf{y}_t) \geq f(\mathbf{x}_t)$ and $\mathbf{y}_t \neq \mathbf{x}_t$".
By QSH*, we denote a QSH where in Step 2. the membership oracle $M_{f,\mathbf{x}_t}$ is replaced by the membership oracle

$$M^*_{f,\mathbf{x}}(\mathbf{y}) = \begin{cases} 1 & \text{if } f(\mathbf{y}) \geq f(\mathbf{x}) \text{ and } y \neq x, \\ 0 & \text{otherwise.} \end{cases}$$

It is important to note that Theorem 1 implies that the probability that the algorithm takes some fixed trajectory $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t$ through the search space is exactly the same in both the RSH and is quantum counterpart QSH. The only place where they differ is in the expected time required by them to make an improvement from $\mathbf{x}_i$ to $\mathbf{x}_{i+1}$.

We are interested in the *query complexity* of the RSHs and QSHs as defined above.

DEFINITION 3 (QUERY COMPLEXITY).
Let $\mathcal{S}$ be a search space with objective function $f\colon \mathcal{S} \to \mathbb{R}$ and mutation operator MUT. *The* query complexity *of the corresponding RSH or QSH is the random variable that denotes the number of queries until the RSH or QSH has found an optimal search point. A* query of a RSH *is an evaluation of f and a* query of a QSH *is a call to the membership oracle associated to f.*

Another random quantity of interest is the number of improvements until a RSH or QSH has found an optimum.

DEFINITION 4 (IMPROVEMENT NUMBER T).
Let $\mathcal{S}$ be a search space with objective function $f\colon \mathcal{S} \to \mathbb{R}$ and mutation operator MUT. *The* improvement number T *of the corresponding RSH or QSH is the random variable that denotes the first point in time t such that $\mathbf{x}_t$ is optimal with respect to f.*

Note that T indeed is the number of improvements in the algorithm since t increases only if the objective function increases.

In order to relate the query complexity to the improvement number, we need the notion of *transition probabilities*.

DEFINITION 5 (TRANSITION PROBABILITY p_t). *Let $\mathcal{S}$ be a search space with objective function $f\colon \mathcal{S} \to \mathbb{R}$ and mutation operator* MUT.

For all $t \in \mathbb{N} = \mathbb{N}_0$, the transition probability p_t *at time t of the corresponding RSH is the probability that $\textsc{mut}(x_t)$ is an improvement, that is,*

$$p_t = \text{Prob}[f(\textsc{mut}(x_t)) > f(x_t)]\,.$$

For a classical RSH, the transition probability tells us how likely we are to advance from $\mathbf{x}_t$ to $\mathbf{x}_{t+1}$ by a single query to f. Therefore, p_t^{-1} equals the expected number of queries needed to advance from $\mathbf{x}_t$ to $\mathbf{x}_{t+1}$. This observation leads us to the following definition.

DEFINITION 6 (T_{RSH}). *Let $\mathcal{S}$ be a search space with objective function $f\colon \mathcal{S} \to \mathbb{R}$ and mutation operator* MUT.

The estimated query complexity T_{RSH} *of the corresponding RSH is the random variable*

$$T_{\text{RSH}} = \sum_{t \le T} p_t^{-1}\,.$$

By the discussion above, we obtain the following lemma.

LEMMA 7. *Let $\mathcal{S}$ be a search space with objective function $f\colon \mathcal{S} \to \mathbb{R}$ and mutation operator* MUT. *Then it holds for the corresponding RSH that the expected query complexity equals the expected estimated query complexity.*

Now let us consider the QSH. Theorem 1 tells us two things. Firstly, if we fix an $\mathbf{x}$ and a t, then the probability $\text{Prob}(\mathbf{x}_t = \mathbf{x})$ is the same for the RSH and the QSH. In particular, the transition probability p_t does not change if we switch to QSH. Secondly, the expected number of queries needed to advance from $\mathbf{x}_t$ to $\mathbf{x}_{t+1}$ is in $\Theta(p_t^{-1/2})$. This leads to the following definition.

DEFINITION 8 (T_{QSH}). *Let $\mathcal{S}$ be a search space with objective function $f\colon \mathcal{S} \to \mathbb{R}$ and mutation operator* MUT.

The estimated query complexity T_{QSH} *of the corresponding QSH is the random variable*

$$T_{\text{QSH}} = \sum_{t \le T} p_t^{-1/2}\,.$$

Again, the discussion above yields us the following lemma.

LEMMA 9. *Let $\mathcal{S}$ be a search space with objective function $f\colon \mathcal{S} \to \mathbb{R}$ and mutation operator* MUT. *Then it holds for the corresponding QSH that the expected query complexity is of the same order as the expected estimated query complexity.*

Finally, we introduce the two RSHs which we compare to their quantum versions: Randomized Local Search (RLS) and the (1+1) Evolutionary Algorithm (EA). We do this by defining their mutation operators such that the quantum version follows directly.

DEFINITION 10 (RLS AND THE (1+1) EA). Randomized Local Search (RLS) *and the* (1+1) Evolutionary Algorithm (EA) *are the elitist (1+1) randomized search heuristics associated with the mutation operators* $\textsc{mut}_{\text{RLS}}$ *and* $\textsc{mut}_{\text{EA}}$, *respectively, which are defined as follows.*

- $\textsc{mut}_{\text{RLS}}(\mathbf{x})$ *with $x \in \{0,1\}^n$ is the probability distribution on $\{0,1\}^n$ obtained by flipping one bit of $\mathbf{x}$ uniformly randomly.*
- $\textsc{mut}_{\text{EA}}(\mathbf{x})$ *with $x \in \{0,1\}^n$ is the probability distribution on $\{0,1\}^n$ obtained by flipping each bit of $\mathbf{x}$ independently with probability $1/n$.*

For the rest of the paper, we fix the search space $\mathcal{S}$ to be the set $\{0,1\}^n$ of bit-strings of length n. A bit-string in $\mathcal{S}$ will be denoted by bold faced Latin letters like $\mathbf{x}$, $\mathbf{y}$ etc. In particular, $\mathbf{x}_t$ will denote the search point of the corresponding RSH or QSH after t improvements, and t runs from 0 to T, which is the total number of improvements. The i-th bit of a bit-string $\mathbf{x}$ will be denoted by $\mathbf{x}(i)$.

4. ONEMAX

The pseudo-Boolean function ONEMAX counts the number of one-bits in a bit-string $\mathbf{x} \in \{0,1\}^n$, that is, let

$$\textsc{OneMax}(\mathbf{x}) := \sum_{i=1}^{n} \mathbf{x}(i)\,. \tag{4.1}$$

The following theorem can be deduced from the results and proofs in [9].

THEOREM 11. *Let $\{\mathbf{x}_t\}_{t\in\mathbb{N}}$ be the search points generated by the (1+1) EA or RLS minimizing* ONEMAX. *Then the expected query time is in $\Theta(n \log n)$.*

We show that the expected query time in the quantum version decreases only by a logarithmic factor.

THEOREM 12. *Let $\{\mathbf{x}_t\}_{t\in\mathbb{N}}$ be the search points generated by the (1+1) QEA or QLS minimizing* ONEMAX. *Then the expected query time is in $\Theta(n)$.*

PROOF. Let us first consider QLS. For all $t \ge 0$, let $k_t = \textsc{OneMax}(\mathbf{x}_t)$ be the number of one-bits in $\mathbf{x}_t$.

Consider $t \ge 1$. The elements of $\mathcal{S}_t$ are precisely those that are obtained from $\mathbf{x}_{t-1}$ by flipping a one-bit. Therefore, $\mathcal{S}_t$ contains exactly k_{t-1} elements, one for every one-bit in $\mathbf{x}_{t-1}$. Therefore the probability that $\textsc{mut}(\mathbf{x}_{t-1}) \in \mathcal{S}_t$ equals $p_t = k_{t-1}/n$. Furthermore, since exactly one one-bit is flipped, we have $k_t = k_{t-1} - 1$. Recursively, we see $k_t = k_0 - t$, and thus $T = k_0$. Hence,

$$\text{E}[T_{\text{QSH}} \mid k_0] = \sum_{t=1}^{k_0} p_t^{-1/2} = \sum_{t=1}^{k_0} \sqrt{\frac{n}{k_0 - t + 1}} = \sum_{t=1}^{k_0} \frac{\sqrt{n}}{\sqrt{t}}\,.$$

Therefore, since $\sum_{t=1}^{k_0} t^{-1/2} \in \Theta\big(\int_1^{k_0} x^{-1/2} dx\big)$, it holds that $\text{E}[T_{\text{QSH}} \mid k_0] \in \Theta(\sqrt{k_0\, n})$.

Since $\mathbf{x}_0$ is chosen uniformly at random from $\{0,1\}^n$ it holds that $k_0 \ge n/2$ with probability at least $1/2$. Therefore, $\text{E}[T_{\text{QSH}}] = \Omega(n)$. On the other hand, $k_0 \le n$, and so $\text{E}[T_{\text{QSH}}] = O(n)$.

Now, let us turn to (1+1) QEA with k_t as for the QLS. Let $t \ge 1$. First note that all vectors in $\mathcal{S}_t$ are obtained from $\mathbf{x}_{t-1}$ by flipping at least one one-bit. Therefore, by the union bound

$$p_t \le \frac{k_{t-1}}{n}\,,$$

as at least one of the k_{t-1} one-bits in $\mathbf{x}_{t-1}$ has to be flipped. Conversely, the set $\mathcal{S}_t$ contains at least all those boolean

vectors that are obtained from $\mathbf{x}_{t-1}$ by flipping a one-bit and not flipping any other bit. If we fix a one-bit, then the probability for this event to happen is

$$\frac{1}{n}\Big(1-\frac{1}{n}\Big)^{n-1} \geq \frac{1}{\mathrm{e}\,n}$$

Since we have exactly k_{t-1} one-bits, we obtain the lower bound

$$p_t \geq \frac{k_{t-1}}{\mathrm{e}\,n}\,.$$

Hence,

$$\sum_{t=1}^{T}\sqrt{\frac{n}{k_{t-1}}} \leq T_{\mathrm{QSH}} \leq \sum_{t=1}^{T}\sqrt{\frac{\mathrm{e}\,n}{k_{t-1}}}\,,$$

that is, $\mathrm{E}[T_{\mathrm{QSH}}] \in \Theta\big(\sqrt{n}\cdot \mathrm{E}\left[\sum_{t=1}^{T} k_{t-1}^{-1/2}\right]\big)$.

We may now rewrite the sum $\sum k_{t-1}^{-1/2}$. For $\ell \in \{1,\dots,n\}$, let $\delta_\ell := 1$ if there is a $t < T$ such that $k_t = \ell$, and $\delta_\ell := 0$ otherwise. Then,

$$\sum_{t=1}^{T} k_{t-1}^{-1/2} = \sum_{\ell=1}^{n} \delta_\ell \cdot \ell^{-1/2}$$

Thus, $\mathrm{E}[T_{\mathrm{QSH}}] \in \Theta\big(\sqrt{n}\cdot\sum_{\ell=1}^{n}\mathrm{E}[\delta_\ell]\cdot\ell^{-1/2}\big)$.

Unfortunately, unlike for the QLS, we may not assume anymore that $k_t = k_{t-1} - 1$. Thus we prove an upper and a lower bound for $\mathrm{E}[\delta_\ell]$. The upper bound is trivial, as $\mathrm{E}[\delta_\ell] \leq 1$.

Next, we show the corresponding lower bound. Let

$$L := \{\ell \in \{1,\dots,n\} \colon \delta_\ell = 1\}.$$

We may expect that the values in L are fairly evenly distributed. Nevertheless, we pessimistically assume that $L = \{n-|L|+1,\dots,n\}$ in order to bound $\sum_{l\geq 1}\delta_\ell \ell^{-1/2}$. Again, $\sum_{\ell=n-|L|+1}^{n}\ell^{-1/2} \in \Omega\left(\int_{n-|L|+1}^{n} x^{-1/2}dx\right)$, and thus

$$\mathrm{E}[T_{\mathrm{QSH}}] \in \Omega\left(\sqrt{n}\cdot\mathrm{E}\left[\sqrt{n}-\sqrt{n-|L|+1}\,\right]\right)\,. \tag{4.2}$$

To bound the size of L, we consider the expected difference between k_{t-1} and k_t. Conditioned on the event that at least one of the one-bits in $\mathbf{x}_{t-1}$ flips, the difference $k_{t-1}-k_t$ is at most the number of further one-bits in $\mathbf{x}_{t-1}$ that flip. Thus,

$$\mathrm{E}[k_{t-1}-k_t] \leq 1 + \frac{k_{t-1}-1}{n} \leq 2 \quad \text{and}$$

$$\mathrm{E}[k_0 - k_t] = \sum_{s=1}^{t}\mathrm{E}[k_{t-1}-k_t] \leq 2t\,.$$

Hence, $\mathrm{E}[k_0-k_r] \leq k_0/2$ for $r = \lfloor k_0/4\rfloor$. By Markov's Inequality [17], this implies that $k_0 - k_r < k_0$, that is, $k_r > 0$ with probability at least $1/3$. Thus with probability at least $1/3$ it holds that $|L| \geq r+1$, since the values $k_0,\dots,k_r$ all differ.

As we have seen in the case of QLS, also $k_0 \geq n/2$ holds with probability at least $1/2$. Thus, with a probability bounded away from zero by a positive constant, $|L| \geq n/6$. Now we can infer $\mathrm{E}[T_{\mathrm{QSH}}] \in \Omega(n)$ by substituting this result in (4.2). □

5. LEADINGONES

The pseudo-Boolean function LeadingOnes counts the number of one-bits preceding the first zero-bit in a bit-string $\mathbf{x} \in \{0,1\}^n$, that is, let

$$\textsc{LeadingOnes}(\mathbf{x}) := \sum_{k=1}^{n}\prod_{i=1}^{k}\mathbf{x}(i)\,. \tag{5.1}$$

The following theorem can be deduced from [9].

THEOREM 13. *Let $\{\mathbf{x}_t\}_{t\in\mathbb{N}}$ be the search points generated by the (1+1) EA or RLS maximizing* LeadingOnes. *Then the expected query time is in $\Theta(n^2)$.*

The expected query time decreases considerably when we use quantum acceleration. However, it does not decrease quadratically.

THEOREM 14. *Let $\{\mathbf{x}_t\}_{t\in\mathbb{N}}$ be the search points generated by the (1+1) QEA or QLS minimizing* LeadingOnes. *Then the expected query time is in $\Theta(n^{3/2})$.*

PROOF. Let us first consider QLS. For all $0 \leq t \leq T$, let $k_t = \textsc{OneMax}(\mathbf{x}_t)$ be the number of one-bits in $\mathbf{x}_t$, and $i_t = \textsc{LeadingOnes}(\mathbf{x}_t)+1$ be the position of the first zero-bit in $\mathbf{x}_t$ (or $n+1$, if no such bit exists).

Consider $t \geq 1$. Then $\mathcal{S}_t$ contains exactly one element, namely the element that is obtained from $\mathbf{x}_{t-1}$ by flipping the i_{t-1}-th bit. Consequently, this element is chosen and afterwards $k_t = k_{t-1}+1$. Thus, $T = n - k_0$ and

$$\mathrm{E}[T_{\mathrm{QSH}} \mid k_0] = \sum_{t=1}^{n-k_0} p_t^{-1/2}\,.$$

Recall that p_t is the probability that $\textsc{mut}(\mathbf{x}_{t-1}) \in \mathcal{S}_t$. This event happens if the i_{t-1}-th bit is flipped, that is, $p_t = 1/n$. Therefore,

$$\mathrm{E}[T_{\mathrm{QSH}} \mid k_0] = \sum_{t=1}^{n-k_0}\sqrt{n} = \sqrt{n}(n-k_0)$$

Since $\mathbf{x}_0$ is chosen uniformly at random from $\{0,1\}^n$ it holds by the Chernoff bounds [17] that $k_0 \leq 2n/3$ with a probability bounded away from zero by a positive constant. Hence, $\mathrm{E}[T_{\mathrm{QSH}}] = \Omega(n^{3/2})$. On the other hand, $k_0 \geq 0$, and therefore $\mathrm{E}[T_{\mathrm{QSH}}] = O(n^{3/2})$.

Now, let us turn to (1+1) QEA with i_t and k_t as for the QLS. For $t \geq 1$, the set $\mathcal{S}_t$ consists of all boolean vectors that are obtained from $\mathbf{x}_{t-1}$ if the i_{t-1}-th bit is flipped and none of the bits $1,\dots,i_{t-1}-1$ is flipped. The probability that $\textsc{mut}(\mathbf{x}_{t-1}) \in \mathcal{S}_t$ is then

$$p_t = \frac{1}{n}\Big(1-\frac{1}{n}\Big)^{i_{t-1}-1}$$

Thus, since $(1-1/n)^{n-1} \geq \mathrm{e}^{-1}$ and $0 \leq i_{t-1} \leq n$,

$$\frac{1}{\mathrm{e}\,n} \leq p_t \leq \frac{1}{n}\,.$$

Unlike for the QLS, we cannot guarantee $k_t = k_{t-1}+1$. However, we know that $i_t \geq i_{t-1}+1$. Thus, we have $T \leq n$ and

$$\mathrm{E}[T_{\mathrm{QSH}}] \leq \sum_{t=1}^{n}\sqrt{e\,n}\,,$$

that is, $\mathrm{E}[T_{\mathrm{QSH}}] \in O(n^{3/2})$.

For the corresponding lower bound, we first make the following observation. Let $t \geq 1$ and $i \geq i_{t-1} + 1$. Since the probability that $\mathbf{x}_0(i) = 1$ is $1/2$ and since the i-th bit had no influence on whether $\mathbf{x}_r \in \mathcal{S}_r$ for $s \leq t$, the probability that $\mathbf{x}_t(i) = 1$ is as well $1/2$ because of the symmetry of the mutation operator.

For $t \geq 1$, we know that $\mathbf{x}_t(i_{t-1}) = 0$ and $\mathbf{x}_t(i) = 1$ for $i \leq i_{t-1} - 1$. Moreover, we have just seen that $\mathbf{x}_t(i) = 1$ with probability $1/2$ for $i \geq i_{t-1} + 1$. Thus, for all $\ell \geq 1$ the probability that $i_t - i_{t-1} = \ell$ is $2^{-\ell}$ and therefore

$$\mathrm{E}[i_t - i_{t-1}] \leq \sum_{\ell \geq 1} \ell \, 2^{-\ell} = 2\,.$$

Since also $\mathrm{E}[i_0] \leq 2$ for the same reason, $\mathrm{E}[i_t] \leq 2(t+1)$ holds for all $t \geq 0$. Thus $\mathrm{E}[i_{\lfloor n/3 \rfloor}] \leq 2(n/3 + 1)$ and the probability that $i_{\lfloor n/3 \rfloor} \leq n$ is bounded away from zero by a constant $c > 0$. Thus, $\mathrm{Prob}[T \geq \lfloor n/3 \rfloor] \geq c$ and

$$\mathrm{E}[T_{\mathrm{QSH}}] \geq c \cdot \mathrm{E}[T_{\mathrm{QSH}} \mid T \geq \lfloor n/3 \rfloor] \geq c \, \lfloor n/3 \rfloor \sqrt{n}\,,$$

that is, $\mathrm{E}[T_{\mathrm{QSH}}] \in \Omega(n^{3/2})$. □

6. DISCREPANCY

The pseudo-Boolean function Disc denotes half the difference in the number of one-bits and zero-bits in a bitstring $\mathbf{x} \in \{0,1\}^n$ of even length n, that is, let

$$\textsc{Disc}(\mathbf{x}) := \left| \frac{n}{2} - \textsc{OneMax}(\mathbf{x}) \right|\,. \tag{6.1}$$

Lemma 15. *Let $n \in \mathbb{N}$ be even and let $\mathbf{x} \in \{0,1\}^n$ be chosen uniformly at random. Then,*

$$\mathrm{E}[\textsc{Disc}(\mathbf{x})] \in \Theta(n^{1/2})\,.$$

Proof. Let $n = 2k$. Then, the lemma follows from

$$\mathrm{E}[\textsc{Disc}(\mathbf{x})] = 2 \sum_{i=0}^{k} (k-i) \tbinom{2k}{i} 2^{-2k} = k \tbinom{2k}{k} 2^{-2k}$$

and from $\binom{2k}{k} \sim 2^{2k}/\sqrt{\pi\, k}$ due to Stirling's formula. □

For the function Disc, we show that the query-complexity of the RSHs and QSHs we consider is asymptotically equal.

Theorem 16. *Let $\{\mathbf{x}_t\}_{t \in \mathbb{N}}$ be the search points generated by the (1+1) EA, RLS, the (1+1) QEA, or QLS minimizing* Disc *with a random starting point $\mathbf{x}_0$. Then the expected query time is in $\Theta(\sqrt{n})$ in all four cases.*

Proof. Without loss of generality, we always assume that the search point $\mathbf{x}_t$ contains at least as many zeroes as ones, i.e. $\textsc{OneMax}(\mathbf{x}_t) \leq n/2$, for the other case follows by a symmetric argument.

For RLS and (1+1) EA we define the drift D_t as

$$D_t = E[\textsc{Disc}(\mathbf{x}_t) - \textsc{Disc}(\mathbf{x}_{t+1})].$$

We show that for all time steps t the drift satisfies the inequality

$$1 \leq D_t \leq 2\,, \tag{6.2}$$

i.e., both the search heuristics have a constant drift. By Theorem 1, the drifts for the quantum algorithms equal the drifts for the classical algorithms, respectively.

The lower bound is trivial since $\textsc{Disc}(\mathbf{x}_{t+1}) < \textsc{Disc}(\mathbf{x}_t)$. Furthermore, for the RLS the drift is always 1 since we flip only one bit in each step. So it remains to prove the upper bound for (1+1) EA.

Let $k_t := n - \textsc{OneMax}(\mathbf{x}_t)$ be the number of zero-bits in $\mathbf{x}_t$. Then by the triangle inequality, the drift is at most $\mathrm{E}[k_t - k_{t+1}]$. (Actually, it will be strictly smaller because it may happen that $k_{t+1} < n/2$.) The expected difference $\mathrm{E}[k_t - k_{t+1}]$ is upper bounded by the expected number of zero-bits that flip, and even more so by the expected number of bits that flip at all. Recall that k_{t+1} is obtained from k_t by flipping each bit with probability $1/n$, under the condition that at least one bit flips. Thus we may conclude

$$\begin{aligned} D_t &\leq \mathrm{E}[k_t - k_{t+1}] \\ &\leq \mathrm{E}[\#\text{ of bit flips from } \mathbf{x}_t \text{ to } \mathbf{x}_{t+1}] \\ &\leq 1 + \sum_i \mathrm{Prob}(i\text{-th bit flips}) \\ &= 2\,. \end{aligned}$$

Adding up D_s for all $0 \leq s \leq t$, equation (6.2) yields bounds for the total drift from $\mathbf{x}_0$ to $\mathbf{x}_t$, for $t \leq T$:

$$t \leq \mathrm{E}[\textsc{Disc}(\mathbf{x}_0)] - \mathrm{E}[\textsc{Disc}(\mathbf{x}_t) \mid T \geq t] \leq 2t.$$

In particular, doing the same process for $t = T$ and computing expected values, we get

$$\mathrm{E}[T] \leq \mathrm{E}[\textsc{Disc}(\mathbf{x}_0)] - \mathrm{E}[\textsc{Disc}(\mathbf{x}_T)] = \mathrm{E}[\textsc{Disc}(\mathbf{x}_0)] \leq 2\,\mathrm{E}[T]\,,$$

which in turn gives

$$\tfrac{1}{2}\,\mathrm{E}[\textsc{Disc}(\mathbf{x}_0)] \leq \mathrm{E}[T] \leq \mathrm{E}[\textsc{Disc}(\mathbf{x}_0)].$$

Now we show that uniformly for all $0 < t \leq T$, the transition probability p_t is bounded by constants. For all $t \geq 0$, let $d_t := \textsc{Disc}(\mathbf{x}_t)$.

For RLS, we have $p_t = 1/n \cdot (n/2 + d_t) \geq 1/2$ for all t, since the discrepancy decreases whenever a zero-bit is flipped. Hence, $1/2 \leq p_t \leq 1$.

For the (1+1) EA, $\mathcal{S}_t$ contains at least those boolean strings obtained from $\mathbf{x}_{t-1}$ by flipping a single zero-bit and no other bit. Since there are $k_{t-1} \geq n/2$ zero-bits in $\mathbf{x}_{t-1}$, we find a lower bound for p_t:

$$p_t \geq k_{t-1} \cdot \frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{2\mathrm{e}}\,.$$

Thus, we have proven $\frac{1}{2\mathrm{e}} \leq p_t \leq 1$ for all $0 < t \leq T$. Therefore,

$$\mathrm{E}[T_{\mathrm{RSH}}] = \mathrm{E}[\sum_{t=1}^{T} p_t^{-1}] = \Theta(\mathrm{E}[T]) = \Theta(\mathrm{E}[\textsc{Disc}(\mathbf{x}_0)]),$$

for both RLS and the (1+1) EA. Of course, since p_t is bounded by constants, $p_t^{1/2}$ is also bounded by constants, and we may also conclude that

$$\mathrm{E}[T_{\mathrm{QSH}}] = \mathrm{E}[\sum_{t=1}^{T} p_t^{-1/2}] = \Theta(\mathrm{E}[T]) = \Theta(\mathrm{E}[\textsc{Disc}(\mathbf{x}_0)]).$$

Finally, by Lemma 15 the expected discrepancy of X_0 is in $\Theta(n^{1/2})$, so the result follows. □

7. CONCLUSION

In this paper, we have presented an approach to evolutionary algorithms on a quantum computer. Essentially, we keep the mutation and selection process from the classical setting and use Grover search in order to find an improved offspring more quickly. Theorem 1 tells us that this does not affect the behavior of the algorithm except for getting faster. The approach is universal, i.e., it works for any mutation operator. Lemma 9 gives us an easy way to compute the expected running time of the quantum algorithms from non-quantum quantities.

Our method yields at most a quadratic speedup. This is similar to other general settings like unordered search [4, 22] or query complexity of local search on a graph [1], in which it is proven or conjectured that quantum computers can give at most a quadratic speedup. The analyzed examples OneMax, LeadingOnes, and Disc show that a substantial speedup is possible (as for LeadingOnes) but is not guaranteed (as for Disc). The harder it is to improve the objective function, the better will quantum acceleration work.

This has a surprising consequence. In order to be able to move on fitness plateaus, in the Step 2.b) of a RSH the condition "$f(\mathbf{y}_t) > f(\mathbf{x}_t)$" is often replaced by the condition "$f(\mathbf{y}_t) \geq f(\mathbf{x}_t)$ and $\mathbf{y}_t \neq \mathbf{x}_t$". We denote this variant by RSH*. In the examples we analyze, this does not make much difference for the classical case. However, for the quantum algorithms, the results differ substantially. Due to lack of space, we summarized the results in Table 1 and omit the proofs, which are very similar to those we carried out. The reason for the different results is that by allowing equality of the objective functions we increase the number of valid successor states and thus we increase the probability to find such a state. But quantum enhancement is more powerful if these probabilities are small. See Lemmas 7 and 9 for a precise statement. However, there are ways to keep quantum enhancement powerful and still allow the algorithm to move to a successor state with unchanged objective value. We hope to discuss these issues in further work.

We have chosen the problems OneMax, LeadingOnes, and Disc for two reasons. Firstly, because they demonstrate a wide range of effects that may occur. Secondly, because they are easy to analyze. However, these are of course only toy problems, and we would like to analyze real combinatorial problems. We aim to do so in subsequent work.

8. REFERENCES

[1] S. Aaronson. Lower bounds for local search by quantum arguments. *SIAM J. Comput.*, 35(4):804–824, 2006.

[2] A. Ambainis. Quantum lower bounds by quantum arguments. *J. Comput. Syst. Sci.*, 64(4):750–767, 2002.

[3] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001.

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

[5] A. Berzina, A. Dubrovsky, R. Freivalds, L. Lace, and O. Scegulnaja. Quantum query complexity for some graph problems. In *SOFSEM*, pages 140–150, 2004.

[6] H.-G. Beyer, H.-P. Schwefel, and I. Wegener. How to analyse evolutionary algorithms. *Theor. Comput. Sci.*, 287(1):101–130, 2002.

[7] M. Boyer, G. Brassard, P. Høyer, and A. Tapp. Tight bounds on quantum searching. *Fortschr. Physik*, 46(4-5):493–505, 1998.

[8] G. Brassard, P. Høyer, and A. Tapp. Quantum counting. In *ICALP*, pages 820–831, 1998.

[9] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theor. Comput. Sci.*, 276:51–81, 2002.

[10] C. Dürr, M. Heiligman, P. Høyer, and M. Mhalla. Quantum query complexity of some graph problems. In *ICALP*, pages 481–493, 2004.

[11] C. Dürr, M. Heiligman, P. Høyer, and M. Mhalla. Quantum query complexity of some graph problems. *SIAM J. Comput.*, 35(6):1310–1328, 2006.

[12] C. Dürr and P. Høyer. A quantum algorithm for finding the minimum. *CoRR*, quant-ph/9607014, 1996.

[13] L. K. Grover. A fast quantum mechanical algorithm for database search. In *STOC*, pages 212–219, 1996.

[14] K.-H. Han and J.-H. Kim. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. *IEEE Trans. Evolutionary Computation*, 6(6):580–593, 2002.

[15] F. Magniez, A. Nayak, P. C. Richter, and M. Santha. On the hitting times of quantum versus random walks. In *SODA*, pages 86–95, 2009.

[16] F. Magniez, A. Nayak, J. Roland, and M. Santha. Search via quantum walk. In *STOC*, pages 575–584, 2007.

[17] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York, NY, USA, 2005.

[18] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, United Kingdom, 2000.

[19] M. Santha. Quantum walk based search algorithms. In *TAMC*, pages 31–46, 2008.

[20] L. Spector, H. Barnum, H. Bernstein, and N. Swamy. Finding a better-than-classical quantum and/or algorithm using genetic programming. In *Evolutionary Computation*, volume 3, pages 2239–2246, 1999.

[21] M. Szegedy. Quantum speed-up of markov chain based algorithms. In *FOCS*, pages 32–41, 2004.

[22] C. Zalka. Grover's quantum searching algorithm is optimal. *Phys. Rev. A*, 60(4):2746–2751, Oct 1999.

[23] S. Zhang. *New quantum algorithms and quantum lower bounds*. PhD thesis, Princeton, NJ, USA, 2006.