Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 41 (2005)

New upper and lower bounds for randomized and quantum Local Search*

Shengyu Zhang †

Abstract

The Local Search problem, which finds a local minimum of a black-box function on a given graph, is of both practical and theoretical importance to many areas in combinatorial optimization, complexity theory and quantum computing. In this paper, we give both new lower and upper bound techniques for randomized and quantum query complexities of Local Search. The lower bound technique works for product graphs. Applying the technique to the Boolean hypercube $\{0,1\}^n$ and the constant dimensional grids $[n]^d$, two particular types of product graphs that recently draw much attention, we get the following tight results.

$$RLS(\{0,1\}^n) = \Theta(2^{n/2}n^{1/2}), \qquad QLS(\{0,1\}^n) = \Theta(2^{n/3}n^{1/6})$$
$$RLS([n]^d) = \Theta(n^{d/2}) \text{ for } d \geq 4, \quad QLS([n]^d) = \Theta(n^{d/3}) \text{ for } d \geq 6.$$

where $RLS(G)$ and $QLS(G)$ are the randomized and quantum query complexities of Local Search on G, respectively. These improve the previous results by Aaronson [2], and Santha and Szegedy[21].

Our new Local Search algorithms work well when the underlining graph expands slowly. As an application to the 2-dimensional grid $[n]^2$, a new quantum algorithm using $O(\sqrt{n}(\log\log n)^{1.5})$ time and queries is given. This improves the previous best known upper bound of $O(n^{2/3})$ [2], and implies that Local Search on grids exhibits different properties at low dimensions.

*This research was supported in part by NSF grants CCR-0310466 and CCF-0426582.

†Computer Science Department, Princeton University, NJ 08544, USA. Email: szhang@cs.princeton.edu

ISSN 1433-8092

1 Introduction

Many important combinatorial optimization problems arising in both theory and practice are **NP**-hard, which forces people to resort to heuristic searches in practice. One popular approach is local search, in which one first defines a *neighborhood structure*, then finds a solution that is locally optimal with respect to this neighborhood structure. In the past two decades, the local search approach has been extensively developed and "has reinforced its position as a standard approach in combinatorial optimization" in practice [1]. Besides the practical applications, local search also has many connections to the complexity theory, especially to the complexity classes **PLS** [1] and **TFNP** [2]. For example, the 2SAT-FLIP problem, an important problem known to be complete in **PLS**, is actually the local search problem with the neighborhood structure being the Boolean hypercube $\{0,1\}^n$ and the objective function being the sum of the weights of the clauses that the truth assignment $x \in \{0,1\}^n$ satisfies. Local search is also related to physical systems including folding proteins and to the quantum adiabatic algorithms [2]. We refer readers to the papers [2, 20, 21] for more discussions and the book [3] for a comprehensive introduction.

Precisely, the Local Search problem on an undirected graph $G = (V, E)$ is defined as follows. Given a function $f : V \to \mathbb{N}$, find a vertex $v \in V$ such that $f(v) \leq f(w)$ for all neighbors w of v. A class of *generic algorithms* that has been widely used in practice is as follows: first set out with an initial point $v \in V$, then repeatedly search a better/best neighbor until it reaches a local minimum. Though empirically this class of algorithms work very well in most applications, relatively few theoretical results are known about how good the generic algorithms are, especially for the randomized (and quantum) algorithms.

Among models for the theoretical studies, the query model has drawn much attention [2, 4, 5, 17, 18, 21]. In this model, $f(v)$ can only be accessed by querying v, and the randomized (and quantum) query complexity, denote by $RLS(G)$ (and $QLS(G)$) is the minimum number of queries needed by a randomized (and quantum) algorithm that solves the problem. Previous upper bounds on a general N-vertex graph G are $RLS(G) = O(\sqrt{N\delta})$ by Aldous [4] and $QLS(G) = O(N^{1/3}\delta^{1/6})$ by Aaronson [2], where δ is the maximum degree of G. Both algorithms are actually the generic algorithms mentioned above, with the initial point picked as the one having the minimum f-value over some random samples. For lower bounds, Aaronson [2] considered two special classes of graphs: the Boolean hypercube $\{0,1\}^n$ and the constant dimensional grid $[N^{1/d}]^d$. He showed that for $\{0,1\}^n$, $RLS(\{0,1\}^n) = \Omega(2^{n/2}/n^2)$ and $QLS(\{0,1\}^n) = \Omega(2^{n/4}/n)$, and that for $[N^{1/d}]^d$, $RLS([N^{1/d}]^d) = \Omega(N^{1/2-1/d}/\log N)$ and $QLS([N^{1/d}]^d) = \Omega(N^{1/4-1/(2d)}/\sqrt{\log N})$. It has also been shown that $QLS([N^{1/2}]^2) = \Omega(N^{1/8})$ by Santha and Szegedy [21]. However, the final values of QLS and RLS on both types of special graphs remain an open problem, explicitly stated in an earlier version of [2] and also (partially) in [21].

In this paper, we give both new lower and upper bound techniques for large classes of graphs which contain $\{0,1\}^n$ and $[N^{1/d}]^d$ as special cases. As a consequence, we completely solve the randomized and quantum query complexities of Local Search on $\{0,1\}^n$ and on $[N^{1/d}]^d$, except for a few small d's in which cases our bounds also significantly improve the previous ones.

Our lower bound technique works for the general product graphs. For two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, their product $G_1 \times G_2$ is a graph $G = (V, E)$ *s.t.* $V = V_1 \times V_2$ and $E = \{((v_1, v_2), (v_1', v_2)) : (v_1, v_1') \in E_1, v_2 \in V_2\} \cup \{((v_1, v_2), (v_1, v_2')) : (v_2, v_2') \in E_2, v_1 \in V_1\}$. We will also use the notion of random walk on graphs to state the theorem. Given a graph $G = (V, E)$, a random walk $W : V \to 2^V$ is *regular* if for each $u \in V$ it holds that $W(u) \subseteq \{u\} \cup \{v : (u, v) \in E\}$ and that $|W(u)| = c$ for some c independent of u. Intuitively, the random walk W proceeds as follows. It starts at some vertex u, and at each step it goes from the current vertex v to a uniformly random vertex in $W(v)$. Denote by $p(u, v, t)$ the probability that the random walk starting at u stops at v after exactly t steps. Let $p_t = \max_{u,v} p(u, v, t)$. The following theorem is a special case of the general Theorem 9 in Section 3.

[1] Polynomial Local Search, introduced by Johnson, Papadimitriou, and Yannakakis [15].

[2] The family of total function problems, introduced by Megiddo and Papadimitriou [19].

Theorem 1 *Suppose $G = G^w \times G^c$ is a product graph, and L is the length of the longest self-avoiding path in G^c. Let $T = \lfloor L/2 \rfloor$, then for any regular random walk W on G^w, we have*

$$RLS(G) = \Omega\left(\frac{T}{\sum_{t=1}^{T} p_t}\right), \qquad QLS(G) = \Omega\left(\frac{T}{\sum_{t=1}^{T} \sqrt{p_t}}\right). \tag{1}$$

The proof uses the quantum adversary method, which was originally proposed by Ambainis [7] and later generalized in different ways [6, 8, 16, 24]. Recently Spalek and Szegedy made the picture clear by showing that all these generalizations are equivalent in power [22]. On the other hand, in proving a particular problem, some of the methods might be easier to use than the others. In our case, the technique proposed by Zhang [24] works very well. Our proofs for the randomized lower bounds will use relational adversary method, proposed by Aaronson [2] inspired by the quantum adversary method.

Both the quantum adversary method and the relational adversary method are parameterized by input sets and weight functions of input pairs. While previous works [2, 21] also use random walks on graphs, a key innovation that distinguishes our work from the previous ones is that we decompose the graph into two parts, the tensor product of which is the original graph. We perform the random walk only in one part, and perform a simple one-way walk in a self-avoiding path in the other part, which serves as a "clock" to record the number of steps taken by the random walk in the first part. The tensor product of these two walks is a random path in the original graph. A big advantage of adding a clock is that the "passing probability", the probability that the random path *passes* a vertex v *within* T steps, is now the "stopping probability", the probability that the random walk in the first part *stops* at v *after* exactly t steps, which is well understood in the classical random walk literature. Another advantage is that since the walk in the second part is self-avoiding, the resulting random path in the original graph is self-avoiding too, which makes the analysis much easier.

Applying it to the two graphs $\{0,1\}^n$ and $[N^{1/d}]^d$, we improve previous results and show tight bounds on both RLS and QLS (in almost all cases).

Theorem 2 $RLS(\{0,1\}^n) = \Theta(2^{n/2}n^{1/2}), \quad QLS(\{0,1\}^n) = \Theta(2^{n/3}n^{1/6}).$

Theorem 3

$$RLS([N^{1/d}]^d) = \begin{cases} \Theta(N^{1/2}) & \text{if } d \geq 4 \\ \Omega((N/\log N)^{1/2}) & \text{if } d = 3 \\ \Omega\left(N^{1/3}\right) & \text{if } d = 2 \end{cases}, \quad QLS([N^{1/d}]^d) = \begin{cases} \Theta(N^{1/3}) & \text{if } d \geq 6 \\ \Omega((N/\log N)^{1/3}) & \text{if } d = 5 \\ \Omega\left(N^{1/2-1/(d+1)}\right) & \text{if } 2 \leq d \leq 4 \end{cases}$$

It is worth to note that to apply Theorem 1, we need not only know the mixing time of the random walks, but also know their behavior before mixing. So the applications are not simply using classical upper bounds for mixing time, but involve heavy analysis on the whole mixing process.

In the second part of the paper, we consider upper bounds for local search. While the generic algorithms [2, 4] are simple and proven to be optimal for many graphs such as above mentioned ones, they are far from optimal for some other graphs. For example, it is not hard to see an $O(\log N)$ *deterministic* algorithm if G is a line graph. So a natural question is to characterize those graphs on which the local search is easy. It turns out that the expansion speed is closely related the local search. For a graph $G = (V, E)$, the distance $|u-v|$ between two vertices u and v is the length of the shortest path connecting them. Let $c(k) = \max_{v \in V} |\{u : |u - v| \leq k\}|$. Apparently, the smaller $c(k)$ is, the more slowly the graph expands. As a special case of Theorem 12 in Section 5, the following upper bounds in terms of $c(k)$ hold.

Theorem 4 *If $c(k) = O(k^\alpha)$ for some constant $\alpha \geq 1$, then*

$$RLS(G) = \begin{cases} O\left(d^{\alpha-1}\log\log d\right) & if\ \alpha > 1 \\ O(\log d \log\log d) & if\ \alpha = 1 \end{cases}, \quad QLS(G) = \begin{cases} O\left(d^{\frac{\alpha-1}{2}}(\log\log d)^{1.5}\right) & if\ \alpha > 1 \\ O(\log d \log\log d) & if\ \alpha = 1 \end{cases} \tag{2}$$

This explains why Local Search on the line graph is easy. Also, it immediately gives a new upper bound for $QLS([N^{1/2}]^2)$ as follows. Together with Theorem 3, this implies that Local Search on grids exhibits different properties at low dimensions because low dimensional grids expand slowly.

Theorem 5 $QLS([N^{1/2}]^2) = O(N^{1/4}(\log\log N)^{1.5})$

Other related results. There are two unpublished results about $RLS([N^{1/2}]^2)$ and $QLS(\{0,1\}^n)$. It is mentioned in [2] that Ambainis showed $QLS(\{0,1\}^n) = \Omega(2^{n/3}/n^{O(1)})$, and it is mentioned in [21] that Verhoeven showed $RLS([N^{1/2}]^2) = \Omega(N^{1/2-\delta})$ for any constant $\delta > 0$. Verhoeven later also showed upper bounds in terms of the genus of the graph [23].

2 Preliminaries and notations

We use $[M]$ to denote the set $\{1, 2, ..., M\}$. For an n-bit binary string $x = x_0...x_{n-1} \in \{0,1\}^n$, let $x^{(i)} = x_0...x_{i-1}(1-x_i)x_{i+1}...x_{n-1}$ be the string obtained by flipping the coordinate i.

A path X in a graph $G = (V, E)$ is a sequence $(v_1, ..., v_l)$ of vertices such that for any pair (v_i, v_{i+1}) of vertices, either $v_i = v_{i+1}$ or $(v_i, v_{i+1}) \in E$. We use $set(X)$ to denote the set of distinct vertices on path X. A path is self-avoiding if v_1, ..., v_l are all distinct. The length of a path $(v_1, ..., v_l)$ is $l-1$. For two vertices $u, v \in V$, the distance $|u-v|_G$ is the length of the shortest path from u to v. The subscript G may be omitted if no confusion is caused.

The (k,l)-hypercube $G_{k,l} = (V, E)$ where $V = [k]^l$ and whose edge set is $E = \{(u, v) : \exists i \in \{0, ..., l-1\},\ s.t.\ |u_i - v_i| = 1,\ \text{and}\ u_j = v_j,\ \forall j \neq i\}$. Sometimes we abuse the notation by using $[k]^l$ to denote $G_{k,l}$. Note that both the Boolean hypercube and the constant dimension grid are special hypercubes.[3]

In an N-vertex graph $G = (V, E)$, a Hamilton path is a path $X = (v_1, ..., v_N)$ such that $(v_i, v_{i+1}) \in E$ for any $i \in [N-1]$ and $set(X) = V$. It is easy to check by induction that every hypercube $[k]^l$ has a Hamilton path. Actually, for $l = 1$, $[k]$ has a Hamilton path $(1, ..., k)$. Now suppose $[k]^l$ has a Hamilton path P, then a Hamilton path for $[k]^{l+1}$ can be constructed as follows, first fix the last coordinate to be 1 and go through P, then change the last coordinate to be 2 and go through P in the reverse order, and then change the last coordinate to be 3 and go through P, and so on. For each (k,l), let $HamPath_{k,l} = (v_1, ..., v_N)$ be the Hamilton path constructed as above, and we define the successor function $H_{k,l}(v_i) = v_{i+1}$ for $i \in [N-1]$.

We use $R_2(f)$ and $Q_2(f)$ to denote the double-sided error random and quantum query complexities of function f. For more details on query models and query complexities, we refer to [10] as an excellent survey.

2.1 One quantum adversary method and the relational adversary method

The quantum adversary method is one of the two powerful tools to prove lower bounds on quantum query complexity. See [14] for an comprehensive survey. In this paper, we will use the quantum adversary method proposed in [24]. The definition and theorem given here are a little more general than the original ones, but the proof remains unchanged.

Definition 1 *Let $F : I^N \to [M]$ be an N-variate function. Let $R \subseteq I^N \times I^N$ be a relation such that $F(x) \neq F(y)$ for any $(x, y) \in R$. A weight scheme consists of three weight functions $w(x, y) > 0$, $u(x, y, i) > 0$ and $v(x, y, i) > 0$ satisfying $u(x, y, i)v(x, y, i) \geq w^2(x, y)$ for all $(x, y) \in R$ and $i \in [N]$ with $x_i \neq y_i$. We further put*

$$w_x = \sum_{y':(x,y')\in R} w(x, y'), \qquad w_y = \sum_{x':(x',y)\in R} w(x', y) \tag{3}$$

$$u_{x,i} = \sum_{y':(x,y')\in R, x_i \neq y'_i} u(x, y', i), \qquad v_{y,i} = \sum_{x':(x',y)\in R, x'_i \neq y_i} v(x', y, i). \tag{4}$$

Theorem 6 [Zhang, [24]] *For any F, R and any weight scheme w, u, v as in Definition 1, we have*

$$Q_2(F) = \Omega\left(\min_{(x,y)\in R, i\in[N]:\ x_i\neq y_i} \sqrt{\frac{w_x w_y}{u_{x,i} v_{y,i}}}\right) \tag{5}$$

[3]Here we identify the Boolean hypercube $\{0,1\}^n$ and $G_{2,n}$ since they are isomorphic.

In [2], Aaronson gives a nice technique to get a lower bound for randomized query complexity. We restate it using a similar language of Theorem 6.

Theorem 7 [Aaronson, [2]] *Let $F : I^N \to [M]$ be an N-variate function. Let $R \subseteq I^N \times I^N$ be a relation such that $F(x) \neq F(y)$ for any $(x, y) \in R$. For any weight function $w : R \to \mathbb{R}^+$, we have*

$$R_2(F) = \Omega\left(\min_{(x,y)\in R, i\in[N], x_i\neq y_i} \max\left\{\frac{w_x}{w_{x,i}}, \frac{w_y}{w_{y,i}}\right\}\right) \tag{6}$$

where

$$w_{x,i} = \sum_{y':(x,y')\in R, x_i\neq y'_i} w(x, y'), \qquad w_{y,i} = \sum_{x':(x',y)\in R, x'_i\neq y_i} w(x', y). \tag{7}$$

Note that we can think of Theorem 7 as having a weight scheme too, which requires that $u(x, y, i) = v(x, y, i) = w(x, y)$. This simple observation is used in the proof of Theorem 2 and 3.

3 Lower bounds for Local Search on product graphs

In this section we prove a theorem stronger than Theorem 1 by relaxing the conditions of the random walk. Suppose we are given a graph $G = (V, E)$, a starting vertex v_0 and an assignment $W : V \times \mathbb{N} \to 2^V$ *s.t.* for each $u \in V$ and $t \in \mathbb{N}$, it holds that $W(u, t) \subseteq \{u\} \cup \{v : (u, v) \in E\}$ and that $|W(u, t)| = c_t$ for some function c of t. The random walk (G, v_0, W) on graph G proceeds as follows. It starts at v_0, and at step $t \in \mathbb{N}$, it goes from the current vertex v_{t-1} to a uniformly random vertex in $W(v_{t-1}, t)$. We say a path $(v_0, v_1, ..., v_T)$ is generated by the random walk if $v_t \in W(v_{t-1}, t)$ for all $t \in [T]$. Denote by $p(u, t_1, v, t_2)$ the probability that the random walk is at v after step t_2 under the condition that the walk is at u after step t_1. Let $p_t = \max_{u,v,t_1,t_2:\ t_2-t_1=t} p(u, t_1, v, t_2)$. For $(u, u') \in E$, let $q(u, u', t_1, v, t_2)$ be the probability that the walk is at v after step t_2, under the condition that the walk is at u after step t_1 and the walk does not go to u' at step $t_1 + 1$. The following lemma on the relation of the two probabilities is obvious.

Lemma 8 *If $|W(u, t_1 + 1)| > 1$, then $q(u, u', t_1, v, t_2) \leq 2p(u, t_1, v, t_2)$.*

Proof By considering the two cases of the step $t_1 + 1$ (going to u' or not), we have

$$p(u, t_1, v, t_2) = \frac{1}{|W(u, t_1 + 1)|} p(u', t_1 + 1, v, t_2) + \left(1 - \frac{1}{|W(u, t_1 + 1)|}\right) q(u, u', t_1, v, t_2). \tag{8}$$

Thus

$$q(u, u', t_1, v, t_2) \leq p(u, t_1, v, t_2) / \left(1 - \frac{1}{|W(u, t_1 + 1)|}\right) \leq 2p(u, t_1, v, t_2). \tag{9}$$

□

Theorem 9 *Suppose $G = G^w \times G^c$ is a product graph, and L is the length of the longest self-avoiding path in G^c. Let $T = \lfloor L/2 \rfloor$, then for the random walk (G^w, v_0^w, W) on G^w, we have*

$$RLS(G) = \Omega\left(\frac{T}{\sum_{t=1}^{T} p_t}\right), \qquad QLS(G) = \Omega\left(\frac{T}{\sum_{t=1}^{T} \sqrt{p_t}}\right). \tag{10}$$

Proof We shall construct a random walk on G by the random walk (G^w, v_0^w, W) on G^w and a simple one-way walk on G^c. Starting from a fixed vertex in G, the walk is proceeded by one step of walk in G^w followed by two steps of walk in G^c. Precisely, fix a self-avoiding path $(z^c_{0,0}, z^c_{1,0}, z^c_{1,1}, z^c_{2,1}, z^c_{2,2}, ..., z^c_{T,T-1}, z^c_{T,T})$ of length $2T$ in G^c. Let the set P contain all the paths $X = (x^w_0 \otimes z^c_{0,0}, x^w_1 \otimes z^c_{0,0}, x^w_1 \otimes z^c_{1,0}, x^w_1 \otimes z^c_{1,1}, ..., x^w_T \otimes z^c_{T-1,T-1}, x^w_T \otimes z^c_{T,T-1}, x^w_T \otimes z^c_{T,T})$ in G such that $x^w_0 = v^w_0$ and $(x^w_0, x^w_1, ..., x^w_T)$ is a path generated by the random walk (G^w, v^w_0, W). Define a problem $\textsc{Path}_P$: given a path $X \in P$, find the end point $x^w_T \otimes z^c_{T,T}$. We are allowed to access X by

querying an oracle O whether a point $x \in set(X)$ and getting the Yes/No answer.[4] The following claim says that the $\textsc{Path}_P$ problem is not much harder than the Local Search problem.

Claim 1 $R_2(\textsc{Path}_P) \leq 2RLS(G), \quad Q_2(\textsc{Path}_P) \leq 2QLS(G)$.

Proof Suppose we have an Q-query randomized or quantum algorithm $\mathcal{A}$ for Local Search, we shall give a $2Q$ corresponding algorithm $\mathcal{B}$ for $\textsc{Path}_P$. For any path $X \in P$, we define a function f_X essentially in the same way as Aaronson did in [2]: for each vertex $v \in G$, let

$$f_X(v) = \begin{cases} |v - x_0^w \otimes z_{0,0}^c|_G + 3T & \text{if } v \notin set(X) \\ 3(T-k) & \text{if } v = x_k^w \otimes z_{k,k}^c \\ 3(T-k)-1 & \text{if } v = x_{k+1}^w \otimes z_{k,k}^c \neq x_k^w \otimes z_{k,k}^c \\ 3(T-k)-2 & \text{if } v = x_{k+1}^w \otimes z_{k+1,k}^c \end{cases} \tag{11}$$

It is easy to check that the only local minimum is $x_T^w \otimes z_{T,T}^c$.

Given an oracle O and an input X of the $\textsc{Path}$ problem, $\mathcal{B}$ simulates $\mathcal{A}$ to find the local minimum of f_X, which is also the end point of X. Whenever $\mathcal{A}$ needs to make a query on v to get $f_X(v)$, $\mathcal{B}$ asks O whether $v \in set(X)$. If $v \notin set(X)$, then $f_X(v) = |v - x_0^w \otimes z_{0,0}^c|_G + 3T$; otherwise, $v = x^w \otimes z_{k+1,k}^c$ or $v = x^w \otimes z_{k,k}^c$ for some $x^w \in V^w$ and k. Note that k is known given the vertex v. So if $v = x^w \otimes z_{k+1,k}^c$, then $x^w = x_{k+1}^w$ and thus $f_X(v) = 3(T-k)-2$. Now consider the case that $v = x^w \otimes z_{k,k}^c$. If $k = 0$, then let $f_X(v) = 3T$ if $v = x_0^w \otimes z_{0,0}^c$ and $f_X(v) = 3T-1$ otherwise. If $k \geq 1$, then $\mathcal{B}$ asks O whether $x^w \otimes z_{k,k-1}^c \in set(X)$. If yes, then $v = x_k^w \otimes z_{k,k}^c$ and thus $f_X(v) = 3(T-k)$; if no, then $v = x_{k+1}^w \otimes z_{k,k}^c \neq x_k^w \otimes z_{k,k}^c$ and thus $f_X(v) = 3(T-k)-1$. Therefore, at most 2 queries on O can simulate one query on f_X, so we have a $2Q$ algorithm for $\textsc{Path}_P$. □

(Continue the proof of Theorem 9) By the claim, it is sufficient to prove lower bounds for $\textsc{Path}_P$. We define a relation R_P as follows.

$$R_P = \{(X,Y) : X \in P,\ Y \in P,\ X \text{ and } Y \text{ has different end points}\} \tag{12}$$

For any pair $(X,Y) \in R_P$, where $X = (x_0^w \otimes z_{0,0}^c, x_1^w \otimes z_{0,0}^c, x_1^w \otimes z_{1,0}^c, x_1^w \otimes z_{1,1}^c, ..., x_T^w \otimes z_{T-1,T-1}^c, x_T^w \otimes z_{T,T-1}^c, x_T^w \otimes z_{T,T}^c)$ and $Y = (y_0^w \otimes z_{0,0}^c, y_1^w \otimes z_{0,0}^c, y_1^w \otimes z_{1,0}^c, y_1^w \otimes z_{1,1}^c, ..., y_T^w \otimes z_{T-1,T-1}^c, y_T^w \otimes z_{T,T-1}^c, y_T^w \otimes z_{T,T}^c)$, we write $X \wedge Y = k$ if $x_0^w = y_0^w$, ..., $x_{k-1}^w = y_{k-1}^w$ but $x_k^w \neq y_k^w$. Note that if $X \wedge Y = k$, then $x_k^w, y_k^w \in W(x_{k-1}^w, k)$ and thus $|W(x_{k-1}^w, k)| \geq 2$. By Lemma 8, this implies that $q(x_{k-1}^w, x_k^w, k-1, v^w, j) \leq 2p_{j-k+1}$. We then choose the weight functions in Theorem 6. Let

$$w(X,Y) = 1/|\{Y' \in P : Y' \wedge X = k\}| = 1/|\{X' \in P : X' \wedge Y = k\}|. \tag{13}$$

Note that it is well-defined because $|\{Y' \in P : Y' \wedge X = k\}| = |\{X' \in P : X' \wedge Y = k\}| = (c_k - 1)c_{k+1}...c_T$. To calculate $w_X = \sum_{Y':(X,Y') \in R_P} w(X,Y')$, we group those Y' that diverge from X at the same place. Then

$$w_X = \sum_{k=1}^{T} \sum_{\substack{Y':(X,Y') \in R_P \\ X \wedge Y' = k}} w(X,Y') = \sum_{k=1}^{T} \sum_{\substack{Y':(X,Y') \in R_P \\ X \wedge Y' = k}} \frac{1}{|\{Y' \in P : Y' \wedge X = k\}|} \tag{14}$$

Note that $\sum_{Y':\ (X,Y') \in R_P,\ X \wedge Y' = k} 1/|\{Y' \in P : Y' \wedge X = k\}| = |\{Y' : (X,Y') \in R_P,\ X \wedge Y' = k\}|/|\{Y' \in P : Y' \wedge X = k\}| = \mathbf{Pr}_{Y'}[(X,Y') \in R_P | X \wedge Y' = k] = \mathbf{Pr}_{Y'}[(y')_T^w \neq x_T^w | Y' \wedge X = k]$ because all paths generated by the random walk W have the same probability $1/(c_1...c_T)$. Also note that $\mathbf{Pr}_{Y'}[(y')_T^w \neq x_T^w | Y' \wedge X = k]$ is nothing but $1 - q(x_{k-1}^w, x_k^w, k-1, x_T^w, T)$, which is at

[4] Note that it is actually an oracle for the following Boolean function $g : \{0,1\}^n \to \{0,1\}$, with $g(x) = 1$ if and only if $x \in set(X)$. So strictly speaking, an input of $\textsc{Path}_P$ should be specified as $set(X)$ rather than X, because in general, it is possible that $X \neq Y$ but $set(X) = set(Y)$. For our problem, however, it is easy to check that for any $X, Y \in P$, it holds that $X = Y \Leftrightarrow set(X) = set(Y)$. Actually, if $X \neq Y$, suppose the first diverging place is k, *i.e.* $x_{k-1}^w = y_{k-1}^w$, but $x_k^w \neq y_k^w$. Then Y will never pass $x_k^w \otimes z_{k,k-1}^c$ because the clock immediately ticks and the time always advances forward. (Or more rigorously, the only point that Y passes through $z_{k,k-1}^c$ is $y_k^w \otimes z_{k,k-1}^c$. Since $y_k^w \neq x_k^w$, $x_k^w \otimes z_{k,k-1}^c \notin set(Y)$.)

least $1-2p_{T-k+1}$. So we have $w_X \geq T - 2\sum_{k=1}^{T} p_{T-k+1} = T - 2\sum_{t=1}^{T} p_t$. Similarly, we have $w_Y \geq T - 2\sum_{t=1}^{T} p_t$ too.

Now we define $u(X,Y,i)$ and $v(X,Y,i)$, where i is a point $x^w_{j+r} \otimes z^c_{j+s,j} \in set(X) - set(Y)$ or $y^w_{j+r} \otimes z^c_{j+s,j} \in set(Y) - set(X)$. Here $(r,s) \in \{(0,0),(1,0),(1,1)\}$, and $0 \leq j \leq j+r \leq T$. Let

$$u(X,Y,x^w_{j+r} \otimes z^c_{j+s,j}) = a_{k,j,r,s} w(X,Y), \quad u(X,Y,y^w_{j+r} \otimes z^c_{j+s,j}) = b_{k,j,r,s} w(X,Y), \tag{15}$$

$$v(X,Y,x^w_{j+r} \otimes z^c_{j+s,j}) = b_{k,j,r,s} w(X,Y), \quad v(X,Y,y^w_{j+r} \otimes z^c_{j+s,j}) = a_{k,j,r,s} w(X,Y). \tag{16}$$

where $a_{k,j,r,s}$ and $b_{k,j,r,s}$ will be given later (satisfying $a_{k,j,r,s} b_{k,j,r,s} = 1$). We now calculate $u_{X,i}$ and $v_{Y,i}$ for $i = x^w_{j+r} \otimes z^c_{j+s,j} \in set(X) - set(Y)$; the other case $i = y^w_{j+r} \otimes z^c_{j+s,j}$ is just symmetric. Note that since $x^w_{j+r} \otimes z^c_{j+s,j} \in set(X) - set(Y)$, we have $k \leq j+r$.

$$u_{X,x^w_{j+r} \otimes z^c_{j+s,j}} = \sum_{k=1}^{j+r} \sum_{\substack{Y':(X,Y') \in R_P,\ X \wedge Y' = k \\ x^w_j \otimes z^c_{j+b} \notin set(Y')}} a_{k,j,r,s} w(X,Y') \tag{17}$$

$$\leq \sum_{k=1}^{j+r} \sum_{Y': X \wedge Y' = k} a_{k,j,r,s} w(X,Y') = \sum_{k=1}^{j+r} a_{k,j,r,s} \tag{18}$$

The computation for $v_{Y,x^w_{j+r} \otimes z^c_{j+s,j}}$ is a little more complicated. By definition,

$$v_{Y,x^w_{j+r} \otimes z^c_{j+s,j}} = \sum_{k=1}^{j+r} \sum_{\substack{X':(X',Y) \in R_P,\ X' \wedge Y = k, \\ x^w_{j+r} \otimes z^c_{j+s,j} \in set(X')}} b_{k,j,r,s} w(X',Y) \tag{19}$$

$$\leq \sum_{k=1}^{j+r} \sum_{\substack{X': X' \wedge Y = k, \\ x^w_{j+r} \otimes z^c_{j+s,j} \in set(X')}} b_{k,j,r,s} w(X',Y) \tag{20}$$

$$= \sum_{k=1}^{j+r} b_{k,j,r,s} \mathbf{Pr}_{X'}[x^w_{j+r} \otimes z^c_{j+s,j} \in set(X') | X' \wedge Y = k] \tag{21}$$

We can see that $\mathbf{Pr}_{X'}[x^w_{j+r} \otimes z^c_{j+s,j} \in set(X') | X' \wedge Y = k]$ is roughly $q(y^w_{k-1}, y^w_k, k-1, x^w_{j+r}, j) + q(y^w_{k-1}, y^w_k, k-1, x^w_{j+r}, j+1)$ except for some corner cases. To be more precise, define $Bound_{k,j,r,s} = 2p_{j-k+2} \cdot \lambda[s = 1 \vee j < T] + 2p_{j-k+1} \cdot \lambda[s = 0 \wedge (k \leq j \vee r = 0)]$, where the Boolean function $\lambda[\phi] = 1$ if ϕ is true and 0 otherwise. Then

Claim 2 $\mathbf{Pr}_{X'}[x^w_{j+r} \otimes z^c_{j+s,j} \in set(X') | X' \wedge Y = k] \leq Bound_{k,j,r,s}$.

Proof We study $\mathbf{Pr}_{X'}[x^w_{j+r} \otimes z^c_{j+s,j} \in set(X') | X' \wedge Y = k]$ case by case. If $s = 1$, then $r = 1$ and $x^w_{j+1} \otimes z^c_{j+1,j} \in set(X')$ if and only if $x^w_{j+1} = (x')^w_{j+1}$. So $\mathbf{Pr}_{X'}[x^w_{j+r} \otimes z^c_{j+s,j} \in set(X') | X' \wedge Y = k] = q(y^w_{k-1}, y^w_k, k-1, x^w_{j+1}, j+1) \leq 2p_{j-k+2}$ by Lemma 8. If $s = 0$, then $x^w_{j+r} \otimes z^c_{j,j} \in set(X')$ if and only if "$x^w_{j+r} = (x')^w_j$ or $x^w_{j+r} = (x')^w_{j+1}$". Also note that $\mathbf{Pr}_{X'}[x^w_{j+r} = (x')^w_j | X' \wedge Y = k] = q(y^w_{k-1}, y^w_k, k-1, x^w_{j+r}, j)$ unless $k = j+1$ and $r = 1$, in which case $\mathbf{Pr}_{X'}[x^w_{j+r} = (x')^w_j | X' \wedge Y = k] = 0$ because $x^w_{j+1} \otimes z^c_{j,j} \notin set(Y)$ but $(x')^w_j \otimes z^c_{j,j} = y^w_j \otimes z^c_{j,j} \in set(Y)$. The other probability $\mathbf{Pr}_{X'}[x^w_{j+r} = (x')^w_{j+1} | X' \wedge Y = k]$ is just $q(y^w_{k-1}, y^w_k, k-1, x^w_{j+r}, j+1)$ if $j \leq T-1$ and it is 0 if $j = T$. Putting all cases together, we know that $\mathbf{Pr}_{Y'}[y^w_{j+r} \otimes z^c_{j+s,j} \in set(Y') | X \wedge Y' = k] \leq Bound_{k,j,r,s} = 2p_{j-k+2} \cdot \lambda[s = 1 \vee j < T] + 2p_{j-k+1} \cdot \lambda[s = 0 \wedge (k \leq j \vee r = 0)]$. □

(Continue the proof of Theorem 9) The claim implies that $v_{Y,x^w_{j+r} \otimes z^c_{j+s,j}} \leq \sum_{k=1}^{j+r} b_{k,j,r,s} Bound_{k,j,r,s}$. The symmetric case of $u(X,Y,i)$ and $v(X,Y,i)$ where i is a point $y^w_{j+r} \otimes z^c_{j+s,j} \in set(Y) - set(X)$ can be dealt with in the same way, yielding $u_{X,y^w_{j+r} \otimes z^c_{j+s,j}} \leq \sum_{k=1}^{j+r} b_{k,j,r,s} Bound_{k,j,r,s}$ and $v_{Y,y^w_{j+r} \otimes z^c_{j+s,j}} \leq \sum_{k=1}^{j+r} a_{k,j,r,s}$.

By the definition of $Bound_{k,j,r,s}$, it holds for any (j,r,s) that $\sum_{k=1}^{j+r} Bound_{k,j,r,s} \leq 4\sum_{t=1}^{T} p_t$ and that $\sum_{k=1}^{j+r} \sqrt{Bound_{k,j,r,s}} \leq 4\sum_{t=1}^{T} \sqrt{p_t}$. Now for the randomized lower bound, $a_{k,j,r,s} = b_{k,j,r,s} = 1$.

$$RLS(G) = \Omega\left(\min_{j,r,s} \max\left\{\frac{T - 2\sum_{t=1}^{T} p_t}{j+r}, \frac{T - 2\sum_{t=1}^{T} p_t}{\sum_{k=1}^{j+r} Bound_{k,j,r,s}}\right\}\right) = \Omega\left(\frac{T}{\sum_{t=1}^{T} p_t}\right). \tag{22}$$

For the quantum lower bound, pick $a_{k,j,r,s} = \sqrt{Bound_{k,j,r,s}}$, and $b_{k,j,r,s} = 1/\sqrt{Bound_{k,j,r,s}}$. Then

$$QLS(G) = \Omega\left(\min_{j,r,s} \sqrt{\frac{\left(T - 2\sum_{t=1}^{T} p_t\right)\left(T - 2\sum_{t=1}^{T} p_t\right)}{\left(\sum_{k=1}^{j+r} \sqrt{Bound_{k,j,r,s}}\right)\left(\sum_{k=1}^{j+r} \sqrt{Bound_{k,j,r,s}}\right)}}\right) = \Omega\left(\frac{T}{\sum_{t=1}^{T} \sqrt{p_t}}\right) \tag{23}$$

□

4 Applications to the two special graphs

In this section, we will apply Theorem 9 to the two special graphs. Note that in both cases, the probability p_t is not easy to upper bound.

4.1 Lower bounds for Local Search on the Boolean Hypercube

In this subsection, we shall prove Theorem 2 by applying Theorem 9 to $\{0,1\}^n$. To this end, we decompose the whole graph into the two parts $\{0,1\}^m$ and $\{0,1\}^{n-m}$. Pick the random walk $(\{0,1\}^m, v_0^w, W)$, where $v_0^w = 0^m \in \{0,1\}^m$ and $W(x,t) = \{i \in \{0,...,m-1\} : x^{(i)}\}$ for each vertex $x = x_0...x_{m-1} \in \{0,1\}^m$ and each $t \in \mathbb{N}$. Finally, note that the longest self-avoiding path of the graph $\{0,1\}^{n-m}$ is a Hamilton path with length $L = 2^{n-m} - 1$. The following bounds on p_t are rather loose but sufficient for our purpose. The proof of the lemma uses the generating function $(z_1 + ... + z_m)^t$ and some techniques inspired by the Fourier analysis.

Lemma 10 *For any $t \in \mathbb{N}$, we have*

$$p_t = \begin{cases} O(m^{-\lceil t/2 \rceil}) & \text{if } t \leq 10 \\ O(m^{-5}) & \text{if } 10 < t \leq m^2 \\ O(2^{-m}) & \text{if } t > m^2 \end{cases} \tag{24}$$

Consider that we put t balls randomly into m bins one by one. The j-th ball goes into the i_j-th bin. Denote by n_i the total number of balls in the i-th bin. We write $n_i \equiv b_i$ if $b_i = n_i \mod 2$. We say that $(i_1,...,i_t)$ *generates the parity sequence* $(b_1,...,b_m)$, or simply $(i_1,...,i_t)$ *generates* $(b_1,...,b_m)$, if $n_i \equiv b_i$ for all $i \in [m]$. For $b_1...b_m \in \{0,1\}^m$, denote by $p^{(t)}[b_1,...,b_m]$ the probability that $n_i \equiv b_i$, $\forall i \in [m]$. Let $p^{(t)} = \max_{b_1,...,b_m} p^{(t)}[b_1,...,b_m]$. It is easy to see that $p^{(t)} = p_t$ in Lemma 10, so we will prove the same bounds in Lemma 10 for $p^{(t)}$.

Proof We start with several simple observations. First, we assume that t and $\sum_{i=1}^{m} b_i$ have the same parity, because otherwise the probability is 0 and the lemma holds trivially. Second, by the symmetry, any permutation of $b_1,...,b_m$ does not change $p^{(t)}[(b_1,...,b_m)]$. Third, $p^{(t)}[(b_1,...,b_m)]$ decreases if we replace two 1's in $b_1,...,b_m$ by two 0's. Precisely, if we have two b_i's being 1, say $b_1 = b_2 = 1$, then $p^{(t)}[(b_1,...,b_m)] < p^{(t)}[(0,0,b_3,...,b_m)]$. In fact, note that

$$p^{(t)}[(b_1,...,b_m)] = \frac{1}{m^t} \sum_{\substack{n_1+...+n_m=t \\ n_i \equiv b_i, i \in [m]}} \frac{t!}{n_1!...n_m!} \tag{25}$$

$$= \frac{1}{m^t} \sum_{\substack{n_3+...+n_m \leq t \\ n_i \equiv b_i, i=3,...,m}} \left(\frac{t!}{(n_1+n_2)!n_3!...n_m!} \sum_{\substack{n_1+n_2=t-n_3-...-n_m \\ n_i \equiv b_i, i=1,2}} \frac{(n_1+n_2)!}{n_1!n_2!} \right) \tag{26}$$

where as usual, let $0! = 1$. If $n_3 + ... + n_m < t$, then

$$\sum_{\substack{n_1+n_2=t-n_3-...-n_m \\ n_i \equiv 1, i=1,2}} \frac{(n_1+n_2)!}{n_1!n_2!} = \sum_{\substack{n_1+n_2=t-n_3-...-n_m \\ n_i \equiv 0, i=1,2}} \frac{(n_1+n_2)!}{n_1!n_2!} \tag{27}$$

If $n_3 + ... + n_m = t$, then the only possible (n_1, n_2) is $(0, 0)$, so

$$\sum_{\substack{n_1+n_2=t-n_3-...-n_m \\ n_i \equiv 1, i=1,2}} \frac{(n_1+n_2)!}{n_1!n_2!} = 0, \qquad \sum_{\substack{n_1+n_2=t-n_3-...-n_m \\ n_i \equiv 0, i=1,2}} \frac{(n_1+n_2)!}{n_1!n_2!} = 1. \tag{28}$$

Thus $p^{(t)}[(1, 1, b_3, ..., b_m)] < p^{(t)}[(0, 0, b_3, ..., b_m)]$.

By the observations, it is sufficient to prove the lemma for the case $p^{(t)}[(0, ..., 0)]$ if t is even, and for the case $p^{(t)}[(1, 0, ..., 0)]$ if t is odd. Note that if t is even, then

$$p^{(t)}[(0, ..., 0)] = \sum_{i=1}^{m} \mathbf{Pr}[i_1 = i]\mathbf{Pr}[(i_2, ..., i_t) \text{ generates } (e_i)] \tag{29}$$

where e_i is the m-long vector with only coordinate i being 1 and all other coordinates being 0. By the symmetry, $p^{(t-1)}[e_1] = ... = p^{(t-1)}[e_m]$, thus $p^{(t)}[(0, ...0)] = p^{(t-1)}[e_1] = p^{(t-1)}[1, 0, ..., 0]$. Therefore, it is enough to show the lemma for even t.

We now express $p^{(t)}[0, ..., 0]$ in two ways. One is to prove the first case ($t \leq 10$) in the lemma, and the other is for the second case ($10 < t \leq m^2$) and the third case ($t > m^2$) in the lemma.

To avoid confusion, we write the number m of bins explicitly as subscript: $p_m^{(t)}[b_1, ..., b_m]$. We consider which bin(s) the first two balls is put into.

$$p_m^{(t)}[0, ..., 0] = \mathbf{Pr}[i_1 = i_2]p_m^{(t-2)}[0, ..., 0] + \mathbf{Pr}[i_1 \neq i_2]p_m^{(t-2)}[1, 1, 0, ..., 0] \tag{30}$$

$$= \frac{1}{m}p_m^{(t-2)}[0, ..., 0] + \frac{m-1}{m}p_m^{(t-2)}[1, 1, 0, ..., 0] \tag{31}$$

To compute $p_m^{(t-2)}[1, 1, 0, ..., 0]$, we consider to put $(t-2)$ balls in m bins. By the analysis of the third observations above, we know that

$$p_m^{(t-2)}[0, ..., 0] - p_m^{(t-2)}[1, 1, 0, ..., 0] \tag{32}$$

$$= \mathbf{Pr}[n_1 = n_2 = 0, n_3 \equiv 0, ..., n_m \equiv 0] \tag{33}$$

$$= \mathbf{Pr}[n_1 = n_2 = 0]\mathbf{Pr}[n_3 \equiv 0, ..., n_m \equiv 0 | n_1 = n_2 = 0] \tag{34}$$

$$= \left(\frac{m-2}{m}\right)^{t-2} p_{m-2}^{(t-2)}[0, ..., 0] \tag{35}$$

Therefore,

$$p_m^{(t)}[0, ..., 0] = \frac{1}{m}p_m^{(t-2)}[0, ..., 0] - \frac{m-1}{m}\left(\frac{m-2}{m}\right)^{t-2} p_{m-2}^{(t-2)}[0, ..., 0] \tag{36}$$

Now using the above recursive formula and the base case $p_m^{(2)}[0, ..., 0] = 1/m$, it is easy (but tedious) to prove by calculations that $p_m^{(t)}[0, ..., 0] = ((t-1)!!/m^{\frac{t}{2}})(1 - o(1))$ for even $t \leq 10$. This proves the first case in the lemma.

For the rest two cases, we shall use generating function and some technique inspired by Fourier analysis. Consider the generating function $(x_1 + ... + x_m)^t = \sum_{n_1+...+n_m=t} \binom{t}{n_1,...,n_m} x_1^{n_1}...x_m^{n_m}$. If $x_i \in \{-1, 1\}$, then $(x_1 + ... + x_m)^t = \sum_{n_1+...+n_m=t} \binom{t}{n_1,...,n_m} (-1)^{|\{i:x_i=-1, n_i \equiv 1\}|}$. We sum it over all $x_1...x_m \in \{-1, 1\}^m$. Note that for those $(n_1, ..., n_m)$ that has some $n_{i_0} \equiv 1$, it holds due to the cancelation that $\sum_{x_1,...,x_m \in \{-1,1\}} (-1)^{|\{i:x_i=-1, n_i \equiv 1\}|} = 0$. On the other hand, if all n_i's are even,

then $\sum_{x_1,...,x_m\in\{-1,1\}}(-1)^{|\{i:x_i=-1,n_i\equiv 1\}|}=2^m$. Thus we have $\sum_{x_1,...,x_m\in\{-1,1\}}(x_1+...+x_m)^t = 2^m\sum_{\substack{n_1+...+n_m=t\\ n_i\equiv 0,i\in[m]}}\binom{t}{n_1,...,n_m}$. Therefore

$$p^{(t)}[0,...,0]=\frac{1}{m^t}\sum_{\substack{n_1+...+n_m=t\\ n_i\equiv 0,i\in[m]}}\binom{t}{n_1,...,n_m} \tag{37}$$

$$=\frac{1}{2^m m^t}\sum_{x_1,...,x_m\in\{-1,1\}}(x_1+...+x_m)^t \tag{38}$$

$$=\frac{1}{2^m m^t}\sum_{i=0}^{m}\binom{m}{i}(m-2i)^t=\frac{1}{2^m}\sum_{i=0}^{m}\binom{m}{i}\left(1-\frac{2i}{m}\right)^t. \tag{39}$$

Note that t is even, so $p^{(t)}[0,...,0]$ decreases if t increases by 2, and this proves the second case of the lemma with the help of the first case. And if $t>m^2/2$, then

$$p^{(t)}[0,...,0]\le\frac{1}{2^m}\left(2+\left(1-\frac{2}{m}\right)^t\sum_{i=1}^{m-1}\binom{m}{i}\right)<2/2^m+e^{-m}=O(1/2^m) \tag{40}$$

This proves the third case of the lemma. □

Now we use the the lemma to prove Theorem 2. For the randomized lower bound, let $m=\lfloor(n+\log_2 n)/2\rfloor$, then $T=\Theta(2^{n/2}/n^{1/2})$ and $\sum_{t=1}^{T}p_t=O(1/n)$. Thus $RLS(\{0,1\}^n)=\Omega(\sqrt{n}2^{n/2})$. For the quantum lower bound, let $m=\lfloor(2n+\log_2 n)/3\rfloor$, then $T=\Theta(2^{n/3}/n^{1/3})$ and $\sum_{t=1}^{T}\sqrt{p_t}=O(1/\sqrt{n})$. Thus $QLS(\{0,1\}^n)=\Omega(2^{n/3}n^{1/6})$.

4.2 Lower bounds for Local Search on the constant dimensional grid

In this section we shall first prove a lower bound weaker than Theorem 3 in Section 4.2.1, and then improve it to Theorem 3 in Section 4.2.2.

4.2.1 A weaker lower bound

To simplify notations, we let $n=N^{1/d}$. As in Section 4.1, we decompose the grid into two parts, $[n]^m$ and $[n]^{d-m}$. For each vertex $x=x_0...x_{m-1}\in[n]^m$ and each $i\in\{0,...,m-1\}$, define $x^{(i),-}=x_0...x_{i-1}\max\{x_i-1,1\}x_{i+1}...x_{m-1}$ and $x^{(i),+}=x_0...x_{i-1}\min\{x_i+1,n\}x_{i+1}...x_{m-1}$. We perform the random walk $([n]^m,v_0^w,W)$ where $v_0^w=00...0\in[n]^m$ and $W(x,t)=\{x^{((t-1)\bmod m),+},x^{((t-1)\bmod m),-}\}$. To analyze the probability p_t in Theorem 9, we first consider the following simpler "line walk". Suppose a particle is initially put at point $i\in\{1,...,n\}$, and in each step the particle moves either to $\max\{1,i-1\}$ or to $\min\{n,i+1\}$, each with probability 1/2. Let $p_{ij}^{(t)}$ denote the probability that the particle starting from point i stops at point j after exact t steps of the walk. For $t\ge 1$, the following proposition gives a very good (actually tight) estimate on $\max_{ij}p_{ij}^{(t)}$.

Proposition 11 *For any $t\ge 1$,*

$$\max_{i,j}p_{ij}^{(t)}=\begin{cases}O(1/\sqrt{t}) & \text{if } t\le n^2\\ O(1/n) & \text{if } t>n^2\end{cases} \tag{41}$$

If there are no the two barriers (1 and n) then $p_{ij}^{(t)}$ is very easy to calculate: $p_{ij}^{(t)}=\binom{t}{t/2+(j-i)/2}$ if $j-i$ and t have the same parity, and 0 otherwise. However, since we now have the two barriers, it is hard to count the number of paths from i to j by exactly t steps. Here inspired by the so-called *reflecting rule*, we will construct a series of 1-1 correspondences to reduce the problem step by step to the no-barrier case.

Proof We consider two settings. One is as in the definition of the short walk, where we have only n points $0, ..., n-1$, and points 0 and $n-1$ are two barriers[5]. Another is the same except that the barriers are removed, and we have infinite points in a line. For each t-bit binary string $x = x_1...x_t$, we use P_i^x and Q_i^x to denote the two paths that starting at i and walk according to x in the two settings. Precisely, at step s, Q_i^x goes left if $x_s = 0$ and goes right if $x_s = 1$. P_i^x goes in the same way except that it will stand still if the point is currently at left (or right) end and it still wants to go left (or right). If the end point of P_i^x is j, then we write $i \rightarrow_t^{P,x} j$. Let $X_{ij}^{(t),P}$ be the set of $x \in \{0,1\}^t$ *s.t.* $i \rightarrow_t^{P,x} j$, and put $n_{ij}^{(t),P} = |X_{ij}^{(t),P}|$. Then by definition, $p_{ij}^{(t)} = n_{ij}^{(t),P}/2^t$. The notations $i \rightarrow_t^{Q,x} j$, $X_{ij}^{(t),Q}$ and $n_{ij}^{(t),Q}$ are similarly defined, with the corresponding P changed to Q. Note that $n_{ij}^{(t),Q} = \binom{t}{t/2+(j-i)/2}$ if $j-i$ and t have the same parity, and 0 otherwise. We now want to upper bound $n_{ij}^{(t),P}$ in terms of $n_{ij}^{(t),Q}$.

For a path P_i^x, if at some step it is at point 0 and wants to go left, we say it *attempts to pass the left barrier*. Similarly for the right barrier. We say a path is in the $\{a_s, b_s\}_{s=1}^l$ category if it first attempts to pass the left barrier for a_1 times, and then attempts to pass the right barrier for b_1 times, and so on. We call each round a stage s, which begins at the time that P_i^x attempts to pass the left barrier for the $(a_1 + ... + a_{s-1} + 1)$-th time, and ends right before the time that P_i^x attempts to pass the left barrier for the $(a_1 + ... + a_s + 1)$-th time. We also split each stage s into two halves, cutting at the time right before the path attempts to pass the right barrier for the $(b_1 + ... + b_{s-1} + 1)$-th time. Note that a_1 may be 0, which means that the path first attempts to pass the right barrier. Also b_l may be 0, which means the the last barrier the path attempts to pass is the left one. But all other a_i, b_i's are positive. Also note that in the case of $l = 0$, the path never attempts to pass either barrier. We partition $X_{ij}^{(t),P}$ as

$$X_{ij}^{(t),P} = \bigcup_{l,\ \{a_s,b_s\}_{s=1}^l} X_{ij}^{(t),P}[\{a_s, b_s\}_{s=1}^l] \tag{42}$$

where $X_{ij}^{(t),P}[\{a_s, b_s\}_{s=1}^l]$ contains those $x \in \{0,1\}^t$ *s.t.* P_i^x is in the category $\{a_s, b_s\}_{s=1}^l$. Put $n_{ij}^{(t),P}[\{a_s, b_s\}_{s=1}^l] = |X_{ij}^{(t),P}[\{a_s, b_s\}_{s=1}^l]|$, thus $n_{ij}^{(t),P} = \sum_l \sum_{\{a_s,b_s\}_{s=1}^l} n_{ij}^{(t),P}[\{a_s, b_s\}_{s=1}^l]$.

Now consider the corresponding paths in $X_{ij}^{(t),Q}$. The following observation relates P_i^x and Q_i^x.

Observation 1 *For each $x \in X_{ij}^{(t),P}[\{a_s, b_s\}_{s=1}^l]$, the following two properties hold for any s.*

1. *In the first half of stage s, the path Q_i^x touches (from right) but does not cross the point $\alpha_s = \sum_{r=1}^{s-1}(b_r - a_r) - a_s$.*
2. *In the second half of stage s, the path Q_i^x touches (from left) but does not cross the point $\beta_s = n - 1 + \sum_{r=1}^{s}(b_r - a_r)$*
3. *The path Q_i^x ends at $\gamma = j + \sum_{s=1}^{l}(b_s - a_s)$*

We let $Y_{i\gamma}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l]$ contain those $x \in \{0,1\}^t$ satisfying the three conditions in the above observation, and denote by $m_{i\gamma}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l]$ the size of the set $Y_{i\gamma}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l]$. Thus the observation says $X_{ij}^{(t),P}[\{\alpha_s, \beta_s\}_{s=1}^l] \subseteq Y_{ij}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l]$, and therefore we have $n_{ij}^{(t),P}[\{a_s, b_s\}_{s=1}^l] \leq m_{i\gamma}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l]$. So it is enough to upper bound $m_{i\gamma}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l]$.

Now for each $x \in Y_{i\gamma}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l]$, if we change the condition 1 in state $s = 1$ by allowing the path to cross the point α_1, and let $Z_{i\gamma}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l]$ be the new set satisfying the new conditions, then $m_{i\gamma}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l] = |Z_{i\gamma}^{(t),Q}[\{\alpha_s, \beta_s\}_{s=1}^l]| - |Z_{i\gamma}^{(t),Q}[\alpha_1 - 1, \beta_1, \{\alpha_s, \beta_s\}_{s=2}^l]|$. In other words, the set of paths touches (from right) but does not cross α_1 is the set of paths touches or crosses α_1 minus the set of paths touches or crosses $\alpha_1 - 1$.

[5]Here we let the n points be $0, ..., n-1$ instead of $1, ..., n$ just to make the later calculation cleaner

Now we calculate $|Z_{i\gamma}^{(t),Q}[\{\alpha_s,\beta_s\}_{s=1}^l]|$ by the so-called reflection rule. Suppose the first time that Q_i^x touches α_1 is t_1. We reflect the first t_1 part of the path Q_i^x with respect to the point α_1. Precisely, let $y=(1-x_1)...(1-x_{t_1})x_{t_1+1}...x_t$, then the paths Q_i^x and $Q_{2\alpha_1-i}^y$ merge at time t_1. And it is easy to check that it is a 1-1 correspondence between $Z_{i\gamma}^{(t),Q}[\{\alpha_s,\beta_s\}_{s=1}^l]$ and $Y_{2\alpha_1-i,\gamma}^{(t),Q}[\beta_1,\{\alpha_s,\beta_s\}_{s=2}^l]$, Here $Y_{2\alpha_1-i,\gamma}^{(t),Q}[\beta_1,\{\alpha_s,\beta_s\}_{s=2}^l]$ is the set of paths starting at $2\alpha_1-i$, satisfying (a) the condition 2 at the first stage, (b) both conditions 1 and 2 at the rest $l-1$ stages, and (c) condition 3. So

$$|Z_{i\gamma}^{(t),Q}[\{\alpha_s,\beta_s\}_{s=1}^l]| = |Y_{2\alpha_1-i,\gamma}^{(t),Q}[\beta_1,\{\alpha_s,\beta_s\}_{s=2}^l]| = m_{2\alpha_1-i,\gamma}^{(t),Q}[\beta_1,\{\alpha_s,\beta_s\}_{s=2}^l] \tag{43}$$

$$= m_{-2a_1-i,\gamma}^{(t),Q}[\beta_1,\{\alpha_s,\beta_s\}_{s=2}^l] \tag{44}$$

$$= m_{-a_1-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l] \tag{45}$$

where (44) is due to the fact that $\alpha_1=-a_1$, and (45) is because that the number of the paths does not change if we move all the paths right by a_1. Similarly, we have

$$|Z_{i\gamma}^{(t),Q}[\alpha_1-1,\beta_1,\{\alpha_s,\beta_s\}_{s=2}^l]| = m_{2\alpha_1-2-i,\gamma}^{(t),Q}[\beta_1,\{\alpha_s,\beta_s\}_{s=2}^l] \tag{46}$$

$$= m_{-a_1-2-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l] \tag{47}$$

Therefore,

$$n_{ij}^{(t),P}[\{a_s,b_s\}_{s=1}^l] \le m_{i\gamma}^{(t),Q}[\{\alpha_s,\beta_s\}_{s=1}^l] \tag{48}$$

$$= m_{-2a_1-i,\gamma}^{(t),Q}[\beta_1,\{\alpha_s,\beta_s\}_{s=2}^l] - m_{-2a_1-2-i,\gamma}^{(t),Q}[\beta_1,\{\alpha_s,\beta_s\}_{s=2}^l] \tag{49}$$

$$= m_{-a_1-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l] \tag{50}$$

$$- m_{-a_1-2-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l] \tag{51}$$

Now for any fixed $l>1$, we consider those categories with $a_1>0$ and $b_l>0$. Other cases can handled similarly. Note that $\alpha_s+a_1=b_1+\sum_{r=2}^{s-1}(b_r-a_r)-a_s$, $\beta_s+a_1=n-1+b_1+\sum_{r=2}^{s}(b_r-a_r)$ and $\gamma+a_1=j+b_1+\sum_{r=2}^{s}(b_r-a_r)$ are all functions of $(b_1,a_2,b_2,...,a_l,b_l)$, not of a_1 any more. Therefore,

$$\sum_{a_1,b_1,...,a_l,b_l>0} n_{ij}^{(t),P}[\{a_s,b_s\}_{s=1}^l] \tag{52}$$

$$\le \sum_{b_1,...,a_l,b_l>0}\sum_{a_1>0}(m_{-a_1-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l] \tag{53}$$

$$- m_{-a_1-2-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l]) \tag{54}$$

$$= \sum_{b_1,...,a_l,b_l>0}(m_{-1-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l] \tag{55}$$

$$+ m_{-2-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l]) \tag{56}$$

Note that due to the parity, only one of $m_{-1-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l]$ and $m_{-2-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l]$ is nonzero. So the summation of them two items is equal to the maximum of them. Now using the similar methods, *i.e.* reflecting with respect to points $(n-1+b_1)$ and $(n+b_1)$, moving the paths left by b_1, and finally collapsing the telescope, we can get

$$\sum_{b_1,...,a_l,b_l>0} m_{-1-i,\gamma+a_1}^{(t),Q}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^l] \tag{57}$$

$$= \sum_{a_2,b_2,...,a_l,b_l>0} \max\{m_{2n+i,\gamma+a_1-b_1}^{(t),Q}[\{\alpha_s+a_1-b_1,\beta_s+a_1-b_1\}_{s=2}^l], \tag{58}$$

$$m_{2n+i+1,\gamma+a_1-b_1}^{(t),Q}[\{\alpha_s+a_1-b_1,\beta_s+a_1-b_1\}_{s=2}^l]\} \tag{59}$$

and

$$\sum_{b_1,\ldots,a_l,b_l>0} m^{(t),Q}_{-2-i,\gamma+a_1}[\beta_1+a_1,\{\alpha_s+a_1,\beta_s+a_1\}_{s=2}^{l}] \tag{60}$$

$$= \sum_{a_2,b_2,\ldots,a_l,b_l>0} \max\{m^{(t),Q}_{2n+i+2,\gamma+a_1-b_1}[\{\alpha_s+a_1-b_1,\beta_s+a_1-b_1\}_{s=2}^{l}], \tag{61}$$

$$m^{(t),Q}_{2n+i+3,\gamma+a_1-b_1}[\{\alpha_s+a_1-b_1,\beta_s+a_1-b_1\}_{s=2}^{l}]\} \tag{62}$$

We continue this process, and finally it is

$$\sum_{a_1,b_1,\ldots,a_l,b_l>0} n^{(t),P}_{ij}[\{a_s,b_s\}_{s=1}^{l}] \le \max\{n^{(t),Q}_{2ln+i+h,\gamma+\sum_{s=1}^{l}(a_s-b_s)} : h=0,\ldots,4l-1\} \tag{63}$$

$$= \max\{n^{(t),Q}_{2ln+i+h,j} : h=0,\ldots,4l-1\} \tag{64}$$

$$= n^{(t),Q}_{2ln+i,j} \tag{65}$$

$$\le \binom{t}{\frac{t}{2}+\frac{j-i-2ln}{2}} \tag{66}$$

Thus

$$\sum_{l>0}\sum_{a_1,b_1,\ldots,a_l,b_l>0} n^{(t),P}_{ij}[\{a_s,b_s\}_{s=1}^{l}] \le \sum_{l>0}\binom{t}{\frac{t}{2}+\frac{j-i}{2}-ln} \tag{67}$$

where $\binom{t}{t'}=0$ if $t'<0$. Since $\binom{t}{t'}$ exponentially decreases when $t'<t/2-\Omega(\sqrt{t})$, we know that

$$\sum_{l>0}\binom{t}{\frac{t}{2}+\frac{j-i}{2}-ln} = \begin{cases} O(\frac{2^t}{\sqrt{t}}) & \text{if } t<n^2 \\ O(\frac{\sqrt{t}}{n}\frac{2^t}{\sqrt{t}}) = O(\frac{2^t}{n}) & \text{if } t\ge n^2 \end{cases} \tag{68}$$

For other categories that $a_1=0$ or $b_l=0$, the same result can be proved similarly, and the $l=0$ is easy since $n^{(t),Q}_{ij}=O(2^t/\sqrt{t})$. Putting all things together, we get the result

$$p^{(t)}_{ij} = \begin{cases} O(1/\sqrt{t}) & \text{if} \quad t\le n^2 \\ O(1/n) & \text{if} \quad t>n^2 \end{cases} \tag{69}$$

for any i and j, which completes our proof. □

Now we use Proposition 11 to prove to get weaker lower bounds for grids. Since the random walk $([n]^m, v_0^w, W)$ is just a product of m line walks, it is not hard to see that the p_t in the random walk $([n]^m, v_0^w, W)$ is equal to $O(1/\sqrt{t^m})$ if $t\le n^2$, and $O(1/n^m)$ if $t>n^2$. Now for the randomized lower bounds, when $d>4$ we pick $m=\lceil d/2\rceil>2$ and we get $RLS([n]^d)=\Omega\left(\frac{n^{d-m}}{O(1)+n^{d-m}/n^m}\right)=\Omega(n^{\lfloor d/2\rfloor})$, which is $\Omega(N^{\frac{1}{2}})$ if d is odd, and $\Omega(N^{\frac{1}{2}-\frac{1}{2d}})$ if d is even. For $d=4,3,2$, we let $m=2,2,1$ respectively, and get $RLS([n]^4)=\Omega(n^2/(\log n+1))=\Omega(N^{1/2}/\log N)$, $RLS([n]^3)=\Omega(n/(\log n+1/n))=\Omega(N^{1/3}/\log N)$, and $RLS([n]^2)=\Omega(n/(\sqrt{n}+1))=\Omega(N^{1/4})$.

For the quantum lower bounds, if $d>6$, we let m be the integer closest to $2d/3$, thus $m>4$. We get $QLS([n]^d)=\Omega\left(\frac{n^{d-m}}{O(1)+n^{d-m}/n^{m/2}}\right)$, which is $\Omega(N^{\frac{1}{3}})$ if $d=3d'$, $\Omega(N^{\frac{1}{3}-\frac{1}{3d}})$ if $d=3d'+1$, and $\Omega(N^{\frac{1}{3}-\frac{1}{6d}})$ if $d=3d'+2$. For $d=6$, let $m=4$ and we have $QLS([n]^6)=\Omega(n^2/\log n)=\Omega(N^{1/3}/\log N)$. For $d=5,4,3$, we let $m=d-2$ and then $QLS([n]^d)=\Omega(n^2/(n^{2-(d-2)/2}+n^{2-(d-2)/2}))=\Omega(n^{d/2-1})=\Omega(N^{1/2-1/d})$. For $d=2$, let $m=1$ and $QLS([n]^2)=\Omega(\frac{n}{n^{3/4}})=\Omega(n^{1/4})=\Omega(N^{1/8})$.

4.2.2 Improvement

One weakness of the above proof is the integer constraint of the dimension m. We now show a way to avoid the problem by allowing m to be any really number between 0 and d-1. The idea is

to partition the grid into many blocks, with different blocks representing different time slots, and the blocks are threaded into one very long block by many paths that are pairwise disjoint. More precisely, we view $[n]^d$ as the product of d line graph $[n]$. For each of the first $d-1$ line graphs, we cut it into n^{1-r} parts evenly, each of size n^r. (Here $r = m/(d-1)$). Then $[n]^{d-1}$ is partitioned into $n^{(d-1)(1-r)}$ smaller grids, all isomorphic to $[n^r]^{d-1}$. Putting the last dimension back, we have $n^{(d-1)(1-r)}$ *blocks*, all isomorphic to $[n^r]^{d-1} \times [n]$. Now the random walk will begin in the first block and within each block, it is just one step of random walk in $[n^r]^{d-1}$ followed by two steps of one-way walk in $[n]$. When the walk runs out of the clock $[n]$, the walk will change to the next block via a particular block-changing path. All block-changing paths are designed to be disjoint, and they "thread" all the blocks to form a $[n^r]^{d-1} \times [L]$ grid, where $L = (n-2n^r)n^{(1-r)(d-1)}$. ($L$ is not $n \cdot n^{(1-r)(d-1)}$ because we need $2n^r$ points for the block-changing paths.)

We now describe the partition and the walk precisely. For $x = x_0...x_{d-1}$ in $[n]^d$, let $x^{(k)=l} = x_0...x_{k-1}lx_{k+1}...x_{d-1}$, and $x^{(k)=(k)+i} = x_0...x_{k-1}(x_k+i)x_{k+1}...x_{d-1}$, where i satisfies $x_k + i \in [n]$. Recall that $x^{(i),-} = x^{(i)=\max\{x_i-1,1\}}$ and $x^{(i),+} = x^{(i)=\min\{x_i+1,n\}}$.

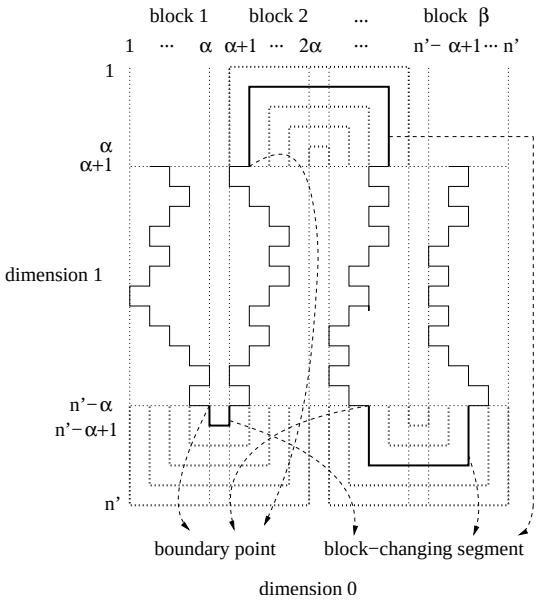

Figure 1: Illustration for changing a block in 2 dimensional grid

For any fixed constant $r \in (0,1)$, let $\alpha = \lfloor n^r \rfloor$, $\beta = \lfloor n^{1-r} \rfloor$ and $n' = \alpha\beta$. Note that $n' \geq (n^r-1)(n^{1-r}-1) = n - o(n)$. We now consider the slightly smaller grid $[n']^d$. Let V_1 be the set $[n']^{d-1} = \{x_0...x_{d-2} : x_i \in [n']\}$. We cut V_1 into β^{d-1} parts $\{x_0...x_{d-2} : (k_i-1)\alpha < x_i \leq k_i\alpha\}_{k_0...k_{d-2} \in [\beta]^{d-1}}$, each of which is a small grid isomorphic to $[\alpha]^{d-1}$. We then refer to the set $\{x_0...x_{d-2}x_{d-1} : (k_i-1)\alpha < x_i \leq k_i\alpha, i = 0,...,d-2, \alpha < x_{d-1} \leq n'-\alpha\}$ as the *block* $(k_0,...,k_{d-2})$. Note that $(k_0,...,k_{d-2})$ can be also viewed as a point in grid $[\beta]^{d-1}$, and there is a Hamilton path $HamPath_{\beta,d-1}$ in $[\beta]^{d-1}$, as defined in Section 2. We call the block $(k'_0,...,k'_{d-2})$ *the next block* of the block $(k_0,...,k_{d-2})$ if $(k'_0,...,k'_{d-2})$, viewed as the point in $[\beta]^{d-1}$, is the next point of $(k_0,...,k_{d-2})$ in $HamPath_{\beta,d-1}$. Note that by our definition of $HamPath_{\beta,d-1}$, we know that $\exists i \in \{0,...,d-2\}$ *s.t.* $k'_j = k_j$ for all $j \neq i$ and $k'_i \in \{k_i+1, k_i-1\}$, only one of $k_0,...,k_{d-2}$. We call the the block $(k_0,...,k_{d-2})$ *the last block* if $(k_0,...,k_{d-2})$ is the last point in $HamPath_{\beta,d-1}$.

Now we define the random walk by describing how a particle may go from start to end. The path set is just all the possible paths the particle goes along. Intuitively, within one block, the last dimension $d-1$ serves as the clock space. So as before, we perform one step of line walk (along the dimension which is the circularly next dimension of the last one that the walk just goes along), followed by two steps of walk in the clock space. If we run out of clock, we say we reach *a boundary point* at the current block, and we change to the next block via a path segment called

block-changing segment. In what follows, we specify how the particle move along a block-changing segment. We always use $x_0...x_{d-1}$ to denote the current position of the particle. Thus the instruction $x_0 = x_0 + 1$, for example, means that the particle moves from $x_0...x_{d-1}$ to $(x_0+1)x_1...x_{d-1}$. We also use $(k_0, ..., k_{d-2})$ to denote the current block which the particle is in.

1. Initially $x_0 = ... = x_{d-2} = 0$, $x_{d-1} = \alpha + 1$, $k_0 = ... = k_{d-2} = 1$.
2. **for** $t = 1$ **to** $(n' - 2\alpha)\beta^{d-1}$,
   Let $t' = \lfloor \frac{t-1}{n'-2\alpha} \rfloor$, $i = (t-1) \bmod (d-1)$
   do either $x_i = \max\{x_i - 1, (k_i - 1)\alpha + 1\}$ or $x_i = \min\{x_i + 1, k_i\alpha\}$ randomly
   if $t \neq k(n' - 2\alpha)$ for some positive integer k,
   do $x_{d-1} = x_{d-1} + (-1)^{t'}$ twice
   else
   if the particle is not in the last block
   (Suppose the current block changes to the next block by increasing k_j by $b \in \{-1, 1\}$)
   do $x_{d-1} = x_{d-1} + (-1)^{t'}$ **for** $(\alpha + 1 - x_j)$ times
   do $x_j = x_j + b$ **for** $2(\alpha + 1 - x_j) - 1$ times
   do $x_{d-1} = x_{d-1} + (-1)^{t'+1}$ **for** $(\alpha + 1 - x_j)$ times
   $k_j = k_j + b$
   else
   The particle stops and the random walk ends

It is easy to check that every boundary point has one unique block-changing segment, and different block-changing segments do not intersect. Thus the block-changing segments thread all the blocks to form a $[\alpha]^{d-1} \times [L]$ grid, where $L = (n' - 2\alpha)\beta^{d-1}$. We can think of the construction the same as a long grid $[\alpha]^{d-1} \times [(n' - 2\alpha)\beta^{d-1}]$. Actually, what we care about is, as before, the probability that the random walk starting from a point $x = x_0...x_{d-1}$ passes another point $x' = x_0...x_{d-1}$ after exactly $t' - t$ steps. Here t is the time that the random path passes x and t' is the time that the path passes x'. Note that t is fixed and known by x itself; similarly for t'. Denote this probability by $\mathbf{Pr}[x \to x']$. Suppose $x_i = (k_i - 1)\alpha + y_i$ and $x'_i = (k'_i - 1)\alpha + y'_i$ for $i \in \{0, ..., d-2\}$.

We first consider the case that one of the two points, say x' is on a block-changing segment. Since different block-changing segments never intersect, a path passes x' if and only if the path passes the boundary point x'' at the beginning of the block-changing segment that x' is in. Also note that the time that the path passes x'' is also t' because the time does not elapse on the block-changing segment. So it holds that $\mathbf{Pr}[x \to x'] = \mathbf{Pr}[x \to x'']$, and it is enough to consider the case that both x and x' are not in clock-changing segments.

Now suppose both x and x' are not in clock-changing segments. In general, x and x' may be not in the same block , so going from x to x' needs to change blocks. Recall that to change from the block $(k_0, ..., k_{d-2})$ to the next one, only one k_i changes by increasing or decreasing by 1. Suppose that to go to x' from x, we change blocks for c times, by changing $k_{i_1}, k_{i_2}, ..., k_{i_c}$ in turn. Let $n_j = |\{s \in [c] : i_s = j\}|$. Note that to get to x' from x after $t' - t$ steps, the coordinate j needs to be x'_j after $t' - t$ steps for each coordinate $j \in \{0, ..., d-2\}$. It is not hard to see that if a block-changing needs to change k_j, then only the coordinate j gets reflected within the current block. That is, suppose the coordinate j is $(k_j - 1)\alpha + y_j$ before the block-changing, then it changes to $(k_j - 1)\alpha + \alpha + 1 - y_j$ after the block-changing. So if $c = 1$, then $\mathbf{Pr}[x \to x']$ is equal to the probability that a random walk in $[\alpha]^{d-1}$ starting from $y_0...y_{d-2}$ stops at $y''_0...y''_{d-2}$ after $t' - t$ steps, where $y''_j = y'_j$ if $j \neq i_1$ and $y''_{i_1} = (k_{i_1} - 1)\alpha + \alpha + 1 - y'_{i_1}$. For general c, $\mathbf{Pr}[x \to x']$ is equal to the probability that a random walk in $[\alpha]^{d-1}$ starting from $y_0...y_{d-2}$ stops at $y''_0...y''_{d-2}$ after $t' - t$ steps, where $y''_j = y'_j$ if n_j is even and $y''_j = (k_j - 1)\alpha + \alpha + 1 - y'_j$ if n_j is odd. Note that the latter probability has nothing to do with the block-changing; it is just the same as we have a clock space $[(n' - 2\alpha)\beta^{d-1}]$ to record the random walk on $[\alpha]^{d-1}$. Thus we can use Proposition 11 to upper bound this probability and just think of the graph as $[n^r]^{d-1} \times [L]$ and use Theorem 9, with $G^w = [n^r]^{d-1}$ and $G^c = [L]$.

Now we have $T = \lfloor L/2 \rfloor$ and $p_t = 1/\sqrt{t^{d-1}}$ for $t \leq n^{2r}$ and $p_t = 1/n^{r(d-1)}$ for $t > n^{2r}$. Now for randomized lower bounds, if $d \geq 4$, then let $r = d/(2d-2)$ and we get $RLS([n]^d) = \Omega(L/(\sum_{t=1}^{n^{d/(d-1)}} 1/\sqrt{t^{d-1}} + L/n^{d/2}))) = \Omega(n^{d/2}) = \Omega(N^{1/2})$. If $d = 3$, let $r = 3/4 - \log\log n/(4\log n)$, and we get $RLS([n]^3) = \Omega((N/\log N)^{1/2})$. For $d = 2$, let $r = 2/3$ and we get $RLS([n]^2) = \Omega(N^{1/3})$.

For the quantum lower bounds, if $d \geq 6$, then let $r = 2d/(3d-3)$ and we get $QLS([n]^d) = N^{1/3}$. If $d = 5$, then let $r = 5/6 - \log\log n/(6\log n)$ and $QLS([n]^5) = (N/\log N)^{1/3}$. For $2 \leq d \leq 5$, we let $r = d/(d+1)$, then $QLS([n]^d) = N^{1/2-1/(d+1)}$. This completes the proof of Theorem 3.

5 New algorithms for Local Search on general graphs

In [4, 2], a randomized and a quantum algorithm for Local Search on general graphs are given as follows. Do a random sampling over all the vertices, find a vertex v in them with the minimum f-value. (For the minimum f-value finding procedure, The randomized algorithm in [4] just queries all these vertices and find the minimum, while the quantum algorithm in [2] uses the algorithm by Durr and Hoyer [12] based on Grover search [13] to get a quadratic speedup.) If v is a local minimum, then return v; otherwise we follow a *decreasing path* as follows. Find a neighbor of v with the minimum f-value, and continue this minimum-value-neighbor search process until getting to a local minimum. We can see that the algorithms actually fall into the generic algorithm category (see Section 1), with the initial point picked as the best one over some random samples.

In this section, we give new randomized and quantum algorithms, which work better than the generic ones when the graph expands slowly. Here the idea is that after finding the minimum vertex v of the sampled points, instead of following the decreasing path of v, we start over within a smaller range, which contains those vertices "close to" v. If this smaller range contains a local minimum for sure, then we can simply search a local minimum in it and do this procedure recursively. But one caveat here is that a straightforward recursion does not work, because a local minimum u in the smaller range may be not a local minimum in the original larger graph G (since u may have more neighbors in G). So we shall find a small range which has a "good" boundary in the sense that all vertices on the boundary have a large f-value.

Now we describe the algorithm precisely, with some notations as follows. For $G = (V, E)$, a given function $f : V \to \mathbb{N}$, a vertex $v \in V$ and a set $S \subseteq V$, let $n(v, S) = |\{u \in S : f(u) < f(v)\}|$. The boundary $B(S)$ of the set S is defined by $B(S) = \{u \in V : \exists v \in V - S \ s.t. \ (u, v) \in E\}$. In particular, $B(V) = \emptyset$. A decreasing path from $v \in V$ is a sequence of vertices $v = v_0, v_1, ..., v_k$ such that $f(v_{i+1}) = \min_{v:(v_i,v)\in E} f(v) < f(v_i)$ for $i = 0, ..., k-1$ and v_k is a local minimum. We write $f(u) \leq f(S)$ if $f(u) \leq f(v)$ for all $v \in S$. In particular, it always holds that $f(u) \leq f(\emptyset)$. Suppose $d = \max_{u,v\in V} |u - v|$ is the diameter of the graph, and $\delta = \max_{v\in V} |\{u : (u, v) \in E\}|$ is the max degree of the graph. In the following algorithm, the asymptotical numbers at the end of some command lines are the numbers of randomized or quantum queries needed for the line. For those commands without any number, no query is needed.

1. $m_0 = d$, $U_0 = V$;
2. $i = 0$;
3. **while** $(|m_i| > 10)$ **do**
   (a) Randomly pick (with replacement) $\lceil \frac{8|U_i|}{m_i} \log \frac{1}{\epsilon_1} \rceil$ vertices from U_i, where $\epsilon_1 = 1/(10 \log_2 d)$;
   (b) Search the sampled vertices for one v_i with the minimal f value.
   - Randomized algorithm: query all the sampled vertices and get v_i. — $O\left(\frac{8|U_i|}{m_i} \log \frac{1}{\epsilon_1}\right)$
   - Quantum algorithm: use Durr and Hoyer's algorithm [12] with the error probability at most $\epsilon_2 = 1/(10 \log_2 d)$. — $O\left(\sqrt{\frac{8|U_i|}{m_i} \log \frac{1}{\epsilon_1}} \log \frac{1}{\epsilon_2}\right)$
   (c) **if** $i = 0$, **then** $u_{i+1} = v_i$;
   else if $f(u_i) \leq f(v_i)$, **then** $u_{i+1} = u_i$;
   else $u_{i+1} = v_i$;

(d) **for** $j = 1, 2, ...$

i. Randomly pick $m_{ij} \in M_i = \{m : m_i/8 \le m \le m_i/2,\ |W(m)| \le 10|U_i|/m_i\}$, where $W(m) = \{w \in U_i : |w - u_{i+1}| = m\}$. Let $W_{ij} = W(m_{ij})$.

ii. Test whether $f(u_{i+1}) \le f(W_{ij})$
- Randomized algorithm: query all vertices in W_{ij}. — $O(|W_{ij}|)$
- Quantum algorithm: use Durr and Hoyer's algorithm [12] on W_{ij} with the error probability at most $\epsilon_3 = 1/(200 \log_2 d)$. — $O\left(\sqrt{|W_{ij}|} \log \frac{1}{\epsilon_3}\right)$

iii. If the answer is Yes, jump out of this **for** loop and go to Step 3e.

(e) $J_i = j$, $m_{i+1} = m_{ij}$, $W_i = W_{ij}$, $U_{i+1} = \{u \in U_i : |u - u_{i+1}| \le m_{i+1}\}$;

(f) $i = i + 1$;

4. $I = i$;

5. Follow a decreasing path of u_I to find a local minimum.
- Randomized algorithm: in each step, query all the neighbors — $O(\delta)$
- Quantum algorithm: in each step, use Durr and Hoyer's algorithm with the error probability at most 1/100 — $O(\sqrt{\delta})$

Define $c(k) = \max_{v \in V} |\{u : |u - v| \le k\}|$. Apparently, the expanding speed of a graph is upper bounded by $c(k)$. The following theorem says that the algorithm is efficient if $c(k)$ is small.

Theorem 12 *The algorithm outputs a local minimum with probability at least 1/2. The randomized algorithm uses* $O\left(\sum_{i=0}^{I-1} \frac{c(m_i)}{m_i} \log \log d\right)$ *queries in expectation, and the quantum algorithm uses* $O\left(\sum_{i=0}^{I-1} \sqrt{\frac{c(m_i)}{m_i}} (\log \log d)^{1.5}\right)$ *queries in expectation.*

In case that $c(k) = O(k^\alpha)$ *for some* $\alpha \ge 1$ *and* $k = 1, ..., d$*, the expected number of queries that the randomized algorithm uses is* $O\left(\frac{d^{\alpha-1}-1}{1-2^{1-\alpha}} \log \log d\right)$ *if* $\alpha > 1$ *and* $O(\log d \log \log d)$ *if* $\alpha = 1$*. The expected number of queries that the quantum algorithm use is* $O\left(\frac{d^{\frac{\alpha-1}{2}}-1}{1-2^{\frac{1-\alpha}{2}}} (\log \log d)^{1.5}\right)$ *if* $\alpha > 1$ *and* $O(\log d \log log d)$ *if* $\alpha = 1$*.*

Several comments before proving the theorem:

1. $\lim_{\alpha \to 1} \frac{d^{\alpha-1}-1}{1-2^{1-\alpha}} = \lim_{\alpha \to 1} \frac{d^{\frac{\alpha-1}{2}}-1}{1-2^{\frac{1-\alpha}{2}}} = \log_2 d$

2. If $\alpha - 1 \ge \epsilon$ for some constant $\epsilon > 0$, then $\frac{d^{\alpha-1}-1}{1-2^{1-\alpha}} = \Theta(d^{\alpha-1})$ and $\frac{d^{\frac{\alpha-1}{2}}-1}{1-2^{\frac{1-\alpha}{2}}} = \Theta(d^{(\alpha-1)/2})$. If further the bound $c(k) = O(k^\alpha)$ is tight in the sense that $N = c(d) = \Theta(d^\alpha)$, then $RLS(G) = O\left(\frac{N}{d} \log \log d\right)$ and $QLS(G) = O\left(\sqrt{\frac{N}{d}} (\log \log d)^{1.5}\right)$.

3. For 2-dimensional grid, $d = \Theta(n)$ and $\alpha = 2$. Thus Theorem 5 follows immediately.

Proof We shall prove the theorem for the quantum algorithm. The analysis of the randomized algorithm is almost the same (and actually simpler). We say W_i is *good* if $f(u_{i+1}) \le f(W_i)$. We shall first prove the following claim, then the theorem follows easily.

Claim 3 *For each* $i = 0, 1, ..., I - 1$*, the following three statements hold.*

1. $n(u_{i+1}, U_{i+1}) \le n(u_{i+1}, U_i) \le m_i/8 \le m_{i+1}$ *with probability* $1 - \epsilon_1 - \epsilon_2$.
2. *If* $n(u_{i+1}, U_i) \le m_i/8$*, then* W_i *is good with probability* $1 - \epsilon_3 J_i$*, and* $\mathbf{E}[J_i] \le 2$.
3. *If* $W_0, ..., W_i$ *are all good, then* $f(u_{i+1}) \le f(B(U_{i+1}))$*, and* $u_{i+1} \notin B(U_{i+1})$.

Proof 1: In Step 3a - 3c, denote by S the set of the $\lceil \frac{8|U_i|}{m_i} \log \frac{1}{\epsilon_1} \rceil$ sampled vertices in Step 3a, and let $a = \min_{u \in S} f(u)$. Then $|\{v \in U_i : f(v) < a\}| \le m_i/8$ with probability at least $1 - \epsilon_1$. Step 3b can find a u achieving the minimum in the definition of a with probability at least $1 - \epsilon_2$.

Put the two things together, we have $n(v_i, U_i) \le m_i/8$ with probability at least $1 - \epsilon_1 - \epsilon_2$. Since $f(u_{i+1}) \le f(v_i)$ (by Step 3c), $U_{i+1} \subseteq U_i$ (by Step 3e) and $m_{i+1} \ge m_i/2$ (by Step 3(d)i), we have $n(u_{i+1}, U_{i+1}) \le n(u_{i+1}, U_i) \le n(v_i, U_i) \le m_i/8 \le m_{i+1}$ with probability at least $1 - \epsilon_1 - \epsilon_2$.

2: We say an m_{ij} is good if the corresponding W_{ij} is good, *i.e.* $f(u_{i+1}) \le f(W_{ij})$. Note that for any $m_{ij} \in [m_i]$, we have $W_{ij} \subseteq U_i$, and also have $W_{ij} \cap W_{ij'} = \emptyset$ if $m_{ij} \ne m_{ij'}$. Therefore, if $n(u_{i+1}, U_i) \le m_i/8$, then at most $m_i/8$ distinct m_{ij}'s in $[m_i]$ are *not* good. Also note the number of distinct m_{ij}'s *s.t.* $|W(m_{ij})| > 10|U_i|/m_i$ is less than $m_i/10$. Therefore, $|M_i| \ge (\frac{3}{8} - \frac{1}{10})|U_i|/m_i > |U_i|/4m_i$. So if $n(u_{i+1}, U_i) \le m_i/8$, a random m_{ij} in M_i is good with probability at least $1/2$, and thus $\mathbf{E}[J_i] \le 2$. Also the probability that all the Grover searches in Step 3(d)ii are correct is at least $1 - J_i\epsilon_3$.

3: We shall first prove $B(U_{i+1}) \subseteq B(U_i) \cup W_i$. In fact, any $s \in B(U_{i+1})$ satisfies that $s \in U_{i+1}$ and that $\exists t \in V - U_{i+1}$ *s.t.* $|s - t| = 1$. Recall that $U_{i+1} \subseteq U_i$, so if $t \in V - U_i$, then $s \in B(U_i)$ by definition. Otherwise $t \in U_i - U_{i+1}$, and thus $t \in U_i$ and $|t - u_{i+1}| > m_{i+1}$ by the definition of U_{i+1}. Noting that $|s - u_{i+1}| \le m_{i+1}$ since $s \in U_{i+1}$, and that $|s - t| = 1$, we have $|s - u_{i+1}| = m_{i+1}$, which means $s \in W_i$. Thus for all $s \in B(U_{i+1})$, either $s \in B(U_i)$ or $s \in W_i$ holds, which implies $B(U_{i+1}) \subseteq B(U_i) \cup W_i$.

Applying the result recursively, we have $B(U_{i+1}) \subseteq B(U_0) \cup W_0 \cup ... \cup W_i = W_0 \cup ... \cup W_i$. Since we have $f(u_{i+1}) \le f(u_i) \le ... \le f(u_1)$ (by Step 3c) and $f(u_{k+1}) \le f(W_k)$ (for $k = 0, ..., i$) by the assumption that all W_k's are good, we know that $f(u_{i+1}) \le f(W_0 \cup ... \cup W_i)$, which implies $f(u_{i+1}) \le f(B(U_{i+1}))$.

For $u_{i+1} \notin B(U_{i+1})$, it is sufficient to prove $u_{i+1} \notin B(U_i)$ and $u_{i+1} \notin W_i$. The latter is easy to see by the definition of W_i. For $u_{i+1} \notin B(U_i)$, we can actually prove $u_{k+1} \notin B(U_k)$ by induction on $k = 0, ..., i$. The base case of $k = 0$ is trivial because $B(U_0) = \emptyset$. Now suppose $u_k \notin B(U_{k-1})$. There are two cases of u_{k+1} by Step 3c. If $f(u_k) \le f(v_k)$, then $u_{k+1} = u_k \notin B(U_{k-1})$. Again by the definition of W_{k-1} we know that $u_k \notin W_{k-1}$ and thus $u_{k+1} = u_k \notin B(U_k)$. The other case is $f(u_k) > f(v_k)$, then $u_{k+1} = v_k$, and therefore $f(u_{k+1}) = f(v_k) < f(u_k) \le f(B(U_k))$, which implies that $u_{k+1} \notin B(U_k)$. □

(Continue the proof of Theorem 12) Then by the claim, we know that with probability at least $1 - I(\epsilon_1 + \epsilon_2) - \sum_{i=0}^{I-1} J_i\epsilon_3$, we will have that

$$n(u_I, U_I) \le m_I, \qquad f(u_I) \le f(B(U_I)), \qquad u_I \notin B(U_I). \tag{70}$$

Note that the correctness of the algorithms follow these three items. Actually, by the last two items, we know that any decreasing path from u_I is contained in U_I. Otherwise suppose $(u_I^0, u_I^1, ..., u_I^T)$ is a decreasing path from u_I (so $u_I^0 = u_I$), and the first vertex out of U_I is u_I^t, then $u_I^{t-1} \in B(U_I)$. Since $u_I^0 \notin B(U_I)$, we have $t - 1 > 0$ and thus $f(u_I^{t-1}) < f(u_I)$, contradicting to $f(u_I) \le f(B(U_I))$. Now together with the first item, we know that any decreasing path from u_I is no more than m_I long. Thus Step 5 will find a local minimum by following a decreasing path.

The error probability of the algorithm is $I(\epsilon_1 + \epsilon_2) + J\epsilon_3 + 10/100$, where $J = \sum_{i=0}^{I-1} J_i$. Since $\mathbf{E}[J] = 2I$, we know by Markov inequality that with $J < 20I$ with probability at least $1/10$. Since $\epsilon_1 = \epsilon_2 = 1/(10 \log_2 d)$ and $\epsilon_3 = 1/(200 \log_2 d)$, and note that $I \le \log_2 d$ because $m_0 = d$ and $m_{i+1} \le \lceil m_i/2 \rceil$. So the total error probability is less than $1/2$.

We now consider the number of queries used in the i-th iteration. Note from Step 1 and Step 3e that $|U_i| \le c(m_i)$ for $i = 0, 1, ..., I - 1$. So Step 3b uses $O(\sqrt{(8|U_i|/m_i) \log\log d} \log\log d) = O(\sqrt{c(m_i)/m_i}(\log\log d)^{1.5})$ queries. Also note from Step 3(d)i that $|W_{ij}| \le 10|U_i|/m_i$, so the number of queries used by Step 3d is $O(\sum_{j=1}^{J_i} \sqrt{c(m_i)/m_i} \log\log d)$ which has the expectation of $O(\sqrt{c(m_i)/m_i} \log\log d)$. Finally, Step 5 uses $O(\sqrt{\delta})$ queries. Note that $\delta = c(1) = O(c(m_I)/m_I)$ where m_I is a constant integer in $[6, 10]$. Altogether, the total expected number of queries used is $O((\sum_{i=0}^{\log_2 d - 1} \sqrt{c(m_i)/m_i})(\log\log d)^{1.5})$.

If $c(k) = O(k^\alpha)$ for some $\alpha \ge 1$ and $k = 1, ..., d$, then $\sum_{i=0}^{\log_2 d - 1} \sqrt{\frac{c(m_i)}{m_i}} = \sum_{i=0}^{\log_2 d - 1} m_i^{(\alpha-1)/2} = \sum_{i=0}^{\log_2 d - 1} (d/2^i)^{(\alpha-1)/2} = \frac{d^\beta - 1}{1 - 2^{-\beta}}$ where $\beta = (\alpha - 1)/2$. (Note that this gives a quantum upper bound of $O(\log d (\log\log d)^{1.5})$ when $\alpha = 1$, but this is worse than the randomized algorithm, which uses $O(\log d \log\log d)$ queries. So if $\alpha = 1$, the quantum algorithm just uses the randomized one.) □

6 Concluding Remarks: further improvements

The paper gives new lower and upper bounds for Local Search problems. Some other random walk can be used to further improve the lower bound on low dimension grid cases. For example, by cutting the 2-dimensional grid into $n^{2/5}$ blocks (each of size $n^{4/5} \times n^{4/5}$) and using a random walk similar to Aaronson's in [2] (but with some modifications to make the path self-avoiding), we can prove $QLS([n]^2) = N^{1/5}/\log N$. But this walk suffers from the fact that the "passing probability" is now $n^{4/5}$ times the "stopping probability". So it only works better at dimension 2. We put the further results in a complete version of the paper.

Acknowledgement

The author thanks Scott Aaronson, Xiaoming Sun and Andy Yao very much for many valuable discussions. Thanks also to Yves Verhoeven for pointing out an error in the upper bound section in a previous version.

References

[1] K. Aardal, S. Hoesel, J.K. Lenstra, L. Stougie. A decade of combinatorial optimization. CWI Tracts 122, pp. 5-14, 1997

[2] S. Aaronson. Lower Bounds for Local Search by Quantum Arguments, Proceedings of the thirty-sixth Annual ACM Symposium on Theory of Computing, pp. 465-474, 2004.

[3] E. Aarts and J. Lenstra, John Wiley & Sons, Inc. New York, NY, USA, 1997

[4] D. Aldous. Minimization algorithms and random walk on the d-cube, Annals of Probability, 11(2), pp.403-413, 1983.

[5] I. Althofer and K. Koschnich. On the deterministic complexity of searching local maxima, Discrete Applied Mathematics 43, pp. 111-113, 1993.

[6] A. Ambainis. Polynomial degree vs. quantum query complexity. Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, pp. 230-239, 2003.

[7] A. Ambainis. Quantum lower bounds by quantum arguments, Journal of Computer and System Sciences, 64, pp. 750-767, 2002.

[8] H. Barnum, M. Saks, M. Szegedy. Quantum query complexity and semidefinite programming. Proceedings of the 18th Annual IEEE Conference on Computational Complexity, pp. 179-193, 2003.

[9] R. Beals, H. Buhrman, R. Cleve, M.Mosca, R. deWolf. Quantum lower bounds by polynomials. Journal of ACM, 48, pp. 778-797, 2001.

[10] H. Buhrman, R. de Wolf. Complexity measures and decision tree complexity: a survey. Theoretical Computer Science, Volume 288, Issue 1, pp. 21-43, 2002.

[11] C. Durr, M. Heiligman, P. Hoyer, M. Mhalla. Quantum query complexity of some graph problems. Proceedings of the 31st International Colloquium on Automata, Lanaguages, and Programming, pp. 481-493, 2004.

[12] C. Durr, P. Hoyer. A quantum algorithm for finding the minimum, 1996. quant-ph/9607014

[13] L. Grover. A fast quantum mechanical algorithm for database search, Proceedings of the 28th Annual ACM Symposium on the Theory of Computing, pp. 212-219, 1996.

[14] P. Hoyer, R. Spalek. Lower bounds on quantum query complexity, to appear in BEATCS, 2005.

[15] D. Johnson, C. Papadimitriou, and M. Yannakakis. How easy is local search, Journal of Computer and System Sciences 37, pp. 429448, 1988.

[16] S. Laplante, F. Magniez. Lower bounds for randomized and quantum query complexity using Kolmogorov arguments, Proceedings of the 19th Annual IEEE Conference on Computational Complexity, pp. 294-304, 2004.

[17] D. Llewellyn and C. Tovey. Dividing and conquering the square. Discrete Applied Mathematics 43, pp. 131-153, 1993.

[18] D. Llewellyn, C. Tovey. and M. Trick. Local optimization on graphs, Discrete Aplied Mathematics 23, pp. 157 - 178, 1989. Erratum: 46, pp. 93-94, 1993.

[19] N. Megiddo, and C. Papadimitriou. On total functions, existence theorems, and computational complexity, Theoretical Computer Science 81, pp. 317324, 1991.

[20] J. Orlin, A. Punnen, A. Schulz. Approximate local search in combinatorial optimization, SIAM Journal on Computing, 33(5), pp. 12011214, 2004.

[21] M.Santha and M. Szegedy. Quantum and classical query complexities of local search are polynomially related, Proceedings of the thirty-sixth annual ACM symposium on Theory of computing, pp. 494-501, 2004.

[22] R. Spalek and M. Szegedy. All quantum adversary methods are equivalent. In Proceedings of 32nd International Colloquium on Automata, Languages and Programming (ICALP'05), pp. 1299-1311, LNCS 3580, Lisboa, Portugal, 2005.

[23] Y. Verhoeven, Enhanced algorithms for Local Search. quant-ph/0506019

[24] S. Zhang. On the power of Ambainis lower bounds, Theoretical Computer Science, 339(2-3), pp. 241-256, 2005.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'