Quantum Complexity of Testing Group Commutativity*

Frédéric Magniez† Ashwin Nayak‡

May 7, 2007

Abstract

We consider the problem of testing the commutativity of a black-box group specified by its k generators. The complexity (in terms of k) of this problem was first considered by Pak, who gave a randomized algorithm involving $O(k)$ group operations. We construct a quite optimal quantum algorithm for this problem whose complexity is in $\tilde{O}(k^{2/3})$. The algorithm uses and highlights the power of the quantization method of Szegedy. For the lower bound of $\Omega(k^{2/3})$, we give a reduction from a special case of Element Distinctness to our problem. Along the way, we prove the optimality of the algorithm of Pak for the randomized model.

1 Introduction

A direction of research in quantum computation pioneered by Grover [Gro96] around search problems in unstructured, structured, or partially structured databases has recently been infused with new ideas for algorithm design. In contrast to problems based on the Hidden Subgroup Problem (HSP) (see for instance Ref. [Mos99]), the speed up for these search problems is often only polynomial.

Usually in search problems, the access to the input is done via an oracle. This leads to the notion of query complexity which measures the number of accesses to the oracle. While no significant lower bounds are known for quantum time complexity, the oracle constraint sometimes enables us to prove such bounds in the query model. For promise problems quantum query complexity indeed can be exponentially smaller than the randomized one. A prominent example is HSP. On the other hand, for total functions, deterministic and quantum query complexities are polynomially related [BBC+01].

In HSP, the group with its all structure is known to the algorithm designer, and the group operations are generally efficiently computable. In the event that the group is not explicitly known, or the group operations are not efficient to implement, it is appropriate to model the group operations by an oracle or a black-box. The notion of *black-box groups* was introduced by Babai and Szemerédi [BS84]. In this model, the elements of a group are encoded by words over a finite alphabet, and the group operations are performed by an oracle (the black-box). The groups are assumed to be specified by generators, and the encoding of group elements is not necessarily unique: different strings may encode the same group element. Mosca [Mos99] showed that one can learn in quantum polynomial time the structure of any black-box abelian group. Such a task is known to be hard classically. Then Watrous [Wat01] pioneered the study of black-box group properties in the quantum context.

*A preliminary version of this paper appeared in *Proceedings of 32nd International Colloquium on Automata, Languages and Programming*, Lecture Notes in Computer Science, pages 1312-1324, Springer-Verlag, Berlin, 2005.

†CNRS-LRI, France. Email : magniez@lri.fr. Address: LRI - bâtiment 490, Université Paris-Sud, 91405 Orsay cedex, France. Partially supported by the EU 5th framework program RESQ IST-2001-37559, and by ACI Cryptologie CR/02 02 0040 and ACI Sécurité Informatique 03 511 grants of the French Research Ministry. Part of the research was done while visiting Perimeter Institute at Waterloo, ON, Canada.

‡University of Waterloo and Perimeter Institute for Theoretical Physics, Canada. Email: anayak@math.uwaterloo.ca. Address: Department of Combinatorics and Optimization and Institute for Quantum Computing, University of Waterloo, 200 University Ave. W., Waterloo, Ontario N2L 3G1, Canada. Research supported in part by NSERC, CIFAR, MITACS, CFI, OIT (Canada) and ARDA (USA). Research at Perimeter Institute for Theoretical Physics is supported in part by the Government of Canada through NSERC and by the Province of Ontario through MRI.

In this context, we study the problem of testing the commutativity of a black-box group (GROUP COMMUTATIVITY) given by its generators. The classical complexity of this problem was first considered by Pak [Pak00]. The straightforward algorithm for the problem has complexity $O(k^2)$, where k is the number of generators, since it suffices to check if every pair of generators commute. Pak presented a surprising randomized algorithm whose complexity is linear in k, and also showed that the deterministic lower bound is quadratic. The linear upper bound on complexity may also be obtained by applying quantum search [Gro96] to locate a pair of generators that do not commute. Using the quantization of random walks by Szegedy [Sze04], we instead present a *sublinear* algorithm with time and query complexity in $\tilde{O}(k^{2/3})$ (**Theorem 3**), where the $\tilde{O}$ notation means that logarithmic multiplicative factors are omitted.

GROUP COMMUTATIVITY bears a deceptive resemblance to ELEMENT DISTINCTNESS. The aim in the former is to detect the presence of a pair of generators which collide in the sense that they do not commute. However, since the group structure is unknown, whether or not a pair of generators collide can only be determined by invoking the group oracle. Moreover, the group oracle provides access to elements from the entire group spanned by the given generators, which may be used towards establishing commutativity. These differences necessitate the use of ideas from Pak's algorithm, the theory of rapidly mixing Markov chains, and perhaps most remarkably, the Szegedy quantization of walks.

GROUP COMMUTATIVITY appears to be the first natural problem for which the approach of Szegedy has no equivalent using other known techniques for constructing quantum algorithms, such as Grover search [Gro96], or the type of quantum walk introduced by Ambainis [Amb04]. Conversely, for Triangle Finding, the approach of Ambainis was more successfully applied. For this problem, Magniez, Szegedy and Santha [MSS05] construct a quantum algorithm that uses recursively two quantum walks *à la* Ref. [Amb04], while the Szegedy quantization of walks seems to give a less query-efficient algorithm. The problems of GROUP COMMUTATIVITY and TRIANGLE FINDING thus give strong evidence that the walks due to Ambainis are not comparable with the ones due to Szegedy.

A recent result of Buhrman and Špalek [Bv06] on matrix product verification also relies on the Szegedy quantization for its worst case *time* complexity. However, for the worst case instances, when there is at most one erroneous entry, the approach of Ambainis gives an algorithm whose query complexity is the same as that due to Szegedy.

We also prove that our algorithm is almost optimal by giving an $\Omega(k^{2/3})$ lower bound for the quantum query complexity of GROUP COMMUTATIVITY (**Theorem 4**). Simultaneously, we give an $\Omega(k)$ lower bound for its randomized query complexity (**Theorem 4**). This lower bound shows that the algorithm of Pak [Pak00] is optimal, and to our knowledge is new. We prove the lower bounds using a reduction from the problem of detecting a unique collision pair of a function, which is a special case of ELEMENT DISTINCTNESS.

2 Preliminaries

2.1 Black-box groups

We suppose that the elements of the group G are encoded by binary strings of length n for some fixed integer n, which we call the *encoding length*. The groups are given by generators, and therefore the *input size* of a group is the product of the encoding length and the number of generators. For simplicity, we also assume that the identity element of the group is given. Note that the encoding of group elements need not be unique, i.e., a single group element may be represented by several strings. If the encoding is not unique, one also needs an oracle for identity tests. Unless otherwise specified, we assume that the encoding is unique in this paper. All of our results apply when the encoding is not unique if one is given an oracle for identity tests.

Since we deal with black-box groups we shall shortly describe them in the framework of quantum computing (see also Refs. [Mos99] or [Wat01]). For a general introduction to quantum computing the reader might consult Refs. [NC00, KSV02]. We work in the quantum circuit model. For a group G of encoding length n, the black-box is given by two oracles O_G and its inverse O_G^{-1}, both operating on $2n$ qubits. For any group elements $g, h \in G$, the effect of the oracles is the following: $O_G|g\rangle|h\rangle = |g\rangle|gh\rangle$ and $O_G^{-1}|g\rangle|h\rangle = |g\rangle|g^{-1}h\rangle$. In this notation we implicitly use the encoding of a group element. We do that everywhere in the paper when there is no ambiguity. Not every binary string of length n necessarily corresponds to a group element. In this case the behaviour of the black-box can be arbitrary.

2.2 Query model

The quantum query model was explicitly introduced by Beals, Buhrman, Cleve, Mosca, and de Wolf [BBC+01]. In this model, as in its classical counterpart, we pay for accessing the oracle, but unlike the classical case, the machine can use the power of quantum parallelism to make queries in superposition.

The state of the computation is represented by three registers, the query register g, the answer register h, and the work register z. The computation takes place in the vector space spanned by all basis states $|g, h, z\rangle$. In the *quantum model* the state of the computation is a complex combination of all basis states which has unit length in the ℓ_2 norm.

For a black-box group the query operator is O_G together with its inverse O_G^{-1}. For oracle function $F : X \to Y$ the query operator is $O_F : |g\rangle|h\rangle \mapsto |g\rangle|h \oplus F(g)\rangle$, where $\oplus$ denotes the bitwise xor operation.

Non-query operations are independent of the oracle. A *k-query algorithm* is a sequence of $(k+1)$ operations $(U_0, U_1, \ldots, U_k)$ where each U_i is unitary. Initially the state of the computation is set to some fixed value $|\bar{0}, \bar{0}, \bar{0}\rangle$. In case of an oracle function, the sequence of operations $U_0, O_F, U_1, O_F, \ldots, U_{k-1}, O_F, U_k$ is applied. For black-box groups, the modified sequence of operations $U_0, O_G^{b_1}, U_1, O_G^{b_2}, \ldots, U_{k-1}, O_G^{b_k}, U_k$ is applied, where $b_i \in \{\pm 1\}$. Finally, one or more qubits designated as output bits are measured to get the outcome of the computation. The quantum algorithms we consider have a probabilistic outcome, and they might give an erroneous answer with non-zero probability. However, the probability of making an error is bounded by some fixed constant $\gamma < 1/2$.

In the query model of computation each query adds one to the *query complexity* of an algorithm, but all other computations are free. The *time complexity* of the algorithm is usually measured in terms of the total circuit size for the unitary operations U_i. We however take a more coarse-grained view of time complexity, and assume that operations such as accessing qubits containing group encodings or updating them, take unit time.

2.3 Quantum walks

We state a simple version of the recent result of Szegedy [Sze04]. Let P be an irreducible (i.e., strongly connected), aperiodic (i.e., non-bipartite), and symmetric Markov chain on a graph $G = (V, E)$ on N vertices. Such a walk is necessarily ergodic, i.e., converges to a unique stationary distribution regardless of the initial state.

Let $P[u, v]$ denote the transition probability from u to v. Let M be a set of marked nodes of V. Assume, one is given a database D that associates some data $D(v)$ to every node $v \in V$. From $D(v)$ we would like to determine if $v \in M$. We expedite this search using a quantum procedure Φ. When operating with D three types of cost are incurred. The cost might denote any measure of complexity such as query or time complexities.

Setup cost S**:** The cost to set up $D(v)$ for a $v \in V$.

Update cost U**:** The cost to update $D(v)$ for a $v \in V$, i.e., moving from $D(v)$ to $D(v')$, where the transition from v to v' is allowed by the Markov chain P.

Checking cost C**:** For $v \in V$, the complexity of checking if $v \in M$ from $D(v)$.

Concerning the quantization of the walk P, one needs to consider the quantum time complexity of its implementation in terms of the following parameters:

Initialization time I: The time complexity for constructing the superposition

$$\frac{1}{\sqrt{N}} \sum_{u,v} \sqrt{P[u, v]}|u, v\rangle.$$

Transition time T**:** The time complexity of realizing the transformation

$$|u, v\rangle \;\; \mapsto \;\; 2\sqrt{P[u, v]} \sum_{v'} \sqrt{P[u, v']}|u, v'\rangle - |u, v\rangle.$$

The Markov chains we construct in this paper are all random walks on regular graphs. For every node u, the probabilities $P[u, v]$ are all equal to $1/d$ or 0, where d is the degree of each node in the graph. The unitary transformation defined above, restricted to the node u in the first register, then corresponds to the Grover diffusion operator [Gro96] on the neighbours of u. The diffusion operator is the unitary matrix $\frac{2}{d}\mathbb{J} - \mathbb{I}$.

In the following theorem, which is the main result of Ref. [Sze04], the notation $O(\cdot)$ denotes the existence of a universal constant so that the expression is an upper bound.

Theorem 1 (Szegedy [Sze04]). *Let δ be the eigenvalue gap of P, and let $\frac{|M|}{|V|} \geq \varepsilon > 0$ whenever M is non-empty. There exists a quantum algorithm that determines if M is non-empty with cost $\mathsf{S}+O((\mathsf{U}+\mathsf{C})/\sqrt{\delta\varepsilon})$, and an additional time complexity of $\mathsf{I}+O(\mathsf{T}/\sqrt{\delta\varepsilon})$.*

Note that in this theorem, when the cost denotes the time complexity, we need to add the additional time complexity term to it.

Szegedy's theorem thus gives us a recipe for constructing and characterizing the behaviour of a quantum walk algorithm by specifying a classical random walk, and analysing its spectral gap and stationary distribution.

2.4 Spectral gap of Markov chains

The spectral gap (or eigenvalue gap) of a Markov chain (with non-negative eigenvalues) is the difference between the largest and the second largest eigenvalue of the probability transition matrix that represents it. Estimating this quantity directly from a description of the matrix is often very difficult. We take an indirect route to estimating this quantity by appealing to its relation with the convergence properties of the Markov chain.

Consider an ergodic Markov chain on state space X with stationary distribution π. Let P_x^t be the probability distribution on X obtained by performing t steps of the Markov chain starting at x. Let $\Delta(t)$ be the maximum over all starting states $x \in X$ of the total variation distance $\|P_x^t - \pi\|$. Then the *mixing time* τ of the Markov chain is defined as the smallest t such that $\Delta(t') \leq \frac{1}{2\mathrm{e}}$ for all $t' \geq t$.

A *coupling* for a Markov chain is a stochastic process on pairs of states (U_t, V_t) such that U_t and V_t, viewed marginally, each evolve according to the Markov chain, and if $U_t = V_t$, then $U_{t+1} = V_{t+1}$. The *coupling time* T is the maximum expected time (over all pairs of initial states (u, v)) for the states U_t, V_t to coincide:

$$T = \max_{u,v} \mathrm{E}[\mathrm{argmin}_t\{U_t = V_t, U_0 = u, V_0 = v\}].$$

We use the following facts about the mixing of Markov chains:

1. [Sin93, Proposition 2.2, Chapter 2] For walks with only non-negative eigenvalues, $\lambda^t \leq \Delta(t) \cdot (\min_u \pi(u))^{-1}$, where λ is the second largest eigenvalue. This bounds the second largest eigenvalue in terms of the total variation distance.

2. (see e.g., Ref. [Ald82]) $\Delta(t) \leq 2\exp(-\lfloor \frac{t}{\tau} \rfloor)$. This relates the total variation distance at any time t to the mixing time τ.

3. [Gri78] $\tau \leq 2\mathrm{e}T$. This bounds the mixing time τ in terms of the coupling time T.

Combining all three relations, we may deduce the following relationship between the spectral gap of a Markov chain and coupling time.

Corollary 1. *For any ergodic Markov chain with only non-negative eigenvalues, the spectral gap $1-\lambda \geq \frac{1}{4\mathrm{e}T}$, where λ is the second largest eigenvalue, and T is the coupling time for any valid coupling defined on $X \times X$.*

Proof. Chaining all three facts listed above, taking t-th roots, and letting $t \to \infty$, we see that

$$\begin{aligned} \lambda &\leq \exp(-\frac{1}{2\mathrm{e}T}) \\ &\leq 1 - \frac{1}{4\mathrm{e}T}, \end{aligned}$$

which is equivalent to the claim. □

2.5 The problems

Here we define the problems we are dealing with. The focus of the paper is on

GROUP COMMUTATIVITY
Oracle: Group operations O_G and O_G^{-1} for an encoding in $\{0,1\}^n$
Input: The value of n and the encoding of generators $g_1, \ldots, g_k$ of G
Output: `Yes` if G is commutative, and `No` otherwise (if there are two indices i, j such that $g_i g_j \neq g_j g_i$)

The next problem is a special instance of a well-studied problem, ELEMENT DISTINCTNESS.

UNIQUE COLLISION
Oracle: A function F from $\{1, \ldots, k\}$ to $\{1, \ldots, k\}$
Input: The value of k
Output: `Yes` if there exists a unique collision pair $x \neq y \in \{1, \ldots, k\}$ such that $F(x) = F(y)$, and `No` if the function is a permutation

This is a promise problem (or a relation) since we do not require a definite output for certain valid oracle functions. We also use a further specialization of the problem when k is even, UNIQUE SPLIT COLLISION, where, in the `Yes` instances, one element of the colliding pair has to come from $\{1, \ldots, k/2\}$ and the other from $\{k/2+1, \ldots, k\}$. We call this a *split* collision. Note that in the positive instances of this problem, the restriction of the function to the two intervals $\{1, \ldots, k/2\}$ and $\{k/2+1, \ldots, k\}$ is injective.

A beautiful application of the polynomial method gives us the optimal query complexity of UNIQUE COLLISION.

Theorem 2 ([AS04, Kut05, Amb05])**.** *The quantum query complexity of* UNIQUE COLLISION *is* $\Omega(k^{2/3})$.

The original results of the works cited above refer to the more general problem ELEMENT DISTINCTNESS, which requires the detection of one or more colliding pairs. This was proven by a randomized reduction from the problem COLLISION which distinguishes between a bijection and a two-to-one function. However, the reduction is still valid for the special case we consider. The reason is that the randomized reduction from COLLISION results in instances of UNIQUE COLLISION with constant probability.

3 A quantum algorithm for GROUP COMMUTATIVITY

We are given a black-box group G with generators $g_1, \ldots, g_k$. The problem is to decide if G is abelian. For technical reasons (see the proof of Lemma 1), and without loss of generality, we assume that g_1 is the identity element.

We denote by S_l the set of all l-tuples of distinct elements of $\{1, \ldots, k\}$. For any $u = (u_1, \ldots, u_l) \in S_l$, we denote by g_u the group element $g_{u_1} \ldots g_{u_l}$. Not all group elements are generated by such products of l generators. However, the subset of group elements we get this way has properties analogous to the entire group (see Lemma 2 below).

Our algorithm is based on the quantization of a random walk on $S_l \times S_l = S_l^2$. We adapt an approach due to Pak, for which we generalize Lemma 1.3 of Ref. [Pak00] to random elements from S_l. Then we show how to walk on S_l^2 for finding a non-commutative element in G, if there is any. We conclude using Theorem 1.

In this section, we let $p = \frac{l(l-1)+(k-l)(k-l-1)}{k(k-1)}$. Observe that when $k = 2l$, then $p = \frac{l-1}{2l-1} \leq \frac{1}{2}$. Moreover, when $l = o(k)$, then $1 - p = \Theta(l/k)$.

Lemma 1. *Let $K \neq G$ be a subgroup of G. Then* $\Pr_{u \in S_l}[g_u \notin K] \geq \frac{1-p}{2}$.

Proof. First we fix a total order (equivalently, a permutation) σ of $\{1, \ldots, k\}$, and we denote by S_l^σ that subset of l-tuples in S_l which respect the total order σ. In other words, $u = (u_1, \ldots, u_l) \in S_l^\sigma$ iff $\sigma^{-1}(u_i) < \sigma^{-1}(u_{i+1})$ for all $1 \leq i < l$.

All the sets S_l^σ have the same size $\binom{n}{l}$. Any tuple u of distinct elements respects exactly $n!/l!$ permutations σ, and therefore occurs in exactly the same number of sets S_l^σ. Thus picking a uniformly random element from S_l is the same as first picking a uniformly random permutation σ, and then picking a random element $u \in S_l^\sigma$. Consequently,

it is enough to prove the theorem for any fixed order σ. The reader may find it helpful to take σ to be the identity permutation to understand the idea behind the proof.

Let i be the smallest index for which $g_{\sigma(i)} \notin K$. Such an i exists since $K \neq G$. Recall that g_1 is the identity element.

Fix an ordered l-tuple u such that $\sigma(i) \notin u$ and $1 \in u$. We denote by v the ordered l-tuple where 1 has been deleted from u, and $\sigma(i)$ has been inserted into it at the appropriate position (that respects the total order). Formally, if $u = (u_1, \ldots, u_m, u_{m+1}, \ldots, u_l)$ such that $\sigma^{-1}(u_m) < i < \sigma^{-1}(u_{m+1})$, then v is obtained by deleting 1 from the $(l+1)$-tuple $(u_1, \ldots, u_m, \sigma(i), u_{m+1}, \ldots, u_l)$. This mapping defines a bijection (a perfect matching) between tuples u such that $\sigma(i) \notin u$ and $1 \in u$, and tuples v such that $\sigma(i) \in v$ and $1 \notin v$. Below we show that for every matched pair of tuples u, v, at least one of the group elements g_u, g_v is not in K.

Consider a matched pair u, v as above. Let $a = g_{u_1} g_{u_2} \cdots g_{u_m}$, and $b = g_{u_{m+1}} \cdots g_{u_l}$. Then $g_u = ab$ and $g_v = a g_{\sigma(i)} b$. Note that because of the choice of i, the group element $a \in K$. If $g_u = ab \in K$, then $b = a^{-1} g_u \in K$ as well. This means that $g_v = a g_{\sigma(i)} b \notin K$, since otherwise, we would have $g_{\sigma(i)} = a^{-1} g_v b^{-1} \in K$. Thus, both g_u and g_v cannot be in K. Therefore

$$\Pr_{u \in S_l^\sigma}[g_u \in K | \sigma(i) \in u \text{ xor } 1 \in u] \;\leq\; \frac{1}{2}.$$

Since for any two indices i, j,

$$\Pr_{u \in S_l^\sigma}[i, j \in u \text{ or } i, j \notin u] \;=\; p \;=\; \frac{l(l-1) + (k-l)(k-l-1)}{k(k-1)},$$

we conclude that

$$\Pr_{u \in S_l^\sigma}[g_u \in K] \;\leq\; (1-p) \times \tfrac{1}{2} + p \times 1,$$

which is at most $(1+p)/2$. □

From Lemma 1, we can generalize Lemma 1.1 of Ref. [Pak00]. For this, we recall the notions of the centre of a group, and the centralizer of a group element. The centralizer $C(g)$ of a group element $g \in G$ is the set of all group elements h that commute with g. This is a subgroup of G. The centre $C(G)$ of the group is the intersection of the centralizers of all groups elements. This is also a subgroup of G, and by definition, the elements of this subgroup commute with every element of the group G.

Lemma 2. *If G is non-commutative then* $\Pr_{u,v \in S_l}[g_u g_v \neq g_v g_u] \geq \frac{(1-p)^2}{4}$.

Proof. If G is non-commutative, then the centre $C(G)$ of G is a proper subgroup. With probability at least $(1-p)/2$, g_u does not belong to $C(G)$ for a random $u \in S_l$ (Lemma 1). We condition upon this event. Since $g_u \notin C(G)$, there is at least one element of G that does not commute with it. So the centralizer of g_u is also a proper subgroup of G. Again, by Lemma 1, the probability that for a random $v \in S_l$, g_v does not belong to the centralizer of g_u is also at least $(1-p)/2$. □

For $u \in S_l$, let t_u be the balanced binary tree with l leaves, whose leaves are from left to right the elements g_{u_i}, for $i = 1, \ldots, l$, and such that each internal node is the group product of its two successors. If l is not a power of 2, we put the deepest leaves to the left.

The random walk on S_l^2 that forms the basis of our quantum algorithm consists of two independent simultaneous walks on S_l. For a pair (u, v) of l-tuples, we maintain the binary trees t_u, t_v as described above as the data.

The random walk on S_l
Suppose the current state is $u \in S_l$.
With probability $1/2$ stay at u; with probability $1/2$, do the following:
– Pick a uniformly random position $i \in \{1, \ldots, l\}$, and a uniformly random index $j \in \{1, \ldots, k\}$.
– If $j = u_m$ for some m, then exchange u_i and u_m, else, set $u_i = j$.
– Update the tree t_u (using $O(\log l)$ group operations). This involves "uncomputing" all the products from the root to the leaf that is being updated, and then computing fresh products from the leaf to the root of the tree.

Lemma 3. *The spectral gap of the walk described above is at least* $\frac{c}{l \log l}$*, for a universal constant* $c \geq \frac{1}{8\mathrm{e}}$*, provided* $l \leq k/2$*.*

Proof. First, we show that the random walk mixes rapidly using a "coupling argument". Then, using a relation between mixing time and the second largest eigenvalue, we get a bound on the spectral gap.

Note that the walk is ergodic and has the uniform distribution on S_l as its stationary distribution π. Thus $\pi(u) = \frac{(k-l)!}{k!}$ for all u.

The eigenvalues of any stochastic matrix P, such as the transition matrix of a Markov chain, all lie in the interval $[-1, 1]$. Suppose we modify the chain by including self-loops at every state, i.e., remaining at the current state with probability $1/2$, and following the transition of the chain with probability $1/2$. Then the transition matrix becomes $(\mathbb{I} + P)/2$, where $\mathbb{I}$ is the identity matrix. The eigenvalues of this matrix lie in the interval $[0, 1]$. Because of such self-loops, all the eigenvalues of our walk above on S_l are non-negative.

In order to find a lower bound for the spectral gap of our random walk on S_l, we use Corollary 1. A coupling for which $T \leq l \log l$ is the obvious one: for any pair $u, v \in S_l$, follow one step of the random walk with the same choice of random position i and index j. This is clearly a valid coupling, since the marginal process on any one of the two tuples is the same as our walk, and if the two tuples are identical, they are modified identically by the walk.

Let d be the Hamming distance between the two tuples u, v. This distance never increases during the coupling process described above. Moreover, in one step of the process, the distance goes down by 1 with probability at least $\frac{d}{2l}$. This is because with probability d/l, the position i is one where u and v are different, and with probability at least $(k-l)/k$, the index j is not one from the positions where u and v are the same. Since $l \leq k/2$, the net probability that the distance decreases by 1 is at least $d/2l$.

By a straightforward calculation, the expected time T for the distance to go to zero is at most $2l \log l$ (since $d \leq l$). Using the relation between λ and T derived in Corollary 1, we get our bound on the spectral gap. □

Theorem 3. *There is a quantum algorithm that solves* GROUP COMMUTATIVITY *with* $O(k^{2/3} \log k)$ *queries and time complexity* $O(k^{2/3} \log^2 k)$*.*

Proof. Our algorithm derived from an application of Theorem 1 to the product of two independent walks on S_l. The database associated with a tuple $u \in S_l$ is the binary tree t_u. Due to the Szegedy Theorem, we need only compute the eigenvalue gap δ of the random walk and the fraction of marked states ε in the uniform distribution on S_l^2.

The stationary distribution for the walk is the uniform distribution on $S_l \times S_l$. So, from Lemma 2 above, the fraction of marked states ε is at least $(1-p)^2/4$. The spectral gap δ for the walk is the same as that on S_l, i.e., $\delta \geq c/(l \log l)$, from Lemma 3.

We start with a uniform distribution over $|u, t_u\rangle|v, t_v\rangle$, where $u, v \in S_l$. The setup cost is at most $2(l-1)$ and the updating cost of the walk is $O(\log l)$. We choose $l = o(k)$ so that $1 - p = \Theta(l/k)$. By Theorem 1, the total query cost is

$$\begin{aligned} & 2(l-1) + O\left(\frac{1}{\sqrt{\delta\varepsilon}} \log l\right) \\ = \quad & 2(l-1) + O\left(\frac{1}{1-p}\sqrt{l \log l} \cdot \log l\right) \\ = \quad & 2(l-1) + O\left(\frac{k}{\sqrt{l}} \log^{3/2} l\right). \end{aligned}$$

This expression is minimized when $l = k^{2/3} \log k$, and the cost is $O(k^{2/3} \log k)$.

The time complexity overhead comes from the initialization and transition times that are both essentially equal to the time complexity of performing a Grover diffusion operation (see Section 2.3). For the initialization, we use a diffusion over S_l^2, whose time complexity is $O(\log(|S_l|^2)) = O(l \log k)$. For the transition, we use a diffusion over a set of size 2 tensor product with a diffusion over a set of size kl, therefore the corresponding time complexity is $O(\log(kl)) = O(\log k)$. □

4 Reduction from UNIQUE SPLIT COLLISION

We begin our presentation of the lower bound by considering the complexity of UNIQUE SPLIT COLLISION. This problem is at least as hard as UNIQUE COLLISION in its query complexity since any bounded-error algorithm for the former can be used to detect an arbitrary collision.

Proposition 1. *The randomized and the quantum query complexity of* UNIQUE SPLIT COLLISION *is respectively* $\Omega(k)$ *and* $\Omega(k^{2/3})$.

Proof. One can prove the $\Omega(k)$ lower bound for classical query complexity by an adversary argument.

For the quantum case, we reduce UNIQUE COLLISION to UNIQUE SPLIT COLLISION by composing the oracle function with a random permutation. Then Theorem 2 (due to Aaronson and Shi [AS04] and Kutin [Kut05], together with Ambainis [Amb05]) implies the lower bound.

Assume that we have an algorithm A for UNIQUE SPLIT COLLISION with constant bounded error $\gamma < 1/4$. We run A on oracle F composed with a random permutation on the domain. If there is a collision in the function F, with probability at least $1/2$, the colliding pair will have one point on either side of $k/2$. This will be detected with probability at least $1 - \gamma$. The overall success probability will be $(1 - \gamma)/2 \geq 3/8$. If there is no collision, the algorithm will make an error with probability at most $\gamma < 1/4$. Using standard techniques, this gap in acceptance probability can be made symmetric around $1/2$ and boosted by repeating the experiment with an independent run of the algorithm A on F composed with a fresh random permutation. For completeness, we include the argument below.

Our final algorithm for UNIQUE COLLISION picks two random permutations, and runs A on the oracle function composed with these permutations. It accepts if any one of the two executions of A accepts. We now show that the error of our algorithm is now upper bounded by $1/4 + \gamma < 1/2$.

If the oracle function F has no collision, the error is upper bounded by $2\gamma < 1/4+\gamma$. If F has a collision, then with probability at most $(1/2)^2 = 1/4$, the colliding pair has no point on either side of $k/2$ for both the randomly chosen permutations. Assume this is not the case, and fix a permutation for which the permuted F has a split collision. Our algorithm accepts this permutation of F with probability at least $1 - \gamma$. Therefore the overall error is upper bounded by $1/4 + \gamma$.

In reducing a positive instance of UNIQUE COLLISION, with probability close to $1/4$, we get inputs for which the algorithm for UNIQUE SPLIT COLLISION need not output a definite answer with probability bounded away from $1/2$. (These are inputs where the colliding pair of indices are both at most $k/2$ or both greater than $k/2$.) Our argument is valid in spite of this, since the acceptance probability in the other case is high. □

We conclude by proving the same lower bound for GROUP COMMUTATIVITY as well. We thus show that the algorithm described in the previous section is almost optimal.

The group involved in the proof of the lower bound is a subgroup G of $U(4k)$, the group under matrix multiplication of $4k \times 4k$ unitary matrices. The generators of G are block diagonal, each with $2k$ blocks of dimension 2×2. Each block is one of the following Pauli matrices:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad Y = XZ = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.$$

No pair of matrices amongst X, Y and Z commute. An encoding of the group G consists in words $\sigma_1 \ldots \sigma_{2k}$ of length $2k$ over the alphabet $\{I, X, Y, Z\}$ together with a sign vector $s = (s_1, s_2, \ldots, s_{2k})$ in $\{+1, -1\}^{2k}$. A tuple $(s, \sigma_1, \ldots \sigma_{2k})$ represents the matrix $\mathrm{diag}(s_1\sigma_1, \ldots, s_{2k}\sigma_{2k})$. We call this encoding the *explicit encoding*.

Theorem 4. *The randomized and the quantum query complexity of* GROUP COMMUTATIVITY *are respectively* $\Omega(k)$ *and* $\Omega(k^{2/3})$.

Proof. We prove the theorem by reducing UNIQUE SPLIT COLLISION to GROUP COMMUTATIVITY. First, we define a group that is non-commutative if and only if the oracle input F for UNIQUE SPLIT COLLISION has a collision. Second, we design a unique encoding of the group elements such that each group operation can be simulated with at most four queries to F. We then conclude our theorem using Proposition 1.

For $i, j \in \{1, 2, \ldots, k\}$, we define $a_{ij}, b_{ij} \in U(2k)$ of the form described above. Both kinds of matrix have the identity matrix in all their blocks except for the i-th and $(j+k)$-th. The i-th block is Y in both a_{ij} and b_{ij}. The $(j+k)$-th block is Z in a_{ij} and X in b_{ij}.

Suppose the oracle for the problem UNIQUE SPLIT COLLISION computes the function $F : \{1, \ldots, k\} \to \{1, \ldots, k\}$. We associate a generator g_i of the type described above with each element i in the domain of F. The generator g_i is $a_{iF(i)}$ if $i \leq k/2$, and it is $b_{iF(i)}$ if $i > k/2$.

Observe that all the generators g_i are distinct, since the i-th block in it is Y, and the rest of the initial k blocks are the identity matrix. This is designed so that we can identify the index i from the explicit encoding of a generator.

For any two distinct points i_1, i_2, if $F(i_1) \neq F(i_2)$, then the generators g_{i_1}, g_{i_2} have distinct blocks $F(i_1) + k$ and $F(i_2) + k$ set to either Z or X. So if the function F is injective, the set of generators $\{g_i\}$ consists of k distinct commuting elements. If there is a collision i_1, i_2 in F with one point on either side of $k/2$, then the same block $F(i_1) + k = F(i_2) + k$ is set to Z and X respectively. Then the generators g_{i_1} and g_{i_2} do not commute, and the group generated by $\{g_i\}$ is non-abelian.

The encoding of the group elements is the explicit encoding defined above except for the generators g_i. The generators g_i are encoded by their corresponding indices i. The input to GROUP COMMUTATIVITY is $1, 2, \ldots, k$.

Now we explain how to simulate the group operations. When an integer i is involved in a group operation, we query the oracle for F at i and construct g_i as defined above. One more query to F is required to erase the value of the function. Otherwise we do not query F. Matrix operations can be performed without incurring any further calls to F.

We also have to take care to output the result of a group operation in the correct encoding. Namely, when the output of a group operation is either a_{ij} or b_{ij}, for some i, j, then we check if it can be encoded by i using one query to F, and one more query to erase the value of the function.

Thus a group operation involves F at most six times, when both of the elements are encoded by integers. Note that a product of two group elements of the form a_{ij} or b_{ij} can never result in another such generator. We can thus improve the number of invocations of the function oracle from six to four per group operation. □

5 Acknowledgements

We thank the anonymous referees for ICALP'05 and *Algorithmica* for their helpful feedback on the paper.

References

[Ald82] David Aldous. Random walks on finite groups and rapidly mixing Markov chains. In *Séminaire de Probabilités XVII*, volume 986 of *Lecture Notes in Mathematics*, pages 243–297. Springer-Verlag, 1981–82.

[Amb04] Andris Ambainis. Quantum walk algorithm for Element Distinctness. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 22–31, 2004.

[Amb05] Andris Ambainis. Polynomial degree and lower bounds in quantum complexity: Collision and Element Distinctness with small range. *Theory of Computing*, 1(3):37–46, 2005.

[AS04] Scott Aaronson and Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. *Journal of the ACM*, 51(4):595–605, 2004.

[BBC+01] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4):778–797, 2001.

[BS84] László Babai and Endre Szemerédi. On the complexity of matrix group problems I. In *Proceedings of the 25th Annual IEEE Symposium on Foundations of Computer Science*, pages 229–240, 1984.

[Bv06] Harry Buhrman and Robert Špalek. Quantum verification of matrix products. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 880–889, 2006.

[Gri78] David S. Griffeath. Coupling methods for Markov processes. In Gian-Carlo Rota, editor, *Studies in Probability and Ergodic Theory*, pages 1–43. Academic Press, New York, NY, USA, 1978.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 212–219, 1996.

[KSV02] Alexei Yu. Kitaev, Alexander H. Shen, and Mikhail N. Vyalyi. *Classical and Quantum Computation*, volume 47 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2002.

[Kut05] Samuel Kutin. Quantum lower bound for the collision problem with small range. *Theory of Computing*, 1(2):29–36, 2005.

[Mos99] Michele Mosca. *Quantum Computer Algorithms*. PhD thesis, University of Oxford, 1999.

[MSS05] Frédéric Magniez, Miklos Santha, and Mario Szegedy. Quantum algorithms for the triangle problem. In *Proceedings of 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1109–1117, 2005.

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, UK, 2000.

[Pak00] Igor Pak. Testing commutativity of a group and the power of randomization. Electronic version at `http://www-math.mit.edu/~pak/research.html`, 2000.

[Sin93] Alistair Sinclair. *Algorithms for Random Generation and Counting: A Markov Chain Approach*. Progress in theoretical computer science. Birkhäuser, Boston, 1993.

[Sze04] Mario Szegedy. Quantum speed-up of Markov chain based algorithms. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 32–41, 2004.

[Wat01] John Watrous. Quantum algorithms for solvable groups. In *Proceedings of 33rd Annual ACM Symposium on Theory of Computing*, pages 60–67, 2001.