Quantum Complexity of Testing Group Commutativity*

Frédéric Magniez[1] and Ashwin Nayak[2]

[1] CNRS–LRI, UMR 8623 Université Paris–Sud, France
[2] University of Waterloo and Perimeter Institute for Theoretical Physics, Canada

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 introduce a new technique of reduction for quantum query complexity. 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 [1] around search problems in unstructured, structured, or partially structured databases has recently seen an extraordinary expansion. In contrast to problems based on Hidden Subgroup Problem (HSP) (see for instance Ref. [2]), 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 modeling access to the input. 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 the HSP. On the other hand, for total functions, deterministic and quantum query complexities are polynomially related [3].

In the 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 [4]. 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

* For their research support, F.M. thanks the EU 5th framework program RESQ and the French Research Ministry, and A.N. thanks Canada's NSERC and CIAR.

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 [2] 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 [5] pioneered the study of black-box group properties in the quantum context.

In this context, we study the problem of testing commutativity of a black-box group (Group Commutativity) given by its generators. The classical complexity of this problem was first considered by Pak [6]. 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 [1] to locate a pair of generators that do not commute. Using the quantization of random walks by Szegedy [7], we instead present a *sublinear* algorithm 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—Grover search [1], or the type of quantum walk introduced by Ambainis [8]. A recent result of Buhrman and Spalek [9] on matrix product verification is in the same situation for its time complexity but not for the query complexity, since the approach of Ambainis gives an algorithm whose query complexity is the same in the worst case.

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 6**). Simultaneously, we give an $\Omega(k)$ lower bound for its randomized query complexity (**Theorem 5**). This lower bound shows that the algorithm of Pak [6] is optimal, and to our knowledge is new. We first state an easier lower bound using a simple reduction from the problem of detecting a unique collision pair of a function, which is a special case of Element Distinctness, when one allows non-unique encoding of the black-box group (**Theorem 4**). For the lower bound for uniquely encoded black-box groups, the proof gets more complex. The randomized case relies upon an adversary argument. The quantum case is subtle. We show the said lower bound for the number of accesses to the given generators. The lower bound also holds for the number of group operations in *generic* quantum algorithms (see Section 4 for a definition). This is shown using a new kind of reduction based on approximation degree of the problem.

2 Preliminaries

2.1 Black-box groups

We will 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 will be 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, 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 will deal with black-box groups we shall shortly describe them in the framework of quantum computing (see also [2] or [5]). For a general introduction to quantum computing the reader might consult [10, 11]. We will work in the quantum circuit model. For a group G of encoding length n, the black-box will be 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 will do that everywhere in the paper when there is no ambiguity. Also, not every binary string of length n necessarily corresponds to a group element. In this case, the behavior 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 [3]. 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 will be 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 $b_i = \pm 1$. Finally, one or more qubits designated as output bits are measured to get the outcome of the computation. The quantum algorithms we consider might give an erroneous answer, but 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 will 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 [7]. Let P be an ergodic and symmetric Markov chain on a graph $G = (V, E)$ on N vertices. We denote by $P[u, v]$ 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 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: The query complexity of $\Phi(D(v))$ for a $v \in V$.

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

In the following theorem, the notation $O(\cdot)$ denotes the existence of a universal constant so that the expression is an upper bound. We now state the main result of [7].

Theorem 1 (Szegedy [7]). *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 $\mathsf{I} + O(\mathsf{T}/\sqrt{\delta\varepsilon})$.*

Moreover, if P is state transitive then the cost of finding a marked element of M is the same as above.

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

2.4 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 will also use a further specialization of the problem when k is even, UNIQUE SPLIT COLLISION, where 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.

2.5 Approximation degree

We describe the notion of approximation degree for oracle decision problems. Let $\mathcal{S}$ be the set of functions from $\{1, \ldots, k\}$ to $\{1, \ldots, k\}$. An *oracle decision problem* is a boolean function on the set $\mathcal{S}$. For every function $F \in \mathcal{S}$, we define the variables x_{ij} which are 1 if $F(i) = j$ and 0 otherwise.

Definition 1 ([3, 12]). *Let $\Phi : \mathcal{S} \to \{0,1\}$ be an* oracle decision problem*. Then the* approximation degree *of Φ is the lowest degree of real multivariate polynomials P in variables x_{ij}, such that $|P(x) - \Phi(F)| \leq 1/3$, for every $F \in \mathcal{S}$*

The following powerful result relates approximation degree to quantum query complexity.

Proposition 1 ([3, 12]). *If the quantum query complexity of Φ is T, then the approximation degree of Φ is at most $2T$.*

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

Theorem 2 ([12–14]). *The approximation degree of* Unique Collision, *and hence its quantum query complexity, is* $\Omega(k^{2/3})$.

The original result 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 detects between a bijection and a two-to-one function. However, the reduction is still valid for the special case we consider. As noticed by Ambainis [14], this reduction also implies the lower bound on the approximation degree.

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 $g_{u_1} \ldots g_{u_l}$. Our algorithm is based on the quantization of a random walk on S_l^2. We will also adapt an approach due to [6]. For this we generalize Lemma 1.3 of [6] for 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 will 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 $1 \leq i < l$. Since $(S_l^\sigma)_\sigma$ is an equitable partition of S_l, picking a random element from S_l is the same as first picking a random permutation σ, and then picking a random element $u \in S_l^\sigma$. Therefore 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.

We denote $g'_j = g_{\sigma(j)}$ for every $j = 1, \ldots, k$. Let i be the smallest index such that $g'_i \notin K$. Such an i exists since $K \neq G$. Let j be such that $g'_j = g_1$, the identity element.

Fix an ordered l-tuple u such that $g'_i \notin u$ and $g'_j \in u$. We denote by v the ordered l-tuple where g'_j has been deleted from u, and g'_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 g'_j from the $(l+1)$-tuple $(u_1, \ldots, u_m, \sigma(i), u_{m+1}, \ldots, u_l)$.

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 $g_v = a g'_i b$. Note that because of the choice of i, $a \in K$. If $g_u = ab \in K$ so that $b \in K$ as well, then $g_v \notin K$. Therefore $\Pr_{u \in S_l^\sigma}[g_u \in K | i \in u \text{ xor } j \in u] \leq \frac{1}{2}$.

Since $\Pr_{u \in S_l^\sigma}[i, j \in u \text{ or } i, j \notin u] = p$, we conclude that $\Pr_{u \in S_l^\sigma}[g_u \in K] \leq (1-p) \times \frac{1}{2} + p \times 1$. □

With the approach of [6] and from Lemma 1, we can generalize easily Lemma 1.1 of [6].

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). Conditioned upon this event, 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 $(1-p)/2$. □

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 will consist of two independent simultaneous walks on S_l. For a pair (u, v) of l-tuples, we will 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).

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 will show that the random walk mixes rapidly using a "coupling argument". Then, using a relation between mixing time and the second largest eigenvalue, we will 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. Moreover, because of the self-loops, all the eigenvalues of the walk are non-negative.

Let P_x^t be the probability distribution on S_l obtained by performing t steps of the walk starting at x. Let $\Delta(t)$ be the maximum over all starting states $x \in S_l$ of the total variation distance $\|P_x^t - \pi\|$. Let τ (the *mixing time*) be 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 will use the following facts about mixing of Markov chains:

1. [15, 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. [16]) $\Delta(t) \leq 2\exp(-\lfloor \frac{t}{\tau} \rfloor)$. This relates the total variation distance at any time t to the mixing time τ.
3. [17] $\tau \leq 2\mathrm{e}T$. This bounds the mixing time τ in terms of the coupling time T.

Combining all three relations, taking t-th roots, and letting $t \to \infty$, we see that $\lambda \leq \exp(-\frac{1}{2\mathrm{e}T}) \leq 1 - \frac{1}{4\mathrm{e}T}$. Thus, the spectral gap is $1 - \lambda \geq \frac{1}{4\mathrm{e}T}$.

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.

Let d be the hamming distance between the two tuples u, v. This distance never increases during the 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 described above between λ and T, we get our bound on the spectral gap. □

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

Proof. The walk is the above described walk on S_l^2. The database associated with a tuple $u \in S_l$ is the binary tree t_u. Using Szegedy's theorem 1, we need only compute the eigenvalue gap of the random walk and the initial success probability (in the uniform distribution).

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

Since 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 will choose $l = o(k)$ so that $1 - p = \Theta(l/k)$. The total query cost is then

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

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. 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. The proof is omitted due to the lack of space.

Proposition 2. *The approximation degree and the quantum query complexity of* Unique Split Collision *are both* $\Omega(k^{2/3})$*. The randomized query complexity of this problem is* $\Omega(k)$*.*

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 will be a subgroup G of $U(2k)$, the group (under matrix multiplication) of $2k \times 2k$ unitary matrices. The generators of G will be block diagonal, each with k blocks of dimension 2×2. Each block will be one of the following three (Pauli) matrices:

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

The group may also involve the remaining Pauli matrix $Y = XZ = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$. No pair of matrices amongst X, Y and Z commute. An encoding of the group consists in words $\sigma_1 \dots \sigma_k$ of length k over the alphabet $\{I, X, Y, Z\}$ together with a sign vector $s = (s_1, s_2, \dots, s_k)$ in $\{+1, -1\}^k$. A tuple $(s, \sigma_1, \dots \sigma_k)$ represents the matrix $\mathrm{diag}(s_1\sigma_1, \dots, s_k\sigma_k)$. We will call this encoding the *explicit encoding.*

Let a_j and b_j be generators that have the identity matrix in all their blocks except for the j-th. The j-th block is Z in a_j and X in b_j.

We describe a connection between Unique Split Collision and Group Commutativity. Suppose the oracle for the problem Unique Split Collision computes the function $F : \{1, \dots, k\} \to \{1, \dots, k\}$. We associate a generator g_i of the type described above with each element i in the domain. The generator g_i is $a_{F(i)}$ if $i \leq k/2$, and it is $b_{F(i)}$ if $i > k/2$. As long as the function F is injective on the two intervals $\{1, \dots, k/2\}, \{k/2 + 1, \dots, k\}$, the set of generators $\{g_i\}$ consists of k distinct elements. None of these generators is contained in the span of the remaining generators.

It is straightforward to check that there is a collision in F (with one point on either side of $k/2$) iff the group generated by $\{g_i\}$ is non-commutative. We use this connection for proving our lower bound. While the main result uses a non-standard method, we first prove a weaker result which explains the intuition behind the final proof to the reader.

Theorem 4. *If non-unique encoding of group elements is allowed, the randomized and the quantum query complexity of* Group Commutativity *are respectively* $\Omega(k)$ *and* $\Omega(k^{2/3})$*.*

Proof. Suppose we allow non unique encoding of the group G. We show that any algorithm A solving Group Commutativity may be adapted to solve Unique Split Collision, with at most four times the query complexity of A. We then conclude our theorem using Proposition 2.

We construct a black-box group which may invoke the oracle for Unique Split Collision to implement group operations. The encoding of the group elements will be either the explicit encoding defined above, or an element of $\{1, \ldots, k\}$. When the encoding is an integer $i \in \{1, \ldots, k\}$, it represents the generator g_i. 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. Group operations can be performed without incurring any further calls to F. Operations on previously computed products also do not cost any queries. Therefore a group operation involves F at most four times, when both of the elements are encoded by integers. The oracle hides the group G with this non-unique encoding, and the input is the sequence of encodings $1, 2, \ldots, k$. □

In the case of unique encoding of group elements by the black-box, the reduction above is not guaranteed to work. The reason is that non-trivial products of generators may evaluate to the value of a generator. These products are represented in explicit form, and therefore our simulation possibly uses two different representations for the generators. We can nevertheless modify our simulation to work, while maintaining essential properties of the algorithm. In the classical model, our simulation preserves the number of oracle queries. In the quantum model, our simulation will produce a polynomial that approximates Unique Split Collision and has degree of the order of the number of queries made by the commutativity algorithm.

In our arguments, we assume that the algorithm never queries the black-box with encodings that did not result from previous queries to the oracle. This family of algorithm are usually called *generic algorithms*. This notion was introduced to cryptography by Nechaev [18] and Shoup [19]. By suitably randomizing the encoding such as in [20], we can ensure that the probability that the algorithm chances upon a valid pair of encoded group elements is $o(1)$, if this input does not result from previous queries. If we choose n, the encoding length, to be $\Omega(\log |G|)$, this probability would be exponentially small in n. We can therefore assume in the black-box group setting that a correct algorithm is always generic, and we make this assumption until the end of this section. These arguments do not generalize easily to the quantum setting; we leave a proof to a more complete version of this paper. We start with the classical simulation.

Theorem 5. *With unique encoding of group elements, the randomized query complexity of* Group Commutativity *is* $\Omega(k)$.

Proof. Our reduction from Unique Split Collision works essentially because all the generators are distinct, and because no generator is contained in the span of the remaining $k - 1$ generators. We modify the simulation in the proof of Theorem 4 so that we record the value of the function F at any point which is queried. We retain the explicit encoding for all group elements except the generators as the encoding for the black-box group. The generators are represented by integers $1, \ldots, k$. A non-trivial product may equal a generator g_i only if the product contains this generator. The value of F at point i would necessarily have been queried for the algorithm to have computed this product. The index i can therefore be located by examining the record of all queries made thus far in the algorithm, and used to encode the generator. □

For the modified simulation in the quantum case, we introduce another *implicit* encoding of elements of G as tuples $(s, x_1, x_2, \ldots, x_k)$, where $s \in \{+1, -1\}^k$, and $x_i \in \{0, 1\}$. A word in this implicit encoding represents the group element $\mathrm{diag}(s_1 I, s_2 I, \ldots, s_k I) \cdot g_1^{x_1} g_2^{x_2} \cdots g_k^{x_k}$. This is a unique encoding of elements in G.

As in the proof of the classical lower bound, we restrict ourselves to generic algorithms. In a generic quantum black-box group algorithm, along every computational path, the queries to the oracle involve either a generator, or a product that was previously computed along that path.

Theorem 6. *With unique encoding of group elements, any generic quantum algorithm for* Group Commutativity *performs $\Omega(k^{2/3})$ group operations (queries to the group oracle).*

Proof. In using a generic algorithm for Group Commutativity to solve Unique Split Collision we now use the implicit encoding of group elements. The generators are specified in this notation, and we will maintain this representation for all the intermediate products that are computed during the algorithm. The query cost of simulating a group operation is no longer $O(1)$. Indeed, we may need up to $O(k)$ queries to F to implement a multiplication of two elements. Nevertheless, we argue that the degree of the polynomial that results from this simulation is of the order of the query cost of the commutativity algorithm.

We refine the proof of Proposition 1 due to [3, 12] to claim that after t queries:

- The amplitude of any basis state is a polynomial of degree at most $O(t)$ in the variables x_{ij}.
- Fix a classical basis state. If a generator g_i occurs in the implicit encoding of group elements stored in the special registers, then a variable x_{ij} (for some j) is a factor of the polynomial that gives the amplitude of that state.

This ensures that in our simulation, the degree of the polynomial corresponding to a basis state does not increase as we query F to implement a group operation involving a previously computed product. In making this claim, we rely on the fact that $x_{ij} \in \{0, 1\}$ for inputs of interest to us, so $x_{ij} x_{ij'} = \delta_{jj'} x_{ij}$. (The degree goes up by $O(1)$ when at least one of the operands is a fresh generator.)

As a consequence, we derive a polynomial of degree of the order of the number of queries made by the commutativity algorithm, and this approximates Unique Split Collision. We can thus conclude the same lower bound as in Proposition 2 for testing commutativity as well. □

Note that the complications due to the unique encoding requirement do not arise if we are concerned with the number of accesses to the input generators. For arbitrary (possibly non-generic) quantum algorithms, the same reduction also directly gives a bound on this notion of complexity.

Proposition 3. *The lower bound of $\Omega(k^{2/3})$ above also holds for the query complexity of any quantum algorithm, if the generators to a* possibly known *group are specified by an input oracle.*

References

1. Grover, L.: A fast quantum mechanical algorithm for database search. In: Proc. of 28th ACM STOC. (1996) 212–219
2. Mosca, M.: Quantum Computer Algorithms. PhD thesis, Univ. of Oxford (1999)
3. Beals, R., Buhrman, H., Cleve, R., Mosca, M., Wolf, R.: Quantum lower bounds by polynomials. J. of the ACM **48** (2001) 778–797
4. Babai, L., Szemerédi, E.: On the complexity of matrix group problems I. In: Proc. of 25th IEEE FOCS. (1984) 229–240
5. Watrous, J.: Quantum algorithms for solvable groups. In: Proceedings of 33rd Symposium on Theory of Computing, ACM (2001) 60–67
6. Pak, I.: Testing commutativity of a group and the power of randomization. Electronic version at `http://www-math.mit.edu/~pak/research.html` (2000)
7. Szegedy, M.: Quantum speed-up of markov chain based algorithms. In: Proc. of 45th IEEE FOCS. (2004) 32–41 Also arXiv.org report quant-ph/0401053.
8. Ambainis, A.: Quantum walk algorithm for Element Distinctness. In: Proceedings of 45th IEEE FOCS. (2004) 22–31
9. Buhrman, H., Spalek, R.: Quantum verification of matrix products. Technical Report quant-ph/0409035, arXiv archive (2004)
10. Nielsen, M., Chuang, I.: Quantum Computation and Quantum Information. Cambridge University Press (2000)
11. Kitaev, A., Shen, A., Vyalyi, M.: Classical and Quantum Computation. Volume 47 of Graduate Studies in Mathematics. AMS (2002)
12. Aaronson, S., Shi, Y.: Quantum lower bound for the collision problem. J. of the ACM **51** (2004) 595–605
13. Kutin, S.: A quantum lower bound for the collision problem. Technical Report quant-ph/0304162, arXiv archive (2003)
14. Ambainis, A.: Quantum lower bounds for collision and element distinctness with small range. Technical Report quant-ph/0305179, arXiv archive (2003)
15. Sinclair, A.: Algorithms for Random Generation and Counting: A Markov Chain Approach. Progress in theoretical computer science. Birkhäuser, Boston (1993)
16. Aldous, D.: Random walks on finite groups and rapidly mixing Markov chains. In: Séminaire de Probabilités XVII. Volume 986 of Lecture Notes in Mathematics., Springer-Verlag (1981–82) 243–297
17. Griffeath, D.: Coupling methods for Markov processes. In Rota, G.C., ed.: Studies in Probability and Ergodic Theory. Academic Press (1978) 1–43
18. Nechaev, V.: Complexity of a determinate algorithm for the discrete logarithm. Mathematical Notes **55** (1994) 165–172
19. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Proc. of Eurocrypt. (1997) 255–266
20. Schnorr, C., Jakobsson, M.: Security of signed ElGamal encryption. In: Proc. of 6th Asiacrypt. (2000) 73–89