An Exact Quantum Polynomial-Time Algorithm for Simon's Problem

Gilles Brassard *

Département IRO
Université de Montréal
C.P. 6128, succursale centre-ville
Montréal (Québec)
Canada H3C 3J7
Email: brassard@iro.umontreal.ca

Peter Høyer †

Dept. of Math. and Computer Science
Odense University
Campusvej 55
DK–5230 Odense M
Denmark
Email: u2pi@imada.ou.dk

Abstract

We investigate the power of quantum computers when they are required to return an answer that is guaranteed to be correct after a time that is upper-bounded by a polynomial in the worst case. We show that a natural generalization of Simon's problem can be solved in this way, whereas previous algorithms required quantum polynomial time in the expected sense only, without upper bounds on the worst-case running time. This is achieved by generalizing both Simon's and Grover's algorithms and combining them in a novel way. It follows that there is a decision problem that can be solved in exact quantum polynomial time, which would require expected exponential time on any classical bounded-error probabilistic computer if the data is supplied as a black box.

1 Introduction

According to the modern version of the Church–Turing thesis, anything that can be computed in polynomial time on a physically realisable device can be computed in polynomial time on a probabilistic Turing machine with bounded error probability. This belief has been seriously challenged by the theory of quantum computing. In particular, Simon [18] provided the first example of a problem that can be solved in polynomial time on a quantum computer, yet any classical bounded-error probabilistic algorithm would require exponential time if the data is supplied as a black box. However, Simon's algorithm is polynomial-time in the expected sense: there is no upper bound on how long

it may run on any given instance if it keeps being unlucky. (The same can be said about Shor's celebrated quantum factoring algorithm [17].)

In this paper, we address the issue of *exact* quantum polynomial time, which concerns problems that quantum computers can solve in guaranteed worst-case polynomial time with zero error probability. Note that this strong requirement would make randomness useless for classical machines: anything you can compute on a classical probabilistic computer with zero error probability in guaranteed worst-case polynomial time can be done in polynomial time by a *deterministic* computer—simply run the probabilistic algorithm with an arbitrarily fixed sequence of coin "tosses".

The study of exact quantum polynomial time is not new. The very first algorithm ever designed to demonstrate an advantage of quantum computers over classical computers, due to Deutsch and Jozsa [14], was of this exact nature. However, it solved a problem that could be handled just as efficiently with a classical probabilistic computer, provided an arbitrarily small (one-sided) error probability is tolerated. Later, Bernstein and Vazirani provided a relativized problem that can be solved in exact quantum polynomial time, but not in time $n^{o(\log n)}$ on any classical bounded-error probabilistic machine [4]. More recently, we constructed such a problem that would require exponential time on any classical bounded-error probabilistic machine [11]. None of these problems were decision problems. [1] Here we recast Simon's problem in a natural group-theoretic framework, we generalize it, and we give an exact quantum polynomial-time algorithm to solve it. This provides the first evidence of an exponential gap between the power of exact quantum computation and that of classical bounded-error probabilistic computation, even for decision problems.

[1] The Deutsch–Jozsa problem gives rise to an oracle decision problem [7, 8]. Also, in the soon-to-be-published journal version of their paper, Bernstein and Vazirani extend their result to a decision problem [5].

Of independent interest are the techniques developed to obtain our results. Two of the most fundamental techniques discovered so far in the field of quantum computation are Simon's [18] and Grover's [15]. Here, we generalize both techniques and we show for the first time that they can be made to work together toward a common goal: our algorithm crucially requires the availability of both these tools.

In this paper, we shall use the term $\mathcal{ZQP}$–algorithm (resp. $\mathcal{QP}$–algorithm) to denote an algorithm that runs in expected (resp. guaranteed worst-case) polynomial time on the quantum computer to solve an arbitrary problem. In particular, $\mathcal{ZQP}$ (resp. $\mathcal{QP}$) is the class of *decision* problems that allow a $\mathcal{ZQP}$–algorithm (resp. a $\mathcal{QP}$–algorithm). [2] To summarize our results, Simon gave a $\mathcal{ZQP}$–algorithm for his problem; we generalize it and give a $\mathcal{QP}$–algorithm. This allows for the construction of an oracle under which there is a decision problem in $\mathcal{QP}$ that would not only lie outside of the classical class $\mathcal{BPP}$—which was known already [5]—but that would require exponential time on any classical bounded-error probabilistic computer.

2 Simon's subgroup problem

We first state Simon's problem [18]. Let $n \geq 1$ be any integer and R any set representable on a quantum computer. Let $(\oplus) : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ denote the bitwise exclusive-or, written using infix notation.

Given: An integer $n \geq 1$ and a function $\rho : \{0,1\}^n \to R$.

Promise: There exists a nonzero element $s \in \{0,1\}^n$ such that for all $g, h \in \{0,1\}^n$, $\rho(g) = \rho(h)$ if and only if $g = h$ or $g = h \oplus s$.

Problem: Find s.

We say of such a function ρ that it *fulfills Simon's promise* with respect to s.

There is a nice group-theoretic interpretation and generalization for Simon's problem, and since that interpretation also helps simplify the notation, we shall use it. Hence, we reformulate Simon's problem as follows.

Let $\mathbb{Z}_2 = \{0,1\}$ denote the additive group of two elements with addition denoted by $\oplus$. For any given integer $n \geq 1$, let G denote the group $\langle \mathbb{Z}_2^n, \oplus \rangle$. For any subset $X \subseteq G$, let $|X|$ denote the cardinality of X and let $\langle X \rangle$ denote the subgroup generated by X. A subset X of a set Y is *proper* if $X \neq Y$. A subset $X \subseteq G$ is *linearly independent* in G if no proper subset of X generates $\langle X \rangle$. If $H \leqslant G$ is a subgroup then $g \in G$ is called a *representative* for the coset $g \oplus H$.

Define a bilinear map $G \times G \to \mathbb{Z}_2$ by

$$g \cdot h = \left(\sum_{i=1}^{n} g_i h_i \right) \bmod 2 \qquad (1)$$

where $g = (g_1, \ldots, g_n)$ and $h = (h_1, \ldots, h_n)$. For any subgroup $H \leqslant G$, let

$$H^\perp = \{ g \in G \mid g \cdot h = 0 \text{ for all } h \in H \} \qquad (2)$$

denote the *orthogonal subgroup* of H. For any subgroups $K \leqslant H \leqslant G$, let $[H : K]$ denote the index of K in H, that is, the number of cosets of K in H. Note that, for all subgroups $H \leqslant G$, we have $(H^\perp)^\perp = H$ and $|H^\perp| = [G : H]$. Using this terminology, we state the following problem.

Given: An integer $n \geq 1$ and a function $\rho : G = \mathbb{Z}_2^n \to R$.

Promise: There exists a subgroup $H_0 \leqslant G$ such that ρ is constant and distinct on each coset of H_0.

Problem: Find a generating set for H_0.

We say of such a function ρ that it *fulfills Simon's promise* with respect to subgroup H_0.

In Simon's original problem [18], H_0 is assumed to have order 2, that is, $H_0 = \{0, s\}$ for some $s \in G$ and the problem is then to find s. We shall, however, in the rest of this paper, refer to the above problem as *Simon's subgroup problem*. Simon gave in [18] a very simple and beautiful quantum algorithm for solving the subgroup problem. We now review the main ideas behind that algorithm, but we use a language which is rather different from Simon's.

A first crucial observation is that, given a generating set for a subgroup, one can easily (classically or quantumly) deduce a generating set for its orthogonal subgroup. This fact is often used in coding theory: given the generator matrix of a binary linear code, it allows to compute the generator matrix of its dual. We state this formally in Propositions 1 and 2 below.

Proposition 1 *There exists a classical deterministic algorithm that, given a subset $X \subseteq G = \mathbb{Z}_2^n$, returns a linearly independent subset of G that generates the subgroup $\langle X \rangle$. Moreover, the algorithm runs in time polynomial in n and linear in the cardinality of X.*

Proposition 2 *There exists a classical deterministic algorithm that, given a linearly independent subset $X \subset G = \mathbb{Z}_2^n$, returns a linearly independent subset of G that generates the orthogonal subgroup of $\langle X \rangle$. Moreover, the algorithm runs in time polynomial in n.*

From these two propositions, and since $(H^\perp)^\perp = H$ for all subgroups H, it follows that to solve Simon's subgroup problem it suffices to find a generating set for $H_0^\perp$. In [18],

13

Simon proved a special case of Theorem 3 below, which gives an efficient quantum algorithm for finding a *random* element of $H_0^\perp$ with respect to the uniform probability distribution.

Theorem 3 *Let $n \geq 1$ be an integer and $\rho : G = \mathbb{Z}_2^n \to R$ be a function that fulfills Simon's promise for some subgroup $H_0 \leqslant G$. Assume that a quantum algorithm to compute ρ is given, together with the value of n.*

Then there exists a quantum algorithm capable of finding a random element of the orthogonal subgroup $H_0^\perp$. Moreover, the algorithm runs in time linear in n and in the time required to compute ρ.

We refer to this algorithm as *Simon's subroutine* and discuss it further in the next section. By repeating the subroutine until one has a generating set for $H_0^\perp$ and then applying Propositions 1 and 2 above, one has solved Simon's subgroup problem. Before this yields an algorithm, however, we need a procedure to determine *when* to stop sampling. Moreover, a bound on the expected number of samples needed to build a generating set for $H_0^\perp$ is required in order to determine the running time of the algorithm.

Consider first the former question of how to determine if a sampled subset Y generates $H_0^\perp$. If H_0 is of known order 2 (as in Simon's original paper) then we stop sampling when Y generates a subgroup of order $|H_0^\perp| = [G : H_0] = 2^{n-1}$. If the order of H_0 is unknown then first observe that since $Y \subseteq H_0^\perp$ then $H_0 \subseteq \langle Y \rangle^\perp$ where equality holds if and only if Y generates $H_0^\perp$ (and not only a proper subgroup). In other words, Y generates $H_0^\perp$ if and only if ρ is constant on $\langle Y \rangle^\perp$. This last condition is easily checked by first applying Propositions 1 and 2 to find a linearly independent set X that generates the orthogonal subgroup $\langle Y \rangle^\perp$, and then evaluating ρ on X. It is thus easy to decide when we can stop sampling.

Consider now the latter question of how many times one must repeat Simon's subroutine in order to obtain a generating set for $H_0^\perp$. More generally, given any finite group H, what is the expected number of elements one must pick from H in order to have a generating set for H when the elements are picked mutually independently with respect to the uniform probability distribution on H? There is a simple (easy to improve) upper bound on this value which can be found as follows. Let $K \leqslant H$ be any proper subgroup. Then the probability that a randomly picked element in H is not in K is at least $1/2$, so after an expected number of at most 2 samples, we have picked an element $z \in H \setminus K$, and hence K is proper in $\langle z, K \rangle$. Since any sequence of proper subgroups in a finite group H can have length at most $\log_2 |H|$, it follows that after an expected number of at most $2 \log_2 |H|$ samples we have found a generating set for H.

By the above remarks, we can summarize the main steps in Simon's $\mathcal{ZQP}$–algorithm for solving his subgroup problem as follows. Assume we have a quantum polynomial-time algorithm to compute ρ. By Theorem 3, we can in polynomial time sample random elements of the orthogonal subgroup $H_0^\perp$. We have efficient routines for testing when to stop sampling and for finding H_0 from $H_0^\perp$. Finally, the expected number of samples needed is logarithmically bounded in the order of the group, giving an overall polynomial-time expected running time to find a generating set for H_0.

In our approach, we also solve Simon's subgroup problem by first finding a generating set for $H_0^\perp$, and we also use the method of finding repeatedly larger and larger subgroups $\langle Y \rangle$ of $H_0^\perp$. However, instead of finding an element in $H_0^\perp$ that is not already in $\langle Y \rangle$ with some bounded probability, we have discovered a method that *guarantees* that the sampled element is taken from the subset $H_0^\perp \setminus \langle Y \rangle$. In addition, our method for finding such an element needs only time polynomial in n and in the time required to compute ρ.

Theorem 4 *Let $n \geq 1$ be an integer and $\rho : G = \mathbb{Z}_2^n \to R$ be a function that fulfills Simon's promise for some subgroup $H_0 \leqslant G$. Assume that a quantum algorithm that computes ρ without making any measurements is given, together with the value of n and a linearly independent subset Y of the orthogonal subgroup $H_0^\perp$.*

Then there exists a quantum algorithm that returns an element of $H_0^\perp \setminus \langle Y \rangle$ provided Y does not generate $H_0^\perp$, and otherwise it returns the zero element. Moreover, the algorithm runs in time polynomial in n and in the time required to compute ρ.

We postpone the proof of this theorem till Section 4.3. Our new $\mathcal{QP}$–algorithm for solving Simon's subgroup problem follows easily from Theorem 4.

Theorem 5 ($\mathcal{QP}$–algorithm for Simon's problem)
Let $n \geq 1$ be an integer and $\rho : G = \mathbb{Z}_2^n \to R$ be a function that fulfills Simon's promise for some subgroup $H_0 \leqslant G$. Then given a quantum polynomial-time (in n) algorithm to compute ρ without making measurements, there exists a $\mathcal{QP}$–algorithm to find a generating set for H_0.

Proof The algorithm consists of two stages. In the first stage, we find a generating set for $H_0^\perp$ as follows. We initialize a counter $i = 0$ and set $Y^{(i)} = \emptyset$ to reflect the fact that we initially do not know any nontrivial elements of the orthogonal subgroup $H_0^\perp$.

We then compute the following process. We apply Theorem 4, giving an element $z^{(i+1)} \in H_0^\perp$. If the outcome $z^{(i+1)}$ is the zero element then we terminate the first stage. Otherwise, we set $Y^{(i+1)} = Y^{(i)} \cup \{z^{(i+1)}\}$ and increment the counter i by 1.

14

We repeat this process until we finally measure the zero element and then we terminate the first stage. Note that each of the subsets $Y^{(j)}$ ($0 \le j \le i$) is linearly independent by Theorem 4. Moreover, by the same theorem, since the final measured element $z^{(i+1)}$ is the zero element we know that $Y^{(i)}$ generates the orthogonal subgroup.

In the second stage, we apply Proposition 2 on the set $Y^{(i)}$ to find a generating set for H_0. This completes our proof of the existence and correctness of the algorithm.

Any linearly independent set in $G = \mathbb{Z}_2^n$ can have cardinality at most n and hence the algorithm applies Theorem 4 at most n times, each taking time polynomial in n. Since the final application of Proposition 2 also runs in polynomial time, the claimed running time follows. $\qquad\square$

3 Simon's quantum algorithm

We assume in this extended abstract that the reader is familiar with the basic notions of quantum computing [1, 6, 10]. We denote a register holding m qubits, all in the zero state, by $|0^m\rangle$. When its dimension is of no importance, we sometimes just write $|0\rangle$. For any nonempty subset $X \subseteq G$, let $|X\rangle$ denote the equally-weighted superposition $\frac{1}{\sqrt{|X|}}\sum_{x\in X}|x\rangle$. In particular, if $g \oplus H_0$ is a coset of H_0, then $|g \oplus H_0\rangle$ denotes the superposition $\frac{1}{\sqrt{|H_0|}}\sum_{h\in H_0}|g \oplus h\rangle$. For any nonempty subset $X \subseteq G$ and any element $g \in G$, let $|\phi_g X\rangle$ denote the superposition $\frac{1}{\sqrt{|X|}}\sum_{x\in X}(-1)^{g\cdot x}|x\rangle$.

Define the one-bit Walsh-Hadamard transform

$$\mathbf{W}_2 = \frac{1}{\sqrt{2}}\sum_{i,j=0}^{1}(-1)^{ij}|i\rangle\langle j|.$$

With respect to the ordered basis $(|0\rangle, |1\rangle)$, this reads $\mathbf{W}_2 = \frac{1}{\sqrt{2}}\left(\begin{smallmatrix}1&1\\1&-1\end{smallmatrix}\right)$. Let $\mathbf{W}_2^n$ denote the Walsh-Hadamard transform applied on each qubit of a system of n qubits. The result of applying $\mathbf{W}_2^n$ to $|g\rangle$, where $g \in G$, is the superposition $|\phi_g G\rangle$. Moreover, for any subgroup $K \leqslant G$ and any elements $g, h \in G$,

$$\mathbf{W}_2^n|\phi_h(g \oplus K)\rangle = (-1)^{g\cdot h}|\phi_g(h \oplus K^\perp)\rangle. \qquad (3)$$

Thus, by applying the Walsh-Hadamard transform, the subgroup is mapped to its orthogonal subgroup, and the phases translate to a coset and vice versa.

A classical function f is evaluated reversibly by the operation $\mathbf{U}_f$ which maps $|x\rangle|y\rangle$ to $|x\rangle|y \oplus f(x)\rangle$ [3]. Note that a second application of $\mathbf{U}_f$ will restore the second register to its original value since $|x\rangle|y \oplus f(x) \oplus f(x)\rangle = |x\rangle|y\rangle$.

Let T_0 be a transversal of H_0 in G, that is, a subset $T_0 \subseteq G$ that consists of exactly one representative from each coset of H_0. Simon's subroutine for finding a random element of $H_0^\perp$, working on the initial state $|0^n\rangle|0\rangle$, can be described as follows.

SIMON'S SUBROUTINE

1. Apply the inverse of transform $\mathbf{W}_2^n$ to the first register [3] producing an equally-weighted superposition of all elements in the group G,

$$\frac{1}{\sqrt{2^n}}\sum_{g\in G}|g\rangle|0\rangle.$$

2. Apply $\mathbf{U}_\rho$, producing a superposition of all cosets of H_0,

$$\frac{1}{\sqrt{2^n}}\sum_{g\in G}|g\rangle|\rho(g)\rangle = \frac{1}{\sqrt{|T_0|}}\sum_{t\in T_0}|t \oplus H_0\rangle|\rho(t)\rangle.$$

3. Apply $\mathbf{W}_2^n$ to the first register, producing a superposition over the orthogonal subgroup $H_0^\perp$,

$$|\Psi\rangle = \frac{1}{\sqrt{|T_0|}}\sum_{t\in T_0}|\phi_t H_0^\perp\rangle|\rho(t)\rangle. \qquad (4)$$

Suppose we measure the first register in the resulting superposition $|\Psi\rangle$. Let z be the outcome. It is immediate that z is a random element of the orthogonal subgroup $H_0^\perp$, which proves Theorem 3 above and implies Simon's $\mathcal{ZQP}$–algorithm for solving his subgroup problem.

Now, consider the crucial cause in Simon's algorithm why it is not a $\mathcal{QP}$–algorithm. Suppose we have already found an independent set $Y \subset H_0^\perp$ that generates only a proper subgroup of $H_0^\perp$. Then, what we would like is to measure an element $z \in H$ such that $Y \cup \{z\}$ is also linearly independent. However, Simon's algorithm promises only that z preserves independence with some probability. Our approach to finding z so that $Y \cup \{z\}$ is *certain* to be linearly independent consists in solutions to the following two subproblems. Suppose we have written $H_0^\perp$ as the internal direct sum of two subgroups, $H_0^\perp = K \oplus \langle Y\rangle$ for some nontrivial $K \leqslant H_0^\perp$. Then our solution, informally, consists of two parts.

1. We give a method for transforming $|H_0^\perp\rangle$ into $|K\rangle$.

2. We give a method for transforming $|K\rangle$ into $|X\rangle$ where $X \subseteq (K\setminus\{0\})$ is a nonempty subset consisting only of some of the nonzero elements of K.

In the next section, we present our solutions (Lemma 7 and Lemma 8, respectively) to these two problems. From these, we then prove Theorem 4 stated above. Our new $\mathcal{QP}$–algorithm for Simon's subgroup problem (Theorem 5) is an easy corollary to that theorem.

---

[3] Of course, we could apply $\mathbf{W}_2^n$ rather than its inverse since this transformation is self-inverse. Nevertheless, it is more natural to think of the operation in terms of the inverse of $\mathbf{W}_2^n$, especially if we wish to extend the notion to non-Abelian groups.

15

4 Our new $\mathcal{QP}$–algorithm

This section consists of three subsections. The first two contain our solutions to the two above-mentioned subproblems, while we combine them in the last subsection to prove Theorem 4.

4.1 Shrinking a subgroup

If we apply Simon's subroutine on the initial zero state $|0^n\rangle|0\rangle$, our quantum system is afterwards in superposition $|\Psi\rangle$ defined in Equation 4. In particular, for every element t in a transversal for H_0, the first register holds the superposition $|\phi_t H_0^\perp\rangle$. If we measure this register then we will measure a random element of $H_0^\perp$. Now, suppose we have earlier measured a nonzero element $y \in H_0^\perp$. Then we would like not to measure y once again, but rather some other element.

The following lemma provides us with a routine that ensures we will not measure y again. Since $y = (y_1, \ldots, y_n)$ is nonzero, it has some nonzero entry, say $y_j = 1$. The idea is to use the jth entry in the first register to change subgroup $H_0^\perp$ into the smaller subgroup $K = \{(h_1, \ldots, h_n) \in H_0^\perp \mid h_j = 0\}$ of all elements in $H_0^\perp$ with 0 in that entry. It follows then that we shall not obtain y again if we measure the first register in this new superposition.

Lemma 6 *Let $H \leqslant G$ be a nontrivial subgroup and let $y = (y_1, \ldots, y_n) \in H$ be a known nonzero element. Let j be such that $y_j = 1$ and let K denote the subgroup $\{(h_1, \ldots, h_n) \in H \mid h_j = 0\}$.*

Then there exists a quantum routine that, for all $g \in G$, given $|\phi_g H\rangle|0\rangle$ returns $|\phi_g K\rangle|g \cdot y\rangle$. Moreover, the routine runs in time linear in n and it uses no measurements.

Proof The routine consists of three unitary operations. Initially, we apply the controlled NOT operation where the jth qubit in the first register is the control bit and the second register holds the target bit. Applying this operation on the input $|\phi_g H\rangle|0\rangle$ produces

$$\frac{1}{\sqrt{|H|}} \sum_{h \in H} (-1)^{g \cdot h} |h\rangle|h_j\rangle$$
$$= \frac{1}{\sqrt{2}} \sum_{i \in \mathbb{Z}_2} (-1)^{g \cdot (iy)} \left(\frac{1}{\sqrt{|K|}} \sum_{k \in K} (-1)^{g \cdot k} |(iy) \oplus k\rangle \right) |i\rangle$$

where $h = (h_1, \ldots, h_n) \in H$, $0y = 0$ and $1y = y$. Then, if the second register holds a 1, we apply the operator defined

by $|x\rangle \mapsto |x \oplus y\rangle$ to the first register. This produces

$$\frac{1}{\sqrt{2}} \sum_{i \in \mathbb{Z}_2} (-1)^{g \cdot (iy)} \left(\frac{1}{\sqrt{|K|}} \sum_{k \in K} (-1)^{g \cdot k} |k\rangle \right) |i\rangle$$
$$= |\phi_g K\rangle \left(\frac{1}{\sqrt{2}} \sum_{i \in \mathbb{Z}_2} (-1)^{g \cdot (iy)} |i\rangle \right).$$

Finally, we apply $\mathbf{W}_2$ to the second register, giving the superposition in the lemma. The routine uses no measurements and its running time is clearly linear in n. $\qquad\square$

The above lemma can easily be generalized to the case in which we have already measured not merely one nonzero element of $H_0^\perp$, but any linearly independent subset of $H_0^\perp$. The solution is then to apply the above lemma repeatedly for each element in that subset.

Lemma 7 *Let $H \leqslant G$ be a nontrivial subgroup and $\{y^{(1)}, \ldots, y^{(m)}\} \subseteq H$ a known linearly independent set in H. Then there exist a subgroup $K \leqslant H$ with $H = K \oplus \langle y^{(1)}, \ldots, y^{(m)} \rangle$ and a quantum routine that, for all $g \in G$, returns $|\phi_g K\rangle|g \cdot y^{(1)}, \ldots, g \cdot y^{(m)}\rangle$ given $|\phi_g H\rangle|0^m\rangle$. Moreover, the routine runs in time linear in nm and it uses no measurements.*

4.2 Removing 0 from a subgroup

Consider first how much we have gained by using the result from the previous subsection. Suppose we first apply Simon's subroutine and then the routine in Lemma 7. Call this combined routine $\mathcal{A}'$. Given a linearly independent set $\{y^{(1)}, \ldots, y^{(m)}\} \subseteq H_0^\perp$, routine $\mathcal{A}'$ produces, on the input $|0^n\rangle|0, 0^m\rangle$, the superposition

$$|\Psi'\rangle = \frac{1}{\sqrt{|T_0|}} \sum_{t \in T_0} |\phi_t K\rangle|\rho(t), t \cdot y^{(1)}, \ldots, t \cdot y^{(m)}\rangle. \tag{5}$$

Here K is given as in Lemma 7. Thus, for every t in a transversal T_0 for H_0, we hold the superposition $|\phi_t K\rangle$ in the first register. By Lemma 7, if we measure the first register, we cannot obtain a previously known element of $H_0^\perp$. Neither can we obtain a nonzero element that is a linear combination of known elements. Nevertheless, it remains possible that we obtain the zero element.

In this subsection, we show how to avoid measuring the zero element. This solves the second subproblem mentioned at the end of Section 3. Our solution does not build on a group-theoretical view as in the previous subsection, but instead on a general view of $\mathcal{A}'$ as a probabilistic quantum algorithm that succeeds with some bounded probability.

We say that a state in the superposition $|\Psi'\rangle$ is *good* if it contains a nonzero element in the first register. States that are not good are said to be *bad*. The success probability

16

of $\mathcal{A}'$ is the probability that we measure a good state by measuring the first register of the system.

If $K = \{0\}$ is the trivial subgroup then the success probability of $\mathcal{A}'$ is clearly zero. Otherwise, $\mathcal{A}'$ succeeds with probability $1 - 1/k$ where $k = |K|$ is the order of K. Thus, we can initially distinguish between these two cases with high probability, but not with certainty. Our result in this subsection is a method to encapsulate $\mathcal{A}'$ in a larger quantum algorithm that succeeds with certainty provided $K \neq \{0\}$ (see Theorem 4 for an example). To obtain this, consider first the more general problem of improving the success probability of a probabilistic quantum algorithm, formulated as follows.

Suppose we are given a quantum algorithm $\mathcal{A}$ that on input $|0\rangle$ returns some superposition $|\Psi\rangle = \sum_{i \in I} |i\rangle |\psi_i\rangle$ for some finite index set $I \subset \mathbb{Z}$. Suppose also that I can be written as the disjunct sum of two sets A and B where A corresponds to the "good" solutions and B to the "bad" solutions, and suppose we are given a quantum algorithm to compute the characteristic function $\chi = \chi_A : I \to \{0,1\}$ of A. Let $|A\rangle = \sum_{i \in A} |i\rangle |\psi_i\rangle$ denote the superposition of good solutions, and $|B\rangle = \sum_{i \in B} |i\rangle |\psi_i\rangle$ the superposition of bad solutions. Write $|\Psi\rangle = |A\rangle + |B\rangle$. Let $a = \langle A|A\rangle$ denote the probability that we measure a good solution, and similarly let $b = \langle B|B\rangle$. Note that $\langle A|B\rangle = 0$ and hence $a + b = 1$.

Using a generalization of the technique in Grover's algorithm [15], we encapsulate $\mathcal{A}$ in a larger quantum algorithm $\mathcal{Q}$ such that the probability that $\mathcal{Q}$ returns a good solution is significantly better compared to the probability that $\mathcal{A}$ returns a good solution. In [9], it is shown that if $\mathcal{A} = \mathbf{W}_2^n$ is the Walsh-Hadamard transform and the probability of success of $\mathcal{A}$ is exactly one quarter ($a = 1/4$) then $\mathcal{Q}$ can be constructed such that it succeeds with certainty.

For our purpose, we require a similar technique which applies in the case that $\mathcal{A}$ is any quantum algorithm that uses no measurements and has success probability exactly one half ($a = 1/2$). To obtain this result, we use complex phases, whereas in Grover's original algorithm only the real phases ± 1 are needed [15]. Let $\imath = \sqrt{-1}$ denote the square root of -1. (Do not confuse imaginary $\imath$ with integer i.) The formal setting and the lemma are as follows.

Lemma 8 *Let $\mathcal{A}$ be a quantum algorithm that uses no measurements and that given $|0\rangle$ returns $|\Psi\rangle = \sum_{i \in I} |i\rangle|\psi_i\rangle$ for some finite index set $I \subset \mathbb{Z}$. Let $\chi : I \to \{0,1\}$ be any Boolean function. Define*

$$
\begin{array}{ll}
A = \{i \in I \mid \chi(i) = 1\} & B = \{i \in I \mid \chi(i) = 0\} \\
|A\rangle = \sum_{i \in A} |i\rangle|\psi_i\rangle & |B\rangle = \sum_{i \in B} |i\rangle|\psi_i\rangle \\
a = \langle A|A\rangle & b = \langle B|B\rangle .
\end{array}
$$

Then there exists a quantum algorithm $\mathcal{Q}$ that on input $|0\rangle$ returns $k|A\rangle + l|B\rangle$ where $k = 2\imath(1 - a) - 1$ and

$l = \imath(1 - 2a)$. *In particular, if $a = \frac{1}{2}$ then the result is $(\imath - 1)|A\rangle$. If $a = 0$ then $|A\rangle$ has norm zero and hence the result is $\imath|B\rangle$. Moreover, $\mathcal{Q}$ runs in time linear in the number of qubits and in the times required to compute $\mathcal{A}$ and $\mathbf{U}_\chi$, and it uses no measurements.*

Proof First note that $B = I \setminus A$ and that $|\Psi\rangle = |A\rangle + |B\rangle$ can be written as a sum of "good" and "bad" solutions with inner product zero, $\langle A|B\rangle = 0$. Note also that the probabilities to measure a "good" or "bad" solution sum to 1: $a + b = 1$. For every $k, l \in \mathbb{C}$ with $|k|^2 a + |l|^2 b = 1$, define the normalized superposition $|\Psi(k, l)\rangle = k|A\rangle + l|B\rangle$. Here $|x|$ denotes the norm of $x \in \mathbb{C}$. Note that $|\Psi(1, 1)\rangle = |\Psi\rangle = \mathcal{A}|0\rangle$.

Now, instead of measuring $|\Psi(1, 1)\rangle = |\Psi\rangle$ immediately, we add one Grover iteration before the measurement. This Grover iteration is not the one from Grover's paper [15] but a generalized version defined as follows. Let the phase-change operator $\mathbf{S}_A$ be defined by

$$
\mathbf{S}_A|i\rangle|\psi_i\rangle = \begin{cases} \imath |i\rangle|\psi_i\rangle & \text{if } i \in A \\ |i\rangle|\psi_i\rangle & \text{if } i \in B. \end{cases}
$$

In a similar manner, let $\mathbf{S}_{\{0\}}$ be the operator that changes the phase by $\imath$ if and only if the state is the zero state.

Define the *Grover iteration* as

$$
\mathbf{G} = \mathcal{A}\,\mathbf{S}_{\{0\}}\,\mathcal{A}^{-1}\,\mathbf{S}_A.
$$

Straightforward calculations show that applying $\mathbf{G}$ on a superposition of the form $|\Psi(k, l)\rangle$ has the same kind of effect as the one in [15]. In particular, we have

$$
\mathbf{G}|\Psi(1, 1)\rangle = |\Psi(2\imath(1 - a) - 1, \imath(1 - 2a))\rangle.
$$

Let $\mathcal{Q}$ be the quantum algorithm in which we first apply $\mathcal{A}$ and then $\mathbf{G}$. Then applying $\mathcal{Q}$ on input $|0\rangle$ produces

$$
\begin{aligned}
\mathcal{Q}|0\rangle &= \mathbf{G}|\Psi(1, 1)\rangle \\
&= (2\imath(1 - a) - 1)|A\rangle + \imath(1 - 2a)|B\rangle,
\end{aligned}
$$

and the first part of the lemma follows.

The phase-change operator $\mathbf{S}_{\{0\}}$ can be applied in time linear in the number of qubits, while $\mathbf{S}_A$ can be applied by computing $\mathbf{U}_\chi$ twice and doing a constant amount of additional work [2]. Hence, $\mathcal{Q}$ runs in time linear in the number of qubits and in the times required to compute $\mathcal{A}$ and $\mathbf{U}_\chi$. $\square$

4.3 Composing our new $\mathcal{QP}$–algorithm

By Lemma 8, we can take a quantum algorithm $\mathcal{A}$ and construct a new quantum algorithm $\mathcal{Q}$ such that if $\mathcal{A}$ succeeds with probability zero then so does $\mathcal{Q}$, and if $\mathcal{A}$ has success probability $1/2$ then $\mathcal{Q}$ succeeds with certainty. Consider the algorithm $\mathcal{A}'$ defined in the beginning of Subsection 4.2. It succeeds with probability zero if $K = \{0\}$ is trivial, and otherwise with probability $1 - 1/k$ where $k = |K|$ is the order of K.

17

At first glance, it seems that we cannot apply Lemma 8 since $\mathcal{A}'$ succeeds with too large a probability. Fortunately, we can get around this problem by redefining what we mean by a state in the superposition $|\Psi'\rangle$ given in (5) being good. Fix an i with $1 \leq i \leq n$. Redefine a state to be good if the ith entry in the first register is 1. What is now the success probability p_i of $\mathcal{A}'$? If all elements in K have 0 in the ith entry then $p_i = 0$. Otherwise, exactly half the elements in K have 0 in the ith entry and exactly half of them have 1, and therefore $p_i = 1/2$. The success probability p_i of $\mathcal{A}'$ is thus either zero or one half.

Consider the set of probabilities $\{p_1, \ldots, p_n\}$. It contains one value for each entry in the first register. If $K = \{0\}$ is trivial then $p_i = 0$ for all $1 \leq i \leq n$. Otherwise, if K is nontrivial then at least for one i with $1 \leq i \leq n$ we have $p_i = 1/2$. This suggests that if we apply Lemma 8 once for each of the n different values of i in order to improve the success probability of $\mathcal{A}'$ from p_i to $2p_i$, then at least one of the measured elements is a nonzero element of K if and only if K is nontrivial. We prove now that this is indeed the case by giving our proof of Theorem 4 stated in Section 2.

Proof of Theorem 4 Write $H_0^\perp = K \oplus \langle Y \rangle$ as the direct sum of a subgroup K and the subgroup generated by the known elements $Y \subset H_0^\perp$.

Let $P = \{i : 1 \leq i \leq n\}$. For every $i \in P$, we construct a quantum algorithm $\mathcal{Q}_i$ that on input $|0^n\rangle|0, 0^m\rangle$ returns an element $z^{(i)}$ of K after a measurement. We then construct a larger algorithm which consists of all the n smaller $\mathcal{Q}_i$ and we show that at least one of the measured elements $z^{(i)}$ is nonzero if and only if K is nontrivial.

Fix an $i \in P$. Define the function $\chi_i : G \to \{0, 1\}$ by $\chi_i(g) = g_i$ where $g = (g_1, \ldots, g_n)$. Thus, $\chi_i(g)$ is 1 if and only if the ith entry of g is 1.

The output of the computation $\mathcal{A}'|0^n\rangle|0, 0^m\rangle$ is the superposition $|\Psi'\rangle$ given in (5). If we measure $|\Psi'\rangle$, let p_i denote the probability that we obtain a state $|g\rangle|x\rangle$ with the ith entry of g equal to 1, $g_i = 1$. If all elements in K hold a 0 in that entry then p_i is zero. Otherwise, half the elements in K hold a 1 in that entry and thus, independently of the content of the second register of $|\Psi'\rangle$, we have that $p_i = 1/2$.

Suppose we apply Lemma 8 on the function χ_i and the algorithm $\mathcal{A}'$ defined above. Let $\mathcal{Q}_i$ denote the resulting quantum algorithm. Consider the content of the first register in the final superposition. By Lemma 8, that register contains only elements from $K \leqslant H_0^\perp$, as did $|\Psi'\rangle$ originally. Moreover, each of these elements holds a 1 in the ith entry if and only if $p_i = 1/2$.

Suppose we measure the first register. Let $z^{(i)} \in K$ be the outcome. If $p_i = 1/2$ then $z^{(i)}$ is nonzero with certainty. Otherwise, that is if $p_i = 0$, then $z^{(i)}$ may or may not be nonzero.

Consider that we run quantum algorithm $\mathcal{Q}_i$ sequentially for each $i \in P$, and follow each run by a measurement of the first register. Suppose K is nontrivial. Let $g \in K$ be any nonzero element and let $i_0 \in P$ be so that $g_{i_0} = 1$ where $g = (g_1, \ldots, g_n)$. Then $p_{i_0} = 1/2$ and therefore, with certainty, the measured element $z^{(i_0)} \in K$ is nonzero. Now, suppose K is trivial. Then, for all $i \in P$, we have that $z^{(i)}$ is the zero element. This completes the first part of the theorem.

Consider the overall running time of this composed quantum algorithm. Each of the transforms $\mathbf{U}_{\chi_i}$ can clearly be implemented in constant time and thus, by Lemma 8, $\mathcal{Q}_i$ runs in time linear in n and in the time required to compute $\mathcal{A}$. Since the composed algorithm consists of running each of the n algorithms $\mathcal{Q}_i$ one after the other, it runs in time polynomial in n and in the time required to compute $\mathcal{A}$ as well. $\qquad\square$

Having proved Theorem 4, we have completed the description and proof of our new $\mathcal{QP}$–algorithm for solving Simon's subgroup problem.

We end this section with a supplementary remark on the total number of times we need to compute function ρ. By the proof of Theorem 5, we apply Theorem 4 at most n times. In each of these applications, we run each of the n quantum algorithms $\mathcal{Q}_i$ ($i \in P$) defined above. Since each $\mathcal{Q}_i$ computes the function ρ a constant number of times this gives an upper bound of $O(n^2)$ evaluations of ρ.

We will show that just $O(n)$ evaluations of ρ suffices. First, restate the above counting argument as follows. For each $i \in P$, our $\mathcal{QP}$–algorithm applies $\mathcal{Q}_i$ at most n times. Since P has cardinality n the number of evaluations of ρ is $O(n^2)$. We now show that it is suffices to run each $\mathcal{Q}_i$ at most once.

Consider the first time we run $\mathcal{Q}_i$ for $i \in P$. There are two cases depending on the outcome $z^{(i)}$. If $z^{(i)}$ is the zero element then we know that all elements in the subgroup K (defined in the beginning of the proof of Theorem 4) hold a 0 in the ith entry. Moreover, this will also be the case in successive iterations when K has shrunk further. Therefore, it would be pointless to run $\mathcal{Q}_i$ again.

On the other hand, if $z^{(i)}$ is nonzero then it fulfills the requirements in Theorem 4 of being a nonzero element preserving independence. Thus, we do not need to run any of the remaining $\mathcal{Q}_i$ algorithms in that application of Theorem 4. Moreover, since the ith entry in $z^{(i)}$ is a 1, we can construct our new subgroup K in the next applications of Theorem 4 such that all elements in the new subgroup K hold a 0 in the ith entry. This is done by letting that entry be the control bit in Lemma 6, that is, by choosing $j = i$. Thus, also in this case, we do not need to run $\mathcal{Q}_i$ again.

18

5 Decision problems

Until now, we have dealt with a version of Simon's problem that consists of finding a generating set for some subgroup $H_0 \leqslant G = \mathbb{Z}_2^n$. In Simon's original setting, this subgroup is of order 2 and the problem reduces to finding its unique nonzero element, called s at the beginning of Section 2. Recall that we say of such a function that it *fulfills Simon's promise* with respect to s. A natural question is whether there exists a *decision* problem in $\mathcal{QP}$ that would require exponential time to decide on any classical bounded-error probabilistic computer. In this section, we give a positive answer in an appropriate oracle setting.

This can be achieved in several ways. The simplest is to note that our algorithm can distinguish with certainty, after guaranteed worst-case polynomial time, between a function $\rho : \{0,1\}^n \rightarrow \{0,1\}^n$ that is a bijection and one that fulfills Simon's promise: Just apply our general algorithm and see if it turns out zero or one generator for H_0. (Recall that Simon's original algorithm could distinguish these two cases with certainty after *expected* polynomial time, or alternatively it could distinguish them *with bounded error probability* after guaranteed worst-case polynomial time.) In his paper [18], Simon proves the existence of an oracle $\mathcal{O}$ and a decision problem L such that (1) no classical probabilistic oracle machine that queries $\mathcal{O}$ fewer than $2^{n/4}$ times on input 1^n can decide L with bounded error probability, and (2) deciding L given $\mathcal{O}$ reduces efficiently and deterministically to the problem of distinguishing between the two types of functions mentioned above. It follows from our algorithm that L can be decided with certainty in guaranteed worst-case polynomial time on a quantum computer, given $\mathcal{O}$ as oracle: $L \in \mathcal{QP}^{\mathcal{O}}$.

Another approach to transforming Simon's problem into a decision problem, which we find more elegant, is to consider an arbitrary function $\gamma : \{0,1\}^+ \rightarrow \{0,1\}$, which is *balanced* in the sense that there are exactly 2^{n-1} strings $x \in \{0,1\}^n$ such that $\gamma(x) = b$ for each $b \in \{0,1\}$ and $n \geq 1$. For example, $\gamma(x)$ could be simply the most significant or the least significant bit of x, or it could be the exclusive-or of all the bits in x. Consider now an integer n and a function $\rho : \{0,1\}^n \rightarrow \{0,1\}^{n-1}$ chosen randomly according to the uniform distribution among all functions that fulfill Simon's promise with respect to some nonzero $s \in \{0,1\}^n$. We prove below (Theorem 9) that no subexponential-time classical probabilistic algorithm can guess $\gamma(s)$ essentially better than at random, except with exponentially small probability, when ρ is provided as an oracle. The probabilities are taken among all choices for ρ, as well as the probabilistic choices made by the algorithm, but *not* over the possible choice for γ: any fixed balanced γ will do. It follows (Corollary 10) that there is an oracle that simultaneously defeats every classical algorithm.

Theorem 9 *Fix an arbitrary balanced function γ and an integer $n \geq 4$. Consider an arbitrary classical probabilistic algorithm that has access to a function oracle $\rho : \{0,1\}^n \rightarrow \{0,1\}^{n-1}$ chosen at random according to the uniform distribution among all functions that fulfill Simon's promise with respect to some s. Assume the algorithm queries its oracle no more than $2^{n/3}$ times. Then there exists an event $\mathcal{E}$ such that (1) $\mathrm{Prob}[\mathcal{E}] < 2^{-n/3}$, and (2) If $\mathcal{E}$ does not occur then the probability that the algorithm correctly returns $\gamma(s)$ is less than $\frac{1}{2} + 2^{-n/3}$. The probabilities are taken over all possible choices of function ρ and the probabilistic choices made by the algorithm. It follows that the algorithm cannot guess the value of $\gamma(s)$ with a probability better than $\frac{1}{2} + 2 \times 2^{-n/3}$.*

Proof Assume that the algorithm has queried its oracle on inputs $x_1, x_2, \ldots, x_k$ for $x_i \in \{0,1\}^n$, $1 \leq i \leq k \leq 2^{n/3}$. Without loss of generality, assume that all the queries are distinct. Let $y_1, y_2, \ldots, y_k$ be the answers obtained from the oracle, i.e. $y_i = \rho(x_i)$ for each i. Define the event $\mathcal{E}$ as *occurring* if there exist i and j, $1 \leq i < j \leq k$, such that $y_i = y_j$. Clearly, the algorithm has discovered the secret s when $\mathcal{E}$ occurs since in that case $s = x_i \oplus x_j$. This allows the algorithm to determine $\gamma(s)$ with certainty. We have to prove that $\mathcal{E}$ is very unlikely and that, unless $\mathcal{E}$ occurs, the algorithm has so little information that it cannot guess $\gamma(s)$ significantly better than at random.

Let $X = \{x_1, x_2, \ldots, x_k\}$ be the set of queries to the oracle and let $Y = \{y_1, y_2, \ldots, y_k\}$ be the corresponding answers. Let $W = \{x_i \oplus x_j \mid 1 \leq i < j \leq k\}$, $S = \{0,1\}^n \setminus (W \cup \{0^n\})$ and let $m < k^2$ be the cardinality of W. Note that $\mathcal{E}$ occurs if and only if $s \in W$ since $y_i = y_j$ if and only if $x_i \oplus x_j = s$. If $\mathcal{E}$ does not occur, we say that any $\hat{s} \in S$ is *compatible* with the available data because it is not ruled out as a possible value for the actual unknown s. Similarly, given any compatible $\hat{s}$, we say that a function $\hat{\rho} : \{0,1\}^n \rightarrow \{0,1\}^{n-1}$ is *compatible* with the available data (and with $s = \hat{s}$) if $\hat{\rho}(x_i) = y_i$ for all $i \leq k$, and if $\hat{\rho}(x) = \hat{\rho}(x')$ if and only if $x \oplus x' = \hat{s}$ for all distinct x and x' in $\{0,1\}^n$.

Assume for the moment that $\mathcal{E}$ has not occurred. Now we prove that there are exactly $(2^n - m - 1)((2^{n-1} - k)!)$ functions that are compatible with the available data. For each compatible $\hat{s}$, exactly $(2^{n-1} - k)!$ of those functions are also compatible with $s = \hat{s}$. It follows that all compatible values for s are equally likely to be correct given the available data, and therefore the only information available about s is that it belongs to S. For this, consider an arbitrary compatible $\hat{s}$. Define $X' = \{x \oplus \hat{s} \mid x \in X\}$. It follows from the compatibility of $\hat{s}$ that $X \cap X' = \emptyset$. Let $Z = \{0,1\}^n \setminus (X \cup X')$. Note that $x \in Z$ if and only if $x \oplus \hat{s} \in Z$. Partition Z in an arbitrary way into $Z_1 \cup Z_2$ so that $x \in Z_1$ if and only if $x \oplus \hat{s} \in Z_2$. The cardinalities of Z_1 and Z_2 are $(2^n - 2k)/2 = 2^{n-1} - k$. Now let

$Y' = \{0,1\}^{n-1} \setminus Y$, also a set of cardinality $2^{n-1} - k$. To each bijection $\xi : Z_1 \to Y'$ there corresponds a function $\hat{\rho}$ compatible with the available data and $s = \hat{s}$ defined by

$$\hat{\rho}(x) = \begin{cases} y_i & \text{if } x = x_i \text{ for some } 1 \le i \le k \\ y_i & \text{if } x = x_i \oplus \hat{s} \text{ for some } 1 \le i \le k \\ \xi(x) & \text{if } x \in Z_1 \\ \xi(x \oplus \hat{s}) & \text{if } x \in Z_2 \,. \end{cases}$$

The conclusion about the number of compatible functions follows from the facts that there are $(2^{n-1} - k)!$ such bijections, each possible function compatible with the available data and $s = \hat{s}$ is counted exactly once by this process, and there are $2^n - m - 1$ compatible choices for $\hat{s}$, each yielding a disjoint set of functions compatible with the available data.

Still considering the case that $\mathcal{E}$ has not occurred, let $A = \{z \in S \mid \gamma(z) = 1\}$ and $B = \{z \in S \mid \gamma(z) = 0\}$. Because we have just seen that each elements of S is equally likely to be the correct value for s, the algorithm's best strategy is to return 1 if $|A| > |B|$ and 0 otherwise. In the best case (for the algorithm), there are 2^{n-1} strings in A and the remaining $2^{n-1} - 1 - m$ strings are in B, in which case the guess is correct with probability

$$\frac{2^{n-1}}{2^n - 1 - m} \le \frac{2^{n-1}}{2^n - k^2} \le \frac{2^{n-1}}{2^n - 2^{2n/3}} < \frac{1}{2} + 2^{-n/3}$$

provided $n \ge 4$.

It remains to prove that the probability that event $\mathcal{E}$ occurs is exponentially small. For this, note that all nonzero values for s are equally likely *a priori* and event $\mathcal{E}$ occurs if and only if $s \in W$. It follows that

$$\text{Prob}[\mathcal{E}] = m/(2^n - 1) < k^2/2^n \le 2^{-n/3},$$

where m is the cardinality of W and $k \le 2^{n/3}$ is the number of oracle queries.

In conclusion, the probability that the algorithm guesses $\gamma(s)$ correctly is less than

$$\text{Prob}[\mathcal{E}] + (1 - \text{Prob}[\mathcal{E}]) \left(\tfrac{1}{2} + 2^{-n/3}\right)$$

$$< 2^{-n/3} + \left(\tfrac{1}{2} + 2^{-n/3}\right) = \tfrac{1}{2} + 2 \times 2^{-n/3} \,.$$

$\square$

The theorem we have just proven says that no classical probabilistic algorithm can guess $\gamma(s)$ much better than at random without spending exponential time on a random function that fulfills Simon's promise, provided that function is supplied as a black box chosen *after* the algorithm has been fixed. Can we find a single function that simultaneously defeats *all* classical probabilistic algorithms? The answer is obviously negative for any fixed finite function.

Nevertheless, the following corollary shows that it is possible to encode an infinite number of such functions into a single oracle so that every classical probabilistic algorithm is defeated infinitely many times. This exhibits an exponential gap between the power of exact quantum computation and that of classical bounded-error probabilistic computation, even for decision problems.

Corollary 10 *There exists an oracle $\mathcal{O}$ relative to which there is a decision problem $L \in \mathcal{QP}^{\mathcal{O}}$ so that, for any classical probabilistic algorithm whose running time is bounded by $2^{n/3}$ on all inputs of size n, there are infinitely many inputs about which the algorithm decides membership in L with probability no better than $\tfrac{1}{2} + 2 \times 2^{-n/3}$.*

Proof Fix some polynomial-time computable balanced function γ once and for all. For any fixed classical probabilistic algorithm, integer n, and function $\rho : \{0,1\}^n \to \{0,1\}^{n-1}$ that fulfills Simon's promise with respect to some s, we say that the algorithm is *defeated* by ρ if it cannot guess $\gamma(s)$ with probability better than $\tfrac{1}{2} + 2 \times 2^{-n/3}$ after taking less than $2^{n/3}$ steps. It follows directly from Theorem 9 that every classical probabilistic algorithm is defeated by at least one ρ for each value of $n \ge 4$. This remains true even if the algorithm is supplied with another arbitrary fixed oracle, in addition to the oracle for ρ.

Define function e by $e(1) = 2$ and $e(i+1) = 2^{e(i)}$ for $i \ge 1$. Let η be an arbitrary function that maps integers to classical probabilistic algorithms such that every algorithm appears infinitely many times in the image of η. For any integer n and functions $\rho : \{0,1\}^n \to \{0,1\}^{n-1}$ and $\sigma : \{0,1\}^+ \to \{0,1\}^\star$, let $[\rho, \sigma] : \{0,1\}^+ \to \{0,1\}^\star$ denote the function that sends x to $\rho(x)$ for all $x \in \{0,1\}^n$ and to $\sigma(x)$ otherwise. We build the required function oracle $\mathcal{O} : \{0,1\}^+ \to \{0,1\}^\star$ by stages: $\mathcal{O}(x) = \mathcal{O}_i(x)$ for all $x \in \{0,1\}^n$ such that $n < e(i+1)$ and $i \ge 1$. Initially, $\mathcal{O}_1(x)$ is the string of size $n - 1$ obtained by removing the most significant bit of x for each $x \in \{0,1\}^n$ and each $n \ge 1$. For each $i \ge 2$, let $n = e(i)$ and define $\mathcal{O}_i$ given $\mathcal{O}_{i-1}$ as follows: choose a function $\rho_i : \{0,1\}^n \to \{0,1\}^{n-1}$ that fulfills Simon's promise with respect to some s_i in a way that defeats algorithm $\eta(i)$ given $[\rho_i, \mathcal{O}_{i-1}]$ as oracle; let $\mathcal{O}_i = [\rho_i, \mathcal{O}_{i-1}]$. Finally define

$$L = \{1^{e(i)} \mid i \ge 2 \text{ and } \gamma(s_i) = 1\}.$$

Note that for each positive integer n, the restriction of $\mathcal{O}$ to $\{0,1\}^n$ is a function that fulfills Simon's promise, and therefore $L \in \mathcal{QP}^{\mathcal{O}}$ by the algorithm given in this paper. On the other hand, consider an arbitrary classical probabilistic algorithm A and let i be one of the infinitely many integers such that $\eta(i) = \mathsf{A}$. We know by construction that ρ_i defeats A given oracle $\mathcal{O}_i$. This means that A cannot

20

guess $\gamma(s_i)$ significantly better than at random after taking less than exponentially many steps on input $1^{e(i)}$. But A would require exponential time even to *formulate* a question of size $e(i + 1) = 2^{e(i)}$ or bigger for its oracle. Since $\mathcal{O}(x) = \mathcal{O}_i(x)$ for all x of size shorter than $e(i + 1)$, it follows that A behaves in the same way on input $1^{e(i)}$ whether it is given $\mathcal{O}$ or $\mathcal{O}_i$ as oracle, unless it takes exponential time. Therefore ρ_i defeats A given oracle $\mathcal{O}$ as well. This happens infinitely often for each classical probabilistic algorithm, which proves the desired result. $\qquad\square$

6 Other Abelian groups

So far, we have restricted our attention to the Abelian group $G = \mathbb{Z}_2^n$. In this section, we discuss how these results generalize to other Abelian groups. We start by considering the natural extension of Simon's problem and subroutine to an arbitrary finite additive Abelian group. Our presentation is kept in group-theoretical terms and our main tools are three quantum operators defined on the group. We also discuss how our own algorithm generalizes.

For every $m \geq 1$, let $\mathbb{Z}_m$ denote the additive cyclic group of order m. For any given n–tuple of positive integers $(m_1, \ldots, m_n)$, let $G = \langle G, + \rangle$ denote the finite additive Abelian group $\mathbb{Z}_{m_1} \oplus \cdots \oplus \mathbb{Z}_{m_n}$. We define the *Abelian subgroup problem* as follows: Given group G and a function ρ defined on G and promised to be constant and distinct on each coset of some unknown subgroup $H_0 \leq G$, find a generating set for H_0.

Our first task is to generalize the concept of orthogonality given by Equations 1 and 2. For every $m \geq 1$, let $\omega_m = \exp(2\pi\imath/m)$ denote the mth principal root of unity. Let $\mathbb{C}^*$ denote the multiplicative group of the nonzero complex numbers. Define a bilinear map $\mu : G \times G \to \mathbb{C}^*$ by

$$\mu(g, h) = \sum_{i=1}^{n} \omega_{m_i}^{g_i h_i} \qquad (6)$$

where $g = (g_1, \ldots, g_n)$ and $h = (h_1, \ldots, h_n)$. We say that an element $g \in G$ is *orthogonal* to a subset $X \subseteq G$ if, for all $x \in X$, we have that $\mu(g, x)$ is the identity of the group $\mathbb{C}^*$, that is, if $\mu(g, x) = 1$. Note the correspondence to the bilinear map in Section 3: There, the image was an additive group with identity 0, while here, the image is a multiplicative group with identity 1. For any subgroup $H \leq G$, let

$$H^\perp = \{g \in G \mid \mu(g, h) = 1 \text{ for all } h \in H\} \qquad (7)$$

denote the set of all elements in G orthogonal to H. Clearly, $H^\perp$ is a subgroup and we refer to it as the *orthogonal subgroup* of H.

For any subgroups $K \leq H \leq G$, let $[H : K]$ denote the index of K in H. As in the simple case $G = \mathbb{Z}_2^n$, we have

the duality relations

$$|H^\perp| = [G : H]$$
$$H^{\perp\perp} = H$$

for all subgroups $H \leq G$.

We now define three fundamental quantum operators for the group G. Together, they extend the ideas and the notation used in Section 3. They are the *quantum Fourier transform* $\mathbf{F}_G$, the *translation operator* τ_t $(t \in G)$, and the *phase-change operator* ϕ_h $(h \in G)$, defined as follows.

$$\mathbf{F}_G = \frac{1}{\sqrt{|G|}} \sum_{g,h\in G} \mu(g, h)|g\rangle\langle h|$$
$$\tau_t = \sum_{g\in G} |t + g\rangle\langle g|$$
$$\phi_h = \sum_{g\in G} \mu(h, g)|g\rangle\langle g|$$

One may readily check that these three G–operators are unitary. Note that when $G = \mathbb{Z}_2^n$ then the transform $\mathbf{F}_G$ is just the Walsh-Hadamard transform $\mathbf{W}_2^n$ used in Section 3. Unsurprisingly, the Fourier transform maps a subgroup $H \leq G$ to its orthogonal subgroup $H^\perp$,

$$\mathbf{F}_G|H\rangle = |H^\perp\rangle.$$

Moreover, the G–operators satisfy the following commutative laws which we state without proof.

Theorem 11 (Commutative laws of the G–operators)
For every $h, t \in G$ we have

$$\mu(h, t)\,\tau_t\,\phi_h = \phi_h\,\tau_t$$
$$\mathbf{F}_G\,\phi_h = \tau_{-h}\,\mathbf{F}_G$$
$$\mathbf{F}_G\,\tau_t = \phi_t\,\mathbf{F}_G.$$

With this setup, we can give a natural extension of Simon's subroutine, denoted $\mathbf{U}_G$, for the general Abelian subgroup problem.

$$\mathbf{U}_G = (\mathbf{F}_G \otimes \mathbf{I}) \circ \mathbf{U}_\rho \circ (\mathbf{F}_G^{-1} \otimes \mathbf{I}) \qquad (8)$$

Here, $\mathbf{I}$ denotes the identity operator, and the notation $\mathbf{U}_1 \otimes \mathbf{U}_2$ means applying the unitary operator $\mathbf{U}_1$ on the first register and $\mathbf{U}_2$ on the second.

Consider that we perform the experiment

$$z = \mathcal{M}_1 \circ \mathbf{U}_G|0\rangle|0\rangle \qquad (9)$$

where $\mathcal{M}_1$ denotes a measurement of the first register with outcome z and where the first register initially holds the identity of the group G.

If $G = \mathbb{Z}_2^n$ then the outcome $z \in G$ is a random element of the orthogonal subgroup $H_0^\perp \leq G$ by the discussion of

21

Simon's subroutine in Section 3. With the help of the commutative laws in Theorem 11, we now give a short proof that this holds for every finite additive Abelian group G.

The experiment given in Equations 8 and 9 consists of four operations. As the first, we apply $\mathbf{F}_G^{-1} \otimes \mathbf{I}$ on the initial zero state $|0\rangle|0\rangle$, producing an equally-weighted superposition of all elements in the group G,

$$\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle|0\rangle.$$

Then, as the second operation, applying $\mathbf{U}_\rho$ gives a superposition of all cosets of H_0,

$$\frac{1}{\sqrt{|T_0|}} \sum_{t \in T_0} |t + H_0\rangle|\rho(t)\rangle = \frac{1}{\sqrt{|T_0|}} \sum_{t \in T_0} (\tau_t|H_0\rangle)|\rho(t)\rangle.$$

Here T_0 denotes a transversal for H_0 in G. Applying, as the third operation, the Fourier transform on the first register produces the superposition

$$\begin{aligned}
|\Psi\rangle &= \frac{1}{\sqrt{|T_0|}} \sum_{t \in T_0} (\mathbf{F}_G \circ \tau_t|H_0\rangle)|\rho(t)\rangle \\
&= \frac{1}{\sqrt{|T_0|}} \sum_{t \in T_0} (\phi_t \circ \mathbf{F}_G|H_0\rangle)|\rho(t)\rangle \\
&= \frac{1}{\sqrt{|T_0|}} \sum_{t \in T_0} (\phi_t|H_0^\perp\rangle)|\rho(t)\rangle.
\end{aligned}$$

Since the operator ϕ_t changes only phases and not amplitudes, a measurement of $\phi_t|H_0^\perp\rangle$ gives the same probability distribution on the possible outcomes as a measurement of $|H_0^\perp\rangle$. It follows that the outcome $z = \mathcal{M}_1|\Psi\rangle$ is a random element of the orthogonal subgroup $H_0^\perp$. This completes our short proof of how the natural generalization of Simon's subroutine can be used to sample random elements of $H_0^\perp$.

The time needed to apply operator $\mathbf{U}_G$ is equal to twice the time to compute $\mathbf{F}_G$ plus the time to compute the function ρ. By a result of Kitaev [16], for all finite additive Abelian groups G, the Fourier transform $\mathbf{F}_G$ can be applied in polynomial time in $\log|G|$. However, his method applies the transform not with perfection, but only with arbitrary good precision (see [16] for details). Yet, this suffices to imply a $\mathcal{ZQP}$–algorithm for the Abelian subgroup problem.

A direct generalization of our $\mathcal{QP}$–algorithm would require the solutions to two problems.

The first problem is that we must be capable of computing the Fourier transform $\mathbf{F}_G$ exactly. Cleve [12] and Coppersmith [13], building on the work of Shor [17], showed that it can be applied exactly in polynomial time whenever G is of smooth order. Here the order of a group G is *smooth* if all its prime factors are at most $\log^c |G|$ for some fixed constant c. Thus, in that case we can also apply $\mathbf{U}_G$

efficiently and exactly, assuming we are given a polynomial-time (in $\log|G|$) algorithm to compute ρ.

The second problem is how to make certain that we find larger and larger subgroups of $H_0^\perp$ at each iteration until we eventually have a generating set for $H_0^\perp$. Suppose we have previously found the subset $Y \subseteq H_0^\perp$ and now we measure some element $z \in H_0^\perp$. For the group $G = \mathbb{Z}_2^n$, we ensured in Section 4.2, via Lemma 8, that z is the zero element of G if Y generates $H_0^\perp$, and otherwise $z \in H_0^\perp \setminus \langle Y \rangle$ is not generated by Y. Thus, in the latter case $Y \cup \{z\}$ generates a subgroup strictly larger than the one generated by Y itself.

Lemma 8 implies that if we can find a function χ defined on G such that χ equals 1 on exactly half the elements of $H_0^\perp$ and χ is 0 on the subgroup generated by $Y \subset H_0^\perp$ then we can ensure that z is nonzero. We can show that this implication holds not only with the above fraction $1/2$, but for any fraction $1/k$ where $k \le \log^c |G|$ for some fixed constant c.

We are currently investigating for which groups of smooth order we can find such a function χ since this would solve the second problem. If, in addition, there is an efficient algorithm to compute χ then this would imply a $\mathcal{QP}$–algorithm for the group under consideration.

As our final example of generalizing our $\mathcal{QP}$–algorithm for Simon's subgroup problem, consider the discrete logarithm problem defined as follows. For every prime p, let $\mathbb{Z}_p^\star$ denote the multiplicative cyclic group of the positive integers smaller than p. The *discrete logarithm problem* is given p, a generator ζ of $\mathbb{Z}_p^\star$, and an element $a \in \mathbb{Z}_p^\star$, find $0 \le r < p$ such that $\zeta^r = a$ in $\mathbb{Z}_p^\star$.

Shor gave in [17] a $\mathcal{ZQP}$–algorithm for this problem. In our language, his solution consists in a reduction to a problem equivalent to a special case of the Abelian subgroup problem, followed by an algorithm for that problem. Let $G = \mathbb{Z}_{p-1}^2$ and define function $\rho : G \to \mathbb{Z}_p^\star$ by

$$\rho((g_1, g_2)) = \zeta^{g_1} a^{g_2}$$

for $g = (g_1, g_2) \in G$. Let $H_0 \le G$ be the cyclic subgroup of order $p - 1$ generated by the element $(r, -1) = (r, p - 2)$. Then ρ is constant and distinct on each coset of H_0. The orthogonal subgroup $H_0^\perp$ has also order $p - 1$ and is generated by $(1, r)$. It is now easy to see that the discrete logarithm problem reduces to finding the unique element $(g_1, g_2) \in H_0^\perp$ for which $g_1 = 1$. We can show that if we are given a quantum algorithm to compute $\mathbf{F}_{p-1}$ exactly then we can find that unique element in worst-case polynomial time (in $\log p$) on a quantum computer. Here $\mathbf{F}_{p-1}$ denotes the quantum Fourier transform for the cyclic group $\mathbb{Z}_{p-1}$.

Theorem 12 ($\mathcal{QP}$–algorithm for Discrete Logarithms)
Let p be a prime and $\zeta \in \mathbb{Z}_p^\star$ be a generator. Then given a quantum algorithm to compute $\mathbf{F}_{p-1}$ exactly, there exists a $\mathcal{QP}$–algorithm that, for all $a \in \mathbb{Z}_p^\star$, finds $0 \le r < p - 1$ such that $\zeta^r = a$ in $\mathbb{Z}_p^\star$.

Let us end this paper by posing the open problem of finding a $\mathcal{QP}$–algorithm for prime factorization.

Acknowledgments

We are grateful to Michel Boyer and Alain Tapp for helpful comments. The second author thanks Joan Boyar for valuable discussions and for her interest in this work. This research was carried out while the second author was at the Laboratoire d'informatique théorique et quantique at Université de Montréal and he thanks the faculty and the students for their hospitality.

References

[1] A. Barenco, "Quantum physics and computers", *Contemporary Physics*, Vol. 38, 1996, pp. 357–389.

[2] A. Barenco, C.H. Bennett, R. Cleve, D.P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J.A. Smolin and H. Weinfurter, "Elementary gates for quantum computation", *Physical Review A*, Vol. 52, 1995, pp. 3457–3467.

[3] C.H. Bennett, "Logical reversibility of computation", *IBM Journal of Research and Development*, Vol. 17, 1973, pp. 525–532.

[4] E. Bernstein and U. Vazirani, "Quantum complexity theory", *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, 1993, pp. 11–20.

[5] E. Bernstein and U. Vazirani, "Quantum complexity theory", final version of [4]. To appear in *SIAM Journal on Computing*.

[6] A. Berthiaume, "Quantum computation", in *Complexity Theory Retrospective II*, L. Hemaspaandra and A. Selman (editors), Springer-Verlag, to appear.

[7] A. Berthiaume and G. Brassard, "The quantum challenge to structural complexity theory", *Proceedings of the 7th Annual IEEE Structure in Complexity Theory Conference*, 1992, pp. 132–137.

[8] A. Berthiaume and G. Brassard, "Oracle quantum computing", *Journal of Modern Optics*, Vol. 41, 1994, pp. 2521–2535.

[9] M. Boyer, G. Brassard, P. Høyer and A. Tapp, "Tight Bounds on Quantum Searching", *Proceedings of the 4th Workshop on Physics and Computation*, New England Complex Systems Institute, 1996, pp. 36–43. Available on Los Alamos e-Print archive (http://xxx.lanl.gov) as quant-ph/9605034.

[10] G. Brassard, "A quantum jump in computer science", in *Computer Science Today*, Lecture Notes in Computer Science, Vol. 1000, Springer-Verlag, 1995, pp. 1–14.

[11] G. Brassard and P. Høyer, "On the power of exact quantum polynomial time", unpublished, 1996. Available on Los Alamos e-Print archive (http://xxx.lanl.gov) as quant-ph/9612017.

[12] R. Cleve, "A note on computing Fourier transformation by quantum programs", Department of Computer Science, University of Calgary, 1994.

[13] D. Coppersmith, "An approximate Fourier transform useful in quantum factoring", IBM T.J. Watson Research Report RC 19642, Yorktown Heights, NY 10598, 1994.

[14] D. Deutsch and R. Jozsa, "Rapid solution of problems by quantum computation", *Proceedings of the Royal Society, London*, Vol. A439, 1992, pp. 553–558.

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

[16] A. Yu. Kitaev, "Quantum measurements and the Abelian stabilizer problem", manuscript, 1995. Available on Los Alamos e-Print archive (http://xxx.lanl.gov) as quant-ph/9511026.

[17] P.W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring", *Proceedings of the 35th Annual IEEE Symposium on the Foundations of Computer Science*, 1994, pp. 124–134. Final version to appear in *SIAM Journal on Computing* under title "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer".

[18] D.R. Simon, "On the power of quantum computation", *Proceedings of the 35th Annual IEEE Symposium on the Foundations of Computer Science*, 1994, pp. 116–123. Final version to appear in *SIAM Journal on Computing*.

23