An exact quantum order finding algorithm and its applications

Muhammad Imran
Institute of Mathematics, Department of Algebra, Budapest University of Technology and Economics, M"uegyetem rkp. 3., Budapest, H-1111, Hungary.
E-mail: mimran@math.bme.hu

Abstract
We present an efficient exact quantum algorithm for order finding problem when a multiple $m$ of the order $r$ is known. The algorithm consists of two main ingredients. The first ingredient is the exact quantum Fourier transform proposed by Mosca and Zalka in [MZ03]. The second ingredient is an amplitude amplification version of Brassard and Hoyer in [BH97] combined with some ideas from the exact discrete logarithm procedure by Mosca and Zalka in [MZ03]. As applications, we show how the algorithm derandomizes the quantum algorithm for primality testing proposed by Donis-Vela and Garcia-Escartin in [DVGET18], and serves as a subroutine of an efficient exact quantum algorithm for finding primitive elements in arbitrary finite fields.

Keywords: Exact quantum algorithm, Order finding algorithm, Primality testing, Primitive elements.

1 Introduction
Shor’s quantum algorithm [Sho97] can determine the order of group elements efficiently, and it serves as the main tool for factoring integers. However, Shor’s algorithm is polynomial-time in the expected sense, which means it may fail with a small probability and in the unlucky case may take a very long time to succeed, even may never terminate. The same case happens with Simon’s algorithm [Sim97]. However, Brassard and Hoyer, in [BH97], came up with an exact quantum polynomial time for Simon’s problem. The Brassard-Hoyer algorithm utilizes a modified version of Grover’s technique in [Gro97] to derandomize Simon’s algorithm. Specifically, they propose a method that, assuming that we can construct a superposition in which the total squared amplitude of the ”desired” constituents (intuitively, the probability of success) is $\frac{1}{2}$, boosts this success probability to 1.

The question about the existence of exact quantum replacements for bounded quantum error probabilistic algorithms is a natural question, as it is analogous to derandomizing probabilistic classical algorithms. Besides, some earliest quantum algorithms that demonstrate the power of quantum computers, such as Deutsch-Jozsa procedure [DJ92] and Bernstein-Vazirani problem [BV97], are exact. It is a difficult open question whether Shor’s factoring algorithm can be derandomized. In [MZ03], Mosca and Zalka successfully derandomize Shor’s algorithm for discrete logarithm problem in a cyclic group of known order. All previous exact quantum algorithms are uniform, which means the circuits for
the algorithms can be classically computed in time polynomial in the logarithm of the inputs, see [NO05] for the details of uniform quantum circuits. In [Mos02], Mosca proposed a slightly less general computational model in which it is allowed the gate $B_r$ with parameter $r$ when $r$ emerges during the computation. Using that model, Mosca successfully derandomized Shor’s factoring algorithm. The square-free decomposition algorithm of Lie et al. [LPDS12] also works in this computational model.

Here we consider the question whether Shor’s order finding algorithm can be derandomized in the uniform computational model assuming some knowledge. As knowing a multiple of the order $Z_{nm}$ would factor $m$ in randomized classical polynomial time, finding orders of group elements with a known multiple of the order is not necessarily as hard as factoring, so a multiple of the order may be a good candidate for such a help. An important example where this help is available is the case of computing multiplicative orders (and testing primitivity) of elements of finite fields.

The main results. In this paper, we show that, given a multiple of the order, we can adjust the total squared amplitude of the ”desired states” in Shor’s order finding algorithm to $\frac{1}{1}$ and use the Brassard-Hoyer amplitude amplification to obtain an exact algorithm, see section $3$. The idea of the exact version of Shor’s order finding algorithm presented in this paper can be generalized to an exact quantum algorithm for hidden subgroup problem in $Z_{nm}$, see [I22]. As applications, we show how the algorithm derandomizes the quantum algorithm for primality testing proposed by Donis-Vela and Garcia-Escartin in [DVGE18], serves as a subroutine of an efficient exact quantum algorithm for finding primitive elements in arbitrary finite fields in a slightly less general computational model proposed by Mosca in [Mos02].

2 Amplitude amplifications

Both Mosca-Zalka amplitude amplification and Brassard-Hoyer amplitude amplification are based on the generalization of Grover search algorithm in [BHT98]. We present a brief review of the general amplitude amplification discussed by Brassad, Hoyer and Tapp in [BHT98].

Given an algorithm $A$ using no measurement, the amplitude amplification is a method to boost the success probability of the algorithm $A$. On initial input $|0\rangle$, the algorithm $A$ returns a pure superposition $A|0\rangle = \sum_{i\in I} |i\rangle |\Gamma_i\rangle$ for some index set $I \subseteq \mathbb{Z}$. We consider $\chi: I \rightarrow \{0, 1\}$ a Boolean function that separates the desired outcome states (all states $|i\rangle |\Gamma_i\rangle$ with $\chi(i) = 1$) from the unwanted states (all states $|i\rangle |\Gamma_i\rangle$ with $\chi(i) = 0$) as follows. Let $A = \{i \in I \mid \chi(i) = 1\}$ and $B = \{i \in I \mid \chi(i) = 0\}$. We write $A|0\rangle = |\Gamma_A\rangle + |\Gamma_B\rangle$, where

$$|\Gamma_A\rangle = \sum_{i \in A} |i\rangle |\Gamma_i\rangle \text{ and } |\Gamma_B\rangle = \sum_{i \in B} |i\rangle |\Gamma_i\rangle.$$  

Hence the success probability of the algorithm $A$ is $a = \langle \Gamma_A | A|0\rangle = ||\Gamma_A||^2$. Therefore, the amplitude amplification operator for the algorithm $A$ is defined as

$$Q(A, \chi, \phi, \varphi) = -AS_0^\varphi A^{-1}S_0^\phi,$$  

where $S_0^\varphi$ and $S_0^\phi$ are phase changing operators defined by

$$S_0^\varphi |i\rangle |\Gamma_i\rangle = \begin{cases} 
\varphi |i\rangle |\Gamma_i\rangle & \text{if } \chi(i) = 1 \\
|i\rangle |\Gamma_i\rangle & \text{otherwise},
\end{cases} \quad \text{and} \quad S_0^\phi |i\rangle |\Gamma_i\rangle = \begin{cases} 
\phi |i\rangle |\Gamma_i\rangle & \text{iff } i = 0 \\
|i\rangle |\Gamma_i\rangle & \text{otherwise},
\end{cases}$$

2
with $\phi$ and $\varphi$ are complex number of unit length.

The operator $Q$ is a generalization of Grover’s iterations applied in his quantum search algorithm [Gro97]. Moreover, by setting $\phi = \varphi = -1$, we have for every $j \geq 0$,

$$Q^j A |0\rangle = k_j |\Gamma_a\rangle + l_j |\Gamma_b\rangle$$

where $k_j = \frac{1}{\sqrt{a}} \sin((2j + 1) \theta)$ and $l_j = \frac{1}{\sqrt{1 - a}} \cos((2j + 1) \theta)$, and $0 \leq \theta \leq \pi/2$ is defined so that $\sin^2 \theta = a = ||\Gamma_a||^2$.

A natural question to ask whether it is possible to boost the success probability to certainty. It turns out there are positive answers to this question. In [BH97], Brassard and Hoyer present an optimal value for the parameters $\phi$ and $\varphi$, namely $\phi = \varphi = \sqrt{-1}$, such that whenever the success probability of an algorithm $A$ is $\frac{1}{2}$, then one application of the amplitude amplification $Q$ boosts the success probability to 1. This is the approach that Brassard and Hoyer use to derandomize Simon’s algorithm. Another positive answer is also presented in [MZ03] by Mosca and Zalka. They use one application of $Q$ with parameters $\phi = \varphi = -1$ to increase the success probability $\frac{1}{4}$ of an algorithm $A$ to 1. They use this variant of amplitude amplification to present an exact quantum Fourier transform and derandomize Shor’s quantum algorithm for discrete logarithm. Therefore, one application of the exact quantum Fourier proposed by Mosca and Zalka requires three applications of the usual quantum Fourier transform.

As one may notice from some previous derandomizations, such as Simon’s algorithm and Shor’s discrete logarithm, the knowledge of the success probability of the algorithms makes the derandomizations possible. Therefore, in the next sections, we show how to use this amplitude amplification approaches to derandomize Shor’s order finding algorithm when a multiple of the order is known and to construct an exact algorithm for primitive finding problem.

3 Exact quantum order finding algorithm

The problem we consider is given an element group $x \in G$ and a multiple $m$ of the unknown order $r$ of $x$, determine the order $r$. The first part of the algorithm is the standard Fourier sampling. We use here an exact version based on the exact quantum Fourier transform of Mosca and Zalka [MZ03]. The standard Fourier sampling procedure maps $|0\rangle |0\rangle$ to $\sum_{k=0}^{m-1} |k\rangle |\Gamma_k\rangle$, where $|\Gamma_k\rangle = \frac{1}{m} \sum_{j=0}^{m-1} \omega^{kj} |x^j\rangle$ and $\omega = e^{2\pi i/m}$. Write $j$ as $j = j_0 + rj_1$ ($0 \leq j_0 \leq r - 1$). Then

$$|\Gamma_k\rangle = \begin{cases} \frac{1}{r} \sum_{j_0=0}^{r-1} \omega^{kj_0} |x^{j_0}\rangle & \text{if } \gcd(m,k) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

whence

$$|\Gamma_k|^2 = \begin{cases} \frac{1}{r} & \text{if } \gcd(m,k) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

In words, we have terms with $|k\rangle$ in the first register only for those $k$ which are multiples of $m/r$. Initially, any $k$ which is nonzero modulo $m$ is useful because $\gcd(k,m)$ is a proper divisor of $r$. We have $\sum_{k \neq 0} |\Gamma_k|^2 = 1 - \frac{1}{r}$. However, fortunately, if we already know a divisor $d$ of $r$ then those values $k$ that give us new information are the non-multiples of $\frac{m}{d}$. We have $\sum_{kd \neq 0} |\Gamma_k|^2 = 1 - \frac{d}{r}$. The point is we do not know $r$. 3
The second part of the algorithm is based on the discussion in the previous paragraph. We maintain a divisor \( d \) of \( r \). We construct iterations of a procedure that increase \( d \). Initially \( d := 1 \). As long as \( d < r \), we find \( k \) such that \( dk \mod m \neq 0 \). Then we replace \( d \) with \( \frac{m}{\gcd(m,k)} \) since this is another divisor of \( r \) greater than \( d \). Hence, \( d \) keeps increasing as long as \( d < r \) and it stops immediately when \( d = r \) as \( dk = 0 \mod m \) for all \( k \) if and only if \( d \) is a multiple of \( r \).

Assume \( d < r \). Let \( \text{rep}(dk) \) be the smallest positive integer representative of \( dk \mod m \). In this case, \( \text{rep}(dk) = d\frac{m}{r} \) for all \( k \). Then \( \text{rep}(dk) \) divides \( m \) and the \( \frac{m}{\text{rep}(dk)} - 1 = \frac{r}{d} - 1 \) positive integers of the form \( td\frac{m}{r} < m \) are the nonzero multiple of \( d\frac{m}{r} \) modulo \( m \). Note that if \( \frac{r}{d} \) is even, then the integers of the form \( td\frac{m}{r} \) with \( m/2 < td\frac{m}{r} < m \) represent just half of multiples of \( d\frac{m}{r} \) modulo \( m \). However, if \( \frac{r}{d} \) is odd, we need to add another multiple of \( d\frac{m}{r} \) modulo \( m \), say \( d\frac{m}{r} \), with weight \( \frac{1}{2} \). The problem is we do not know \( d\frac{m}{r} \). However, fortunately, for at least one integer \( 0 \leq j \leq \log_2 m \), namely for \( j = \lfloor \log_2 d\frac{m}{r} \rfloor \), the interval \( (0,2^j] \) contains only \( d\frac{m}{r} \) and no other multiple of \( d\frac{m}{r} \) as if \( j - 1 < \log_2 d\frac{m}{r} \leq j \) then \( d\frac{m}{r} \leq 2^j \) and \( 2d\frac{m}{r} > 2^j \).

| Algorithm 1 Exact quantum order finding algorithm |
|-----------------------------------------------|
| 1: Initialize: \( d \leftarrow 1 \), \( \text{Found} \leftarrow 1 \); |
| 2: while \( \text{Found} > 0 \) do |
| 3: for \( j = -1, \ldots, \lfloor \log_2 m \rfloor \) do |
| 4: \( \chi_j(k,b) = \left\{ \begin{array}{ll} 1 & \text{if } \text{rep}(dk) \geq \frac{r}{d} \text{ or } b = 1 \text{ and } 0 < \text{rep}(dk) \leq 2^j; \\ 0 & \text{otherwise}; \end{array} \right. \) |
| 5: \( U_j : |0\rangle|0\rangle|0\rangle|0\rangle \mapsto |\psi_j\rangle = \frac{1}{\sqrt{2^j}} \sum|k\rangle|\Gamma_k\rangle|0\rangle|\chi_j(k,b); \) \( \triangleright\triangleright\triangleright \) where \( k \in \{0,1,\ldots,m-1\} \), \( b \in \{0,1\} \). |
| 6: Apply the amplitude amplified version of \( U_j \) to obtain \( |\psi'_j\rangle = \sum|\chi_j(k,b)\rangle|\Gamma_k\rangle|0\rangle|\chi_j(k,b); \) |
| 7: Look at the \( |k\rangle\)-register; |
| 8: if \( dk \neq 0 \mod m \) then |
| 9: \( d \leftarrow \frac{m}{\gcd(m,k)} \); |
| 10: else |
| 11: \( \text{Found} \leftarrow \text{Found} - 1 \); |
| 12: end if |
| 13: end for |
| 14: end while |

Each round consists of iterations for \( j = -1, \ldots, \lfloor \log_2 m \rfloor \) instead of starting with index \( j = 0 \) to cover both cases when \( \frac{r}{d} \) is even and when \( \frac{r}{d} \) is odd. The case when \( \frac{r}{d} \) is even is covered at least once, when \( j = -1 \) where the interval \( (0,2^j] \) does not contain any integer. While the case when \( \frac{r}{d} \) is odd is covered at least once, when \( j = \lfloor \log_2 d\frac{m}{r} \rfloor \).

As in each round before termination, the size of \( d \) is increased by at least a factor 2 and it stops immediately when \( d = r \), we need at most \( \lfloor \log_2 r \rfloor \) rounds of iterations. The overall number of calls to the exact Fourier transform or its inverse is \( O(\log m \log r) \).

4 Applications

4.1 Exact quantum algorithm for primality testing

There are various efficient primality testing algorithms for a given large integer \( n \) based on the factorization of \( n - 1 \), see for example [Pra75]. The algorithms are based on the Lucas-Lehmer condition in [Leh27]. Donis-Vela and Garcia-Escartin in [DVGE18] present an alternative quantum algorithm for testing whether a given number \( n \) is a prime or not, without factoring \( n - 1 \).

The quantum algorithm proposed by Donis-Vela and Garcia-Escartin requires Shor’s order finding algorithm. In fact, Shor’s order finding algorithm is the only quantum part of the algorithm. The primality testing algorithm works as follows. To test primality of
an integer \( n \), we choose a random \( 1 < x < n \). The condition \( n \) to be composite is when \( \gcd(x, n) > 1 \) or \( x^{\frac{n-1}{2}} \neq \pm 1 \mod n \). If \( x^{\frac{n-1}{2}} \equiv -1 \mod n \), then we compute the order of \( x \). If the order of \( x \) is \( n-1 \), then \( n \) is prime. Therefore, we can use the exact algorithm to derandomize the quantum part of the primality testing algorithm because a multiple \( n-1 \) of the order of \( x \) is known.

### 4.2 Exact quantum algorithm for primitive finding problem

It is well known that the density of primitive elements in a finite field \( \mathbb{F}_q \) is large enough so that the simple method of choosing a small number of elements in \( \mathbb{F}_q \) at random is in fact a probabilistic polynomial time algorithm for finding primitive elements. However, the existence of a deterministic or even Las Vegas polynomial time algorithm for finding primitive elements in arbitrary finite fields is still an open question. Almost all known classical algorithms for testing a generator require factoring \( q-1 \). In this section, we present an exact quantum algorithm for finding primitive elements using the algorithm in section 2 as a subroutine.

Let \( x \) be an element of \( G = \mathbb{F}_q^* \). The exact algorithm consists of two parts. The first part is testing primitivity using the exact algorithm 1. The second part is an exact procedure to find an element in \( G \setminus \langle x \rangle \).

The procedure for the second part is the following. We start with the uniform superposition \( |G\rangle = \frac{1}{\sqrt{m}} \sum_{y \in G} |y\rangle \) where \( m = q-1 \). If \( x \) is an element of order \( r < m \), then

\[
\left| \frac{1}{\sqrt{m}} \sum_{y \in G \setminus \langle x \rangle} |y\rangle \right|^2 = 1 - \frac{r}{m} \geq 1/2,
\]

and one can adjust the success probability to \( \frac{1}{2} \) as follows. Let \( B_r \) denote the unitary operator that rotates \( |0\rangle \) to \( \sqrt{1 - \frac{m}{2r(m-r)}} |0\rangle + \sqrt{\frac{m}{2r(m-r)}} |1\rangle \). Therefore, by attaching an additional one qubit register to \( |G\rangle = \frac{1}{\sqrt{m}} \sum_{y \in G} |y\rangle |0\rangle \) and applying \( B_r \) on the second register, we have the superposition

\[
\frac{1}{\sqrt{m}} \sum_{y \in G} |y\rangle \left( \sqrt{1 - \frac{m}{2r(m-r)}} |0\rangle + \sqrt{\frac{m}{2r(m-r)}} |1\rangle \right).
\]

Let \( \chi : G \times \{0,1\} \to \{0,1\} \) be the Boolean function defined as \( \chi(y,b) = 1 \) if and only if \( y \in G \setminus \langle x \rangle \) and \( b = 1 \). Therefore, the total squared amplitudes of states \( |y\rangle |b\rangle \) with \( \chi(y,b) = 1 \) is \( \frac{1}{2} \). Hence, we apply Brassard-Hoyer amplitude amplification to obtain a state \( |y\rangle |1\rangle \) with probability 1 where \( y \in G \setminus \langle x \rangle \). Let \( \mathcal{P} \) denote the exact procedure discussed previously.

Now we can find a primitive element in \( \mathbb{F}_q^* \) as follows. Let \( x \in \mathbb{F}_q^* \). For each round, we compute the order of \( x \) using the exact algorithm 1. If the order of \( x \) is less than \( q-1 \), then we do the following procedure: find an element \( y \in \mathbb{F}_q^* \setminus \langle x \rangle \) and then replace \( x \) with \( \gcd(x,y) \).

**Algorithm 2** Algorithm for computing the probability of primitive elements

1. Choose an element \( x \in \mathbb{F}_q^* \).
2. Compute the order \( r \) of \( x \) using **algorithm 1**;
3. while \( r < q-1 \) do
   4. Use the procedure \( \mathcal{P} \) to obtain \( y \in \mathbb{F}_q^* \setminus \langle x \rangle \);
   5. \( x \leftarrow \gcd(x,y) \);
6. end while
As in each round, the size of $G$ is decreased by at least a factor of 2, we need at most $\log_2 q$ rounds to find a primitive element in $\mathbb{F}_q^*$. Therefore, the overall number of calls to the exact Fourier transform is $O(\log^3 q)$.

5 Discussions

Shor’s quantum algorithm for order finding in modulo $m$ requires $O((\log \log m) \log^3 m)$ quantum operations with $O(\log \log m)$ uses of modular exponentiations. The main subroutines of Shor’s order finding algorithm are modular exponentiation and quantum Fourier transform. Modular exponentiation needs $O(\log m)$ multiplications, hence $O(\log^3 m)$ is the total complexity for a modular exponentiation while quantum Fourier transform is quadratic in $\log m$, see [Sho97].

On the other hand, the exact algorithm presented in section 3 needs $O(\log^2 m)$ exact quantum Fourier transform of Mosca-Zalka [MZ03] and the same complexity for modular exponentiations as in Shor’s algorithm. As observed in section 2, one call to the exact quantum Fourier transform requires three calls to the standard quantum Fourier transform, then the number of the standard quantum Fourier transform used in the exact algorithm is still $O(\log^2 m)$. Thus, the exact algorithm requires $O(\log^4 m)$ quantum operations. Therefore, the overhead of the exact version from the standard Shor’s algorithm is $O(\log m)$.

The exact algorithm in section 3 does not imply the existence of an exact quantum factoring algorithm because the possible multiple $\varphi(n)$ of the order of $x$ is unknown in general. However, the algorithm can be used to derandomize several quantum algorithms such as primality testing and primitive finding algorithm where in the latter is the first deterministic algorithm for finding primitive element in any finite field $\mathbb{F}_p$.

Acknowledgement. The author is grateful to Gabor Ivanyos for his helpful comments and suggestions.

References

[BH97] G. Brassard and P. Høyer. An exact quantum polynomial-time algorithm for Simon’s problem. In ISTCS 97, pages 12–23, 1997.

[BHT98] G. Brassard, P. Høyer, and A. Tapp. Quantum counting. In ICALP 98, 1998.

[BV97] E. Bernstein and U. Vazirani. Quantum complexity theory. SIAM J. Comput., 26:1411–1473, 1997.

[DJ92] D. Deutsch and R. Jozsa. Rapid solution of problems by quantum computation. Proc. Math. Phys. Eng. Sci., 439:553 – 558, 1992.

[DVGE18] A. Donis-Velaand and JC. Garcia-Escartin. A quantum primality test with order finding. Quantum Inf. Comput., 18(13-14):1143–1151, 2018.

[Gro97] L.K. Grover. Quantum mechanics helps in searching for a needle in a haystack. Phys. Rev. Lett., 79(2):325–328, 1997.

[I22] M. Imran and G. Ivanyos. An exact quantum hidden subgroup algorithm and its applications to solvable groups. Technical Report 2202.04047 [quant-ph], arXiv, 2022.
[Leh27] Derrick H Lehmer. Tests for primality by the converse of fermat’s theorem. *Bulletin of the American Mathematical Society*, 33(3):327–340, 1927.

[LPDS12] J. Li, X. Peng, J. Du, and D. Suter. An efficient exact quantum algorithm for the integer square-free decomposition problem. *Sci. Rep.*, 2:1–5, 2012.

[Mos02] M. Mosca. On the quantum derandomization of algorithms. Presentation at the MSRI workshop on Quantum Information Processing, https://www.msri.org/workshops/204/schedules/1233, 2002.

[MZ03] M. Mosca and C. Zalka. Exact quantum Fourier transforms and discrete logarithm algorithms. *Int. J. Quantum Inf.*, 02:91–100, 2003.

[NO05] H. Nishimura and M. Ozawa. Uniformity of quantum circuit families for error-free algorithms. *Theor. Comput. Sci.*, 332:487–496, 2005.

[Pra75] Vaughan R Pratt. Every prime has a succinct certificate. *SIAM Journal on Computing*, 4(3):214–220, 1975.

[Sho97] P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[Sim97] D. Simon. On the power of quantum computation. *SIAM J. Comput.*, 26:1474–1483, 1997.