Improving the Success Probability for Shor's Factorization Algorithm

Guoliang Xu, Daowen Qiu, Xiangfu Zou and Jozef Gruska

Abstract In Shor's factorization algorithm (SFA), the task is to find a non-trivial factor of a given composite integer N. Briefly said, SFA works as follows. It chooses randomly an integer $y < N$ and checks whether y and N are co-primes. If y is co-prime with N, then SFA runs a special quantum subroutine to obtain the order $2r$ of N with a certain probability ($2r$ is here an integer). In the original SFA and all previous SFAs, if $2r$ was an even integer and $y^r \not\equiv -1(\text{mod } N)$, then SFA used y and $2r$ to get a non-trivial factor of N. However, if the result r' obtained by the quantum order finding subroutine was not $2r$, or $2r$ was not an even integer, or $y^r \equiv -1(\text{mod } N)$, then the quantum subroutine had to be run again (and perhaps again and again). In this paper, we show that the three constraints are strong and the success probability for the quantum subroutine can be improved. In general, if a non-trivial factor of N can be got, we can call the result r' an *available* result. Naturally, two issues arise: (1) If one of these constraints does not hold, whether these results r' can also be used to make SFA succeed sometimes? (2) If there exist some other *available* results, then what is the success probability when these results are considered? This paper proves that some factorization results are still *available* or possible even if not all of the above constraints are met, and, in addition, that a new success probability can be bigger than those of the previous SFAs. Finally, in order to demonstrate a potential of our approach, we consider factorizationof those

G. Xu · D. Qiu (✉)
Institute of Computer Science Theory, School of Data and Computer Science,
Sun Yat-sen University, Guangzhou 510006, China
e-mail: issqdw@mail.sysu.edu.cn

D. Qiu
SQIG–Instituto de Telecomunicações, Departamento de Matemática,
Instituto Superior Técnico, Av. Rovisco Pais, 1049-001 Lisbon, Portugal

X. Zou
School of Mathematics and Computational Science, Wuyi University,
Jiangmen 529020, China

J. Gruska
Masaryk University, Botanicka 68a, 60200 Brno, Czech Republic

integers N that are used as moduli for RSA, that is those N that are products of two safe primes, and we show that in this case the fault probability can be reduced to $O(1/N)$ with our method.

1 Introduction

It is already well known that quantum computers can solve certain problems more efficiently than classical computers. Shor's factoring algorithm (SFA)—a probabilistic algorithm—is one of the examples where quantum algorithms can outperform even the most efficient known classical algorithms [1]. SFA finds a non-trivial factor of a big composite integer N, exponentially faster than the best known classical algorithms [1].

1.1 The State of the Art

SFA uses a special order-finding algorithm (OFA), which needs a quantum oracle, as a subroutine [1, 2] in order to factorize an N. In particular, SFA first chooses randomly a co-prime $y < N$. Then SFA calls OFA to obtain the order $2r$ of y with a known probability ($2r$ is an integer). Finally, if $2r$ is an even and $y^r \not\equiv -1 (\text{mod } N)$, one can use y and r to obtain a non-trivial factor of N [1, 2].

In the previous approaches [1–6], in order to make SFA to succeed, a chosen *good* co-prime y should satisfy the following conditions:

$$\textit{The order } 2r \textit{ of } y \textit{ is even, i.e., } r \textit{ is an integer;} \tag{1}$$

$$y^r \not\equiv -1 (\text{mod } N). \tag{2}$$

If a *not good* enough y is selected, OFA needs to choose another co-prime and repeat computation again. In the worst case, the probability for y being *good* could be raised from $1/2$, as specified in Refs. [1, 2], to $3/4$ by choosing y having Jacobi-Symbol -1 (Leander'2002)[5]. The reason behind is that numbers of this form always satisfy Eq. (1). In particular, such an improvement of SFA implies that the success probability will be 1 for integers used as moduli in the RSA crypto protocol [7] (The reason being that in such a case the integer to be factorized is known to be the product of two safe primes).

In 2015, Lawson [4] showed that a factor of a given N can be found from a square co-prime, providing it could have been found from its (non-square) root (If $y = a^b$ where a, b are integers and a is a non-square integer, then a is called the root of y), for which it is not necessary to consider odd orders. In addition, their derivations and calculations showed that avoiding non-square co-primes raises the probability of success of SFA lower bounded by $\left(1 - \frac{1}{4\sqrt{N}}\right)^{-1} - 1 = \frac{1}{4\sqrt{N}-1} \in O\left(\frac{1}{\sqrt{N}}\right)$ [4].

Using y as the input, OFA succeeds if

$$r' \in \{r, 2r\}. \tag{3}$$

Note that $r' = r$ is considered as an *available* result, in that sense that the order we wanted to find is even and then its factor 2 may be reduced. Furthermore, OFA needs to call the oracle $O((\log N)^3)$ times and therefore succeeds with the probability $O(1)$ [2]. This result implies that the success probability of OFA is $O\left(\frac{1}{(\log N)^3}\right)$.

As a probabilistic algorithm, the success probability depends on the integer N to be factorized, on the co-prime y chosen and on the result of the quantum order finding subroutine. Since N is an input, it is the choice of a co-prime y that determines the success probability of SFA. Some ways how to choose good co-primes from the above point of view, are specified in the Refs. [4, 5]. This paper makes a full use of these results, and this way raises the success probability of SFA.

Recently, Refs. [3, 6, 8] analysed various simulations and the experimental realizations of SFA. Their results showed that SFA has indeed potential for practical applications and the quantum subroutine of SFA still is the most complex part of it. That also implies that to reduce the fault probability of SFA is not only interesting, but also practically important task.

1.2 Our Contributions

In order to raise the success probability of SFAs, our idea is to loose Eqs. (2), (3) as much as possible. In order to do that we will consider the following two problems:

- (P1) If Eq. (2) or Eq. (3) does not hold, what other results can be used to make SFA to succeed?
- (P2) If the answer of (P1) is yes, then what is the probability that needed results are obtained?

To our best knowledge, no one dealt with such problems so far. Moreover, a proper answer to (P1) is clearly not obvious, and the key problem is therefore to find some other *useful* results. In this paper we can do that, and as a consequence, we can improve efficiency of SFAs and to give a lower bound for the raising success probability of SFA.

In this paper we show that the SFA may get a non-trivial factor of N also from the more general results r'. Namely, even if one of the conditions Eq. (2) or Eq. (3) does not hold, SFA may still succeed. Then, we apply the result obtained to improve SFA and we will analyze the raising probability. In particular, for the widely-used moduli N for the RSA protocol [7], being the product of two safe primes, the fault probability of which is $O(1/\sqrt{N})$ in Leander' 2002 can be reduced to $O(1/N)$ with our result.

1.3 An Overview

The rest of this paper goes as follows. We first introduce some necessary notations, concepts and results needed in the rest of the paper, in Sect. 2. Afterwards, a generalization of the use of the usual quantum order-finding subroutine in SFA is provided in Sect. 3. As the next, in Sect. 4, we improve efficiency, namely success probability, of SFA using the results from Sect. 3. Finally, we apply our results to factorization of moduli of RSA protocols in Sect. 5, see [7]. Finally, some conclusions are discussed in Sect. 6.

2 Preliminaries

In this section, we first provide some notations, concepts and basic related results from discrete mathematics. Afterwards, we present the main results of this paper.

2.1 Basic Notations, Concepts and Algebraic Results

Notations needed:

(1) Any odd, composite natural integer N has a unique prime factorization

$$N = \prod_{i=1}^{k} p_i^{\alpha_i} \tag{4}$$

where $k > 1$ (If $k = 1$ and $\alpha_1 = 1$, then N is prime; If $k = 1$ and $\alpha_1 > 1$, integer factorization can be done using a classical algorithm [2]), $2 < p_1 < p_2 < \ldots < p_k$ and $\alpha_i \geq 1$ for all $i \leq k$.

(2) $\mathrm{ord}_N(y) = 2r$ denotes the order of y in $\mathbb{Z}_N^*$, i.e., the smallest positive integer $2r$ satisfying $y^{2r} \equiv 1(\mathrm{mod}\ N)$. We can know that $\mathrm{ord}_N(y) = 2\mathrm{ord}_N(y^2)$ if r is an integer. Unless explicitly stated, the order of y means the order of y in $\mathbb{Z}_N^*$.

(3) r_i is the order of y in $\mathbb{Z}_{p_i^{\alpha_i}}^*$. $\widetilde{r}$ is the smaller integer of the set $\{r, r/2\}$ where r is an integer. Namely,

$$\widetilde{r} = \begin{cases} r, & \text{if } r \text{ is odd;} \\ r/2, & \text{if } r \text{ is even.} \end{cases} \tag{5}$$

r' denotes a result of the quantum subroutine in SFA. That is, r' is the denominator of the fraction obtained by the continued fractions (CF) algorithm [2]. When the fraction is 0, let $r' = 1$.

(4) The greatest common divisor of integers a and b is denoted by $\gcd(a, b)$. The least common multiple of a and b is denoted by $\operatorname{lcm}\{a, b\}$.
(5) $a \mid b$ means a divides b, and $a \nmid b$ means a does not divide b. $a \mid b \mid c$ means that a divides b and b divides c.
(6) $\varphi(N)$ is Euler phi function and is defined to be the number of non-negative integers less than N which are co-prime to N. In particular, let $\varphi(1) = 1$.
(7) $R = \{r' \mid \text{There exists an index } i \text{ such that } \widetilde{r_i}|r'|2r\}$, and $R_1 = \{r' \mid r' \in R$ and $r' < r\}$. We can see that $R_1 = R - \{r, 2r\}$.

Basic results needed:

(1) Denote
$$\mathbb{Z}^*_{p_i^{\alpha_i}} = \{g^{s_i}_{p_i^{\alpha_i}} (\text{mod } p_i^{\alpha_i}) | 1 \le s_i \le \varphi(p_i^{\alpha_i})\}, \tag{6}$$
where $g_{p_i^{\alpha_i}}$ is a generator of $\mathbb{Z}^*_{p_i^{\alpha_i}}$.
(2) Suppose that y is chosen randomly such that $\gcd(y, N) = 1$. From that it follows that $\gcd(y, p_i^{\alpha_i}) = 1$. In addition,
$$\operatorname{ord}_N(y) = 2r = \operatorname{lcm}\{r_i \mid 1 \le i \le k\}. \tag{7}$$
Moreover, if r is an integer, Eq. (7) implies that at least one of r_i is even.
(3) Let $y \equiv g^{s_i}_{p_i^{\alpha_i}} (\text{mod } p_i^{\alpha_i}) \in \mathbb{Z}^*_{p_i^{\alpha_i}}$ and $\operatorname{ord}_{p_i^{\alpha_i}}(y) = r_i = \operatorname{ord}_{p_i^{\alpha_i}}(g^{s_i}_{p_i^{\alpha_i}})$. Then, $\varphi(p_i^{\alpha_i})|s_i r_i$, and, we have
$$r_i = \frac{\varphi(p_i^{\alpha_i})}{\gcd(\varphi(p_i^{\alpha_i}), s_i)}. \tag{8}$$
Thus $\operatorname{ord}_{p_i^{\alpha_i}}(y^2) = \widetilde{r_i}$. If r is an integer,
$$r = \operatorname{lcm}\{\widetilde{r_i} \mid 1 \le i \le k\}. \tag{9}$$

Comment: In this paper, we mainly improve the following three versions of SFA [1, 2, 5]. We will call them as Algorithms 1, 2 and 3, respectively. For a more thorough introduction and analysis to SFA, see Refs. [1, 2, 5].

2.2 Our Main Results

The answers to problems (P1) and (P2) given in this paper go as follows.

- (A1) We generalize Eqs. (2), (3) to

$$y^r \not\equiv -1 (\text{mod } N) \tag{10}$$

and

$$r' \in R, \tag{11}$$

or

$$y^r \equiv -1 (\text{mod } N) \tag{12}$$

and

$$r' \in R_1. \tag{13}$$

- (A2) The raising success probability of each quantum order-finding subroutines will then be

$$O\left(\sum_{r' \in R_1} \frac{\varphi(r')}{2r}\right) \tag{14}$$

  This is to be compared with Algorithm 3.

Finally, as already mentioned, we show that when factorization is the task for those integers N that are products of two safe primes, then the fault probability can be reduced to $O(1/N)$.

Example 1 Let $N = 77,\ y = 2$ with order $2r = 30$. Almost all possible results $r' \in \{1, 2, 3, 5, 6, 10, 15, 30\}$. Then, the order of 2 in $\mathbb{Z}_7^*$ $r_1 = 3$, and the order of 2 in $\mathbb{Z}_{11}^*$ $r_2 = 10$. In the previous approaches [1–6], $r' \in \{15, 30\}$ is usable. With our method, $r' \in R = \{3, 5, 6, 10, 15, 30\}$ is usable. If $y = 24$ with the order $2r = 30$ is chosen. Almost all possible result $r' \in \{1, 2, 3, 5, 6, 10, 15, 30\}$. Then, the order of 24 in $\mathbb{Z}_7^*$ $r_1 = 6$, and the order of 24 in $\mathbb{Z}_{11}^*$ $r_2 = 10$. In the previous approaches [1–6], no r' is usable, and SFA fails. With our method, $r' \in R = \{3, 5, 6, 10\}$ is usable. By that the example is finished. □

3 An Analysis of Possible Impacts of Possible Outcomes of Quantum Subroutines in SFA

In this section we analyze the possible impacts of possible outcomes of quantum subroutines in SFA. At first we prove a theorem (and later analyze its impact) which deals with specific *availability* results of the quantum subroutine. Afterwards, we test the approach used for the case that chosen co-prime is an integer with Jacobi-Symbol -1. This is of importance since in such a case the even order, i.e., Eq. (1), can be ensured [5]. Furthermore, the result r' being a factor of the even order can be also ensured with a high probability as desired [2].

3.1 Outcomes of the Order-Finding Subroutine that Are Good Enough

In this subsection, we will make clear that even some more general results of the quantum order-finding subroutine in SFA will be good enough for factorization in some cases.

If $y^{2r} \equiv 1 (\bmod N)$, then $N \mid (y^{2r} - 1)$ implies

$$\prod_{i \in S_1} p_i^{\alpha_i} \mid (y^r - 1) \tag{15}$$

and

$$\prod_{i \in S_2} p_i^{\alpha_i} \mid (y^r + 1). \tag{16}$$

The sets S_0 and S_1 can be uniquely determined as follows. $S_1 \cap S_2 = \varnothing$ because $(y^r + 1) - (y^r - 1) = 2$ implies $\gcd(y^r + 1, y^r - 1) \le 2 < p_i$. Furthermore, $S_1 \cup S_2 = \{i \mid 1 \le i \le k\}$ and $S_2 \ne \varnothing$, since $S_2 = \varnothing$ implies $S_1 = \{i \mid 1 \le i \le k\}$ and $N \mid (y^r - 1)$, in contradiction to $\mathrm{ord}_N(y) = 2r$. SFA then tells that it would yield factors from $\gcd(y^r \pm 1, N)$ if $S_1 \ne \varnothing$ and that it will fail if not.

Alternatively, we explore a possibility to obtain a factor of N using a factor of the order. Obviously, an immediate benefit is that we do not need unnecessary repetitions of the order finding subroutine. Moreover, that implies that SFA may succeed even if $S_1 = \varnothing$. In order to show that the following lemma will be needed.

Lemma 1 *If $r_i = ord_{p_i^{\alpha_i}}(y)$, where $p_i > 2$ is a prime, then $y^{\widetilde{r_i}} \equiv \pm 1 (mod\ p_i^{\alpha_i})$.*

Proof If r_i is odd, the result is straightforward. If r_i is even, then

$$p_i^{\alpha_i} \mid (y^{r_i} - 1) = (y^{r_i/2} - 1)(y^{r_i/2} + 1). \tag{17}$$

Note that $p_i^{\alpha_i} \nmid (y^{r_i/2} - 1)$, because $p_i^{\alpha_i} \mid (y^{r_i/2} - 1)$ implies that $\mathrm{ord}_{p_i^{\alpha_i}}(y)$ divides $r_i/2$, in contradiction to $r_i = \mathrm{ord}_{p_i^{\alpha_i}}(y)$. If $p_i^{\alpha_i} \nmid (y^{r_i/2} + 1)$, then $p_i \mid \gcd (y^{r_i/2} - 1, y^{r_i/2} + 1)$ implies that

$$2 < p_i \le \gcd(y^{r_i/2} - 1, y^{r_i/2} + 1) \le (y^{r_i/2} + 1) - (y^{r_i/2} - 1) = 2. \tag{18}$$

However, $2 < 2$ is a contradiction and therefore the proof is finished. □

Using Lemma 1, we can get the following theorem.

Theorem 1 *If $r' \le r$ and $r' \in R$, then a non-trivial factor of N can be obtained from $\gcd(y^{r'} \pm 1, N)$, with the exception of the case that $S_1 = \varnothing$ and $r' = r$.*

Proof According to Lemma 1, $\widetilde{r_i} \mid r' \mid 2r$ implies

$$y^{r'} = (y^{\widetilde{r_i}})^{r'/\widetilde{r_i}} \equiv (\pm 1)^{r'/\widetilde{r_i}} \equiv \pm 1 (\text{mod } p_i^{\alpha_i}), \tag{19}$$

i.e.,

$$p_i^{\alpha_i} \mid (y^{r'} - 1) \text{ or } p_i^{\alpha_i} \mid (y^{r'} + 1). \tag{20}$$

Note that $S_1 = \varnothing$ and the order of y is $2r$ implies that r is the least value such that $y^r \equiv -1(\text{mod } N)$. In fact, if there exists a $\hat{r}$ with $\hat{r} < r$, such that $y^{\hat{r}} \equiv -1(\text{mod } N)$, $y^{2\hat{r}} \equiv 1(\text{mod } N)$ and $2\hat{r} < 2r$, in contradiction to $\text{ord}_N(y) = 2r$. Thus, $S_1 = \varnothing$ and $r' < r$ leads to

$$N \nmid (y^{r'} - 1) \text{ and } N \nmid (y^{r'} + 1). \tag{21}$$

At the same time, $S_1 \neq \varnothing$ and $r' \leq r$ also implies Eq. (21).

According to Eqs. (20), (21), at least one non-trivial factor of N can be got from $\gcd(y^{r'} \pm 1, N)$ except the case that it holds $S_1 = \varnothing$ and $r' = r$. By that the proof is finished. □

In order to understand consequences of Theorem 1 better, we provide and analyze two examples. Example 2 is the case that $y^r \not\equiv -1(\text{mod } N)$, and Example 3 is the case that $y^r \equiv -1(\text{mod } N)$. Observe that the Jacobi-Symbol of the co-prime y in Example 3 is not -1, but it is enough to illustrate usefulness of Theorem 1.

Example 2 Let $N = 77$, $y = 2$ with order $2r = 30$. Consider the result $r' \in \{1, 2, 3, 5, 6, 10, 15, 30\}$. Then, Algorithm 1 yields the order 30 from $\{15, 30\}$, and gets two non-trivial factors of 77 from $\gcd(2^{15} \pm 1, 77)$. However, according to Theorem 1, we can get a non-trivial factor of 77 from the set $\{3, 5, 6, 10, 15, 30\}$ by $\gcd(2^{\widetilde{r'}} \pm 1, 77)$. That is because the order 3 of 2 in $\mathbb{Z}_7^*$ or half the order 10 of 2 in $\mathbb{Z}_{11}^*$ divides $r' \in \{3, 5, 6, 10, 15, 30\}$. This way the example is finished. □

Example 3 Choose $y = 24$ with the order $2r = 30$ in Example 2. Consider the result $r' \in \{1, 2, 3, 5, 6, 10, 15, 30\}$. Then, Algorithm 1 yields the order 30 from $\{15, 30\}$ and fails, since $24^{15} \equiv -1(\text{mod } 77)$ which does not satisfy Eq. (2). However, according to Theorem 1, SFA can be used to get a non-trivial factor of 77 from the set $\{3, 5, 6, 10\}$ by $\gcd(2^3 \pm 1, 77)$ or $\gcd(2^5 \pm 1, 77)$. That is because the half of the order 6 of 24 in $\mathbb{Z}_7^*$ or half the order 10 of 24 in $\mathbb{Z}_{11}^*$ divides $r' \in \{3, 5, 6, 10\}$ and $r' \neq r$. By that the example is finished. □

3.2 A Test and Its Efficiency

In this subsection, we give a Test algorithm which can help to see whether SFA succeeds. Then, we analyze its efficiency in SFA. The Test algorithm is as follows.

Algorithm 4 *A Test Algorithm*

Inputs: (1) A big integer N; (2) A co-prime y; (3) An integer r' (to be a result of the order finding subroutine)

Outputs: Either an integer to be a non-trivial factor of N or "A bad result"or "A bad co-prime".

- (T.1) If r' is even, compute $\gcd(y^{r'/2} \pm 1, N)$ and $\gcd(y^{r'} + 1, N)$; If r' is odd, compute $\gcd(y^{r'} \pm 1, N)$. If one of these is a non-trivial factor, return that factor.
- (T.2) If $y^{r'} \not\equiv 1(\mathrm{mod}\ N)$, output "A bad result".
- (T.3) Output "A bad co-prime". □

By Theorem 1, a *good* result r' is the multiple of a $\widetilde{r_i}$ and a factor of $2r$. Note that $\{x|x \text{ is a factor of } 2r\} = \{x|x \text{ is a factor of } r\} \cup \{2x|x \text{ is a factor of } r\}$. Because $r = \mathrm{lcm}\{\widetilde{r_i}|1 \leq i \leq k\} = \mathrm{lcm}\{\mathrm{ord}_{p_i^{\alpha_i}}(y^2)|1 \leq i \leq k\} = \mathrm{ord}_N(y^2)$, $\{x|x \text{ is a factor of } r\}$ is enough to get the non-trivial factor of N. In fact, that a factor can be obtained by $\gcd(y^{r'} \pm 1, N)$ implies that it can be obtained by $\gcd(y^{\widetilde{r'}} \pm 1, N)$ or $\gcd(y^{2\widetilde{r'}} + 1, N)$. It can therefore be seen that $\widetilde{r'}$ corresponds to r'_{y^2} which denotes the result of a single quantum process of SFA with a black box $U_{y^2,N}$ which performs the transformation $|j\rangle|k\rangle \rightarrow |j\rangle|y^{2j}k(\mathrm{mod}\ N)\rangle$. Consequently, $\{x|x \text{ is a factor of } 2r\}$ can be reduced to $\{x|x \text{ is a factor of } r\}$. This suggests that to exploit factors of $\mathrm{ord}_N(y^2) = r$ is more convenient than to make use of the fact $\mathrm{ord}_N(y) = 2r$.

Now, we start our analysis.

First, according to Refs. [1, 2], a single quantum subroutine of SFA succeeds if Eqs. (1)–(3) hold. This is also the meaning of r' being *available* in Refs. [1, 2].

According to Eqs. (1)–(3), the probability of r' being *available* is

$$\Pr(y^r \not\equiv -1(\mathrm{mod}\ N)) \sum_{r' \in \{r,2r\}} \Pr(r'). \tag{22}$$

Here,

$$\Pr(r') \geq \sum_{s \in \mathbb{Z}^*_{r'}} \Pr\left(|\frac{s}{r'} - \frac{j}{2^t}| \leq \frac{1}{2^{t+1}}\right), \tag{23}$$

and we know that

$$\Pr\left(|\frac{s}{r'} - \frac{j}{2^t}| \leq \frac{1}{2^{t+1}}\right) \in O\left(\frac{1}{2r}\right) \tag{24}$$

where $s \in \mathbb{Z}^*_{r'}$ [1, 2]. Therefore, in Eq. (22),

$$\sum_{r' \in \{r,2r\}} \Pr(r') \in O\left(\sum_{r' \in \{r,2r\}} \frac{\varphi(r')}{2r}\right). \tag{25}$$

Test algorithm focuses on Eq. (22) to improve SFA further. As we have seen, Eqs. (2), (3) are generalized to Eqs. (10), (11) or Eqs. (12), (13). So far, we have adapted Eq. (22) to the sum of

$$\Pr(y^r \not\equiv -1 (\bmod N)) \sum_{r' \in R} \Pr(r') \tag{26}$$

and

$$\Pr(y^r \equiv -1 (\bmod N)) \sum_{r' \in R_1} \Pr(r'). \tag{27}$$

This sum minus Eq. (22) implies

$$\sum_{r' \in R_1} \Pr(r') \in O\left(\sum_{r' \in R_1} \frac{\varphi(r')}{2r}\right). \tag{28}$$

As a result, Eq. (28) implies that the raising success probability increases in SFA does not decrease in SFA when Algorithm 4 is used.

4 The Raising Success Probability

In this section, we estimate the efficiency of using the Test algorithm in SFA.

The success probability of each quantum process is the probability of r' being *available* that is good enough. It consists of three parts: $|2rj \pmod{2^t}| \leq r (0 \leq j \leq 2^t - 1)$, as well as constraints on y and r'. We present the success probability of the single quantum process of SFA for Algorithms 1, 2, 3 and Algorithm 3 with the Test algorithm, respectively. Here, δ_1 and δ_2 are possible increments which depend on a specific N, and $\delta_1 = \delta_2 = 0$ is the lower bound of the corresponding probability. Moreover, ε can be set up as required in advance.

The success probabilities of each quantum process for Algorithms 1, 2, 3 and Algorithm 3 with Test algorithm are determined as follows.

- Algorithm 1 [1]. The probability of $|2rj \pmod{2^t}| \leq r (0 \leq j \leq 2^t - 1)$ is asymptotically bounded from below by $4/\pi^2$ [1]. The probability of y being *good*—satisfying Eqs. (1), (2)—is bigger than $1/2$ and the probability of $r' \in \{r, 2r\}$ is $\varphi(r)/r$ [1]. Thus, the success probability is bounded from below by

$$\frac{4}{\pi^2} \times \left(\frac{1}{2} + \delta_1\right) \times \frac{\varphi(r) + \varphi(2r)}{2r}, 0 \leq \delta_1 \leq \frac{1}{2}. \tag{29}$$

Remark 1 Note that

$$\varphi(r) + \varphi(2r) = \sum_{r' \in \{r, 2r\}} \varphi(r') = \sum_{r' \in R - R_1} \varphi(r'). \tag{30}$$

Furthermore, if $\gcd(2, r) = 1$, then $\varphi(2r) = \varphi(2)\varphi(r) = \varphi(r)$ [2]. If $\gcd(2, r) = 2$, we know that

$$\{x|\gcd(x,2r)=1\}=\{x|\gcd(x,r)=1\}\cup\{r+x|\gcd(x,r)=1\}, \tag{31}$$

where $1 \leq x \leq r-1$; Therefore, $\varphi(2r)=2\varphi(r)$. As a result,

$$\varphi(2r)=\frac{r}{\tilde{r}}\times\varphi(r). \tag{32}$$

- Algorithm 2 [2]. By adding some additional qubits to the register, Ref. [2] raises the probability of $|2rj \pmod{2^t}| \leq r (0 \leq j \leq 2^t-1)$ to $1-\varepsilon$. Thus,

$$(1-\varepsilon)\times\left(\frac{1}{2}+\delta_1\right)\times\frac{\varphi(r)+\varphi(2r)}{2r}, 0\leq\delta_1\leq\frac{1}{2}. \tag{33}$$

- Algorithm 3 [5]. By choosing the integer with Jacobi-Symbol -1, Ref. [5] raises the lower bound of the probability of y being *good* to $3/4$. Therefore,

$$(1-\varepsilon)\times\left(\frac{3}{4}+\delta_2\right)\times\frac{\varphi(r)+\varphi(2r)}{2r}, 0\leq\delta_2\leq\frac{1}{4}. \tag{34}$$

- Algorithm 3 with the Test algorithm. According to Theorem 1, the success probability for this algorithm is

$$\frac{1-\varepsilon}{2r}\left[(\frac{3}{4}+\delta_2)(\sum_{r'\in R}\varphi(r'))+(\frac{1}{4}-\delta_2)(\sum_{r'\in R_1}\varphi(r'))\right]. \tag{35}$$

When compared with the Algorithm 3 [5], the raising probability is

$$\frac{1-\varepsilon}{2r}(\sum_{r'\in R_1}\varphi(r')) \tag{36}$$

what follows from Eq. (35) minus Eq. (34). Thus, the total raising probability for SFA is

$$\sum_{2r}\Pr(\mathrm{ord}_N(y)=2r)\frac{1-\varepsilon}{2r}\left(\sum_{r'\in R_1}\varphi(r')\right) \tag{37}$$

Consequently, we can see that by loosing the constraints Eqs. (1)–(3) we can obtain an *available* result and make a full use of that result.

5 An Example

In this section, we show how much can our results be used to improve success probability when the RSA moduli [7] are factorized. In doing that we make use the fact that such moduli are products of two safe primes.

Example 4 Let $N = p_1 p_2$ where $p_i - 1 = 2q_i > 10^{100}$, $\log_2(p_1/p_2)$ is small, $|p_1 - p_2|$ is big, p_i and q_i are odd safe primes and $i = 1, 2$. Numbers of this form are likely candidates for moduli of the RSA [7] protocols.

In this case, we will make use of the following remark.

Remark 2 When Algorithm 3 [5] is used in Example 4, Eq. (34) becomes

$$(1-\varepsilon) \times \frac{\varphi(r)}{r}. \tag{38}$$

Proof First,

$$\left(\frac{y}{N}\right) = -1 = \left(\frac{y}{p_1}\right)\left(\frac{y}{p_2}\right) \tag{39}$$

implies that

$$\left(\frac{y}{p_1}\right) = 1 \text{ and } \left(\frac{y}{p_2}\right) = -1 \tag{40}$$

or

$$\left(\frac{y}{p_1}\right) = -1 \text{ and } \left(\frac{y}{p_2}\right) = 1. \tag{41}$$

Without loss of generality, we can assume that Eq. (40) is true. Combining with Eqs. (6), (8) and (40) implies that s_1 is even and s_2 is odd, and

$$r_1 = \frac{q_1}{\gcd(q_1, s_1/2)} \text{ is odd} \tag{42}$$

and

$$r_2 = \frac{2q_2}{\gcd(2q_2, s_2)} \text{ is even.} \tag{43}$$

According to Eq. (7), $\mathrm{ord}_N(y) = 2r = \mathrm{lcm}\{r_1, r_2\}$. Combined with Eqs. (42), (43), we have $r = \mathrm{lcm}\{r_1, r_2/2\}$. Then, $r_1 \mid r$ implies that $y^r \equiv 1 (\mathrm{mod}\ p_1)$. However, $y^r \equiv -1 (\mathrm{mod}\ N)$ implies that $y^r \equiv -1 (\mathrm{mod}\ p_1)$. Thus, $(\frac{y}{N}) = -1$ implies that Eq. (2) always holds and $\delta_2 = \frac{1}{4}$ in Eq. (34).

Then, applying Eq. (8),

$$r_i = \frac{\varphi(p_i)}{\gcd(\varphi(p_i), s_i)} = \frac{2q_i}{\gcd(2q_i, s_i)}, 1 \le s_i \le 2q_i. \tag{44}$$

Thus, r_i must be the factor of $2q_i$, i.e., $1, 2, q_i$ and $2q_i$. According to Eqs. (40), (41), (r_1, r_2) is one of $(1, 2)$, $(2, 1)$, $(1, 2q_2)$, $(2q_1, 1)$, $(2, q_2)$, $(q_1, 2)$, $(q_1, 2q_2)$, and $(2q_1, q_2)$. We have

$$\text{ord}_N(y) = \text{lcm}\{r_1, r_2\} \in \{2, 2q_1, 2q_2, 2q_1q_2\}. \tag{45}$$

Combining with Remark 2 the Eq. (38) can be obtained. By that the proof is finished. □

Now, we can start to determine the improvement success probability in the following remark.

Remark 3 Compared with Algorithm 3 [5], the raising probability of Algorithm 3 with the Test algorithm for RSA [7] is greater than $2/\sqrt{N}$ in each quantum process.

Proof First, note that the raising probability in the Algorithm 3 is

$$P_1 = \sum_t \Pr(r' = t, \frac{t}{2} \mid 2r = t)\Pr(2r = t) \tag{46}$$

where t can take any value in $\{2, 2q_1, 2q_2, 2q_1q_2\}$. Equivalently, it can be written as

$$P_1 = \sum_t \Pr(r'_{y^2} = t \mid r = t)\Pr(r = t) \tag{47}$$

where $t \in \{1, q_1, q_2, q_1q_2\}$.

In Algorithm 3 with the Test algorithm, according to Eq. (44), $\widetilde{r_1}$ and $\widetilde{r_2}$ is one of $(1, 1)$, $(q_1, 1)$, $(1, q_2)$ and (q_1, q_2). Thus, $\text{ord}_N(y^2) = r = \text{lcm}\{\widetilde{r_1}, \widetilde{r_2}\}$ is one of $1, q_1, q_2$ and q_1q_2. According to the discussion in Sect. 3.1, the success probability of Algorithm 3 with the Test algorithm is

$$P_2 = \sum_t \Pr(\widetilde{r_i} \text{ divides } 2r'_{y^2} \mid r = t)\Pr(r = t) \tag{48}$$

where $t \in \{1, q_1, q_2, q_1q_2\}$. Using Eqs. (47), (48), the raising probability is:

$$P_2 - P_1 \geq \sum_{j=1}^{2} \Pr(r'_{y^2} = q_j \mid r = q_1q_2)\Pr(r = q_1q_2). \tag{49}$$

According to Eqs. (44), (45), we have

$$\Pr(r = q_1q_2) = \prod_{j=1}^{2} \Pr(\widetilde{r_j} = q_j) = \frac{(q_1 - 1)(q_2 - 1)}{q_1q_2}. \tag{50}$$

Because $r'_{y^2} = r/\gcd(r, s)$ where $1 \leq s \leq r$, we get

$$\sum_{j=1}^{2} \Pr(r'_{y^2} = q_j \mid r = q_1 q_2) = \frac{q_1 + q_2 - 2}{q_1 q_2}. \tag{51}$$

Computing the product of Eqs. (50), (51), we have

$$\begin{aligned}
P_2 - P_1 &\geq \frac{(q_1 - 1)(q_2 - 1)}{q_1 q_2} \frac{q_1 + q_2 - 2}{q_1 q_2} && (52)\\
&> \frac{1}{q_1} + \frac{1}{q_2} - \frac{1}{q_1^2} - \frac{1}{q_2^2} - \frac{5}{q_1 q_2} && (53)\\
&> \frac{1}{q_1} + \frac{1}{q_2} - \frac{7}{\min^2\{q_1, q_2\}} && (54)\\
&> \frac{1}{\min\{q_1, q_2\}} && (55)\\
&> \frac{2}{\min\{p_1, p_2\}} && (56)\\
&> \frac{2}{\sqrt{N}}. && (57)
\end{aligned}$$

where $\min^2\{q_1, q_2\} > 7\max\{q_1, q_2\}$ is used in that $\log_2(p_1/p_2)$ is small and $p_i - 1 = 2q_i > 10^{100}$. By that the proof is finished. □

Meanwhile, according to Eqs. (50), (51), we can see that the success probability is close to 1 even if SFA does not have to its disposal the ideal value of the order from the order determining algorithm. However, Algorithm 3 with the Test algorithm is behind the possibility to increase success probability to 1. In fact, only for $r'_{y^2} = 1$ and $r = q_1 q_2$, we can not obtain a factor in the improved version of the SFA. But in such a case the raising probability is

$$\Pr(r'_{y^2} = 1 \mid r = q_1 q_2) \Pr(r = q_1 q_2). \tag{58}$$

It is not difficult to verify that Eq. (58) is $O(1/N)$.

Furthermore, using a similar argument with Eq. (38), we can show that the fault probability in Ref. [5] is $O(1/\sqrt{N})$. Thereby, we have reduced the fault probability to $O(1/N)$ in Example 3. This way the example is finished. □

As a consequence, Algorithm 3 with the Test algorithm can use the *available* result to obtain a non-trivial factor of large integers and can improve the success probability in related quantum processes.

Example 5 Let $N = 77 = 7 \times 11$. $\varphi(77) = \varphi(7) \times \varphi(11) = 6 \times 10 = 60$. In these numbers, the number of numbers satisfying Eq. (39) is 30, and the number of numbers satisfying Eq. (40) is 15. According to Eqs. (42), (43), $r_1 \in \{1, 3\}$ and $r_2 \in \{2, 10\}$. Thus, $2r = \operatorname{lcm}\{r_1, r_2\} \in \{2, 6, 10, 30\}$. Numbers in $\mathbb{Z}_7^*$ with $r_1 = 1$ is 1, and with $r_1 = 3$ is in the set $\{2, 4\}$. Meanwhile, numbers in $\mathbb{Z}_{11}^*$ with $r_2 = 2$ is 10, and with

$r_2 = 10$ is in the set $\{2, 6, 7, 8\}$. Here, we list these numbers of the order $2r \in \{2, 6, 10, 30\}$, respectively.

- (1) The numbers with $2r = 2$: 43. According to Theorem 1, all factors of the order is usable.
- (2) The numbers with $2r = 6$: 32, 65. According to Theorem 1, all factors of the order is usable.
- (3) The numbers with $2r = 10$: 8, 29, 50, 57. According to Theorem 1, all factors of the order is usable.
- (4) The numbers with $2r = 30$: 2, 18, 30, 39, 46, 51, 72, 74. According to Theorem 1, $r' \in \{3, 5, 6, 10, 15, 30\}$ is usable, and only $r' \in \{1, 2\}$ is unusable. Note that we can get $r' \in \{1, 2\}$ if the numerator of the fraction obtained by the continued fractions (CF) algorithm [2] is 15 and 30. Therefore, this probability is $1/15$. The raising probability is the case that $r' \in \{3, 5, 6, 10\}$ and the probability is $2/5$. And, we choose the numbers with $2r = 30$ with a probability $8/15$. As a result, the fault probability is reduced to $8/225$.

This way the example is finished. □

6 Conclusion

This paper showed that we can get a non-trivial factor even from some "unuseful" results of the order finding subroutines in previous Shor's factoring algorithms [1, 2, 5]. This discovery shows that the previous constraints are only a sufficient but not necessary condition for SFA being successful in factorization.

A natural research challenge is therefore now to extend results obtained in this paper and to show some other unideal values of the results of the order finding algorithm that can be actually used to factorize with sufficiently large probability.

Acknowledgements This work is supported in part by the National Natural Science Foundation of China (Nos. 61572532, 61272058), the National Natural Science Foundation of Guangdong Province of China (No. 2017B030311011), and the Fundamental Research Funds for the Central Universities of China (No. 17lgjc24). Qiu is also Funded by FCT project UID/EEA/50008/2013.

References

1. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997)
2. Nielson, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000)
3. Bocharov, A., Roetteler, M., Svore, K.M.: Factoring with qutrits: shor's algorithm on ternary and metaplectic quantum architectures. arXiv:1605.02756v3 [quant-ph] (2016)
4. Lawson, T.: Odd orders in Shor's factoring algorithm. Quantum Inf. Process. **14**(3), 831–838 (2015)

5. Leander, G.: Improving the success probability for shor's factoring algorithm. arXiv: 0208183 [quant-ph] (2002)
6. Nagaich, S., Goswami, Y.C.: Shor's algorithm for quantum numbers using matlab simulator. In: International Conference on Advanced Computing Communication Technologies, Panipat, 165–168 (2015)
7. Cramer, R., Shoup, V.: Signature schemes based on the strong RSA assumption. ACM Trans. Inf. Syst. Secur. **3**(3), 46–51 (1999)
8. Davies, J.T., Rickerd, C.J., Grimes, M.A., Guney, D.O.: An n-bit general implementation of shor's quantum period-finding algorithm. arXiv:1612.07424v1 [quant-ph] (2016)