Robustness and performance scaling of a quantum computer with respect to a class of static defects

Y. S. Nam and R. Blümel
Department of Physics, Wesleyan University, Middletown, Connecticut 06459-0155, USA

(Received 10 October 2013; published 10 December 2013)

Competing computationally with experimental groups for the construction of scaling quantum computers, we simulate a complete quantum-gate by quantum-gate implementation of Shor's algorithm on a classical 128-core cluster computer. The resulting virtual quantum computer serves as a convenient quantum laboratory for the investigation of the effect of defects in the quantum circuitry. The class of defects studied here is the removal of all rotation gates with rotation angles $\theta < \pi/2^b$. Factoring semiprimes $N = 21,33,35,39,55,57$, we find that the quantum computer still operates with acceptable performance (success probability of factoring) down to $b = 2$. This is surprising since the deletion of rotation gates results in large errors in the arithmetic circuitry of the quantum computer. Extrapolating on the basis of these results we conclude that for quantum computers of practical interest more than 99% of rotation gates may be discarded with acceptable consequences in quantum computer performance. This result may be of interest to experimental physicists and quantum engineers currently embarked on designing efficient circuitry for scaling quantum computers.

DOI: 10.1103/PhysRevA.88.062310 PACS number(s): 03.67.Lx

I. INTRODUCTION

Although the power of modern supercomputers seems limitless, there are many unsolved problems of theoretical and practical interest that are squarely beyond the capabilities of any classical computer. One of these problems is the factorization of a large semiprime $N = pq$, where p and q, the prime factors of N, are about the same size. Several powerful methods have been developed, such as the quadratic number sieve [1] or the general number field sieve [2], to attack the factorization problem. However, even if we constructed a classical supercomputer the size of the universe, factoring a 5000 digit semiprime would still be impossible [3]. In fact, the security of many public-key cryptosystems [4], among them the Rivet-Shamir-Adleman (RSA) cryptosystem [5], depend precisely on this observation.

When it comes to quantum computing, however, the story is quite different. A quantum computer running Shor's algorithm [6] is capable of factoring current RSA semiprimes in a reasonable amount of time and therefore poses a threat to RSA-encrypted documents. Quantum algorithms, such as Shor's algorithm, make full use of quantum features, such as superposition and entanglement, that allow for types of information processing that are not available to classical computers.

The enormous potential of quantum computers notwithstanding, only a few "useful" quantum algorithms are known to date. One of them, and arguably the most important one, is Shor's algorithm, which consequently has been under intense investigation both theoretically [3,7–23] and experimentally [24–28]. In 2001, e.g., Vandersypen *et al.* [24] successfully implemented Shor's algorithm, factoring $N = 15$ on a liquid nuclear magnetic resonance (NMR) quantum computer. Subsequently, Lu *et al.* [25] and Lanyon *et al.* [26], again factoring $N = 15$, successfully demonstrated a quantum computer using a photonic system. This was followed by another successful demonstration by Politi *et al.* [27]. Recently, in 2012, Martín-López *et al.* [28] used a qubit recycling technique to factor $N = 21$, setting a new record for the largest semiprime factored on actual quantum computer hardware.

Despite the pioneering accomplishments of the authors of Refs. [24–28], all experiments to date use various compiled versions of Shor's algorithm, i.e., tailor-made algorithms that take advantage of the preknowledge of the two prime factors of the semiprimes used in the experiments ($N = 15 = 3 \times 5$ and $N = 21 = 3 \times 7$). These algorithms will not work for any other N than the one they are designed for, making the resulting quantum computers special purpose quantum computers. Defining a *scaling* quantum computer as one that accepts different values of N, the current implementations of Shor's algorithm are said to be nonscaling. However, considering that even for the small N currently used in the experiments ($N = 15$ and $N = 21$), thousands of quantum-gate operations are needed to run a complete, scaling, experimental implementation of Shor's algorithm, the use of compiled, highly optimized versions of Shor's algorithm is currently unavoidable and complete, scaling experimental implementations of Shor's algorithm remain elusive. Therefore, given the experimental situation, and for the time being, the behavior of the complete (scaling) version of Shor's algorithm can only be studied via numerical simulations on classical computers [3,9–11,29–34].

The current experimental limitations provide the motivation for our numerical work. Running a complete implementation of Shor's algorithm on a classical cluster computer, we report here the results on factoring semiprimes up to $N = 57$. In addition, with the help of a shortcut, i.e., executing the modular exponentiation part of Shor's algorithm classically, we are able to factor semiprimes up to $N = 1,034,273$.

Being able to factor substantially larger N than are possible experimentally is not the only advantage of our numerical implementation. On a computer we have access to each individual quantum gate, which lets us enable or disable any individual quantum gate at will, something which is extremely difficult to accomplish experimentally. Making use of our free access to quantum gates, our main result is that an enormous number of quantum gates may be pruned from Shor's algorithm without significantly compromising the performance of the resulting streamlined quantum computer.

We readily admit that our lead with respect to experiment will only be temporary. The (classical) computer resources needed for our simulations grow exponentially in the number of qubits, which is unsustainable beyond even modest values of N. Therefore, sooner or later, the experiment will catch up to the simulations and then easily surpass them as far as N is concerned. But for now, even if only temporarily, numerical simulation has the edge.

Our paper is organized in the following way. A detailed description of the gate-by-gate decomposition of Shor's algorithm is presented in Sec. II. The banded Shor algorithm is presented in Sec. III. Our virtual quantum computer may be operated in two modes, A and B. In mode A, we perform the modular exponentiation part of Shor's algorithm classically, which results in a significant speedup and memory savings that allow us to study factorization of semiprimes up to $N \sim 10^6$. In Sec. IV we present the theory and simulation results of our mode-A calculations. A new 40-qubit result that took about three months to compute is also presented. The theory and simulation results of mode B, a complete quantum-gate by quantum-gate implementation of Shor's algorithm, is presented in Sec. V. Both our mode-A and mode-B calculations show that a substantial number of quantum gates may be saved in future experimental implementations of Shor's algorithm following a pruning strategy, which we call banding (see Sec. III). In Sec. VI we discuss our results and in Sec. VII we summarize and conclude our paper.

II. SHOR'S ALGORITHM

Shor's algorithm may be divided into two parts: (1) modular exponentiation (ME) and (2) period finding (PF). Given a semiprime $N = pq$ to be factored, the ME part of Shor's algorithm, supplied with an integer seed x between 1 and N that is coprime to N, computes $x^r \bmod N$ for an integer exponent r. Defining the mapping

$$f(r) = x^r \bmod N, \tag{1}$$

the PF part of Shor's algorithm, then, determines the period ω of f, i.e., the smallest integer $\omega > 0$ for which $f(\omega) = 1$, via a quantum Fourier transform (QFT). For a successful factoring, ω needs to meet the following two conditions: (i) ω needs to be even and (ii) $(x^{\omega/2} - 1) \bmod N \neq 0$. If any one of these two conditions are not met, we try a different seed x. When both conditions are satisfied, the two factors p and q of N are then obtained via

$$p = \gcd(x^{\omega/2} - 1, N), \quad q = \gcd(x^{\omega/2} + 1, N), \tag{2}$$

where gcd denotes the greatest common divisor.

To date, there have been multiple proposed methods of implementing Shor's algorithm on a quantum computer; a list of several different architectures is available in Table IV of Ref. [35]. Notable ones include, but are not limited to, those proposed by Vedral *et al.* [12], Beckman *et al.* [13], Zalka *et al.* [14], Beauregard [15], Van Meter *et al.* [16,17], Takahashi *et al.* [18], and Kutin [19]. In this paper, from the numerous methods available, we focus on Beauregard's representation of Shor's algorithm, which is based on QFT operations. The reason behind this choice is due to the possibility of approximating part of the circuit using a particular method of circuit pruning (banding), which will be introduced in detail in Sec. III.

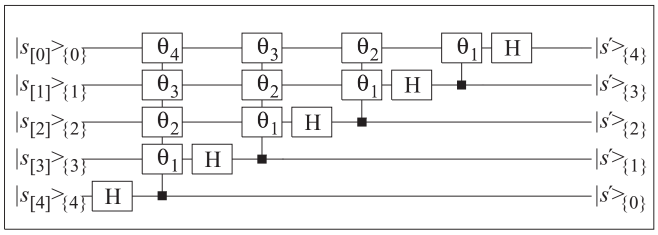


FIG. 1. Logic circuit of a five-qubit QFT. H denotes the Hadamard gate and θ_j denotes the coherently controlled two-qubit conditional rotation gates with rotation angles $\theta_j = \pi/2^j$.

A. Period finding

The first step of the PF part of Shor's algorithm is a QFT defined as

$$|s'\rangle = \hat{U}^{(\mathrm{QFT})}|s\rangle = \frac{1}{\sqrt{2^n}} \sum_{l=0}^{2^n-1} e^{\frac{2\pi i s l}{2^n}} |l\rangle, \tag{3}$$

where n is the number of qubits, $|s\rangle$ and $|s'\rangle$ indicate input and output states, respectively, and $|l\rangle$ are the basis states of the 2^n-dimensional Hilbert space $\mathcal{H}^{2^n}$ spanned by the n qubits. Since our quantum computer is based on qubits, it is natural to decompose $\mathcal{H}^{2^n}$ into the tensor product of n two-dimensional Hilbert spaces according to

$$\mathcal{H}^{2^n} = \mathcal{H}^2_{\{n-1\}} \otimes \mathcal{H}^2_{\{1\}} \otimes \mathcal{H}^2_{\{0\}}, \tag{4}$$

where $\mathcal{H}^2_{\{m\}}$ denotes the mth Hilbert space. In this notation, for instance, with $l_{[m]}$ the mth binary digit of an integer l,

$$|l\rangle = |l_{[n-1]}\rangle_{\{n-1\}} \cdots |l_{[1]}\rangle_{\{1\}} |l_{[0]}\rangle_{\{0\}}. \tag{5}$$

This decomposition with its associated $\{\ldots\}$ notation has additional advantages since in our quantum circuits qubits and states do not always correspond. For example, the state $|a\rangle$ of the jth qubit may actually correspond to the mth Hilbert space $\mathcal{H}^2_{\{m\}}$, in which case we write $|a\rangle_{\{m\}}$. This notation is of particular convenience in connection with QFT circuits where the output has to be read in reverse order (see Ref. [36] and Fig. 1).

In the decomposed Hilbert space notation, with $|0\rangle_{\{m\}}$ and $|1\rangle_{\{m\}}$ as basis states of the mth Hilbert space, (3) may be written as

$$|s'\rangle = \frac{1}{\sqrt{2^n}} \prod_{m=0}^{n-1} \sum_{j=0}^{1} e^{2\pi i (.s_{[m]} s_{[m-1]} \cdots s_{[0]}) j} |j\rangle_{\{n-m-1\}}, \tag{6}$$

where $s_{[\nu]}$ denotes the νth binary digit of s, and

$$.s_{[m]} s_{[m-1]} \cdots s_{[0]} = \sum_{\nu=0}^{m} s_{[\nu]} 2^{-(m-\nu+1)}. \tag{7}$$

Expanding the product in (6), we obtain explicitly [36]

$$|s'\rangle = \frac{1}{\sqrt{2^n}} (|0\rangle + e^{2\pi i (.s_{[0]})}|1\rangle)_{\{n-1\}} (|0\rangle + e^{2\pi i (.s_{[1]} s_{[0]})}|1\rangle)_{\{n-2\}} \cdots (|0\rangle + e^{2\pi i (.s_{[n-1]} s_{[n-2]} \cdots s_{[0]})}|1\rangle)_{\{0\}}, \tag{8}$$

which demonstrates that the QFT may be realized with a series of Hadamard gates and phase rotation gates. A fully quantum mechanical realization of the QFT in (8) is shown in Fig. 1.

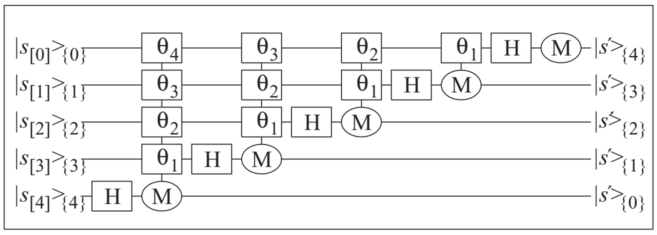

$|s_{[0]}\rangle_{\{0\}}$
$|s_{[1]}\rangle_{\{1\}}$
$|s_{[2]}\rangle_{\{2\}}$
$|s_{[3]}\rangle_{\{3\}}$
$|s_{[4]}\rangle_{\{4\}}$
θ4
θ3
θ2
θ1
H
M
$|s'\rangle_{\{4\}}$
$|s'\rangle_{\{3\}}$
$|s'\rangle_{\{2\}}$
$|s'\rangle_{\{1\}}$
$|s'\rangle_{\{0\}}$

FIG. 2. Logic circuit of a five-qubit semiclassical QFT. The meaning of H and θ_j is the same as in Fig. 1. M denotes the measurement gate.

We emphasize that in the case of the PF part of Shor's algorithm since the QFT is directly followed by measurements, we are allowed to interchange the orders of the controlled phase rotation gates with the measurements. This way, by replacing the coherently controlled two-qubit phase rotation gates in Fig. 1 with single-qubit phase rotation gates that are classically controlled by the results of measurements, we obtain the semiclassical QFT circuit shown in Fig. 2 [20]. We note that due to the presence of measurements, the semiclassical QFT cannot be used to construct the quantum ME, which, as shown in Sec. II B, requires the coherent QFT.

B. Quantum modular exponentiation

In this section we show that ME may be broken down into modular multiplications (MM), which then may be expressed as a sequence of modular additions (MA). This shows that MA is the basic building block of ME. Then, we separate the modulo part from the addition part in MA and show that the addition part may be realized with phase rotations when performed in Fourier space.

From elementary algebra we know that any integer r may be written in binary form as

$$r = 2^0 r_{[0]} + 2^1 r_{[1]} + \cdots + 2^{n-1} r_{[n-1]}, \tag{9}$$

where n is an integer that satisfies $r < 2^n$ and $r_{[k]}$ denotes the kth binary digit of r. Inserting (9) in (1), we obtain

$$f(r) = x^{2^0 r_{[0]} + 2^1 r_{[1]} + \cdots + 2^{n-1} r_{[n-1]}} \bmod N, \tag{10}$$

which may be written as consecutive MMs according to

$$f(r) = g_{n-1}(\ldots(g_1(g_0(1)))\ldots), \tag{11}$$

where

$$g_k(b) = \left(b \times x^{2^k r_{[k]}}\right) \bmod N. \tag{12}$$

Using

$$x^{2^k r_{[k]}} = 2^0\left(x^{2^k r_{[k]}}\right)_{[0]} + 2^1\left(x^{2^k r_{[k]}}\right)_{[1]} + \cdots + 2^{n'-1}\left(x^{2^k r_{[k]}}\right)_{[n'-1]} \tag{13}$$

in (12), where n' is yet another arbitrary integer that satisfies $x^{2^k r_{[k]}} < 2^{n'}$, we obtain

$$g_k(b) = h_{k,n'-1}(b,\ldots,h_{k,1}(b,h_{k,0}(b,0))\ldots), \tag{14}$$

where

$$h_{k,j}(b,a) = \left[a + b \times 2^j \left(x^{2^k r_{[k]}}\right)_{[j]}\right] \bmod N, \tag{15}$$

which is equivalent to MA. This completes the decomposition of ME into MAs.

The coherent decomposition of MA into addition and modulo parts is straightforward. First, defining an $n'+1$ (qu)bit adder $\hat{u}$, we write the functional expression of addition of two integers $a,b < 2^{n'}$ in the form

$$\hat{u}_a|b\rangle = |b+a\rangle. \tag{16}$$

The inverse operation is defined as

$$\hat{u}_a^{-1}|b\rangle = \begin{cases} |b-a\rangle, & \text{if} \quad b \geqslant a, \\ |2^{n'+1} - (a-b)\rangle, & \text{if} \quad b < a, \end{cases} \tag{17}$$

which is equivalent to subtraction. Noticing that $(a+b) \bmod N$ (MA of a and b with respect to N) is (i) $a+b-N$ if $a+b \geqslant N$ or (ii) $a+b$ if $a+b < N$ for integers $a,b < N < 2^{n'}$, adding a to b then subtracting N results in

$$\hat{u}_N^{-1}\hat{u}_a|b\rangle = \begin{cases} |a+b-N\rangle, & \text{if} \quad a+b \geqslant N, \\ |2^{n'+1} - [N-(a+b)]\rangle, & \text{if} \quad a+b < N, \end{cases} \tag{18}$$

which correctly yields case (i). Since $a+b-N < 2^{n'}$ and $2^{n'+1} - [N-(a+b)] > 2^{n'}$, case (ii) may be captured correctly with the help of (a) an overflow and (b) an auxiliary (qu)bit in the following way. Defining $\hat{v}(|t\rangle;|c\rangle)$ as a controlled-NOT operation, linear in its two arguments, according to

$$\hat{v}(|t\rangle;|c\rangle) = \begin{cases} |t\rangle, & \text{if} \quad c = 0, \\ |(t+1) \bmod 2\rangle, & \text{if} \quad c = 1, \end{cases} \tag{19}$$

for $|t\rangle$ a target state and $|c\rangle$ a control state, both of which may be $|0\rangle$ or $|1\rangle$, an initialized auxiliary state of $|0\rangle$ as a target, controlled by an overflow state, differentiates the two different cases in (18), namely,

$$\hat{v}\left(|0\rangle;\left[\hat{u}_N^{-1}\hat{u}_a|b\rangle\right]_{\{n'\}}\right) = \begin{cases} |0\rangle, & \text{if} \quad a+b \geqslant N, \\ |1\rangle, & \text{if} \quad a+b < N, \end{cases} \tag{20}$$

where $[\hat{u}_N^{-1}\hat{u}_a|b\rangle]_{\{n'\}}$ is the state of the n'th qubit, or the overflow state, of $\hat{u}_N^{-1}\hat{u}_a|b\rangle$. Since for case (ii) we would like $a+b$ as our MA result that needs an addition of N to the current state in (18), whereas for case (i) we should not alter its current state, as we already have the correct MA output, we make use of a controlled addition defined as, for $|c\rangle$ a binary state

$$\hat{u}_a(|b\rangle;|c\rangle) = \begin{cases} |b\rangle, & \text{if} \quad c = 0, \\ |a+b\rangle, & \text{if} \quad c = 1, \end{cases} \tag{21}$$

with the target state of (18) and the controlling state of (20) for the controlled addition of N to obtain

$$\hat{u}_N\left(\hat{u}_N^{-1}\hat{u}_a|b\rangle;\hat{v}\left(|0\rangle;\left[\hat{u}_N^{-1}\hat{u}_a|b\rangle\right]_{\{n'\}}\right)\right) = \begin{cases} |a+b-N\rangle, & \text{if} \quad a+b \geqslant N, \\ |a+b\rangle, & \text{if} \quad a+b < N, \end{cases} \tag{22}$$

which is the desired result of MA.

In practice, the auxiliary state needs to be restored to its initial state $|0\rangle$ for recycling. This may also be achieved coherently with the help of an overflow and an auxiliary state. Since the auxiliary state after the MA operation in (22) is (20), we simply need to design a sequence of functions that differentiate the two cases (i) and (ii), which can be used for the restoration process. One way is to subtract a from (22), namely,

$$\hat{u}_a^{-1}\hat{u}_N\big(\hat{u}_N^{-1}\hat{u}_a|b\rangle;\hat{v}\big(|0\rangle;\big[\hat{u}_N^{-1}\hat{u}_a|b\rangle\big]_{\{n'\}}\big)\big) = \begin{cases}|2^{n'+1}-(N-b)\rangle, & \text{if} \quad a+b \geqslant N,\\ |b\rangle, & \text{if} \quad a+b<N,\end{cases} \tag{23}$$

which yields $2^{n'+1}-(N-b) > 2^{n'}$ and $b < 2^{n'}$. Defining yet another operation $\hat{w}$ on a binary state $|c\rangle$ as

$$\hat{w}|c\rangle = |(c+1) \bmod 2\rangle, \tag{24}$$

we notice that

$$\hat{v}\big(\hat{v}\big(|0\rangle;\big[\hat{u}_N^{-1}\hat{u}_a|b\rangle\big]_{\{n'\}}\big);\hat{w}\big(\big[\hat{u}_a^{-1}\hat{u}_N\big(\hat{u}_N^{-1}\hat{u}_a|b\rangle;\hat{v}\big(|0\rangle;\big[\hat{u}_N^{-1}\hat{u}_a|b\rangle\big]_{\{n'\}}\big)\big)\big]_{\{n'\}}\big)\big) = 0, \tag{25}$$

which restores the auxiliary state to $|0\rangle$ for both cases (i) and (ii). Adding a to (23) after the restoration in (25), we once more obtain

$$\hat{u}_a\hat{u}_a^{-1}\hat{u}_N\big(\hat{u}_N^{-1}\hat{u}_a|b\rangle;\hat{v}\big(|0\rangle;\big[\hat{u}_N^{-1}\hat{u}_a|b\rangle\big]_{\{n'\}}\big)\big) = \begin{cases}|a+b-N\rangle, & \text{if} \quad a+b \geqslant N,\\ |a+b\rangle, & \text{if} \quad a+b<N,\end{cases} \tag{26}$$

which completes a coherent and reusable MA architecture that has separate addition and modulo parts.

So far we have coherently broken ME down into adders. Hence, the last step required to complete the decomposition process, i.e., breaking ME into elementary coherent gates, is now to construct a coherent adder circuit. Executing the addition transform (16) in Fourier space, we would like to obtain

$$\hat{u}_a|b\rangle = \hat{U}^{(\mathrm{QFT})^{-1}}\hat{u}_a^{(\mathcal{F})}\hat{U}^{(\mathrm{QFT})}|b\rangle, \tag{27}$$

where $\hat{U}^{(\mathrm{QFT})^{-1}}$ denotes the unitary inverse of $\hat{U}^{(\mathrm{QFT})}$ and $\hat{u}_a^{(\mathcal{F})}$ denotes the addition operation in Fourier space. Since we have $\hat{U}^{(\mathrm{QFT})^{-1}}\hat{u}_a^{(\mathcal{F})}\hat{U}^{(\mathrm{QFT})}|b\rangle = |a+b\rangle$, multiplying $\hat{U}^{(\mathrm{QFT})}$ on both sides results in

$$\hat{u}_a^{(\mathcal{F})}\hat{U}^{(\mathrm{QFT})}|b\rangle = \hat{U}^{(\mathrm{QFT})}|a+b\rangle. \tag{28}$$

Using (6) in (28) and comparing phase factors, we have

$$\hat{u}_a^{(\mathcal{F})}\big(e^{2\pi i(.b_{[m]}b_{[m-1]}\cdots b_{[0]})j}|j\rangle_{\{n-m-1\}}\big) = e^{2\pi i[.(a+b)_{[m]}(a+b)_{[m-1]}\cdots(a+b)_{[0]}]j}|j\rangle_{\{n-m-1\}}. \tag{29}$$

Since any integer multiple of $2\pi i$ in the exponents of (29) leaves the expression unchanged, replacing $(.b_{[m]}b_{[m-1]}\cdots b_{[0]})$ with $(b_{[n-1]}b_{[n-2]}\cdots b_{[m+1]}.b_{[m]}b_{[m-1]}\cdots b_{[1]}b_{[0]})$, we obtain

$$e^{2\pi i(.b_{[m]}b_{[m-1]}\cdots b_{[0]})j} = e^{2\pi i\frac{b}{2^{m+1}}j}, \tag{30}$$

where we used

$$b_{[n-1]}b_{[n-2]}\cdots b_{[m+1]}.b_{[m]}b_{[m-1]}\cdots b_{[1]}b_{[0]} = \frac{b}{2^{m+1}}. \tag{31}$$

Similarly, for the right-hand side of (29), we obtain

$$e^{2\pi i[.(a+b)_{[m]}(a+b)_{[m-1]}\cdots(a+b)_{[0]}]j} = e^{2\pi i\frac{a+b}{2^{m+1}}j}. \tag{32}$$

Inserting (30) and (32) into (29) and solving for $\hat{u}_a^{(\mathcal{F})}$, we have

$$\hat{u}_a^{(\mathcal{F})}|j\rangle_{\{n-m-1\}} = |j^{(a)}\rangle_{\{n-m-1\}} \tag{33}$$

with

$$|j^{(a)}\rangle_{\{n-m-1\}} = e^{2\pi i\frac{a}{2^{m+1}}j}|j\rangle_{\{n-m-1\}}. \tag{34}$$

The circuit diagram for $\hat{u}_a^{(\mathcal{F})}$ in (33), i.e., the quantum Fourier adder (QFA), is shown in Fig. 3.

III. BANDWIDTH

To break currently employed RSA codes, we would need to factor semiprimes N whose bit lengths are of the order of several thousands. Constructing a quantum computer with thousands of qubits and running Shor's algorithm with the exact QFT circuits as shown in Figs. 1 and 2, however, is strictly impossible since this would require a realization of phase rotation gates with angles $\lesssim\frac{2\pi}{2^{1000}}$. Given the finite precision and accuracy laboratory tools can provide, realizing even a relatively modest $\frac{2\pi}{2^{100}}$ phase rotation gate is still unrealistic. Therefore, in this section, we present one way of relaxing such a stringent requirement, namely, banding [3,9–11,21].

Defining an integer b, which we call the bandwidth, we band a quantum circuit by removing all the phase rotation gates with an angle smaller than $\frac{\pi}{2^b}$. In the case of Shor's algorithm, constructed with the method shown in Sec. II, we can band the following two main circuits: (i) QFT and (ii) QFA. It is, in fact, the possibility of banding these two

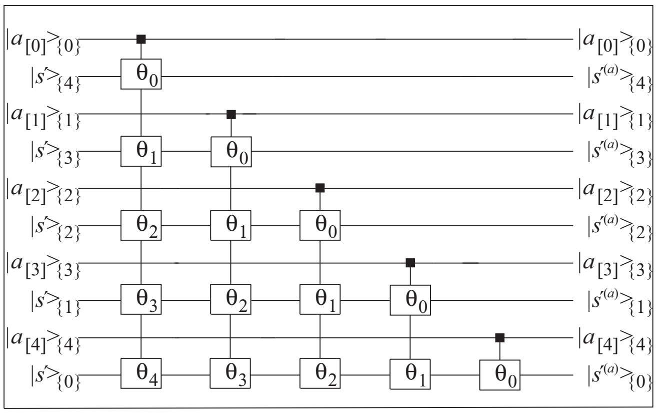

$|a_{[0]}\rangle_{\{0\}}$
$|s\rangle_{\{4\}}$
$|a_{[1]}\rangle_{\{1\}}$
$|s\rangle_{\{3\}}$
$|a_{[2]}\rangle_{\{2\}}$
$|s\rangle_{\{2\}}$
$|a_{[3]}\rangle_{\{3\}}$
$|s\rangle_{\{1\}}$
$|a_{[4]}\rangle_{\{4\}}$
$|s\rangle_{\{0\}}$
θ_0 θ_1 θ_2 θ_3 θ_4
$|s^{(a)}\rangle_{\{4\}}$
$|s^{(a)}\rangle_{\{3\}}$
$|s^{(a)}\rangle_{\{2\}}$
$|s^{(a)}\rangle_{\{1\}}$
$|s^{(a)}\rangle_{\{0\}}$

FIG. 3. Logic circuit of a five-qubit QFA. The number to be added is a, which, if known, may be classically implemented, i.e., by replacing the coherently controlled two-qubit rotation gates with classically controlled single-qubit rotation gates. θ_j denotes the rotation angle $\pi/2^j$.

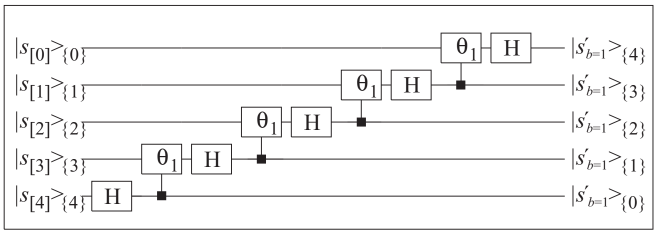

$|s_{[0]}\rangle_{\{0\}}$
$|s_{[1]}\rangle_{\{1\}}$
$|s_{[2]}\rangle_{\{2\}}$
$|s_{[3]}\rangle_{\{3\}}$
$|s_{[4]}\rangle_{\{4\}}$
θ_1
H
$|s'_{b=1}\rangle_{\{4\}}$
$|s'_{b=1}\rangle_{\{3\}}$
$|s'_{b=1}\rangle_{\{2\}}$
$|s'_{b=1}\rangle_{\{1\}}$
$|s'_{b=1}\rangle_{\{0\}}$

FIG. 4. Logic circuit of a five-qubit BQFT for bandwidth $b = 1$. In general, a BQFT of bandwidth b retains all rotation gates θ_j with $j \leqslant b$ and discards all rotation gates θ_j with $j > b$.

circuits that are centrally used in both the ME and PF parts of Shor's algorithm that provides us with the advantage of using the particular architecture in Ref. [15]. In this section, therefore, we will introduce the banded QFT (BQFT) and the banded QFA (BQFA).

We start by removing phase rotations that are smaller than $\frac{\pi}{2^b}$ in (6), i.e.,

$$|s_b'\rangle = [\hat{U}^{(\mathrm{QFT})}]_b|s\rangle$$
$$= \frac{1}{\sqrt{2^n}} \prod_{m=0}^{n-1} \sum_{j=0}^{1} e^{2\pi i(.s_{[m]}s_{[m-1]}\cdots s_{[m-b]})j}|j\rangle_{\{n-m-1\}}, \quad (35)$$

where $[\hat{U}^{(\mathrm{QFT})}]_b$ denotes the unitary operator for the BQFT with bandwidth b. Writing out the products in (35) in the basis of binary states of Hilbert space, we obtain

$$|s_b'\rangle = (|0\rangle + e^{2\pi i(.s_{[0]})}|1\rangle)_{\{n-1\}}(|0\rangle + e^{2\pi i(.s_{[1]}s_{[0]})}|1\rangle)_{\{n-2\}}$$
$$\ldots(|0\rangle + e^{2\pi i(.s_{[n-1]}s_{[n-2]}\cdots s_{[n-b-1]})}|1\rangle)_{\{0\}}, \quad (36)$$

which shows that the BQFT may be realized with Hadamard gates and phase rotation gates. A fully quantum mechanical circuit diagram of the BQFT is shown in Fig. 4.

From Sec. II A we recall that the QFT circuit may be constructed semiclassically if it is used for the PF part of Shor's algorithm. In fact, the semiclassical QFT may also be banded to result in the banded semiclassical QFT as shown in Fig. 5. The only difference in the banding mechanism here from that of the fully quantum version in Fig. 4 is that the deleted gates are classically controlled single-qubit phase rotation gates instead of coherently controlled two-qubit phase rotation gates.

The BQFT as an approximation of the exact QFT has already been described and investigated in the literature [3,9,10,21]. Although the idea may be traced back to Ref. [22], banding other quantum circuits such as QFA has not yet been studied in detail. Therefore, studying the BQFA is a main focus of this paper. In analogy to the BQFT, we turn the QFA

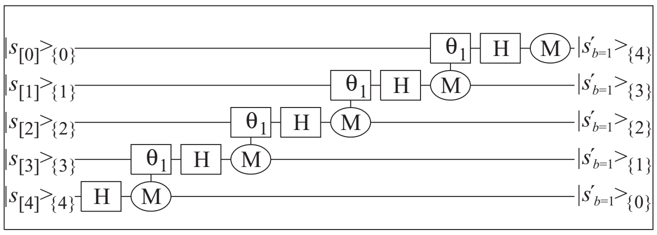

$|s_{[0]}\rangle_{\{0\}}$
$|s_{[1]}\rangle_{\{1\}}$
$|s_{[2]}\rangle_{\{2\}}$
$|s_{[3]}\rangle_{\{3\}}$
$|s_{[4]}\rangle_{\{4\}}$
θ_1
H
M
$|s'_{b=1}\rangle_{\{4\}}$
$|s'_{b=1}\rangle_{\{3\}}$
$|s'_{b=1}\rangle_{\{2\}}$
$|s'_{b=1}\rangle_{\{1\}}$
$|s'_{b=1}\rangle_{\{0\}}$

FIG. 5. Logic circuit of a five-qubit banded semiclassical QFT for bandwidth $b = 1$.

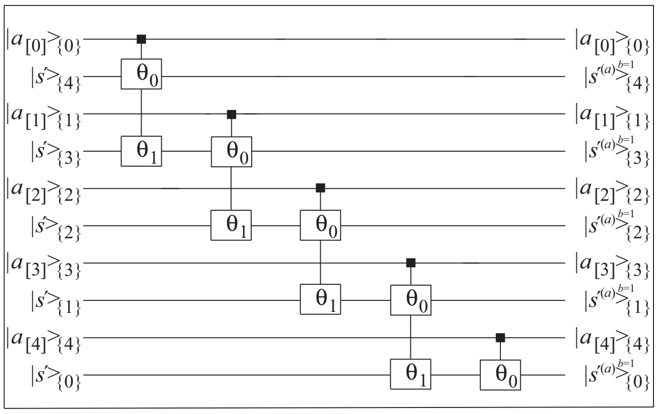

$|a_{[0]}\rangle_{\{0\}}$
$|s\rangle_{\{4\}}$
$|a_{[1]}\rangle_{\{1\}}$
$|s\rangle_{\{3\}}$
$|a_{[2]}\rangle_{\{2\}}$
$|s\rangle_{\{2\}}$
$|a_{[3]}\rangle_{\{3\}}$
$|s\rangle_{\{1\}}$
$|a_{[4]}\rangle_{\{4\}}$
$|s\rangle_{\{0\}}$
θ_0
θ_1
$|s'^{(a)}_{b=1}\rangle_{\{4\}}$
$|s'^{(a)}_{b=1}\rangle_{\{3\}}$
$|s'^{(a)}_{b=1}\rangle_{\{2\}}$
$|s'^{(a)}_{b=1}\rangle_{\{1\}}$
$|s'^{(a)}_{b=1}\rangle_{\{0\}}$

FIG. 6. Logic circuit of a five-qubit BQFA. Shown is the case with bandwidth $b = 1$, i.e., only rotation gates θ_j with $j \leqslant 1$ are kept.

operator in (33) into the BQFA operator according to

$$\left[\hat{u}_a^{(\mathcal{F})}\right]_b|j\rangle_{\{n-m-1\}} = e^{2\pi i(.a_{[m]}a_{[m-1]}\cdots a_{[m-b]})j}|j\rangle_{\{n-m-1\}}, \quad (37)$$

where a is the number to be added and b is the bandwidth we imposed. As an illustration of (37) we show in Fig. 6 the circuit diagram of a five-qubit BQFA with $b = 1$.

IV. MODE A: HYBRID IMPLEMENTATION OF SHOR'S ALGORITHM

So far in this paper we have demonstrated how Shor's algorithm may be constructed with a series of coherent quantum circuits (see Sec. II) of which two, i.e., QFT and QFA, can be banded as an approximation method (see Sec. III). In this section we present one way of simulating Shor's algorithm on a classical computer, a hybrid mode, called mode A, in which we simulate the PF part of Shor's algorithm, using the semiclassical BQFT supplied with the classical result of the ME part of Shor's algorithm. The goal here is to compute the scaling laws of success probabilities of factoring as a function of n that allow us to extrapolate the number of gates needed for large $n \sim 1000$.

Our virtual quantum computer running Shor's algorithm has two registers: a control register (register I), which is used for the PF part of Shor's algorithm and a computational register (register II) on which the ME part is executed. In mode A, since the ME part is performed classically, the periodicity ω is implemented in register I directly and we need not simulate register II. Therefore, with n the number of qubits in register I, we define the success probability of factoring, or the absolute performance of the quantum computer, as the sum of probabilities of obtaining any one of the integers closest to integer multiples of $2^n/\omega$ in register I (the expected peak locations in Fourier space given input with periodicity ω), i.e.,

$$\tilde{P}(n,\omega) = \sum_{j=0}^{\omega-1} \tilde{P}_j(n,\omega), \quad (38)$$

where $\tilde{P}_j(n,\omega)$ denotes the probability of obtaining an integer l_j in register I, where

$$l_j = \left(\frac{2^n}{\omega}\right)j + \beta_j, \quad (39)$$

and β_j, a rational number, ranging from $-1/2$ to $1/2$, ensures that l_j is an integer. Introducing the bandwidth b_{PF} in the PF part of Shor's algorithm and defining $\tilde{P}_j(n,b_{\mathrm{PF}},\omega)$ as the probability of obtaining $|l_j\rangle$ as a readout of register I when BQFT instead of the exact QFT is used, we define, with (38), the scaled performance as

$$P(n,b_{\mathrm{PF}},\omega) = \frac{\tilde{P}(n,b_{\mathrm{PF}},\omega)}{\tilde{P}(n,b_{\mathrm{PF}} = n-1,\omega)}, \tag{40}$$

where

$$\tilde{P}(n,b_{\mathrm{PF}},\omega) = \sum_{j=0}^{\omega-1} \tilde{P}_j(n,b_{\mathrm{PF}},\omega). \tag{41}$$

The scaled and the absolute success probabilities (40) and (41), respectively, serve as the basis for our performance measure. We note that the choice of a single state around $|l_j\rangle$ as a proxy for the performance is well justified since all states under a Fourier peak respond in unison to a varying bandwidth b_{PF} (see Ref. [3]).

Supplied with the exact ME part of Shor's algorithm, the initial input state in register I of the PF part reads

$$|\psi_i\rangle_{\mathrm{I}} = \frac{1}{\sqrt{K(s_0)}} \sum_{k=0}^{K(s_0)-1} |s_0 + k\omega\rangle_{\mathrm{I}}, \tag{42}$$

where $K(s_0)$ is the number of elements in the equivalence class

$$[s_0] = \{s_0 + k\omega, 0 \leqslant k \leqslant K(s_0) - 1\}, \tag{43}$$

where the representative s_0 ranges between 0 and $\omega - 1$, inclusively. Since the number space in register I ranges from 0 to $2^n - 1$, where n is the number of qubits in register I, which, for a semiprime $N = pq$, is defined as

$$n = \lfloor 2\log_2(N) + 1 \rfloor, \tag{44}$$

where $\lfloor \cdots \rfloor$ is the floor function [37], we note that on average $K(s_0)$ is $2^n/\omega$.

Now applying the BQFT with bandwidth b_{PF} defined in (35) on (42), we obtain the final output state

$$|\psi_f(b_{\mathrm{PF}})\rangle_{\mathrm{I}} = \frac{1}{\sqrt{2^n K(s_0)}} \sum_{k=0}^{K(s_0)-1} \prod_{m=0}^{n-1} \times \sum_{j=0}^{1} e^{2\pi i[.s(k)_{[m]}s(k)_{[m-1]}\cdots s(k)_{[m-b_{\mathrm{PF}}]}]j} |j\rangle_{\{n-m-1\}}, \tag{45}$$

where

$$s(k) = s_0 + k\omega. \tag{46}$$

Thus, with (40) and (41), the normalized success probability is

$$P(n,b_{\mathrm{PF}},\omega) = \frac{\sum_{j=0}^{\omega-1} |\langle l_j|\psi_f(b_{\mathrm{PF}})\rangle_{\mathrm{I}}|^2}{\sum_{j=0}^{\omega-1} |\langle l_j|\psi_f(b_{\mathrm{PF}} = n-1)\rangle_{\mathrm{I}}|^2}. \tag{47}$$

Since we are interested in the effect of bandwidth b_{PF} on the performance of a quantum computer, specifically how it scales in n, we need to average out the remaining argument ω in $P(n,b_{\mathrm{PF}},\omega)$ above. Defining the order-averaged scaled performance as

$$P_N(n,b_{\mathrm{PF}}) = \frac{\sum_{k=1}^{a(N)} \nu(\omega_k) P(n,b_{\mathrm{PF}},\omega_k)}{\sum_{k=1}^{a(N)} \nu(\omega_k)}, \tag{48}$$

where $a(N)$ is the number of useful orders for a given semiprime N and $\nu(\omega)$ is the multiplicity of a given order ω, i.e., the number of seeds x of the order ω, we present the order-averaged scaled performance (48) for n ranging from 9 to 40 in Fig. 7. Testing our earlier results in Refs. [3,10,11] which imply that quantum computers follow the scaling law

$$P_{b_{\mathrm{PF}}}(n) = 2^{-1.1\times 2^{-2b_{\mathrm{PF}}}(n-8)}, \tag{49}$$

we report here that the new results confirm the scaling law up to $n = 40$. This is at the limit of what can be achieved on our current computer facility, a 128-core cluster computer. To generate the additional $n = 40$ data point in Fig. 7, we chose

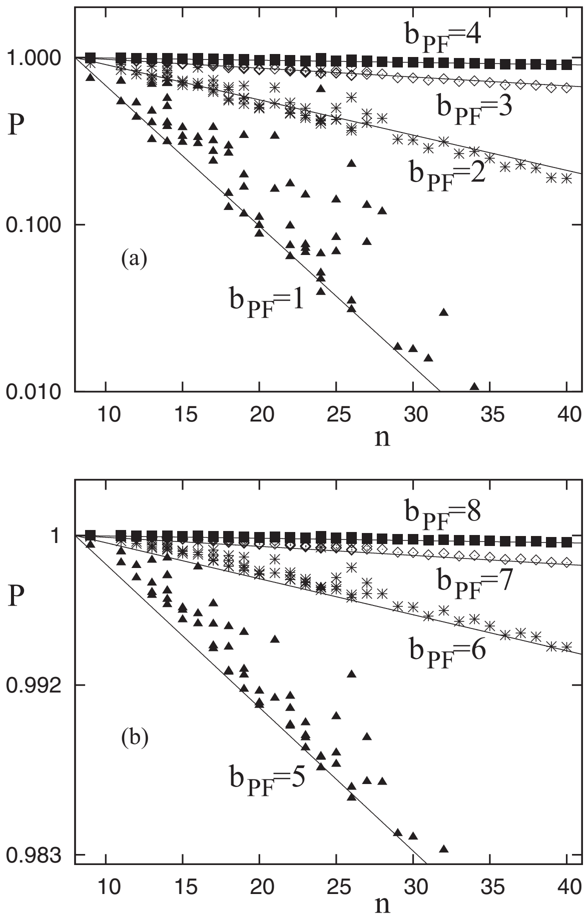


FIG. 7. Mode A scaled performance measure P represented by the properly averaged success probability (48) for successful factorization of sample semiprimes N. With bit length of $N \sim n/2$, the normalized probability is shown as a function of n for eight different PF-bandwidths. (a) $b_{\mathrm{PF}} = 1$ (triangles), $b_{\mathrm{PF}} = 2$ (asterisks), $b_{\mathrm{PF}} = 3$ (diamonds), and $b_{\mathrm{PF}} = 4$ (squares). (b) $b_{\mathrm{PF}} = 5$ (triangles), $b_{\mathrm{PF}} = 6$ (asterisks), $b_{\mathrm{PF}} = 7$ (diamonds), and $b_{\mathrm{PF}} = 8$ (squares). Solid lines through the data points are the fit functions (49).

$N = 1{,}034{,}273 = 1013 \times 1021$ since its two prime factors are relatively close to each other, a case known to be hard to factor [38]. Concerning the data displayed in Fig. 7, we note that there are up to 72 different orders for a chosen N, and only after averaging over all these different orders does the data corresponding to a given N appear as a single point in Fig. 7.

Absolute performance data may be obtained analytically from the scaled performance data in Fig. 7. First, noticing that the absolute performance of our virtual quantum computer running in mode A with bandwidth $b_{\rm PF}$ is

$$\tilde{P}(n,b_{\rm PF},\omega) = P(n,b_{\rm PF},\omega)\sum_{j=0}^{\omega-1} |\langle l_j|\psi_f(b_{\rm PF}=n-1)\rangle_{\rm I}|^2, \tag{50}$$

where we used (40) and (47), we immediately see that the analytical expression of the sum in (50), which is the same as $\tilde{P}(n,\omega)$ in (38), will yield the desired absolute performance conversion right away. Applying a full bandwidth QFT to (42) and suppressing the argument s_0 of K, we obtain

$$|\psi_f\rangle_{\rm I} = \frac{1}{\sqrt{2^n K}}\sum_{l=0}^{2^n-1}\sum_{k=0}^{K-1} e^{2\pi i \frac{(s_0+k\omega)}{2^n} l}|l\rangle, \tag{51}$$

where $|\psi_f\rangle_{\rm I}$ indicates the final output state of register I. Evaluating the sum in (50) together with (39) and (51), we obtain the analytical absolute performance measure of the quantum computer as

$$\begin{aligned}\tilde{P}(n,\omega) &= \sum_{j=0}^{\omega-1} |\langle l_j|\psi_f(b_{\rm PF}=n-1)\rangle_{\rm I}|^2\\ &= \sum_{j=0}^{\omega-1}\frac{1}{2^n K}\left|\sum_{k=0}^{K-1} e^{2\pi i k\omega l_j/2^n}\right|^2\\ &= \sum_{j=0}^{\omega-1}\frac{\sin^2(K\pi\omega l_j/2^n)}{2^n K\,\sin^2(\pi\omega l_j/2^n)}.\end{aligned} \tag{52}$$

Inserting (52) in (50), we obtain the desired conversion

$$\tilde{P}(n,b_{\rm PF},\omega) = P(n,b_{\rm PF},\omega)\sum_{j=0}^{\omega-1}\frac{\sin^2(K\pi\omega l_j/2^n)}{2^n K\,\sin^2(\pi\omega l_j/2^n)}. \tag{53}$$

Applying the order averaging in analogy to (48), we obtain

$$\tilde{P}_N(n,b_{\rm PF}) = \frac{\sum_{k=1}^{a(N)} \nu(\omega_k)\tilde{P}(n,b_{\rm PF},\omega_k)}{\sum_{k=1}^{a(N)} \nu(\omega_k)} \tag{54}$$

as the order-averaged absolute performance measure of the quantum computer running Shor's algorithm factoring N in mode A. Numerical results according to (54) are shown in Fig. 8, where the solid lines are

$$\tilde{P}_{b_{\rm PF}}(n) = 0.774 \times 2^{-1.1\times 2^{-2b_{\rm PF}}(n-8)}. \tag{55}$$

We note that compared to the solid lines in Fig. 7 [see (49)], (55) is a factor 0.774 smaller, but is otherwise identical with (49).

The factor 0.774 arises due to the following reason. Previously, in Ref. [3], it has been demonstrated that the

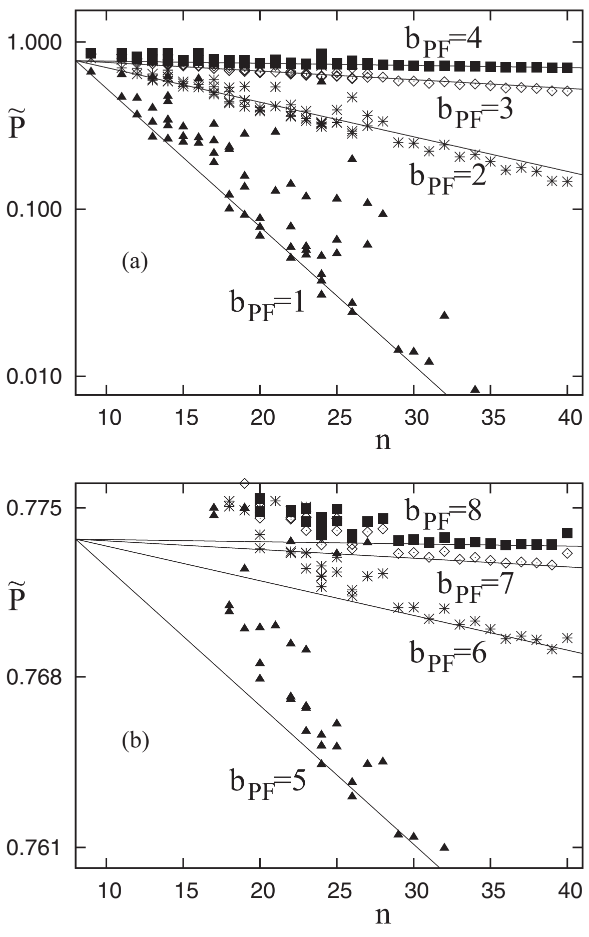


FIG. 8. Mode A absolute performance measure $\tilde{P}$ represented by the properly averaged success probability (54) for successful factorization of sample semiprimes N. With bit-length of $N \sim n/2$, the absolute probability is shown as a function of n for 8 different PF-bandwidths. (a) $b_{\rm PF} = 1$ (triangles), $b_{\rm PF} = 2$ (asterisks), $b_{\rm PF} = 3$ (diamonds), and $b_{\rm PF} = 4$ (squares). (b) $b_{\rm PF} = 5$ (triangles), $b_{\rm PF} = 6$ (asterisks), $b_{\rm PF} = 7$ (diamonds), and $b_{\rm PF} = 8$ (squares). Solid lines through the data points are the fit functions (55).

average order for a given odd, non-complete-square semiprime N scales like

$$\langle\langle\omega\rangle\rangle \approx N/5, \tag{56}$$

where the inner average indicates order averaging, i.e.,

$$\langle\omega\rangle = \frac{\sum_{k=1}^{a(N)} \nu(\omega_k)\omega_k}{\sum_{k=1}^{a(N)} \nu(\omega_k)}, \tag{57}$$

and the outer average denotes our binning process [3], i.e.,

$$\begin{aligned}\langle\langle\omega\rangle\rangle(N^{(i)}) &= \frac{1}{\chi(N^{(i)}+250)-\chi(N^{(i)}-250)}\\ &\quad\times\sum_{\lambda=\chi(N^{(i)}-250)+1}^{\chi(N^{(i)}+250)}\langle\omega\rangle_\lambda,\\ N^{(i)} &= 500\left(i-\frac{1}{2}\right),\quad i=1,\ldots,20,\end{aligned} \tag{58}$$

where $\chi(N)$ is the odd, non-complete-square semiprime counting function and $\langle\omega\rangle_\lambda$ is the average ω in (57) for the

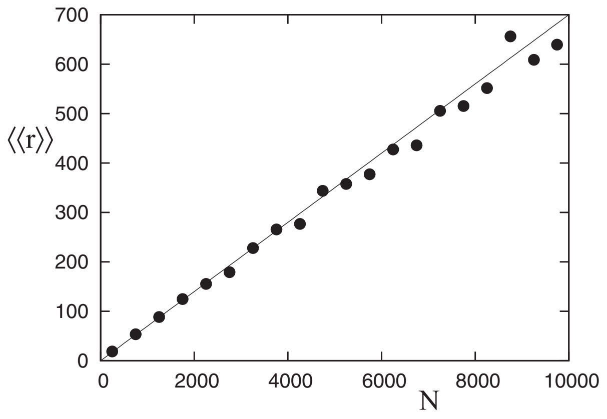


FIG. 9. Doubly-averaged $\langle\langle r\rangle\rangle$ defined according to (58) as a function of N. The solid line through the data points is the fit function (63).

λth semiprime. Using (39) in (52), we obtain

$$\sum_{j=0}^{\omega-1}\frac{\sin^2(K\pi\omega l_j/2^n)}{2^nK\sin^2(\pi\omega l_j/2^n)}=\sum_{j=0}^{\omega-1}\frac{\sin^2(K\pi\omega\beta_j/2^n)}{2^nK\sin^2(\pi\omega\beta_j/2^n)},\qquad(59)$$

where we used the π periodicity of the square of the sine function. Motivated by an increasing $\langle\langle\omega\rangle\rangle$ in N, we would like to transform the sum in (59) into an integral as an approximation whose accuracy increases for increasing ω. A closer inspection of (39), however, shows that l_j has a periodic structure with periodicity r given by

$$\omega=2^\alpha r,\qquad(60)$$

where r is the odd part of ω. This allows us to simplify (59) once more to

$$\sum_{j=0}^{\omega-1}\frac{\sin^2(K\pi\omega\beta_j/2^n)}{2^nK\sin^2(\pi\omega\beta_j/2^n)}=2^\alpha\sum_{j=0}^{r-1}\frac{\sin^2(K\pi\omega\beta_j/2^n)}{2^nK\sin^2(\pi\omega\beta_j/2^n)},\qquad(61)$$

where β_j now is in the form

$$\beta_j=\frac{\zeta(j)}{r}\qquad(62)$$

with $|\zeta(j)|<r/2$ an integer that is uniformly distributed between $-r/2$ and $r/2$. This inspires us to investigate how r scales in n since an r increasing in n will allow us to employ the integral approximation we would like to use in (61). Numerical results for $\langle\langle r\rangle\rangle$ with the same double average as used in (56) are shown in Fig. 9. The fit line has an N dependence of

$$\langle\langle r\rangle\rangle=\frac{7}{100}N,\qquad(63)$$

which implies that we may indeed use the integral transform to approximate the sum in (61). Using (i) $K\approx 2^n/\omega$ and (ii) $\omega<N\ll 2^n$ in (61) and turning the sum into an integral ranging from $-1/2$ to $1/2$ with β now a continuous variable, we obtain

$$2^\alpha\sum_{j=0}^{r-1}\frac{\sin^2(K\pi\omega\beta_j/2^n)}{2^nK\sin^2(\pi\omega\beta_j/2^n)}\approx\frac{1}{r}\sum_{j=0}^{r-1}\frac{\sin^2(\pi\beta_j)}{(\pi\beta_j)^2}\approx\int_{-1/2}^{1/2}\frac{\sin^2(\pi\beta)}{(\pi\beta)^2}d\beta,\qquad(64)$$

where

$$\int_{-1/2}^{1/2}\frac{\sin^2(\pi\beta)}{(\pi\beta)^2}d\beta\approx 0.774.\qquad(65)$$

This completes the explanation of the origin of the factor 0.774 in (55).

As an aside we note a typo in Eqs. (65) and (120) in Ref. [3]: $\varphi_E(N)$ in these two equations needs to be replaced by the proper normalization $\sum_{k=1}^{a(N)}\nu(\omega_k)$, where the sum is over all *useful* orders ω_k, as in Eqs. (48) and (57) in this paper. However, since the computations in Ref. [3] had been done with the correct normalization, the results and conclusions in Ref. [3] remain unchanged.

In Fig. 8 we observe that the performance scaling in the low n region is not as well represented by the fit line (55) as in the higher n region, especially for the larger bandwidth cases. This is no surprise since the integral approximation holds well for the large n region only. We also expect this deviation for the following reason. In case ω is a power of 2, i.e.,

$$\omega=2^\alpha,\qquad(66)$$

we note that

$$K=\frac{2^n}{\omega}=2^{n-\alpha}\qquad(67)$$

for any s_0, and in (39)

$$\beta_j=0\qquad(68)$$

since $l_j=jK$. Inserting (67) and (68) into (52), we obtain

$$\tilde{P}(n,\omega=2^\alpha)=1.\qquad(69)$$

Combined with the result proved in Sec. IV of Ref. [3], i.e., that for $\omega=2^\alpha$ the scaled performance of the quantum computer is 100% for any bandwidth $b_{\rm PF}$, we have

$$\tilde{P}(n,b_{\rm PF},\omega=2^\alpha)=\tilde{P}(n,b_{\rm PF}=n-1,\omega=2^\alpha).\qquad(70)$$

We conclude that, together with (69),

$$\tilde{P}(n,b_{\rm PF},\omega=2^\alpha)=1.\qquad(71)$$

This means that we can decompose the fit line (55) into two parts: (i) power 2 orders that result in perfect performance and (ii) nonpower 2 orders that result in an imperfect performance. Defining the weight μ_N for a given semiprime N in terms of the multiplicity of power 2 orders according to

$$\mu_N=\frac{\sum_{k=1}^{a(N)}\nu(\omega_k)\theta(\omega_k)}{\sum_{k=1}^{a(N)}\nu(\omega_k)},\qquad(72)$$

where $\theta(\omega)$ is a binary function defined as

$$\theta(\omega=2^\alpha r)=\begin{cases}0, & \text{if}\quad r\neq 1,\\ 1, & \text{if}\quad r=1,\end{cases}\qquad(73)$$

we observe that μ_N shows a power-law behavior in N. To extract the power, we bin μ_N in (72) logarithmically, i.e.,

$$\mu(\overline{N}_i)=\frac{1}{\chi\left(10^{\log_{10}\overline{N}_i+\frac14}\right)-\chi\left(10^{\log_{10}\overline{N}_i-\frac14}\right)}\times\sum_{\lambda=\chi(10^{\log_{10}\overline{N}_i-\frac14})+1}^{\chi(10^{\log_{10}\overline{N}_i+\frac14})}\mu_\lambda,$$

$$\overline{N}_i=10^{\frac{2i+3}{4}},\quad i=1,\ldots,6,\qquad(74)$$

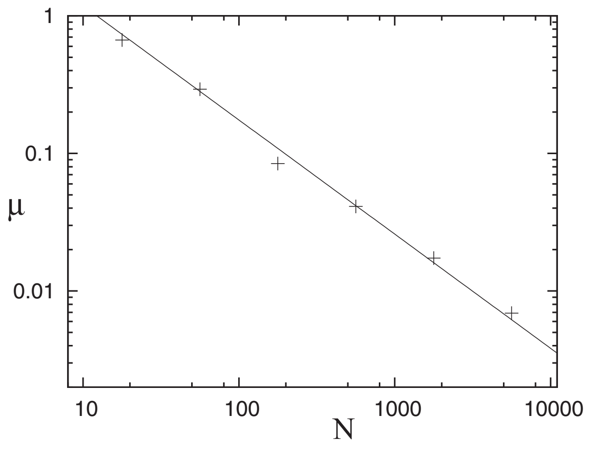


FIG. 10. Power 2 order multiplicity μ as a function of N, binned according to (74). The solid line is the fit function (75).

where μ_λ is the μ in (72) for the λth semiprime. Figure 10 shows μ as a function of N as a log-log-plot with N ranging up to 10^4. We find numerically

$$\mu \approx \frac{8}{N^{0.83}}, \tag{75}$$

which is the solid line shown in Fig. 10. Together with

$$N \sim 2^{n/2}, \tag{76}$$

implied by our choice of n in (44), we obtain

$$\mu \approx 2^{-0.415n+3}. \tag{77}$$

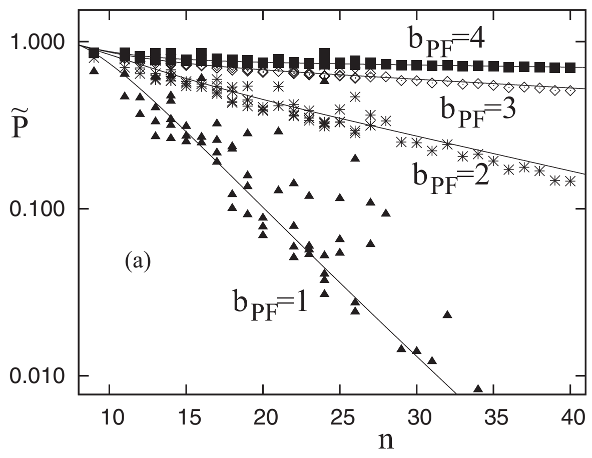
Decomposing the $\tilde{P}_{b_{\rm PF}}(n)$ fit line (55) into the two parts (i) and (ii), we obtain with (77)

$$\tilde{P}_{b_{\rm PF}}(n) = \mu + (1-\mu) \times 0.774 \times 2^{-1.1\times 2^{-2b_{\rm PF}}(n-8)}. \tag{78}$$

As shown in Fig. 11, we find that the new fit line (78) with the power 2 order correction is in better agreement with the absolute performance data than the original fit line (55). We note that even the new line does not fit the data perfectly, however, since the scaling in the low n region, especially for large $b_{\rm PF}$, is not exponential to start with (see Ref. [3]). In addition, the poor quality of the integral approximation [see (64)] in this regime results in the scattering and the deviation of data points from the fit line (78), which becomes more prominent as N becomes smaller since the smaller N, the fewer the number of orders of N that are present. To see this better, we plot our data with extended range in $\tilde{P}$ in Fig. 12. We observe that in the low n regime the new line (78), as it should, fits the data much better than the old line (55) with some remaining, but noticeable, scattering of the plot symbols around the improved fit line.

V. MODE B: COMPLETE IMPLEMENTATION OF SHOR'S ALGORITHM

In Sec. IV we investigated a hybrid implementation of Shor's algorithm, providing the quantum computer with the classically computed ME result. In this section we study a complete implementation of Shor's algorithm where both ME and PF are performed quantum mechanically. We show that a quantum computer running Shor's algorithm, when



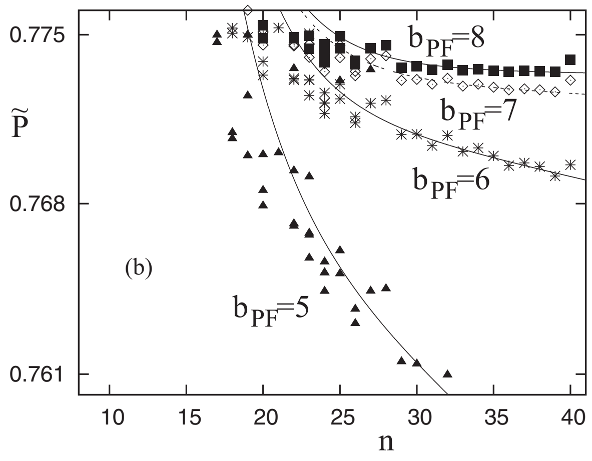


FIG. 11. Mode A absolute performance measure $\tilde{P}$ for sample semiprimes N (as a function of n as in Fig. 8) with improved fit functions (78) (solid lines). The PF bandwidth $b_{\rm PF}$ ranges from $b_{\rm PF} = 1$ to $b_{\rm PF} = 8$. (a) $b_{\rm PF} = 1$ (triangles), $b_{\rm PF} = 2$ (asterisks), $b_{\rm PF} = 3$ (diamonds), and $b_{\rm PF} = 4$ (squares). (b) $b_{\rm PF} = 5$ (triangles), $b_{\rm PF} = 6$ (asterisks), $b_{\rm PF} = 7$ (diamonds), and $b_{\rm PF} = 8$ (squares).

constructed according to Beauregard's method [15] (see Sec. II), is robust against QFT and QFA banding defined in Sec. III. Specifically, we report here the scaling law of its performance and compare it to numerical results.

Our mode-B implementation is based on the circuits presented in Ref. [15]; we build ME from MMs that can be decomposed into consecutive MAs, which we construct by combining adders that are executed in Fourier space and modular parts that are executed using qubit recycling (see Sec. II B for the explicit construction). Assessing the number of qubits needed for running our simulation in mode B, we note that the number of qubits used in the simulation is, for L the bit length of the semiprime N to be factored, (i) $2L+2$ for the ME part and (ii) $2L$ for the PF part, making the total number of qubits in the simulated quantum computer $4L+2$. This is so because QFA, for instance, requires $L+1$ qubits for adding two L-bit integers, where the additional qubit arises from the need to have an overflow qubit. Since the modulo part demands an auxiliary qubit to be used, the computational register, which is capable of executing the quantum MA, then, consists of $L+2$ qubits. Since MM now is constructed via consecutive applications of MA, requiring $L+2$ qubits, and since the MM

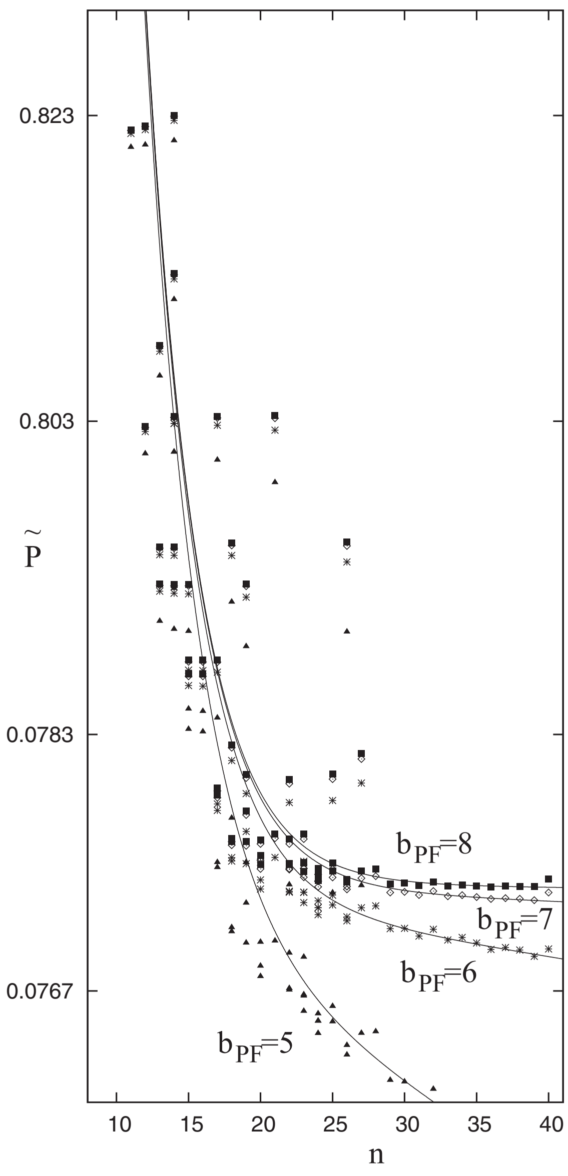


FIG. 12. Extended presentation of mode A absolute performance $\tilde{P}$ for sample semiprimes N as a function of n for bandwidth $b_{\rm PF} = 5$ (triangles), $b_{\rm PF} = 6$ (asterisks), $b_{\rm PF} = 7$ (diamonds), and $b_{\rm PF} = 8$ (squares). Solid lines are the fit functions in (78).

operation needs a quantum register that stores the intermediate L-bit computational results from the previous MM operation, we need a total of $2L + 2$ qubits to realize the MM gate defined in (12). Since no extra qubits are required to obtain ME from MM as shown in (11), we conclude that $2L + 2$ qubits are required to realize ME. Counting $2L$ qubits used for the PF part of Shor's algorithm [6], we obtain $4L + 2$ as the total number of physical qubits. We remark that, in principle, a bus qubit is necessary for running a quantum computer. Hence, the effective total number of qubits that are simulated on our virtual quantum computer is $4L + 3$.

Implementing the quantum circuit presented in Ref. [15], this time, we build the ME part of Shor's algorithm equipped with BQFT and BQFA as it would be used on an actual quantum computer. We choose, as mentioned previously, the number of qubits in register I that is to be used for the PF part of Shor's algorithm to be

$$n = 2L \tag{79}$$

and the number of qubits that contain the result of ME to be

$$n' = L. \tag{80}$$

Unlike in mode A (see Sec. IV), this time, the PF part of Shor's algorithm is not fed with input states that are periodic in ω [see (43)]. Rather it is provided with the result of the banded ME. In general, with $b_{\rm ME}$ the bandwidth imposed on the BQFT and the BQFA in the ME part, the initial state reads

$$|\psi_i\rangle = \frac{1}{\sqrt{2^n}} \sum_{r=0}^{2^n-1} \left(|r\rangle_{\rm I} \sum_{y=0}^{2^{n'}-1} |y\rangle_{\rm II\,A} \left\{ \sum_{y'=0}^{2^{n'+1}-1} |y'\rangle_{\rm II\,B} \times \left[\sum_{c=0}^{1} |c\rangle \left(\sum_{\beta=0}^{1} \Lambda^{(b_{\rm ME})}_{r,y,y',c,\beta} |\beta\rangle \right) \right] \right\} \right), \tag{81}$$

where $|r\rangle_{\rm I}$ represents register I states, $|y\rangle_{\rm II\,A}$ contains the ME results, $|y'\rangle_{\rm II\,B}$ indicates the states of the quantum register employed in MA, including the overflow qubit, $|c\rangle$ is the auxiliary qubit used in MA, and $|\beta\rangle$ is the bus qubit state of the quantum computer with the associated amplitudes $\Lambda^{(b_{\rm ME})}$, whose normalization is

$$\sum_{y=0}^{2^{n'}-1} \sum_{y'=0}^{2^{n'+1}-1} \sum_{c=0}^{1} \sum_{\beta=0}^{1} \left| \Lambda^{(b_{\rm ME})}_{r,y,y',c,\beta} \right|^2 = 1. \tag{82}$$

We note that since each elementary quantum gate is executed exactly, introducing bandwidth to our circuit does not alter the bus state. Hence, we write

$$|\psi_i\rangle = \frac{1}{\sqrt{2^n}} \sum_{r=0}^{2^n-1} \left\{ |r\rangle_{\rm I} \sum_{y=0}^{2^{n'}-1} |y\rangle_{\rm II\,A} \times \left[\sum_{y'=0}^{2^{n'+1}-1} |y'\rangle_{\rm II\,B} \left(\sum_{c=0}^{1} |c\rangle \Lambda^{(b_{\rm ME})}_{r,y,y',c} \right) \right] \right\}, \tag{83}$$

where we suppressed the bus state $|\beta\rangle = |0\rangle$ here and in the following, and used

$$\Lambda^{(b_{\rm ME})}_{r,y,y',c,\beta} = \delta_{\beta,0} \Lambda^{(b_{\rm ME})}_{r,y,y',c}, \tag{84}$$

where $\delta_{m,m'}$ is 1 if $m = m'$ and is 0 if $m \neq m'$.

Applying the BQFT with bandwidth $b_{\rm PF}$ in the PF part of Shor's algorithm, and recalling that the performance of the quantum computer is defined as the probability of obtaining any one of l_j in (39), we obtain the absolute performance of the quantum computer running Shor's algorithm in mode B with seed x and its associated order ω_x as

$$\tilde{P}(n,b_{\rm ME},b_{\rm PF},\omega_x;x) = \sum_{j=0}^{\omega_x-1} \sum_{y=0}^{2^{n'}-1} \sum_{y'=0}^{2^{n'+1}-1} \sum_{c=0}^{1} \times \left| \frac{1}{\sqrt{2^n}} \sum_{r=0}^{2^n-1} \prod_{m=0}^{n-1} \Lambda^{(b_{\rm ME})}_{r,y,y',c} \times e^{2\pi i (.r_{[m]} r_{[m-1]} \cdots r_{[m-b_{\rm PF}]})(l_j)_{[n-m-1]}} \right|^2. \tag{85}$$

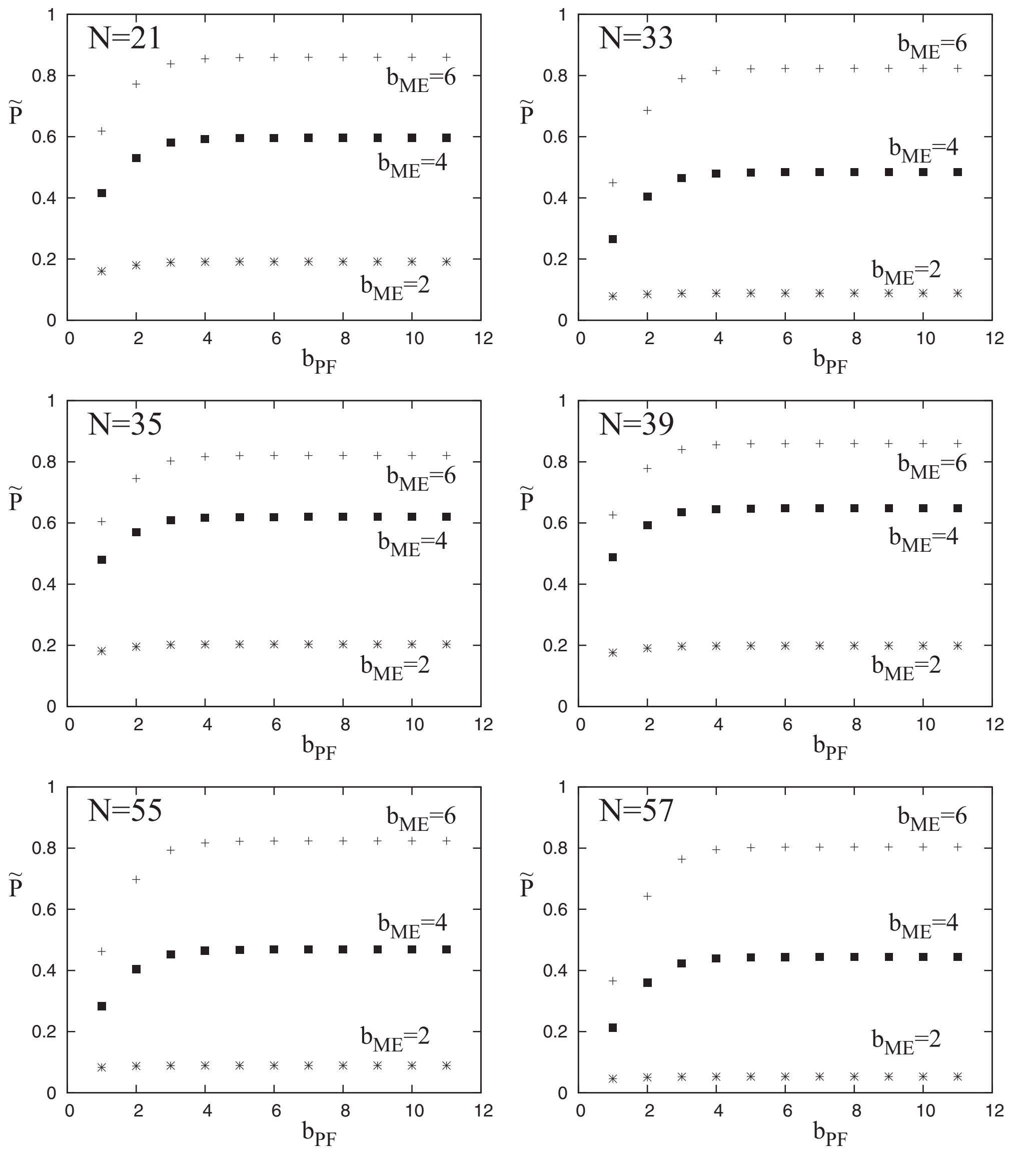


FIG. 13. Mode B absolute performance measure $\tilde{P}$ represented by the properly averaged success probabilities (86) for successful factorization of semiprimes $N = 21,33,35,39,55$, and 57 as a function of PF bandwidth b_{PF} for several ME bandwidths. $b_{ME} = 2$ (asterisks), $b_{ME} = 4$ (squares), and $b_{ME} = 6$ (crosses).

Since a semiprime N has multiple seeds that can be used to factor, we employ a seed averaging scheme to measure the average performance of the quantum computer

$$\tilde{P}_N(n,b_{ME},b_{PF}) = \frac{\sum_{\upsilon=1}^{\Upsilon(N)} \tilde{P}(n,b_{ME},b_{PF},\omega_{x(\upsilon)};x(\upsilon))}{\Upsilon(N)}, \qquad (86)$$

where $x(\upsilon)$ is the υth seed of N that is useful for factoring (see Sec. II) and $\Upsilon(N)$ is the useful seed counting function. In Fig. 13, we show the results of computing (86) for semiprimes $N = 21,33,35,39,55,$ and 57. We note that each semiprime has up to 30 useful seeds that need to be computed individually for different b_{ME} for the ME part of Shor's algorithm, then for different b_{PF} for the PF part of Shor's algorithm. The data for $N = 51$ are not shown due to the fact that all of its orders are powers of 2, leading to a nonscaling behavior in b_{PF}.

Normalizing the absolute performance given a seed x in (86) for each different bandwidth in the ME part of Shor's algorithm with that of the full bandwidth in the PF part of Shor's algorithm, we plot $1 - P$ in Fig. 14, where P is the scaled performance as defined in (40) together with seed averaging in (86), namely,

$$P_{N,b_{ME}}(b_{PF}) = \sum_{\upsilon=1}^{\Upsilon(N)} \frac{1}{\Upsilon(N)} \frac{\tilde{P}(n,b_{ME},b_{PF},\omega_{x(\upsilon)};x(\upsilon))}{\tilde{P}_N(n,b_{ME},b_{PF} = n-1,\omega_{x(\upsilon)};x(\upsilon))}. \qquad (87)$$

We find quantitatively that

$$1 - P_{N,b_{ME}}(b_{PF}) \approx 2^{-2b_{PF}} \qquad (88)$$

(see solid lines in Fig. 14). While for our current simulations ($L = 6$) the number of gates is relatively modest (of the order of 30 000), the number of quantum states that need to be processed by these gates is exponentially large. Therefore, the accuracy of our virtual quantum computer is currently $\approx 10^{-5}$.

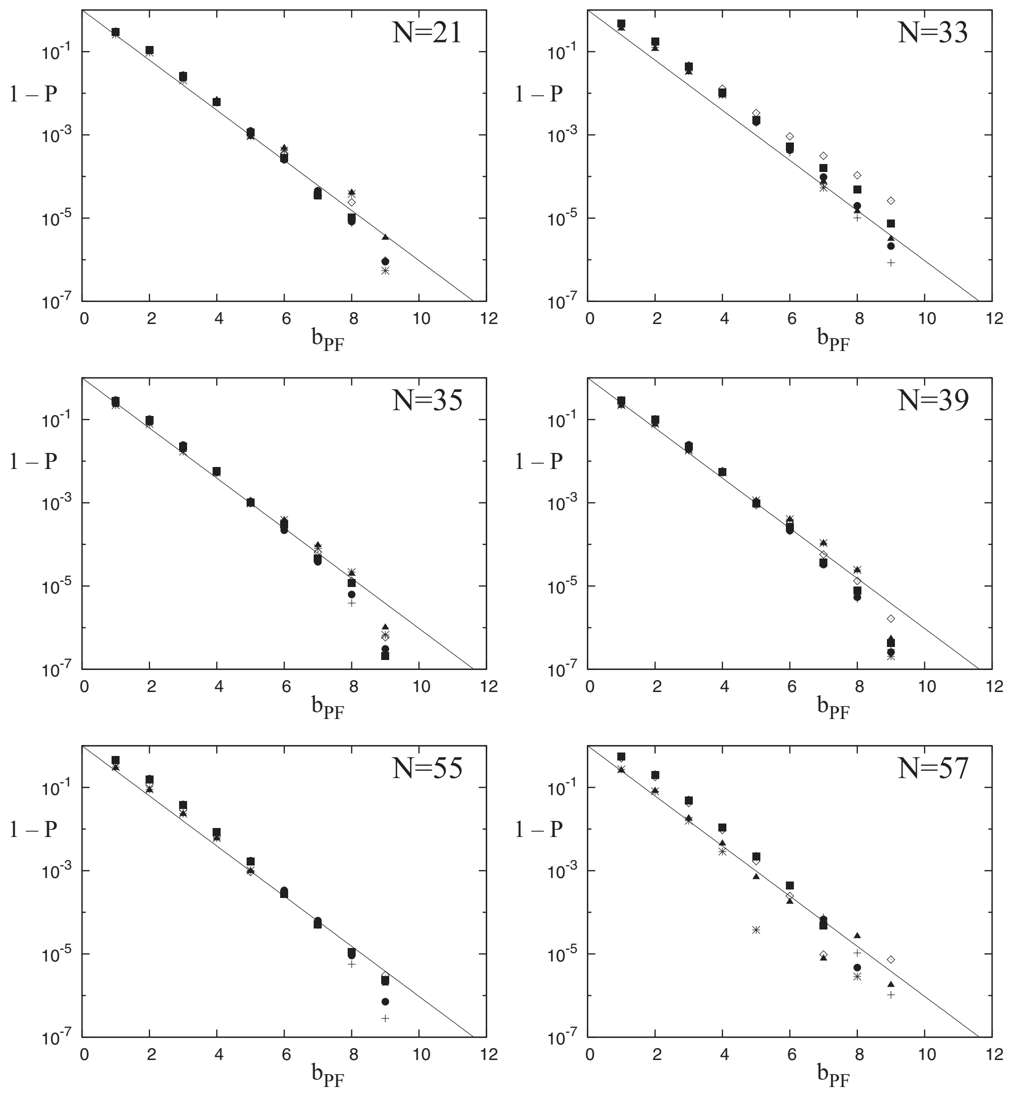


FIG. 14. Mode B PF bandwidth b_{PF} scaling of $1-P$ [see Eq. (87)] for semiprimes $N=21,33,35,39,55$, and 57 with a proper seed average over $\{x(\upsilon)\}$. The ME bandwidth ranges from $b_{\mathrm{ME}}=1$ to $b_{\mathrm{ME}}=6$; $b_{\mathrm{ME}}=1$ (triangles), $b_{\mathrm{ME}}=2$ (asterisks), $b_{\mathrm{ME}}=3$ (diamonds), $b_{\mathrm{ME}}=4$ (squares), $b_{\mathrm{ME}}=5$ (circles), and $b_{\mathrm{ME}}=6$ (pentagons). Solid lines are the fit functions (88).

Some of the data for $b_{\mathrm{PF}} \gtrsim 8$ are, therefore, omitted as they are at the limit of our accuracy.

VI. DISCUSSION

When it comes to factoring a large semiprime, currently available analytical and numerical techniques fail. However powerful the breakthroughs made in the past century, such as the quadratic number sieve [1] that routinely factors a 100-decimal-digit semiprime, or the more advanced general number field sieve (GNFS) [2] that was used to factor the RSA challenge number RSA-768 [39], they cannot factor even a moderate-sized semiprime of, say, 5000 decimal digits [3].

Quantum computation, a new paradigm of computing, changes the story entirely. Taking advantage of classically impossible logic operations (such as, e.g., the "square root of NOT"), a quantum computer, running Shor's algorithm, allows us to factor a large semiprime with exponential speedup when compared to its classical counterpart. Of course, even with today's quantum control techniques, we are far from realizing an actual quantum computer that can factor any meaningful semiprime that might lead to a security breach. This point is driven home by the fact that to date the record largest semiprime factored experimentally using a quantum computer is $N=21$ [28]. In addition, no experimental demonstration of quantum semiprime factoring has implemented a complete, scaling Shor algorithm. Instead, compiled, highly optimized versions of Shor's algorithm are used. Hence, any streamlining of quantum algorithms will help realize a scaling quantum computer. Precisely such a simplification of Shor's algorithm is suggested in our paper: The replacement of the full QFT and QFA with their banded versions. That such a replacement of the quantum circuitry works at all is not obvious. This is illustrated by the following example. Let us look at the result of a banded adder with the bandwidth $b=2$ that performs the addition $19+37$ modulo 64. The expected result is 56. However, as shown in Fig. 15, the adder performs far from ideally, producing the correct result, 56, in only 62% of the

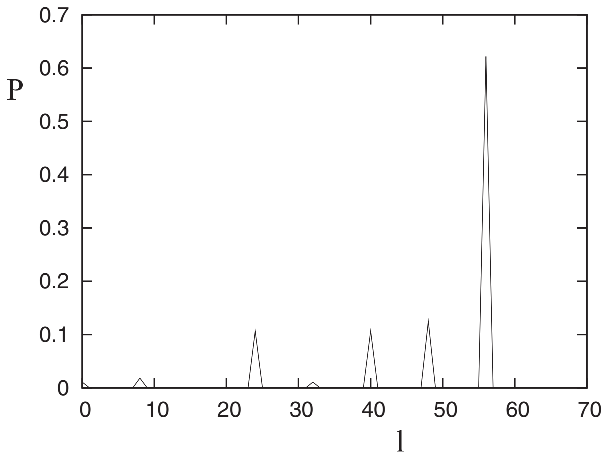


FIG. 15. Probability distribution of the results of the banded adder with the bandwidth $b = 2$ performing $19 + 37$ modulo 64.

cases. The adder also produces the manifestly wrong results 24, 40, and 48 with each more than 10% probability, in addition to the wrong results 0, 8, and 32 with $\sim$1% probability. That such an erroneous adder, which, for factoring $N = 57$, is applied more than 700 times, still allows us to factor semiprimes with acceptable probability is nothing short of astonishing. This points to a general principle of robustness of quantum computers with respect to static defects. Thus, without sacrificing performance, large numbers of quantum gates may be pruned, resulting in a streamlined architecture that is much more conducive to experimental implementation. In addition, we show how the banded Shor algorithm scales.

Simulations of Shor's algorithm themselves have been performed before, most of which have focused on parallel implementation on a multiprocessor classical computer. Notable ones include the following. (i) Obenland *et al.* [29] simulated Shor's algorithm with the adder shown in Ref. [13] and focused on the scaling of the speedup and the execution time in the number of classical processors. (ii) Niwa *et al.* [30] simulated Shor's algorithm with the adder in Ref. [23], which is very close to the one presented in Ref. [12]. They investigated the effects of decoherence and operational errors, using the classical ME results (akin to our mode A) for semiprimes $N > 15$. (iii) De Raedt *et al.* [31] illustrated the use of a portable software package simulating quantum computers on massively parallel classical supercomputers, implementing Shor's algorithm as a test case. The ME part of Shor's algorithm was simulated without the auxiliary qubit array. (iv) Tabakin *et al.* [32] created a convenient parallel computing environment. Other works more closely related to our work include (v) Fowler and Hollenberg [9], addressing the effect of bandwidth introduction to the PF part of Shor's algorithm and (vi) García-Mata *et al.* [33,34] focusing on the influence of noise on the performance of the quantum computer running Shor's algorithm using the classical ME results. The pioneering investigations described in (i) to (vi) all differ from our implementation in various ways, from different adder architectures to the presence or the absence of the auxiliary qubit array, with the ME or the PF parts of Shor's algorithm simulated fully coherently or not. Most importantly, however, our implementation focuses on the influence of simultaneous banding of the PF part and the QFT-based ME part of Shor's algorithm running on a virtual, scalable, fully coherent quantum computer including auxiliary qubits that are executed at an elementary quantum gate level.

We note that even though we use 128 classical computing cores simultaneously in our simulations, we are at the limit of our computing resources: Running our virtual quantum computer in its hybrid mode, it took us three months of CPU time to compute the data, as each semiprime needs to be factored with different orders individually, each with up to 72 different orders. This means that not only do we suggest banding for the purpose of practical implementation of quantum computers but also as a way to simulate larger virtual quantum computers with more qubits. Running our virtual quantum computer in its complete mode due to its memory requirement scaling exponentially in the number of qubits $4L + 3$, where L is the bit length of the semiprime to be factored, we are currently limited to $L \leqslant 6$. Nonetheless, this is enough to break the current experimental record of $N = 21$. Increasing L to 7, however, is within our reach and will be explored in future work.

For now it is interesting to explore how many rotation gate operations may be saved by banding the quantum computer. We start by counting the total number of rotation gates n_t of the ideal, nonbanded quantum computer of $n = 4L + 3$ qubits. We obtain

$$n_t = 18L^4 + 42L^3 + 26L^2 - L. \tag{89}$$

Introducing banding with bandwidth $b_{\rm PF}$ in the period finding part of Shor's algorithm, we save

$$n_{s,\rm PF} = \frac{(2L - b_{\rm PF})(2L - b_{\rm PF} - 1)}{2} \tag{90}$$

rotation gates. Introducing banding with bandwidth $b_{\rm ME}$ in the modular exponentiation part of Shor's algorithm, we save

$$n_{s,\rm ME} = (L - b_{\rm ME})(L - b_{\rm ME} + 1)(18L^2 + 4L) \tag{91}$$

rotation gates. Adding $n_{s,\rm PF}$ and $n_{s,\rm ME}$, we obtain the total number of saved rotation gates

$$\begin{aligned} n_s = {} & \frac{(2L - b_{\rm PF})(2L - b_{\rm PF} - 1)}{2} \\ & + (L - b_{\rm ME})(L - b_{\rm ME} + 1)(18L^2 + 4L). \end{aligned} \tag{92}$$

Although we factored semiprimes up to $N = 57$ and found that $b_{\rm PF} = 4$ and $b_{\rm ME} = 5$ is sufficient to factor these semiprimes with a 70% success rate, it is too early to extrapolate these results to the $n \sim 1000$ regime. This requires more statistics to be accumulated by running our virtual quantum computer for larger n. However, if the mode-A results are any indication, it is perhaps possible to factor large semiprimes on $n \sim 1000$ qubit quantum computers with $b_{\rm PF} = b_{\rm ME} = 8$. In this case, for $L = 2048$, according to (92), 314 378 411 210 788 rotation gates, or 99.167% of the total number of rotation gates, may be discarded, resulting in a substantial simplification of the required quantum circuitry.

Gate pruning, as suggested in this paper, does not reduce circuit depth. As a consequence, gate pruning will not affect the run time of Shor's algorithm as implemented in this paper. In the same vein, gate pruning does not affect the total number of qubits required in our implementation of Shor's algorithm.

The reduction of run time and number of qubits is a challenge for quantum algorithm development, a topic beyond the scope of our present paper. In the present paper, we take an existing quantum algorithm, i.e., Shor's algorithm, and investigate how it can be "trimmed" in a scalable way to actually have a chance to be implemented in practice. However, while not shortening execution time, or reducing the total number of required qubits, the possibility of gate pruning has profound advantages in terms of the actual realization of a scaling quantum computer. Since quantum gates are prone to error, any reduction in the number of quantum gates is an advantage when it comes to actually building a working quantum computer. Since according to our estimates, for a computer of contemporary interest, more than 99% of rotation gates may be trimmed from the quantum circuit with acceptable consequences for the performance, only 1% of rotation gates actually need to be realized. In our opinion, this is an important step forward toward the actual realization of scaling quantum computers.

VII. SUMMARY AND CONCLUSION

Using banding as a special way of optimizing Shor's algorithm, we investigated the performance of a quantum computer running Shor's algorithm in two different modes, a hybrid mode, called mode A, with classical ME and quantum mechanically executed PF, and a complete mode, called mode B, with quantum mechanically executed ME and PF. Mode A allows us to study the effects of BQFT, while mode B allows us to study the effects of BQFT and BQFA combined. Our contributions can be summarized in the following advances.

(1) Constructing a virtual 128-core quantum computer running Shor's algorithm based on the QFT as suggested in Ref. [15], fully decomposed into elementary quantum gates.

(2) Confirming an earlier result of exponential scaling of the normalized performance P in hybrid implementation (classical ME; quantum PF) up to and including $N = 1{,}034{,}273$, using $n = 40$ qubits, the largest virtual quantum computer simulated so far.

(3) An improved scaling law for the absolute performance $\tilde{P}$ in the hybrid mode of a virtual quantum computer using the doubly averaged $\langle\langle r\rangle\rangle$ and the weight of the power 2 orders μ.

(4) Absolute performance results of the complete implementation of Shor's algorithm when the QFA and the QFT in the ME part of Shor's algorithm are pruned via banding.

(5) Confirmation of the previously found [11] 2^{-2b} scaling of $1 - P$ of a virtual quantum computer running in its complete mode for additional semiprimes other than $N = 21$ such as $N = 33,35,39,55$, and 57. The data shown in this paper are seed-averaged (with up to 30 different seeds per semiprime) according to (86), whereas in Ref. [11] $N = 21$ was factored with the single seed $x = 2$. In addition the case $N = 21$ was run with $n = 12$ in this paper, whereas it was run with the lower $n = 10$ in Ref. [11]. This demonstrates the scalable nature of the quantum computer.

We believe that based on the numerical and the analytical work presented in this paper, in conjunction with Refs. [3,10,11], the PF part of Shor's algorithm, equipped with a BQFT, is well understood. This includes the absolute performance, an absolute lower bound on the factoring success of the banded Shor's algorithm running in hybrid mode. We have also simulated a more demanding part of Shor's algorithm, the ME part, fully coherently, keeping the exact quantum entanglement during the entire simulation, including the auxiliary qubits, factoring up to and including $N = 57$. Numerically confirming the scalability of our virtual quantum computer, we see a surprising robustness of the quantum computer when pruned with banding, resulting in substantial savings in required quantum circuitry.

ACKNOWLEDGMENTS

The authors are grateful for a generous allotment of computer resources by the Wesleyan Scientific Computing Center. R.B. acknowledges financial support by Grant No. 216687 of the Research Council of Norway.

[1] C. Pomerance, in *Computational Methods in Number Theory, Part I*, Math. Centre Tract, Vol. 154, edited by H. W. Lenstra, Jr., and R. Tijdeman (Mathematisch Centrum, Amsterdam, 1982), pp. 89–139.
[2] J. P. Buhler, H. W. Lenstra, Jr., and C. Pomerance, in *The Development of the Number Field Sieve*, Lecture Notes in Mathematics Vol. 1554, edited by A. K. Lenstra and H. W. Lenstra, Jr. (Springer, New York, 1993), pp. 50–94.
[3] Y. S. Nam and R. Blümel, Phys. Rev. A **87**, 032333 (2013).
[4] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography* (CRC, Boca Raton, FL, 1997).
[5] R. Rivest, A. Shamir, and L. Adleman, Comm. ACM **21**, 120 (1978).
[6] P. W. Shor, in *Proceedings of the 35th Annual Symposium on the Foundations of Computer Science*, edited by S. Goldwasser, (IEEE, Los Alamitos, CA, 1994), pp. 124–134.
[7] N. D. Mermin, *Quantum Computer Science* (Cambridge University Press, Cambridge, England, 2007).
[8] R. Blümel, *Foundations of Quantum Mechanics—From Photons to Quantum Computers* (Jones and Bartlett, Sudbury, MA, 2010).
[9] A. G. Fowler and L. C. L. Hollenberg, Phys. Rev. A **70**, 032329 (2004).
[10] Y. S. Nam and R. Blümel, Phys. Rev. A **86**, 044303 (2012).
[11] Y. S. Nam and R. Blümel, Phys. Rev. A **87**, 060304 (2013).
[12] V. Vedral, A. Barenco, and A. Ekert, Phys. Rev. A **54**, 147 (1996).
[13] D. Beckman, A. N. Chari, S. Devabhaktuni, and J. Preskill, Phys. Rev. A. **54**, 1034 (1996).
[14] C. Zalka, arXiv:quant-ph/9806084.
[15] S. Beauregard, Quantum Inf. Comput. **3**, 175 (2003).
[16] B.-S. Choi and R. Van Meter, ACM Journal on Emerging Technologies in Computing Systems **8**, 237 (2004).

[17] R. Van Meter and K. M. Itoh, Phys. Rev. A **71**, 052320 (2005).
[18] Y. Takahashi and N. Kunihiro, Quantum Inf. Comput. **6**, 184 (2006).
[19] S. A. Kutin, arXiv:quant-ph/0609001.
[20] R. B. Griffiths and C.-S. Niu, Phys. Rev. Lett. **76**, 3228 (1996).
[21] D. Coppersmith, arXiv:quant-ph/0201067.
[22] T. G. Draper, arXiv:quant-ph/0008033v1.
[23] C. Miquel, J. P. Paz, and R. Perrazo, arXiv:quant-ph/9601021v1.
[24] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, Nature (London) **414**, 883 (2001).
[25] C.-Y. Lu, D. E. Browne, T. Yang, and J.-W. Pan, Phys. Rev. Lett. **99**, 250504 (2007).
[26] B. P. Lanyon, T. J. Weinhold, N. K. Langford, M. Barbieri, D. F. V. James, A. Gilchrist, and A. G. White, Phys. Rev. Lett. **99**, 250505 (2007).
[27] A. Politi, J. C. F. Matthews, and J. L. O'Brien, Science **325**, 1221 (2009).
[28] E. Martín-López, A. Laing, T. Lawson, R. Alvarez, X.-Q. Zhou, and J. L. O'Brien, Nat. Photon. **6**, 773 (2012).
[29] K. M. Obenland and A. M. Despain, arXiv:quant-ph/9804039.
[30] J. Niwa, K. Matsumoto, and H. Imai, Phys. Rev. A **66**, 062317 (2002).
[31] K. De Raedt, K. Michielsen, H. De Raedt, B. Trieu, G. Arnold, M. Richter, Th. Lippert, H. Watanabe, and N. Ito, Comput. Phys. Comm. **176**, 121 (2007).
[32] F. Tabakin and B. Juliá-Díaz, Comput. Phys. Comm. **180**, 948 (2009).
[33] I. García-Mata, K. M. Frahm, and D. L. Shepelyansky, Phys. Rev. A **75**, 052311 (2007).
[34] I. García-Mata, K. M. Frahm, and D. L. Shepelyansky, Phys. Rev. A **78**, 062323 (2008).
[35] A. Pavlidis and D. Gizopoulos, Quantum Inf. Comput. **14**, 0649 (2014).
[36] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2000).
[37] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics: A Foundation for Computer Science*, 2nd ed. (Addison-Wesley, Reading, MA, 1994).
[38] N. Koblitz, *A Course in Number Theory and Cryptography* (Springer, New York, 1994).
[39] T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, H. te Riele, A. Timofeev, and P. Zimmermann, in *CRYPTO'10 Proceedings of the 30th Annual Conference on Advances in Cryptology* (Springer, Berlin, 2010), pp. 333–350.