Performance scaling of Shor's algorithm with a banded quantum Fourier transform

Y. S. Nam* and R. Blümel

Department of Physics, Wesleyan University, Middletown, Connecticut 06459-0155, USA

(Received 1 August 2012; published 23 October 2012)

In excellent agreement with our numerical simulations of Shor's algorithm, equipped with a truncated quantum Fourier transform of bandwidth b, we find that its performance scales $\sim 2^{-\xi_b n}$, where n is the number of qubits, $\xi_b = 1.1 \times 2^{-2b}$, and the bandwidth b is the number of quantum states coupled by the quantum Fourier transform. Nonexponential behavior is observed for small n and explained analytically. The large-n exponential scaling implies that $b = 7$ is sufficient to operate a 1000-qubit quantum computer running Shor's algorithm on the 95% performance level and implies hardware savings of the order of half a million rotation gates.

DOI: 10.1103/PhysRevA.86.044303 PACS number(s): 03.67.Lx

Together with Grover's database search algorithm [1,2], Shor's algorithm [3] is one of the most powerful quantum algorithms. Implemented on a quantum computer, Shor's algorithm is capable of cracking RSA encryption codes [4] that currently form the backbone of internet security [5,6]. No wonder, then, that Shor's algorithm has been presented and investigated in detail in the literature (see, e.g., [7–10]). So far, RSA is safe from attacks via Shor's algorithm, since quantum computers do not yet exist that could run Shor's algorithm for the large semiprimes N currently used in RSA encryption algorithms [4–6,8,9,11]. However, the more we simplify its hardware requirements, the sooner Shor's algorithm will come online in a regime relevant for Internet security. In this context, Coppersmith [10] was the first to notice that substantial hardware savings may be realized by eliminating, with negligible performance penalty, most of the quantum gates of the quantum Fourier transform [8,9], the centerpiece of Shor's algorithm. According to Coppersmith, this leaves an approximate quantum Fourier transform [10,12] that performs well when used in connection with Shor's algorithm. Written in matrix form, this Fourier transform has a banded appearance of bandwidth b, coupling only the b nearest neighbors of any given quantum state. Therefore, this type of Fourier transform is called a *banded quantum Fourier transform*. Testing the performance of the banded quantum Fourier transform in numerical experiments with semiprimes of up to 20 binary digits, we confirm that a banded quantum Fourier transform indeed leads to substantial hardware savings without seriously compromising the effectiveness of Shor's algorithm. In addition, improving on previous results [9,10,12], we present analytical scaling functions that measure the performance of Shor's algorithm as a function of the bandwidth and the number of qubits. Our analytical scaling functions agree well with our numerical results.

Shor's algorithm is based on Miller's algorithm [8,9,11] for factoring semiprime numbers $N = pq$, where $p \neq q$ are two prime numbers. Given a number $x < N$, relatively prime to N, the smallest power ω of x is determined such that $x^\omega \bmod N = 1$. x is called the seed and ω is called the order of x. Now, define $A = x^{\omega/2} - 1$ and $B = x^{\omega/2} + 1$. Then, if (i) ω is even and (ii) $B \neq 0 \bmod N$, the two prime factors of N are revealed as (up to ordering) $p = \gcd(A,N)$ and $q = \gcd(B,N)$, where gcd denotes the greatest common divisor. If condition i or ii is violated, we simply try another x. It is known [8,9] that finding a proper seed x has a high probability of success and very few x need to be tried before a suitable seed is found. Since determining ω on a classical computer is an algorithmically hard problem, it is impossible (with currently known classical factoring algorithms) to factor large semiprimes N on a classical computer if the number of decimal digits of N exceeds, say, 1000. To do so in a reasonable time (say, about a month) would require a classical computer exceeding the current size of the universe. This is where Shor's quantum algorithm comes in. Shor noticed [3] that determining ω on a quantum computer is exponentially faster than performing the same task on a classical computer, thus opening the possibility of factoring large N within reasonable run times.

The centerpiece of Shor's algorithm is a quantum Fourier transform,

$$|l\rangle = \frac{1}{\sqrt{2^n}} \sum_{s=0}^{2^n-1} \exp(2\pi i s l/2^n)|s\rangle, \tag{1}$$

where n is the number of qubits. A circuit diagram of (1), which also incorporates a measurement of the output state $|l\rangle$, is shown in Fig. 1(a). On a quantum computer, as shown in Fig. 1, the phases in (1) are realized by conditional phase rotation gates, which rotate the phase of the target qubit k by an angle

$$\theta_{|k-j|} = \frac{\pi}{2^{|k-j|}} \tag{2}$$

if the control qubit j is measured to be in state $|1\rangle$. We notice immediately that for large n, $\sim$1000, as required for factoring semiprimes N of practical interest, a complete implementation of quantum Fourier transform (1) would require the quantum hardware to distinguish phase differences of the order of $1/2^{1000} \approx 10^{-300}$, an impossible task under any circumstances. Not even fault-tolerant encoding [8,9] would solve this problem, since hardware, present or future, capable of distinguishing such small phase differences does not exist and cannot be built in our universe. This begs the question: What happens if we ignore gates that cannot be implemented? Eliminating unrealizable gates leads to a banded quantum Fourier transform [see, e.g., Fig. 1(b)] in which only the coupling to b nearest-neighbor qubits is retained [Fig. 1(b) illustrates the case $b = 1$]. Coppersmith [10] found

*ynam@wesleyan.edu

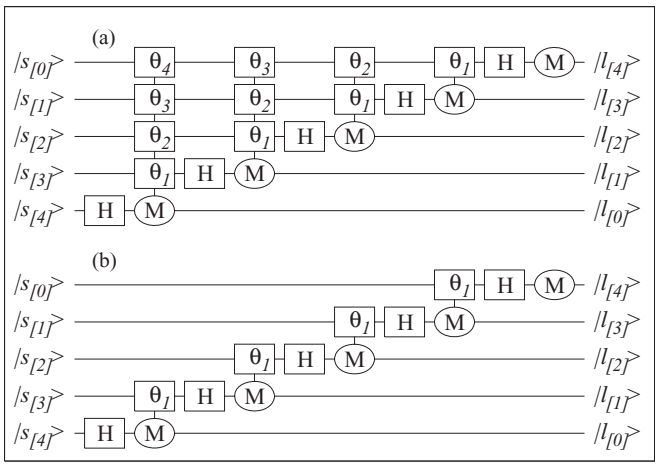

FIG. 1. Logic circuit of a five-qubit quantum Fourier transform, illustrating the concept of bandwidth, defined as the number b of off-diagonal quantum states coupled by the quantum Fourier transform. (a) Full implementation (bandwidth $b = 4$); (b) truncated implementation (bandwidth $b = 1$). H, θ, and M denote Hadamard, single-qubit conditional rotation, and measurement gates, respectively.

that truncation to a quantum Fourier transform of bandwidth b misses at most the phase

$$\Delta\varphi = 2\pi n 2^{-b-1}, \tag{3}$$

where n is the number of qubits. Coppersmith's estimate is rigorous but pertains only to the total "missing phase" of the quantum Fourier transform, which Coppersmith then qualitatively relates to the performance of Shor's algorithm [10]. An improvement of (3) was obtained by Fowler and Hollenberg [12], who directly investigated the performance of Shor's algorithm when equipped with a banded quantum Fourier transform. For the probability of a successful factorization, Fowler and Hollenberg obtained [12]

$$P^{(\mathrm{FH})} \sim 2^{-(n/2)/4^{b-1}} = 2^{-\xi_b^{(\mathrm{FH})} n}, \tag{4}$$

where

$$\xi_b^{(\mathrm{FH})} = 2 \times 2^{-2b}. \tag{5}$$

Since they pertain to different quantities, (3) and (4) cannot be compared directly. However, Fowler and Hollenberg's result (4) confirms Coppersmith's suggestion of a rapid (exponential) convergence of Shor's algorithm in b. While Fowler and Hollenberg used a stochastic model to arrive at their scaling result, (4), we use direct factorization of actual semiprimes N. This allows us to test the scaling relation (4) and derive an improved scaling relation for the performance of Shor's algorithm with a banded quantum Fourier transform.

Shor's algorithm determines the order ω of a seed x for a given semiprime N as the period of the function $f(t) = x^t \bmod N$, t integer, via a quantum Fourier transform that produces an output state $|\psi_{\mathrm{out}}\rangle$, consisting of a superposition of all n-qubit binary states $|l\rangle$. A measurement of $|\psi_{\mathrm{out}}\rangle$, then, has a high probability of collapsing into a state $|l_j\rangle$, where

$$l_j = \frac{2^n}{\omega} j + \beta_j, \quad j = 0, \ldots, \omega - 1, \tag{6}$$

and $-1/2 < \beta_j < 1/2$ guarantees that l_j is an integer. If ω is a power of 2, all β_j are 0 and Shor's algorithm performs perfectly, no matter what the bandwidth. Since for large N the probability of encountering this case is practically 0, we do not consider this special case any further. In the general case (the most probable case), where ω is not a power of 2, we write

$$\omega = 2^\alpha r, \tag{7}$$

where r and α are integers and r is odd. In this case most of the β_j are nonzero.

The probability of collapse into one of states (6) is

$$\tilde{P}(n,b) = \frac{1}{\omega K^2} \sum_{j=0}^{\omega-1} \left| \sum_{k=0}^{K-1} \exp\{i[\Phi(n,s_k,l_j) - \varphi(n,b,s_k,l_j)]\} \right|^2, \tag{8}$$

where

$$s_k = s_0 + k\omega, \quad k = 0, \ldots, K-1, \tag{9}$$

and s_0, $0 \leqslant s_0 \leqslant \omega - 1$. We found that our results are insensitive to the choice of s_0. We used $s_0 = 0$ ($s_0 = n - 1$) in our analytical (numerical) calculations. Denoting by $a_{[\nu]}$ the νth binary digit of an integer a,

$$\Phi(n,s,l) = \pi \sum_{m=0}^{n-1} \sum_{\mu=0}^{m} \frac{s_{[n-m-1]} l_{[\mu]}}{2^{m-\mu}}, \tag{10}$$

in (8) is the total phase angle experienced by a state $|s\rangle$ mapped into a state $|l\rangle$ by a full, n-qubit quantum Fourier transform ($b = n - 1$), and

$$\varphi(n,b,s,l) = \pi \sum_{m=b+1}^{n-1} \sum_{\mu=0}^{m-b-1} \frac{s_{[n-m-1]} l_{[\mu]}}{2^{m-\mu}} \tag{11}$$

is the "missing phase" due to the restriction of the bandwidth of the quantum Fourier transform to b off-diagonal couplings [see Fig. 1(b)]. The exact maximum value of φ, consistent with Coppersmith's bound (3), is

$$\varphi_{\max}(n,b) = 2\pi[2^{-b-1}(n-b) + 2^{-n} - 2^{-b}]. \tag{12}$$

We evaluate $\tilde{P}(n,b)$ according to the following procedure. Given n and b, we look for semiprimes $N = pq$ such that

$$n = \lfloor 2\log_2(N) + 1 \rfloor, \tag{13}$$

where $\lfloor z \rfloor$ is the floor function, i.e., the largest integer smaller than z. Then we randomly choose a seed x of order ω and compute $\tilde{P}(n,b)$ according to (8). Since the l_j values, (6), capture most (about 77%) but not all of the useful output of Shor's algorithm [a measurement of $|\psi_{\mathrm{out}}\rangle$ may collapse into an $|l\rangle$ state not contained in (6) but equally useful for factoring], we normalize $\tilde{P}(n,b)$ to its maximal-bandwith case $\tilde{P}(n,b = n-1)$ to obtain the normalized probability,

$$P(n,b) = \tilde{P}(n,b)/\tilde{P}(n,b = n-1). \tag{14}$$

We choose $P(n,b)$ in (14) as a quantitative measure for the performance of Shor's algorithm as a function of n and b.

Figure 2 shows $P(n,b)$ as a function of n for four values of b, obtained by factoring up to 12 sample semiprimes for each $n = 9, \ldots, 39$. Already for $b = 4$, i.e., without implementing most of the rotation gates required by a full Fourier transform,

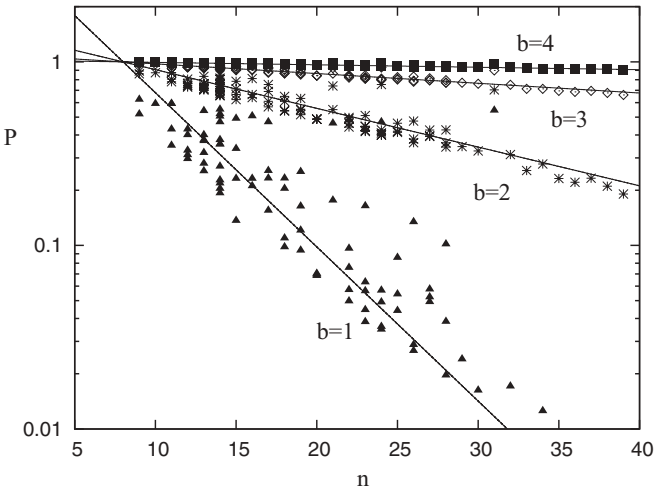

FIG. 2. Probability for successful factorization of sample semiprimes N of binary length $\log_2(N) \sim n/2$ [for details see (13)] using Shor's algorithm, supplied with a quantum Fourier transform of bandwidth b. Shown are the data for up to 12 randomly selected semiprimes N per n, factored with $b = 1$ (triangles), $b = 2$ (asterisks), $b = 3$ (diamonds), and $b = 4$ (squares). Solid lines are the scaling functions, (23), with decay constants, (24).

we observe excellent factoring performance in the 90% region over the entire range of $n < 40$. This observation has an immediate and important consequence for the construction of quantum computers: substantial hardware savings without significant sacrifice in performance. For fixed b and large n, as predicted by (4), we observe exponential decay of $P(n,b)$ as a function of n. However, as extracted from Fig. 2 and discussed further below, our decay constants ξ_b are somewhat smaller than predicted by (5), resulting in an even better performance of Shor's algorithm "in practice" than suggested by (5), based on a probabilistic model.

For small n, not noticeable on the scale of Fig. 2, $P(n,b)$ is not exponential. To analyze the small-n behavior, we start from (8) and note that for $\beta_j = 0$ the corresponding term in (8) contributes an amount $1/\omega$ to $\tilde{P}(n,b)$. Since $\beta_j = 0$ occurs whenever j/r is an integer, there are ω/r such cases, which contribute a total amount $1/r$ to $\tilde{P}(n,b)$. Turning now to the j terms with $\beta_j \neq 0$, we found that $\Phi(n,s_k,l_j)$ varies slowly with k, while $\varphi(n,b,s_k,l_j)$ is an erratic function of k. Therefore, we write j terms with $\beta_j \neq 0$ in (8) approximately as

$$\tilde{P}_j(n,b) \approx \frac{1}{\omega K^2} \left| \sum_{k=0}^{K-1} \exp[i\,\Phi(n,s_k,l_j)] \langle \exp(-i\varphi) \rangle \right|^2, \quad (15)$$

where $\langle\ \rangle$ indicates a statistical average over the probability distribution of φ. Numerically we found that, apart from isolated spikes, the distribution of φ is Gaussian in $\varphi/\varphi_{\max}$ with a variance of about $1/200$. Therefore, approximately,

$$\langle \exp[-i\varphi(n,b)] \rangle \approx \exp\left[-\varphi^2_{\max}(n,b)/200\right]. \quad (16)$$

The k sum in (15) can be evaluated analytically:

$$\sum_{k=0}^{K-1} \exp[i\,\Phi(n,s_k,l_j)] = \frac{\sin(\pi\beta_j)}{\sin(\pi\beta_j/K)}. \quad (17)$$

Since there are $(\omega - \omega/r)$ terms with $\beta_j \neq 0$, we obtain

$$\tilde{P}(n,b) \approx \frac{1}{r} + \left(1 - \frac{1}{r}\right) g \exp\left[-\varphi^2_{\max}(n,b)/100\right], \quad (18)$$

where

$$g = \frac{1}{\omega - \omega/r} \sum_{\beta_j \neq 0} \frac{\sin^2(\pi\beta_j)}{(\pi\beta_j)^2}. \quad (19)$$

Defining

$$\bar{f} = \int_{-1/2}^{1/2} \frac{\sin^2(\pi\beta)}{(\pi\beta)^2}\, d\beta \approx 0.774, \quad (20)$$

g can be expressed approximately as

$$g \approx \frac{\bar{f} - 1/r}{1 - 1/r}, \quad (21)$$

and $P(n,b)$, valid for small n, as

$$P(n,b) \approx \tilde{P}(n,b)/\bar{f}. \quad (22)$$

This fully analytical expression for (14) is shown as the solid lines in Fig. 3. We plot $1 - P(n,b)$ to increase our sensitivity in the small-n region. The analytical result captures the small-n behavior very well. Also shown are the crossover points (arrows) that mark the transition from the small-n, nonexponential regime to the large-n, exponential regime (dashed lines). The transition points move toward larger n for larger b so that the region of nonexponential behavior increases for increasing b.

Returning to Fig. 2, we note that the slope of $P(n,b)$ is a sensitive function of b. Indeed, (4) predicts exponential scaling of the slope in b. To test this prediction, we write $P(n,b)$ in

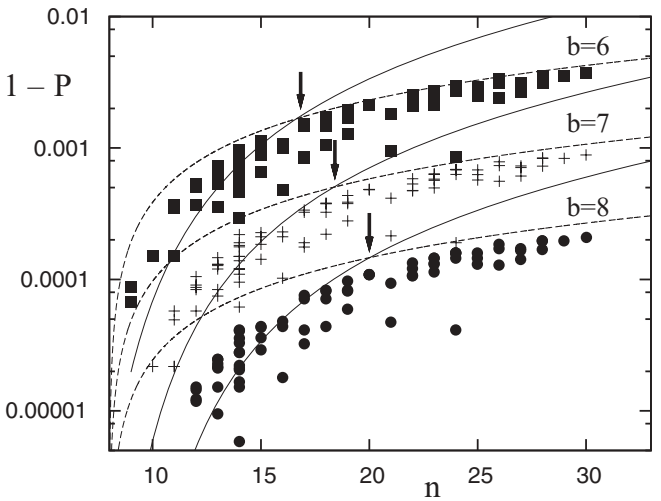

FIG. 3. Small-n behavior of semiprimes N for $b = 6$ (squares), $b = 7$ (crosses), and $b = 8$ (circles). Solid lines are the analytical performance functions $P(n,b)$ in (22), with $f \to \bar{f}$ according to (20). Dashed lines are their large-n, exponential behavior. Crossover points between the small-n, nonexponential and the large-n, exponential behavior are marked by arrows.

the form

$$P(n,b) \approx 2^{-\xi_b(n-n_b)}, \tag{23}$$

where ξ_b and n_b are constants. We graphically extracted the decay constants ξ_b from Fig. 2 and found

$$\xi_b = 1.1 \times 2^{-2b} \tag{24}$$

and $n_b \approx 8$. The quality of the scaling function, (23), is illustrated by the solid lines in Fig. 2, which are plotted using (23) with (24). Comparing (24) with (5), we find the same exponential dependence on b. Our prefactor of 1.1, however, is smaller than the pre-factor predicted by (5), which, according to our results, implies a more optimistic performance scaling than predicted by (5).

As an additional test of performance scaling we plot

$$\eta(n,b) = -\log_2[P(n,b)]/\xi_b \tag{25}$$

versus n for all the data points in Fig. 2, where ξ_b is defined in (24). The result is shown in Fig. 4. The data points in Fig. 4 cluster around the straight line $y = n - 8$ (solid line in Fig. 4) as expected if the scaling in (23) were exact. The near-collapse of the data points in Fig. 4 confirms the approximate performance scaling of Shor's algorithm with a banded quantum Fourier transform, as suggested in (23) and (24).

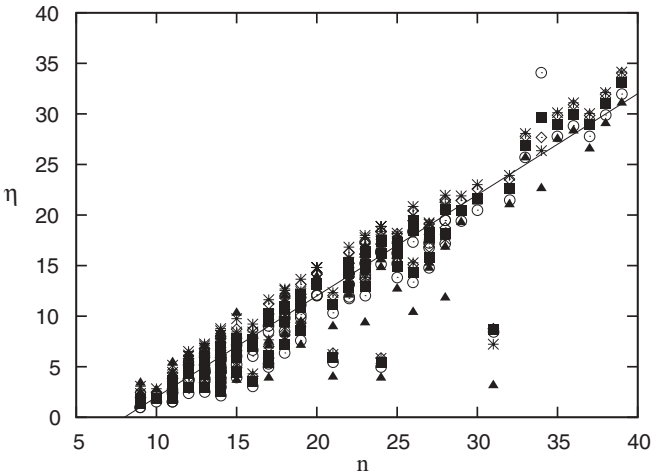

FIG. 4. Data points in Fig. 2 scaled according to (25). The scaled data cluster around $y = n - 8$ (solid line).

Extrapolating our results to $n = 1000$ qubits, $P(n,b)$ reaches a performance level of 95% with a bandwidth of only seven qubits. Thus, without a substantial reduction of performance, of the order of half a million rotation gates can be saved when implementing a 1000-qubit quantum computer running Shor's algorithm.

[1] L. K. Grover, in *Proceedings, 28th Annual ACM Symposium on the Theory of Computing (STOC), Philadelphia* (ACM Press, New York, 1996), pp. 212–219.
[2] L. K. Grover, Am. J. Phys. **69**, 769 (2001).
[3] P. W. Shor, in *Proceedings of the 35th Annual Symposium on the Foundations of Computer Science*, edited by S. Goldwasser (IEEE Press, Los Alamitos, CA, 1994), pp. 124–134.
[4] R. Rivest, A. Shamir, and L. Adleman, Comm. ACM **21**, 120 (1978).
[5] D. Boneh, Notices Am. Math. Soc. **46**, 203 (1999).
[6] S. Robinson, SIAM News **36**(5) (2003).
[7] A. Ekert and R. Jozsa, Rev. Mod. Phys. **68**, 733 (1996).
[8] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000).
[9] N. D. Mermin, *Quantum Computer Science* (Cambridge University Press, Cambridge, 2007).
[10] D. Coppersmith, arXiv:quant-ph/0201067.
[11] R. Blümel, *Foundations of Quantum Mechanics—From Photons to Quantum Computers* (Jones and Bartlett, Sudbury, 2010).
[12] A. G. Fowler and L. C. L. Hollenberg, Phys. Rev. A **70**, 032329 (2004).