Fast parallel circuits for the quantum Fourier transform[*]

Richard Cleve John Watrous
University of Calgary[†]

Abstract

We give new bounds on the circuit complexity of the quantum Fourier transform (QFT). We give an upper bound of $O(\log n + \log\log(1/\varepsilon))$ on the circuit depth for computing an approximation of the QFT with respect to the modulus 2^n with error bounded by ε. Thus, even for exponentially small error, our circuits have depth $O(\log n)$. The best previous depth bound was $O(n)$, even for approximations with constant error. Moreover, our circuits have size $O(n\log(n/\varepsilon))$.

As an application of this depth bound, we show that Shor's factoring algorithm may be based on quantum circuits with depth only $O(\log n)$ and polynomial size, in combination with classical polynomial-time pre- and post-processing.

Next, we prove an $\Omega(\log n)$ lower bound on the depth complexity of approximations of the QFT with constant error. This implies that the above upper bound is asymptotically tight (for a reasonable range of values of ε).

We also give an upper bound of $O(n(\log n)^2 \log\log n)$ on the circuit size of the *exact* QFT modulo 2^n, for which the best previous bound was $O(n^2)$.

Finally, based on our circuits for the QFT with power-of-2 moduli, we show that the QFT with respect to an arbitrary modulus m can be approximated with accuracy ε with circuits of depth $O((\log\log m)(\log\log 1/\varepsilon))$ and size polynomial in $\log m + \log(1/\varepsilon)$.

1. Introduction and summary of results

In this paper we consider the quantum circuit complexity of the *quantum Fourier transform (QFT)*. The QFT is the key quantum operation at the heart of Shor's quantum algorithms for factoring and computing discrete logarithms [34] and the known extensions and variants of these algorithms (see, e.g., Kitaev [24], Boneh and Lipton [7], Grigoriev [19], and Cleve, Ekert, Macchiavello, and Mosca [11]). The QFT also plays a key role in extensions of Grover's quantum searching technique [20], due to Brassard, Høyer, and Tapp [8] and Mosca [28].

[*]Research partially supported by Canada's NSERC.

[†]Dept. of Computer Science, University of Calgary, Calgary, Alberta, Canada T2N 1N4. `{cleve,jwatrous}@cpsc.ucalgary.ca`

Let us recall the *discrete Fourier transform (DFT)*; for a given dimension m the DFT is a linear operator on $\mathbb{C}^m$ mapping $(a_0, a_1, \ldots, a_{m-1})$ to $(b_0, b_1, \ldots, b_{m-1})$, where

$$b_x = \sum_{y=0}^{m-1} (e^{2\pi i/m})^{x\cdot y}\, a_y. \qquad (1)$$

The DFT has many important applications in classical computing, essentially due to the efficiency of the *fast Fourier transform (FFT)*, which is an algorithm that computes the DFT with $O(m\log m)$ arithmetic operations, as opposed to the obvious $O(m^2)$ method. The FFT algorithm was proposed by Cooley and Tukey in 1965 [12], though its origins can be traced back to Gauss in 1866 [16]. The FFT plays an important role in digital signal processing, and it has been suggested [35] as a contender for the second most important nontrivial algorithm in practice, after fast sorting.

The *quantum Fourier transform (QFT)* is a unitary operation that essentially performs the DFT on the amplitude vector of a quantum state—the QFT maps the quantum state $\sum_{x=0}^{m-1} \alpha_x|x\rangle$ to the state $\sum_{x=0}^{m-1} \beta_x|x\rangle$, where

$$\beta_x = \tfrac{1}{\sqrt{m}} \sum_{y=0}^{m-1} (e^{2\pi i/m})^{x\cdot y}\, \alpha_y.$$

The QFT can be approximated by quantum circuits of size polynomial in $\log m$, and for certain m the QFT can be performed exactly with polynomial-size quantum circuits.

The fact that the QFT can be performed by quantum circuit with size polynomial in $\log m$ for some values of m was first observed by Shor [33]. In the case where $m = 2^n$, there exist quantum circuits performing the QFT with $O(n^2)$ gates, which was proved by Coppersmith [13] (see also [10]). These circuits are based on a recursive description of the QFT that is analogous to the description of the DFT exploited by the FFT. While in some sense these quantum circuits are exponentially faster than the classical FFT, the task that they perform is quite different. The QFT does not explicitly produce any of the values $\beta_0, \beta_1, \ldots, \beta_{m-1}$ as output (nor does it explicitly obtain any of the values $\alpha_0, \alpha_1, \ldots, \alpha_{m-1}$ as input). Intuitively, the difference between performing a DFT and a QFT can be thought of as being analogous to the difference between computing all the probabilities that comprise a probability distribution and sampling a probability distribution—the latter task being frequently much easier.

Coppersmith [13] also proposed quantum circuits that approximate the QFT with error bounded by ε, and showed that such approximations can be computed by circuits of size $O(n \log(n/\varepsilon))$ for modulus 2^n. Kitaev [24] showed how the QFT for an arbitrary modulus m can be approximated by circuits with size polynomial in $\log(m/\varepsilon)$. For most information processing purposes, it suffices to use such approximations of quantum operations (for ε ranging from constant down to $1/n^{O(1)}$). Indeed, since it seems rather implausible to physically implement quantum gates with perfect accuracy, the need to ultimately consider approximations is likely inevitable. Thus, we believe the most relevant consideration is to approximately compute the QFT, though exact computations of the QFT are still of interest as part of the theory of quantum computation.

Moore and Nilsson [27] showed that encoding and decoding for standard quantum error-correcting codes could be done by logarithmic-depth quantum circuits, and noted that Coppersmith's circuits for the QFT can be arranged so as to have depth $2n - 1$ (but apparently not less than this). Similarly, the techniques of Shor and of Kitaev for the QFT have polynomial depth. Our first result shows that it is possible to compute good approximations of the QFT with logarithmic-depth quantum circuits.

Theorem 1 *For any n and ε there is a quantum circuit approximating the QFT modulo 2^n with precision ε that has depth $O(\log n + \log\log(1/\varepsilon))$ and size $O(n \log(n/\varepsilon))$.*

By an approximation of a unitary operation U with *precision* ε, we mean a unitary operation V (possibly acting on additional ancilla qubits) with the following property. For any input (pure) quantum state, the Euclidean distance between applying U to the state and V to the state is at most ε (in the Hilbert space that includes the input/output qubits and the ancilla qubits). Also, whenever we refer to *circuits*, there is an implicit technical assumption that the circuits belong to a logarithmic-space uniformly generated family via a *classical* Turing machine. This is a straightforward extension of uniformity definitions for classical circuits (which are discussed in [15, 23]).

In Section 8, we consider an approach for parallelizing Shor's QFT method, which may be described as a mixed-radix method, that gives somewhat worse bounds.

The proof of Theorem 1 follows the general approach introduced by Kitaev [24], with several efficiency improvements and parallelizations. In particular, we introduce a new method for parallel multiprecision phase estimation.

An immediate benefit of the QFT circuits from Theorem 1 regards fault-tolerant implementations of the QFT. Using the most efficient techniques known for fault-tolerant implementation of quantum circuits (see [1, 25, 30]), our circuits for the QFT can be implemented with a size increase of only a poly-logarithmic factor, to $O(n(\log n)^c)$. In contrast, these techniques result in at least a linear increase in size for any linear-depth approximate QFT—for instance, for the approximate QFT circuits in [13], the resulting size is $O(n^2(\log n)^c)$.

It has long been known that the bottleneck of the quantum portion of Shor's factoring algorithm is not the QFT, but rather is the modular exponentiation step. If it were possible to perform modular exponentiation by classical (or quantum) circuits with poly-logarithmic depth and polynomial size then it would be possible to implement Shor's factoring algorithm in poly-logarithmic depth with a polynomial number of qubits. Although no such algorithm is known for modular exponentiation, we can prove the following weaker result, which nevertheless implies that quantum computers need only run for logarithmic time for factoring to be feasible.

Theorem 2 *There is an algorithm for factoring n-bit integers that consists of: a classical pre-processing stage, computed by a polynomial-size classical circuit; followed by a quantum information processing stage, computed by an $O(\log n)$-depth polynomial-size quantum circuit; followed by a classical post-processing stage, computed by a polynomial-size classical circuit.*

It is interesting to note that this theorem implies that logarithmic-depth quantum circuits cannot be simulated in polynomial time unless factoring is in BPP.

We also consider the minimum depth *required* for approximating the QFT. It is fairly easy to show that computing the QFT *exactly* requires depth at least $\log n$. However, this is less clear in the case of approximations—and we exhibit in Theorem 6 a problem related to the QFT whose depth complexity decreases from $\log n$ in the exact case to $O(\log\log n)$ for approximations with precision $1/n^{O(1)}$. Nevertheless, we show the following.

Theorem 3 *Any quantum circuit consisting of one- and two-qubit gates that approximates the QFT with precision $\frac{1}{10}$ or smaller must have depth at least $\log n$.*

This implies that the depth upper bound in Theorem 1 is asymptotically tight for a reasonable range of values of ε.

We also show that, if size rather than depth is the primary consideration, it is possible to compute the QFT *exactly* with a near-linear number of gates.

Theorem 4 *For every n there exists a quantum circuit that exactly computes the QFT modulo 2^n that has size $O(n(\log n)^2 \log\log n)$ and depth $O(n)$.*

Theorem 4 is based on a nonstandard recursive description of the QFT [10] combined with an asymptotically fast multiplication algorithm [32].

While it is sufficient to use the QFT with respect to power-of-2 moduli in Shor's algorithm, one may wish to perform the QFT with respect to other moduli when considering other problems. By exploiting a relationship among QFTs of different moduli noted by Hales and Halgren [21], we prove that the QFT for an arbitrary modulus can be performed in poly-logarithmic depth.

Theorem 5 *For any m and ε there is a quantum circuit approximating the QFT modulo m with precision ε that has depth $O((\log\log m)(\log\log(1/\varepsilon)))$ and size polynomial in $\log m + \log(1/\varepsilon)$.*

The remainder of this paper is organized as follows. In Section 2, we review some definitions and introduce notation used in subsequent sections. In Section 3 we prove the depth and size bounds for quantum circuits approximating the QFT for any power-of-2 modulus as claimed in Theorem 1. In Section 4 we prove Theorem 2 by demonstrating how Shor's factoring algorithm can be arranged so as to require only logarithmic-depth quantum circuits. In Section 5 we prove the lower bound for the QFT in Theorem 3. In Section 6 we prove the size bound claimed in Theorem 4 for exactly performing the QFT. In Section 7 we discuss the situation when the modulus for the QFT is not necessarily a power of 2, and in Section 8 we discuss the special case of "smooth" moduli considered in Shor's original method for performing the QFT. We conclude with Section 9, which mentions some open questions relating to this paper.

2. Definitions and notation

Notation for special quantum states: For a given modulus m, we will identify each $x \in \mathbb{Z}_m$ with its binary representation $x_{n-1}\ldots x_1x_0 \in \{0,1\}^n$, for $n = \lceil \log m \rceil$. For $x \in \mathbb{Z}_m$, the state $|x\rangle = |x_{n-1}\ldots x_1x_0\rangle$ is a *computational basis state*, and the state $|\psi_x\rangle = \frac{1}{\sqrt{m}}\sum_{y=0}^{m-1}(e^{2\pi i/m})^{x\cdot y}|y\rangle$ is a *Fourier basis state* (with *phase parameter* x). As noted in [11], when $m = 2^n$, $|\psi_x\rangle$ can be factored as follows

$$\begin{aligned}&|\psi_{x_{n-1}\ldots x_1x_0}\rangle\\ &\quad= \frac{1}{\sqrt{2^n}}(|0\rangle + e^{2\pi i(0.x_0)}|1\rangle)(|0\rangle + e^{2\pi i(0.x_1x_0)}|1\rangle)\cdots\\ &\qquad\cdots(|0\rangle + e^{2\pi i(0.x_{n-1}\ldots x_1x_0)}|1\rangle). \qquad (2)\end{aligned}$$

For convenience, we define $|\mu_\theta\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i\theta}|1\rangle)$ for each $\theta \in \mathbb{R}$. Using this notation, we can rewrite Eq. 2 as

$$|\psi_{x_{n-1}\ldots x_0}\rangle = |\mu_{0.x_0}\rangle|\mu_{0.x_1x_0}\rangle\cdots|\mu_{0.x_{n-1}\ldots x_1x_0}\rangle. \qquad (3)$$

Definition of the QFT: The *quantum Fourier transform (QFT)* (with modulus m) is the unitary operation that maps $|x\rangle$ to $|\psi_x\rangle$ (for all $x \in \mathbb{Z}_m$).

Mappings related to the QFT: A *quantum Fourier state computation (QFS)* is any unitary operation that maps $|x\rangle|0\rangle$ to $|x\rangle|\psi_x\rangle$ (for all $x \in \mathbb{Z}_m$). We refer to approximations of a QFS as *Fourier state estimation*. A *quantum Fourier phase computation (QFP)* is any unitary operation that maps $|\psi_x\rangle|0\rangle$ to $|\psi_x\rangle|x\rangle$ (for all $x \in \mathbb{Z}_m$). We refer to approximations of a QFP as *Fourier phase estimation*. As pointed out by Kitaev [24], the QFT can be computed by composing a QFS and the inverse of a QFP as follows: $|x\rangle|0\rangle \mapsto |x\rangle|\psi_x\rangle \mapsto |0\rangle|\psi_x\rangle$.

Quantum gates: All of the quantum circuits that we construct will be composed of three types of unitary gates. One is the one-qubit *Hadamard* gate, H, which maps $|x\rangle$ to $\frac{1}{\sqrt{2}}(|0\rangle + (-1)^x|1\rangle)$ (for $x \in \{0,1\}$). Another is the one-qubit *phase shift* gate, $P(\theta)$, where θ is a parameter of the form $x/2^n$ (for $x \in \mathbb{Z}_{2^n}$). $P(\theta)$ maps $|x\rangle$ to $e^{2\pi i\theta x}|x\rangle$ (for $x \in \{0,1\}$). Finally, we use the two-qubit *controlled*-phase shift gate, c-$P(\theta)$, which maps $|x\rangle|y\rangle$ to $e^{2\pi i\theta xy}|x\rangle|y\rangle$ (for $x, y \in \{0,1\}$). Note that controlled-NOT and Toffoli gates can be composed of these gates.

3. New depth bounds for the QFT

The main purpose of this section is to prove Theorem 1.

First, we review the approach of Kitaev [24] for performing the QFT for an arbitrary modulus m. By linearity, it is sufficient to give a circuit that operates correctly on computational basis states. Given a computational basis state $|x\rangle$, first create the Fourier basis state with phase parameter x (which can be done easily if $|x\rangle$ is not erased in the process). The system is now in the state $|x\rangle|\psi_x\rangle$. Now, by performing Fourier phase estimation, the state $|x\rangle|\psi_x\rangle$ can be approximated from the state $|0\rangle|\psi_x\rangle$. Therefore, by performing the inverse of Fourier phase estimation on the state $|x\rangle|\psi_x\rangle$, a good estimate of the state $|0\rangle|\psi_x\rangle$ is obtained.

The particular phase estimation procedure used by Kitaev does not readily parallelize, but, in the case where the modulus is a power of 2, we give a new phase estimation procedure that does parallelize. This procedure requires several copies of the Fourier basis state rather than just one. To insure that the entire process parallelizes, we must also parallelize the creation of the Fourier basis state as well as the process of copying and uncopying this state.

The basic steps of our technique are as follows:

1. Create the Fourier basis state, which is the QFS mapping $|x\rangle|0\rangle \mapsto |x\rangle|\psi_x\rangle$.

2. Copy the Fourier basis state, which is the mapping $|\psi_x\rangle|0\rangle\cdots|0\rangle \mapsto |\psi_x\rangle|\psi_x\rangle\cdots|\psi_x\rangle$.

3. Erase the computational basis state by estimating the phase of the Fourier basis state, which is the mapping $|x\rangle|\psi_x\rangle|\psi_x\rangle\cdots|\psi_x\rangle \mapsto |0\rangle|\psi_x\rangle|\psi_x\rangle\cdots|\psi_x\rangle$.

4. Reverse step 2, which corresponds to the mapping $|\psi_x\rangle|\psi_x\rangle\cdots|\psi_x\rangle \mapsto |\psi_x\rangle|0\rangle\cdots|0\rangle$.

Each of these components is discussed in detail in the subsections that follow. Throughout we assume the modulus is $m = 2^n$.

3.1. Fourier state computation and estimation

The first step is the creation of the Fourier basis state corresponding to a given computational basis state $|x\rangle$. This corresponds to the QFS mapping

$$|x\rangle|0\rangle \mapsto |x\rangle|\psi_x\rangle. \tag{4}$$

First let us consider a circuit that performs this transformation exactly. By Eq. 2 (equivalently, Eq. 3), it suffices to compute the states $|\mu_{0.x_0}\rangle, |\mu_{0.x_1x_0}\rangle, \ldots, |\mu_{0.x_{n-1}\ldots x_1x_0}\rangle$ individually.

The circuit suggested by Figure 1 performs the required transformation for $|\mu_{0.x_j\ldots x_0}\rangle$.

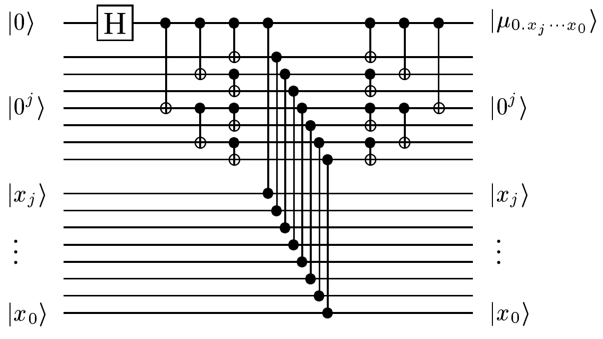

Figure 1. Quantum circuit for preparation of $|\mu_{0.x_j\cdots x_0}\rangle$.

In this figure we have not labelled the controlled phase shift gates, c-$P(\theta)$ (such gates are defined in Section 2), which are the gates in the center drawn as two solid circles connected by a line. For each such gate, the phase θ depends on j and on the particular qubit of $|x_{n-1}\ldots x_1x_0\rangle$ on which the gate acts. The value of θ for the controlled phase shift acting on $|x_i\rangle$ is 2^{i-j-1} (for $i \in \{0,1,\ldots,j\}$). The remaining gates consist of a Hadamard gate and several controlled-NOT gates. The depth of this circuit is $O(\log n)$ and the size is $O(n)$.

If such a circuit is to be applied for $j = 0,\ldots,n-1$, in order to perform the mapping in Eq. 4, then the qubits $|x_{n-1}\rangle,\ldots,|x_1\rangle,|x_0\rangle$ must first be copied several times ($n-i$ times for $|x_i\rangle$) to allow the controlled phase shift gates to operate in parallel. This may be performed (and inverted appropriately) in size $O(n^2)$ and depth $O(\log n)$ in the most obvious way. Thus we have that a QFS transform can be performed exactly by circuits of depth $O(\log n)$ and size $O(n^2)$.

For our purposes, we need to reduce the size of the QFS circuit, which is possible if we only require an approximate QFS circuit with error bounded by ε. This is achieved by constructing a circuit similar to the exact QFS circuit, except that controlled phase shift gates are not included whenever the phase θ is $O(\varepsilon/n^2)$. Thus, for $j = 0,1,\ldots,n-1$, only the controlled phase shifts corresponding to the wires $|x_j\rangle,\ldots,|x_{j-k+1}\rangle$ are performed, where $k \in 2\log(n/\varepsilon) + O(1)$. This results in the following depth and size bound.

Theorem 6 *For any n and ε there is a quantum circuit that approximates a QFS mapping modulo 2^n with precision ε that has depth $O(\log\log(n/\varepsilon))$ and size $O(n\log(n/\varepsilon))$.*

3.2. Copying a Fourier state

In this section, we show how to efficiently produce several copies of an n-qubit Fourier basis state from one copy. This is a unitary operation that acts on k n-qubit registers (thus kn qubits in all) and maps $|\psi_x\rangle|0^n\rangle\cdots|0^n\rangle$ to $|\psi_x\rangle|\psi_x\rangle\cdots|\psi_x\rangle$ for all $x \in \{0,1\}^n$. The copying circuit will be exact and have size $O(kn)$ and depth $O(\log(kn))$. The setting of k will be $O(\log(n/\varepsilon))$.

Consider the problem of producing two copies of a Fourier state from one. Define the *(reversible) addition* and *(reversible) subtraction* operations as the mappings $|x\rangle|y\rangle \mapsto |x\rangle|y+x\rangle$ and $|x\rangle|y\rangle \mapsto |x\rangle|y-x\rangle$ (respectively), where $x,y \in \{0,1\}^n$ and additions and subtractions are performed as integers modulo 2^n. By appealing to classical results about the complexity of arithmetic (a good exposition can be found in Chapter 29 of [14]), one can construct quantum circuits of size $O(n)$ and depth $O(\log n)$ for these operations (using an ancilla of size $O(n)$). It is straightforward to show that applying a subtraction operation to the state $|\psi_x\rangle|\psi_y\rangle$ results in the state $|\psi_{x+y}\rangle|\psi_y\rangle$. The state $|\psi_0\rangle$ can be obtained from $|0^n\rangle$ by applying a Hadamard transform independently to each qubit. Therefore, the copying operation can begin with a state of the form $|0^n\rangle|\psi_x\rangle$ and consist of these two steps: (i) Apply H to each of the first n qubits, and (ii) apply the subtraction operation to the $2n$ qubits. The resulting state will be $|\psi_x\rangle|\psi_x\rangle$.

An obvious method for computing k copies of a Fourier state is to repeatedly apply the above doubling operation. This will result in a quantum circuit of size $O(kn)$; however, its depth will be $O((\log k)(\log n))$, which is too large for our purposes.

The depth bound can be improved to $O(\log(kn))$ by applying other classical circuit constructions to efficiently implement the *(reversible) prefix addition* and *(reversible) telescoping subtraction* operations, which are the mappings

$$|x_1\rangle|x_2\rangle\cdots|x_k\rangle \quad\mapsto\quad |x_1\rangle|x_1+x_2\rangle\cdots|x_1+\cdots+x_k\rangle$$

$$|x_1\rangle|x_2\rangle\cdots|x_k\rangle \quad\mapsto\quad |x_1\rangle|x_2-x_1\rangle\cdots|x_k-x_{k-1}\rangle$$

(respectively), where $x_1, x_2, \ldots, x_k \in \{0,1\}^n$. Before addressing the issue of efficiently implementing these operations, let us note that the copying operation can be performed by starting with the state $|0^n\rangle \cdots |0^n\rangle|\psi_x\rangle$ and performing these two steps: (i) Apply H to all of the first $(k-1)n$ qubits, and (ii) apply the telescoping subtraction operation to the kn qubits. The resulting state will be $|\psi_x\rangle \cdots |\psi_x\rangle$.

Now, to implement the prefix addition and telescoping subtraction, note that they are inverses of each other. This means that it is sufficient to implement each one efficiently by a classical (nonreversible) circuit, and then combine these to produce a reversible circuit by standard techniques in reversible computing [5]. The telescoping subtraction clearly consists of $k-1$ subtractions that can be performed in parallel, so the nonreversible size and depth bounds are $O(kn)$ and $O(\log n)$ respectively.

The prefix addition is a little more complicated. It relies on a combination of well-known tools in classical circuit design. One of them is the following general result of Ladner and Fischer [26] about parallel prefix computations.

Theorem 7 (Ladner and Fischer) *For any associative binary operation $\circ$, the mapping*

$$(x_1,\ x_2,\ \ldots,\ x_k) \mapsto (x_1,\ x_1 \circ x_2,\ \ldots,\ x_1 \circ \cdots \circ x_k)$$

can be computed by a size $O(k)$ and depth $O(\log k)$ circuit consisting of gates of the form $(x,\ y) \mapsto (x,\ x \circ y)$.

Another tool is the so-called *carry-save adder*, which is a circuit that takes three n-bit integers x, y, z as input and produces two n-bit integers s, c as output, such that $x+y+z = s+c$ (recall that addition is in modulo 2^n arithmetic). It is remarkable that a carry-save adder can be implemented with *constant depth* and size $O(n)$ (see [14]). By combining two carry-save adders, one can implement a size $O(n)$ and depth $O(1)$ *four-two adder*, that performs the mapping $(x,y,z,w) \mapsto (x,y,s,c)$, where $x+y+z+w = s+c$. Now, consider the *pairwise representation* of each n-bit integer z as a pair of two n-bit integers (z', z'') such that $z = z' + z''$. This representation is not unique, but it is easy to convert to and from the pairwise representation: the respective mappings are $z \mapsto (z, 0^n)$ and $(z', z'') \mapsto z' + z''$. The useful observation is that the four-two adder performs integer addition in the pairwise representation scheme, and it does so in constant depth and size $O(n)$.

Now, the following procedure computes prefix addition in size $O(kn)$ and depth $O(\log k + \log n) = O(\log(kn))$. The input is $(x_1,\ x_2,\ \ldots,\ x_k)$. (i) Convert the k integers into their pairwise representation. (ii) Apply the parallel prefix circuit of Theorem 7 to perform the prefix additions in the pairwise representation scheme. (iii) Convert the k integers from their pairwise representation to their standard form. The output will be $(x_1,\ x_1 + x_2, \ldots, x_1 + \cdots + x_k)$, as required.

Note that step 4 of the main algorithm has a circuit of identical size and depth to the one just described, as it is simply its inverse.

3.3. Estimating the phase of a Fourier state

Finally, we explain the third step of the main algorithm, which corresponds to the mapping

$$|\psi_x\rangle|\psi_x\rangle \cdots |\psi_x\rangle|x\rangle \mapsto |\psi_x\rangle|\psi_x\rangle \cdots |\psi_x\rangle|0\rangle \qquad (5)$$

for $x \in \{0,1\}^n$. The number of copies of $|\psi_x\rangle$ required for this step depends on the error bound ε; we will require $k \in O(\log(n/\varepsilon))$ copies.

First note that, by Eq. 3, k copies of $|\psi_x\rangle$ comprise k copies of $|\mu_{0.x_j \ldots x_0}\rangle$ for each $j \in \{0, 1, \ldots, n-1\}$. Our approach can be explained intuitively by considering the problem of how to measure these kn qubits in order to obtain a good estimator of the string $x_{n-1} \ldots x_1 x_0$. A solution to this can then be translated via standard techniques into a quantum circuit that computes the inverse of the mapping in Eq. 5.

It should be noted that, in general, x_j *cannot* be estimated well by measuring the k copies of $|\mu_{0.x_j \ldots x_0}\rangle$ alone. For example, when $x_{n-1} \ldots x_1 x_0 = 100 \ldots 0$, measuring $|\mu_{0.100\ldots0}\rangle$ will yield little information about x_{n-1}, since $|\mu_{0.100\ldots0}\rangle$ is exponentially close to $|\mu_{0.011\ldots1}\rangle$ (even though $|100 \ldots 0\rangle$ is orthogonal to $|011 \ldots 1\rangle$).

The approach that we follow is to perform measurements on the k copies of $|\mu_{0.x_j \ldots x_0}\rangle$ in order to obtain information about x_j *as a function of* x_{j-1}. After this is performed in parallel for each $j > 0$, and x_0 is determined (easily by measuring $|\mu_{0.x_0}\rangle$), the information can be combined to determine $x_{n-1} \ldots x_1 x_0$.

Two types of measurements are used here. For half of the k copies of $|\mu_{0.x_j \ldots x_0}\rangle$, a so-called σ_z *measurement* is performed. This is a measurement with respect to the basis $\{\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle, \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\} = \{|\mu_{0.0}\rangle, |\mu_{0.1}\rangle\}$. For the remaining copies of $|\mu_{0.x_j \ldots x_0}\rangle$, a σ_y *measurement* is performed, which is a measurement with respect to the basis $\{\frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle, \frac{1}{\sqrt{2}}|0\rangle - \frac{i}{\sqrt{2}}|1\rangle\} = \{|\mu_{0.01}\rangle, |\mu_{0.11}\rangle\}$.

Let us now examine what these measurements reveal when they are applied to a state of the form $|\mu_{0.x_j \ldots x_0}\rangle$. It is helpful to think of the number $0.x_j \ldots x_0 \in [0,1)$ as a point on a circle with unit circumference (with 0 identified with 1). A σ_z measurement of $|\mu_{0.x_j \ldots x_0}\rangle$ has outcome probabilities $\frac{1}{2} + \frac{1}{2}\cos(2\pi(0.x_j \ldots x_0))$ (for $|\mu_{0.0}\rangle$), and $\frac{1}{2} - \frac{1}{2}\cos(2\pi(0.x_j \ldots x_0))$ (for $|\mu_{0.1}\rangle$). If $0.11 < 0.x_j \ldots x_0 < 1$ or $0 \leq 0.x_j \ldots x_0 < 0.01$, then $x_j = x_{j-1}$ and the outcome is biased towards $|\mu_{0.0}\rangle$. If $0.01 < 0.x_j \ldots x_0 < 0.11$ then $x_j \neq x_{j-1}$ and the outcome is biased towards $|\mu_{0.1}\rangle$. At the boundary points, 0.01

and 0.11, the distribution is unbiased. Thus, if a statistically significant bias among the $k/2$ outcomes of the σ_z measurements occurs then x_j can be inferred in terms of x_{j-1}. Let **P** (*propagate*) denote the condition $x_j = x_{j-1}$ and let **N** (*negate*) denote the condition $x_j \neq x_{j-1}$. When $0.x_j \ldots x_0$ is at or near one of the two boundary points, the two outcomes of a σ_z measurement have nearly the same probability and are hard to distinguish. In these cases, the outcomes of the σ_y measurements are needed.

For a σ_y measurement the outcome probabilities are $\frac{1}{2} \pm \frac{1}{2}\sin(2\pi(0.x_j \ldots x_0))$ and the properties are similar to those of a σ_z measurement, except that the two ranges that are distinguished are: $0 < 0.x_j \ldots x_0 < 0.1$, which corresponds to $x_j = 0$; and, $0.1 < 0.x_j \ldots x_0 < 1$, which corresponds to $x_j = 1$. A statistically significant bias among the $k/2$ outcomes of the σ_y measurements enables an inference $x_j = 0$ or $x_j = 1$ to be made. Let **0** and **1** denote these two conditions.

For the $|\mu_{0.x_j \ldots x_0}\rangle$ states, the most frequent of the k outcomes among the σ_z and σ_y measurements is taken (resolving ties arbitrarily). The result will be the condition **P**, **N**, **0**, or **1**. The probability that the condition is an incorrect one is exponentially small with respect to k. To see why this is so, recall the following result about independent Bernoulli trials (see, e.g., [17]). Let $a_1, \ldots, a_t$ be independent Bernoulli trials with probability p of success and $b_1, \ldots, b_t$ be independent Bernoulli trials with probability q of success, where $p > q$. Then

$$\Pr\left[\sum_{i=1}^{t} a_i \leq \sum_{i=1}^{t} b_i\right] < 2e^{-(p-q)^2\, t/2}.$$

When this is applied in our context, the value of $p - q$ is $\frac{1}{2}|\cos(2\pi(0.x_j \ldots x_0))| - \frac{1}{2}|\sin(2\pi(0.x_j \ldots x_0))| \geq \frac{1}{2}$ and $t = k/2$, so the probability of an error is less than $2e^{-k/16}$ for each $j \in \{0, 1, \ldots, n-1\}$. The circuit computing the n conditions has depth $O(\log k) = O(\log\log(n/\varepsilon))$ and size $O(nk) = O(n\log(n/\varepsilon))$.

Now, assume that $L_{n-1}, \ldots, L_1, L_0 \in \{\mathbf{P}, \mathbf{N}, \mathbf{0}, \mathbf{1}\}$ are correct conditions for $x_{n-1}, \ldots, x_1, x_0$ (respectively), and that L_0 is **0** or **1**. From these values, the bits of $x_{n-1} \ldots x_1 x_0$ can be deduced by tracing through the conditions. For example, the string of conditions **1P1NNPP0** corresponds to the binary string 11101000. The simplest way to trace through the conditions is to start at the right side and proceed left, one condition at a time—this is a sequential process that takes n steps.

However, these deductions can also be computed in parallel. A method for doing this can be seen by identifying each condition with a 2×2 matrix as follows

$$\mathbf{P} = \left(\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right),\ \ \mathbf{N} = \left(\begin{smallmatrix}0 & 1\\ 1 & 0\end{smallmatrix}\right),\ \ \mathbf{0} = \left(\begin{smallmatrix}1 & 1\\ 0 & 0\end{smallmatrix}\right),\ \ \mathbf{1} = \left(\begin{smallmatrix}0 & 0\\ 1 & 1\end{smallmatrix}\right).$$

Then, tracing through the conditions is equivalent to computing products of the matrices. To obtain all the bits of $x_{n-1} \ldots x_1 x_0$, all suffix products of the string of conditions must be computed, which can be performed in parallel by the parallel prefix method of Theorem 7. The depth of the circuit for this is $O(\log n)$ and the size is $O(n)$.

It follows from the above that there is circuit of depth $O(\log n + \log\log(n/\varepsilon)) = O(\log n + \log\log(1/\varepsilon))$ and size $O(n\log(n/\varepsilon))$ that estimates the string $x_{n-1} \ldots x_1 x_0$. What remains is to show how to convert this into a quantum circuit without measurements that approximates the mapping in Eq. 5. This follows from standard results based on ideas in [6] about converting quantum circuits that perform measurements and produce classical information with small error probability into unitary operations (without measurements) that can operate on data in superposition. It should be noted that a state $|\psi_x\rangle$ can be conserved throughout the computation to ensure that errors corresponding to different values of x are orthogonal.

4. Factoring via logarithmic-depth quantum circuits

In this section we discuss a simple modification of Shor's factoring algorithm that factors integers in polynomial time using logarithmic-depth quantum circuits. It is important to note that we are not claiming the existence of logarithmic-depth quantum circuits that take as input some integer N and output a non-trivial factor of N with high probability—the method will require (polynomial time) classical pre-processing and post-processing that is not known to be parallelizable. The motivation for this approach is that, under the assumption that quantum computers can be built, one may reasonably expect that quantum computation will be expensive while classical computation will be inexpensive.

The main bottleneck of the quantum portion of Shor's factoring algorithm is the modular exponentiation. Whether or not modular exponentiation can be parallelized is a long-standing open question that is not resolved here. Instead, we show that sufficient classical pre-processing allows parallelization of the part of the quantum circuit associated with the modular exponentiation. Combined with our logarithmic-depth circuits for the QFT, we obtain the result claimed in Theorem 2.

In order to describe our method, let us briefly review Shor's factoring algorithm, including the reduction from factoring to order-finding. It is assumed the input is a n-bit integer N that is odd and composite.

1. (Classical) Randomly select $a \in \{2, \ldots, N-1\}$. If $\gcd(a, N) > 1$ then output $\gcd(a, N)$, otherwise continue to step 2.
2. (Quantum) Attempt to find information about the order of a in $\mathbb{Z}_N^*$:
   a. Initialize a $2n$-qubit register and an n-qubit register to state $|0\rangle|0\rangle$.

b. Perform a Hadamard transform on each qubit of the first register.

c. (Modular exponentiation step.) Perform the unitary mapping $|x\rangle|0\rangle \mapsto |x\rangle|a^x \bmod N\rangle$.

d. Perform the quantum Fourier transform on the first register and measure (in the computational basis). Let y denote the result.

3. (Classical) Use the continued fraction algorithm to find relatively prime integers k and r, $0 \le k < r < N$, such that $|y/2^{2n} - k/r| \le 2^{-2n}$. If $a^r \equiv 1 \pmod N$ then continue to step 4, otherwise repeat step 2.

4. (Classical) If r is even, compute $d = \gcd(a^{r/2} - 1, N)$ and output d if it is a nontrivial factor of N. Otherwise go to step 1.

The key idea is to use the fact (noted in [34]) that much of the work required for the modular exponentiation step can be shifted to the classical computation in step 1 of the procedure. In step 1, the numbers

$$b_0 = a,\ b_1 = a^2,\ \ldots,\ b_{2n-1} = a^{2^{2n-1}} (\bmod N)$$

can be computed in polynomial-time. With this information available in step 2, the modular exponentiation step reduces to applying a unitary operation mapping $|x\rangle|0\rangle$ to $|x\rangle|b_0^{x_0} \cdot b_1^{x_1} \cdots b_{2n-1}^{x_{2n-1}} \bmod N\rangle$. This is essentially an iterated multiplication problem, where one is given $2n$ n-bit integers $b_0^{x_0}, b_1^{x_1}, \ldots, b_{2n-1}^{x_{2n-1}}$ as input and the goal is to compute their product. The most straightforward way to do this is to perform pairwise multiplications following the structure of a binary tree with $2n$ leaves. Each multiplication can be performed with depth $O(\log n)$ and size $O(n^2)$. The underlying binary tree has depth $\log(2n)$ and $2n-1$ internal nodes. Thus, the entire process can be performed with depth $O((\log n)^2)$ and size $O(n^3)$.

There are alternative methods for performing iterated multiplication achieving various combinations of depth and size. In particular, it was recently proved by Chiu, Davida, and Litow [9] (improving on results in [4]) that a product such as we have above can be computed by $O(\log n)$ depth boolean circuits of polynomial size. While the size is likely to exceed the $O(n^3)$ bound obtained above, the result has an interesting consequence regarding simulations of logarithmic-depth quantum circuits: if logarithmic-depth quantum circuits can be simulated in polynomial time, then factoring can be done in polynomial time as well.

5. Lower bounds

Logarithmic-depth lower bounds for *exact* computations with two-qubit gates are fairly easy to obtain, based on the fact that the state of some output qubit (usually) critically depends on every input qubit. Since, by Eq. 3, the last qubit of $|\psi_{x_{n-1}\ldots x_1 x_0}\rangle$ is in state $\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_{n-1}\ldots x_1 x_0)}|1\rangle)$, its value depends on all n qubits of the corresponding input state $|x_{n-1}\ldots x_1 x_0\rangle$. The depth of the circuit must be at least $\log n$ for this to be possible. This lower bound proof applies not only to the QFT, but also to QFS computations (defined in Section 2). This is because the output of a QFS on input $|x\rangle|0\rangle$ includes the state $|\psi_x\rangle$.

On the other hand, *approximate* computations can sometimes be performed with much lower depth than their exact counterparts. For example, we have already seen in Theorem 6 (Section 3.1) that a QFS can be computed with precision ε by a quantum circuit with depth $O(\log\log(n/\varepsilon))$. Thus, for $\varepsilon \in 1/n^{O(1)}$, the QFS circuit depth need only be $O(\log\log n)$. Although this suggests that it is conceivable for a sub-logarithmic-depth circuit to approximate the QFT with precision $1/n^{O(1)}$, Theorem 3 implies that this is not possible. We now prove this theorem.

Let C be a quantum circuit that approximates the *inverse* QFT with precision $\frac{1}{10}$. In this section, since we will need to consider distances between mixed states, we adopt the *trace distance* as a measure of distance (see Chapter 9 of [29] for an excellent discussion of distance measures between quantum states, and for further references). The trace distance between two states with respective density operators ρ and σ is given as $D(\rho,\sigma) = \frac{1}{2}\mathrm{Tr}|\rho - \sigma|$, where, for an operator A, $|A| = \sqrt{A^\dagger A}$. For a pair of pure states $|\phi\rangle$ and $|\phi'\rangle$, their trace distance is $\sqrt{1 - |\langle\phi|\phi'\rangle|^2}$, which is upper bounded by their Euclidean distance.

On input $|\psi_{x_{n-1}\ldots x_1 x_0}\rangle$, the output state of C contains an approximation of $|x_{n-1}\ldots x_1 x_0\rangle$. In particular, one of the output qubits of C should be in a state that is an approximation of $|x_{n-1}\rangle$ within $\frac{1}{10}$. Let us refer to this as the *high-order* output qubit of C. If the depth of C is less than $\log n$ then the high-order output qubit of C cannot depend on all n of its input qubits. Let $k \in \{0,1,\ldots,n-1\}$ be such that the high-order output qubit does not depend on the k^{th} input qubit (where we index the input qubits right to left starting from 0). Let $r = n - k - 1$.

Set $z = 2^n - 1$, which is $11\ldots1 = 1^n$ in binary. Following Eq. 3, we have $|\psi_z\rangle = |\mu_{0.1}\rangle|\mu_{0.11}\rangle\cdots|\mu_{0.1^n}\rangle$. Consider the state $|\psi_{z+2^r}\rangle$. As $z + 2^r = 0^{n-r}1^r \pmod{2^n}$, we see that

$$|\psi_{z+2^r}\rangle = |\mu_{0.1}\rangle\cdots|\mu_{0.1^r}\rangle|\mu_{0.01^r}\rangle|\mu_{0.001^r}\rangle\cdots|\mu_{0.0^{n-r}1^r}\rangle.$$

Note that, on input $|\psi_z\rangle$, the high-order output qubit of C approximates $|1\rangle$ with precision $\frac{1}{10}$; whereas, on input $|\psi_{z+2^r}\rangle$, the high-order output qubit of C approximates $|0\rangle$ with precision $\frac{1}{10}$.

Now, we consider a state $|\psi'_z\rangle$, which has an interesting relationship with both $|\psi_z\rangle$ and $|\psi_{z+2^r}\rangle$. Define

$$|\psi'_z\rangle = |\mu_{0.1}\rangle\cdots|\mu_{0.1^r}\rangle|\mu_{0.01^r}\rangle|\mu_{0.1^{r+2}}\rangle\cdots|\mu_{0.1^n}\rangle.$$

The states $|\psi'_z\rangle$ and $|\psi_z\rangle$ are identical, except in their k^{th} qubit positions (which are orthogonal: $|\mu_{0.01^r}\rangle$ vs. $|\mu_{0.1^{r+1}}\rangle$). Since the high-order output qubit of C does not depend on its k^{th} input qubit, it is the same for input $|\psi'_z\rangle$ as for input $|\psi_z\rangle$. Therefore, the state of the high-order output qubit of C on input $|\psi'_z\rangle$ is within $\frac{1}{10}$ of $|1\rangle$.

On the other hand, the trace distance between $|\psi'_z\rangle$ and $|\psi_{z+2^r}\rangle$ can be calculated to be below 0.7712, as follows. The two states are identical in qubit positions $n-1, n-2, \ldots, k$. In qubit position $k-1$, the two states differ by an angle of $\frac{\pi}{4}$, in qubit position $k-2$ the two states differ by an angle of $\frac{\pi}{8}$, and so on. Therefore,

$$\begin{aligned}\langle\psi'_z|\psi_{z+2^r}\rangle &= \langle\mu_{0.1^{r+2}}|\mu_{0.0^21^r}\rangle\cdots\langle\mu_{0.1^n}|\mu_{0.0^{n-r}1^r}\rangle\\ &= \cos(\tfrac{\pi}{2^2})\cos(\tfrac{\pi}{2^3})\cdots\cos(\tfrac{\pi}{2^{n-k-1}})\\ &> \cos(\tfrac{\pi}{2^2})\cos(\tfrac{\pi}{2^3})\cos(\tfrac{\pi}{2^4})\cdots\\ &> 0.6366.\end{aligned}$$

This implies that the trace distance between $|\psi'_z\rangle$ and $|\psi_{z+2^r}\rangle$ is less than $\sqrt{1-(0.6366)^2} < 0.7712$. Since the trace distance is contractive, it follows that the state of the high-order output of C on input $|\psi'_z\rangle$ has trace distance less than 0.7712 from the state of high-order output of C on input $|\psi_{z+2^r}\rangle$. But, by the triangle inequality, this implies that the trace distance between $|0\rangle$ and $|1\rangle$ is less than $\frac{1}{10}+0.7712+\frac{1}{10} < 1$, which is a contradiction, since $|0\rangle$ and $|1\rangle$ are orthogonal. This completes the proof of Theorem 3.

6. New size bounds for the QFT

In this section, we prove Theorem 4. Let F_{2^n} denote the Fourier transform modulo 2^n, which acts on n qubits.

The standard quantum circuit for F_{2^n} can be described recursively as follows (where the controlled-phase shift gates c-$P(\theta)$ are defined in Section 2).

Standard recursive circuit description for F_{2^n}:

1. Apply $F_{2^{n-1}}$ to the first $n-1$ qubits.
2. For each $j \in \{1,\ldots,n-1\}$, apply c-$P(1/2^{n-j+1})$ to the j^{th} and n^{th} qubit.
3. Apply H to the n^{th} qubit.

The resulting circuit consists of $n(n-1)/2$ two-qubit gates and n one-qubit gates.

Below is a more general recursive circuit description for F_{2^n}, parameterized by $m \in \{1,\ldots,n-1\}$ (based on [10]). This coincides with the above circuit when $m=1$. When $m>1$, it can be verified that the circuit does not change very much. It has exactly the same gates, though the relative order of the two-qubit gates (which all commute with each other) changes.

Generalized recursive circuit description for F_{2^n}:

1. Apply $F_{2^{n-m}}$ to the first $n-m$ qubits.
2. For each $j \in \{1,\ldots,n-m\}$ and $k \in \{1,\ldots,m\}$, apply c-$P(1/2^{k-j+1})$ to the j^{th} and $(n-m+k)^{\text{th}}$ qubit.
3. Apply F_{2^m} to the last m qubits.

Our new quantum circuits are based on this generalized recursive construction with $m=\lfloor n/2\rfloor$, except that they use a more efficient method for performing the transformation in Step 2. Step 2 consists of $(n-m)m$ (or approximately $n^2/4$) two-qubit gates. The key observation is that Step 2 computes the mapping that, for $x\in\{0,1\}^{n-m}$ and $y\in\{0,1\}^m$, takes the state $|x\rangle|y\rangle$ to the state $(e^{2\pi i/2^n})^{x\cdot y}|x\rangle|y\rangle$, where $x\cdot y$ denotes the product of x and y interpreted as binary integers. From this, it can be shown that Step 2 can be computed using any classical method for integer multiplication in conjunction with some one-qubit phase shift gates (of the form $P(\theta)$, defined in Section 2).

Currently, the best known asymptotic circuit size for integer multiplication, due to Schönhage and Strassen [32], is $O(n\log n\log\log n)$. This can be translated into a reversible computation of the same size that we will denote as S. For $x\in\{0,1\}^{n-m}$ and $y\in\{0,1\}^m$, S maps the state $|x\rangle|y\rangle|0^n\rangle$ to $|x\rangle|y\rangle|x\cdot y\rangle$. (There are $O(n)$ additional ancilla qubits that are not explicitly indicated. Each of these begins and ends in state $|0\rangle$.)

Improved Step 2 in general circuit description for F_{2^n}:

2a. Apply S to the $2n$ qubits.

2b. For each $k\in\{1,\ldots,n\}$, apply $P(1/2^k)$ to the $(n+k)^{\text{th}}$ qubit.

2c. Apply S^{-1} to the $2n$ qubits.

Using this improved Step 2 in the generalized recursive circuit description for F_{2^n} results in a total number of gates that satisfies the recurrence

$$T_n = T_{\lceil n/2\rceil} + T_{\lfloor n/2\rfloor} + O(n\log n\log\log n),$$

which implies that $T_n \in O(n(\log n)^2\log\log n)$. It is straightforward to also show that the circuit has depth $O(n)$ and width $O(n)$.

7. Arbitrary moduli

In this section we sketch a proof of Theorem 5, which states that it is possible to approximate the QFT with respect to an arbitrary modulus m in parallel with high accuracy. This can be done using our circuits for the QFT modulo 2^k for $k=\lfloor\log m\rfloor+O(1)$. The depth of the circuit is $O(\log n\log\log(1/\varepsilon))$ and the size is polynomial in $n+\log(1/\varepsilon)$.

The method exploits a relation between QFTs with different moduli that was used by Hales and Hallgren [21] in regard to the so-called *Fourier Sampling* problem (see also Høyer [22] for an extension and simplified proof).

The basic components of the technique are as follows:

1. Create a Fourier state with modulus m, which is the mapping $|x\rangle|0\rangle \mapsto |x\rangle|\psi_x\rangle$.

2. Copy the Fourier state, which corresponds to the mapping $|x\rangle|\psi_x\rangle|0\rangle\cdots|0\rangle \mapsto |x\rangle|\psi_x\rangle|\psi_x\rangle\cdots|\psi_x\rangle$.

3. Apply the inverse Fourier transform modulo 2^k on each state $|\psi_x\rangle$, which is the mapping
$$|x\rangle|\psi_x\rangle\cdots|\psi_x\rangle \mapsto |x\rangle\left(F^\dagger_{2^k}|\psi_x\rangle\right)\cdots\left(F^\dagger_{2^k}|\psi_x\rangle\right).$$

4. Compute $\lfloor ym2^{-k}+1/2\rfloor \bmod m$ for each computational basis state y occurring among the collections of qubits on which $F^\dagger_{2^k}$ was performed. An observation of $F^\dagger_{2^k}|\psi_x\rangle$ in the computational basis yields some y with $\lfloor ym2^{-k}+1/2\rfloor = x$ with probability exceeding $1/2+\delta$ for some constant δ, which implies that with high probability the most frequent value obtained for $\lfloor ym2^{-k}+1/2\rfloor \bmod m$ will be x. XOR this result to the qubits in state $|x\rangle$, and reverse the computation of each $\lfloor ym2^{-k}+1/2\rfloor$ and y. With high probability the mapping
$$|x\rangle\left(F^\dagger_{2^k}|\psi_x\rangle\right)\cdots\left(F^\dagger_{2^k}|\psi_x\rangle\right) \\ \mapsto |0\rangle\left(F^\dagger_{2^k}|\psi_x\rangle\right)\cdots\left(F^\dagger_{2^k}|\psi_x\rangle\right)$$
has been performed.

5. Reverse steps 3 and 2, giving the mapping
$$|0\rangle\left(F^\dagger_{2^k}|\psi_x\rangle\right)\cdots\left(F^\dagger_{2^k}|\psi_x\rangle\right) \mapsto |0\rangle|\psi_x\rangle|0\rangle\cdots|0\rangle.$$

Unfortunately some of the methods used in the power of 2 case (such as using carry-save adders and approximating the individual qubits of the Fourier basis states) do not seem to work in this case, which results in the slightly worse depth bound. The overall size bound increases as well, but is still polynomial.

It is interesting to note that this method does not require the larger modulus to be a power of 2—effectively the method shows that the QFT modulo m for any modulus m can be efficiently approximated given a black box that approximates the QFT modulo m' for any sufficiently large m'. Further technical details regarding this method will appear in the final version of this paper.

8 Shor's "mixed-radix" QFT

We conclude with a brief discussion of Shor's original "mixed radix" method for computing the QFT, as it too can be parallelized (although to our knowledge not as efficiently as the power-of-2 case discussed previously in this paper).

Shor's original method for computing the QFT is based on the Chinese Remainder Theorem and its consequences regarding $\mathbb{Z}_m$ for given modulus m. Here the modulus is $m = m_1m_2\cdots m_k$ for $m_1,\ldots,m_k$ pairwise relatively prime and $m_j \in O(\log m)$. Thus $k \in O(\log m/\log\log m)$ is somewhat less than the number of bits of m, and each m_j has length logarithmic in the length of m. Taking m_j to be the j^{th} prime results in a sufficiently dense collection of moduli m for factoring [33] (see Rosser and Schoenfeld [31] for explicit bounds and a detailed analysis of such bounds).

Although stated somewhat differently by Shor, the mixed radix QFT method may be described as follows:

1. For each $j = 1,\ldots,k$ define $f_j = m/m_j$ and set $g_j \in \mathbb{Z}_{m_j}$ such that $g_j \equiv f_j^{-1} \pmod{m_j}$.

2. Define C to be the (reversible) operator acting as follows for each $x \in \{0,\ldots,m-1\}$:
$$C : |x\rangle \mapsto |(x \bmod m_1),\ldots,(x \bmod m_k)\rangle$$

3. Define A to be the (reversible) operator such that
$$A : |x_1,\ldots,x_k\rangle \mapsto |g_1x_1,\ldots,g_kx_k\rangle$$
for each $(x_1,\ldots,x_k) \in \mathbb{Z}_{m_1}\times\cdots\times\mathbb{Z}_{m_k}$.

4. Let F_m and F_{m_j} denote the QFT for moduli m and m_j, $j = 1,\ldots,k$, respectively. Then the following relation holds:
$$F_m = C^\dagger(F_{m_1}\otimes\cdots\otimes F_{m_k})AC. \qquad (6)$$
Thus, to perform the QFT modulo m on $|x\rangle$, first convert x to its *modular representation* $(x_1,\ldots,x_k)$ using the operator C, multiply each x_j by g_j (modulo m_j), perform the QFT modulo m_j independently on coefficient j (for each j), then apply the inverse of C to convert back to the ordinary representation of elements in $\mathbb{Z}_m$.

The numbers computed in step 1 are used in the standard proof of the Chinese Remainder Theorem: given $x_1,\ldots,x_k$, we have that for $x = \sum_{j=1}^k f_jg_jx_j \bmod m$, the congruence $x \equiv x_j \pmod{m_j}$ is satisfied for each j. Thus the operator C can be implemented efficiently, since the mapping
$$x \mapsto ((x \bmod m_1),\ldots,(x \bmod m_k))$$

and its inverse are efficiently computable (e.g., with size $O(\log^2 m)$ circuits [2]). In the present case C can be parallelized to logarithmic depth, since each of the moduli are small. Similarly, the operator A can be parallelized to logarithmic depth.

To see that Eq. 6 holds, we may simply examine the action of the operator on the right hand side on computational basis states:

$$\begin{aligned}
&C^\dagger(F_{m_1} \otimes \cdots \otimes F_{m_k})AC|x\rangle \\
&\quad = C^\dagger(F_{m_1} \otimes \cdots \otimes F_{m_k})|g_1x_1, \ldots, g_kx_k\rangle \\
&\quad = \tfrac{1}{\sqrt{m}}C^\dagger \sum_{y_1,\ldots,y_k} \exp(2\pi i \textstyle\sum_{j=1}^k f_jg_jx_jy_j/m)|y_1,\ldots,y_k\rangle \\
&\quad = \tfrac{1}{\sqrt{m}} \sum_y \exp(2\pi ixy/m)|y\rangle \\
&\quad = F_m|x\rangle.
\end{aligned}$$

Finally, the QFTs modulo $m_1, \ldots, m_k$ can be done simultaneously in order to parallelize the entire process. Originally, Shor suggested implementing each of these operations by circuits of size m_j, since any quantum operation can be computed by circuits with exponential-size quantum circuits [3]. This results in a linear-depth circuit overall, although the circuit will be exact.

However, we may try to compute each F_{m_j} more efficiently. There are a few possibilities for how to do this, all (apparently) requiring approximations of each F_{m_j}. First, we may apply the method of Kitaev [24] to approximate these QFTs. Alternately, we may use the arbitrary modulus method we have proposed in Section 7. Finally, we have noted that this method works for any two moduli (not just for the larger modulus a power of 2) so we could recurse using the mixed-radix method to approximate each F_{m_j}. In all cases, our analysis has revealed that the mixed radix method results in worse size and/or depth bounds than the power of 2 method presented in Section 3.

9. Conclusion

We have proved several new bounds on the circuit complexity of approximating the quantum Fourier transform, and have applied these bounds to the problem of factoring using quantum circuits. There are several related open questions, a few of which we will now discuss.

First, is it possible to perform the quantum Fourier transform *exactly* using logarithmic- or poly-logarithmic-depth quantum circuits? The best currently known upper bound on the depth of the exact QFT is linear in the number of input qubits.

Next, can the efficiency of our techniques be improved significantly? We have concentrated on asymptotic analyses of our circuits, and we believe it is certain that our circuits can be optimized significantly for "interesting" input sizes (perhaps several hundred to a few thousand qubits).

Finally, the fact that the quantum Fourier transform can be performed in logarithmic depth suggests the following question: are there interesting natural problems in BQNC (bounded-error quantum NC) not known to be in NC or RNC? For instance, computing the greatest common divisor of two n-bit integers and computing $a^b \bmod c$ and $a^{-1} \bmod c$ for n-bit integers a, b, and c is not known to be possible using polynomial-size circuits with depth poly-logarithmic in n in the classical setting. Are there logarithmic- or poly-logarithmic-depth quantum circuits for these problems? Greenlaw, Hoover and Ruzzo [18] list several other problems not known to be classically parallelizable, all of which are interesting problems to consider in the quantum setting.

Acknowledgments

We would like to thank Dorit Aharonov, Wayne Eberly, Chris Fuchs, Daniel Gottesman, Patrick Hayden, Hoi-Kwong Lo, and John Preskill for helpful discussions.

References

[1] D. Aharonov and M. Ben-Or. Fault tolerant quantum computation with constant error. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 176–188, 1997.

[2] E. Bach and J. Shallit. *Algorithmic Number Theory, Volume I: Efficient Algorithms*. MIT Press, 1996.

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

[4] P. Beame, S. Cook, and H. J. Hoover. Log depth circuits for division and related problems. *SIAM Journal on Computing*, 15(4):994–1003, 1986.

[5] C. H. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17:525–532, 1973.

[6] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997.

[7] D. Boneh and R. Lipton. Quantum cryptanalysis of hidden linear functions. In *Advances in Cryptology – Crypto'95*, volume 963 of *Lecture Notes in Computer Science*, pages 242–437. Springer-Verlag, 1995.

[8] G. Brassard, P. Høyer, and A. Tapp. Quantum counting. In *Proceedings of the 25th International Colloquium on Automata, Languages and Programming*, volume 1443 of *Lecture Notes in Computer Science*, pages 820–831, 1998.

[9] A. Chiu, G. Davida, and B. Litow. NC1 division. Manuscript, November 1999.

[10] R. Cleve. A note on computing quantum Fourier transforms by quantum programs. Manuscript. Available at http://www.cpsc.ucalgary.ca/~cleve/papers.html, 1994.

[11] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca. Quantum algorithms revisited. *Proceedings of the Royal Society, London*, A454:339–354, 1998.

[12] J. W. Cooley and J. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19:297–301, 1965.

[13] D. Coppersmith. An approximate Fourier transform useful in quantum factoring. Technical Report RC19642, IBM, 1994.

[14] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press/McGraw-Hill, 1990.

[15] D.-Z. Du and K.-I. Ko. *Theory of Computational Complexity*. John Wiley & Sons, 2000.

[16] C. F. Gauss. Theoria interpolationis methodo nova tractata. In *Werke III, Nachlass*, pages 265–330. Königliche Gesellschaft der Wissenschaften, 1866. Reprinted by Georg Olms Verlag, Hildesheim, New York, 1973.

[17] O. Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*. Springer, 1999.

[18] R. Greenlaw, H. J. Hoover, and W. Ruzzo. *Limits to Parallel Computation*. Oxford University Press, 1995.

[19] D. Grigoriev. Testing shift-equivalence of polynomials using quantum machines. In *Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation*, pages 49–54, 1996.

[20] L. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pages 212–219, 1996.

[21] L. Hales and S. Hallgren. Quantum Fourier sampling simplified. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 330–338, 1999.

[22] P. Høyer. *Quantum Algorithms*. PhD thesis, Odense University, Denmark, 2000.

[23] D. S. Johnson. A catalog of complexity classes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science: Volume A, Algorithms and Complexity*, pages 67–171. MIT Press/Elsevier, 1990.

[24] A. Kitaev. Quantum measurements and the abelian stabilizer problem. Los Alamos Preprint Archive, quant-ph/9511026, 1995.

[25] E. Knill, R. Laflamme, and W. Zurek. Threshold accuracy for quantum computation. Los Alamos Preprint Archive, quant-ph/9610011, 1996.

[26] R. Ladner and M. Fischer. Parallel prefix computation. *Journal of the ACM*, 27(4):831–838, 1980.

[27] C. Moore and M. Nilsson. Parallel quantum computation and quantum codes. Los Alamos Preprint Archive, quant-ph/9808027, 1998.

[28] M. Mosca. Quantum searching and counting by eigenvector analysis. In *Proceedings of Randomized Algorithms, Workshop of MFCS 98*, 1998.

[29] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[30] J. Preskill. Fault-tolerant quantum computation. In H.-K. Lo, S. Popescu, and T. P. Spiller, editors, *Introduction to Quantum Computation and Information*, pages 213–269. World Scientific, 1998.

[31] J. B. Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics*, 6:64–94, 1962.

[32] A. Schönhage and V. Strassen. Schnelle multiplikation großer zahlen. *Computing*, 7:281–292, 1971.

[33] P. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

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

[35] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.