International Journal of Quantum Information
Vol. 11, No. 1 (2013) 1350008 (11 pages)
© World Scientific Publishing Company
DOI: 10.1142/S0219749913500081

World Scientific
www.worldscientific.com

QUANTUM PHASE ESTIMATION WITH AN ARBITRARY NUMBER OF QUBITS

CHEN-FU CHIANG

Département de Physique,
Université de Sherbrooke Sherbrooke,
Québec, Canada J1K 2R1S
Chen-Fu.Chiang@USherbrooke.ca

Received 23 September 2012
Revised 12 February 2013
Accepted 15 February 2013
Published 11 April 2013

Due to the great difficulty in scalability, quantum computers are limited in the number of qubits during the early stages of the quantum computing regime. In addition to the required qubits for storing the corresponding eigenvector, suppose we have additional k qubits available. Given such a constraint k, we propose an approach for the phase estimation for an eigenphase of exactly n-bit precision. This approach adopts the standard recursive circuit for quantum Fourier transform (QFT) in [R. Cleve and J. Watrous, Fast parallel circuits for quantum fourier transform, *Proc. 41st Annual Symp. on Foundations of Computer Science* (2000), pp. 526−536.] and adopts classical bits to implement such a task. Our algorithm has the complexity of $O(n \log k)$, instead of $O(n^2)$ in the conventional QFT, in terms of the total invocation of rotation gates. We also design a scheme to implement the factorization algorithm by using k available qubits via either the continued fractions approach or the simultaneous Diophantine approximation.

Keywords: Quantum Fourier transform; quantum phase estimation; quantum circuit; diophantine approximation.

1. Introduction

Quantum phase estimation (QPE) is a key quantum operation in many quantum algorithms.[1−5] Phase estimation is extensively used to solve a variety of problems, such as hidden subgroup, graph isomorphism, quantum walk, quantum sampling, adiabatic computing, order-finding and large number factorization. QPE comprises two components: *phase kick back* and *inverse quantum Fourier transform* (*QFT*). The implementation of QFT has been described in numerous research articles.[6−10] The physical implementation (algorithms based on QFT) is highly constrained by the

requirement of (1) high-precision controlled rotation gates (phase shift operators), which remain difficult to realize and (2) sufficient number of qubits to approximate the eigenphase to a required precision.

At the early stage of a quantum computing implementation, we can imagine that scalability could be an issue. The quantum resources could be limited, in terms of available quantum qubits and quantum gates. From that perspective, efficient implementations of quantum algorithms are essential when available quantum resources are scarce. For instance, Parker and Plenio[11] show that a single pure qubit together with a collection of $\log_2 N$ qubits in an arbitrary mixed (or pure) state is sufficient to implement Shor's factorization algorithm efficiently to factorize a large number N. Such implementation addresses the issue of limited qubits but introduces the concern for the decoherence.

In this paper, we are interested in the following two aspects. (1) Given certain available qubits, assuming $k + \log_2 N$ qubits in total, we want to have an efficient way to implement QPE and use as few controlled rotation gates (c-r.g.) as possible. (2) Apply this technique to Shor's factorization algorithm along with simultaneous Diophantine approximation[12] to investigate the feasible implementation structure when the available qubits are limited. We assume only one copy of the eigenvector $|u\rangle$ (requiring $\log_2 N$ qubits) and additional k qubits are available. One copy of the eigenvector implies a restriction on the use of controlled-U gates: all controlled-U gates should be applied on the workspace register (k qubits).

One copy of an eigenvector is a reasonable assumption because multiple copies of $|u\rangle$ would imply the requirement for extra multiple of $\log_2 N$ qubits for storage. Hence, it is practical as we are considering the case that the available qubits are scarce. Thus, the entire process is a single circuit ($\lceil n/k \rceil$ stages) that *cannot be divided into parallel processes*. Under such an assumption, for approaches that require repetitions, such as Kitaev's[7] and others,[9] parallelization cannot be done and the circuit depth is the same as the size of the circuit. On the other hand, if we have enough qubits for storing multiple copies of eigenvector $|u\rangle$, we should choose Kitaev's approach because the processes can thus be run in parallel. Throughout the rest of the article, we will refer to the k available qubits as the qubits used in the workspace register.

Generally speaking, quantum circuits for QFT implemented in different approaches[6–10] would require the same number of controlled-U gates but different numbers of rotation gates. We are interested in using the recursive approach, along with some classical resources, to implement the inverse QFT. We bound the number of required rotation gates from above.

We give an overview of the conventional QPE technique in Sec. 2. We detail our algorithms and the analysis in Sec. 3, including a brief analysis of Kitaev's original approach.[7] An application of our approach along with simultaneous Diophantine approximation to the factorization problem is given in Sec. 3.4. Finally, we state our conclusion in Sec. 4.

2. Approach Based on QFT

One of the standard methods to approximate the phase of a unitary matrix is QPE based on QFT. The structure of this method is depicted in Fig. 1.

The QPE algorithm requires two registers and contains two stages. Suppose the eigenphase of unitary U is $\varphi = 0.\varphi_1\varphi_2\cdots\varphi_n$ in the binary representation such that

$$U|u\rangle = \mathrm{e}^{2\pi i\varphi}|u\rangle. \tag{1}$$

Then, the first register is prepared as a composition of n qubits initialized in the state $|0\rangle$. The second register is initially prepared in the state $|u\rangle$. The first stage prepares a uniform superposition over all possible states and then applies controlled-U^{2^l} operations. Consequently, the state becomes

$$\frac{1}{2^{n/2}}\sum_{l=0}^{2^n-1}\mathrm{e}^{2\pi i\varphi l}|l\rangle. \tag{2}$$

The second stage in the QPE algorithm is the QFT† operation. At each step (starting from the least significant bit) by using the information from previous steps, the inverse Fourier transform transforms the state

$$\frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i 2^l\varphi}|1\rangle) \tag{3}$$

to get closer to one of the states

$$\begin{aligned}\frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i 0.0}|1\rangle) &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad \text{or}\\ \frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i 0.1}|1\rangle) &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).\end{aligned} \tag{4}$$

Suppose φ is precise to the 3rd bit, that is $\varphi = 0.\varphi_1\varphi_2\varphi_3$. As shown in Fig. 2, each step (dashed-line box) uses the result of previous steps, where phase shift operators are defined as

$$R_l \equiv \begin{bmatrix}1 & 0\\ 0 & \mathrm{e}^{2\pi i/2^l}\end{bmatrix} \tag{5}$$

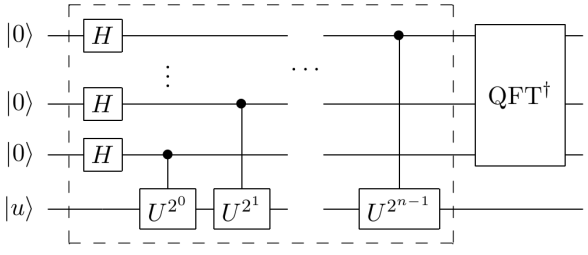

Fig. 1. Standard QPE with n qubits as ancilla. The dash-line box is the phase kickback.

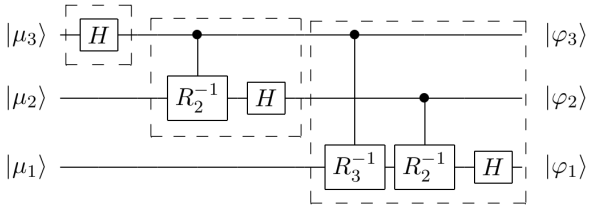

Fig. 2. 3-qubit inverse QFT where $1 \leq j \leq 3, |\mu_j\rangle = \frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i(0.\varphi_j \cdots \varphi_3)}|1\rangle)$.

for $2 \leq l \leq 3$. By concatenating φ_1, φ_2 and φ_3, we obtain φ. Therefore, when φ is precise to the n-th bit, the total number of rotation gate invocations is $O(n^2)$.

3. Our Algorithm

Before proceeding to our algorithm, we provided the description of the recursive circuit for QFT[6] technique. In Ref. 6, in addition to the recursive circuit, the authors also adopted the technique by Schönhage and Strassen[13] for integer multiplication. However, the integer multiplication is performed via classical computation. We use the classical bits and operators to compute the parameter (the desired phase shift) of a quantum rotation gate. The algorithm structure is explained in Sec. 3.2.

3.1. *Standard recursive circuit description for* $F_{2^n}^{\dagger}$

Let $F_{2^n}^{\dagger}$ denote the inverse Fourier transform modulo 2^n that acts on n qubits. The standard quantum circuit for $F_{2^n}^{\dagger}$ can be described recursively as follows. Let us denote this circuit as $RF_{2^n}^{\dagger}$.

(1) Suppose the state of the work register after phase kickback is

$$\begin{aligned} |\psi\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i 0.\varphi_1 \cdots \varphi_n}|1\rangle) \otimes \cdots \otimes \frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i 0.\varphi_n}|1\rangle) \\ &= |\mu_1\rangle \otimes \cdots \otimes |\mu_n\rangle \end{aligned} \tag{6}$$

(2) Apply $F_{2^m}^{\dagger}$ to the last m qubits ($|\mu_{n-m+1}\rangle \otimes \cdots \otimes |\mu_n\rangle$).
(3) Read out and store the values of the m qubits in classical bits ($c_1 \cdots c_m$).
(4) Compute rotation angle: $f(c_1 \cdots c_m) = \sum_{i=1}^{m} (\frac{1}{2})^i \cdot c_i$.
(5) For each $j \in \{1, 2, \ldots, n-m\}$, apply the rotation gate $R^{\dagger}_{\frac{f(c_1 \cdots c_m)}{2^{n-m-j+1}}}$ to the jth qubit.

Here the rotation gate $R^{\dagger}_{\frac{f(c_1 \cdots c_m)}{2^{n-m-j+1}}}$ is defined as

$$R_q^{\dagger} = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{e}^{-2\pi i \frac{f(c_1 \cdots c_m)}{2^{n-m-j+1}}} \end{pmatrix}$$

(6) Apply $F_{2^{n-m}}^{\dagger}$ to the first $n-m$ qubits.

For simplicity, let us assume that n is some power of 2. Then step 5 is the step that resets the disturbing eigenphase bits for the first $n-m$ qubits because all the disturbing eigenphase bits from the last m bits will be cleared. The number of required rotation operations for such a step is $n/2$ (suppose we choose $m = n/2$) as we have to reset for each qubit in the last $n-m$ qubits.

It is clear to that the total number of required rotation gates is

$$T_n = T_{n/2} + T_{n/2} + n/2, \tag{7}$$

where $T_1 = 1$. Hence, the complexity is $O(n \log n)$[a] for such a recursive circuit.

3.2. *The algorithm structure*

Given k ancillary bits initialized in $|0\rangle$ and eigenvector $|u\rangle$ of unitary U where $U|u\rangle = \mathrm{e}^{2\pi i\varphi}|u\rangle$ as input, we want to estimate the eigenphase of U precise to the nth bit. The algorithm comprises $\lceil n/k \rceil$ stages that run in sequence. At each stage, we perform phase kickback, controlled rotation operation and recursive inverse Fourier transform to obtain k eigenphase bits. Once the last stage finishes, we can concatenate the obtained eigenphase bits, resulting in an estimated eigenphase of φ. For the details, please refer to Algorithm 1 listed below.

Algorithm 1 Phase estimation with variable number of qubits

Input: k ancillary bits initialized in $|0\rangle$ and eigenvector $|u\rangle$ of unitary U where $U|u\rangle = \mathrm{e}^{2\pi i\varphi}|u\rangle$.

Step I:
At stage j, where $j \in \{1, \ldots, \lceil n/k \rceil\}$, run phase kick back on k qubits by using the controlled U^{2^l} operations. Note that $l \in \{n-jk, n-jk+1, \ldots, n-(j-1)k-1\}$.

Step II:
For $t \in \{1, \ldots, k\}$, apply the rotation gate $R^{\dagger}_{\frac{F[j-1]}{2^{k-t+1}}}$ to the t_{th} qubit.

Apply the generalized recursive circuit $F^{\dagger}_{2^k}$.
Read out the result to k classical bits $(c_1 \cdots c_k)$ (the actual label is $c_{n-jk+1} \cdots c_{n-(j-1)k}$).
Compute the value $F[j] = f(c_1 \cdots c_k) + \frac{F[j-1]}{2^k}$ where $f(c_1 \cdots c_k) = \sum_{i=1}^{k} (\frac{1}{2})^i \cdot c_i$.
Reset k qubits to $|0\rangle$

Step III:
Repeat Step I and Step II $\lceil n/k \rceil$ times (i.e. $\lceil n/k \rceil$ stages)

Output:
Concatenate the n classical bits $c_1, \ldots, c_n$, resulting in an estimated eigenphase $\varphi = 0.c_1c_2c_3\ldots$.

[a] In this work, log is always of base 2, unless otherwise specified.

Let us write the eigenphase φ in the binary presentation as $0.\varphi_1 \cdots \varphi_n$. Let $|\psi\rangle = |0\rangle^{\otimes k}|u\rangle$ be the initial state at stage j before the phase kickback. After Step I, we obtain the state

$$|\Phi\rangle_1 = \frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i 0.\varphi_{n-jk+1}\cdots\varphi_{n-1}\varphi_n}|1\rangle) \otimes \cdots \otimes \frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i 0.\varphi_{n-jk+k}\cdots\varphi_{n-1}\varphi_n}|1\rangle). \quad (8)$$

It is clear to see that for the tth qubit that the eigenphase discovered from previous stages is shifted to the right by $k-t+1$ bits in the binary presentation. At the beginning of Step II, by applying the rotation gate $R^{\dagger}_{\frac{1}{2^{k-t+1}}F[j-1]}$,[b] we reset the discovered eigenphase in those k qubits. Hence, we obtain the state

$$|\Phi\rangle_{2-1} = \frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i 0.\varphi_{n-jk+1}\cdots\varphi_{n-jk+k}}|1\rangle) \otimes \cdots \otimes \frac{1}{\sqrt{2}}(|0\rangle + \mathrm{e}^{2\pi i 0.\varphi_{n-jk+k}}|1\rangle). \quad (9)$$

Now we have reduced the scenario to the case where the disturbing eigenphase bits from previous stages are reset to 0. Hence, we can use the general recursive circuit for the inverse QFT to obtain the eigenphase bits $(\varphi_{n-jk+1}, \varphi_{n-jk+2}, \ldots, \varphi_{n-jk+k})$.

Once we obtain the k eigenphase bits, we can read out and store them in classical bits to compute $F[j]$. We refer interested readers to Ref. 14 for the details in this semiclassical approach. The value, $F[j]$ will be used again in the next stage for resetting the previous $j \times k$ eigenphase bits. Fig. 3 depicts the process of a single iteration.

3.3. *The analysis*

The cost of our algorithm has two parts: classical and quantum. For the classical part, we need n classical bits, $k+2$ doubles[c] and $k+2$ classical operators. n classical bits are used to store *all of the* observed eigenphase bits. At any given stage (say j), two primitive doubles,[c] X and Y are required such that we have

$$X = F[j], \quad Y = F[j-1].$$

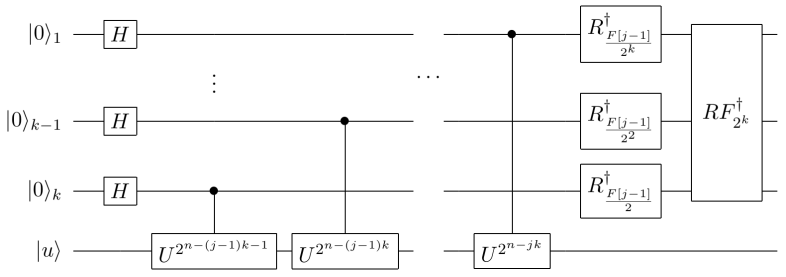

Fig. 3. At stage j: with k qubits as ancilla, k rotation operations and one $RF^{\dagger}_{2^k}$ operation.

[b] $F[0] = 0$.

[c] Assume a classical double data structure is of size 64 bits.

To generate k different rotation angle operators (see the first substep of Step II in Algorithm 1), we need k doubles (Reg[k], an array of k doubles) and k operators[d] to generate the parameter,

$$\frac{F[j-1]}{2^{k-t+1}},$$

of a quantum rotation gate for the t-th qubit at the j-th iteration where $t \in \{1, 2, \ldots, k\}$.

Once the k eigenphase bits are stored in classical bits in the j-th iteration, a classical operator computes $F[j]$ such that

$$F[j] = X = f(c_1 \cdots c_k) + \frac{Y}{2^k}.$$

Then another operator sets $Y = X$. By doing so, double X and Y can be reused in the next iteration. Therefore, classically n classical bits, $k+2$ doubles and $k+2$ classical operators are needed. The same device (classical requirement) can be used inside the recursive circuit since our approach is sequential, not parallelled. The classical requirements are summed in Table 1.[c]

For the quantum part, the number of total rotation gate invocations in our approach would be

$$k \log k + (\lceil n/k \rceil - 1)((k + k \log k)) \approx O(n \log k). \tag{10}$$

The reasoning is as follows. At stage $j = 1$, the rotation operations only occur inside the recursive inverse Fourier transform RF_{2^k} as $F[0] = 0$. For stage $j = 2, \ldots, \lceil \frac{n}{k} \rceil$, it is required to have rotation gates $R^{\dagger}_{\frac{F[j-1]}{2^{k-t+1}}}$, where $1 \leq t \leq k$, to reset the dangling eigenphase bits before the recursive inverse Fourier transform RF_{2^k}. Based on the cost function for RF_{2^k} derived in Eq. (7), we obtain the cost for our approach as shown in Eq. (10).

For comparison with other known existing approaches, in the following section we will briefly describe the analysis and the result rendered in Ref. 9 regarding Kitaev's original approach.[7]

3.3.1. *Kitaev's original approach*

In this approach, a series of Hadamard tests are performed for each eigenphase bit in order to recover the phase correctly. Suppose the precision up to the n-th bit is required, then in each test the phase[e] $\phi_l = 2^{l-1}\varphi (1 \leq l \leq n)$ must be computed up to

Table 1. Required classical bits.

Register type	Required number of bits
X (classical register)	64
Y (classical register)	64
Reg[k] (classical register)	$64k$
Classical bits for eigenphase	n

[d] Because we can generate those parameters in parallel.

[e] See Sec. 2 for the description of the eigenphase φ and the unitary U.

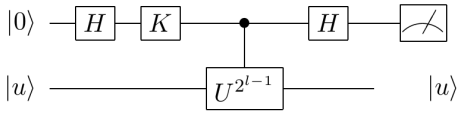

Fig. 4. Hadamard test with extra phase shift operator.

precision 1/16. We perform the Hadamard test on the l-th eigenphase bit, starting from $l = n$ down to 1, as depicted in Fig. 4.

When $K = I_2$, the probabilities of post-measurement of the Hadamard test are

$$\Pr(0|k) = \frac{1 + \cos(2\pi\varphi_k)}{2}, \quad \Pr(1|k) = \frac{1 - \cos(2\pi\varphi_k)}{2}. \tag{11}$$

However, a cosine cannot distinguish ϕ_l and $-\phi_l$. We need to choose $K = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$ to be able to distinguish. The probabilities of the post-measurement states based on the modified Hadamard test become

$$\Pr(0|l) = \frac{1 - \sin(2\pi\phi_l)}{2}, \quad \Pr(1|l) = \frac{1 + \sin(2\pi\phi_l)}{2}. \tag{12}$$

We then can recover ϕ_l from the estimates of the probabilities. To obtain the required precision of 1/16 for ϕ_l, we can run an iteration of Hadamard tests to estimate $\Pr(1|l)$ to some precision.

Theorem 1.[9] *Assume U is a unitary matrix with eigenvalue $e^{2\pi i\varphi}$ and corresponding eigenvector $|u\rangle$. Suppose $\varphi = 0.\varphi_1 \cdots \varphi_n$ and let $\phi_l = 2^{l-1}\varphi$ $(1 \le l \le n)$. To obtain the required precision of 1/16 for ϕ_l such that the recovered $\tilde{\varphi}$ is precise to the n-th bit with constant success probabilty greater than $\frac{1}{2}$, for each ϕ_l we need to run at least $55 \ln n$ trials of Hadamard tests when using Kitaev's approach.*

We refer the interested readers to Ref. 9 for the details. Since we have n stages for ϕ_l, the required invocation of a rotation gate (Hadamard in this case) in Kitaev's approach is $O(n \ln n)$. Suppose the controlled rotation gates are precise, we list the comparison between Kitaev's approach, the conventional QFT-based approach and our approach in Table 2.

3.4. *An application*

In this section, we will focus on how to use k available qubits to implement the quantum factorization algorithm. Shor's factorization algorithm provides a

Table 2. The number of quantum rotation gates invocations.

Approach type	Conventional	Kitaev's	Ours
Complexity	$O(n^2)$	$O(n \ln n)$	$O(n \log k)$

polynomial approach to factorize a large number N. Suppose N is an L bit composite number of interest. There is no known classical algorithm for factoring in only polynomial time, i.e. that can factor in time $O(L^c)$ for some constant c. The most difficult integers to factor in practice using existing algorithms are those that are products of two large primes of *similar size*, and for this reason these are the integers used in cryptographic applications. The largest such semiprime yet factored was RSA-768, a 768-bit number with 232 decimal digits.[15]

Quantumly, it is shown such a task can be done by using $O(L^3)$ operations. The algorithm is two-fold. It first runs phase estimation to obtain the eigenphase $\varphi = 0.\varphi_1\varphi_2\cdots \approx s/r$ where r is the order of an arbitrary element x (that is $x^r = 1(\text{mod } N)$). The second part of the algorithm involves the continued fractions algorithm to approximate s/r, based on the eigenphase we obtain in phase estimation, in order to recover the order r. If r is even, then we know that $(x^{r/2}+1)(x^{r/2}-1) = 0(\text{mod } N)$ and we successfully factorize N into a product of two large numbers of similar size.

However, using the continued fraction algorithm leads inevitably to a squaring of the number to be factored. This follows from the following theorem.

Theorem 2.[8] *Suppose s/r is a rational number such that*

$$|\frac{s}{r} - \varphi| \leq \frac{1}{2r^2}.$$

Then s/r is a convergent of the continued fraction for φ, and thus can be computed in $O(L^3)$ operations using the continued fractions algorithm.

This in turn doubles the length, approximately to $2L+1$ qubits, of the quantum registers in order to achieve required precision $1/2r^2$ since $1 \leq r \leq N \leq 2^L$. Park and Plenio[11] show that they can implement the algorithm by use of 1 qubit[f] along with the semiclassical approach.[14] For such a design, the whole circuit (quantum-wise) consists of $2L+1$ stages of recovering φ_i, where $1 \leq i \leq 2L+1$, and calculating a controlled rotation for the next stage. After obtaining all the φ_i, the post-processing (continued fractions) recovers the order r.

In the work by Seifert,[12] he proposes an alternative to approximate the order by using the simultaneous Diophantine approximation.[16] The theorem is as follows.

Theorem 3.[12] *Let N be the product of two randomly chosen primes of equal size, i.e. of the same length in the binary representations. There exists a randomized polynomial-time quantum algorithm that factors N and uses quantum registers of binary length $\lceil(1+\epsilon)\log N\rceil$, where ϵ is an arbitrarily small positive constant.*[g]

[f] Throughout this section, we also do not count the number of qubits, $\log N$ to be exact, required by the eigenvector of the unitary.

[g] ϵ determines the dimension $d = \frac{1}{1-\frac{1}{1+\epsilon}}$ needed for the good simultaneous Diophantine approximation. It is shown[16] that the complexity is upper bounded from above by $O(L^{12})$ independent of the dimension d.

In such a design, more computations are shifted from the quantum computation part to the classical computation part, in comparison to Shor's algorithm. This might be of importance to practical realizations of a quantum computer. It is also clear that the simultaneous Diophantine approach only requires $(1+\epsilon)\log N$ qubits, that is the precision requirement of $\frac{1}{2^{L(1+\epsilon)}}$ for the phase estimation, to guarantee the existence of a polynomial quantum algorithm for the factorization problem.

Given the constraint that we only have k qubits available for implementation, we have the following scheme (see Algorithm 2).

Clearly this is the tradeoff between the computational complexity (even though both are polynomial) and the available qubits. Quantumly they both have the same number of invocation of the unitary U. However, based on Eq. (10), the number of total quantum rotation gates invocation in the first case is approximately $(2L+1)\times \log k + (2L+1) - k$ while that of the second case is approximately $L(1+\epsilon)\log k + L(1+\epsilon) - k$. At the early stage of a quantum computing implementation, k is probably significantly less than L. In such a scenario, the number of total rotation gate invocation in the first case is approximately twice of that in the second case.

Furthermore, another important issue we need to consider is the decoherence. Despite the fact that the complexity for case I is smaller (classically), it is more costly quantumly. The difference in quantum resources might be amplified when the implementation of error correction is considered as the first case has more stages and more rotation gate invocations.

Algorithm 2 Factorization: Choice of approximation approach

Input: k ancillary bits initialized in $|0\rangle$ and eigenvector $|u\rangle$ of unitary U where $U|u\rangle = \mathrm{e}^{2\pi i\varphi}|u\rangle$.

Case I: Continued Fractions
Choose $n = 2L+1$.
Run algorithm 1 to approximate φ and the number of stages is $\lceil \frac{2L+1}{k} \rceil$.
Run the continued fractions algorithm to recover the order r from the approximated φ.

Case II: Diophantine Approximation
Choose $n = L(1+\epsilon)$.
Run algorithm 1 to approximate φ and the number of stages is $\lceil \frac{L(1+\epsilon)}{k} \rceil$.
Run the simultaneous Diophantine approximation algorithm to recover the order r from the approximated φ.

4. Conclusion

We expect the cost of classical computation to be fairly inexpensive in comparison to its quantum counterpart. Our approach provides a way to obtain the eigenphase

when the number of available qubits is rather limited. It invokes $O(n \log k)$ rotation gates and this gain comes from (1) the use of recursive circuits for $QFT^{\dagger}$ and (2) the use of the classical bits and classical operators.

Another obstacle of high-precision rotation gates (phase shift operators) is not addressed yet. For future work, we could combine with another approach[9] to approximate the eigenphase with variable number of qubits and arbitrary constant-precision operators.

Acknowledgments

C. C. gratefully acknowledges the support of Lockheed Martin Corporation and NSF grants CCF-0726771 and CCF-0746600. The author would like to thank David Poulin and Pawel Wocjan for useful comments and suggestions.

References

1. S. Hallgren, Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem, in *Proc. 34th ACM Symp. on Theory of Computing* (2002).
2. P. Shor, Algorithms for quantum computation: Discrete logarithms and factoring, in *Proc. FOCS* (1994) pp. 124−134.
3. P. Shor, *SIAM J. Comput.* **26** (1997) 1484.
4. M. Szegedy, Quantum speed-up of Markov chain based algorithms, in *Proc. 45th Annual IEEE Sympo. on Foundations of Computer Science* (2004), pp. 32−41.
5. P. Wocjan, C. Chiang, D. Nagaj and A. Abeyesinghe, *Phys. Rev.* **80** (2009), pp. 022340.
6. R. Cleve and J. Watrous, Fast parallel circuits for quantum fourier transform, in *Proc. Annual Symp. Foundations of Computer Sciences* (2000), pp. 526−536.
7. A. Kitaev, A. Shen and M. Vyalyi, *Classical and Quantum Computation*, Vol. 47, Graduate Studies in Mathematics (American Mathematical Society, 2002).
8. M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, 2000).
9. H. Ahmedi and C. Chiang, *Quantum Inform. Comput.* **12** (2012).
10. D. Cheung, Improved bounds for the approximate QFT, in *Proc. Winter Int. Symp. Information and Communication Technologies* (*WISICT*), Trinity College Dublin, (2004), pp. 1−6.
11. S. Parker and M. Plenio, *Phys. Rev. Lett.* **85** (2000) 3049.
12. J. Seifert, Using fewer qubits in Shor's factorization algorithm via simultaneous diophantine approximation, in *Proc. Conf. Topics in Cryptology* (2001), pp. 319−327.
13. A. Schönhage and V. Straßen, *Computing* **7** (1971) 281.
14. R. Griffiths and C. Niu, *Phys. Rev. Lett.* **76** (1996) 3228.
15. T. Kleinjung *et al.*, Factorization of a 768-bit RSA modulus, International Association for Cryptologic Research, Retrieved 2010-08-09.
16. J. Lagarias, *SIAM J. Comput.* **14** (1985) 196.