The Royal Society

Quantum algorithms revisited

By R. Cleve[1], A. Ekert[2], C. Macchiavello[2,3] and M. Mosca[2,4]

[1]*Department of Computer Science, University of Calgary, Calgary, Alberta, Canada T2N 1N4*
[2]*Clarendon Laboratory, Department of Physics, University of Oxford, Parks Road, Oxford OX1 3PU, UK*
[3]*I.S.I. Foundation, Villa Gualino, Viale Settimio Severo 65, 1033 Torino, Italy*
[4]*Mathematical Institute, University of Oxford, 24–29 St. Giles', Oxford OX1 3LB, UK*

Quantum computers use the quantum interference of different computational paths to enhance correct outcomes and suppress erroneous outcomes of computations. A common pattern underpinning quantum algorithms can be identified when quantum computation is viewed as multiparticle interference. We use this approach to review (and improve) some of the existing quantum algorithms and to show how they are related to different instances of quantum phase estimation. We provide an explicit algorithm for generating any prescribed interference pattern with an arbitrary precision.

Keywords: quantum computation; quantum factoring; quantum networks; quantum algorithms; quantum phase estimation

1. Introduction

Quantum computation is based on two quantum phenomena: quantum interference and quantum entanglement. Entanglement allows one to encode data into non-trivial multiparticle superpositions of some preselected basis states, and quantum interference, which is a *dynamical process*, allows one to evolve initial quantum states (inputs) into final states (outputs) modifying intermediate multiparticle superpositions in some prescribed way. Multiparticle quantum interference, unlike single-particle interference, does not have any classical analogue and can be viewed as an inherently quantum process.

It is natural to think of quantum computations as multiparticle processes (just as classical computations are processes involving several 'particles' or bits). It turns out that viewing quantum computation as multiparticle interferometry leads to a simple and a unifying picture of known quantum algorithms. In this language, quantum computers are basically multiparticle interferometers with phase shifts that result from operations of some quantum logic gates. To illustrate this point, consider, for example, a Mach–Zehnder interferometer (figure 1*a*).

A particle, say a photon, impinges on a half-silvered mirror, and, with some probability amplitudes, propagates via two different paths to another half-silvered mirror which directs the particle to one of the two detectors. Along each path between the two half-silvered mirrors is a phase shifter. If the lower path is labelled as state $|0\rangle$

Proc. R. Soc. Lond. A (1998) **454**, 339–354
Printed in Great Britain

TEX Paper

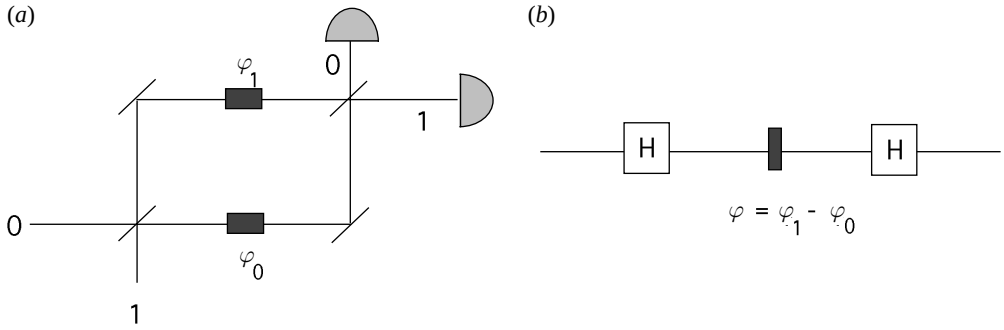

Figure 1. (*a*) Scheme of a Mach–Zehnder interferometer with two phase shifters. The interference pattern depends on the difference between the phase shifts in different arms of the interferometer. (*b*) The corresponding quantum network representation.

and the upper one as state $|1\rangle$, then the state of the particle in between the half-silvered mirrors and after passing through the phase shifters is a superposition of the type $1/\sqrt{2}(|0\rangle + \mathrm{e}^{\mathrm{i}(\phi_1-\phi_0)}|1\rangle)$, where ϕ_0 and ϕ_1 are the settings of the two phase shifters. This is illustrated in figure 1*a*. The phase shifters in the two paths can be tuned to effect any prescribed relative phase shift $\phi = \phi_1 - \phi_0$ and to direct the particle with probabilities $\frac{1}{2}(1+\cos\phi)$ and $\frac{1}{2}(1-\cos\phi)$, respectively, to detectors '0' and '1'. The second half-silvered mirror effectively erases all information about the path taken by the particle (path $|0\rangle$ or path $|1\rangle$) which is essential for observing quantum interference in the experiment.

Let us now rephrase the experiment in terms of quantum logic gates. We identify the half-silvered mirrors with the single-qubit *Hadamard* transform (H), defined as

$$|0\rangle \xrightarrow{H} 1/\sqrt{2}(|0\rangle + |1\rangle), \quad |1\rangle \xrightarrow{H} 1/\sqrt{2}(|0\rangle - |1\rangle). \tag{1.1}$$

The Hadamard transform is a special case of the more general *Fourier* transform, which we shall consider in § 4.

We view the phase shifter as a single-qubit gate. The resulting network corresponding to the Mach–Zehnder interferometer is shown in figure 1*b*. The phase shift can be 'computed' with the help of an auxiliary qubit (or a set of qubits) in a prescribed state $|u\rangle$ and some controlled-U transformation where $U|u\rangle = \mathrm{e}^{\mathrm{i}\phi}|u\rangle$ (see figure 2). Here, the controlled U means that the form of U depends on the logical value of the control qubit; for example, we can apply the identity transformation to the auxiliary qubits (i.e. do nothing) when the control qubit is in state $|0\rangle$ and apply a prescribed U when the control qubit is in state $|1\rangle$. The controlled-U operation must be followed by a transformation which brings all computational paths together, like the second half-silvered mirror in the Mach–Zehnder interferometer. This last step is essential to enable the interference of different computational paths to occur—for example, by applying a Hadamard transform. In our example, we can obtain the following sequence of transformations on the two qubits:

$$\begin{aligned} |0\rangle|u\rangle &\xrightarrow{H} 1/\sqrt{2}(|0\rangle + |1\rangle)|u\rangle \xrightarrow{c-U} 1/\sqrt{2}(|0\rangle + \mathrm{e}^{\mathrm{i}\phi}|1\rangle)|u\rangle \\ &\xrightarrow{H} (\cos(\tfrac{1}{2}\phi)|0\rangle - \mathrm{i}\sin(\tfrac{1}{2}\phi)|1\rangle)\mathrm{e}^{\mathrm{i}\phi/2}|u\rangle. \end{aligned} \tag{1.2}$$

We note that the state of the auxiliary register $|u\rangle$, being an eigenstate of U, is not altered along this network, but its eigenvalue $\mathrm{e}^{\mathrm{i}\phi}$ is 'kicked back' in front of the $|1\rangle$ component in the first qubit. The sequence (1.2) is the exact simulation of the

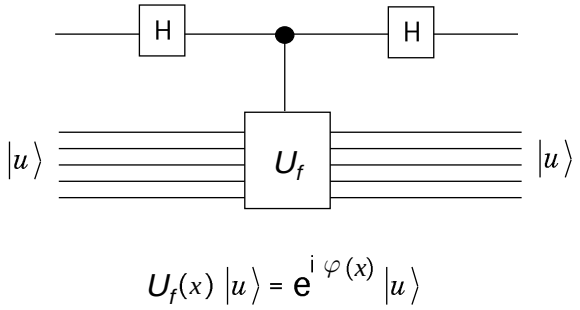

Figure 2. Network representation for the phase shift transformation of equation (1.2). Here, x is a label for the state of the first qubit.

Mach–Zehnder interferometer and, as we will illustrate in the following sections, the kernel of quantum algorithms.

The rest of the paper is organized as follows. In the next section, we discuss Deutsch's problem (1985) which shows how differentiation between interference patterns (different phase shifts) can lead to the formulation of computational problems. Then, in § 3, we review, in a unified way, generalizations of Deutsch's problem, and propose further ones. In § 4, we discuss an alternative and convenient way to view the quantum Fourier transform. In § 5, we propose an efficient method for phase estimation based on the quantum Fourier transform. In order to illustrate how some of the existing algorithms can be reformulated in terms of the multiparticle interferometry and the phase estimation problem, in § 6 we rephrase Shor's order-finding algorithm (used to factor) using the phase estimation approach. Finally, in § 7, we present a universal construction which generates any desired interference pattern with arbitrary accuracy. We summarize the conclusions in § 8.

2. Deutsch's problem

Since quantum phases in the interferometers can be introduced by some controlled-U operations, it is natural to ask whether effecting these operations can be described as an interesting computational problem. In this section, we illustrate how interference patterns lead to computational problems that are well suited to quantum computations, by presenting the first such problem that was proposed by Deutsch (1985).

To begin with, suppose that the phase shifter in the Mach–Zehnder interferometer is set either to $\phi = 0$ or to $\phi = \pi$. Can we tell the difference? Of course we can. In fact, a single instance of the experiment determines the difference: for $\phi = 0$ the particle *always* ends up in the detector '0' and for $\phi = \pi$ *always* in the detector '1'. Deutsch's problem is related to this effect.

Consider the Boolean functions f that map $\{0, 1\}$ to $\{0, 1\}$. There are exactly four such functions: two constant functions ($f(0) = f(1) = 0$ and $f(0) = f(1) = 1$) and two 'balanced' functions ($f(0) = 0$, $f(1) = 1$ and $f(0) = 1$, $f(1) = 0$). Informally, in Deutsch's problem, one is allowed to evaluate the function f *only once* and required to deduce from the result whether f is constant or balanced (in other words, whether the binary numbers $f(0)$ and $f(1)$ are the same or different). Note that we are not asked for the particular values $f(0)$ and $f(1)$ but for a global property of f. Classical intuition tells us that to determine this global property of f, we have to evaluate both $f(0)$ and $f(1)$ anyway, which involves evaluating f twice. We shall see that

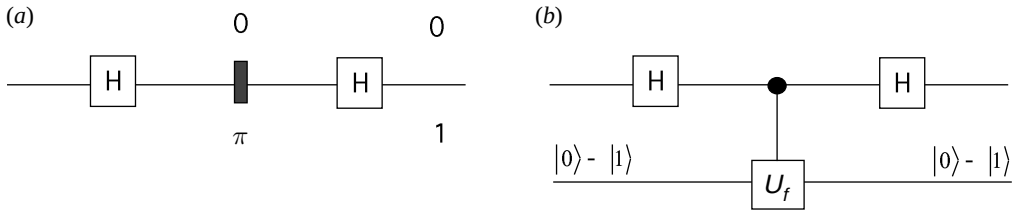

Figure 3. Network representation of Deutsch's algorithm.

this is not so in the setting of quantum information, where we can solve Deutsch's problem with a single function evaluation, by employing an algorithm that has the same mathematical structure as the Mach–Zehnder interferometer.

Let us formally define the operation of 'evaluating' f in terms of the *f-controlled-NOT* operation on two bits: the first contains the input value and the second contains the output value. If the second bit is initialized to 0, the f-controlled-NOT maps $(x, 0)$ to $(x, f(x))$. This is clearly just a formalization of the operation of computing f. In order to make the operation reversible, the mapping is defined for *all* initial settings of the two bits, taking (x, y) to $(x, y \oplus f(x))$. Note that this operation is similar to the controlled-NOT (see, for example, Barenco *et al.* 1995), except that the second bit is negated when $f(x) = 1$, rather than when $x = 1$.

If one is only allowed to perform classically the f-controlled-NOT operation once, on any input from $\{0, 1\}^2$, then it is *impossible* to distinguish between balanced and constant functions in the following sense. Whatever the outcome, both possibilities (balanced and constant) remain for f. However, if quantum mechanical superpositions are allowed, then a single evaluation of the f-controlled-NOT suffices to classify f. Our quantum algorithm that accomplishes this is best represented as the quantum network shown in figure 3*b*, where the middle operation is the f-controlled-NOT, whose semantics in quantum mechanical notation are

$$|x\rangle|y\rangle \stackrel{f-c-N}{\longrightarrow} |x\rangle|y \oplus f(x)\rangle. \tag{2.1}$$

The initial state of the qubits in the quantum network is $|0\rangle(|0\rangle - |1\rangle)$ (apart from a normalization factor, which will be omitted in the following). After the first Hadamard transform, the state of the two qubits has the form $(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$. To determine the effect of the f-controlled-NOT on this state, first note that, for each $x \in \{0, 1\}$,

$$|x\rangle(|0\rangle - |1\rangle) \stackrel{f-c-N}{\longrightarrow} |x\rangle(|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle) = (-1)^{f(x)}|x\rangle(|0\rangle - |1\rangle). \tag{2.2}$$

Therefore, the state after the f-controlled-NOT is

$$((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle)(|0\rangle - |1\rangle). \tag{2.3}$$

That is, for each x, the $|x\rangle$ term acquires a phase factor of $(-1)^{f(x)}$, which corresponds to the eigenvalue of the state of the auxiliary qubit under the action of the operator that sends $|y\rangle$ to $|y \oplus f(x)\rangle$.

This state can also be written as

$$(-1)^{f(0)}(|0\rangle + (-1)^{f(0)\oplus f(1)}|1\rangle), \tag{2.4}$$

which, after applying the second Hadamard transform, becomes

$$(-1)^{f(0)}|f(0) \oplus f(1)\rangle. \tag{2.5}$$

Therefore, the first qubit is finally in state $|0\rangle$ if the function f is constant, and in state $|1\rangle$ if the function is balanced, and a measurement of this qubit distinguishes these cases with certainty.

This algorithm is an improved version of the first quantum algorithm for this problem proposed by Deutsch (1985), which accomplishes the following. There are three possible outcomes: 'balanced', 'constant' and 'inconclusive'. For any f, the algorithm has the property that, with probability $\frac{1}{2}$, it outputs 'balanced' or 'constant' (correctly corresponding to f), and, with probability $\frac{1}{2}$, it outputs 'inconclusive' (in which case, no information is determined about f). This is a task that no classical computation can accomplish (with a single evaluation of the f-controlled-NOT gate). In comparison, our algorithm can be described as *always* producing the output 'balanced' or 'constant' (correctly). Tapp (1997, personal communication) independently discovered an algorithm for Deutsch's problem that is similar to ours.

Deutsch's result laid the foundation for the new field of quantum computation, and was followed by several other quantum algorithms for various problems, which all seem to rest on the same generic sequence: a Fourier transform, followed by an f-controlled-U, followed by another Fourier transform. (In some cases, such as Grover's 'database search' algorithm (1996), this sequence is a critical component to a larger algorithm (see Appendix B)). We illustrate this point by reviewing several of these other algorithms in the sections that follow.

3. Generalizations of Deutsch's problem

Deutsch's original problem was subsequently generalized by Deutsch & Jozsa (1992) for Boolean functions $f : \{0,1\}^n \to \{0,1\}$ in the following way. Assume that, for one of these functions, it is 'promised' that it is either constant or balanced (i.e. has an equal number of 0 outputs as 1s), and consider the goal of determining which of the two properties the function actually has.

How many evaluations of f are required to do this? Any classical algorithm for this problem would, in the worst case, require $2^{n-1}+1$ evaluations of f before determining the answer with certainty. There is a quantum algorithm that solves this problem with a single evaluation of f. The algorithm is presented in figure 4, where the control register is now composed of n qubits, all initially in state $|0\rangle$, denoted as $|00\cdots0\rangle$, and, as in the quantum algorithm for Deutsch's simple problem, an auxiliary qubit is employed, which is initially set to the state $|0\rangle - |1\rangle$ and is not altered during the computation. Also, the n-qubit Hadamard transform H is defined as

$$|x\rangle \xrightarrow{H} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle, \tag{3.1}$$

for all $x \in \{0,1\}^n$, where

$$x \cdot y = (x_1 \wedge y_1) \oplus \cdots \oplus (x_n \wedge y_n) \tag{3.2}$$

(i.e. the scalar product modulo two). This is equivalent to performing a one-qubit Hadamard transform on each of the n qubits individually. The actual computation of the function f is by means of an f-controlled-NOT gate (the middle gate in figure 4), which acts as

$$|x\rangle|y\rangle \xrightarrow{f-c-N} |x\rangle|y \oplus f(x)\rangle. \tag{3.3}$$

This is similar to equation (2.1), except that now $x \in \{0,1\}^n$.

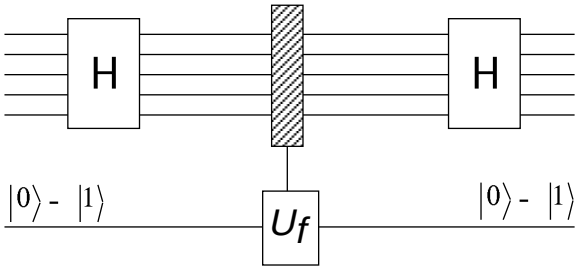

Figure 4. Network representation of Deutsch–Jozsa's and Bernstein–Vazirani's algorithms.

Stepping through the execution of the network, the state after the first n-qubit Hadamard transform is applied is

$$\sum_{x\in\{0,1\}^n} |x\rangle(|0\rangle - |1\rangle), \tag{3.4}$$

which, after the f-controlled-NOT gate, is

$$\sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle(|0\rangle - |1\rangle). \tag{3.5}$$

Finally, after the last Hadamard transform, the state is

$$\sum_{x,y\in\{0,1\}^n} (-1)^{f(x)\oplus(x\cdot y)}|y\rangle(|0\rangle - |1\rangle). \tag{3.6}$$

Note that the amplitude of $|00\cdots0\rangle$ is $\sum_{x\in\{0,1\}^n}((-1)^{f(x)})/2^n$, so if f is constant then this state is $(-1)^{f(00\cdots0)}|00\cdots0\rangle(|0\rangle - |1\rangle)$; whereas, if f is balanced then, for the state of the first n qubits, the amplitude of $|00\cdots0\rangle$ is zero. Therefore, by measuring the first n qubits, it can be determined with certainty whether f is constant or balanced. Note that, as in Deutsch's simple example, this entails a single f-controlled-NOT operation. (This is a slight improvement of Deutsch & Jozsa's original algorithm, which involves two f-controlled-NOT operations.)

Following Deutsch & Jozsa, Bernstein & Vazirani (1993) formulated a variation of the above problem that can be solved with the same network. Suppose that $f : \{0,1\}^n \to \{0,1\}$ is of the form

$$f(x) = (a_1 \wedge x_1) \oplus \cdots \oplus (a_n \wedge x_n) \oplus b = (a \cdot x) \oplus b, \tag{3.7}$$

where $a \in \{0,1\}^n$ and $b \in \{0,1\}$, and consider the goal of determining a. Note that such a function is constant if $a = 00\cdots0$ and balanced otherwise (though a balanced function need not be of this form). Furthermore, the classical determination of a requires at least n f-controlled-NOT operations (since a contains n bits of information and each classical evaluation of f yields a single bit of information). Nevertheless, by running the quantum network given in figure 4, it is possible to determine a with a single f-controlled-NOT operation.

The initial conditions are the same as above. In this case, equation (3.5) takes the simple form

$$\sum_{x\in\{0,1\}^n} (-1)^{(a\cdot x)\oplus b}|x\rangle(|0\rangle - |1\rangle), \tag{3.8}$$

which, after the final Hadamard transform, becomes

$$(-1)^b \sum_{x,y\in\{0,1\}^n} (-1)^{x\cdot(a\oplus y)}|y\rangle(|0\rangle - |1\rangle), \tag{3.9}$$

which is equivalent to $(-1)^b|a\rangle(|0\rangle - |1\rangle)$. Thus, a measurement of the control register yields the value of a. (Bernstein & Vazirani's algorithm is similar to the above, except that it employs two f-controlled-NOT operations instead of one. Also, this problem, and its solution, is very similar to the search problems considered by Terhal & Smolin (1997).)

The network construction presented in this section (figure 4) can be generalized to the case of a Boolean function $f : \{0,1\}^n \to \{0,1\}^m$ (with $m \leqslant n$), with the promise that the parity of the elements in the range of f is either constant or evenly balanced (i.e. its output values all have the same parity, or half of them have parity 0 and half have parity 1). In this case, by choosing an auxiliary register composed of m qubits, and setting all of them in the initial state $(|0\rangle - |1\rangle)$, it is possible to solve the problem with certainty in one run of the network. As in the above case, the function is constant when the n qubits of the first register are detected in state $|00\cdots0\rangle$, and evenly balanced otherwise.

A particular subclass of the above functions consists of those that are of the form $f(x) = (A\cdot x)\oplus b$, where A is an $m\times n$ binary matrix, b is a binary m-tuple and $\oplus$ is applied bitwise (this can be thought of as an affine linear function in modulo-two arithmetic). The output string of f has constant parity if $(11\cdots1)\cdot A = (00\cdots0)$ and has balanced parity otherwise. It is possible to determine all the entries of A by evaluating the function f only m times, via a suitable multiqubit f-controlled-NOT gate of the form

$$|x\rangle|y\rangle \stackrel{f-c-N}{\longrightarrow} |x\rangle|y\oplus f(x)\rangle, \tag{3.10}$$

where $x\in\{0,1\}^n$ and $y\in\{0,1\}^m$. The network described below is a generalization of that in figure 4, and determines the n-tuple $c\cdot A$, where c is any binary m-tuple. The auxiliary register is composed of m qubits, which are initialized to the state

$$(|0\rangle + (-1)^{c_1}|1\rangle)(|0\rangle + (-1)^{c_2}|1\rangle)\cdots(|0\rangle + (-1)^{c_m}|1\rangle). \tag{3.11}$$

(This state can be 'computed' by first setting the auxiliary register to the state $|c_1c_2\cdots c_m\rangle$ and then applying a Hadamard transform to it.) The n-qubit control register is initialized in state $|00\cdots0\rangle$, and then a Hadamard transform is applied to it. Then the f-controlled-NOT operation is performed, and is followed by another Hadamard transform to the control register. It is straightforward to show that the control register will then reside in the state $|c\cdot A\rangle$. By running the network m times with suitable choices for c, all the entries of A can be determined. Høyer (1997) independently solved a problem that is similar to the above, except that f is an Abelian group homomorphism, rather than an affine linear function.

4. Another look at the quantum Fourier transform

The quantum Fourier transform (QFT) on the additive group of integers modulo 2^m is the mapping

$$|a\rangle \stackrel{F_{2^m}}{\longrightarrow} \sum_{y=0}^{2^m-1} \mathrm{e}^{(2\pi\mathrm{i}ay)/2^m}|y\rangle, \tag{4.1}$$

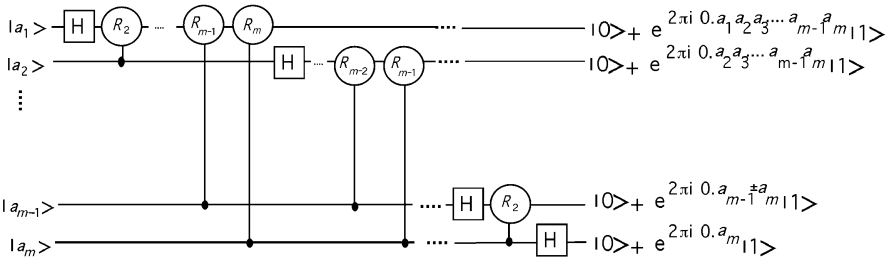

Figure 5. A network for F_{2^m} shown acting on the basis state $|a_1 a_2 \cdots a_m\rangle$. At the end, the order of the output qubits is reversed (not shown in diagram).

where $a \in \{0, \ldots, 2^m - 1\}$ (Coppersmith 1994). Let a be represented in binary as $a_1 \cdots a_m \in \{0,1\}^m$, where $a = 2^{m-1}a_1 + 2^{m-2}a_2 + \cdots + 2^1 a_{m-1} + 2^0 a_m$ (and similarly for y).

It is interesting to note that the state (4.1) is unentangled, and can in fact be factorized as

$$(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_m)}|1\rangle)(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_{m-1}a_m)}|1\rangle) \cdots (|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_1 a_2 \ldots a_m)}|1\rangle). \tag{4.2}$$

This follows from the fact that

$$\mathrm{e}^{2\pi \mathrm{i} a y/2^m}|y_1 \cdots y_m\rangle = \mathrm{e}^{2\pi \mathrm{i}(0.a_m)y_1}|y_1\rangle \mathrm{e}^{2\pi \mathrm{i}(0.a_{m-1}a_m)y_2}|y_2\rangle \cdots \mathrm{e}^{2\pi \mathrm{i}(0.a_1 a_2 \ldots a_m)y_m}|y_m\rangle, \tag{4.3}$$

so the coefficient of $|y_1 y_2 \cdots y_m\rangle$ in (4.1) matches that in (4.2).

A network for computing F_{2^n} is shown in figure 5.

In the above network, R_k denotes the unitary transformation

$$R_k = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{e}^{2\pi \mathrm{i}/2^k} \end{pmatrix}. \tag{4.4}$$

We now show that the network shown in figure 5 produces the state (4.1). The initial state is $|a\rangle = |a_1 a_2 \cdots a_m\rangle$ (and $a/2^m = 0.a_1 a_2 \ldots a_m$ in binary). Applying H to the first qubit in $|a_1 \cdots a_m\rangle$ produces the state

$$(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_1)}|1\rangle)|a_2 \cdots a_m\rangle.$$

Then applying the controlled R_2 changes the state to

$$(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_1 a_2)}|1\rangle)|a_2 \cdots a_m\rangle.$$

Next, the controlled R_3 produces

$$(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_1 a_2 a_3)}|1\rangle)|a_2 \cdots a_m\rangle,$$

and so on, until the state is

$$(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_1 \ldots a_m)}|1\rangle)|a_2 \cdots a_m\rangle.$$

The next H yields

$$(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_1 \cdots a_m)}|1\rangle)(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_2)}|1\rangle)|a_3 \cdots a_m\rangle$$

and the controlled R_2 to R_{m-1} yield

$$(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_1 \cdots a_m)}|1\rangle)(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_2 \cdots a_m)}|1\rangle)|a_3 \cdots a_m\rangle. \tag{4.5}$$

Continuing in this manner, the state eventually becomes

$$(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_1 \cdots a_m)}|1\rangle)(|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_2 \cdots a_m)}|1\rangle) \cdots (|0\rangle + \mathrm{e}^{2\pi \mathrm{i}(0.a_m)}|1\rangle),$$

which, when the order of the qubits is reversed, is state (4.2).

Note that, if we do not know $a_1 \cdots a_m$, but are given a state of the form (4.2), then $a_1 \cdots a_m$ can be easily extracted by applying the inverse of the QFT to the state, which will yield the state $|a_1 \cdots a_m\rangle$.

5. A scenario for estimating arbitrary phases

In § 1, we noted that differences in phase shifts by π can, in principle, be detected exactly by interferometry, and by quantum computations. In §§ 2 and 3, we reviewed powerful computational tasks that can be performed by quantum computers, based on the mathematical structure of detecting these phase differences. In this section, we consider the case of *arbitrary* phase differences, and show in simple terms how to obtain good estimators for them, via the quantum Fourier transform. This phase estimation plays a central role in the fast quantum algorithms for factoring and for finding discrete logarithms discovered by Shor (1994). This point has been nicely emphasized by the quantum algorithms presented by Kitaev (1995) for the Abelian stabilizer problem.

Suppose that U is any unitary transformation on n qubits and $|\psi\rangle$ is an eigenvector of U with eigenvalue $\mathrm{e}^{2\pi \mathrm{i}\phi}$, where $0 \leqslant \phi < 1$. Consider the following scenario. We do not explicitly know U or $|\psi\rangle$ or $\mathrm{e}^{2\pi \mathrm{i}\phi}$, but instead are given devices that perform controlled-U, controlled-U^{2^1}, controlled-U^{2^2} (and so on) operations. Also, assume that we are given a single preparation of the state $|\psi\rangle$. From this, our goal is to obtain an m-bit estimator of ϕ.

This can be solved as follows. First, apply the network of figure 6. This network produces the state

$$(|0\rangle + \mathrm{e}^{2\pi \mathrm{i} 2^{m-1}\phi}|1\rangle)(|0\rangle + \mathrm{e}^{2\pi \mathrm{i} 2^{m-2}\phi}|1\rangle) \cdots (|0\rangle + \mathrm{e}^{2\pi \mathrm{i}\phi}|1\rangle) = \sum_{y=0}^{2^m-1} \mathrm{e}^{2\pi \mathrm{i}\phi y}|y\rangle. \quad (5.1)$$

As noted in the last section, in the special case where $\phi = 0.a_1 \cdots a_m$, the state $|a_1 \cdots a_m\rangle$ (and hence ϕ) can be obtained by just applying the inverse of the QFT (which is the network of figure 5 in the backwards direction). This will produce the state $|a_1 \cdots a_m\rangle$ exactly (and hence ϕ).

However, ϕ is not in general a fraction of a power of two (and may not even be a rational number). For such a ϕ, it turns out that applying the inverse of the QFT produces the best m-bit approximation of ϕ with probability at least $4/\pi^2 = 0.405\ldots$. To see why this is so, let $a/2^m = 0.a_1 \cdots a_m$ be the best m-bit estimate of ϕ. Then $\phi = a/2^m + \delta$, where $0 < |\delta| \leqslant 1/2^{m+1}$. Applying the inverse QFT to state (5.1) yields the state

$$\begin{aligned} \frac{1}{2^m} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^m-1} \mathrm{e}^{(-2\pi \mathrm{i} xy)/2^m} \mathrm{e}^{2\pi \mathrm{i}\phi y}|x\rangle &= \frac{1}{2^m} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^m-1} \mathrm{e}^{(-2\pi \mathrm{i} xy)/2^m} \mathrm{e}^{2\pi \mathrm{i}(a/2^m+\delta)y}|x\rangle \\ &= \frac{1}{2^m} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^m-1} \mathrm{e}^{(2\pi \mathrm{i}(a-x)y)/2^m} \mathrm{e}^{2\pi \mathrm{i}\delta y}|x\rangle \quad (5.2) \end{aligned}$$

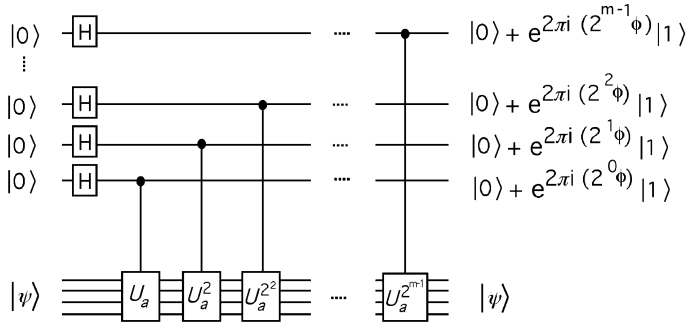

Figure 6. A network illustrating estimation of phase ϕ with m-bit precision. The same network forms the kernel of the order-finding algorithm discussed in § 6.

(for clarity, we are now including the normalization factors) and the coefficient of $|a_1 \cdots a_m\rangle$ in the above is the geometric series

$$\frac{1}{2^m} \sum_{y=0}^{2^m-1} (\mathrm{e}^{2\pi \mathrm{i}\delta})^y = \frac{1}{2^m} \left(\frac{1 - (\mathrm{e}^{2\pi \mathrm{i}\delta})^{2^m}}{1 - \mathrm{e}^{2\pi \mathrm{i}\delta}} \right). \tag{5.3}$$

Since $|\delta| \leqslant 1/2^{m+1}$, it follows that $|2\pi\delta 2^m| \leqslant \pi$, and thus $|1 - \mathrm{e}^{2\pi \mathrm{i}\delta 2^m}| \geqslant |2\pi\delta 2^m|/\frac{1}{2}\pi = |4\delta 2^m|$. Also, $|1 - \mathrm{e}^{2\pi \mathrm{i}\delta}| \leqslant |2\pi\delta|$. Therefore, the probability of observing $a_1 \cdots a_m$ when measuring the state is

$$\left| \frac{1}{2^m} \left(\frac{1 - (\mathrm{e}^{2\pi \mathrm{i}\delta})^{2^m}}{1 - \mathrm{e}^{2\pi \mathrm{i}\delta}} \right) \right|^2 \geqslant \left(\frac{1}{2^m} \left(\frac{4\delta 2^m}{2\pi\delta} \right) \right)^2 = \frac{4}{\pi^2}. \tag{5.4}$$

Note that the above algorithm (described by networks in figures 5 and 6) consists of m controlled-U^{2^k} operations, and $O(m^2)$ other operations.

In many contexts (such as that of the factoring algorithm of Shor), the above positive probability of success is sufficient to be useful; however, in other contexts, a higher probability of success may be desirable. The success probability can be amplified to $1 - \epsilon$ for any $\epsilon > 0$ by inflating m to $m' = m + O(\log(1/\epsilon))$, and rounding off the resulting m'-bit string to its most significant m bits. The details of the analysis are in Appendix C.

The above approach was motivated by the method proposed by Kitaev (1995), which involves a sequence of repetitions for each unit U^{2^j}. The estimation of ϕ can also be obtained by other methods, such as the techniques studied for optimal state estimation by Massar & Popescu (1995), Derka *et al.* (1997) and the techniques studied for use in frequency standards by Huelga *et al.* (1997). Also, it should be noted that the QFT, and its inverse, can be implemented in the fault tolerant 'semiclassical' way (see Griffiths & Niu 1996).

6. The order-finding problem

In this section, we show how the scheme from the previous section can be applied to solve the order-finding problem, where one is given positive integers a and N which are relatively prime and such that $a < N$, and the goal is to find the minimum positive integer r such that $a^r \bmod N = 1$. There is no known classical procedure for doing this in time polynomial in n, where n is the number of bits of N. Shor (1994) presented a polynomial–time quantum algorithm for this problem, and noted that,

since there is an efficient classical randomized reduction from the factoring problem to order-finding, there is a polynomial–time quantum algorithm for factoring. Also, the quantum order-finding algorithm can be used directly to break the RSA cryptosystem (see Appendix A).

Let us begin by assuming that we are also supplied with a prepared state of the form

$$|\psi_1\rangle = \sum_{j=0}^{r-1} \mathrm{e}^{(-2\pi \mathrm{i} j)/r} |a^j \bmod N\rangle. \tag{6.1}$$

Such a state is not at all trivial to fabricate; we shall see how this difficulty is circumvented later. Consider the unitary transformation U that maps $|x\rangle$ to $|ax \bmod N\rangle$. Note that $|\psi_1\rangle$ is an eigenvector of U with eigenvalue $\mathrm{e}^{2\pi \mathrm{i}(1/r)}$. Also, for any j, it is possible to implement a controlled-U^{2^j} gate in terms of $O(n^2)$ elementary gates. Thus, using the state $|\psi_1\rangle$ and the implementation of controlled-U^{2^j} gates, we can directly apply the method of § 5 to efficiently obtain an estimator of $1/r$ that has $2n$ bits of precision with high probability. This is sufficient precision to extract r.

The problem with the above method is that we are aware of no straightforward efficient method to prepare state $|\psi_1\rangle$. Let us now suppose that we have a device for the following kind of state preparation. When executed, the device produces a state of the form

$$|\psi_k\rangle = \sum_{j=0}^{r-1} \mathrm{e}^{(-2\pi \mathrm{i} k j)/r} |a^j \bmod N\rangle, \tag{6.2}$$

where k is randomly chosen (according to the uniform distribution) from $\{1, \ldots, r\}$. We shall first show that this is also sufficient to efficiently compute r, and then later address the issue of preparing such states. For each $k \in \{1, \ldots, r\}$, the eigenvalue of state $|\psi_k\rangle$ is $\mathrm{e}^{2\pi \mathrm{i}(k/r)}$, and we can again use the technique from § 5 to efficiently determine k/r with $2n$ bits of precision. From this, we can extract the quantity k/r exactly by the method of continued fractions. If k and r happen to be coprime then this yields r; otherwise, we might only obtain a divisor of r. Note that we can efficiently *verify* whether or not we happen to have obtained r by checking if $a^r \bmod N = 1$. If verification fails then the device can be used again to produce another $|\psi_k\rangle$. The expected number of random trials until k is coprime to r is $O(\log\log(N)) = O(\log n)$.

In fact, the expected number of trials for the above procedure can be improved to a constant. This is because, given any two independent trials which yield k_1/r and k_2/r, it suffices for k_1 and k_2 to be coprime to extract r (which is then the least common denominator of the two quotients). The probability that k_1 and k_2 are coprime is bounded below by

$$1 - \sum_{p \text{ prime}} \Pr[p \text{ divides } k_1] \Pr[p \text{ divides } k_2] \geqslant 1 - \sum_{p \text{ prime}} 1/p^2 > 0.54. \tag{6.3}$$

This was also noted by Knill (1995) and Shor (1995).

Now, returning to our actual setting, where we have no special devices that produce random eigenvectors, the important observation is that

$$|1\rangle = \sum_{k=1}^{r} |\psi_k\rangle, \tag{6.4}$$

and $|1\rangle$ *is* an easy state to prepare. Consider what happens if we use the previous

quantum algorithm, but with state $|1\rangle$ substituted in place of a random $|\psi_k\rangle$. In order to understand the resulting behaviour, imagine if, initially, the control register were measured with respect to the orthonormal basis consisting of $|\psi_1\rangle, \ldots, |\psi_r\rangle$. This would yield a uniform sampling of these r eigenvectors, so the algorithm would behave exactly as the previous one. Also, since this imagined measurement operation is with respect to an orthonormal set of eigenvectors of U, it commutes with all the controlled-U^{2^j} operations, and hence will have the same effect if it is performed at the *end* rather than at the beginning of the computation. Now, if the measurement were performed at the end of the computation, then it would have no effect on the outcome of the measurement of the control register. This implies that state $|1\rangle$ can in fact be used in place of a random $|\psi_k\rangle$, because the relevant information that the resulting algorithm yields is *equivalent*. This completes the description of the algorithm for the order-finding problem.

It is interesting to note that the algorithm that we have described for the order-finding problem, which follows Kitaev's methodology, results in a network (figure 6 followed by figure 5 backwards) that is identical to the network for Shor's algorithm, although the latter algorithm was derived by an apparently different methodology. The sequence of controlled-U^{2^j} operations is equivalent to the implementation (via repeated squarings) of the modular exponentiation function in Shor's algorithm. This demonstrates that Shor's algorithm, in effect, estimates the eigenvalue corresponding to an eigenstate of the operation U that maps $|x\rangle$ to $|ax \bmod N\rangle$.

7. Generating arbitrary interference patterns

We will show in this section how to generate specific interference patterns with arbitrary precision via some function evaluations. We require two registers. The first we call the control register; it contains the states we wish to interfere. The second we call the auxiliary register and it is used solely to induce relative phase changes in the first register.

Suppose the first register contains n bits. For each n-bit string $|x\rangle$, we require a unitary operator U_x. All of these operators U_x should share an eigenvector $|\Psi\rangle$ which will be the state of the auxiliary register. Suppose the eigenvalue of $|\Psi\rangle$ for x is denoted by $\mathrm{e}^{2\pi\mathrm{i}\phi(x)}$. By applying a unitary operator to the auxiliary register conditioned upon the value of the first register we will get the following interference pattern:

$$\sum_{x=0}^{2^n-1} |x\rangle|\Psi\rangle \rightarrow \sum_{x=0}^{2^n-1} |x\rangle U_x(|\Psi\rangle) = \sum_{x=0}^{2^n-1} \mathrm{e}^{2\pi\mathrm{i}\phi(x)}|x\rangle|\Psi\rangle. \tag{7.1}$$

The controlled-U_f gate that was described in § 2 can be viewed in this way. Namely, the operator $U_{f(0)}$ which maps $|y\rangle$ to $|y \oplus f(0)\rangle$ and the operator $U_{f(1)}$ which maps $|y\rangle$ to $|y \oplus f(1)\rangle$ have common eigenstate $|0\rangle - |1\rangle$. The operator $U_{f(j)}$ has eigenvalue $\mathrm{e}^{2\pi\mathrm{i}(f(j)/2)}$ for $j = 0, 1$.

In general, the family of unitary operators on m qubits which simply add a constant integer $k \bmod 2^m$ share the eigenstates

$$\sum_{y=0}^{2^m-1} \mathrm{e}^{-2\pi\mathrm{i}(ly/2^m)}|y\rangle, \quad l \in \{1, 1, \ldots, 2^m - 1\} \tag{7.2}$$

and kickback a phase change of $\mathrm{e}^{2\pi\mathrm{i}(kl/2^m)}$.

For example, suppose we wish to create the state $|0\rangle + \mathrm{e}^{2\pi \mathrm{i}\phi}|1\rangle$, where $\phi = 0.a_1a_2a_3\cdots a_m$.

We could set up an auxiliary register with m qubits and set it to the state

$$\sum_{y=0}^{2^m-1} \mathrm{e}^{-2\pi \mathrm{i}\phi y}|y\rangle. \tag{7.3}$$

By applying the identity operator when the control bit is $|0\rangle$ and the 'add $1 \bmod 2^m$' operator, U_1, when the control bit is $|1\rangle$, we see that

$$|0\rangle \sum_{y=0}^{2^m-1} \mathrm{e}^{-2\pi \mathrm{i}\phi y}|y\rangle$$

gets mapped to itself and

$$|1\rangle \sum_{y=0}^{2^m-1} \mathrm{e}^{-2\pi \mathrm{i}\phi y}|y\rangle$$

goes to

$$\begin{aligned} |1\rangle \sum_{y=0}^{2^m-1} \mathrm{e}^{-2\pi \mathrm{i}\phi y}|y+1 \bmod 2^m\rangle &= \mathrm{e}^{2\pi \mathrm{i}\phi}|1\rangle \sum_{y=0}^{2^m-1} \mathrm{e}^{-2\pi \mathrm{i}\phi (y+1)}|y+1 \bmod 2^m\rangle \\ &= \mathrm{e}^{2\pi \mathrm{i}\phi}|1\rangle \sum_{y=0}^{2^m-1} \mathrm{e}^{-2\pi \mathrm{i}\phi y}|y\rangle. \end{aligned} \tag{7.4}$$

An alternative is to set the m-bit auxiliary register to the eigenstate

$$\sum_{y=0}^{2^m-1} \mathrm{e}^{-(2\pi \mathrm{i}/2^m)y}|y\rangle \tag{7.5}$$

and conditionally apply U_ϕ which adds $a = a_1a_2\cdots a_m$ to the auxiliary register. Similarly, the state

$$|1\rangle \sum_{y=0}^{2^m-1} \mathrm{e}^{-(2\pi \mathrm{i}/2^m)y}|y\rangle$$

goes to

$$\begin{aligned} |1\rangle \sum_{y=0}^{2^m-1} \mathrm{e}^{-(2\pi \mathrm{i}/2^m)y}|y+a \bmod 2^m\rangle &= \mathrm{e}^{2\pi \mathrm{i}\phi}|1\rangle \sum_{y=0}^{2^m-1} \mathrm{e}^{-(2\pi \mathrm{i}/2^m)(y+a)}|y+a \bmod 2^m\rangle \\ &= \mathrm{e}^{2\pi \mathrm{i}\phi}|1\rangle \sum_{y=0}^{2^m-1} \mathrm{e}^{-(2\pi \mathrm{i}/2^m)y}|y\rangle. \end{aligned} \tag{7.6}$$

Similarly, if $\phi = ab/2^m$ for some integers a and b, we could also obtain the same phase 'kickback' by starting with state

$$\sum_{y=0}^{2^m-1} \mathrm{e}^{-2\pi \mathrm{i}(a/2^m)y}|y\rangle \tag{7.7}$$

and conditionally adding b to the second register.

The method using eigenstate

$$\sum_{y=0}^{2^m-1} \mathrm{e}^{-(2\pi \mathrm{i}/2^m)y}|y\rangle \tag{7.8}$$

has the advantage that we can use the same eigenstate in the auxiliary register for any ϕ. So in the case of an n-qubit control register where we want phase change $\mathrm{e}^{2\pi \mathrm{i}\phi(x)}$ for state $|x\rangle$, and if we have a reversible network for adding $\phi(x)$ to the auxiliary register when we have $|x\rangle$ in the first register, we can use it on a superposition of control inputs to produce the desired phase 'kickback' $\mathrm{e}^{2\pi \mathrm{i}\phi(x)}$ in front of $|x\rangle$. Which functions $\phi(x)$ will produce a useful result, and how to compute them, depends on the problems we seek to solve.

8. Conclusions

Various quantum algorithms, which may appear different, exhibit remarkably similar structures when they are cast within the paradigm of multiparticle interferometry. They start with a Fourier transform to prepare superpositions of classically different inputs, followed by function evaluations (i.e. f-controlled unitary transformations) which induce interference patterns (phase shifts), and are followed by another Fourier transform that brings together different computational paths with different phases. The last Fourier transform is essential to guarantee the interference of different paths.

We believe that the paradigm of estimating (or determining exactly) the eigenvalues of operators on eigenstates gives helpful insight into the nature of quantum algorithms and may prove useful in constructing new and improving existing algorithms. Other problems whose algorithms can be deconstructed in a similar manner are Simon's algorithm (1993), Shor's discrete logarithm algorithm (1994), Boneh & Lipton's algorithm (1995) and Kitaev's more general algorithm for the Abelian stabilizer problem (1995), which first highlighted this approach.

We have also shown that the evaluation of classical functions on quantum superpositions can generate arbitrary interference patterns with any prescribed precision, and have provided an explicit example of a universal construction which can accomplish this task.

We thank David Deutsch, David DiVincenzo, Ignacio Cirac and Peter Høyer for helpful discussions and comments.

This work was supported in part by the European TMR Research Network ERP-4061PL95-1412, CESG, Hewlett-Packard, The Royal Society London, the US National Science Foundation under grant no. PHY94-07194 and Canada's NSERC. Part of this work was completed during the 1997 Elsag–Bailey–I.S.I. Foundation research meeting on quantum computation.

Appendix A. Cracking RSA

What we seek is a way to compute $M \bmod N$ given M^e, e and N; that is, a method for finding eth roots in the multiplicative group of integers modulo N (this group is often denoted by $\boldsymbol{Z}_N^*$ and contains the integers coprime to N). It is still an open question whether a solution to this problem necessarily gives us a polynomial time randomized algorithm for factoring. However, factoring does give a polynomial time algorithm for finding eth roots for any e relatively prime to $\phi(N)$ and thus for cracking RSA. Knowing the prime factorization of N, say $\prod p_1^{a_1} p_2^{a_2} \cdots p_k^{a_k}$, we

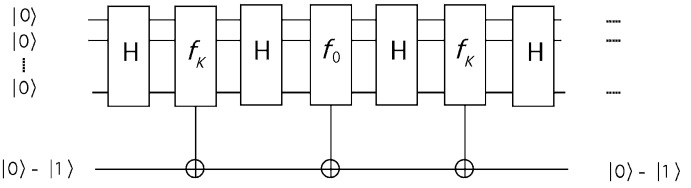

Figure 7. Network representation of Grover's algorithms. By repeating the basic sequence $2^{n/2}$ times, value k is obtained at the output with probability greater than 0.5.

can easily compute $\phi(N) = N\prod_{i=1}^{n}(1-1/p_i)$. Then we can compute d such that $ed \equiv 1 \bmod \phi(N)$, which implies $M^{ed} \equiv M \bmod N$.

However, to crack a particular instance of RSA, it suffices to find an integer d such that $ed \equiv 1 \bmod r$ where r is the order of M modulo N (i.e. the least positive integer such that $M^r \equiv 1 \bmod N$); so $ed = rk + 1$ for some integer k. We would then have $C^d \equiv M^{ed} \equiv M^{rk+1} \equiv M \bmod N$.

Since e is relatively prime to $\phi(N)$, it is easy to see that the order od M is equal to the order of $C \equiv M^e$. So given $C \equiv M^e$, we can compute r using Shor's algorithm and then compute d satisfying $de \equiv 1 \bmod \mathrm{ord}(P)$ using the extended Euclidean algorithm. Thus, we do not need several repetitions of Shor's algorithm to find the order of a for various random a; we just find the order of C and solve for M regardless of whether or not this permits us to factor N.

Appendix B. Concatenated interference

The generic sequence: a Hadamard/Fourier transform, followed by an f-controlled-U, followed by another Hadamard/Fourier transform, can be repeated several times. This can be illustrated, for example, with Grover's database search algorithm (1996). Suppose we are given (as an oracle) a function f_k which maps $\{0,1\}^n$ to $\{0,1\}$ such that $f_k(x) = \delta_{xk}$ for some k. Our task is to find k. Thus in a set of numbers from 0 to $2^n - 1$ one element has been 'tagged' and by evaluating f_k we have to find which one. To find k with a probability of 50%, any classical algorithm, be it deterministic or randomized, will need to evaluate f_k a minimum of 2^{n-1} times. In contrast, a quantum algorithm needs only $O(2^{n/2})$ evaluations. Grover's algorithm can be best presented as a network shown in figure 7.

Appendix C. Amplifying success probability when estimating phases

Let ϕ be a real number satisfying $0 \leqslant \phi < 1$ which is not a fraction with denominator 2^m, and let $a/2^m = 0.a_1a_2\cdots a_m$ be the closest m-bit approximation to ϕ so that $\phi = q/2^m + \delta$, where $0 < |\delta| \leqslant 1/2^{m+1}$. For such a ϕ, we have already shown that applying the inverse of the QFT to (5.1) and then measuring yields the state $|a\rangle$ with probability at least $4/\pi^2 = 0.405\ldots$.

Without loss of generality, assume $0 < \delta \leqslant 1/(2^m+1)$. For t satisfying $-2^{m-1} \leqslant t < 2^{m-1}$, let α_t denote the amplitude of $|a - t \bmod 2^m\rangle$. It follows from (5.2) that

$$\alpha_t = \frac{1}{2^m}\left(\frac{1-(\mathrm{e}^{2\pi\mathrm{i}(\delta+t/2^m)})^{2^m}}{1-\mathrm{e}^{2\pi\mathrm{i}(\delta+t/2^m)}}\right). \tag{C 1}$$

Since

$$|1-\mathrm{e}^{2\pi\mathrm{i}(\delta+t/2^m)}| \leqslant 2\pi(\delta+t/2^m)/\tfrac{1}{2}\pi = 4(\delta+t/2^m) \tag{C 2}$$

then

$$|\alpha_t| \leqslant \left| \frac{2}{2^m 4(\delta + t/2^m)} \right| \leqslant \frac{1}{2^{m+1}(\delta + t/2^m)}. \tag{C 3}$$

The probability of getting an error greater than $k/2^m$ is

$$\begin{aligned} \sum_{k \leqslant t < 2^{m-1}} |\alpha_t|^2 + \sum_{-2^{m-1} \leqslant t < -k} |\alpha_t|^2 &\leqslant \sum_{t=k}^{2^{m-1}-1} \frac{1}{4(t + 2^m\delta)^2} + \sum_{t=-2^{m-1}}^{-(k+1)} \frac{1}{4(t + 2^m\delta)^2} \\ &\leqslant \sum_{t=k}^{2^{m-1}-1} \frac{1}{4t^2} + \sum_{t=k+1}^{2^{m-1}} \frac{1}{4(t - \frac{1}{2})^2} \\ &\leqslant \sum_{t=2k}^{2^m-1} \frac{1}{4(\frac{1}{2}t)^2} < \int_{2k-1}^{2^m-1} \frac{1}{t^2} < \frac{1}{2k-1}. \end{aligned} \tag{C 4}$$

So, for example, if we wish to have an estimate that is within $1/2^{n+1}$ of the value ϕ with probability at least $1 - \epsilon$, it suffices to use this technique with $m = n + \lceil \log_2 (1/2\epsilon + \frac{1}{2}) \rceil$ bits.

References

Barenco, A., Bennett, C. H., Cleve, R., DiVincenzo, D. P., Margolus, N., Shor, P., Sleater, T., Smolin, J. & Weinfurter, H. 1995 *Phys. Rev.* A **52**, 3457.

Barenco, A., Deutsch, D., Ekert, A. & Jozsa, R. 1995 *Phys. Rev. Lett.* **74**, 4083.

Bernstein, E. & Vazirani, U. 1993 *Proc. 25th Ann. ACM Symp. on the Theory of Computing*, pp. 11–20. New York: ACM Press.

Boneh, D. & Lipton, R. J. 1995 Advances in cryptology. In *Proc. Crypto '95, Lecture Notes in Computer Science*, pp. 424–437. Berlin: Springer.

Coppersmith, D. 1994 An approximate Fourier transform useful in quantum factoring. IBM research report no. RC19642.

Derka, R., Bužek, V. & Ekert, A. 1997 e-print quant-ph/9707028.

Deutsch, D. 1985 *Proc. R. Soc. Lond.* A **400**, 97.

Deutsch, D. & Jozsa, R. 1992 *Proc. R. Soc. Lond.* A **439**, 553.

Ekert, A. & Jozsa, R. 1996 *Rev. Mod. Phys.* **68**, 733.

Griffiths, R. B. & Niu, C.-S. 1996 *Phys. Rev. Lett.* **76**, 3228.

Grover, L. 1996 *Proc. 28th Ann. ACM Symp. on the Theory of Computing*, p. 212. New York: ACM Press.

Høyer, P. 1997 Quantum Algorithms. Paper for qualifying exam at Odense University, Denmark.

Huelga, S. F., Macchiavello, C., Pellizzari, T., Ekert, A., Plenio, M. B. & Cirac J. I. 1997 e-print quant-ph/9707014.

Kitaev, A. 1995 Quantum measurements and the Abelian stabilizer problem. e-print quant-ph/9511026.

Knill, E. 1995 Los Alamos National Laboratory Report LAUR-95-2225 (also available at `http://www.c3.lanl.gov/ knill`).

Massar, S. & Popescu, S. 1995 *Phys. Rev. Lett.* **74**, 1259.

Shor, P. 1994 *Proc. 35th Ann. Symp. on the Foundation of Computer Science*, p. 124. Los Alamitos, CA: IEEE Computer Society.

Shor, P. 1995 e-print quant-ph/9508027.

Terhal, B. & Smolin, J. 1997 e-print quant-ph/9705041.