Contribution to Complexity

On quantum algorithms

Richard Cleve, Artur Ekert, Leah Henderson, Chiara Macchiavello and Michele Mosca

Centre for Quantum Computation
Clarendon Laboratory, University of Oxford, Oxford OX1 3PU, U.K.

Department of Computer Science
University of Calgary, Calgary, Alberta, Canada T2N 1N4

Theoretical Quantum Optics Group
Dipartimento di Fisica "A. Volta" and I.N.F.M. - Unità di Pavia
Via Bassi 6, I-27100 Pavia, Italy

Abstract

Quantum computers use the quantum interference of different computational paths to enhance correct outcomes and suppress erroneous outcomes of computations. In effect, they follow the same logical paradigm as (multi-particle) interferometers. We show how most known quantum algorithms, including quantum algorithms for factorising and counting, may be cast in this manner. Quantum searching is described as inducing a desired relative phase between two eigenvectors to yield constructive interference on the sought elements and destructive interference on the remaining terms.

1 From Interferometers to Computers

Richard Feynman [1] in his talk during the First Conference on the Physics of Computation held at MIT in 1981 observed that it appears to be impossible to simulate a general quantum evolution on a classical probabilistic computer in an *efficient* way. He pointed out that any classical simulation of quantum evolution appears to involve an exponential slowdown in time as compared to the natural evolution since the amount of information required to describe the evolving quantum state in classical terms generally grows exponentially in time. However, instead of viewing this as an obstacle, Feynman regarded it as an opportunity. If it requires so much computation to work out what will happen in a complicated multiparticle interference experiment then, he argued, the very act of setting up such an experiment and measuring the outcome is tantamount to performing a complex computation. Indeed, all quantum multiparticle interferometers *are* quantum computers and some interesting computational problems can be based on estimating internal phase shifts in these interferometers. This approach leads to a unified picture of quantum algorithms and has been recently discussed in detail by Cleve *et al.* [2].

Let us start with the textbook example of quantum interference, namely the double-slit experiment, which, in a more modern version, can be rephrased in terms of Mach-Zehnder interferometry (see Fig. 1).

A particle, say a photon, impinges on a beam-splitter (BS1), and, with some probability amplitudes, propagates via two different paths to another beam-splitter (BS2) which directs the particle to one of the two detectors. Along each path between the two beam-splitters, is a phase shifter (PS). If the lower path is labelled as state $|\,0\rangle$ and the upper one as state $|\,1\rangle$ then the particle, initially in path $|\,0\rangle$, undergoes the following sequence of transformations

$$|\,0\rangle \quad \xrightarrow{\text{BS1}} \quad \frac{1}{\sqrt{2}}(|\,0\rangle + |\,1\rangle)$$

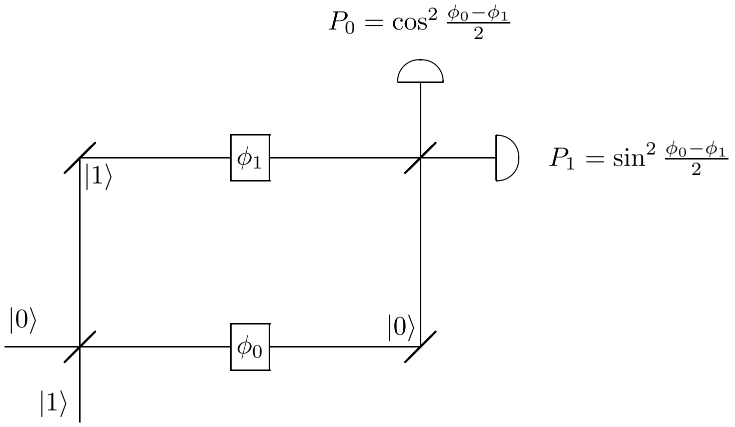

$P_0 = \cos^2 \frac{\phi_0-\phi_1}{2}$
$P_1 = \sin^2 \frac{\phi_0-\phi_1}{2}$
ϕ_1
$|1\rangle$
$|0\rangle$
ϕ_0
$|0\rangle$
$|1\rangle$

Figure 1: 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.

$$
\begin{aligned}
&\xrightarrow{\text{PS}} \quad \frac{1}{\sqrt{2}}(e^{i\phi_0}\,|\,0\rangle + e^{i\phi_1}\,|\,1\rangle) = e^{i\frac{\phi_0+\phi_1}{2}}\frac{1}{\sqrt{2}}(e^{i\frac{\phi_0-\phi_1}{2}}\,|\,0\rangle + e^{-i\frac{\phi_0-\phi_1}{2}}\,|\,1\rangle) \\
&\xrightarrow{\text{BS2}} \quad e^{i\frac{\phi_1+\phi_2}{2}}(\cos\tfrac{1}{2}(\phi_0-\phi_1)\,|\,0\rangle + i\sin\tfrac{1}{2}(\phi_0-\phi_1)\,|\,1\rangle), \qquad (1)
\end{aligned}
$$

where ϕ_0 and ϕ_1 are the settings of the two phase shifters and the action of the beam-splitters is defined as

$$
\begin{aligned}
|\,0\rangle &\longrightarrow \tfrac{1}{\sqrt{2}}(|\,0\rangle + |\,1\rangle) \\
|\,1\rangle &\longrightarrow \tfrac{1}{\sqrt{2}}(|\,0\rangle - |\,1\rangle) \qquad (2)
\end{aligned}
$$

(and extends by linearity to states of the form $\alpha\,|\,0\rangle + \beta\,|\,1\rangle$). Here, we have ignored the $e^{i\frac{\phi_0+\phi_0}{2}}$ phase shift in the reflected beam, which is irrelevant because the interference pattern depends only on the *difference* between the phase shifts in different arms of the interferometer. The phase shifters in the two paths can be tuned to effect any prescribed relative phase shift $\phi = \phi_0 - \phi_1$ and to direct the particle with probabilities $\cos^2\left(\frac{\phi}{2}\right)$ and $\sin^2\left(\frac{\phi}{2}\right)$ respectively to detectors "0" and "1".

The roles of the three key ingredients in this experiment are clear. The first beam splitter prepares a superposition of possible paths, the phase shifters modify quantum phases in different paths and the second beam-splitter combines all the paths together. As we shall see in the following sections, quantum algorithms follow this interferometry paradigm: a superposition of computational paths is prepared by the Hadamard (or the Fourier) transform, followed by a quantum function evaluation which effectively introduces phase shifts into different computational paths, followed by the Hadamard or the Fourier transform which acts somewhat in reverse to the first Hadamard/Fourier transform and combines the computational paths together. To see this, let us start by rephrasing Mach-Zehnder interferometry in terms of quantum networks.

2 Quantum gates & networks

In order to avoid references to specific technological choices (hardware), let us now describe our Mach-Zehnder interference experiment in more abstract terms. It is convenient to view this experiment as

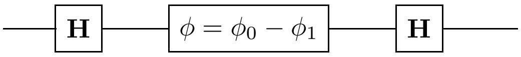


Figure 2: A quantum network composed of three single qubit gates. This network provides a hardware-independent description of any single-particle interference, including Mach-Zehnder interferometry.

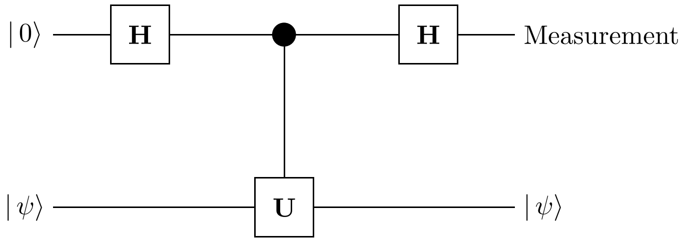


Figure 3: Phase factors can be introduced into different computational paths via the controlled-U operations. The controlled-U means that the form of U depends on the logical value of the control qubit (the upper qubit). Here, we apply the identity transformation to the auxiliary (lower) qubits (i.e. 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 auxiliary or the target qubit is initially prepared in state $|\,\psi\rangle$ which is one of the eigenstates of U.

a *quantum network* with three quantum logic gates (elementary unitary transformations) operating on a qubit (a generic two-state system with a prescribed computational basis $\{|\,0\rangle\,,|\,1\rangle\}$). The beam-splitters will be now called the Hadamard gates and the phase shifters the phase shift gates (see Fig. 2).

The Hadamard gate is the single qubit gate $\mathbf{H}$ performing the unitary transformation known as the Hadamard transform given by (Eq. 2)

$$\mathbf{H} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \qquad\qquad |\,x\rangle \; -\boxed{\mathbf{H}}- \; |\,0\rangle + (-1)^x\,|\,1\rangle \tag{3}$$

The matrix is written in the basis $\{|\,0\rangle\,,|\,1\rangle\}$ and the diagram on the right provides a schematic representation of the gate $\mathbf{H}$ acting on a qubit in state $|\,x\rangle$, with $x = 0, 1$. Using the same notation we define the phase shift gate ϕ as a single qubit gate such that $|\,0\rangle \mapsto |\,0\rangle$ and $|\,1\rangle \mapsto e^{i\phi}\,|\,1\rangle$,

$$\phi = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix} \qquad\qquad |\,x\rangle \; -\boxed{\phi}- \; e^{ix\phi}\,|\,x\rangle \;\cdot \tag{4}$$

Let us explain now how the phase shift ϕ can be "computed" with the help of an auxiliary qubit (or a set of qubits) in a prescribed state $|\,\psi\rangle$ and some controlled-U transformation where $U\,|\,\psi\rangle = e^{i\phi}\,|\,\psi\rangle$ (see Fig. 3).

Here the controlled-U is a transformation involving two qubits, where the form of U applied to the auxiliary or target qubit 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 $|\,1\rangle$. In our example shown in Fig. 3, we obtain the following sequence of transformations on the two qubits

$$
\begin{aligned}
|\,0\rangle\,|\,\psi\rangle \xrightarrow{H} \tfrac{1}{\sqrt{2}}(|\,0\rangle + |\,1\rangle)\,|\,\psi\rangle &\xrightarrow{c-U} \tfrac{1}{\sqrt{2}}(|\,0\rangle + e^{i\phi}\,|\,1\rangle)\,|\,\psi\rangle \\
&\xrightarrow{H} e^{(i\frac{\phi}{2})}(\cos\tfrac{\phi}{2}\,|\,0\rangle + i\sin\tfrac{\phi}{2}\,|\,1\rangle)\,|\,\psi\rangle\,. \qquad (5)
\end{aligned}
$$

We note that the state of the auxiliary register $|\,\psi\rangle$, being an eigenstate of U, is not altered along this network, but its eigenvalue $e^{i\phi}$ is "kicked back" in front of the $|\,1\rangle$ component in the first qubit. The sequence (5) is equivalent to the steps of the Mach-Zehnder interferometer (1) and, as was shown in [2], the kernel of most known quantum algorithms.

3 The first quantum algorithm

Since quantum phases in 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.

Suppose an experimentalist, Alice, who runs the Mach-Zehnder interferometer delegates the control of the phase shifters to her colleague, Bob. Bob is allowed to set up any value $\phi = \phi_0 - \phi_1$ and Alice's task is to estimate ϕ. Clearly for general ϕ this involves running the device several times until Alice accumulates enough data to estimate probabilities P_0 and P_1, however, if Bob promises to set up ϕ either at 0 or at π then a single-shot experiment can deliver the conclusive outcome (click in detector "0" corresponds to $\phi = 0$ and in detector "1" corresponds to $\phi = \pi$). The first quantum algorithm proposed by David Deutsch in 1985 [3] is related to this effect.

We have seen in the previous section that a controlled-U transformation can be used to produce a particular phase shift on the control qubit corresponding to its eigenvalue on the auxiliary qubit. If two eigenvalues of the controlled-U transformation lead to different orthogonal states in the control qubit, a single measurement on this qubit will suffice to distinguish the two cases.

For example 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 $f(0) = 1, f(1) = 0$). It turns out that it is possible to construct a controlled function evaluation such that two possible eigenvalues are produced which may be used to determine whether the function is constant or balanced. This is done in the following way.

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 initialised 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 $(x, y \oplus f(x))$, where $\oplus$ denotes addition modulo two.

A single evaluation of the f-controlled-NOT on quantum superpositions suffices to classify f as constant or balanced. This is the real advantage of the quantum method over the classical. Classically if the f-controlled-NOT operation may be performed only once then it is *impossible* to distinguish between balanced and constant functions. Whatever the outcome, both possibilities (balanced and constant) remain for f. This corresponds to our classical intuition about the problem since it involves determining not particular values of $f(0)$ and $f(1)$, but a global property of f. Classically to determine this global property of f, we have to evaluate both $f(0)$ and $f(1)$, which involves evaluating f twice.

Deutsch's quantum algorithm has the same mathematical structure as the Mach-Zehnder interferometer, with the two phase settings $\phi = 0, \pi$. It is best represented as the quantum network shown in Fig. 4, where the middle operation is the f-controlled-NOT, which can be defined as:

$$
|\,x\rangle\,|\,y\rangle \xrightarrow{f-c-N} |\,x\rangle\,|\,y \oplus f(x)\rangle\ . \qquad (6)
$$

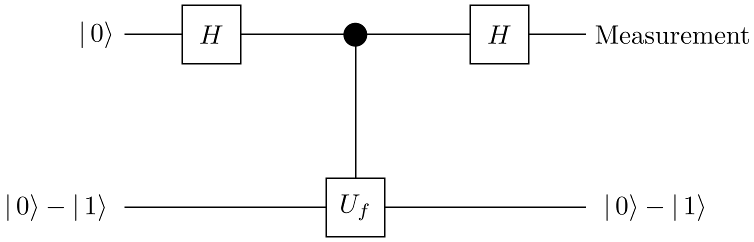


Figure 4: Quantum network which implements Deutsch's algorithm. The middle gate is the f-controlled-NOT which evaluates one of the four functions $f\,:\,\{0,1\} \mapsto \{0,1\}$. If the first qubit is measured to be $|\,0\rangle$, then the function is constant, and if $|\,1\rangle$, the function is balanced.

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 $x \in \{0,1\}$,

$$|\,x\rangle\,(|\,0\rangle - |\,1\rangle) \overset{f-c-N}{\longrightarrow} |\,x\rangle\,(|\,0 \oplus f(x)\rangle - |\,1 \oplus f(x)\rangle) = (-1)^{f(x)}\,|\,x\rangle\,(|\,0\rangle - |\,1\rangle)\;. \tag{7}$$

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

$$((-1)^{f(0)}\,|\,0\rangle + (-1)^{f(1)}\,|\,1\rangle)(|\,0\rangle - |\,1\rangle)\;. \tag{8}$$

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

$$(-1)^{f(0)}(|\,0\rangle + (-1)^{f(0)\oplus f(1)}\,|\,1\rangle)(|\,0\rangle - |\,1\rangle)\;, \tag{9}$$

which, after applying the second Hadamard transform to the first qubit, becomes

$$(-1)^{f(0)}\,|\,f(0) \oplus f(1)\rangle\,(|\,0\rangle - |\,1\rangle)\;. \tag{10}$$

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.

The Mach-Zehnder interferometer with phases ϕ_0 and ϕ_1 each set to either 0 or π can be regarded as an implementation of the above algorithm. In this case, ϕ_0 and ϕ_1 respectively encode $f(0)$ and $f(1)$ (with π representing 1), and a single photon can query both phase shifters (i.e. $f(0)$ and $f(1)$) in superposition. More recently, this algorithm (Fig. 4) has been implemented using a very different quantum physical technology, nuclear magnetic resonance [4, 5].

More general algorithms may operate not just on single qubits, as in Deutsch's case, but on sets of qubits or 'registers'. The second qubit becomes an auxiliary register $|\,\psi\rangle$ prepared in a superposition of basis states, each weighted by a different phase factor,

$$|\,\psi\rangle = \sum_{y=0}^{2^m-1} e^{-2\pi i y/2^m}\,|\,y\rangle\;. \tag{11}$$

In general, the middle gate which produces the phase shift is some controlled function evaluation. A controlled function evaluation operates on its second input, the 'target', according to the state of the first input, the ‘control’. A controlled function f applied to a control state $|\,x\rangle$, and a target state $|\,\psi\rangle$ gives

$$|\,x\rangle\,|\,\psi\rangle \longrightarrow |\,x\rangle\,|\,\psi + f(x)\rangle\,. \tag{12}$$

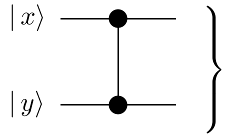
where the addition is mod 2^m. Hence for the register in state (11)

$$|\,x\rangle \sum_{y=0}^{2^m-1} e^{-2\pi i y/2^m}\,|\,y\rangle \longrightarrow e^{2\pi i f(x)/2^m}\,|\,x\rangle \sum_{y=0}^{2^m-1} e^{-2\pi i (y+f(x))/2^m}\,|\,y+f(x)\rangle = e^{2\pi i f(x)/2^m}\,|\,x\rangle\,|\,\psi\rangle\,. \tag{13}$$

Effectively a phase shift proportional to the value of $f(x)$ is produced on the first input.

We will now see how phase estimation on registers may be carried out by networks consisting of only two types of quantum gates: the Hadamard gate $\mathbf{H}$ and the conditional phase shift $\mathbf{R}(\phi)$. The conditional phase shift is the two-qubit gate $\mathbf{R}(\phi)$ defined as

$$\mathbf{R}(\phi) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{pmatrix}$$



$$e^{ixy\phi}\,|\,x\rangle\,|\,y\rangle\,. \tag{14}$$

The matrix is written in the basis $\{|\,0\rangle\,|\,0\rangle\,, |\,0\rangle\,|\,1\rangle\,, |\,1\rangle\,|\,0\rangle\,, |\,1\rangle\,|\,1\rangle\}$, (the diagram on the right shows the structure of the gate). For some of the known quantum algorithms, when working with registers, the Hadamard transformation, corresponding to the beamsplitters in the interferometer, is generalised to a quantum Fourier transform.

4 Quantum Fourier transform and computing phase shifts

The discrete Fourier transform is a unitary transformation of a s–dimensional vector

$$(f(0), f(1), f(2), \ldots, f(s-1)) \rightarrow (\tilde{f}(0), \tilde{f}(1), \tilde{f}(2), \ldots, \tilde{f}(s-1)) \tag{15}$$

defined by:

$$\tilde{f}(y) = \frac{1}{\sqrt{s}} \sum_{x=0}^{s-1} e^{2\pi i x y/s} f(x), \tag{16}$$

where $f(x)$ and $\tilde{f}(y)$ are in general complex numbers. In the following, we assume that s is a power of 2, i.e., $s = 2^n$ for some n; this is a natural choice when binary coding is used.

The quantum version of the discrete Fourier transform (QFT) is a unitary transformation which can be written in a chosen computational basis $\{|0\rangle, |1\rangle, \ldots, |2^n-1\rangle\}$ as,

$$|x\rangle \longmapsto \frac{1}{\sqrt{s}} \sum_{y=0}^{s-1} \exp(2\pi i x y/s)\,|y\rangle. \tag{17}$$

More generally, the QFT effects the discrete Fourier transform of the input amplitudes. If

$$\mathrm{QFT} : \sum_x f(x)|x\rangle \longmapsto \sum_y \tilde{f}(y)|y\rangle, \tag{18}$$

then the coefficients $\tilde{f}(y)$ are the discrete Fourier transforms of the $f(x)$’s.

A given phase $\phi_x = 2\pi x/2^n$ can be encoded by a QFT. In this process the information about ϕ_x is distributed between states of a register. Let x be represented in binary as $x_0 \ldots x_{n-1} \in \{0,1\}^n$,

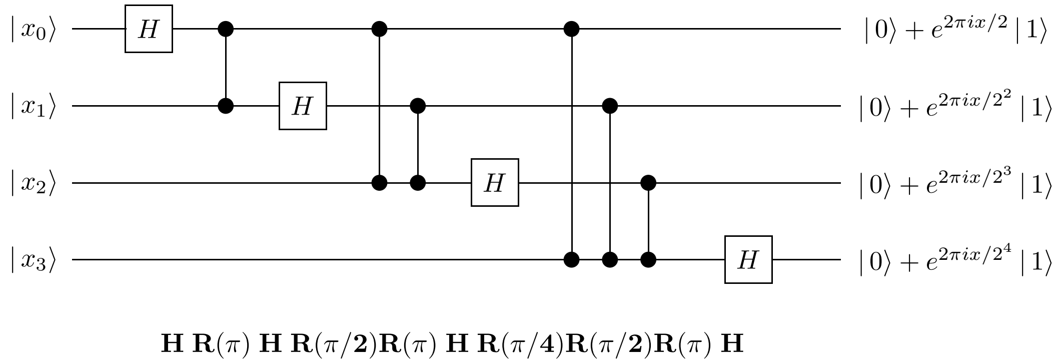

$|x_0\rangle$
$|x_1\rangle$
$|x_2\rangle$
$|x_3\rangle$
H
$|0\rangle + e^{2\pi i x/2}\,|1\rangle$
$|0\rangle + e^{2\pi i x/2^2}\,|1\rangle$
$|0\rangle + e^{2\pi i x/2^3}\,|1\rangle$
$|0\rangle + e^{2\pi i x/2^4}\,|1\rangle$
$\mathbf{H}\ \mathbf{R}(\pi)\ \mathbf{H}\ \mathbf{R}(\pi/2)\mathbf{R}(\pi)\ \mathbf{H}\ \mathbf{R}(\pi/4)\mathbf{R}(\pi/2)\mathbf{R}(\pi)\ \mathbf{H}$

Figure 5: The quantum Fourier transform (QFT) network operating on four qubits. If the input state represents number $x = \sum_k 2^k x_k$ the output state of each qubit is of the form $|\,0\rangle + e^{i2^{n-1-k}\phi_x}\,|\,1\rangle$, where $\phi_x = 2\pi x/2^n$ and $k = 0, 1, 2 \ldots n-1$. N.B. there are three different types of the $R(\phi)$ gate in the network above: $R(\pi)$, $R(\pi/2)$ and $R(\pi/4)$. The size of the rotation is indicated by the distance between the 'wires'.

where $x = \sum_{i=0}^{n-1} x_i 2^i$ (and similarly for y). An important observation is that the QFT of x, $\sum_{y=0}^{s-1} \exp(2\pi i x y/s)\,|y\rangle$, is unentangled, and can in fact be factorised as

$$(|\,0\rangle + e^{i\phi_x}\,|\,1\rangle)(|\,0\rangle + e^{i2\phi_x}\,|\,1\rangle)\cdots(|\,0\rangle + e^{i2^{n-1}\phi_x}\,|\,1\rangle)\ . \tag{19}$$

The network for performing the QFT is shown in Fig. 5. The input qubits are initially in some state $|\,x\rangle = |\,x_0\rangle\,|\,x_1\rangle\,|\,x_2\rangle\,|\,x_3\rangle$ where $x_0x_1x_2x_3$ is the binary representation of x, that is, $x = \sum_{i=0}^{3} x_i 2^i$. As the number of qubits becomes large, the rotations $R(\pi/2^n)$ will require exponential precision, which is impractical. Fortunately, the algorithm will work even if we omit the small rotations, [6, 7].

The general case of n qubits requires a simple extension of the network following the same pattern of **H** and **R** gates.

States of the form (19) are produced by function evaluation in a quantum computer. Suppose that U is any unitary transformation on m qubits and $|\,\psi\rangle$ is an eigenvector of U with eigenvalue $e^{i\phi}$. The scenario is that we do not explicitly know U or $|\,\psi\rangle$ or $e^{i\phi}$, but instead are given devices that perform controlled-U, controlled-U^{2^1}, controlled-U^{2^2} and so on until we reach controlled-$U^{2^{n-1}}$. Also, assume that we are given a single preparation of the state $|\,\psi\rangle$. From this, our goal is to obtain an n-bit estimator of ϕ.

In a quantum algorithm a quantum state of the form

$$(|\,0\rangle + e^{i2^{n-1}\phi}\,|\,1\rangle)(|\,0\rangle + e^{i2^{n-2}\phi}\,|\,1\rangle)\cdots(|\,0\rangle + e^{i\phi}\,|\,1\rangle) \tag{20}$$

is created by applying the network of Fig. 6.

Then, in the special case where $\phi = 2\pi x/2^n$, the state $|\,x_0\cdots x_{n-1}\rangle$ (and hence ϕ) can be obtained by just applying the inverse of the QFT (which is the network of Fig. 5 in the backwards direction and with the qubits in reverse order). If x is an n-bit number this will produce the state $|\,x_0\cdots x_{n-1}\rangle$ exactly (and hence the exact value 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 $\phi = 2\pi\omega$, it turns out that applying the inverse of the QFT produces the best n-bit approximation of ω with probability at least $4/\pi^2 \approx 0.41$ [2]. The probability of obtaining the best[1]

[1] Though this process produces the best estimate of ω with significant probability, it is not necessarily the best

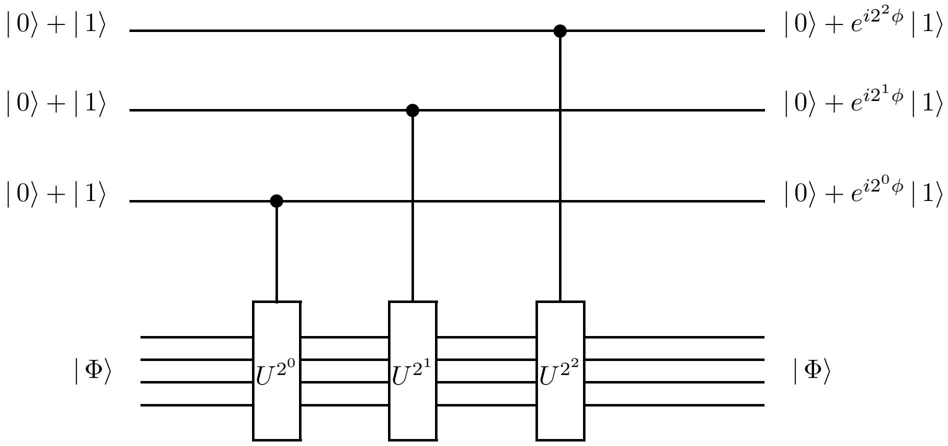


Figure 6: The network which computes phase shifts in Shor's algorithms; it also implements the modular exponentiation function via repeated squarings.

estimate can be made $1-\delta$ for any δ, $0<\delta<1$, by creating the state in equation (20) but with $n+O(\log(1/\delta))$ qubits and rounding the answer off to the nearest n bits [2].

5 Examples

We will now illustrate the general framework described in the preceding section by showing how some of the most important quantum algorithms can be viewed in this light. We start with Shor's quantum algorithm for efficient factorisation (for a comprehensive discussion of quantum factoring see [9, 10, 2]).

5.1 Quantum Factoring

Shor's quantum factoring of an integer N is based on calculating the period of the function $f(x) = a^x \bmod N$ for a randomly selected integer a between 1 and N. For any positive integer y, we define $y \bmod N$ to be the remainder (between 0 and $N-1$) when we divide y by N. More generally, $y \bmod N$ is the unique positive integer $\overline{y}$ between 0 and $N-1$ such that N evenly divides $y-\overline{y}$. For example, $2 \bmod 35 = 2$, $107 \bmod 35 = 2$, and $-3 \bmod 35 = 32$. We can test if a is relatively prime to N using the Euclidean algorithm. If it is not, we can compute the greatest common divisor of a and N using the extended Euclidean algorithm. This will factor N into two factors N_1 and N_2 (this is called *splitting* N). We can then test if N_1 and N_2 are powers of primes, and otherwise proceed to split them if they are composite. We will require at most $\log_2(N)$ splittings before we factor N into its prime factors. These techniques are summarised in [11].

It turns out that for increasing powers of a, the remainders form a repeating sequence with a period r. We can also call r the *order* of a since $a^r = 1 \bmod N$. Once r is known, factors of N are obtained by calculating the greatest common divisor of N and $a^{r/2} \pm 1$.

Suppose we want to factor 35 using this method. Let $a = 4$. For increasing x the function $4^x \bmod 35$ forms a repeating sequence $4, 16, 32, 29, 9, 1, 4, 16, 29, 32, 9, 1, \ldots$. The period is $r = 6$, and $a^{r/2} \bmod 35 = 29$. Then we take the greatest common divisor of 28 and 35, and of 30 and 35, which gives us 7 and 5, respectively, the two factors of 35. Classically, calculating r is at least as difficult as trying to factor N; the execution time of the best currently-known algorithms grows exponentially with the number of digits in N. Quantum computers can find r very efficiently.

estimator of ω, since, for example, we might be able to to obtain as close an estimate with higher probability. See [8] for details.

Consider the unitary transformation U_a that maps $|\,x\rangle$ to $|\,ax \bmod N\rangle$. Such a transformation is realised by simply implementing the reversible classical network for multiplication by a modulo N using quantum gates. The transformation U_a, like the element a, has order r, that is, $U_a^r = I$, the identity operator. Such an operator has eigenvalues of the form $e^{\frac{2\pi ik}{r}}$ for $k = 0, 1, 2, \ldots, r-1$. In order to formulate Shor's algorithm in terms of phase estimation let us apply the construction from the last section taking

$$|\,\psi\rangle = \sum_{j=0}^{r-1} e^{\frac{-2\pi ij}{r}} \left|\,a^j \bmod N\right\rangle \ . \tag{21}$$

Note that $|\,\psi\rangle$ is an eigenvector of U_a with eigenvalue $e^{2\pi i(\frac{1}{r})}$. Also, for any j, it is possible to implement efficiently a controlled-$U_a^{2^j}$ gate by a sequence of squaring (since $U_a^{2^j} = U_{a^{2^j}}$). Thus, using the state $|\,\psi\rangle$ and the implementation of controlled-$U_a^{2^j}$ gates, we can directly apply the method of the last section to efficiently obtain an estimator of $\frac{1}{r}$.

The problem with the above method is that we are aware of no straightforward efficient method to prepare state $|\,\psi\rangle$, however, let us notice that almost any state $|\,\psi_k\rangle$ of the form

$$|\,\psi_k\rangle = \sum_{j=0}^{r-1} e^{-\frac{2\pi ikj}{r}} \left|\,a^j \bmod N\right\rangle \ , \tag{22}$$

where k is from $\{0, \ldots, r-1\}$ would also do the job. For each $k \in \{0, 1, \ldots, r-1\}$, the eigenvalue of state $|\,\psi_k\rangle$ is $e^{2\pi i(\frac{k}{r})}$. We can again use the technique from the last section to efficiently determine $\frac{k}{r}$ and if k and r are coprime then this yields [2] r. Now the key observation is that

$$|\,1\rangle = \sum_{k=1}^{r} |\,\psi_k\rangle \ , \tag{23}$$

and $|\,1\rangle$ *is* an easy state to prepare.

If we substituted $|\,1\rangle$ in place of $|\,\psi\rangle$ in the last section then effectively we would be estimating one of the r, randomly chosen, eigenvalues $e^{2\pi i(\frac{k}{r})}$. This demonstrates that Shor's algorithm, in effect, estimates the eigenvalue corresponding to an eigenstate of the operation U_a that maps $|\,x\rangle$ to $|\,ax \bmod N\rangle$. A classical procedure - the continued fractions algorithm - can be employed to estimate r from these results. The value of r is then used to factorise the integer.

5.2 Finding hidden subgroups

A number of algorithms can be generalised in terms of group theory as examples of finding hidden subgroups. For any $g \in G$, the coset gK, of the subgroup K is defined as $\{gK|g \in G\}$. Say we have a function f which maps a group G to a set X, and f is constant on each coset of the subgroup K, and distinct on each coset, as illustrated in Figure 7. In other words, $f(x) = f(y)$ if and only if $x - y$ is an element of K.

In Deutsch's case, $G = \{0, 1\}$ with addition mod 2 as the group operation, and X is also $\{0, 1\}$. There are two possible subgroups K: $|\,0\rangle$, and G itself. We are given a black-box U_f for computing f

$$|\,x\rangle\,|\,y\rangle \to |\,x\rangle\,|\,y \oplus f(x)\rangle \,.$$

[2] If the estimate $y/2^m$ of k/r satisfies

$$\left|\frac{y}{2^m} - \frac{k}{r}\right| < \frac{1}{2N^2},$$

then there is a unique rational of the form $\frac{a}{b}$ with $0 < b \leq N$ satisfying

$$\left|\frac{y}{2^m} - \frac{a}{b}\right| < \frac{1}{2N^2}.$$

Consequently, $a/b = k/r$, and the continued fractions algorithm will find the fraction for us. We might be unlucky and get a k like 0, but with even 2 repetitions with random k we can find r with probability at least 0.54 [2].

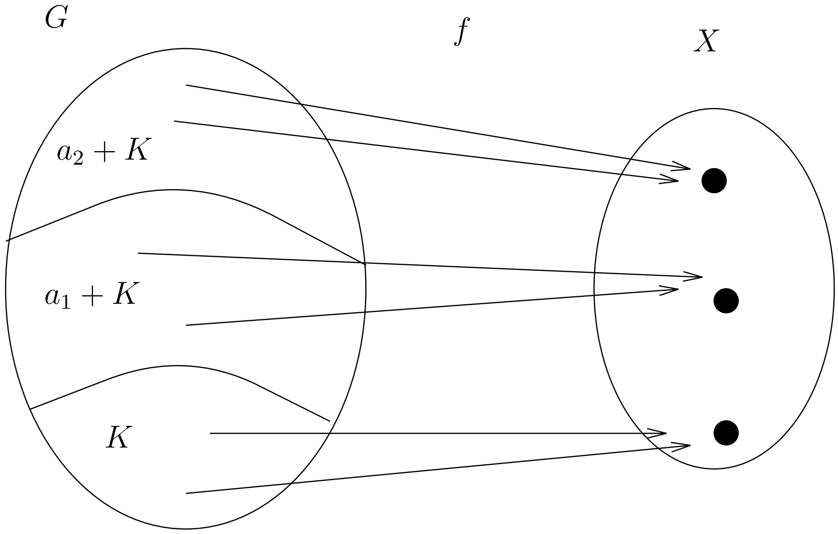


Figure 7: A function f mapping elements of a group G to a set X with a hidden subgroup K. This means that $f(g_1) = f(g_2)$ if and only if g_1 and g_2 are in the same coset of K.

There are two cosets of the subgroup $\{0\}$: $\{0\}$ and $\{1\}$. If the function is defined to be constant and distinct on each coset, it must be balanced. On the other hand, there is only one coset of the other subgroup G, the group itself. In this case the function is constant. With our specially chosen eigenvector $|\,0\rangle - |\,1\rangle$ the algorithm always outputs $|\,0\rangle$ if $K = \{0, 1\}$ (f is constant), and $|\,1\rangle$ if $K = \{0\}$, (f is balanced). Therefore we can view Deutsch's algorithm as distinguishing between the 'hidden subgroups'.

The hidden subgroup problem also encompasses the problem of finding orders of elements in a group, of which the factoring algorithm is a special case. In quantum factoring, we wish to find the order r of the element a in some group represented by X. Here G is the group of integers $\mathbf{Z}$ and K is the additive subgroup $r\mathbf{Z}$ of integer multiples of r, where r is the order of a, and a is from the multiplicative group of integers modulo N. The function f maps x to $a^x \bmod N$.

The output $|\,y\rangle$ in this case estimates an element which is orthogonal [3] to the subgroup K. The output $|\,z\rangle$ corresponds to the estimate $z/2^n$ of the eigenvalue k/r of the operator U_a which maps $|\,x\rangle$ to $|\,ax\rangle$ (that is, the operator which maps $|\,f(g)\rangle$ to $|\,f(g+1)\rangle$) on the eigenvector $|\,\psi_k\rangle$. In general, for any function f mapping a finitely generated Abelian group G to a finite set X, the quantum network shown in figure 8 will output an estimate of a random element orthogonal to the hidden subgroup K. With enough such elements, we can easily determine K using linear algebra.

By framing algorithms in terms of hidden subgroups, it may be possible to think of other problems associated with this structure in groups which we can treat with quantum algorithms. A number of algorithms have already been cast in this language, including Deutsch's problem [3, 2], Simon's problem [12], factoring integers [9], finding discrete logarithms [9], Abelian stabilisers [13], self-shift-equivalences [14], and others [15] (see [16] and [17] for details).

5.3 Quantum Counting and Searching

The first quantum algorithm for searching was constructed by Grover [18]. This has led to a large class of searching and counting algorithms.

[3] By *orthogonal* here, we are not referring to the orthogonality of states in our computational Hilbert space. When we say k/r is orthogonal to $K = r\mathbf{Z}$, we mean that $\exp(2\pi i z \frac{k}{r}) = 1$ for every $z \in K$. This notion of orthogonality generalises to groups with several generators as well.

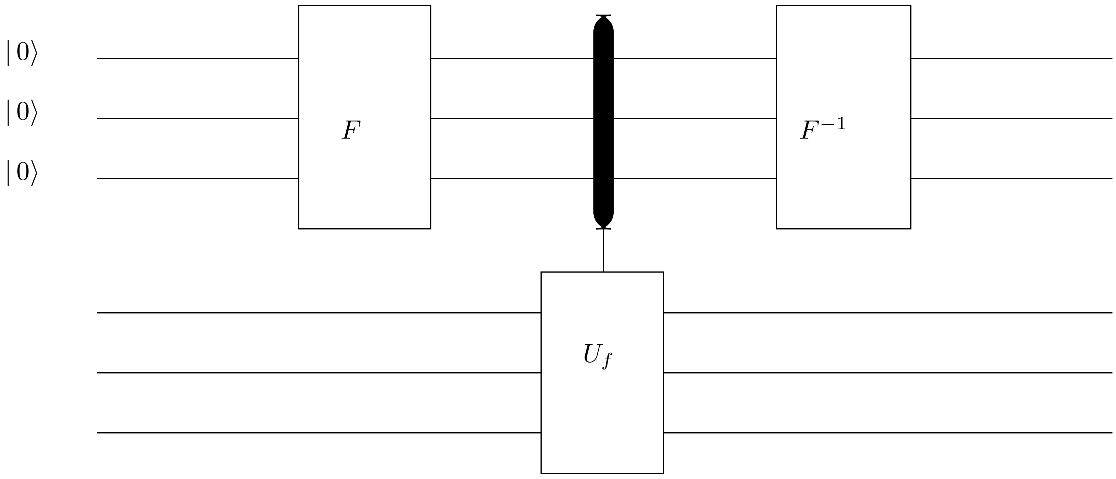


Figure 8: The generic structure of a quantum network solving any instance of the hidden subgroup problem. The first register contains tuples of integers corresponding to the Abelian group G. The role of the first Fourier transform is to create a superposition of many computational paths corresponding to different elements of G. The evaluation of the function simply kicks phases back into the control register states, and the final inverse Fourier transform produces the estimates of the eigenvalues of operators related to the function f. The set of eigenvalues corresponding to a particular eigenvector produces an element orthogonal to K. By collecting enough such orthogonal elements we can efficiently find a generating set for K.

We again consider a function f, this time mapping us from a set X to the set $\{0, 1\}$.

We might wish to decide if there is a solution to $f(x) = 1$ (the *decision* problem) , or to actually find a solution to $f(x) = 1$ (the *searching* problem). We might be more demanding and want to know how many solutions x there are to $f(x) = 1$ (*counting problem*). Small cases of the searching [19, 20] and counting [21] algorithms have been implemented using NMR technology.

In this section we will show how approximate quantum counting can easily be phrased as an instance of phase estimation, and quantum searching as an instance of inducing a desired relative phase between two eigenvectors.

In the following sections analysing quantum counting and searching, we will be considering the *Grover iterate*

$$G = -A\,U_0\,A^{-1}\,U_f \tag{24}$$

which was defined in [18] with A as the Hadamard transform. It was later generalised in [22], [23], [24] and [25] with A being any transformation such that $A\,|\,0\rangle$ contains a solution to $f(x) = 1$ with non-zero amplitude, i.e. $|\,\langle x\,|\,A\,|\,0\rangle\,|^2 > 0$ for some x with $f(x) = 1$. The operator U_f maps

$$|\,x\rangle \to -\,|\,x\rangle$$

for all x satisfying $f(x) = 1$, and the operator U_0 maps

$$|\,0\rangle \to -\,|\,0\rangle$$

leaving the remaining basis states alone. Note that this U_f is slightly different than the standard U_f which maps $|\,x\rangle\,|\,b\rangle$ to $|\,x\rangle\,|\,b \oplus f(x)\rangle$, but can be easily obtained from it by setting $|\,b\rangle$ to $|\,0\rangle - |\,1\rangle$.

5.3.1 Quantum Counting

Quantum counting was first discussed in [25], where it was observed that the Grover iterate is almost periodic with a period dependent on the number of solutions. Therefore the techniques of period-finding, as in Shor's algorithm, were applied [24]. It is also possible to think of the problem as a phase estimation (see [26]).

We simply observe that the eigenvalues[4] of G are 1, -1, $e^{2\pi i\omega_j}$, and $e^{-2\pi i\omega_j}$ where $f(x)=1$ has j solutions and

$$e^{2\pi i\omega_j} = 1 - 2j/N + 2i\sqrt{j/N - (j/N)^2}.$$

Let X_1 denote the set of solutions to $f(x)=1$, and X_0 denote the set of solutions to $f(x)=0$. Estimating ω_j (or $-\omega_j$) will give us information about the number of solutions to $f(x)=1$. For example, for small ω_j, the number of solutions, j, is roughly $N\pi^2\omega_j^2$ since $\cos(2\pi\omega_j) = 1 - 2j/N \approx 1 - 2\pi^2\omega_j^2$ for small ω_j.

We can use the techniques of the previous sections to estimate this phase ω_j provided we know how to create a starting state containing the eigenvectors with eigenvalues $e^{2\pi i\omega_j}$ and $e^{-2\pi i\omega_j}$. For non-trivial j, these eigenvectors are given by

$$|\,\psi_+\rangle = \frac{1}{\sqrt{2}}(|\,X_1\rangle + i\,|\,X_0\rangle) \tag{25}$$

$$|\,\psi_-\rangle = \frac{1}{\sqrt{2}}(|\,X_1\rangle - i\,|\,X_0\rangle) \tag{26}$$

where

$$|\,X_1\rangle = \frac{1}{\sqrt{j}}\sum_{f(x)=1}|\,x\rangle \tag{27}$$

$$|\,X_0\rangle = \frac{1}{\sqrt{N-j}}\sum_{f(x)=0}|\,x\rangle\,. \tag{28}$$

Fortunately, the starting state

$$A\,|\,0\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|\,x\rangle$$

is equal to

$$\frac{1}{\sqrt{2}}(e^{-2\pi i\theta_j}\,|\,\psi_+\rangle + e^{2\pi i\theta_j}\,|\,\psi_-\rangle) \tag{29}$$

for some real number θ_j, which is not important as far as counting is concerned, since all that is required for the phase estimation procedure is any superposition of these two eigenvectors of G.

Thus using a controlled-G, controlled-G^2, ..., and a controlled-G^{2^n}, (as done with controlled-Us in Figure 6) and applying a quantum Fourier transform, we can get an n-bit estimate of either ω_j or $-\omega_j$. This gives us an estimate of j, the number of solutions. Note that, unlike in the case of finding orders, there are in general no short-cuts for computing higher powers of G. That is, computing G^{2^n} requires 2^n repetitions of G.

Quantum algorithms for approximate counting require roughly only square root of the number of calls a classical algorithm would require.

[4] The eigenvalue -1 has multiplicity $j-1$, 1 has multiplicity $N-j-1$, and $e^{2\pi i\omega}$ and $e^{-2\pi i\omega}$ each have multiplicity 1. If $j=0$, then 1 has multiplicity N, (note that $e^{2\pi i\omega_0} = e^{-2\pi i\omega_0} = 1$), if $j=N$, then -1 has multiplicity N, ($e^{2\pi i\omega_N} = e^{-2\pi i\omega_N} = -1$).

5.3.2 Quantum searching

While estimating the number of solutions to $f(x) = 1$ is a special case of quantum phase estimation, the algorithm for searching for these solutions can be viewed as a clever use of the phase kick-back technique to induce a desired relative phase between two eigenvectors of G. The state $|\, X_1\rangle$ is a superposition of solutions to $f(x) = 1$, so it is itself a solution which it is possible for us to construct. We note that

$$|\, X_1\rangle = |\, \psi_+\rangle + |\, \psi_-\rangle \tag{30}$$

and our starting state for quantum searching is

$$A\,|\, 0\rangle = e^{-2\pi i\theta_j}\,|\, \psi_+\rangle + e^{2\pi i\theta_j}\,|\, \psi_-\rangle\,. \tag{31}$$

Each iteration of G kicks back a phase of $e^{2\pi i\omega_j}$ in front of $|\, \psi_+\rangle$ and $e^{-2\pi i\omega_j}$ in front of $|\, \psi_-\rangle$. So k iterations of G produces the state

$$A\,|\, 0\rangle = \frac{1}{\sqrt{2}}(e^{2\pi i(k\omega_j - \theta_j)})\,|\, \psi_+\rangle + e^{-2\pi i(k\omega_j - \theta_j)}\,|\, \psi_-\rangle\,. \tag{32}$$

Since we seek

$$|\, X_1\rangle = \frac{1}{\sqrt{2}}(|\, \psi_+\rangle + |\, \psi_-\rangle)$$

we want to choose the number of iterations k so that

$$k\omega_j - \theta_j \tag{33}$$

is as close to an integer as possible. When j is small, this means selecting the number of iterations close to

$$\frac{\pi}{4}\sqrt{N/j}. \tag{34}$$

Note that any classical algorithm would require N/j evaluations of f before finding a solution to $f(x) = 1$ with high probability.

6 Concluding remarks

Multi-particle interferometers can be viewed as quantum computers and any quantum algorithm follows the typical structure of a multi-particle interferometry sequence of operations. This approach seems to provide an additional insight into the nature of quantum computation and, we believe, will help to unify all quantum algorithms and relate them to different instances of quantum phase estimation.

7 Acknowledgements

This work was supported in part by the European TMR Research Network ERP-4061PL95-1412, Hewlett-Packard and Elsag-Bailey, The Royal Society, CESG and the Rhodes Trust. R.C. is partially supported by Canada's NSERC.

References

[1] R. Feynman: Simulating physics with computers. Int. J. Theor. Phys. **21**, 1982, pp. 467-488.

[2] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca: Quantum Algorithms Revisited, Proc. R. Soc. Lond. A **454**, 1998, pp. 339–354. See also LANL preprint/quant-ph/9708016.

[3] D. Deutsch: Quantum-theory, the Church-Turing principle and the universal quantum computer. Proc. R. Soc. Lond. A **400**,1985, pp. 97-117.

[4] J. Jones and M. Mosca: Implementation of a quantum algorithm on a nuclear-magnetic resonance quantum computer. J. Chem. Phys. **109**, pp. 1648-1653. See also LANL preprint quant-ph/9801027.

[5] I. Chuang, L. Vandersypen, X. Zhou, D. Leung and S. Lloyd: Experimental realisation of a quantum algorithm. Nature, **393**, 1998, pp. 143-146. See also LANL preprint quant-ph/9801037.

[6] D. Coppersmith: An Approximate Fourier Transform Useful in Quantum Factoring, IBM Research Report No. RC19642, 1994.

[7] A. Barenco, A. Ekert, K. Suominen and P. Törma: Approximate quantum Fourier-transform and decoherence. Phys. Rev. A **54**, 1996, pp. 139-146. See also LANL preprint quant-ph/9601018.

[8] W. van Dam, G. D'Ariano, A. Ekert, C. Macchiavello and M. Mosca: Estimating Phase Rotations on a Quantum Computer, preprint.

[9] P.Shor: Algorithms for quantum computation: Discrete logarithms and factoring. Proc. 35th Annual Symposium on Foundations of Computer Science, 1994, pp. 124–134. See also LANL preprint quant-ph/9508027.

[10] A. Ekert and R. Jozsa: Quantum computation and Shor's factoring algorithm, Rev. Mod. Phys. **68**, 733, 1996, pp. 733-753.

[11] A. Menezes, P. van Oorschot, and S. Vanstone: Handbook of Applied Cryptography, CRC Press, London, 1996.

[12] D. Simon: On the Power of Quantum Computation. Proc. 35th Annual Symposium on Foundations of Computer Science, 1994, pp. 116-123.

[13] A. Kitaev: Quantum measurements and the Abelian stabiliser problem. LANL preprint quant-ph/9511026, 1995.

[14] D. Grigoriev,: Testing the shift-equivalence of polynomials by deterministic, probabilistic and quantum machines. Theoretical Computer Science, **180**, 1997, pp. 217-228.

[15] D. Boneh, and R. Lipton: Quantum cryptanalysis of hidden linear functions (Extended abstract). Lecture Notes on Computer Science, **963**, 1995, pp.424-437.

[16] M. Mosca and A. Ekert: Hidden subgroups and estimation of eigenvalues on a quantum computer. To appear in the Proc. of the 1st International NASA Conference on Quantum Computing and Quantum Information Processing, Lecture Notes on Computer Science, 1998.

[17] P. Høyer: Conjugated Operators in Quantum Algorithms. preprint, 1997.

[18] L. Grover: A fast quantum mechanical algorithm for database search, Proc. 28 Annual ACM Symposium on the Theory of Computing, ACM Press New York, 1996, pp. 212-219. Journal version, "Quantum Mechanics helps in searching for a needle in a haystack", appeared in *Physical Review Letters*, **79** (1997) 325-328. See also LANL preprint quant-ph/9706033.

[19] N. Gershenfeld, I. Chuang and M. Kubinec: Experimental implementation of fast quantum searching. Phys. Rev. Lett., **80**, 1998, pp. 3408-3411.

[20] J. Jones, R. Hansen and M. Mosca: Implementation of a quantum search algorithm on a quantum computer. Nature, **393**, 1998, pp. 344-346. See also LANL preprint quant-ph/9805069.

[21] J. Jones and M. Mosca: Approximate quantum computing on an NMR ensemble quantum computer. Submitted. See LANL preprint quant-ph/quant-ph/9808056.

[22] G. Brassard and P. Høyer: An exact quantum polynomial-time algorithm for Simon's problem. Proceedings of the Fifth Israeli Symposium on Theory of Computing and Systems, IEEE Computer Society Press, 1997, pp.12-23. See also LANL preprint quant-ph/9704027.

[23] L. Grover: A framework for fast quantum mechanical algorithms. Proc. 30th Annual ACM Symposium on the Theory of Computing, 1998. See also LANL preprint quant-ph/9711043.

[24] G. Brassard, P. Høyer and A. Tapp: Quantum Counting, Proc. 25th International Colloquium on Automata, Languages and Programming, Lecture Notes on Computer Science, **1443**, pp. 820-831, 1998. See also LANL preprint quant-ph/9805082.

[25] M. Boyer, G. Brassard, P. Høyer and A. Tapp: Tight bounds on quantum searching, Proceedings of the Fourth Workshop on Physics and Computation, 1996, pp. 36-43. Forschritte Der Physik, Special issue on quantum computing and quantum cryptography, **4**, pp. 493-505, 1998. See also LANL preprint quant-ph/9605034.

[26] M. Mosca: Quantum Searching and Counting by Eigenvector Analysis. Proceedings of Randomized Algorithms, satellite workshop of MFCS '98. Available at www.eccc.uni-trier.de/eccc-local/ECCC-LectureNotes/randalg/.