Contemporary Mathematics

Quantum Algorithms In Group Theory

Michael Batty, Samuel L. Braunstein, Andrew J. Duncan, and Sarah Rees

Abstract. We present a survey of quantum algorithms, primarily for an intended audience of pure mathematicians. We place an emphasis on algorithms involving group theory.

Contents

1. Introduction

It has been known for some time that simulating quantum mechanics (on a computer based on classical mechanics) takes time which is exponential in the size of the system. This is because the total quantum state behaves as a tensor product of the individual states [**35**]. In 1982 Feynman [**20**] asked whether we could build a computer based on the principles of quantum mechanics to facilitate this task of simulation. Deutsch, in 1985, [**14**] extended the question and asked whether there are *any* problems which can be solved more efficiently on such a quantum computer. He answered the question in the affirmative, within the abstract setting of black boxes and query complexity. This was done by demonstrating a property of a black box function which requires two evaluations for its determination on a classical computer, but only one evaluation on a quantum computer. This work was generalised by Deutsch and Jozsa [**15**] in 1992 with an algorithm to distinguish between constant and balanced functions in a single evaluation. This is an exponential speed-up over the

The project is supported by EPSRC MathFIT grant GR/87406.
The second author currently holds a Royal Society – Wolfson Research Merit Award.

deterministic classical case. However, use of classical non–deterministic algorithms removes this exponential gap.

Shor's celebrated algorithm [**53**], [**54**] for factoring integers efficiently on a quantum computer gave the subject a huge boost in 1994, because of its applicability to the RSA cryptosystem. This factoring is possible due to the ability of the quantum computer to find the period of a function quickly, given that we are promised in advance that the function is periodic. It does this using the *quantum Fourier transform*, which can be constructed efficiently using a technique similar to that of the fast Fourier transform. Another well-known quantum algorithm is Grover's algorithm [**23**], which performs an unstructured search in a list of size N in time $O(\sqrt{N})$. This uses a wholly different approach to Shor's algorithm and is based on the geometric idea of rotating towards a solution within the quantum state. It is worth noting that a quantum computer can be simulated on a classical computer (albeit very slowly). So something which is classically algorithmically undecidable (for example the word problem for finitely presented groups) remains undecidable on a quantum computer. Thus we only gain improvements in efficiency: the impact of the model of quantum computing described here is on complexity theory. We note in passing that that our model of quantum computing, though the most generally accepted is not the only one that has been proposed. For example *adiabatic* quantum computing [**18**], [**19**] is based on a continuous version of the usual model, in which the evolution of the quantum system is given by its Hamiltonian which is dependent on a parameter which varies smoothly from 0 to 1. Interesting methods and questions arise from the study of this model: see [**57**].

What is interesting to us is that group theory is playing an increasingly important part in providing algorithms which are amenable to quantum computing. The modern setting for the class of quantum algorithms which use the Fourier transform (including Shor's algorithm and the Deutsch-Jozsa algorithm) is the *hidden subgroup problem*. We describe this and mention the cases where it has and has not been solved. The general case of the hidden subgroup problem (for an arbitrary finite group) is still open and is known to include the graph isomorphism problem as a special case.

The main strand of group theory in quantum computing consists of efficient quantum algorithms in finite groups, of which the hidden subgroup problem is but one. Results relating to the group non-membership problem are proved by Watrous in [**59**], where he shows that this problem lies in a quantum complexity class analogous to Babai and Moran's *Merlin-Arthur games* class MA (as defined in [**2**]). Our article culminates with another recent algorithm due to Watrous [**60**], which efficiently finds the order of a black box solvable group. This builds on Shor's algorithm but also contains essential new ingredients which seem to crucially depend on group-theoretic structure.

Sections 2,3,4 and 6 of these notes are based on a course of eight lectures given by the first named author to staff and postgraduate students at the University of Newcastle upon Tyne in the summer of 2002. He thanks all those who took part for their interest and enthusiasm. He also thanks John Watrous for useful conversations at the University of Calgary in March 2003.

Sources we found especially useful while preparing this document were [**8**] and [**49**] for a first overview of quantum computing, [**29**] and [**45**] for further depth and general background, with [**16**] and [**48**] providing the more detailed aspects of Shor's algorithm; the former for its exposition of the number-theoretic aspects in particular, and the latter for its account of implementing the quantum Fourier transform efficiently. It should be noted at this point that we have taken Shor's approach to this algorithm rather than that of Kitaev [**38**], mainly because the former came to our attention first. For the Deutsch-Jozsa algorithm, [**46**] and [**12**] provided good recent accounts. We made use of Jozsa's survey [**33**] for the hidden subgroup problem, as well as many sources quoted in Section 4.9. Watrous' work was taken from the original sources, although we have changed some of the notation to make it clearer to ourselves.

We thank the referee for careful reading of the manuscript many helpful remarks.

2. The basics of quantum computing

2.1. An overview. Before we start to describe the mathematical nuts and bolts of quantum computing it is perhaps worth describing informally what happens during a quantum computation, and how it differs from a classical computation. Later, in Section 2.3, we shall give a fuller and more formal account of our standing model of quantum computation.

A quantum system is both constrained by and enriched by the quantum mechanics which apply to the physical device used to store and manipulate data. These devices may consist for example of ion traps, optical photons or nuclear magnetic resonance systems, among others (see [**45**] for a brief account of possible technologies). At this stage the reader may start to feel anxiety through lack of knowledge of quantum physics, but in fact no physics is required to understand the computational model described in this article: such knowledge is needed only to understand where the seemingly strange rules come from. In any case we shall cover what's needed as and when required.

The memory (register) of a classical computer consists of a set of *classical bits*, each of which can be in one of two states, 0 or 1. We can therefore view the memory of an n–bit computer as the set $\mathbb{Z}_2^n$, the direct sum of n copies of $\mathbb{Z}_2 = \mathbb{Z}/2\mathbb{Z}$. The states of such a computer are binary sequences of length n, which we regard as elements of $\mathbb{Z}_2^n$. Computations then consist of sequences of functions $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$, which allow the state of the system to be transformed. For example the classical NOT gate is the function $\neg : \mathbb{Z}_2 \to \mathbb{Z}_2$ given by $\neg(i) = i + 1 \mod 2$. The final state determines the result of the computation.

The memory of a quantum computer consists of a finite dimensional complex vector space V, with an inner product (in fact a Hilbert space). A state of this quantum computer is a unit vector in V. Given a set X we denote by $\mathbb{C}X$ the complex vector space with basis the elements of X. For example $\mathbb{C}\mathbb{Z}_2$ is a 2–dimensional vector space. Corresponding to the classical n–bit computer above we have the n *quantum bit* or n*–qubit* quantum computer which has memory consisting of the 2^n–dimensional vector space $V = \mathbb{C}\mathbb{Z}_2^{\otimes n}$ (by which we mean the n–fold tensor product of $\mathbb{C}\mathbb{Z}_2$). Note that the dimension of $\mathbb{C}\mathbb{Z}_2^{\otimes n}$ is the same as the dimension of $\mathbb{C}(\mathbb{Z}_2^n)$: elements of the basis of both vector spaces are in one to one correspondence with n–tuples of elements of $\mathbb{Z}_2$. However the inner product in these spaces is not the same and we shall see, in due course, that it is the tensor product which best encapsulates the physical characteristics of quantum mechanics.

A quantum computation consists of three phases, an input phase, an evolutionary phase and a measurement or output phase. We assume that we can prepare simple input states, for example states consisting of basis vectors. Computations then consist of sequences of unitary linear transformations $\phi : V \to V$ which alter the state of the quantum computer from v to $\phi(v)$. For example, if $\mathbb{Z}_2$ has elements 0 and 1 we may identify these with basis vectors $\mathbf{e}_0$, $\mathbf{e}_1$ of the vector space $\mathbb{C}\mathbb{Z}_2$. Then, corresponding to the classical NOT gate above, we have the quantum NOT gate: the unitary transformation $\overline{\neg} : \mathbb{C}\mathbb{Z}_2 \to \mathbb{C}\mathbb{Z}_2$ given by $\overline{\neg}(\alpha\mathbf{e}_0 + \beta\mathbf{e}_1) = \alpha\mathbf{e}_1 + \beta\mathbf{e}_0$, for all $\alpha, \beta \in \mathbb{C}$. That is, $\overline{\neg}$ permutes the basis of $\mathbb{C}\mathbb{Z}_2$. Observe that if the computer is in state $(\mathbf{e}_0 - \mathbf{e}_1)/\sqrt{2}$ then one application of the quantum NOT gate results in the state $(-\mathbf{e}_0 + \mathbf{e}_1)/\sqrt{2}$. This behaviour, and its higher dimensional analogues, give rise to what is known as *quantum parallelism.*

Quantum computing also derives additional power from the existence of unitary transformations which, unlike the previous example, do not arise from permutations of the basis of the quantum system. For example there is a unitary map, of $\mathbb{C}\mathbb{Z}_2$ to itself, sending basis vectors $\mathbf{e}_0$ and $\mathbf{e}_1$ to $(\mathbf{e}_0 + \mathbf{e}_1)/\sqrt{2}$ and $(\mathbf{e}_0 - \mathbf{e}_1)/\sqrt{2}$, respectively. There is no corresponding classical transformation as the images of the basis vectors cannot be stored on a one–bit classical register.

After a computation on a classical computer the output can be observed and preserved for future use if necessary. A major difficulty in quantum computation is that the process of observation, or measurement, of the state of the quantum computer may, according to the laws of quantum mechanics, alter the state of the system at the instant of measurement. That is, measurement entails a transformation of the quantum state and the result observed is the transformed state. The process is, however, probabilistic rather than non–deterministic. For instance if the quantum state is $\alpha_1 v_1 + \cdots \alpha_m v_m$, where $\alpha_i \in \mathbb{C}$, $\{v_i : i = 1, \ldots, m\}$ is an orthonormal basis for V and $|\alpha_1|^2 + \cdots + |\alpha_m|^2 = 1$, then, under a standard measurement scheme, we observe v_i with probability $|\alpha_i|^2$. Typically in a quantum algorithm states are manipulated to change probabilities so that some characteristic of the output can be detected. In fact most quantum algorithms are Monte Carlo algorithms which have some probability of success bounded away from zero. This allows us to make the probability of failure of the algorithm arbitrarily small, by repetition.

2.2. Dirac's bra–ket notation and conventions for linear algebra. We refer the reader to [**27**] as standard reference for linear algebra and [**37**] for further details of linear operators. We consider only finite dimensional complex vector spaces equipped with inner products. A *quantum state* means a unit vector in such a space. A *superposition* means a linear combination of given vectors which is of unit length. An *amplitude* is a coefficient of a vector expressed in terms of some fixed basis. If we have a superposition of vectors in which all the (non–zero) amplitudes have equal size then we say we have a *uniform* superposition. These terms are all standard within the field of quantum computation.

The use of *ket* $|\cdot\,\rangle$ and *bra* $\langle\cdot\,|$ to denote vectors and their duals is also standard in quantum mechanics. Although we do not use it heavily in this article it is a concise and manageable notation, once it becomes familiar, so we include a description before moving on to a detailed discussion of quantum computation.

A vector, with label ψ, is written using the ket notation as $|\psi\rangle$. For instance, if X is a set then $\{|x\rangle : x \in X\}$ denotes an orthonormal basis for $\mathbb{C}X$, with the usual inner product. Also, if V is an m–dimensional inner product space then we write $\{|0\rangle, \ldots |m-1\rangle\}$ to denote an orthonormal basis of V. Our convention, which seems to be the standard in quantum mechanics, is that the inner product on V is linear in the second variable and conjugate–linear in the first variable. Using the bra–ket notation we write the inner product of $(|x\rangle, |y\rangle) \in V \times V$ as $\langle x|y\rangle$. If W is also an inner product space and T is a linear operator from V to W then $\langle x|T|y\rangle$ denotes the inner product of $(|x\rangle, T(|y\rangle))$, where $x \in W$ and $y \in V$ (and the inner product is that of W). There is an immediate pay–off to the use of this notation as follows. If V has orthonormal basis $\{|v_1\rangle, \ldots, |v_m\rangle\}$ and W has orthonormal basis $\{|w_1\rangle, \ldots, |w_n\rangle\}$ then the matrix of T with respect to these bases is easily seen to have (i,j)th entry $\langle w_i|T|v_j\rangle$.

Given T as above we write $T^\dagger$ for the *adjoint* of T: that is the unique linear operator $T^\dagger$ from W to V such that $\langle T^\dagger(|x\rangle)|y\rangle = \langle x|T|y\rangle$ (where the inner product on the left hand side is that of V). If A is the matrix of T then the matrix of $T^\dagger$ is the conjugate transpose of A and is also denoted $A^\dagger$. A linear operator T is *unitary* if $T^\dagger T = I = TT^\dagger$.

We may regard a vector $|x\rangle \in V$ as a linear operator from $\mathbb{C}$ to V, taking 1 to $|x\rangle$. Then the dual of $|x\rangle$ is $|x\rangle^\dagger$, the linear functional taking $|y\rangle$ to $\langle x|y\rangle$. Using bra notation, we write $\langle x|$ for $|x\rangle^\dagger$ and then $\langle x|\,(|y\rangle) = \langle x|y\rangle$. To reconcile this with the more familiar row and column notation for vectors and their duals, suppose that $|x\rangle$ and $|y\rangle$ are the column vectors $(1, i, 2-i)^T$ and $(2i, 1+i, 1)^T$, respectively. Then $|x\rangle^\dagger$ is $(1, -i, 2+i)$ and so

$$\langle x|\,(|y\rangle) = (1, -i, 2+i)(2i, 1+i, 1)^T = 3 + 2i = \langle (1, i, 2-i)^T, (2i, 1+i, 1)^T\rangle = \langle x|y\rangle\,.$$

Continuing in the spirit of this example, suppose V and W have orthonormal bases $\{v_1, v_2, v_3\}$ and $\{w_1, w_2\}$, respectively, with respect to which $|x\rangle = (a_1, a_2, a_3)^T$ and $|y\rangle = (b_1, b_2)^T$. Then we can evaluate the outer, or tensor, product of $|x\rangle$ and $|y\rangle$ which is given by $(a_1, a_2, a_3)^T(\overline{b}_1, \overline{b}_2)$. This is the 3×2 matrix with (i,j)th entry $a_i\overline{b}_j = \langle v_i|x\rangle \langle y|w_j\rangle$. This suggests that if V and W are vector spaces, $x \in W$ and $y \in V$, then $|x\rangle\langle y|$ be defined as the linear operator from V to W

given the rule

$$|x\rangle\langle y|\,(|v\rangle) = |x\rangle\,\langle y|v\rangle = \langle y|v\rangle\,|x\rangle\,, \ \text{ for } \ v\in V,$$

that is, $|x\rangle\langle y|$ is the tensor product of $|x\rangle$ and $|y\rangle$. In fact $|x\rangle\langle y| \in W^\dagger \otimes V$, where $W^\dagger$ denotes the dual space of W.

We may also regard $|x\rangle\langle y|$ as the bilinear map $V \times W^\dagger \to \mathbb{C}$ which sends $|v\rangle\langle w|$ to

$$\langle w|\,(|x\rangle\langle y|)\,|v\rangle = \langle w|x\rangle\,\langle y|v\rangle\,.$$

For example $\mathbb{C}\mathbb{Z}_2$ has basis $\{|0\rangle\,,|1\rangle\}$ and $|0\rangle\langle 1|$ is the linear operator sending $|1\rangle$ to $|0\rangle$ and $|0\rangle$ to the zero vector $\mathbf{0}$. If we identify $|0\rangle$ and $|1\rangle$ with $(1,0)^T$ and $(0,1)^T$, respectively, then this linear operator has matrix

$$\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

The Dirac notation for the transformation is more concise than the matrix representation. In fact the size of matrices required to describe linear maps grows exponentially in the number of qubits of the quantum computer.

As another example, the quantum NOT gate can be written as

$$|0\rangle\langle 1| + |1\rangle\langle 0|.$$

Tensor products of vector spaces are always over $\mathbb{C}$. If $|v\rangle \in V$ and $|w\rangle \in W$ then $|v\rangle \otimes |w\rangle$ may be written as either $|v\rangle\,|w\rangle$ or as $|vw\rangle$ and we use all these notations interchangeably, as convenient. Moreover this notation extends to n–fold tensor products in the obvious way. For example the tensor product $\mathbb{C}\mathbb{Z}_2 \otimes \mathbb{C}\mathbb{Z}_2$ is a 4–dimensional vector space with basis consisting of $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$. More generally $\mathbb{C}\mathbb{Z}_2^{\otimes n}$ is 2^n dimensional and has a basis consisting of $|0\cdots 0\rangle\,,\ldots,|1\cdots 1\rangle$, which we can write as $|0\rangle\,,\ldots,|2^n-1\rangle$, by identifying $(i_0\cdots i_{n-1}) \in \mathbb{Z}_2^n$ with the integer $2^{n-1}i_0 \times \cdots \times 2^0 i_{n-1}$; that is, by regarding elements of $\mathbb{Z}_2^n$ as binary expansions of integers.

Note that if M and N are complex inner product spaces then $M \otimes N$ can be made into an inner product space by defining the inner product of $a\otimes b$ with $c \otimes d$ as $\langle a\otimes b|c\otimes d\rangle = \langle a, c\rangle\langle b, d\rangle$. Thus if $x_i, y_i \in V_i$ and $|x\rangle = |x_1\cdots x_n\rangle$ and $|y\rangle = |y_1\cdots y_n\rangle$ are elements of an n–fold tensor product $\otimes_{i=1}^n V_i$, then

$$\langle x|y\rangle = \prod_{i=1}^{n} \langle x_i|y_i\rangle\,. \tag{2.1}$$

In particular if $V = \mathbb{C}\mathbb{Z}_2^{\otimes n}$ and $|x\rangle$ and $|y\rangle$ are elements of $\mathbb{Z}_2^n$ then

$$\langle x|y\rangle = \prod_{i=1}^{n} \delta_{x_i y_i}, \tag{2.2}$$

where δ_{ij} is the Kronecker delta.

Suppose that $\theta : V_1 \to V_2$ and $\phi : W_1 \to W_2$ are linear transformations. Then $\theta\otimes\phi$ is a linear transformation of $V_1 \otimes W_1 \to V_2 \otimes W_2$. Suppose that A and B are the matrices of θ and ϕ with respect to some fixed bases of V_i and W_i, $i = 1, 2$. These bases induce bases of $V_1 \otimes W_1$ and $V_2 \otimes W_2$ in a natural way: if v and w are basis vectors of V_1 and W_1, respectively, then $v\otimes w$ is a basis vector of $V \otimes W$. We adopt the convention that these natural induced bases are ordered so that the matrix $A \otimes B$ of $\theta\otimes\phi$ is the *right Kronecker product*

$$\begin{pmatrix} a_{11}B & \cdots & a_{1m}B \\ \vdots & & \vdots \\ a_{n1}B & \cdots & a_{nm}B \end{pmatrix}$$

of A and B.

2.3. The postulates of quantum mechanics. We shall now describe more thoroughly how the physical laws of quantum mechanics give rise to a model of quantum computation and establish a framework within which the theory of quantum computation may be developed. Our account is taken largely from [**45**], where further details may be found.

A quantum computer consists of a quantum mechanical system, necessarily isolated from the surrounding environment, so that its behaviour may be externally controlled and is not disturbed by events unrelated to the control procedures. The following postulates provide a model for such a system. Discussion of whether this is the best or correct model of quantum mechanics is outside the scope of this article.

Postulate 1: Quantum mechanical systems

> Associated to an isolated quantum mechanical system is a complex inner product space V. The state of the system at any time is described by a unit vector in V.

As we are concerned with computation using limited resources we consider only finite dimensional systems. The basic system we shall consider is the 2-dimensional space $\mathbb{C}\mathbb{Z}_2$ with basis $\{|0\rangle, |1\rangle\}$, known as a single *qubit* system. A state of a single qubit system is a vector $\alpha|0\rangle + \beta|1\rangle$, where $|\alpha|^2 + |\beta|^2 = 1$. If neither α nor β is zero the state is called a superposition. For example, $(|0\rangle + |1\rangle)/\sqrt{2}$ is a superposition, which is also uniform.

We may also consider the n–dimensional space $\mathbb{C}\mathbb{Z}_n$ with basis $\{|0\rangle, \dots, |n-1\rangle\}$ as a basic quantum system. However it seems likely that physical implementations of quantum computers will normally be restricted to systems built up from qubits. We defer consideration of more complex systems until after Postulate 4.

Next we consider how transformation of the system from one state to another may be realised: that is how to program the computer.

Postulate 2: Evolution

> Evolution of an isolated quantum mechanical system is described by unitary transformations. The states $|v_1\rangle$ and $|v_2\rangle$ of the system at times t_1 and t_2, respectively, are related by a unitary transformation ϕ, which depends only on t_1 and t_2, such that $\phi(|v_1\rangle) = |v_2\rangle$.

In the case of quantum computing evolution takes place at discrete intervals of time, finitely often, so evolution of the system is governed by a finite sequence of unitary transformations. In the classical setting, elementary computable functions are commonly referred to as *gates*. Analogously, certain basic unitary transformations of a complex vector space are referred to as *quantum gates*. In this article, since we do not wish to become involved in discussion of the technical details of implementation of unitary transformations on a quantum computer, we shall refer to any unitary transformation as a gate.

Postulate 3: Measurement

> A measurement of a quantum system consists of a set $\{M_m : m = 1, \dots, k\}$ of linear operators on V, such that
>
> $$\sum_{m=1}^{k} M_m^\dagger M_m = I. \tag{2.3}$$
>
> The measurement result is one of the indices m. If V is in state $|v\rangle$ then the probability that m observed is
> $$p(m) = \langle v|M_m^\dagger M_m|v\rangle = \langle M_m(|v\rangle)|M_m|v\rangle\rangle.$$
> If m is observed then the state of V is transformed from v to
> $$\frac{M_m|v\rangle}{\sqrt{p(m)}}.$$

Observe that (2.3) implies that p is a probability measure as

$$\sum_{m=1}^{k} p(m) = \sum_{m=1}^{k} \langle v|M_m^\dagger M_m|v\rangle = \langle v|I|v\rangle = 1.$$

We usually restrict attention to the special case of measurement where M_m is self–adjoint and $M_m^2 = M_m$, for all m, and $M_m M_n = 0$, when $m \neq n$. Such measurements are called *projective* measurements. In the case of projective measurement there are mutually orthogonal subspaces $P_1, \ldots, P_k$ of V such that $V = \sum_m P_m$ and $M_m = \sum_i |i\rangle\langle i|$, for some orthonormal basis $\{|i\rangle : i = 0, \ldots, d_m - 1\}$ of P_m. That is M_m is projection onto P_m. It turns out, as shown in [**45**] that, with the use of Postulates 2 and 4, measurements of the type described in Postulate 3 can be achieved using only projective measurements.

In fact most measurements we make will be of the form $\{M_m = |m\rangle\langle m| : m = 0, \ldots, n-1\}$, where V is n–dimensional and the basis used to describe the state vector and evolution of V is $\{|m\rangle : m = 0, \ldots, n-1\}$. These are called *measurements with respect to the computational basis.* In this article all measurements are taken with respect to the computational basis, unless they're explicitly defined.

For example suppose we have a single qubit system in state $Q = a|0\rangle + b|1\rangle$, where $|a|^2 + |b|^2 = 1$. If we observe Q with respect to the computational basis we obtain 0 with probability $\langle Q|0\rangle\langle 0|Q\rangle = |a|^2$ and 1 with probability $\langle Q|1\rangle\langle 1|Q\rangle = |b|^2$. The quantum system enters the state

$$\frac{|0\rangle\langle 0|Q\rangle}{|a|} = \frac{a}{|a|}|0\rangle,$$

if 0 is measured and

$$\frac{|1\rangle\langle 1|Q\rangle}{|b|} = \frac{b}{|b|}|1\rangle,$$

if 1 is measured. It turns out, as we'll see in Section 2.4, that these factors, $a/|a|$ and $b/|b|$, of modulus 1 can be ignored and we can assume that the system is either in state $|0\rangle$ or $|1\rangle$ after measurement. Note that if i was measured then further measurements in the computational basis will result in i with probability 1.

We shall write

$$p_M(|\psi\rangle \to a)$$

for the probability that a is observed when a register containing $|\psi\rangle$ is measured with a measurement M.

We shall often abuse notation by saying that $M_m|v\rangle$ is observed when we mean that m is observed, as the result of a measurement. In particular, when measuring with respect to the computational basis it's often convenient to say that $|x\rangle$, instead of x, has been observed. This abuse extends to the notation for the probability that m is observed.

We now consider how single qubit systems may be put together to build larger systems.

Postulate 4: Composite systems

> Given quantum mechanical systems associated to vector spaces V and W there is a composite quantum mechanical system associated to $V \otimes W$.

By induction this extends to composites of any finite number of systems. The composite of 2 single qubit systems is $\mathbb{C}\mathbb{Z}_2 \otimes \mathbb{C}\mathbb{Z}_2$, a 4–dimensional space with basis vectors $|00\rangle$,$|01\rangle$, $|10\rangle$,$|11\rangle$. We call this a 2*–qubit system* or 2*–qubit quantum register.* Similarly an n*–qubit system* or *quantum register* is a copy of the 2^n–dimensional space $\mathbb{C}\mathbb{Z}_2^{\otimes n}$, which, as described in Section 2.2, has basis $\{|i\rangle : i = 0, \ldots, 2^n - 1\}$. We shall by default assume that n–qubit system is equipped with this standard basis; and immediately introduce an exception. Given an m–qubit system V_m and an

n–qubit system V_n, we may form the mn–qubit system $V_m \otimes V_n$, which is naturally equipped with the basis $\{|ij\rangle : i = 0, \dots, 2^m - 1, j = 0, \dots, 2^n - 1\}$.

An n–qubit system is the quantum analogue of the classical n–bit computer. Note that whereas n–bits can contain, at any one time, one of 2^n possible values, a quantum computer can be in a superposition of all of these values, in principle in infinitely many different ways. This completes our description of quantum mechanical systems. We shall investigate some of the consequences in the next few sections.

2.4. Phase factors. Suppose that we have measurement $\{M_m : m = 1, \dots, k\}$ of an n–qubit system. The probability that m is observed when the system is in state $|x\rangle$ is then

$$\begin{aligned} p(m, |x\rangle) &= \langle x|M_m^\dagger M_m|x\rangle \\ &= \langle e^{i\theta}x|M_m^\dagger M_m|e^{i\theta}x\rangle \\ &= p(m, e^{i\theta}\,|x\rangle), \end{aligned}$$

for any real number θ. We call a complex number of modulus 1 a *phase factor*. Therefore, as far as the probabilities of what is observed are concerned, multiplication by a phase factor has no effect.

Also, $T(e^{i\theta}\,|x\rangle) = e^{i\theta}T\,|x\rangle$, for any linear transformation T. Hence, if the system starts in state $|x\rangle$ and after evolution or measurement (or both) its final state is $|y\rangle$ then we can say that starting in state $e^{i\theta}\,|x\rangle$ the final state is $e^{i\theta}\,|y\rangle$.

Therefore the introduction of the phase factor $e^{i\theta}$ is essentially invisible to the quantum computer. Consequently we regard the states $|x\rangle$ and $e^{i\theta}\,|x\rangle$ as the same state. We now see that after a measurement in the computational basis, as in Section 2.2, we may assume that the quantum system projects to a basis vector of the system (with coefficient 1).

Note that what we have said about phase factors applies only to scalar multiples of the entire unit vector which comprises the state. Altering individual coefficients of a state by a phase factor may indeed change the state. For example $i(|0\rangle + |1\rangle)/\sqrt{2}$ and $(|0\rangle + |1\rangle)/\sqrt{2}$ both correspond to the same state of a quantum system but are not the same as $(|0\rangle + i\,|1\rangle)/\sqrt{2}$.

2.5. Multiple measurements. We'll often have cause to employ more than one measurement as part of a single algorithm. One obvious question is whether or not applying first one measurement and then a second is equivalent to applying a single measurement. The answer is yes, and the single measurement is the obvious one, as we shall now see. Let $M = \{M_m : m = 1, \dots, k\}$ and $N = \{N_n : n = 1, \dots, l\}$ be measurements. Then, from the definition of measurement, we have

$$\sum_{m,n} (M_m N_n)^\dagger (M_m N_n) = \sum_{m,n} N_n^\dagger M_m^\dagger M_m N_n = \sum_n N_n^\dagger (\sum_m M_m^\dagger M_m) N_n = \sum_n N_n^\dagger I N_n = I.$$

Therefore $L = \{L_{(n,m)} = M_m N_n : m = 1, \dots, k, n = 1, \dots, l\}$ is a measurement.

We claim that measurement using N followed by measurement using M is equivalent to measurement using L. Suppose then that our system is in state $|x\rangle$ and that we first measure using N. Then the probability of observing n is

$$p_N(n) = \langle x|N_n^\dagger N_n|x\rangle$$

and, if n is observed, the system will then be in state

$$|y\rangle = \frac{N_n\,|x\rangle}{\sqrt{p_N(n)}}.$$

Now, with the system in state $|y\rangle$, making a measurement using M the probability of observing m is

$$p_M(m) = \langle y|M_m^\dagger M_m|y\rangle = \frac{1}{p_N(n)}\langle x|N_n^\dagger M_m^\dagger M_m N_n|x\rangle$$

and, if m is observed, the system will be in state

$$|z\rangle = \frac{M_m|y\rangle}{\sqrt{p_M(m)}} = \frac{M_m N_n|x\rangle}{\sqrt{p_M(m)}\sqrt{p_N(n)}}.$$

Hence the probability of measuring n and then m, or (n, m), is

$$p_N(n)p_M(m) = \langle x|N_n^\dagger M_m^\dagger M_m N_n|x\rangle$$

which is, by definition, the probability $p_L(n, m)$ of observing (n, m) using L. Also, if (n, m) is observed using L then the system will be in state

$$\frac{M_m N_n|x\rangle}{\sqrt{p_L(n,m)}} = |z\rangle.$$

Therefore measurement using L is equivalent to measurement using N then M.

2.6. Distinguishing states. In classical computation it is reasonable to assume that if we have a register which may take one of two distinct values then we can tell which of these values the register contains. One consequence of the measurement postulate is that this is not always the case in quantum computation. In fact we can tell apart orthogonal states but we cannot tell apart states which are not orthogonal.

Following [**45**], to see that we can't distinguish non–orthogonal states, assume that we have a quantum system which we know contains either the state $|\psi_1\rangle$ or the state $|\psi_2\rangle$, and that we know what both of these states are. If $|\psi_1\rangle$ and $|\psi_2\rangle$ are not orthogonal then we can write $|\psi_2\rangle = \alpha|\psi_1\rangle + \beta|\theta\rangle$, for some unit vector $|\theta\rangle$ orthonormal to $|\psi_1\rangle$ and $\alpha, \beta \in \mathbb{C}$, with $\alpha \neq 0$ and so $|\beta| < 1$. Now assume that we have a measurement $\{M_m : m = 1, \ldots, k\}$ with which we can distinguish between $|\psi_1\rangle$ and $|\psi_2\rangle$. That is to say we have sets S_1 and S_2 such that $\{1, \ldots, k\} = S_1 \cup S_2$, where $S_1 \cap S_2 = \emptyset$ and we observe $m \in S_i$ if and only if the system is in state $|\psi_i\rangle$ (before the measurement).

If we define $E_i = \sum_{m\in S_i} M_m^\dagger M_m$, for $i = 1, 2$, then

$$\langle\psi_i|E_i|\psi_i\rangle = 1, \text{ for } i = 1, 2, \tag{2.4}$$

since the probability of observing $m \in S_i$ is 1 if the system is in state $|\psi_i\rangle$. Also, $E_1 + E_2 = I$ so $\langle\psi_i|E_1 + E_2|\psi_i\rangle = 1$, for $i = 1, 2$, which implies that

$$\langle\psi_1|E_2|\psi_1\rangle = 0 = \langle\psi_2|E_1|\psi_2\rangle.$$

As $M_m^\dagger M_m$ is a positive operator (see [**37**], for example) so is E_i and so $\sqrt{E_i}$ is defined (and is a self–adjoint). We now have

$$\langle\psi_1|E_2|\psi_1\rangle = \left\langle\psi_1|\sqrt{E_2}^\dagger\sqrt{E_2}|\psi_1\right\rangle = 0,$$

so $\sqrt{E_2}|\psi_1\rangle = 0$. This implies that $\sqrt{E_2}|\psi_2\rangle = \sqrt{E_2}(\alpha|\psi_1\rangle + \beta|\theta\rangle) = \beta\sqrt{E_2}|\theta\rangle$ so

$$\langle\psi_2|E_2|\psi_2\rangle = |\beta|^2\langle\theta|E_2|\theta\rangle \leq |\beta|^2 < 1,$$

contrary to (2.4) (where the first inequality follows from (2.3).)

The conclusion is that there is no such measurement. On the other hand, suppose we start with known states $|\psi_1\rangle, \ldots, |\psi_k\rangle$ which are pairwise orthogonal. To see that we can distinguish between them define a measurement $\{M_m : m = 0, \ldots, k\}$, where $M_m = |\psi_i\rangle\langle\psi_i|$, $m \geq 1$, and $M_0 = I - \sum_{m=1}^k M_m$. Then, if the system is in state $|\psi_i\rangle$ the probability of measuring i is $\left\langle\psi_i|M_i^\dagger M_i|\psi_i\right\rangle = 1$. Thus, given that we know the system is in one of these k states, this measurement allows us to determine which one.

2.7. No cloning. Many classical algorithms make use of a copy function which replicates, or clones, the contents of a given register, whatever they happen to be. This can be achieved using the classical conditional not gate, CNOT, as follows. Suppose we have a 1–bit register, that is a copy of $\mathbb{Z}_2$, and we wish to copy its contents. Define the function CNOT from $\mathbb{Z}_2 \bigoplus \mathbb{Z}_2$ to itself by $\mathrm{CNOT}(x, y) = (x, x \oplus y)$, where $\oplus$ denotes addition modulo 2. To our 1–bit register we adjoin a second 1–bit register in state 0: so if the first register is in state x we now have a 2-bit register, that is $\mathbb{Z}_2 \bigoplus \mathbb{Z}_2$, in state $(x, 0)$. Applying the conditional not function to this register we obtain

$$\mathrm{CNOT}(x, 0) = (x, x).$$

Both the first and second 1–bit registers now contain the original contents of the first register. This process easily generalises to allow copying of n–bit registers.

The question is whether we can do the same thing on a quantum computer. This would mean, given an n–qubit register V, finding some fixed state $|s\rangle$ of V and a unitary transformation U of $V \otimes V$ such that

$$U\,|x\rangle\,|s\rangle = |x\rangle\,|x\rangle\,, \text{ for all } x \in V. \tag{2.5}$$

It turns out that, as a consequence of the following lemma, this is impossible.

LEMMA 2.1. *Let $|x_1\rangle$ and $|x_2\rangle$ be distinct unit vectors in V and let T be a unitary transformation of $V \otimes V$. If there is a unit vector $|s\rangle \in V$ such that $T\,|x_i\rangle\,|s\rangle = |x_i\rangle\,|x_i\rangle$, for $i = 1$ and 2, then $|x_1\rangle$ and $|x_2\rangle$ are orthogonal.*

PROOF. Since T is unitary we have

$$\langle x_1 s|x_2 s\rangle = \left\langle x_1 s|T^\dagger T|x_2 s\right\rangle = \langle x_1 x_1|x_2 x_2\rangle\,.$$

That is, using (2.1),

$$\langle x_1|x_2\rangle = \langle x_1|x_2\rangle\,\langle s|s\rangle = (\langle x_1|x_2\rangle)^2,$$

so $\langle x_1|x_2\rangle = 0$ or 1. As $|x_1\rangle \neq |x_2\rangle$, and $|x_1\rangle$ and $|x_2\rangle$ are unit vectors, it follows from the Cauchy–Schwarz inequality that $|x_1\rangle$ and $|x_2\rangle$ are orthogonal, as required. □

Now let U be a unitary transformation of $V \otimes V$ and, for a fixed $|s\rangle \in V$, set $R = \{|x\rangle \in V : U\,|x\rangle\,|s\rangle = |x\rangle\,|x\rangle\}$. The dimension of V is 2^n so it follows, from the lemma above, that $|R| \leq 2^n$. This applies to any unitary transformation U and any $|s\rangle \in V$, so there can be no U satisfying (2.5). Moreover we can copy at most 2^n fixed, predetermined states, but they must be orthonormal. In fact, given a set $\{|x_i\rangle : i = 1, \ldots, 2^n\}$ of 2^n orthonormal vectors of V, we have a set $\{|x_i\rangle\,|s\rangle : i = 1, \ldots, 2^n\}$ of 2^n orthonormal vectors of $V \otimes V$. We can extend this set to a basis of $V \otimes V$. Since $\{|x_i\rangle\,|x_i\rangle : i = 1, \ldots, 2^n\}$ can also be extended to a basis of $V \otimes V$, we may define a unitary transformation of $V \otimes V$ which maps $|x_i\rangle\,|s\rangle$ to $|x_i\rangle\,|x_i\rangle$, for all i. Hence we may copy up to 2^n fixed orthonormal states.

In a classical probabilistic algorithm we may repeat a calculation a fixed number of times (in order to have a high probability of success) by first copying the input and then performing the calculation on each copy. However on a quantum computer, due to the fact that we cannot copy arbitrary states, to repeat the calculation we must repeat the entire algorithm.

2.8. Entangled states. Let $x \in V \otimes W$. Then x is said to be *disentangled* if there exist $v \in V$ and $w \in W$ such that $x = v \otimes w$. Otherwise x is said to be *entangled*. The definition extends in the obvious way to n-fold tensor products. An entangled unit vector of an n–qubit system is called an *entangled state*. For example, $|00\rangle + |11\rangle$ is an entangled state. In fact, if there exist $a, b, c, d \in \mathbb{C}$ such that

$$\begin{aligned} |00\rangle + |11\rangle &= (a\,|0\rangle + b\,|1\rangle) \otimes (c\,|0\rangle + d\,|1\rangle) \\ &= ac|00\rangle + bc|10\rangle + ad|01\rangle + bd|11\rangle, \end{aligned}$$

then $bc = ad = 0$ and $ac = bd = 1$, which is impossible.

Entangled states play an important role in quantum teleportation and in super–dense coding. For details of these applications see, for example, [**45**] or [**9**].

2.9. Observing multiple qubit systems. A quantum algorithm may use different parts of its register for different purposes and so it is often convenient to view a quantum system as a composite of sub–systems. This amounts to decomposing the vector space comprising the system into a tensor product of vector sub–spaces. The purpose of doing this is usually so that measurement may be taken on one part of the system but not the other. We now consider how this may be arranged.

Suppose that the mn–qubit system V is the tensor product of m and n–qubit systems Q and R, respectively. Assume that we have measurements $M = \{M_a : a = 1, \dots, k\}$ and $N = \{N_b : b = 1, \dots, l\}$ of Q and R. Then we may define measurements $\hat{M} = \{\hat{M}_a = M_a \otimes I : a = 1, \dots, k\}$ and $\hat{N} = \{\hat{N}_b = I \otimes N_b : b = 1, \dots, l\}$ of $Q \otimes R$. As in Section 2.5 we obtain a measurement $L_{ab} = \{\hat{M}_a \hat{N}_b : a = 1, \dots, k, b = 1, \dots l\}$. This time we have $\hat{M}_a \hat{N}_b = (M_a \otimes I)(I \otimes N_b) = (I \otimes N_b)(M_a \otimes I) = \hat{N}_b \hat{M}_a$, for all a, b. Therefore, as in Section 2.5 measurement with $\hat{N}$ followed by $\hat{M}$ is equivalent to measurement with L and, in addition, the same is true of measurement with $\hat{M}$ followed by $\hat{N}$. We call a measurement $\hat{M}$ induced in this way from a measurement on Q *a measurement of register* Q and say that a is *observed in register* Q.

Example 2.2. Suppose that we have a 2–qubit system $V = \mathbb{C}\mathbb{Z}_2 \otimes \mathbb{C}\mathbb{Z}_2$ and $Q = R = \mathbb{C}\mathbb{Z}_2$ and we have measurements $M = \{M_0, M_1\}$ and $N = \{N_0, N_1\}$, where $M_i = N_i = |i\rangle\langle i|$. Then $\hat{M}_i = |i\rangle\langle i| \otimes I$, $\hat{N}_j = I \otimes |j\rangle\langle j|$ and $L_{ij} = |i\rangle\langle i| \otimes |j\rangle\langle j|$. Consider the state

$$|\psi\rangle = a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle.$$

where $|a|^2 + |b|^2 + |c|^2 + |d|^2 = 1$. To see what happens if we measure with $\hat{M}$ we may rewrite $|\psi\rangle$ as

$$|0\rangle \otimes (a\,|0\rangle + b\,|1\rangle) + |1\rangle \otimes (c\,|0\rangle + d\,|1\rangle).$$

Then it's clear that $\hat{M}_0\,|\psi\rangle = |0\rangle \otimes (a\,|0\rangle + b\,|1\rangle)$ and $\hat{M}_1\,|\psi\rangle = |1\rangle \otimes (c\,|0\rangle + d\,|1\rangle)$. Hence

$$p_{\hat{M}}(|\psi\rangle \to 0) = |a|^2 + |b|^2 = u^2 \text{ and } p_{\hat{M}}(|\psi\rangle \to 1) = |c|^2 + |d|^2 = v^2,$$

where $u, v \geq 0$.

Observations of 0 or 1 in register Q will therefore result in the quantum state

$$|0\rangle \otimes \left(\frac{a}{u}\,|0\rangle + \frac{b}{u}\,|1\rangle\right) \text{ or } |1\rangle \otimes \left(\frac{c}{v}\,|0\rangle + \frac{d}{v}\,|1\rangle\right) \text{ respectively.}$$

Similarly, if $w, x \geq 0$ are such that $w^2 = |a|^2 + |c|^2$ and $x^2 = |b|^2 + |d|^2$ then measuring V with $\hat{N}$ the probabilities of observing 0 and 1 in register R are

$$p_{\hat{N}}(|\psi\rangle \to 0) = w^2 \text{ and } p_{\hat{N}}(|\psi\rangle \to 1) = x^2.$$

If 0 or 1 is observed in register R then the resulting quantum state is

$$\left(\frac{a}{w}\,|0\rangle + \frac{c}{w}\,|1\rangle\right) \otimes |0\rangle \text{ or } \left(\frac{b}{x}\,|0\rangle + \frac{d}{x}\,|1\rangle\right) \otimes |1\rangle \text{ respectively.}$$

We now consider the effect of measurements in registers Q and R on a disentangled state $|xy\rangle = |x\rangle \otimes |y\rangle$, where $|x\rangle \in Q$ and $|y\rangle \in R$. We shall see that in this case the probabilities of observing a in register Q or b in register R are independent. We have $\hat{M}_a^\dagger \hat{M}_a = M_a^\dagger M_a \otimes I$ so the probability of observing a if we measure V with $\hat{M}$ is

$$p_{\hat{M}}(|xy\rangle \to a) = \left\langle xy|\hat{M}_a^\dagger \hat{M}_a|xy\right\rangle = \left\langle x \otimes y|M_a^\dagger M_a \otimes I|x \otimes y\right\rangle = \left\langle x|M_a^\dagger M_a|x\right\rangle \langle y|y\rangle\,.$$

Since $|xy\rangle$ is a unit vector we may assume that $|x\rangle$ and $|y\rangle$ are unit vectors, by writing

$$|xy\rangle = \frac{1}{|\,|xy\rangle\,|}\,|xy\rangle = \frac{|x\rangle}{|\,|x\rangle\,|}\frac{|y\rangle}{|\,|y\rangle\,|},$$

if necessary. Hence $\langle y|y\rangle = 1$ and

$$p_{\hat{M}}(|xy\rangle \to a) = \left\langle x|M_a^\dagger M_a|x\right\rangle = p_M(|x\rangle \to a),$$

the probability that a is observed if Q is measured with M. Similarly

$$p_{\hat{N}}(|xy\rangle \to b) = p_N(|y\rangle \to b)$$

the probability that b is observed if R is measured with N. The probability of observing both a in register Q and b in register R when V is in state $|xy\rangle$ is

$$\begin{aligned} p_L(ab) = \left\langle xy|L_{ab}^\dagger L_{ab}|xy\right\rangle &= \left\langle x\otimes y|M_a^\dagger M_a \otimes N_b^\dagger N_b|x\otimes y\right\rangle \\ &= \left\langle x|M_a^\dagger M_a\right\rangle \left\langle y|N_b^\dagger N_b|y\right\rangle \\ &= p_{\hat{M}}(a)p_{\hat{N}}(b). \end{aligned}$$

Hence $p_{\hat{M}}(|xy\rangle \to a)$ and $p_{\hat{N}}(|xy\rangle \to b)$ are independent.

Example 2.3. With V, Q, R, M and N as in the previous example, let $|x\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ and $|y\rangle = \gamma\,|0\rangle + \delta\,|1\rangle$, where where $|\alpha|^2 + |\beta|^2 = 1$ and $|\gamma|^2 + |\delta|^2 = 1$. Then

$$|x\rangle \otimes |y\rangle = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle.$$

Hence

$$p_{\hat{M}}(0) = |(\alpha\gamma)|^2 + |\alpha\delta|^2 \text{ and } p_{\hat{N}}(0) = |(\alpha\gamma)|^2 + |\beta\gamma|^2,$$

while

$$p_L(00) = |\alpha\gamma|^2.$$

Hence

$$p_{\hat{M}}(0)p_{\hat{N}}(0) = |\alpha\gamma|^2(|\alpha\gamma|^2 + |\beta\gamma|^2 + |\alpha\delta|^2 + |\beta\delta|^2) = |\alpha\gamma|^2 = p_L(00).$$

On the other hand, in general, the measurements of individual registers do not have independent probabilities. To see this consider the entangled state $(|00\rangle+|11\rangle)/\sqrt{2}$ in the system of the previous example. Measuring in register Q we observe either 0 or 1, each with a probability of 1/2. The same holds for register R. If these probabilities are independent then the probability of observing 00 should be 1/4. However, it's easy to see that the probability of observing 00 is also 1/2. In fact after the first measurement the system projects to one of the states $|00\rangle$ or $|11\rangle$. Measuring the former we observe 00 with probability 1 and 11 with probability 0.

The above example illustrates the fact that if $|\psi\rangle$ is disentangled then probabilities of observing a in register Q and b in register R are independent, where $|a\rangle$ and $|b\rangle$ range over basis vectors of Q and R. The converse is *not* true. For example take $|\psi\rangle = [|00\rangle + |01\rangle + |10\rangle - |11\rangle]/2$. The probability $p_{\hat{M}}(a)$ of observing a in register Q is 1/2, as is the probabilty $p_{\hat{N}}(b)$ of observing b in register R, for all a and b in $\{0, 1\}$. As the probabilty $p_L(ab)$ of observing $|ab\rangle$ is 1/4, the probabilities $p_{\hat{M}}$ and $p_{\hat{N}}$ are independent. However $|\psi\rangle$ is entangled.

2.10. Quantum parallelism. We develop here the notion of parallel quantum computation which we touched on briefly in Section 5.1. A basic tool in the operation of this scheme is the family of functions we describe next.

The *single qubit Walsh–Hadamard transformation* is the unitary operator W on a single qubit system given by

$$\begin{aligned} W(|0\rangle) &= \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \text{ and} \\ W(|1\rangle) &= \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right). \end{aligned} \tag{2.6}$$

We can express W more concisely by writing

$$\begin{aligned} W(|x\rangle) &= \frac{1}{\sqrt{2}}\left(|0\rangle + (-1)^x |1\rangle\right) \\ &= \frac{1}{\sqrt{2}}\sum_{k=0}^{1}(-1)^{kx}|k\rangle. \end{aligned}$$

It is easy to see by direct calculation that W is an involution, that is $W^2 = I_2$. Moreover, if we ignore the complex coefficients W is reflection in the line which makes an angle of $\pi/8$ with the $|0\rangle$–axis.

The *n–bit Walsh–Hadamard transformation* W_n is defined to be $W^{\otimes n}$. As W is an involution we have $W_n^2 = I_2^{\otimes n} = I_{2^n}$, so W_n is also an involution. Applied to $|0\rangle^{\otimes n}$, W_n generates a uniform linear combination of the integers from 0 to $2^n - 1$, i.e.

$$W_n(|0\cdots 0\rangle) = \frac{1}{\sqrt{2^n}}\sum_{x=1}^{2^n-1}|x\rangle.$$

For example,

$$\begin{aligned} W_2|00\rangle &= (W\otimes W)(|0\rangle\otimes|0\rangle) \\ &= W|0\rangle \otimes W|0\rangle \\ &= \frac{1}{2}\left((|0\rangle + |1\rangle)\otimes(|0\rangle + |1\rangle)\right) \\ &= \frac{1}{2}\left(|0\rangle\otimes|0\rangle + |0\rangle\otimes|1\rangle + |1\rangle\otimes|0\rangle + |1\rangle\otimes|1\rangle\right) \\ &= \frac{1}{\sqrt{2^2}}\left(|00\rangle + |01\rangle + |10\rangle + |11\rangle\right). \end{aligned}$$

This generalises in the obvious way to W_n and allows us, starting with the simple basis state $|0\cdots 0\rangle$, to prepare a uniform superposition of all basis vectors.

For computation a more concise notation is convenient and to this end we define the following notation. Let $|x\rangle$ and $|y\rangle$ be basis vectors in an n–qubit system, where x and y are n–bit binary integers. Define

$$x\cdot y = \sum_{i=0}^{n-1} x_i y_i.$$

(Note that this is not the inner product of $|x\rangle$ and $|y\rangle$ (see page 5). It extends to a symmetric bilinear form on $|x\rangle$ and $|y\rangle$ regarded as vectors in a 2^n–dimensional space over $\mathbb{Z}_2$. However it may be zero when $|x\rangle$ and $|y\rangle$ are both non–zero.) Now, setting $m = 2^n$, we have

$$\begin{aligned}
W_n |x\rangle &= \bigotimes_{i=0}^{n-1} W |x_i\rangle \\
&= \bigotimes_{i=0}^{n-1} \frac{1}{\sqrt{2}} \sum_{k_i=0}^{1} (-1)^{k_i x_i} |k_i\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{k_0=0}^{1} \cdots \sum_{k_{n-1}=0}^{1} (-1)^{x_0 k_0} \cdots (-1)^{x_{n-1} k_{n-1}} |k_0 \cdots k_{n-1}\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{k_0 \cdots k_{n-1}=0}^{m-1} (-1)^{x_0 k_0} \cdots (-1)^{x_{n-1} k_{n-1}} |k_0 \cdots k_{n-1}\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{k=0}^{m-1} (-1)^{x \cdot k} |k\rangle . \qquad (2.7)
\end{aligned}$$

The Walsh–Hadamard functions allow us to prepare the input to parallel computations. Now we consider the computations themselves. Let $f : \mathbb{Z}_2^m \to \mathbb{Z}_2^k$ be a function, not necessarily invertible. As we're not assuming that f is invertible we cannot use it, as it is, as a transformation in a quantum computer. However, at the expense of introducing some extra storage space we can devise a unitary transformation to simulate f. We require a quantum system V which is the tensor product of an m–qubit quantum system with a k–qubit quantum system. Recall that V has basis consisting of the vectors $|x\rangle \otimes |y\rangle$, where x and y are binary representations of integers in $\{0, \ldots, 2^m - 1\} = \mathbb{Z}_2^m$ and $\{0, \ldots, 2^k - 1\} = \mathbb{Z}_2^k$ respectively. Define the linear transformation

$$U_f : |x\rangle \otimes |y\rangle \mapsto |x\rangle \otimes |y \oplus f(x)\rangle ,$$

where $\oplus$ denotes addition in the group $\mathbb{Z}_2^k$ (known as "bitwise exclusive OR" in the literature). For fixed x, we see that $y \oplus f(x)$ takes every value in $\mathbb{Z}_2^k$ exactly once, as y varies over $\{0, \ldots, 2^k - 1\}$. Therefore U_f simply permutes all 2^{m+k} basis elements of V and it follows that it is unitary. Moreover $U_f(|x\rangle \otimes |0\rangle) = |x\rangle \otimes |f(x)\rangle$ and in this sense U_f simulates f. The map U_f is referred to as the *standard oracle* for the function f. The standard oracle may thus be used to simulate any function, invertible or not, on a quantum computer. It follows that any function which may be carried out by a classical computer may also be carried out by a quantum computer.

In the case where f is a bijection, and only in this case, we can define the simpler and more obvious oracle $|x\rangle \to |f(x)\rangle$. This is called the *minimal* or *erasing oracle* for f. Its relation to the standard oracle is considered in [**36**]. Furthermore, in [**1**] a problem is given in which a minimal oracle is shown to be exponentially more powerful than a standard oracle.) The form of U_f above may seem strange, but in fact it originates in classical reversible computing and has been adapted for the purposes of quantum computing. See [**39**] for more details of reversible computing.

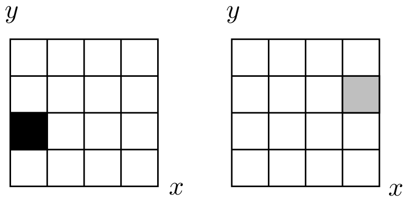
If we apply U_f to

$$W_m(|0\rangle^{\otimes m}) \otimes |0\rangle^{\otimes k}$$

we obtain

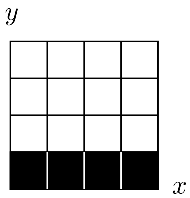
$$\begin{aligned}
U_f \left(\frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |0\rangle \right) &= \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} U_f(|x\rangle \otimes |0\rangle) \\
&= \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |f(x)\rangle .
\end{aligned}$$

We can view this as a simultaneous computation of f on all possible values of x, although the fact that $|f(x)\rangle$ is associated with the state $|x\rangle$, for all x, may sometimes be a problem. Creation of this kind of state is often referred to as *quantum parallelism* and is an easy and standard first



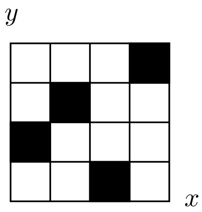
$|0\rangle \otimes |1\rangle$ $\qquad$ $-|3\rangle \otimes |2\rangle$

FIGURE 1. Basis states.



$(W|0\rangle) \otimes |0\rangle$

FIGURE 2. A uniform superposition in the first register



$U_f((W|0\rangle) \otimes |0\rangle)$

FIGURE 3. A quantum state corresponding to the graph of f

step in many quantum computations. The tricky part is to glean useful information from this (extremely entangled) output state.

EXAMPLE 2.4. Suppose that $m = k = 2$, so that $\mathbb{Z}_2^m = \mathbb{Z}_2^k = \mathbb{Z}_2^2 = \{0, \dots, 3\}$. Let f be the function defined by $f(0) = 1$, $f(1) = 2$, $f(2) = 0$ and $f(3) = 3$. The quantum system V is the tensor product $\mathbb{C}\mathbb{Z}_2^2 \otimes \mathbb{C}\mathbb{Z}_2^2$ and has basis $\{|x\rangle \otimes |y\rangle : 0 \leq x, y \leq 3\}$. We can represent the elements of this basis on a 3×3 grid, with x indexing the horizontal squares and y the vertical ones. For many of the algorithms we consider, the quantum state is always in a uniform superposition of an r–element subset of the set of basis elements, with phase (coefficient) $\pm 1/\sqrt{r}$. We represent such states by using a black square for a coefficient of $1/\sqrt{r}$, a grey square for a coefficient of $-1/\sqrt{r}$ and a white square for a coefficient of 0. For example, a basis state is represented by a single square as in Figure 1 and the state $(W|0\rangle) \otimes |0\rangle$ is represented as in Figure 2. If we apply U_f to this state we obtain the state shown in Figure 3, which can be considered as a uniform superposition over all of the points of the graph of f.

It's useful, as an exercise, to calculate U_f applied to $(W|0\rangle) \otimes |1\rangle$, $(W|0\rangle) \otimes |2\rangle$ and $(W|0\rangle) \otimes |3\rangle$ in turn, to understand how U_f is constructed, and why it is reversible. In fact, U_f is always an involution (i.e. $U_f \circ U_f = I$), regardless of what f is.

EXAMPLE 2.5. Pictures of the quantum system, as in the previous example, can also be used to help understand measurements of quantum states, and how they relate to entanglement. All

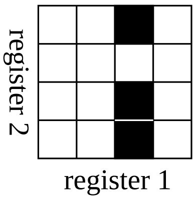


FIGURE 4. A disentangled state

measurements are taken with respect to the computational basis (see Section 2.3). A disentangled state such as, for example,

$$|2\rangle \otimes \frac{1}{\sqrt{3}}(|0\rangle + |1\rangle + |3\rangle)$$

looks the same in every nonzero column (up to phase), so measuring the first register (which always results in 2) doesn't affect the distribution of non–zero coefficients in the second register (see Figure 4). On the other hand, if we have an entangled state such as, for example,

$$\frac{1}{\sqrt{6}}(|0\rangle \otimes |0\rangle + |0\rangle \otimes |2\rangle - |2\rangle \otimes |1\rangle + |2\rangle \otimes |2\rangle - |2\rangle \otimes |3\rangle + |3\rangle \otimes |3\rangle)$$

we can see directly from Figure 5 how measurements of the first register affect the distribution on the second. If we observe 0 in the first register, then there is a probability of 1/2 of observing 0

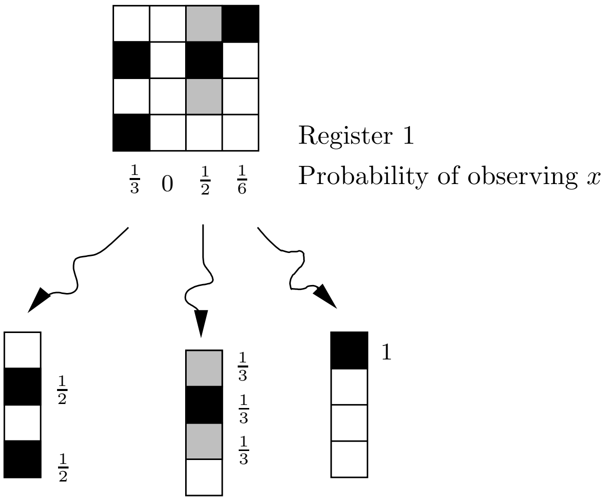


FIGURE 5. Measurement of an entangled state

in the second register, and a probability of 1/2 of observing 2. If we observe 2 in the first register then we have probabilities of 1/3 of observing either 2, 3 or 4 and if we observe 3 in the first register then we observe 3 with certainty in the second register.

Some care is required in the interpretation of these diagrams. For example, the diagram of the entangled state $|00\rangle - |01\rangle - |10\rangle + |11\rangle$ will suggest different results in the second register after measurement of the first register: depending on whether 0 or 1 is observed. However these results differ only by a phase factor (of -1) so are, in fact, the same.

3. The Deutsch–Jozsa algorithm

3.1. Oracles and query complexity. Deutsch [**14**] was the first to show that a speed–up in complexity is possible when passing from classical to quantum computations. It is important to understand that the complexity referred to is *query complexity.* The idea is that we have a "black box" or "oracle", as described in Section 2.10, which evaluates some function (classically this would be a function on integers, but in the quantum scenario we allow evaluations on complex vectors). Query complexity simply addresses how many times we have to ask the oracle to evaluate the function on some input, in order to determine some property of f. It ignores how many quantum gates we require to actually implement the function. For a genuine upper bound on time complexity we must demonstrate efficient implementation of the oracle. For example, this is the case in Shor's algorithm in the next section where we show that the quantum Fourier transform can be implemented using a number of gates logarithmic in the size of the input.

Again, a lower bound on the query complexity of a given algorithm, with respect to a specific oracle, does not necessarily give a lower bound on time complexity of the algorithm. This is because we know nothing about the operation of the oracle: if we knew how the oracle worked then perhaps we could see how to do without it. Within the context of query complexity (relative to a specific oracle) many quantum algorithms have been *proved* to be more efficient than any classical counterpart. However, so far, not a single instance exists where we can say the same about true time complexity. To do so would usually require a lower bound on the classical complexity of a given problem and this often brings us up against difficult open problems in classical complexity theory. For example, Shor's algorithm is a (non–deterministic) quantum algorithm for factoring integers. Although no classical polynomial time algorithm is *known* for this problem, whether or not such an algorithm exists is an open question.

Given two functions f and g from $\mathbb{N}$ to $\mathbb{R}$ we say that $f = O(g)$ if there exist constants $c, k \in \mathbb{R}$ such that $|f(n)| \leq c|g(n)| + k$, for all $n \in \mathbb{N}$. We also say that $f = \Omega(g)$ if $g = O(f)$. Thus O is used to describe upper bounds and Ω to describe lower bounds.

3.2. The single qubit case of Deutsch's algorithm. The unitary maps involved in quantum computing can often be represented pictorially via *quantum circuit diagrams.* An operator U on a single quantum register is represented as in Figure 6. We also draw gates for operators on

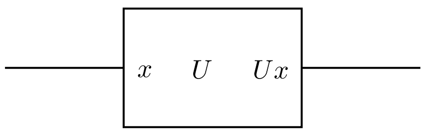


Figure 6. A single–qubit gate

two or more quantum registers: the set–up for quantum parallelism as described in Section 2.10 is shown in Figure 7. A similar circuit is shown in Figure 8. Here an additional Walsh–Hadamard

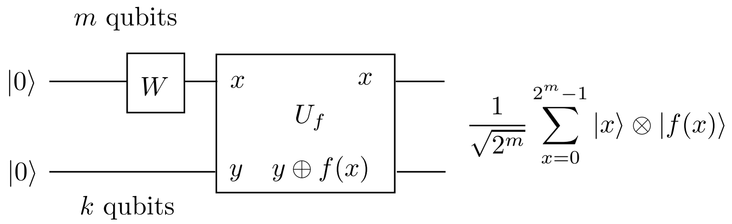


Figure 7. The circuit for quantum parallelism

gate operates on the second register, transforming its contents into a uniform superposition, before entering the U_f gate.

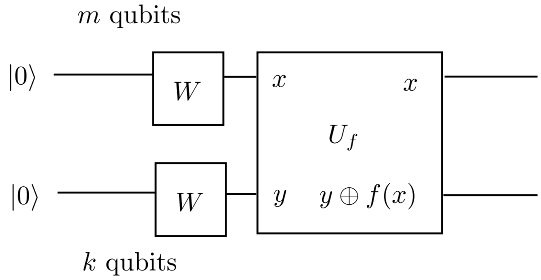


FIGURE 8. An insensitive quantum circuit

Computing the final state of this system we have

$$\begin{aligned} |0\rangle \otimes |0\rangle \quad &\overset{W\otimes W}{\mapsto} \quad \frac{1}{\sqrt{2^{m+k}}} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^k-1} |x\rangle \otimes |y\rangle \\ &\overset{U_f}{\mapsto} \quad \frac{1}{\sqrt{2^{m+k}}} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^k-1} |x\rangle \otimes |y \oplus f(x)\rangle \\ &= \quad \frac{1}{\sqrt{2^{m+k}}} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^k-1} |x\rangle \otimes |y\rangle \,, \end{aligned}$$

because $y \oplus f(x)$ takes each possible value exactly once, as y ranges over $\{0, \dots, 2^k - 1\}$. This computation can certainly not be used to gain any information about f, because its final state is the same, whatever f is. However, Deutsch showed that, with $k = 1$, if we alter input to the second register to $|1\rangle$ then we can obtain some information on the nature of f.

Deutsch's algorithm concerns functions $f : \{0, 1\} \to \{0, 1\}$. We call such a function *constant* if $f(0) = f(1)$ and *balanced* if $f(0) \neq f(1)$. Given such a function suppose that we wish to determine whether f is constant or balanced (it must be one or the other). Classically, this requires two evaluations of f. Let U_f be the standard oracle for f (see Section 2.10). We shall show that a quantum computer only needs a single evaluation of the oracle to determine whether f is constant or balanced (with certainty). The quantum circuit for the algorithm is shown in Figure 9. After

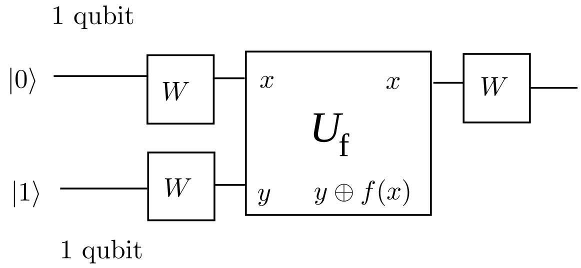


FIGURE 9. The quantum circuit for Deutsch's algorithm

passing through the Walsh–Hadamard gates, the registers are in the state

$$(W \otimes W)(|0\rangle \otimes |1\rangle) = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).$$

Now, if $x \in \{0,1\}$, we have

$$\begin{aligned} U_f\left(|x\rangle \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)\right) &= |x\rangle \otimes \left(\frac{|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle}{\sqrt{2}}\right) \\ &= |x\rangle \otimes \frac{1}{\sqrt{2}}\left(\begin{cases} |0\rangle - |1\rangle & \text{if } f(x) = 0 \\ |1\rangle - |0\rangle & \text{if } f(x) = 1 \end{cases}\right) \\ &= (-1)^{f(x)}|x\rangle \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \end{aligned}$$

Therefore, by linearity, after passing through the U_f gate the system is in state

$$\left(\frac{(-1)^{f(0)}\,|0\rangle + (-1)^{f(1)}\,|1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

which is equal to

$$\pm\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right), \text{ if } f \text{ is constant,}$$

and

$$\pm\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right), \text{ if } f \text{ is balanced.}$$

After passing through the final Walsh–Hadamard gate, the first qubit is in state

$$\begin{array}{ll} \pm\,|0\rangle & \text{if } f \text{ is constant} \\ \pm\,|1\rangle & \text{if } f \text{ is balanced} \end{array}.$$

So measuring this qubit, with respect to the computational basis, we observe 0 with probability 1, if f is constant, and 1 with probability 1, if f is balanced. Since f has only been evaluated once, this demonstrates that quantum computers are strictly more efficient than classical computers, when we are referring to deterministic black box query complexity.

In fact Deutsch's algorithm puts each of the functions in Figure 10 in one of the two classes: constant or balanced. With the notation of Figure 10, write $U_i = U_{f_i}$. Then the operation of

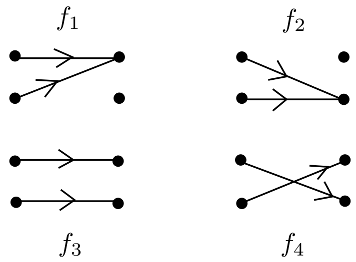


FIGURE 10. The four possible functions $\{0,1\} \to \{0,1\}$

Deutsch's algorithm on each function can be pictorially represented as in Figure 11.

3.3. The general Deutsch–Jozsa algorithm. The generalisation of the algorithm of the previous section to m qubits is due to Deutsch and Jozsa [**15**] (see also [**12**] for the improved version which we present here). A function $f : \mathbb{C}\mathbb{Z}_2^m \to \{0,1\}$ is called *balanced* if $|f^{-1}(0)| = |f^{-1}(1)| = 2^{m-1}$. Assume we know only that f is either constant or balanced, and we that wish to determine which of these properties f has. Classically this requires $2^{m-1}+1$ evaluations. However on a quantum computer it can be done with a single evaluation of an oracle. The circuit for this algorithm is the same as the Deutsch algorithm, apart from the number of input qubits in the first register, and is shown in Figure 12.

Assume then that we have a function $f : \mathbb{Z}_2^m \to \{0,1\}$ which is either balanced or constant. Again, we employ the standard oracle U_f for f. We use a composite system with an m–qubit and

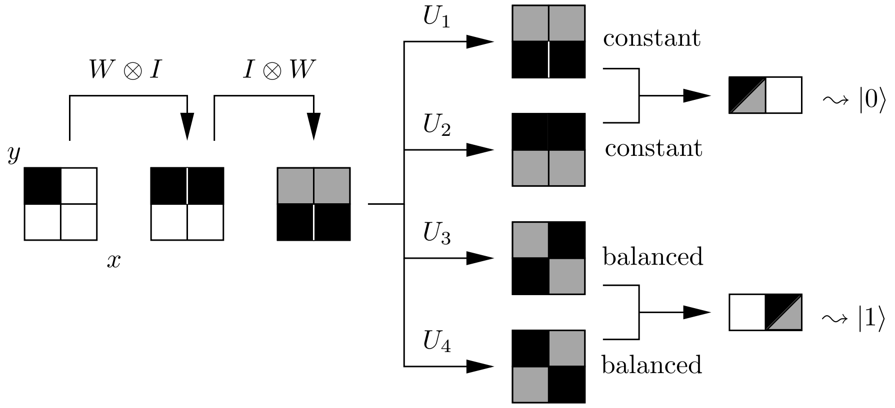


FIGURE 11. Quantum states created by Deutsch's algorithm

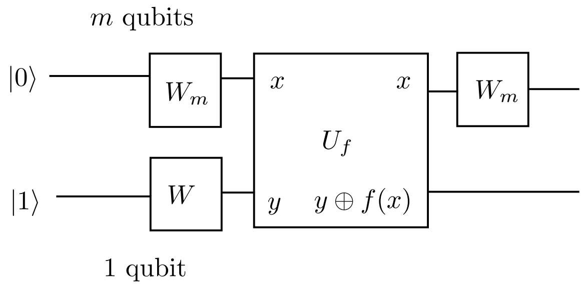


FIGURE 12. The quantum circuit for the Deutsch–Jozsa algorithm

a 1–qubit register. This time we begin with the basis state $|0\rangle^{\otimes m}|1\rangle$ to which we apply $W_m \otimes W$ to obtain the state

$$\frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x\rangle \otimes W|1\rangle .$$

Passing this through the U_f gate the state of the system becomes

$$\frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} (-1)^{f(x)} |x\rangle \otimes W|1\rangle .$$

Ignoring the second qubit, which is the same for any function, what we have done here is to encode the function as the coefficients of a em single state. We shall define $\mathcal{D}(f)$ to be the state

$$\mathcal{D}(f) = \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} (-1)^{f(x)} |x\rangle . \tag{3.1}$$

If f is constant then applying W_m to $\mathcal{D}(f)$ will give $\pm|0\rangle$, and if we measure the state we will always observe 0.

If f is balanced then we need to make a more careful analysis of the resulting state. In any case, after applying W_m to $\mathcal{D}(f)$ we have, using (2.7),

$$\frac{1}{2^m} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^m-1} (-1)^{f(x)\oplus(x\cdot y)} |y\rangle .$$

Since f is balanced the coefficient of $|0\rangle$ in this state is

$$\frac{1}{2^m}\sum_{x=0}^{2^m-1}(-1)^{f(x)}=0.$$

Hence, when we measure the state we *never* observe 0. Thus we can distinguish between balanced and constant functions.

Note that we can only create $\mathcal{D}(f)$ for functions $f:\{0,\ldots,2^{m-1}\}\to\{0,\ldots,2^{k-1}\}$ in the case where $k=1$, but the general idea is that it is easier to manipulate a single register to gain information about a function than a register pair containing the standard 2–register functional superposition provided by the U_f gate. This will also apply later, in Shor's algorithm where a measurement of the second register puts the first register into a certain state which we work with, and also in Grover's algorithm, which manipulates the amplitudes of $\mathcal{D}(f)$ to ensure that we have a high probability of observing an x for which $f(x)=1$.

In conclusion, we have exhibited a problem which takes time $O(2^m)$ classically but takes time $O(1)$ on a quantum computer.

The black box query complexity speed–up is hence exponential, from the classical to quantum setting. However the exponential speed up is not really robust when compared to probabilistic algorithms. If one is willing to accept output which is not correct with absolute certainty then, given any $\varepsilon>0$, an answer correct with probability $1-\varepsilon$ is classically attainable using at most $\log_2(1/\varepsilon)$ queries. Thus, effectively, we have only a constant factor speed up, for any fixed permissible probability of error.

In this section we have studied properties of functions which can be deduced exactly from $\mathcal{D}(f)$ with a single quantum measurement. In a related work [**32**], properties of functions which can be deduced from the standard state $\sum_x|x\rangle\otimes|f(x)\rangle$ with a single measurement are studied . Even when a certain probability of failure is allowed this class of functions turns out to be very restricted.

4. Shor's algorithm and factoring integers

4.1. Overview. Shor's Algorithm, first published in 1994, factors an integer N in time $O(L^2 \log(L) \log\log(L))$, where $L = \log(N)$. No classical algorithm of this time complexity is known. The problem of factoring integers turns out to be reducible to that of finding the order of elements of a finite cyclic group. That is, given an order finding algorithm, an algorithm may be constructed which will, with high probability return a non–trivial factor of a given composite integer. Moreover this algorithm may be run quickly on a classical computer. We describe the reduction of factoring to order finding below.

Order finding in a cyclic group is a special case of the more general problem of finding the period of a function which is known to be periodic.

DEFINITION 4.1. *If f is a function from a cyclic group A (written additively) to a set S then we say that f is* periodic*, with* period $r \in A$*, if the following two conditions hold. For all $x \in A$*

(i) $f(x) = f(x + r)$ and
(ii) if $f(x) = f(x + y)$ then $r|y$.

Given a periodic function from $\mathbb{Z}$ to $\mathbb{Z}_N$ the best known classical algorithm to determine r requires $O(N)$ steps. By contrast, using Shor's algorithm, [**53**] [**54**], on a quantum computer only $O((\log N)^2)$ steps are required.

As shown by Kitaev [**38**], the essential element of Shor's algorithm is the use of the Quantum Fourier transform to find an eigenvalue of a unitary transformation. This technique was first used in this way by Simon [**55**] [**56**] to generalise Deutsch's algorithm. Deutsch's algorithm, Simon's generalisation and the problems of period finding can all be regarded as cases of a problem known as the hidden subgroup problem. We shall describe this problem and various other applications of the Fourier transform in 4.9. Here we first outline the reduction of factoring of integers to order finding, in Section 4.2. We then describe the essential ingredient to Shor's algorithm, namely the Quantum Fourier transform and in Section 4.3. The algorithm itself is described in Section 4.5 and we end the section with a brief description of the implementation of the Quantum Fourier transform and an outline of the continued fractions algorithm which is necessary to extract information from the quantum part of the main algorithm.

4.2. Factoring and period finding. It is well known that the ability to find the period of functions effectively would lead to an efficient algorithm for factoring integers. In order to see this suppose we wish to factor the integer N. We may clearly assume that N is odd and, since there exist effective probabilistic tests for prime powers [**40**], that N is divisible by more than one odd prime. Consider the function $F_N : \mathbb{Z}_N \to \mathbb{Z}_N$ given by

$$F_N(a) = y^a \bmod N,$$

where y is a randomly chosen integer in the range $0 \leqslant y < N$. Using the Euclidean algorithm on y and N we either find a factor of N or we find that y is coprime to N. We may therefore assume that $\gcd(y, N) = 1$. If y and N may both be represented as strings of at most L bits then the total resource required by this step is $O(L^3)$ since this is a bound on the cost of running the Euclidean algorithm [**40**, p. 13]. With $\gcd(y, N) = 1$ the function F_N takes distinct values $1, y, \ldots, y^{r-1}$, where r is the (unknown to us) multiplicative order of y modulo N. Thus F is periodic of period r. Suppose now that we have computed the multiplicative order r of y modulo N, using our hypothetical period finding algorithm on F_N. Since $N|y^r - 1$ the Euclidean algorithm applied to N and $y^r - 1$ merely returns the factor N of N. On the other hand, if r is even then

$$1 = y^r = (y^{r/2} - 1)(y^{r/2} + 1) \mod N.$$

As r is minimal with the property that $y^r = 1 \mod N$ it follows that $N \nmid y^{r/2} - 1$, from which we see that N and $y^{r/2} + 1$ have a common factor greater than 1. We now run the Euclidean algorithm with input N and $y^{r/2}+1$. If $N \nmid y^{r/2}+1$ then we obtain a non–trivial factor of N, again in time at most $O(L^3)$. This step succeeds if and only if we happen to choose y such that y has even order, r, and in addition $N \nmid y^{r/2}+1$. What is the probability of success? Following [**16**] we count the number of integers y which result in failure. We have $1 \le y \le N$ and $\gcd(y,N)=1$, so that $y \in \mathbb{Z}_N^*$ the group of units of $\mathbb{Z}_N$, and r is the order of y in this group. Note that the order of $\mathbb{Z}_N^*$ is $\phi(N)$, where ϕ is Euler's totient function: that is, $\phi(N)$ is the number of integers between 1 and N which are coprime to N (see [**40**, p. 19] for example). The notation of the following lemma is ambiguous, as $|x|$ denotes the order of a group element x and $|X|$ denotes the cardinality of a set X, but the meaning should be clear from the context.

LEMMA 4.2 ([**16**]). *Let $N = p_1^{\alpha_1}\cdots p_m^{\alpha_m}$ be the collected prime factorisation of an odd composite integer N and let*

$$S = \{s \in \mathbb{Z}_N^* : |s| \text{ is odd or } s^{|s|/2} = -1\}.$$

Then $|S| \le \phi(N)/2^m$.

PROOF. For $n \in \mathbb{Z}$ define $l(n)$ to be the greatest integer d such that $2^d | n$.

Consider first the group $\mathbb{Z}_{p^\alpha}^*$, where p is an odd prime and α a positive integer. This group is cyclic of even order $\phi = \phi(p^\alpha) = p^{\alpha-1}(p-1)$ with generator x, say. Every element of $\mathbb{Z}_{p^\alpha}^*$ is of the form x^k, for some $k \ge 0$, and for exactly half the elements k is even. Let $g = x^k$ be an element of $\mathbb{Z}_{p^\alpha}^*$ and suppose the order of g is r. If k is even then

$$g^{\phi/2} = x^{k\phi/2} = (x^\phi)^{k/2} = 1,$$

so $r|\phi/2$ and it follows that $l(r) < l(\phi)$. Conversely if k is odd then $1 = g^r = x^{kr}$ so $\phi | rk$ and, since ϕ is even, this implies $l(\phi) \le l(r)$. Therefore precisely half of the elements g of $\mathbb{Z}_{p^\alpha}^*$ satisfy $l(|g|) < l(\phi)$.

Now consider $y \in \mathbb{Z}_N^*$. The Chinese remainder theorem [**28**] states that $\mathbb{Z}_N^*$ is isomorphic to $\bigoplus_{i=1}^m \mathbb{Z}_{p_i^{\alpha_i}}^*$ under the map taking y to $(y_1,\dots,y_m)$, where $y = y_i \in \mathbb{Z}_{p_i^{\alpha_i}}^*$, for $i=1,\dots,m$. Let y have order r in $\mathbb{Z}_N^*$ and let y_i have order r_i in $\mathbb{Z}_{p^{\alpha_i}}^*$. Applying the above isomorphism we see that $r_i | r$ so $l(r_i) \le l(r)$. First suppose that r is odd. Then $l(r) = 0$ so $l(r_i) = 0$, for $i = 1,\dots,m$. Now suppose that r is even but that $y^{r/2} = -1$ in $\mathbb{Z}_N^*$. Using the Chinese remainder theorem again it follows that $y_i^{r/2} = -1$ in $\mathbb{Z}_{p^{\alpha_i}}^*$ and so $r_i \nmid r/2$. Hence $l(r_i) = l(r)$, for $i = 1,\dots m$. We have shown that if $y \in S$ then $l(r_i) = l(r)$, for $i = 1,\dots,m$. To complete the proof we need only show that this is possible for at most $\phi(N)/2^{m-1}$ elements of $\mathbb{Z}_N^*$.

Fix $y_1 \in \mathbb{Z}_{p_1^{\alpha_1}}^*$ and let $r_1 = |y_1|$. From the first paragraph of the proof it follows that there are at most

$$\prod_{i=2}^m \frac{\phi(p_i^{\alpha_i})}{2}$$

elements $(y_2,\dots,y_m)$ of $\bigoplus_{i=2}^m \mathbb{Z}_{p_i^{\alpha_i}}^*$ such that $l(r_1) = l(|y_i|)$, for $i = 2,\dots = m$. Summing over all elements of $\mathbb{Z}_{p_1^{\alpha_1}}^*$, there are at most

$$\phi(p_1^{\alpha_1}) \prod_{i=2}^m \frac{\phi(p_i^{\alpha_i})}{2} = \phi(N)/2^{m-1}$$

elements $(y_1,\dots,y_m)$ of $\bigoplus_{i=1}^m \mathbb{Z}_{p_i^{\alpha_i}}^*$ with $l(|y_1|) = \cdots = l(|y_m|)$. □

Now let y be an integer chosen uniformly at random from

$$\{y \in \mathbb{Z} : 1 \le y \le N-1 \text{ and } \gcd(y,N) = 1\}.$$

Then from Lemma 4.2 the probability that y has even order in $\mathbb{Z}_N^*$ and $y^{|y|/2} \ne -1 \mod N$ is at least $1 - 1/2^{m-1}$. The resource cost of the procedure is $O(L^3)$, in addition to the cost of the order finding routine, and repeating it sufficiently often we can find a factor of N, with probability as close to 1 as we like. Shor's quantum algorithm, described below, finds the order of $y \in \mathbb{Z}_N^*$, with probability $1-\varepsilon$, for given $\varepsilon > 0$, in time polynomial in L. Hence combining Shor's algorithm with the above we have a probabilistic algorithm for factoring integers which runs in time polynomial in $L = \log(N)$.

4.3. The quantum Fourier transform. The Quantum Fourier Transform is more commonly known (to mathematicians) as the Discrete Fourier Transform. That is a Fourier transform on a discrete group: which is defined using characters of irreducible representations. Since we'll be mainly concerned with Abelian groups we need to know very little about characters, and we summarise what is necessary here (for more detail see [**31**] or [**22**]).

A *character* of a finite Abelian group G over a field k is a homomorphism from G to the multiplicative group k^* of non–zero elements of k. (More generally a character is the trace of a representation.) We shall only consider characters over $\mathbb{C}$ here. The set of characters of G is denoted $\hat{G}$. Since G is finite, characters must be roots of unity. In particular the characters of the finite cyclic group $\mathbb{Z}_m$ are the homomorphisms $\chi_m^c = \chi^c$ defined by

$$\chi^c(a) = e^{2\pi iac/m}, \text{ where } a \in \mathbb{Z}_m,$$

for $c = 0, \ldots, m-1$. It's easy to verify that the map $\chi : \mathbb{Z}_m \to \hat{\mathbb{Z}}_m$ defined by $\chi(a) = \chi^a$ is an isomorphism from $\mathbb{Z}_m$ to $\hat{\mathbb{Z}}_m$. (This result extends to all finite Abelian groups.)

We shall use the following simple property of characters of cyclic groups.

LEMMA 4.3. *The characters of* $\mathbb{Z}_m$ *satisfy*
(i)

$$\sum_{a=0}^{m-1} \chi^c(a) = \begin{cases} m & \text{if } c = 0 \mod m \\ 0 & \text{if } c \neq 0 \mod m \end{cases}. \tag{4.1}$$

(ii) (Orthogonality of characters)

$$\sum_{a=0}^{m-1} \chi^c(a)\overline{\chi^d(a)} = \begin{cases} m & \text{if } c = d \mod m \\ 0 & \text{if } c \neq d \mod m \end{cases}. \tag{4.2}$$

PROOF.

(i) This is easy to see if $c = 0$. If $c \neq 0$ then $\chi^c(1) = e^{2\pi ic/m}$ so $\chi^c(1) \neq 1$ and is a root of the polynomial $z^m - 1 = (z-1)(z^{m-1} + \cdots + 1)$. The result follows, since $[\chi^c(1)]^k = \chi^c(k)$.
(ii) This follows easily from (4.1), given the observations that $\overline{\chi^c(x)} = \chi^c(-x)$, $\chi^c(-x) = \chi^{-c}(x)$ and $\chi^c(x)\chi^d(x) = \chi^{c+d}(x)$.

□

For a finite group G the $|G|$–dimensional complex vector space

$$\mathbb{C}G = \bigoplus_{g \in G} \mathbb{C}\, |g\rangle$$

is a ring called the *complex group algebra*, with multiplication defined by

$$\left(\sum_{g \in G} a(g)\, |g\rangle\right) \left(\sum_{h \in G} b(h)\, |h\rangle\right) = \sum_{g \in G} c(g)\, |g\rangle\,,$$

where $a(g), b(g) \in \mathbb{C}$, for all $g \in G$, and $c(g) = \sum_{x \in G} a(x)b(x^{-1}g)$. The group algebra may also be regarded as the ring of maps from G to $\mathbb{C}$, the map a sending g to $a(g)$ corresponding to $\sum_{g \in G} a(g)\, |g\rangle$. An element of $\mathbb{C}\mathbb{Z}_n$ will be said to be *periodic* if it is periodic as a map from $\mathbb{Z}_n$ to $\mathbb{C}$.

The *quantum Fourier transform* on $\mathbb{Z}_n$ is a $\mathbb{C}$–linear map $\mathcal{Q} = \mathcal{Q}_n$ from $\mathbb{C}\mathbb{Z}_n$ to itself defined on the basis vector $|x\rangle$ by

$$\mathcal{Q}\,|x\rangle = \frac{1}{\sqrt{n}}\sum_{c=0}^{n-1}\overline{\chi^c(x)}\,|c\rangle\,,$$

where $\bar{z}$ denotes the complex conjugate of z. Extending by linearity we obtain the image of a general element:

$$\mathcal{Q}\left(\sum_{x=0}^{n-1}\alpha(x)\,|x\rangle\right) = \sum_{c=0}^{n-1}\hat{\alpha}(c)\,|c\rangle\,,$$

where

$$\hat{\alpha}(c) = \frac{1}{\sqrt{n}}\sum_{x=0}^{n-1}\alpha(x)e^{-\frac{2\pi cxi}{n}}. \tag{4.3}$$

We have seen one instance of the Fourier transform already: the single bit Walsh–Hadamard transformation W is equal to $\mathcal{Q}_2$. By comparing coefficients of basis vectors it is easy to see that $W_n \neq \mathcal{Q}_n$, for $n \geq 2$, although it is true that $W_n\,|0\rangle = \mathcal{Q}_n\,|0\rangle$, for all n.

To see that the quantum Fourier transform is a unitary transformation consider its matrix Q, relative to the basis $|x\rangle$, $x \in \mathbb{Z}_n$, for $\mathbb{C}\mathbb{Z}_n$. We have $Q = (a_{c,x})$, where the row c, column x entry is

$$a_{c,x} = \frac{1}{\sqrt{n}}\overline{\chi^c(x)},\ 0 \leq c, x \leq n-1.$$

Therefore the conjugate transpose of Q is $Q^\dagger = (b_{c,x})$, where

$$b_{c,x} = \frac{1}{\sqrt{n}}\chi^x(c),$$

and a straightforward computation using (4.2) shows that $QQ^\dagger = I$. This shows that $\mathcal{Q}$ is a unitary transformation and allows us to write down the image of its inverse, the *inverse Fourier transform* $\mathcal{Q}^{-1}$ on a basis vector $|x\rangle$. That is

$$\mathcal{Q}^{-1}\,|x\rangle = \frac{1}{\sqrt{n}}\sum_{c=0}^{n-1}\chi^c(x)\,|c\rangle\,.$$

In Section 4.7 we shall show how the Fourier transform may be implemented using standard quantum gates.

A crucial property of the quantum Fourier transform is that it identifies periods. That is, if f is periodic, with period r, and r divides n then $\mathcal{Q}(f)$ has non–zero coefficients only at basis vectors $|c\rangle$ which are multiples of n/r.

LEMMA 4.4. *Let $f \in \mathbb{C}\mathbb{Z}_n$ and suppose that f is periodic of period r, where $r|n$. Then*

$$\hat{f}(c) = \begin{cases} \dfrac{\sqrt{n}}{r}\displaystyle\sum_{s=0}^{r-1} f(s)\overline{\chi_n^c(s)}, & \textit{if } c = 0 \mod n/r \\ 0, & \textit{otherwise} \end{cases}. \tag{4.4}$$

PROOF.

$$\hat{f}(c) = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} f(k)\overline{\chi^c(k)}$$

$$= \frac{1}{\sqrt{n}} \sum_{a=0}^{r-1} \sum_{s=0}^{\frac{n}{r}-1} f(a+sr)\overline{\chi^c(a+sr)}$$

$$= \frac{1}{\sqrt{n}} \sum_{a=0}^{r-1} f(a)\overline{\chi^c(a)} \sum_{s=0}^{\frac{n}{r}-1} \overline{\chi^c(sr)}.$$

The result follows on applying Lemma 4.3 modulo n/r. □

4.4. Period finding for beginners. It is instructive to look first at a restricted case of the period finding algorithm. Assume that we have a function $f : \mathbb{Z} \to \mathbb{Z}_N$ which is periodic of period r, as in Definition 4.1. In the restricted case we shall assume that N–ary quantum bits are available, so a basic quantum system is an N-dimensional vector space. Consequently we shall assume, for this section only, that transformations that we've previously defined using qubits are also defined and implemented for such N–ary quantum bits. We shall in addition assume that $r|N$ so that f induces a periodic function from $\mathbb{Z}_N$ to itself. Of course this is an artificial constraint, especially since it is r we're trying to find, but it illustrates the operation of the algorithm without involving the technical detail of the general case.

We begin with two registers, of one N-ary quantum bit each, in initial state $|0\rangle\,|0\rangle$. To the first register we apply the Walsh–Hadamard transform to obtain the state

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

As in previous sections, we assume that a U_f–gate which evaluates f is available. We now apply this to our state to obtain the superposition

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

At this point we could apply the quantum Fourier transform to the first register of the state. However it simplifies notation and makes no difference to the outcome if we observe the second register first (see Example 4.5 below). So, we now observe the second register and obtain, uniformly at random, some value y_0 in the image of f. The system then projects to the state

$$\frac{1}{\sqrt{|f^{-1}(y_0)|}} \sum_{x\in f^{-1}(y_0)} |x\rangle\,|y_0\rangle\,.$$

Since f is periodic in the sense of Definition 4.1, there is precisely one value x_0 with $0 \le x_0 < r$ such that $f(x_0) = y_0$. If we set $K = N/r$ then, in the first register, we have the state

$$|\psi\rangle = \frac{1}{\sqrt{K}} \sum_{k=0}^{K-1} |x_0 + kr\rangle$$
$$= \sum_{x=0}^{N-1} \psi(x)\,|x\rangle\,,$$

where

$$\psi(x) = \begin{cases} \dfrac{1}{\sqrt{K}} = \sqrt{\dfrac{r}{N}}, & \text{if } r|x - x_0 \\ 0, & \text{otherwise} \end{cases}.$$

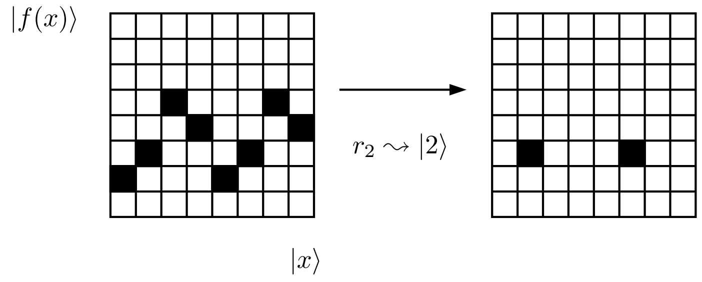


FIGURE 13. Collapsing to a periodic state

Thus $\psi \in \mathbb{C}\mathbb{Z}_N$ is periodic with period r. For example, for the function drawn in Figure 13, where $N = 8$ and $r = 4$, we have observed $|2\rangle$ in the second register and this has set the value x_0 to 1. Notice that, since y_0 is a random element of the image of f, observation of the first register at this stage returns a value $x_0 + kr$, for some uniformly random $k \in \{0, \ldots, N/r - 1\}$. That is, observation of the periodic state $|\psi\rangle$ will simply give a uniformly random value in $\{0, \ldots, N-1\}$ and yields no information on the value of r at all!

Now, if we apply $\mathcal{Q}$ to the first register then, using Lemma 4.4, we obtain

$$\mathcal{Q}\,|\psi\rangle = \sum_{c=0}^{N-1} \hat{\psi}(c)\,|c\rangle\,,$$

where

$$\hat{\psi}(c) = \begin{cases} \dfrac{\sqrt{N}}{r}\displaystyle\sum_{s=0}^{r-1} \psi(s)\overline{\chi^c(s)}, & \text{if } c = 0 \mod \frac{N}{r} \\ 0, & \text{otherwise} \end{cases}$$

$$= \begin{cases} \dfrac{1}{\sqrt{r}}\overline{\chi^c(x_0)}, & \text{if } c = 0 \mod \frac{N}{r} \\ 0, & \text{otherwise} \end{cases},$$

since $\psi(x_0) = 1/\sqrt{K}$ and $\psi(s) = 0$ for all $s \neq x_0$. Therefore, setting $c(s) = sN/r$,

$$\mathcal{Q}\,|\psi\rangle = \frac{1}{\sqrt{r}}\sum_{s=0}^{r-1} \overline{\chi^{c(s)}(x_0)}\,|c(s)\rangle\,. \tag{4.5}$$

Now we observe this state. We obtain a value $c = c(s)$ which is a multiple of N/r. In fact, for a uniformly random $s \in \{0, \ldots, r-1\}$ we have

$$c = s\frac{N}{r} \implies \frac{c}{N} = \frac{s}{r}$$

where the fraction c/N is known. If we reduce c/N down to lowest terms (using the Euclidean algorithm) then we may determine r, as the denominator of this irreducible fraction, provided that $\gcd(s, r) = 1$. If s and r are not coprime then we will obtain a proper factor of r and not r itself. To see that this is not really a problem, we appeal to a result from number theory (see [**28**] for example):

$$\liminf\left(\frac{\phi(n)}{n/\log_e \log_e n}\right) = e^{-\gamma}$$

where γ is a constant known as *Euler's constant.* This means that if we choose a random number from $\{0,\dots,n-1\}$ then the probability $p(n)$ that it is coprime to n satisfies

$$p(n) = \frac{\phi(n)}{n} \geqslant \frac{e^{-\gamma}}{\log_e \log_e n}. \tag{4.6}$$

So we obtain a number coprime to N with probability $1-\varepsilon$, where $\varepsilon > 0$ can be made arbitrarily small, by repeating the above observation $O(\log_e \log_e N)$ times.

In summary, at each iteration the process outputs a number c. A value r may then be read off from the equality $c/N = s/r$, where $\gcd(r,s) = 1$. Repeating the process sufficiently many times we may compute the period of f, to within a given probability, as the least common multiple of the non–zero r values.

Example 4.5. Suppose we wish to factorise 21. We begin by choosing an integer coprime to 21, say 4, as described in Section 4.2. The function $f : \mathbb{Z}_{21} \to \mathbb{Z}_{21}$ is given by $f(k) = 4^k$ and has period r, which we wish to find. After applying W_{21} and U_f we have the state

$$\begin{aligned}\frac{1}{\sqrt{21}}[&(|0\rangle + |3\rangle + |6\rangle + |9\rangle + |12\rangle + |15\rangle + |18\rangle)\,|1\rangle \\ &+(|1\rangle + |4\rangle + |7\rangle + |10\rangle + |13\rangle + |16\rangle + |19\rangle)\,|4\rangle \\ &+(|2\rangle + |5\rangle + |8\rangle + |11\rangle + |14\rangle + |17\rangle + |20\rangle)\,|16\rangle].\end{aligned}$$

Observing the second register we obtain, with probability $1/3$, one of 1, 4 or 16. The first register then contains $|\psi_0\rangle$, $|\psi_1\rangle$ or $|\psi_2\rangle$, where $|\psi_s\rangle = \sum_{k=0}^{6} |s+3k\rangle$.

Applying $\mathcal{Q}_{21}$ to these states we have

$$\begin{aligned}\mathcal{Q}\,|\psi_0\rangle &= \frac{1}{\sqrt{3}}(|0\rangle + |7\rangle + |14\rangle) \text{ or} \\ \mathcal{Q}\,|\psi_1\rangle &= \frac{1}{\sqrt{3}}(|0\rangle + \omega\,|7\rangle + \omega^2\,|14\rangle) \text{ or} \\ \mathcal{Q}\,|\psi_2\rangle &= \frac{1}{\sqrt{3}}(|0\rangle + \omega^2\,|7\rangle + \omega\,|14\rangle),\end{aligned}$$

where $\omega = e^{-2\pi i/3}$. Whichever of these we have, observation now yields, with equal probabilities, $c = 0$, $c = 7$ or $c = 14$. If we observe $c = 0$ then the process must be run again. If we observe $c = 7$ then $c/N = 7/21 = 1/3$ and the denominator of this fraction is r. Similarly, if $c = 14$ we read r off from $c/N = 14/21 = 2/3$. (Of course the example has been set up in the knowledge that the order of 4 divides 21, merely for purposes of illustration. We should not be misled to the conclusion that the example generalises to a simple method for factoring integers.)

Notice that if we had omitted the observation of the second register before application of $\mathcal{Q}$ then, applying $\mathcal{Q}$ would have resulted in the state

$$\frac{1}{3}[(|0\rangle + |7\rangle + |14\rangle)\,|1\rangle + (|0\rangle + \omega\,|7\rangle + \omega^2\,|14\rangle)\,|4\rangle + (|0\rangle + \omega^2\,|7\rangle + \omega\,|14\rangle)\,|16\rangle]$$

and observation would have given the same result as before.

4.5. Advanced period finding. There are two immediate problems to be overcome in implementation of the algorithm in general. First of all we have defined quantum computation in terms of qubits, not N–ary bits. In this setting we need at least $L = \lceil \log(N) \rceil$ qubits to represent N as an binary integer. This means that we shall have to run the algorithm using the quantum Fourier transform $\mathcal{Q}_q$ for some q which is not equal to N. Secondly, we cannot assume that the period r divides q (or N). If q/r is not an integer then a periodic function $f : \mathbb{Z} \to \mathbb{Z}_N$ of period r does not induce a well–defined function from $\mathbb{Z}_q$ to $\mathbb{Z}_N$.

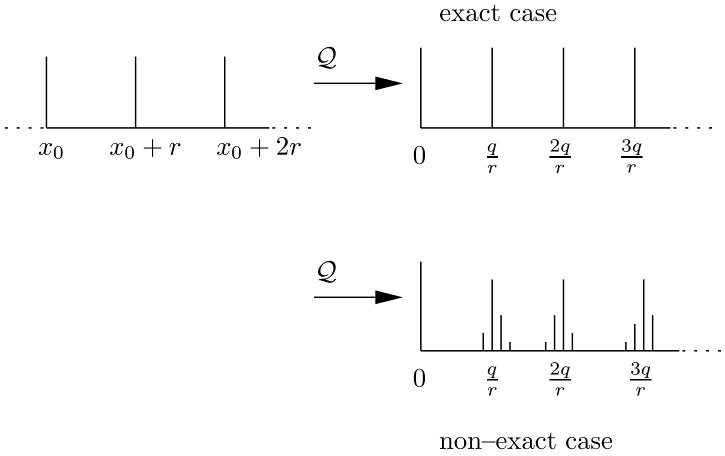


FIGURE 14. Exact and non–exact cases for the quantum Fourier transform

Suppose then that we have a periodic function $f : \mathbb{Z} \to \mathbb{Z}_N$ of period r, as before, but now we do not assume $r|N$. To address the first problem above we choose $q = 2^n > N$, for some integer n. We shall see below that the success of the algorithm depends on making a good choice for q. Recall that we identify $a_0 \otimes \cdots \otimes a_{n-1} \in \mathbb{Z}_2^{\otimes n}$ with a binary integer via the map b such that $b(a_0 \otimes \cdots \otimes a_{n-1}) = 2^{n-1}a_0 + \cdots a_{n-1}$. We use the composite $f \circ b : \mathbb{Z}_2^{\otimes n} \to \mathbb{Z}_N$ to simulate f. Replacing f by this function observe that it satisfies the conditions that, for all x such that $0 \le x < q - r$,

(i) $f(x) = f(x + r)$ and
(ii) $f(x + y) = f(x)$ implies $r|y$, if $0 \le y < q - r - x$.

We shall also use the map b to identify $\mathbb{Z}_2^{\otimes n}$ with the set $\{0, \ldots, q-1\}$, which we regard as $\mathbb{Z}_q$. In particular this identification allows us to apply the Fourier transform $\mathcal{Q}_q$ to $\mathbb{Z}_2^{\otimes n}$.

We can set up our algorithm, along the lines of Section 4.4, starting with two registers, the first of n qubits and the second of L qubits in initial state $|0\rangle\,|0\rangle$. As before we apply Walsh–Hadamard W_n and a U_f gate to obtain the state

$$\frac{1}{\sqrt{q}} \sum_{x=0}^{q-1} |x\rangle\,|f(x)\rangle\,. \tag{4.7}$$

However, if q/r is not an integer then after applying the quantum Fourier transform $\mathcal{Q}_q$ to the first register of (4.7) there may be non–zero coefficients $\hat{\psi}(c)$ at values of c which are not multiples of q/r. Nevertheless it turns out that the non–zero coefficients are clustered around points close to multiples of q/r (see Figure 14). This means that, if a good choice of q is made, there is a high probability of an observation returning a value close enough to one of these multiples to allow the use of a classical algorithm, based on properties of continued fractions, to yield the value of r.

First we shall state the required property of continued fractions and describe how it can be used to extract information from the algorithm. Then we'll complete the description of the algorithm.

The property we need of continued fractions is the following. The necessary definitions and background can be found in Section 4.8.

THEOREM 4.6. **[28]** *If $x \in \mathbb{Q}$ satisfies*

$$\left|\frac{p}{r} - x\right| < \frac{1}{2r^2}$$

then p/r is a convergent of the continued fraction expansion of x.

(This theorem holds for all $x \in \mathbb{R}$, but we do not need this here and we've only defined continued fractions for rational numbers.)

To see how this helps suppose that, as in Section 4.4, we apply $\mathcal{Q}_q$ to the first register of (4.7) and then observe to obtain a value c. If

$$\left|c - s\frac{q}{r}\right| = q\left|\frac{c}{q} - \frac{s}{r}\right| < \frac{q}{2r^2},$$

for some integer s then Theorem 4.6 implies that s/r is a convergent of the continued fraction expansion of c/q. As c and q are known these convergents may all be calculated using the Euclidean algorithm. If, in addition, $\gcd(s, r) = 1$ then the denominator of one of the convergents is r.

Furthermore, given s with $0 \le s < r$ there is a unique integer c_s such that $-r/2 \le c_s r - sq < r/2$ and for such c_s we have

$$\left|\frac{c_s}{q} - \frac{s}{r}\right| < \frac{1}{2q}. \tag{4.8}$$

This motivates the choice of q as the unique integer $q = 2^n$ such that $N^2 \le q < 2N^2$: for then, with s and c_s as above,

$$\left|\frac{c_s}{q} - \frac{s}{r}\right| \le \frac{1}{2q} < \frac{1}{2N^2} < \frac{1}{2r^2}, \tag{4.9}$$

since $r < N$.

Given this choice of q, if we observe c_s for s such that $\gcd(r, s) = 1$, then we can compute the convergents of the continued fraction expansion of c_s/q using the continued fraction algorithm, as described in Section 4.8. From Theorem 4.6 and (4.9), s/r is among these convergents. It is a further consequence of our choice of q that s/r is the unique convergent of c_s/q satisfying the inequality (4.8). To see this suppose that a and b are positive integers with $b < N$ such that a/b also satisfies (4.8). Then

$$\left|\frac{s}{r} - \frac{a}{b}\right| \le \left|\frac{s}{r} - \frac{c}{q}\right| + \left|\frac{c}{q} - \frac{a}{b}\right| < \frac{1}{q}.$$

This implies that $|sb - ar| < rb/q$ and as $N^2 \ge q$ it follows that $|sb - ar| < 1$ so that $s/r = a/b$. Therefore, as claimed, s/r is the unique convergent satisfying (4.8). This being the case we can use (4.8) to find s/r amongst the convergents of c_s/q and this allows us to compute r. The time taken to do this, once c_s has been observed, is therfore $O(L^3)$. Therefore we shall need to know the probability of observing c_s, for s such that $\gcd(r, s) = 1$.

To start with we observe the second register of (4.7) and obtain some value $b \in \mathbb{Z}_N$. As before, there is a such that $f(a) = b$ and $0 \le a < r$, and so $f^{-1}(b) = \{a + kr : 0 \le k < K_a\}$, where K_a is the greatest integer such that $(K_a - 1)r + a < q$. In the first register we now have

$$\begin{aligned}
|\psi\rangle &= \frac{1}{\sqrt{K_a}} \sum_{k=0}^{K_a-1} |a + kr\rangle \\
&= \sum_{x=0}^{q-1} \psi(x) |x\rangle,
\end{aligned}$$

where

$$\psi(x) = \begin{cases} \dfrac{1}{\sqrt{K_a}}, & \text{if } r|x - a \\ 0, & \text{otherwise} \end{cases}.$$

We now apply $\mathcal{Q}$ to the first register and obtain

$$\mathcal{Q}|\psi\rangle = \sum_{c=0}^{q-1} \hat{\psi}(c) |c\rangle, \tag{4.10}$$

where, using (4.3) and an arguement similar to that of the proof of Lemma 4.4,

$$\hat{\psi}(c) = \frac{1}{\sqrt{K_a q}} \sum_{k=0}^{K_a-1} e^{-2\pi i c(a+kr)/q}. \tag{4.11}$$

We now observe the first register and use the following estimate, which we prove in Section 4.6, following [**16**] and [**29**].

PROPOSITION 4.7. *The probability of observing* (4.10) *and obtaining a value* c_s *such that*

$$\left|\frac{c_s}{q} - \frac{s}{r}\right| < \frac{1}{2r^2}$$

and $\gcd(s,r) = 1$ *is at least*

$$\frac{4\phi(r)}{\pi^2 r^2}\left(1 - \left(\frac{\pi r}{2q}\right)^2\right).$$

In particular, if $N \geq 158$ *and* $r \geq 19$ *then this probability is at least* $1/10\log_e \log_e(N)$*.*

Summary: the period finding algorithm

Given a function $f : \mathbb{Z} \to \mathbb{Z}_N$ which is periodic of period r perform the following steps.

(1) First check whether f has period $r < 19$. The number of operations this requires depends on f. For example if f is modular exponentiation, as in Shor's algorithm, then then this step requires $O(log^3(N))$ operataions.
(2) Compute $L = \lceil \log(N) \rceil$ and set $q = 2^n$, where $2\log(N) \leq n < 2\log(N) + 1$. This may be done using a classical algorithm in $O(L)$ operations.
(3) Prepare first and second registers Q and R of n and L qubits, respectively, in state $|0\rangle|0\rangle \in Q \otimes R$.
(4) Apply $W_n \otimes I_L$ to the state of **(3)**. The Walsh–Hadamard transformation W_n may be implemented using n single qubit Walsh–Hadamard gates so the number of operations required in this step is $O(n) = O(L)$.
(5) In this step we assume the existence of a unitary transformation U_f from $Q \otimes R$ to itself, which maps basis vector $|x\rangle|y\rangle$ to $|x\rangle|f(x) \oplus y\rangle$. Apply U_f to the output of **(4)** to give (4.7). The complexity of this step is dependent on f. For example if f is modular exponention then U_f may be implemented using $O(L^3)$ operations.
(6) Observe the second register of (4.7) and project to a state $|\psi\rangle|b\rangle$, where $|\psi\rangle \in Q$ and $b \in \mathbb{Z}_N$.
(7) Apply $\mathcal{Q}_q \otimes I_L$ to $|\psi\rangle|b\rangle$ to obtain (4.10) in the first register (and $|b\rangle$ in the second). We show in Section 4.7 that $\mathcal{Q}_q$ may be implemented in $O(n^2) = O(L^2)$ operations.
(8) Observe the state of **(7)** and obtain a basis vector $|c\rangle|b\rangle$.
(9) Use the continued fraction algorithm (see Section 4.8) to find the convergents of c/q and output a candidate r' for the period of f. This requires $O(L^3)$ operations.

Modulo the complexity of f the above procedure requires $O(L^3)$ operations. ¿From Proposition 4.7 the final step is succesful with probability $1/10\log_e \log_e(N)$. Hence we repeat the algorithm $O(\log\log(N)) = O(log(L))$ times and compute the least common multiple l of the non–zero values r' output in the last step. Then with high probability, $l = r$, the period of f. Hence we have computed r in $O(L^3 \log(L))$ operations, subject to the time required to compute f and the probability of error. In the case of Shor's algorithm, where f is exponentiation modulo N, it follows that we can perform quantum modular exponentiation in time $O(L^3 \log(L))$, where $L = \log(N)$. As the reduction of factoring to period finding described in Section 4.2 requires $O(L^3)$ operations we have a quantum algorithm for factoring an integer N in time $O(L^3 \log(L))$. In fact this bound is not tight and, by using faster algorithms for integer arithmetic, Shor [**54**] obtains the bound of $O(L^2 \log(L) \log\log(L))$ mentioned at the begining of this section.

4.6. Probability estimates. We shall need the following result from [**51**].

THEOREM 4.8. *For $r \geq 3$,*

$$\frac{r}{\phi(r)} < e^{\gamma} \log_e \log_e(r) + \frac{2.50637}{\log_e \log_e(r)},$$

where $\gamma = \lim_{n\to\infty}(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} - \log_e(n)) = 0.57721556649\ldots$ *is Euler's constant.*

Recall from Section 4.5 that f is a periodic function from $\mathbb{Z}_q$ to $\mathbb{Z}_N$ of period r, that $L, n, N, r, q \in \mathbb{Z}$, with $1 \leq r < N$, $q = 2^n$, $N^2 \leq q < N$ and $L = \log\lceil N\rceil$. We have a 2 register quantum system with first and second registers of n and L qubits, respectively. The first register is in the state given by (4.10) and the second in state $|b\rangle$, where $b = f(a)$. In addition K_a is the largest integer such that $(K_a - 1)r + a < q$. In the special case of Section 4.4 we measured (4.5) and, with high probability, observed c such that $cr - Ns = 0$, for some s with $\gcd(r, s) = 1$. In the general case we shall show that when we measure (4.10) there is a high probability of observing c such that $-r/2 \leq cr - sq < r/2$, for some integer s such that $\gcd(r, s) = 1$. This will be enough to allow us to compute r.

PROOF OF PROPOSITION 4.7. Recall that, given $s \in \mathbb{Z}$, we write c_s for the unique integer such that $-r/2 \leq c_s r - sq < r/2$. Simple calculations show that $0 \leq c_s < q$ if and only if $0 \leq s < r$ and also that if $c_s = c_t$ then $s = t$. Hence the integers $c_0, \ldots, c_{r-1}$ are distinct and lie in $[0, q)$.

For each s with $0 \leq s < r$ write $\epsilon_s = c_s r - sq$ and define

$$\theta_s = \frac{2\pi\epsilon_s}{q}.$$

The conditions on K_a imply that

$$\frac{q}{r} - 1 < K_a < \frac{q}{r} + 1. \tag{4.12}$$

Therefore, for s such that $0 \leq \epsilon_s < r/2$ and for k such that $0 \leq k < K_a$, we have

$$0 \leq k\theta_s < \frac{\pi k r}{q} < \pi.$$

Hence, for such s, the points $e^{-ik\theta_s}$ lie in the lower half–plane, for $k = 0, \ldots, K_a - 1$ (see Figure 15). Similarly, for s such that $-r/2 \leq \epsilon_s < 0$ the points $e^{-ik\theta_s}$ lie in the upper half–plane, for

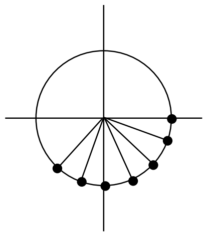


FIGURE 15. The points $e^{-ik\theta_s}$, for s such that $0 \leq \epsilon_s < r/2$.

$k = 0, \ldots, K_a - 1$.

It follows (see Figure 16) that for all s such that $0 \leq s < r$

$$\left|\sum_{k=0}^{K_a-1} e^{-i\theta_s k}\right| \geq \left|\sum_{k=0}^{K_a-1} e^{\frac{-i\pi r k}{q}}\right|. \tag{4.13}$$

This is known as *constructive interference.*

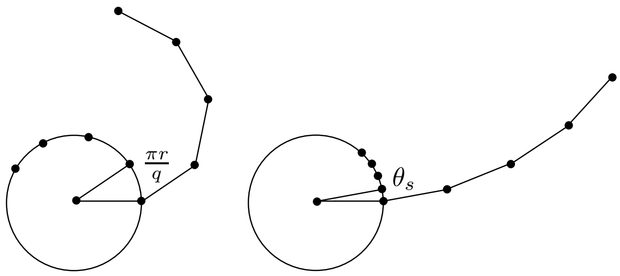


FIGURE 16. Different rates of constructive interference

We shall denote by $p_1(c)$ the probability that c is observed when the first register is measured with respect to the computational basis. The state (4.10) has coefficients given by (4.11) so

$$
\begin{aligned}
p_1(c) &= \frac{1}{K_a q}\left|\sum_{k=0}^{K_a-1} e^{-\frac{2\pi i c(kr+a)}{q}}\right|^2 \\
&= \frac{1}{K_a q}\left|e^{-\frac{2\pi i ca}{q}}\sum_{k=0}^{K_a-1} e^{-\frac{2\pi i ckr}{q}}\right|^2 \\
&= \frac{1}{K_a q}\left|\sum_{k=0}^{K_a-1} e^{-\frac{2\pi i ckr}{q}}\right|^2. \qquad (4.14)
\end{aligned}
$$

Since

$$\frac{-2\pi i c_s r}{q} = \frac{-2\pi i(sq+\epsilon_s)}{q} = -2\pi i s - \frac{2\pi i \epsilon_s}{q},$$

we have

$$e^{(-2\pi i c_s r)/q} = e^{(2\pi i \epsilon_s)/q} = e^{-i\theta_s k}, \qquad (4.15)$$

for $0 \le s < r$.

Combining (4.13), (4.14) and (4.15) we have

$$
\begin{aligned}
p_1(c) &\ge \frac{1}{K_a q}\left|\sum_{k=0}^{K_a-1} e^{-(i\pi rk)/q}\right|^2 \\
&= \frac{1}{K_a q}\left|\frac{\left(e^{-(i\pi r)/q}\right)^{K_a}-1}{e^{-(i\pi r)/q}-1}\right|^2 \\
&= \frac{\sin^2(\pi r K_a/2q)}{K_a q \sin^2(\pi r/2q)}, \qquad (4.16)
\end{aligned}
$$

using the identity $\left|e^{ix}-1\right|^2 = 4\sin^2(x/2)$, for $x \in \mathbb{R}$.

¿From (4.12) we have

$$0 < \frac{\pi}{2}\left(1-\frac{r}{q}\right) < \frac{K_a \pi r}{2q} < \frac{\pi}{2}\left(1+\frac{r}{q}\right) < \pi,$$

so

$$\sin\left(\frac{K_a \pi r}{2q}\right) > \sin\left(\frac{\pi}{2}\left(1+(-1)^d\left(\frac{r}{q}\right)\right)\right), \qquad (4.17)$$

for $d = 1$ or for $d = -1$. As r/q is small we have (using a Taylor series expansion)

$$\sin^2\left(\frac{\pi}{2}\left(1 \pm \frac{r}{q}\right)\right) \geq 1 - \left(\frac{\pi r}{2q}\right)^2 \tag{4.18}$$

and

$$\sin(\pi r/2q) \leq \pi r/2q. \tag{4.19}$$

Combining (4.16), (4.17), (4.18) and (4.19) we see that the probability of observing c_s, for some s such that $0 \leq s < r$, is

$$\begin{aligned} p_1(c_s) &\geq \frac{1}{K_a q}\left(1 - \left(\frac{\pi r}{2q}\right)^2\right)\left(\frac{2q}{\pi r}\right)^2 \\ &= \frac{q}{K_a}\left(1 - \left(\frac{\pi r}{2q}\right)^2\right)\left(\frac{2}{\pi r}\right)^2 \\ &> \frac{rq}{r+q}\left(1 - \left(\frac{\pi r}{2q}\right)^2\right)\left(\frac{2}{\pi r}\right)^2 \\ &\geq \frac{4}{\pi^2 r}\left(1 - \left(\frac{\pi r}{2q}\right)^2\right). \qquad (4.20) \end{aligned}$$

As discussed in Section 4.4 the probability that $s \in \{0, \dots, r-1\}$ is coprime to r is $\phi(r)/r$. Together with (4.9) and (4.20) this yields the first statement of the proposition.

For the final statement we note that if $N \geq 158$ then

$$\frac{4}{\pi^2}\left(1 - \left(\frac{\pi r}{2q}\right)^2\right) \geq \frac{2}{5}.$$

Since there are r distinct integers, $c_0, \dots, c_{r-1}$ this implies that $p_1(c_s) \geq 2/5$. From Theorem 4.8 it follows that if $r \geq 19$ then

$$\frac{\phi(r)}{r} > \frac{1}{4\log_e \log_e(r)} > \frac{1}{4\log_e \log_e(N)},$$

and the final statement of the proposition follows. □

4.7. Efficient implementation of the quantum Fourier transform. As in the case of the Deutsch–Jozsa algorithm it is the complexity of Shor's algorithm which is of interest. Shor showed that his algorithm could be implemented efficiently using small quantum circuits because this is also true of the quantum Fourier transform on $\mathbb{Z}_{2^n}$. In fact this implementation of the quantum Fourier transform is essentially an adaptation of the standard fast Fourier transform technology to quantum computation. Here we'll show that it is possible to implement the quantum Fourier transform on $\mathbb{Z}_n$ using at most n^2 2–qubit gates.

If U is a single qubit quantum gate, then we define the *controlled U–gate*, a unitary transformation of a 2–qubit system, by

$$|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes U.$$

If the first qubit is $|1\rangle$ then U is applied to the second qubit. If it is $|0\rangle$ then the identity I is applied to the second qubit. The controlled U–gate is depicted in a quantum circuit diagram in Figure 17.

Let R_ϕ be the unitary transformation (called a *phase shift*) on a single qubit given by the matrix

$$\begin{pmatrix} 1 & 0 \\ 0 & e^{-i\phi} \end{pmatrix}$$

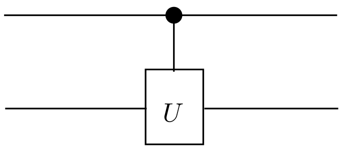


FIGURE 17. A controlled U–gate

Let B_k ($k \in \mathbb{N}$) denote $R_{\pi/2^k}$. Now consider the 4–qubit circuit in Figure 18, which is constructed using controlled B_k–gates. We claim that this performs the quantum Fourier transform on $\mathbb{Z}_{2^4}$. The first Walsh–Hadamard gate applies $W \otimes I^{\otimes 3}$ to $|k_3k_2k_1k_0\rangle$ and results in

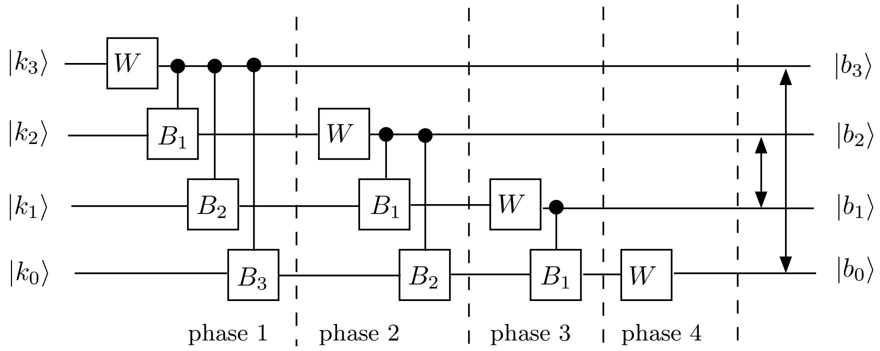


FIGURE 18. A quantum circuit for the 4–qubit quantum Fourier transform

$$W \otimes I^{\otimes 3} |k_3k_2k_1k_0\rangle = \begin{cases} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |k_2k_1k_0\rangle & \text{if } k_3 = 0 \\ \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \otimes |k_2k_1k_0\rangle & \text{if } k_3 = 1 \end{cases}.$$

This can be written as

$$\frac{1}{\sqrt{2}}(|0\rangle + e^{-\pi i k_3} |1\rangle) \otimes |k_2k_1k_0\rangle.$$

The map corresponding to the first controlled–B_1 gate is

$$|0\rangle \langle 0| \otimes I^{\otimes 3} + |1\rangle \langle 1| \otimes R_{\pi/2} \otimes I^{\otimes 2}.$$

After passing through this gate the quantum state becomes

$$\begin{cases} \frac{1}{\sqrt{2}} \left(|00k_1k_0\rangle + e^{-\pi i k_3}|10k_1k_0\rangle\right) & \text{if } k_2 = 0 \\ \frac{1}{\sqrt{2}} \left(|01k_1k_0\rangle + e^{-\pi i k_3} |1\rangle \otimes e^{-\pi i/2} |1\rangle \otimes |k_1k_0\rangle\right) & \text{if } k_2 = 1 \end{cases}$$

which can be written as

$$\frac{1}{\sqrt{2}} \left(|0\rangle + e^{-\pi i \left(k_3 + \frac{k_2}{2}\right)} |1\rangle\right) \otimes |k_2k_1k_0\rangle.$$

Using similar calculations, after phase 1 the state becomes

$$\frac{1}{\sqrt{2}} \left(|0\rangle + e^{-\pi i \left(k_3 + \frac{k_2}{2} + \frac{k_1}{4} + \frac{k_0}{8}\right)} |1\rangle\right) \otimes |k_2k_1k_0\rangle.$$

That is, we have

$$|k\rangle \mapsto \frac{1}{\sqrt{2}} \left(|0\rangle + e^{-\frac{2\pi k i}{16}} |1\rangle\right) \otimes |k_2k_1k_0\rangle,$$

where $|k\rangle = |k_3k_2k_1k_0\rangle$, and we can further rewrite the right hand side as

$$\frac{1}{\sqrt{2}} \sum_{b_0=0}^{1} e^{-\frac{2\pi i k b_0}{16}} |b_0k_2k_1k_0\rangle.$$

Similarly, after phase 2 we obtain

$$\frac{1}{\sqrt{2^2}} \sum_{b_0=0}^{1} \sum_{b_1=0}^{1} e^{-\frac{2\pi i(b_0+2b_1)}{16}} |b_0 b_1 k_1 k_0\rangle$$

and after phase 4 the state becomes

$$\frac{1}{\sqrt{2^4}} \sum_{b_0=0}^{1} \sum_{b_1=0}^{1} \sum_{b_2=0}^{1} \sum_{b_3=0}^{1} e^{-\frac{2\pi i(b_0+2b_1+4b_2+8b_3)}{16}} |b_0 b_1 b_2 b_3\rangle.$$

Swapping qubits $3 \leftrightarrow 0$ and $1 \leftrightarrow 2$, and setting $|b\rangle = |b_3 b_2 b_1 b_0\rangle$, gives

$$\frac{1}{\sqrt{2^4}} \sum_{b=0}^{2^4-1} e^{-\frac{2\pi i k b}{2^4}} |b\rangle = \mathcal{Q}_{2^4} |k\rangle \,,$$

the quantum Fourier transform on $\mathbb{Z}_{2^4}$ applied to $|k\rangle \in \mathbb{Z}_{2^4}$. This generalises in a straightforward manner to give the quantum Fourier transform of $\mathbb{Z}_{2^n}$, for arbitrary n. Note that all the gates used are 2–qubit gates. In the general case the number of gates used is $(n+1)n/2 \leq n^2$, as claimed.

4.8. The continued fractions algorithm. A *(finite) continued fraction* is an expression of the form

$$a_0 + \cfrac{1}{a_1 + \frac{1}{a_2 + \frac{1}{\cdots + \frac{1}{a_n}}}}$$

where $a_0 \in \mathbb{Z}$, $a_i \in \mathbb{N}$, for each $i > 0$, and $n \geq 0$. This finite continued fraction is denoted $[a_0, \ldots, a_n]$ and clearly represents a unique rational number. Conversely, using the Euclidean algorithm, it can be seen that a positive rational number can be expressed uniquely as a finite continued fraction $[a_0, \ldots, a_n]$, with $a_n > 1$ (see [**28**] or [**40**]).

Example 4.9. As

$$\begin{aligned} 125 &= 3 \cdot 37 + 14 \\ 37 &= 2 \cdot 14 + 9 \\ 14 &= 1 \cdot 9 + 5 \\ 9 &= 1 \cdot 5 + 4 \\ 5 &= 1 \cdot 4 + 1 \end{aligned}$$

we have

$$\begin{aligned} \frac{125}{37} &= 3 \cdot \frac{37}{37} + \frac{14}{37} \\ &= 3 + \frac{1}{\frac{37}{14}} \\ &= 3 + \frac{1}{2 \cdot \frac{14}{14} + \frac{9}{14}} \\ &= 3 + \frac{1}{2 + \frac{1}{\frac{14}{9}}} \\ &\vdots \\ &= 3 + \cfrac{1}{2 + \frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{4}}}}} \end{aligned}$$

and so the continued fraction representing 125/37 is $[3, 2, 1, 1, 1, 4]$.

The j^{th} *convergent* of the continued fraction $[a_0,\ldots,a_n]$ is the expression $[a_0,\ldots,a_j] = p_j/q_j$, say. Continuing the above example the convergents of $125/37$ are $[3] = 3$, $[3,2] = 7/2$, $[3,2,1] = 10/3$, $[3,2,1,1] = 17/5$, $[3,2,1,1,1] = 27/8$, and $125/37$. Clearly we may compute the convergents of a given rational number using the Euclidean algorithm as in the above example and this allows us to make use of Theorem 4.6 in Shor's algorithm. The complexity of this algorithm, known as the *continued fraction algorithm* is the same as that of the Euclidean algorithm: that is $O(L^3)$ operations are required to compute the continued fraction of p/q, where $L = \max\{\log(p), \log(q)\}$.

4.9. The hidden subgroup problem. The problems of factoring integers and finding the period of functions may both be regarded as instances of the "hidden subgroup problem" which we discuss in this section.

Let G be a group and X a set and let $f : G \to X$ be a function. Assume that there is a subgroup $K \le G$ such that

(i) f restricted to gK is constant, for all $g \in G$, and
(ii) if $gK \ne hK$ then $f(g) \ne f(h)$.

Then we say that f is a *hidden subgroup function* which *hides* the subgroup K and that K is the *hidden subgroup* of f. The *hidden subgroup problem* is to find the hidden subgroup of a given hidden subgroup function. That is, to solve the problem we must find a generating set for K. (In some weaker versions of the problem it is only required that random elements of K are found.) We are concerned here with the complexity of this problem and, in particular, whether or not it can be solved more quickly using quantum rather than classical techniques. We shall say that the hidden subgroup problem can be solved *efficiently* if there is an algorithm which outputs generators of K in time bounded by some polynomial in $\log|G|$. This is a simplification, as in practice it is necessary to consider how G is encoded and the effect of this encoding on the complexity of the problem. However for the purposes of the present brief discussion it is enough to assume that our quantum system has access to elements of G in some appropriate form.

First of all we point out that the hidden subgroup problem has a number of intrinsically interesting special cases. For example, a periodic function f with period r, in the sense of Section 4.1, on a cyclic group $C = \langle x\rangle$, hides the subgroup $\langle x^r\rangle$. Therefore, period finding, and hence factoring of integers are, as claimed, particular cases of this problem.

Another problem which may be described in this way is that of finding discrete logarithms. Given a cyclic group G of order n generated by an element g the *discrete logarithm problem* is, given $a \in G$, to find the least positive integer r such that $a = g^r$. To formulate this as a hidden subgroup problem consider the function $f : \mathbb{Z}_n \times \mathbb{Z}_n \to \mathbb{Z}_n$ given by $f(x,y) = g^x a^{-y}$. This is a homomorphism with kernel the subgroup K generated by $(r,1) \in \mathbb{Z}_n \times \mathbb{Z}_n$. Therefore f hides the subgroup K. Clearly finding the generator of K gives us r. In fact if we can obtain any element $(s,t) \in K$ then we can compute $r = s/t$ (in time $O(\log^2(n))$) as long as we know that K is generated by an element of the form $(r,1)$. The U.S. Digital Signature Algorithm is based on the assumption that no polynomial algorithm is known for the discrete logarithm problem [**42**]. Details of an efficient quantum algorithm for this problem may be found in [**45**].

As a third example consider a group G acting on a set X. If $x \in X$ then we may define a function $f : G \to X$ by $f(g) = g \cdot x$, for $g \in G$. Then f hides the stabiliser of x.

As a further example we mention that the graph isomorphism problem may be viewed as a special case of the hidden subgroup problem (see [**33**, Section 6]). No polynomial time algorithm for the graph isomorphism problem is known but on the other hand it is not known to be NP–complete. (A problem is NP if there is a classical non–deterministic polynomial time algorithm for its solution. A problem is NP–complete if every problem which is NP may be reduced, efficiently, to this problem. See [**47**] for further details.) As a problem which seems likely to lie outside the class of problems classically solvable in polynomial time and which also seems unlikely to be NP–complete, the graph isomorphism problem is a good test case for the power of quantum computation. In its formulation as a hidden subgroup problem it becomes a question of finding a hidden subgroup of a permutation group (of degree twice the number of vertices of the graphs in question). An efficient quantum algorithm for the hidden subgroup problem in permutation groups would therefore give rise to an efficient quantum algorithm for the graph isomorphism problem. However, at the time of writing, there are very few non–Abelian groups for which polynomial time quantum algorithms for the hidden subgroup problem have been found.

The first quantum algorithm for a hidden subgroup problem appears to be Deutsch's algorithm, where the hidden subgroup of $\mathbb{Z}_2$ is either trivial or the entire group. However the subject begins in earnest with Simon's algorithm, [**55**] and [**56**], for a restricted case of the hidden subgroup problem in $\mathbb{Z}_2^n$. Simon's algorithm uses the Walsh–Hadamard transform on $\mathbb{Z}_2^n$ to extract information from a quantum state, but the hidden subgroup here must have order 2. Shor [**53**] realised that it was possible to implement the quantum Fourier transform for $\mathbb{Z}_{2^n}$ and used it instead of the Walsh–Hadamard transform to generalise Simon's algorithm. This resulted in the factorisation algorithm described above. Subsequently methods for implementing and applying the quantum Fourier transform to a wider class of Abelian groups were developed by a number of people including Shor [**54**], Cleve [**11**], Coppersmith [**13**] and Deutsch. Kitaev implemented the quantum Fourier transform [**38**] for arbitrary finitely generated Abelian groups and used it to construct his *phase estimation* algorithm which finds eigenvalues of unitary transformations. More precisely, Kitaev's phase estimation algorithm, given a unitary transformation and one of its eigenvectors $|u\rangle$, will return a value ϕ, where $e^{2\pi i\phi}$ is the eigenvalue corresponding to $|u\rangle$ (see for example [**45**]). Kitaev used the phase estimation algorithm to solve the problem of finding stabilisers, as described above, where G is a finitely generated Abelian group, and showed how this gives rise to efficient algorithms for factoring integers and for the discrete logarithm problem. Mosca and Ekert [**44**] have shown that the phase estimation algorithm can be used to solve the general hidden subgroup problem in finitely generated Abelian subgroups.

Moving away from Abelian groups Ettinger and Hoyer construct an algorithm which solves the hidden subgroup problem in the finite dihedral group G, using at most a polynomial (in $\log|G|$) number of calls to the unitary transformation U_f simulating f (as in Section 4.4). However their algorithm requires exponential time to interpret the output. That is, the part of the algorithm analagous to the continued fractions post processing in Shor's algorithm requires exponentially many operations. Ettinger, Hoyer and Knill [**17**] generalise this to show that there is a quantum algorithm for the hidden subgroup problem in an arbitrary finite group G which requires $O(\log|G|)$ calls to U_f. However they do not give explicit implementation of the measurements required, and the post processing part of the algorithm is again exponential. Püschel, Rötteler and Beth [**50**], [**52**] have implemented the quantum Fourier transform for the wreath product $\mathbb{Z}_2^n \wr \mathbb{Z}_2$ and hence solve the hidden subgroup problem efficiently in these groups. Hallgren, Russell and Ta-Shma [**26**] have shown that the special case of the hidden subgroup problem where K is a normal subgroup of a finite group G can be solved efficiently on a quantum computer. Their algorithm uses the Fourier transform for an arbitrary finite group to distinguish K. Ivanyos, Magniez and Santha [**30**] generalise [**52**] by constructing polynomial time quantum algorithms for the hidden subgroup problem in specific finite groups: namely groups having small commutator subgroups and groups which have an elementary Abelian normal 2–subgroup of small index or with cyclic factor group. (Here *small* means of order polynomial in $\log|G|$.)

Friedl, Ivanyos, Magniez, Santha and Sen [**21**] complete and generalise much of the above by extending these last results to solvable groups satisfying the following condition on their commutator subgroups. A finite Abelian group A is said to be *smooth* if it is the direct sum of an elementary Abelian p–group, for some prime p, with a group of order polynomial in $|A|$. A finite solvable group G is said to be *smoothly* solvable if the Abelian factors of its derived series are smooth Abelian groups. In [**21**] efficient quantum algorithms are constructed for the hidden subgroup problem in finite solvable groups which have smoothly solvable commutator subgroups. These include semidirect products of the form $\mathbb{Z}_p^k \rtimes \mathbb{Z}_2$, where p is a prime (which reduce to finite dihedral groups of order $2p$ when $k = 1$) and groups of upper triangular matrices of bounded dimension over a finite field. In fact their work builds on quantum algorithms for solvable groups developed by Watrous [**60**] and Cheung and Mosca [**10**].

Moore, Rockmore, Russell and Schulman [**43**] show that q–hedral groups $\mathbb{Z}_p \rtimes \mathbb{Z}_q$ have quantum efficiently solvable hidden subgroup problem, when $q = (p-1)/g(\log(p))$, for some polynomial g. They also prove that quantum efficiency of the hidden subgroup problem is closed under taking certain extensions, as follows. Suppose that there is an efficient quantum algorithm for the hidden subgroup problem in the group H. Let G be a group with normal subgroup N such that $G/N = H$ and $|N| = g(\log |H|)$, for some polynomial g. Then there is an efficient quantum algorithm for the hidden subgroup problem in G.

Hallgren [**25**] considers the problem of finding the period of a periodic function f from the group $\mathbb{R}$ to a set X, where the period may be irrational. Hallgren's quantum algorithm runs in time polynomial in an appropriatly defined input size. This gives rise to efficient algorithms for a number of computational problems of algebraic number theory: the solution of Pell's equation, the pricipal ideal problem and determination of the class group. No efficient classical algorithms are known for any of these problems. (A full and self–contained exposition of Hallgren's algorithm and it's application to number theoretic problems may be found in [**34**].) In [**41**] Lomonaco and Kauffman consider the hidden subgroup problem in $\mathbb{R}$ and various other Abelian groups which are not finitely generated.

5. Grover's algorithm

5.1. Overview. Grover's algorithm [**24**] operates in a quite different way to Shor's. The basic problem is an unstructured search: We are given an N element set X and a map $P : X \to \{0,1\}$ and are required to find $x \in X$ such that $P(x) = 1$. We call a value of x such that $P(x) = 1$ a *solution* to the search problem. In the first four parts of this Section we consider the algorithm for the case where we know that there are exactly $M \geq 1$ solutions. Then, in Section 5.5, we show how techniques developed by Brassard, Høyer and Tapp can be used to remove this constraint. In Section 5.2 we describe the algorithm and then in Sections 5.3 and 5.4 we explain why it works.

No extra information is known about P, we merely have an oracle to evaluate $P(x)$ for a given $x \in X$. Classically, the best algorithm (exhaustive testing) requires $N - M + 1$ evaluations to find a solution x with certainty, since the first $N - M$ elements tested may be non–solutions. Probabilistically, we would expect to find a result after $N/2M$ evaluations. In contrast, Grover's quantum algorithm performs the search in time $O(\sqrt{N/M})$ on a quantum computer.

The idea of the algorithm is, roughly speaking, the following. Suppose that N has size 2^n and, as before, we prepare the standard superposition of all possible outputs (entangled with inputs):

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \otimes |P(x)\rangle\,.$$

We wish to find a state $|x\rangle \otimes |1\rangle$ for some x. By direct measurement at this stage, there is only a probability of $M/\sqrt{2^n}$ of finding such a state. In the worst case when there is only one solution this falls to $1/\sqrt{2^n}$. The strategy is to increase the amplitude of vectors of the form $|x\rangle \otimes |1\rangle$ and decrease the amplitude of those of the form $|x\rangle \otimes |0\rangle$, until the state approximates

$$\frac{1}{\sqrt{M}} \sum_{i=1}^{M} |x_i\rangle \otimes |1\rangle\,, \tag{5.1}$$

where the solution set is $\{x_1, \ldots, x_M\}$. Measuring this altered state then gives a solution with high probability.

5.2. A circuit for Grover's algorithm. Let us consider how this strategy may be carried out in practice. We assume, for simplicity, that $N = 2^n$, for some positive integer n and that we know in advance that there are exactly M solutions, where $M \geq 1$. The algorithm uses the standard oracle U_P for the function P which, as in Section 2.10 maps $|x\rangle \otimes |y\rangle$ to $|x\rangle \otimes |P(x) \oplus y\rangle$. Thus the quantum system underlying the algorithm consists of a first register of n qubits and second register, called the *oracle workspace*, of a single qubit. As in the description of Deutsch's algorithm in Section 3.3, we begin with the state $|0\rangle^{\otimes n} \otimes |1\rangle$ to which is applied $W^n \otimes W$ followed by U_P. As in Section 3.3, the first register then contains

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{P(x)} |x\rangle = \mathcal{D}(P), \tag{5.2}$$

as defined in (3.1). Note that the oracle maps the state $|x\rangle \otimes |w\rangle$ to $(-1)^{P(x)}\,|x\rangle \otimes |w\rangle$, so we may regard the second register as unchanged and the amplitude of the first register as multiplied by -1 if and only if x is a solution.

We now need to magnify the amplitudes of the vectors $|x\rangle$ where x is a solution. This is accomplished using *inversion about the mean* which may be defined as the unitary transformation $F = W_n T W_n$, where T is the *conditional phase shift* operator given by

$$T\,|0\rangle = |0\rangle \quad \text{and} \quad T\,|x\rangle = -\,|x\rangle\,, \quad \text{for all} \quad x \neq 0.$$

We shall discuss inversion about the mean in more detail in Section 5.3, for the time being assuming that it does what we require of it: that is to increase negative amplitudes and decrease positive

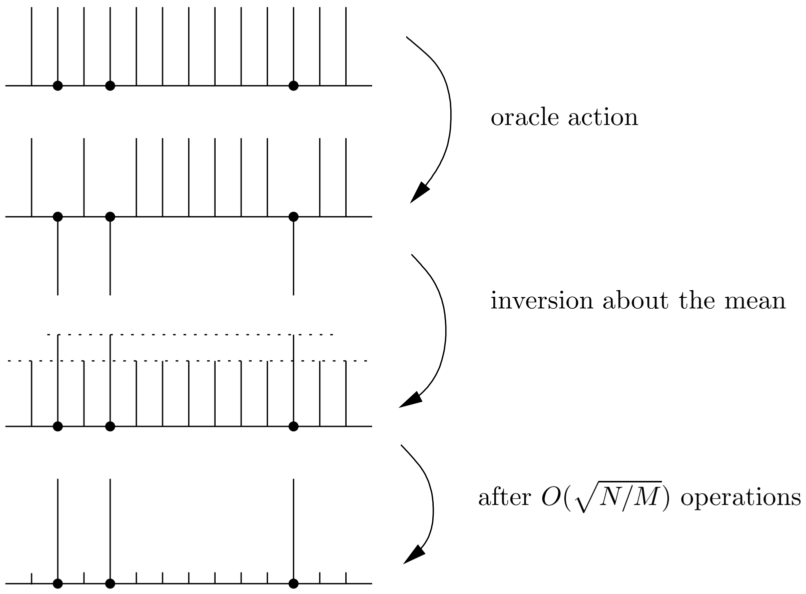


Figure 19. Operation of Grover's algorithm

ones. With this assumption apply F to the state $\mathcal{D}(P)$ of (5.2) (that is we apply $F \otimes I$ to our quantum system). As shown in Figure 19 we then repeat the process, applying U_P followed by F untill the amplitudes of the solutions approach $1/\sqrt{M}$ and the amplitudes of all other basis vectors approach zero, as in (5.1). We call the composite function $\mathcal{G} = (F \otimes I) \circ U_P$ the *Grover operator*. The question is how do we know how many iterations of $\mathcal{G}$ to allow before halting. It will become apparent in Section 5.4, where we answer this question, that we must choose the number R of iterations carefully, as the amplitudes of the solutions do not approach a steady state but rather oscillate, so too many iterations will be as bad as too few. As we shall see, the required number of iterations is $O(\sqrt{N/M})$.

Grover's algorithm can be depicted using the circuit diagrams in Figures 20 and 21.

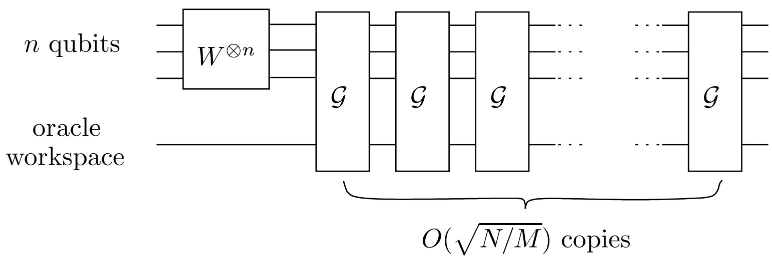


Figure 20. The quantum circuit for Grover's algorithm

5.3. Inversion about the mean. We now explain why the operation F of Section 5.1 is called "inversion about the mean" and behaves as shown in Figure 19. It is easy to verify that the conditional phase shift operator T satisfies

$$T = 2|0\rangle\langle 0| - I.$$

Inversion about the mean is then given by

$$F = W^{\otimes n}(2|0\rangle\langle 0| - I)W^{\otimes n}. \tag{5.3}$$

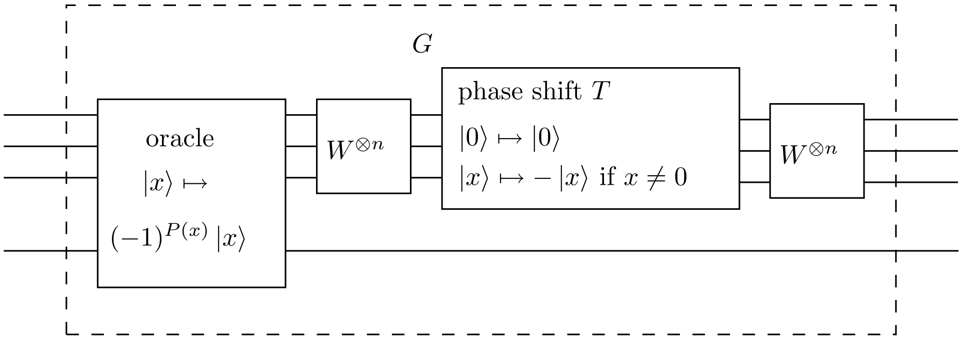


FIGURE 21. Decomposition of the Grover operator

If we let

$$|\psi\rangle = W^{\otimes n}|0\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

then we have

$$\langle\psi| = \langle 0|W^{\otimes n} = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \langle x|.$$

It follows, from (5.3), that

$$F = 2|\psi\rangle\langle\psi| - I. \tag{5.4}$$

Now consider the action of this operator on a general quantum state. We have

$$(2|\psi\rangle\langle\psi| - I) \sum_k \alpha_k |k\rangle = 2 \sum_k \alpha_k |\psi\rangle\langle\psi|k\rangle - \sum_k \alpha_k |k\rangle.$$

Now,

$$\begin{aligned}
\sum_k \alpha_k |\psi\rangle\langle\psi|k\rangle &= \sum_{k=0}^{N-1} \alpha_k \cdot \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \langle x|k\rangle |\psi\rangle \\
&= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \alpha_k |\psi\rangle \\
&= \frac{1}{\sqrt{N}} \sum_k \alpha_k \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \\
&= A \sum_{x=0}^{N-1} |x\rangle,
\end{aligned}$$

where

$$A = \frac{1}{N} \sum_k \alpha_k$$

is the average (mean) of $\{\alpha_k\}$. So

$$F\left(\sum_{k=0}^{N-1} \alpha_k\,|k\rangle\right) = \sum_{k=0}^{N-1} (2A - \alpha_k)|k\rangle.$$

Therefore F acts by reflecting the amplitudes α_k about their mean value A.

5.4. The Grover operator as a rotation. Suppose that there are $M \geq 1$ solutions in a search set of size N. Let

$$|a\rangle = \frac{1}{\sqrt{N-M}} \sum_{P(x)=0} |x\rangle, \quad |b\rangle = \frac{1}{\sqrt{M}} \sum_{P(x)=1} |x\rangle.$$

The initial state of the system is

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

which can be written as

$$|\psi\rangle = \sqrt{\frac{N-M}{N}}|a\rangle + \sqrt{\frac{M}{N}}|b\rangle. \tag{5.5}$$

We claim that the Grover operator keeps the quantum state in the plane spanned by $|a\rangle$ and $|b\rangle$, i.e. that the subspace $S = \text{span}\{|a\rangle, |b\rangle\}$ of the quantum system is invariant under operation of $\mathcal{G}$. Since the oracle U_P acts on both the first and second registers it is convenient to define an operator O of the first register by $U_P(|x\rangle \otimes |w\rangle) = (O\,|x\rangle) \otimes |w\rangle$. This is possible since U_P leaves $|w\rangle$ unchanged. Thus O determins the action of the oracle on the first register, given that the second register is in state $|w\rangle$. Now let $G = F \circ O$. Then the operation of $\mathcal{G}$ on the first register is determined by G. We aim to show that S is invariant under G.

We first consider the action of the oracle on S. We have $O|a\rangle = |a\rangle$ and $O|b\rangle = -|b\rangle$. Thus

$$O(\alpha|a\rangle + \beta|b\rangle) = \alpha|a\rangle - \beta\,|b\rangle \in \text{span}\{|a\rangle\,, |b\rangle\},$$

so S is invariant under the action of the oracle. Geometrically, $O|_S$ is a reflection in the line through the origin defined by $|a\rangle$ (by which we mean the set of points $\alpha\,|a\rangle$, for $\alpha \in \mathbb{C}$).

Next we consider the action of inversion about the mean, that is the operator F, on S. From (5.4) we have

$$F\left(\alpha|a\rangle + \beta|b\rangle\right) \quad = \quad 2\alpha\langle\psi|a\rangle|\psi\rangle + 2\beta\langle\psi|b\rangle|\psi\rangle - \alpha|a\rangle - \beta|b\rangle.$$

Since $|\psi\rangle \in S$ it follows that S is invariant under F and, with the above, this implies that S is invariant under G, as required.

Moreover, it is easy to see that $F|\psi\rangle = |\psi\rangle$ and that if $|\phi\rangle$ is orthogonal to $|\psi\rangle$ then $F|\phi\rangle = -|\phi\rangle$. Thus $F|_S$ is a reflection in the line through the origin defined by $|\psi\rangle$. Thus $G|_S$, being the composition of two reflections in lines through the origin, is a rotation about the origin of the plane S. To find the angle of rotation note that since $1 \leq M \leq N$ we have $0 \leq \sqrt{N-M/N} < 1$, so that there exists $\theta \in \mathbb{R}$ such that $0 < \theta \leq \pi$ and $\cos(\theta/2) = \sqrt{N-M/N}$. From (5.5) we have therefore

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|a\rangle + \sin\left(\frac{\theta}{2}\right)|b\rangle.$$

Thus $G|_S$ is an anticlockwise rotation about the origin, through an angle θ, as shown in Figure 22. Hence

$$G|\psi\rangle = \cos\left(\frac{3\theta}{2}\right)|a\rangle + \sin\left(\frac{3\theta}{2}\right)|b\rangle$$

and in general,

$$G^k|\psi\rangle = \cos\left(\frac{2(k+1)\theta}{2}\right)|a\rangle + \sin\left(\frac{2(k+1)\theta}{2}\right)|b\rangle.$$

If we rotate $|\psi\rangle$ through $\cos^{-1}\left(\sqrt{M/N}\right)$ radians then we obtain a state close to the desired vector $|b\rangle$. Measuring this state we will, with high probability, observe x such that $P(x) = 1$, that is a

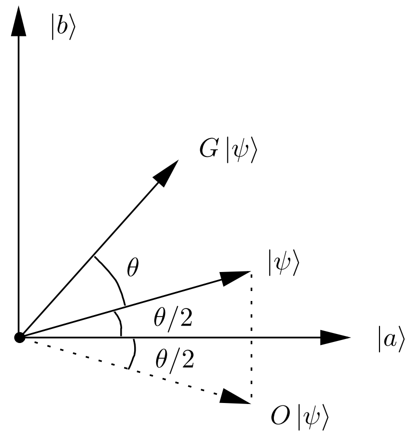


FIGURE 22. Geometric interpretation of the Grover operator

solution to the search problem. Thus the number of times we should iterate the Grover operator is given by

$$R = \left\lfloor \frac{\cos^{-1}\left(\sqrt{M/N}\right)}{\theta} \right\rfloor .$$

If $M \leqslant N/2$ then $\theta/2 \geqslant \sin(\theta/2) = \sqrt{M/N}$. Thus we obtain, in this case,

$$\frac{\pi}{4}\sqrt{\frac{N}{M}}$$

as an upper bound for R. Note that if we iterate approximately $\pi\sqrt{N/M}/2$ times, then we have rotated back almost to $-|a\rangle$, and the probability of obtaining a solution is much worse again. So determining the appropriate number of iterations is a delicate matter. In the case where $M = 1$ as in Grover's original paper, the number of iterations required is approximately $\pi\sqrt{N}/4$.

It can be shown that if $M \geqslant N/2$, i.e. more than half of the elements of the search set are solutions, then the number of iterations required *increases* with M! (See [**45**].) However, if we know in advance that $M \geq N/2$ then sampling the set X at random, and checking for a solution using the oracle, we'll find a solution with probability at least 1/2, with only one call to the oracle. Even when it is not known in advance whether or not $M \geq N/2$, by doubling the size of the first register and padding it with non–solutions we can assume, at very low cost, that in fact $M \leq N/2$ and so use the above bound R on the number of iterations required (see [**45**] for details). In conclusion the number of Grover operations $\mathcal{G}$ required for a solution to be found with high probability is $O(\sqrt{N/M})$.

We show in the next section, how we can estimate M when it is not known in advance. Also, note the following.

(i) If the required probability of error is less than a given constant then the oracle in Grover's algorithm can also be implemented in time $O(\sqrt{N})$ to search an unstructured database. (See [**45**].)
(ii) The time complexity of unstructured quantum search algorithms is known to be $\Omega(\sqrt{N})$ (see [**5**]).

5.5. The Brassard–Høyer–Tapp counting algorithm. In [**6**], Brassard, Høyer and Tapp describe general conditions under which Grover's techniques may be used. They also give a novel method for approximating the number of values x for which a boolean function P satisfies

$P(x) = 1$. Some related work, on the eigenvalue analysis of the operator described in this section, is also due to Mosca. A more developed version by all four authors above appears in [**7**]. Given a system in state $|\psi\rangle$ the idea is to find a superposition of $G^m |\psi\rangle$, for all values of m in a given range, where ψ and G are defined in Section 5.4. Since G is a rotation, the above state will be periodic in m. So as in Shor's algorithm, we may apply the quantum Fourier transform of m which will find the period of the above state (efficiently). ¿From this we can estimate the number of solutions (without actually finding any!) and hence the number of rotations required for Grover's algorithm above to find a solution with high probability.

Given the Grover operator G for P, define the *counting operator* for P to be

$$C : |m\rangle \otimes |\psi\rangle \mapsto |m\rangle \otimes G^m |\psi\rangle .$$

Assume that the value m can take any value in $\{0, \ldots, R = 2^r - 1\}$ Then the quantum circuit in Figure 23 will estimate the period of $C \circ (W \otimes W) |0\rangle \otimes |0\rangle$. The accuracy to which we can

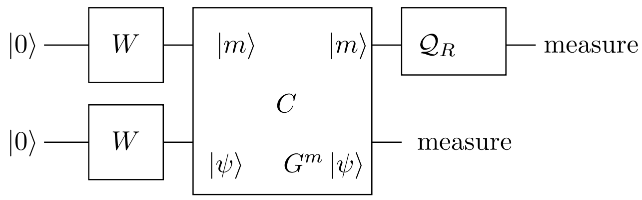


FIGURE 23. Brassard, Høyer and Tapp's counting circuit

estimate this period depends on how large a value of R we take. More precisely, it is shown in [**6**] (theorem 5) that if $t = |P^{-1}(1)| \leqslant N/2$ and c is the output of this circuit, then

$$|c - t| < \frac{2\pi}{R}\sqrt{tN} + \frac{\pi^2}{R^2}N \text{ with probability at least } \frac{8}{\pi^2}.$$

6. Watrous' algorithms for solvable groups

Watrous [**59**],[**60**] has recently produced some group-theoretic work in quantum computing which has quite a different nature to the hidden subgroup problem. In particular, he describes a Monte Carlo quantum algorithm which finds the order of a finite solvable group in polynomial time. This uses some of the techniques of Shor's algorithm, but also provides some new methods which depend critically on group structure.

The group G which is input to the algorithm is given as a finite black box group [**4**]. This means that we assume the existence of some description of G as a set of binary strings of fixed length n. Multiplication and inversion of elements are performed, each at unit cost, by an oracle which knows this description. The input to the algorithm is a finite generating set for G, together with the oracle.

The representation of the elements of G as binary strings gives us a natural association between the elements of G and a subset of the basis elements of a 2^n dimensional vector space (that is, a register of n qubits). The notation $|g\rangle$ will be used to denote the basis element associated with the group element g. One useful byproduct of Watrous' algorithm (and a vital step within the algorithm) is the computation of a uniform superposition

$$\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle$$

in one of the registers.

Classical Monte Carlo algorithms are already known which compute a polycyclic generating set for a finite solvable group in polynomial time [**3**]. Hence we can assume that our starting point for the algorithm is a generating set $g_1, \ldots g_k$ for which the subgroups $H_j = \langle g_1, \ldots g_j\rangle$ form a subnormal series (that is $H_j \lhd H_{j+1}$, for each j). In this case, each quotient group H_j/H_{j-1} is cyclic, of order r_j (we define H_0 to be the identity subgroup). The set of products of the form

$$g_k^{a_k} \ldots g_1^{a_1},$$

with each a_j ranging from 0 to $r_j - 1$, provides a normal form, and the group order is the product of the integers r_j. The problem is reduced to finding the r_j, as the orders of the cyclic factor groups.

The algorithm works up the chain of the subgroups H_j, and so splits naturally into k steps. We shall describe just one such step, the j-th step, which computes r_j.

6.1. Step j of the algorithm.

The j-th step of the algorithm splits into two phases. The first phase computes r_j, as the period of the function

$$f_j : \mathbb{Z} \to H_j/H_{j-1}$$

defined by the rule

$$f_j(a) = g_j^a H_{j-1}$$

using a fairly straightforward variant of Shor's algorithm. The second phase, which involves some rather intricate calculation, uses knowledge of the integer r_j to compute a uniform superposition of the elements of H_j. This superposition is then used as input for the first phase of the next step. We shall describe both phases below. To make the notation easier, from now on we shall abbreviate r_j to r, g_j to g and H_{j-1} to H. In this case, H_j is equal to $\langle g\rangle H$. Following Watrous we use the notation $|H\rangle$ for the superposition

$$\frac{1}{\sqrt{|H|}} \sum_{h \in H} |h\rangle$$

and analogously $|\langle g\rangle H\rangle$ for the superposition over $\langle g\rangle H$.

The computation takes place in a large tensor product space with a number of different registers, of two different types, which (like Watrous) we call R registers and A registers. The R registers are used to store superpositions of basis vectors indexed by the elements of G. The A registers are used to store superpositions of basis vectors indexed by the integers in some finite range $0, \dots M-1$ (We call the set of such integers $\mathbb{Z}_M$. Note that this is a subset of $\mathbb{Z}$ which is not quite the same as the cyclic group of integers mod M.) How big M needs to be differs between the two phases.

Recall that the quantum Fourier transform $\mathcal{Q}_M$ acts on an A register as

$$\mathcal{Q}_M : |a\rangle \mapsto \frac{1}{\sqrt{M}} \sum_{b \in \mathbb{Z}_M} e^{-2\pi i ab/M} |b\rangle .$$

(In fact, Watrous uses the convention which calls this the inverse transform, but for consistency we use the notation of the rest of this article.) Group multiplication is provided by the unitary transformation U_G which acts on pairs of R registers as

$$U_G : |g\rangle \otimes |h\rangle \mapsto |g\rangle \otimes |gh\rangle .$$

A related unitary transformation V_G^g acts on a pair of registers, one an A register the other an R register, as

$$V_G^g : |a\rangle \otimes |h\rangle \mapsto |a\rangle \otimes |g^a h\rangle$$

and is a vital ingredient to the variant of Shor's algorithm.

6.2. The first phase of step j. We shall describe this phase only briefly, aiming only to exhibit it as a variant of Shor's algorithm, which is described in detail in Section 4.

Here the A register needs to cover integers in the range $\mathbb{Z}_N$, where N is 'large enough', basically $2^{2n+O(log 1/\epsilon)}$, where ϵ is to bound the probability of error. At the beginning of the j-th step the A register contains $|0\rangle$, and the R register contains the uniform superposition $|H\rangle$.

We apply the inverse quantum Fourier transform $\mathcal{Q}_N^{-1}$ to the A register, then V_G^g to the pair of registers, then the quantum Fourier transform $\mathcal{Q}_N$ to the A register.

$$|0\rangle \otimes |H\rangle$$

$$\downarrow \mathcal{Q}_N^{-1} \otimes I$$

$$\frac{1}{\sqrt{N}} \sum_{a \in \mathbb{Z}_N} |a\rangle \otimes |H\rangle$$

$$\downarrow V_G^g$$

$$\frac{1}{\sqrt{N}} \sum_{a \in \mathbb{Z}_N} |a\rangle \otimes |g^a H\rangle$$

$$\downarrow \mathcal{Q}_N \otimes I$$

$$\frac{1}{N} \sum_{a \in \mathbb{Z}_N} \sum_{b \in \mathbb{Z}_N} e^{-2\pi i ab/r} |b\rangle \otimes |g^a H\rangle$$

Observing A gives some b in $\mathbb{Z}_N$ which is (with high probability) a good approximation for κ/r_j, for κ random.

¿From now on we follow Shor's standard procedure, as described in Section 4. Using continued fractions, we can find integers u, v such that $u/v = \kappa/r$, and with high probability u and v are coprime. We repeat this process an appropriate number of times to give the required bound on the probability of error, and then find r as the lcm of the v values.

6.3. The second phase of step j. The second phase has to extend the uniform superposition of H in the R register to a uniform superposition of $\langle g \rangle H$, using the knowledge of r acquired during the first phase. The aim of this section is to explain that computation of $|\langle g \rangle H\rangle$ from $|H\rangle$.

In fact we produce several copies of $|\langle g \rangle H\rangle$ from several copies of $|H\rangle$. More precisely, we use $m = k - j + 2$ A registers and m R registers to produce $m - 1$ copies of $|\langle g \rangle H\rangle$ from m copies of $|H\rangle$. At the beginning of this calculation, each of the m R registers contains the superposition $|H\rangle$. At the end, $m - 1$ of them contain $|\langle g \rangle H\rangle$, and can be used in the next step, and the other must be discarded.

In this phase the A register is used to store the integers in $\mathbb{Z}_r$. During the phase we use the inverse quantum Fourier transform $\mathcal{Q}_r^{-1}$, the transformation V_G^g, and U_G.

The computation has two stages. During the first stage, we work with pairs of registers, each pair consisting of one A register and one R register. We do the same computation with each pair, so to describe this stage we need only say what happens to one such pair. At the beginning of the stage the A register is set to $|0\rangle$ and the R register to $|H\rangle$. First we apply the inverse quantum Fourier transform $\mathcal{Q}_r^{-1}$ to the contents of the A register only, then we apply V_G^g to the pair of registers, and then $\mathcal{Q}_r^{-1}$ again to the A register. Then we observe the A register, which projects onto the observed value. Denote the output state of this stage by $|\psi\rangle$. Then we have

$$|0\rangle \otimes |H\rangle$$

$$\downarrow \mathcal{Q}_r^{-1} \otimes I$$

$$\frac{1}{\sqrt{r}} \sum_{a \in \mathbb{Z}_r} |a\rangle \otimes |H\rangle$$

$$\downarrow V_G^g$$

$$\frac{1}{\sqrt{r}} \sum_{a \in \mathbb{Z}_r} |a\rangle \otimes |g^a H\rangle$$

$$\downarrow \mathcal{Q}_r^{-1} \otimes I$$

$$\frac{1}{r} \sum_{a \in \mathbb{Z}_r} \sum_{b \in \mathbb{Z}_r} e^{2\pi i a b/r} |b\rangle \otimes |g^a H\rangle$$

$$\downarrow \text{measure } A$$

$$\frac{1}{\sqrt{r}} \sum_{a \in \mathbb{Z}_r} e^{2\pi i a b/r} |g^a H\rangle = |\psi\rangle$$

The state $|\psi\rangle$ is almost what we want. It is precisely what we want if we struck lucky and observed $|0\rangle$ in the A register. But otherwise it contains coefficients $e^{2\pi i a b/r}$ which we would like to be able to replace by 1's.

What follows forms the crucial part of Watrous' argument. He shows that if we have superpositions $|\psi\rangle$ and $|\psi'\rangle$ in two distinct R registers, then we can operate on the pair of registers in such a way that afterwards $|\psi\rangle$ is set to exactly the superposition we want, and $|\psi'\rangle$ is unchanged (and hence it can be used again to 'correct' a different $|\psi\rangle$, in a different R register). In fact we apply the operator U_G to the pair of registers c times in succession for a carefully chosen value of c. The notation U_G^c is used to denote the composition of U_G, with itself, c times.

We suppose that

$$|\psi\rangle = \frac{1}{\sqrt{r}} \sum_{a \in \mathbb{Z}_r} e^{2\pi i ab/r} |g^a H\rangle, \quad |\psi'\rangle = \frac{1}{\sqrt{r}} \sum_{a' \in \mathbb{Z}_r} e^{2\pi i a'b'/r} \left|g^{a'} H\right\rangle.$$

In order to calculate the effect of U_G^c on $|\psi\rangle \otimes |\psi'\rangle$, it helps first to consider the effect on $|\psi'\rangle$ alone of the R register operator $M_{g^a h}$, which is defined by the rule

$$M_{g^a h} : |x\rangle \mapsto |g^a h x\rangle.$$

We see that

$$
\begin{aligned}
|\psi'\rangle = \frac{1}{\sqrt{r}} \sum_{a' \in \mathbb{Z}_r} e^{2\pi i a'b'/r} \left|g^{a'} H\right\rangle &= \frac{1}{\sqrt{r|H|}} \sum_{a' \in \mathbb{Z}_r} e^{2\pi i a'b'/r} \sum_{h' \in H} \left|g^{a'} h'\right\rangle \\
&\downarrow M_{g^a h} \\
&\frac{1}{\sqrt{r|H|}} \sum_{a' \in \mathbb{Z}_r} e^{2\pi i a'b'/r} \sum_{h' \in H} \left|g^a h g^{a'} h'\right\rangle \\
&= \frac{1}{\sqrt{r|H|}} \sum_{a' \in \mathbb{Z}_r} e^{2\pi i a'b'/r} \sum_{h' \in H} \left|g^{a+a'} h'' h'\right\rangle, \quad \text{for some } h'' \text{ (since } H \lhd \langle g \rangle H) \\
&= \frac{1}{\sqrt{r|H|}} \sum_{a' \in \mathbb{Z}_r} e^{2\pi i a'b'/r} \sum_{h' \in H} \left|g^{a+a'} h'\right\rangle \quad \text{(rewriting } h' \text{ for } h''h') \\
&= \frac{1}{\sqrt{r|H|}} e^{-2\pi i ab'/r} \sum_{a' \in \mathbb{Z}_r} e^{2\pi i (a'+a)b'/r} \sum_{h' \in H} \left|g^{a'+a} h'\right\rangle \\
&= \frac{1}{\sqrt{r}} e^{-2\pi i ab'/r} \sum_{a' \in \mathbb{Z}_r} e^{2\pi i (a'+a)b'/r} \left|g^{a'+a} H\right\rangle \\
&= \frac{1}{\sqrt{r}} e^{-2\pi ab'/r} \sum_{a'' \in \mathbb{Z}_r} e^{2\pi i a''b'/r} \left|g^{a''} H\right\rangle, \quad \text{for } a'' \in \mathbb{Z}_r,\ a'' \equiv a' + a \bmod r \\
&= e^{-2\pi i ab'/r} |\psi'\rangle.
\end{aligned}
$$

Note that in going from the third last to the second last line, the equation $g^{a'+a}H = g^{a''}H$ follows from the fact gH has order r in $\langle g \rangle H/H$. Now, since $U_G(|g^a h\rangle \otimes |h'\rangle) = |g^a h\rangle \otimes M_{g^a h} |h'\rangle$,

$$
\begin{aligned}
|\psi\rangle \otimes |\psi'\rangle &= \frac{1}{\sqrt{r}} \sum_a e^{2\pi i ab/r} |g^a H\rangle \otimes |\psi'\rangle \\
&\downarrow U_G \\
&\frac{1}{\sqrt{r}} \sum_a e^{2\pi i ab/r} \sum_h |g^a h\rangle \otimes M_{g^a h} |\psi'\rangle \\
&= \frac{1}{\sqrt{r}} \sum_a e^{2\pi i ab/r} \sum_h |g^a h\rangle \otimes e^{-2\pi i ab'/r} |\psi'\rangle \\
&= e^{-2\pi i ab'/r} |\psi\rangle \otimes |\psi'\rangle.
\end{aligned}
$$

Hence, provided that $cb' = b \bmod r$,

$$
\begin{aligned}
& |\psi\rangle \otimes |\psi'\rangle \\
& \downarrow U_G^c \\
= \quad & e^{-2\pi i a b' c/r} \, |\psi\rangle \otimes |\psi'\rangle \\
= \quad & e^{-2\pi i a b'/r} \, |\psi\rangle \otimes |\psi'\rangle \\
= \quad & \frac{1}{\sqrt{r}} \sum_{a \in \mathbb{Z}_r} |g^a H\rangle \otimes |\psi'\rangle \\
= \quad & |\langle g\rangle H\rangle \otimes |\psi'\rangle \, .
\end{aligned}
$$

The first of the two R-registers thus contains the state $|\langle g\rangle H\rangle$.

References

[1] S. Aaronson, *Quantum lower bound for the collision problem*, Proc. 34th Symposium on Theory of Computing, Montreal, Quebec, Canada, 2002. Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0111102`.

[2] L. Babai and S. Moran, *Arthur–Merlin games: a randomized proof system and a hierarchy of complexity classes*, J. Comput. System Sci. **36** 2 (1988), 254–276.

[3] L. Babai, G. Cooperman, L. Finkelstein, E. Luks and A. Seress, *Fast Monte Carlo algorithms for permutation groups*, J. Comput. System Sci. **50** (1995), 296–308.

[4] L. Babai and E. Szemeredi, *On the complexity of matrix group problems I*, in Proc. 25th Annual Symposium on Foundations of Computer Science, 1984, 229–240, Singer Island, Florida, 24-26 October 1984. IEEE.

[5] M. Boyer, G. Brassard, P. Høyer and A. Tapp, *Tight bounds on quantum search*, in Proceedings of the Workshop on Physics Of Computation: PhysComp '96 (Los Alamitos, CA, 1996), Institute of Electrical and Electronic Engineers Computer Society Press. Quantum Physics Archive `http://arxiv.org/abs/quant-ph/9605034`.

[6] G. Brassard, P. Høyer and A. Tapp, *Quantum counting*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/9805082`.

[7] G. Brassard, P. Høyer, M. Mosca and A. Tapp, *Quantum amplitude amplification*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0005055`.

[8] S. L. Braunstein, *Quantum computation*, Encyclopedia of Applied Physics, Update 1, Wiley–VCH, Berlin, 1999, . 239–256.

[9] S. L. Braunstein, G. M. D'Ariano, G. J. Milburn and M. F. Sacchi, *Universal teleportation with a twist*, Phys. Rev. Lett. **85** (2000), 2733–2736.

[10] K. Cheung and M. Mosca, *Decomposing finite Abelian groups*, J. Quantum Inf. Comp. **1** (3) (2001) 26–32.

[11] R. Cleve, *A note on computing Fourier transforms by quantum programs*, Unpublished, http://pages.cpsc.ucalgary.ca/ cleve/papers.html.

[12] R. Cleve, A. Ekert, C. Macchiavello and M. Mosca, *Quantum algorithms revisited*, Proc. Roy. Soc. London, Ser. A **454**, 339 (1998), 339–354.

[13] D. Coppersmith, *An approximate Fourier transform useful in quantum factoring*, IBM Research Report RC19642 (1994).

[14] D. Deutsch, *Quantum theory, the Church–Turing Principle and the universal quantum computer*, Proc. Roy. Soc. London, Ser. A **400** (1985), 97–117.

[15] D. Deutsch and R. Jozsa, *Rapid solution of problems by quantum computation*, Proc. Roy. Soc. London, Ser. A **439** (1992), 553–558.

[16] A. Ekert and R. Jozsa, *Quantum computation and Shor's factoring algorithm*, Rev. Mod. Phys. **68** (3) (1996), 733–753.

[17] M. Ettinger, P. Hoyer and E. Knill, *Hidden subgroup states are almost orthogonal*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/9901034`.

[18] E. Farhi, J. Goldstone, S. Gutmann and M. Sipser, *Quantum computation by adiabatic evolution*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0001106`

[19] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren and D. Preda, *A quantum adiabatic evolution algorithm applied to random instances of an NP–complete problem*, Science, **292**, April (2001) 472–476.

[20] R. Feynman, *Simulating physics with computers. Physics of computation, Part II (Dedham, Mass., 1981)*, Internat. J. Theoret. Phys., **21** (1981/82) 467–488.

[21] K. Friedl, G. Ivanyos, F. Magniez, M. Santha, and P. Sen, *Hidden translation and orbit coset in quantum computing*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0211091`.

[22] W. Fulton and F. Harris, *Representation theory: a first course*, (Graduate Texts in Mathematics **129**) Springer Verlag, 1991.

[23] L. K. Grover, *A fast quantum mechanical algorithm for database search*, Proc. 28th ACM Symp. Theory of Computing, 212–219, ACM, New York, 1996.
[24] L. K. Grover, *Quantum mechanics helps in searching for a needle in a haystack*, Phys. Rev. Lett., **79** (1997), 325–328.
[25] S. Hallgren, *Polynomial–time quantum algorithms for Pell's equation and the principal ideal problem*, In Proc. 34th Annual ACM Symposium on Theory of Computing, Montreal, Quebec, Canada, 2002.
[26] S. Hallgren, A. Russell and A. Ta–Shma, *Normal subgroup reconstruction and quantum computing using group representations*, Proc. 32nd Annual ACM Symposium on the Theory of Computing – STOC, ACM Press, New York (2000) 627–635.
[27] P. Halmos, *Finite dimensional vector spaces*, Springer Verlag; 2nd edition (December 1986).
[28] G. H. Hardy and E. M. Wright, *An introduction to the theory of numbers*, (5th Edition), Oxford University Press (1938) (1993 reprint).
[29] M. Hirvensalo, *Quantum computing*, Natural Computing Series, Springer–Verlag, (2001).
[30] G. Ivanyos, F. Magniez and M. Santha, *Efficient quantum algorithms for some instances of the non–Abelian hidden subgroup problem*, *In* Proc. 13th ACM SPAA (2001), 263–270.
[31] G. D. James and M. W. Liebeck, *Representations and characters of groups*, Cambridge University Press; (1993)
[32] R. Jozsa, *Characterizing classes of functions computable By quantum parallelism*, Proc. Roy. Soc. London, Ser. A (1991) **435**, 563–574.
[33] R. Jozsa, *Quantum factoring, discrete logarithms and the hidden subgroup problem*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0012084`.
[34] R. Jozsa, *Notes on Hallgren's efficient quantum algorithm for solving Pell's equation*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0302134`.
[35] R. Jozsa and N. Linden, *On the role of entanglement in quantum computational speed–up*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0201143`.
[36] E. Kashefi, A. Kent, V. Vedral and K. Banaszek, *A comparison of quantum oracles*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0109104`.
[37] R. V. Kadison and J. R. Ringrose, *fundamentals of operator algebras*, Volume I, Academic Press, New York (1983).
[38] A. Yu. Kitaev, *Quantum measurements and the Abelian stabiliser problem*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/9511026`
[39] A. Yu. Kitaev, *Quantum computations: algorithms and error correction*, Russian Math. Surveys, 52 6 (1997), 1191–1249.
[40] N. I. Koblitz, *A course in number theory and cryptography*, (Graduate Texts in Mathematics, **114**), Springer Verlag (1994).
[41] S. J. Lomonaco and L. H. Kauffman *continuous quantum hidden subgroup algorithms.* Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0304084`.
[42] A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, *Handbook of applied cryptography*, Series on Discrete Mathematics and its Applications, CRC Press (1997).
[43] C. Moore, D. Rockmoore, A. Russell and L. J. Schulman, *The hidden subgroup problem in affine groups: basis selection and Fourier sampling*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0211124`.
[44] M. Mosca and A. Ekert, *The hidden subgroup problem and eigenvalue estimation on a quantum computer*, Proc. 1st NASA International Conference on Quantum Computing and Quantum Communication, Palm Springs, USA, Lecture Notes in Computer Science 1509 (1999).
[45] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, Cambridge University Press, Cambridge, 2000.
[46] N. T. Ouellette, *Quantum computation*, notes available at `http://newton.swarthmore.edu/research/Nick.pdf`
[47] C. M. Papadimitrou, *Computational complexity*, Addison–Weseley, Reading, Massachussets (1994).
[48] A. O. Pittenger, *An introduction to quantum computing algorithms*, Progress in Computer Science and Applied Logic **19**, Birkhäuser (Boston), (2000).
[49] E. Rieffel and W. Polak, *An introduction to quantum computing for non–physicists*, ACM Comput. Surveys, **32** (3), (2000) 300–335.
[50] M. Püschel, M. Rötteler and T. Beth, *Fast quantum Fourier transforms for a alass of non–Abelian groups.* Quantum Physics Archive `http://arxiv.org/abs/quant-ph/9807064`.
[51] J. B. Rosser and L. Schoenfeld, *Approximate formulas for some functions of prime numbers*, Illinois J. of Math. 6:1 (1962), 64–94.
[52] M. Rötteler and T. Beth, *Polynomial–time solution to the hidden subgroup problem for a class of non–Abelian groups*, Quantum Physics Archive `http://arxiv.org/abs/quant-ph/9812070`.
[53] P. W. Shor, *Algorithms for quantum computation: discrete logs and factoring*, Proc. 35th Symposium on the Foundations of Computer Science (1994), 124–134.
[54] P. W. Shor, *Polynomial–time algorithms for prime factorisation and discrete logarithms on a quantum computer*, SIAM J. Comput. 26 **5** (1997), 1484–1509.
[55] D. Simon, *On the power of quantum computation*, Proc. 35th Annual Symposium on the Foundations of Computer Science, (IEEE Computer Society, Los Alamitos) (1994), 116–123.

[56] D. Simon, *On the power of quantum computation*, SIAM J. Comput. **26**(5) (1997), 1474–1483.
[57] W. van Dam, M. Mosca and U. Vazirani, *How powerful is adiabatic quantum computation?* Quantum Physics Archive `http://arxiv.org/abs/quant-ph/0206003`.
[58] J. Wallace, *A brief history of quantum computation*, `http://www.dcs.ex.ac.uk/~jwallace/history.htm`
[59] J. Watrous, *Succinct quantum proofs for properties of finite groups*, Proc. of the 41st Annual Symposium on Foundations of Computer Science (2000), 537–546.
[60] J. Watrous, *Quantum algorithms for solvable groups*, Proc. 33rd ACM Symposium on Theory of Computing (2001), 60–67.

Michael Batty†,
e–mail: Michael.Batty@ncl.ac.uk

Samuel L. Braunstein‡,
e–mail: schmuel@cs.york.ac.uk

Andrew J. Duncan†,
e–mail: A.Duncan@ncl.ac.uk

Sarah Rees†,
e–mail: Sarah.Rees@ncl.ac.uk

† Department of Mathematics,
School Of Mathematics and Statistics,
Merz Court,
University of Newcastle upon Tyne,
Newcastle upon Tyne,
NE1 7RU,
UK.

‡ Department of Computer Science,
University of York,
York,
YO10 5DD,
UK.