Quantum Computation

"Because nature isn't classical, dammit..."
Richard P. Feynman

André Berthiaume[1]

ABSTRACT Historically, Turing machines have been the paradigm by which we defined computability and efficiency. This is based on Church's thesis that everything effectively computable can also be computed on a Turing machine. But since our world behaves quantum mechanically, it seems reasonable to also consider computing models that make use of quantum mechanical properties. First stated by Benioff [Ben82] and Feynman [Fey82], this idea was formalized by Deutsch [Deu85] when he introduced his quantum computer and, later on, quantum gate arrays. This paper gives an introduction to quantum computing and briefly looks at a few results in quantum computation, not the least of which is Shor's polynomial time factoring algorithm ([Sho94] and [Sho95]).

1 The Need for Quantum Mechanics

Why introduce quantum mechanics in computation? The opening quote, if a little blunt, captures the essence of the answer. At the center of computer science are two questions: *what problems are computable* and *how efficiently can they be computed?* Historically, (probabilistic) Turing machines have been the paradigm by which we defined computability and efficiency. This is based on Church's thesis that everything effectively computable can also be computed on a Turing machine. But since our world behaves quantum mechanically, it seems reasonable to also consider computing models that make use of quantum mechanical properties. First stated by Benioff [Ben82] and Feynman [Fey82], this idea was formalized by Deutsch [Deu85] when he defined the first quantum computing model that made full use ofn quantum superposition. This paper gives an introduction to Deutsch's quantum computing model and briefly looks at a few results in quantum computation, not the least of which is Shor's polynomial time factoring algorithm.

Before defining a quantum computing model, some basic notions of quantum mechanics must be introduced. A comprehensive presentation of quantum mechanics is beyond the scope of this paper, but fortunately only the

[1]Centrum voor Wiskunde en Informatica, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands (berthiau@cwi.nl).

very simplest systems are used for quantum computation: two-state systems or finite groups of two-state systems. The next section introduces the relevant notions for these special cases. Quantum gate arrays will be defined next and, after a few examples of their capabilities, the results leading to Shor's algorithm will be briefly reviewed. The last section addresses some of the difficulties facing the actual construction of a quantum computer.

Because of space restriction, it will not be possible to include the historical context which lead to many of the subjects discussed here. Where possible, we will give references for more detailed accounts. For a more exhaustive review of the history of quantum computation (dating back almost 50 years), the reader should consult [Ben96].

2 Basic Principles of Quantum Mechanics

We now introduce the basic rules of quantum mechanics through a series of principles. Young's celebrated two-slit experiment will serve as a background to illustrate these principles.

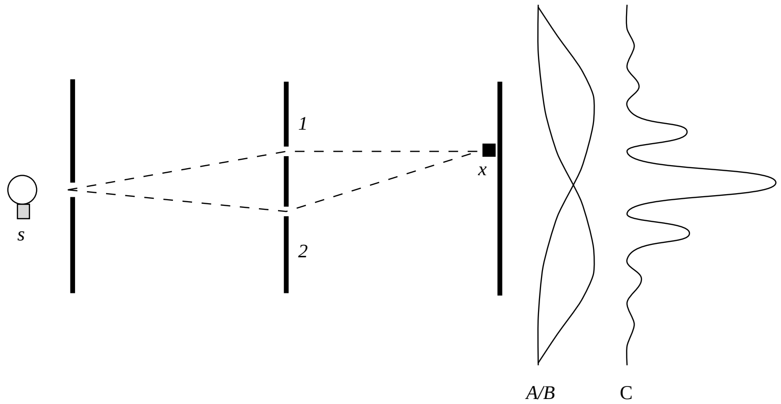


FIGURE 1. Young's two-slits experiment. Curves A and B show the light intensity when only one hole is open. Curve C shows the interference pattern when both holes are open. (the curves are exaggerated)

In Young's experiment (figure 1), light coming out of a hole in the left wall must go through two small holes in the center wall. A detector on the right wall measures the light intensity at different positions along the length of the wall. If only one hole is open, the intensity reaches its maximum at a position directly in line with that hole and the source s. As the detector moves away from that position, the intensity slowly fades and eventually vanishes. When *both* holes are open, the intensity pattern is *not* the sum of the two one-hole intensities, as one would expect, but an alternation of bright and dark fringes. This effect is caused by the *interference* of the light coming out from both holes. Surprisingly, the interference persists even when the source s is dim enough to send only one photon at a time; if many runs are made and a photon count is kept for various positions, the same pattern of bright and dark fringes appears. Each photon seems to interfere with itself.

2.1 Probability Amplitudes

The self-interference appearing in Young's experiment is just one example illustrating that classical intuition cannot be applied to quantum systems. The purpose of this section is not to explain *why* such strange behavior appears at the quantum level but merely to state the rules for these behaviors. Following Feynman's example [FLS64], these rules are presented as the *principles* of quantum mechanics.

Definition 2.1 *For a given experiment, an* **event** *is a set of initial and final conditions.*

For example, an event in Young's experiment is "*a photon leaves the source s and arrives in the detector at position x*". The goal of quantum mechanics is to predict whether an event can happen or not. The first principle of quantum mechanics defines the probability of an event actually happening.

First Principle: The probability p of an event is given by the square norm of a complex number α (called a *probability amplitude* or simply *amplitude*).

$$p = \|\alpha\|^2$$

The probability amplitude of an event will be noted as follow:

$$\langle\ \textit{final condition}\ |\ \textit{initial condition}\ \rangle$$

For example, the above event can be written as

$$\langle \text{ a photon is detected at position } x \mid \text{a photon leaves } s\ \rangle$$

or more succinctly $\langle x|s\rangle$. The bracket notation, due to Dirac, is reminiscent of conditional probabilities and can be read as: $\langle x|s\rangle$ is the amplitude of detecting a photon at position x given that a photon left the source s. In fact, amplitudes can be treated in the same way as probabilities. Consider again Young's experiment. If a photon leaves the source and arrives at the detector, it must do so by going through the holes in the wall. That is to say, the event $\langle x|s\rangle$ can be broken down in two sequential events: the photon first leaves the source s and arrives to the middle wall, then the photon comes out of this middle wall and arrives at the detector. But to go through the middle wall, the photon has two option: either through hole 1 or hole 2. The following two principles indicate how the laws for addition and multiplication of probabilities also apply to amplitudes.

Second Principle: If an event can be divided in two sequential sub-events, the amplitude of the event is the product of the amplitudes for each of the sub-events.

Third Principle: If an event can occur in several alternative ways, then the amplitude of the event is the sum of the amplitudes for each alternative taken separately.

From these two principles:

$$\begin{aligned} \langle x|s\rangle &= \langle x|\text{wall}\rangle\langle \text{wall}|s\rangle && (1.1)\\ &= \langle x|1\rangle\langle 1|s\rangle + \langle x|2\rangle\langle 2|s\rangle && (1.2)\end{aligned}$$

where $\langle x|i\rangle$ is the amplitude of a photon arriving at x given it came out of hole i and $\langle i|s\rangle$ is the amplitude of a photon entering hole i given that it left the source s. Equation 1.2 implicitly considers terms of the form $\langle x|1\rangle\langle 2|s\rangle$ or $\langle x|2\rangle\langle 1|s\rangle$ to be equal to zero. Informally speaking, these would be asking: "what is the amplitude that a photon leaves s, goes through hole 1, comes out of hole 2 and then arrives at x?" and similarly for the other case. The answer must include the amplitude of going from hole 1 to hole 2 ($\langle 2|1\rangle$) and vice versa ($\langle 1|2\rangle$). So the correct bracket form for the above questions should be: $\langle x|1\rangle\langle 1|2\rangle\langle 2|s\rangle$ and $\langle x|2\rangle\langle 2|1\rangle\langle 1|s\rangle$. One can verify experimentally that whenever a photon is detected entering one hole, it is never detected coming out of the opposite hole. Equations 1.3 and 1.4 express this situation in terms of amplitudes of events.

$$\begin{aligned} \langle 1|1\rangle = \langle 2|2\rangle &= 1 && (1.3)\\ \langle 1|2\rangle = \langle 2|1\rangle &= 0 && (1.4)\end{aligned}$$

No matter where the detector is positioned, the amplitude $\langle x|s\rangle$ is completely determined by the amplitudes to transit to and from the two holes. For this reason, the holes are natural elements for expressing events. This leads to the following definition:

Definition 2.2 *The set $B = \{i \mid i \text{ is the label of some condition}\}$ is a set of* **basis states** *if for all $i, j \in B$*

$$\langle i|j\rangle = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

and for any initial condition Y and final condition X, we have

$$\langle X|Y\rangle = \sum_{i\in B} \langle X|i\rangle\langle i|Y\rangle$$

Fourth Principle: Any event can be described in terms of a set of basis states by giving the transition amplitudes to and from those basis states.

Young's experiment seems to have only one possible set of basis states (namely, $B = \{1, 2\}$) but actually, for any experiment, there is an infinite number of sets of basis state. Some appear naturally, such as the two holes in this experiment, and others less so. The notion of basis states can be better understood through an analogy with a three dimensional real vector space V and the dot product in that space. Let the three vectors

$$\vec{e_1} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \vec{e_2} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \text{ and } \vec{e_3} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

be the canonical basis of V and let

$$\vec{A} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \text{ and } \vec{B} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

be two vectors in V. The following is a non-standard yet valid expression for the dot product of $\vec{A}$ and $\vec{B}$:

$$\begin{aligned} \vec{A} \cdot \vec{B} &= a_1 b_1 + a_2 b_2 + a_3 b_3 \\ &= (\vec{A} \cdot \vec{e_1})(\vec{e_1} \cdot \vec{B}) + (\vec{A} \cdot \vec{e_2})(\vec{e_2} \cdot \vec{B}) + (\vec{A} \cdot \vec{e_3})(\vec{e_3} \cdot \vec{B}) \\ &= \sum_{i=1}^{3} (\vec{A} \cdot \vec{e_i})(\vec{e_i} \cdot \vec{B}) \end{aligned}$$

This last line is similar to the equation for $\langle X|Y\rangle$ in definition 2.2. The vectors $\vec{A}$ and $\vec{B}$ correspond to the two conditions X and Y and the set of canonical basis vectors corresponds to the set of basis states. In this sense, $\langle X|Y\rangle \equiv \langle X| \cdot |Y\rangle$. The analogy can be pushed further still. Consider the following:

$$\begin{aligned} \vec{B} &= b_1 \vec{e_1} + b_2 \vec{e_2} + b_3 \vec{e_3} \\ &= \sum_{i=1}^{3} \vec{e_i} b_i \\ &= \sum_{i=1}^{3} \vec{e_i}(\vec{e_i} \cdot \vec{B}) \end{aligned}$$

This simply is the previous equation for $\vec{A} \cdot \vec{B}$ where we abstracted $\vec{A}$. In other words, $\vec{B}$ is the vector sum of its components along each of the basis vectors $\vec{e_1}$, $\vec{e_2}$ and $\vec{e_3}$. Similarly, the initial states Y of $\langle X|Y\rangle$ can be expressed in terms of the set of basis states by abstracting X.

$$|Y\rangle = \sum_{i\in B} |i\rangle\langle i|Y\rangle \tag{1.5}$$

(Note: $\langle\cdot|\cdot\rangle$ is just a scalar number, so $|i\rangle\langle i|Y\rangle = \langle i|Y\rangle|i\rangle$ but the second form is preferred.) The left and right halves of $\langle\cdot|\cdot\rangle$ are named *bra* and *ket* respectively. Equation 1.5 defines the state of an initial condition Y as function of B, the set of basis states. The ket $|Y\rangle$ is called a *state vector* and it lies in a complex vector space (Hilbert space) yspanned by the basis vector associated with the basis states in B. Similarly,

$$\langle X| = \sum_{j\in B} \langle X|j\rangle\langle j|$$

defines the state of a final condition X as function of B. The $\langle X|$ and $|Y\rangle$ could also be written using another set of basis state just as $\vec{B}$ can be written in a basis other than the canonical one. Section 2.2 and 2.4 will present examples where multiple sets of basis states naturally appear. For a basis set B, the state $|Y\rangle$ is said to be in *quantum superposition* of the basis states in B, if more than one $\langle i|Y\rangle$ is non-zero.

By the first principle, the square norm of an amplitude gives a probability of the corresponding event. By the fourth principle, any event can be expressed as a function of the amplitude corresponding to each basis states. Since probabilities must add up to 1, there should be some constraint on the amplitudes:

Fifth Principle: For any set of basis states B and for any initial condition Y:

$$\sum_{i\in B} \|\langle i|Y\rangle\|^2 = 1$$

This follows from the definition of a set of basis states which has to be complete (all possibilities are accounted for) and orthogonal ($\langle i|j\rangle = 0$ if $i \neq j$, 1 otherwise).

We leave as an exercise the proof of the following theorem (recall that α^* is the complex conjugate of α.)

Theorem 2.1 *For any condition A and B, $\langle B|A\rangle = \langle A|B\rangle^*$.*

From which we derive that if $|\varphi\rangle = \sum_{i\in B} \alpha_i|i\rangle$ then

$$\langle\varphi| = \sum_{i\in B} \alpha_i\langle i|$$

All these principles can be put together in the following situation: consider a system whose initial condition is expressed by the state vector

$$|Y\rangle = \sum_{i\in B} \beta_i|i\rangle$$

where $\beta_i = \langle i|Y\rangle$, the amplitude of the system under consideration in each of the basis states i. Now consider a general final condition

$$\langle X| = \sum_{j \in B} \alpha_j \langle j|$$

where $\alpha_i = \langle X|i\rangle$. What is the amplitude of finding the system in condition X given that it was initially prepared in Y?

$$\langle X|Y\rangle = \left(\sum_{j \in B} \alpha_j \langle j|\right) \left(\sum_{i \in B} \beta_i |i\rangle\right) = \sum_{i \in B} \alpha_i \beta_i \qquad \text{by def. 2.2}$$

2.2 Qubits and How to Observe Them

The five principles presented in section 2 are completely general and apply to any quantum system. However, for the sake of clarity, we will limit our attention to systems of interest for quantum computation, namely *qubits* and *quantum registers.* We begin by defining a qubit, the quantum version of bit, as defined by Schumacher [Sch95].

Definition 2.3 *A* **qubit** *is a quantum state* $|\varphi\rangle$ *of the form*

$$|\varphi\rangle = \alpha|\mathbf{0}\rangle + \beta|\mathbf{1}\rangle$$

where $\alpha, \beta \in \mathrm{C}$ *and* $\|\alpha\|^2 + \|\beta\|^2 = 1$

The above definition leaves the actual medium of a qubit completely undefined. It is of no importance whether the qubit is encoded in the polarization of a photon, the spin of an atom, the up/down orientation of lamp-post or the alive/dead state of Schrödinger's poor cat[2] as long as the object is treated according to the principles given in this introduction. There is nothing *in principle* that forbids one from getting quantum mechanical effects with lamp-posts or cats. In practice, however, it might be easier to use photons or atoms. The word "easier" should be taken loosely; in the last section, we will discuss briefly the issues regarding actual implementations of qubits and, more generally, quantum computers.

The main difference between qubits and classical bits is that a bit can only be set to either $\mathbf{0}$ or $\mathbf{1}$while a qubit $|\varphi\rangle$ can take on any (uncountable) quantum superposition of $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$. This means an infinite amount of information could potentially be encoded in a single qubit by appropriately defining the amplitudes α and β. Unfortunately, what goes in does not necessarily come out. Quantum mechanics imposes very strict rules as to

[2]My apologies to Simone and Hugin and all other cats who might read this paper.

how to extract information out of quantum state. This is done through a mathematical construct called an *observable*. Let $|\varphi\rangle$ be the state of a quantum system. We have a probe P at our disposal to measure some property of $|\varphi\rangle$. This property could be the direction, the position or even a simple yes/no question. We need to model the action of the probe P on the state $|\varphi\rangle$.

Definition 2.4 *Let H be the Hilbert space used to represent the state vectors of a quantum system. An* **observable** *$\mathcal{O}$ is a set of subspaces $E_1, E_2, \ldots, E_k \subseteq H$ such that these subspaces completely partition H. i.e.*

$$E_1 \times E_2 \times \cdots \times E_k = H$$

and

$$\forall\, i,j \in \{1,\ldots,k\}, i \neq j : \quad E_i \perp E_j$$

An observable is the mathematical representation of the probes P. The next principle defines the effect of an observation of a state vector.

Sixth Principle: Let $|\varphi\rangle$ be a state vector in a state space H and let $\mathcal{O} = \{E_1, E_2, \ldots, E_k\}$ be an observable. Since $\mathcal{O}$ partitions H, $|\varphi\rangle$ can be expressed as a linear superposition of its components along each of the E_i's:

$$|\varphi\rangle = \sum_{i=1}^{k} \alpha_i |\varphi_{E_i}\rangle$$

where $|\varphi_{E_i}\rangle$ lies in E_i. Observing the state $|\varphi\rangle$ with $\mathcal{O}$ will cause the following:

1. One of the E_i will be selected with probability $\|\alpha_i\|^2$.
2. The state $|\varphi\rangle$ will "collapse" to $|\varphi_{E_i}\rangle$ (renormalized).
3. The only classical information given by $\mathcal{O}$ is which subspace (i) was selected. All information not in $|\varphi_{E_i}\rangle$ is lost.

To each possible output value of the probe corresponds a subspace in the observable. Since all these values are different from each other, the corresponding subspace must be orthogonal. Again, any observable is allowed in principle for observing a quantum state. Whether the physical apparatus that corresponds to that specific observable is easy to build is a different matter entirely.

The standard observable for a qubit is $\mathcal{B} = \{E_{\mathbf{0}}, E_{\mathbf{1}}\}$ where $E_{\mathbf{0}}$ and $E_{\mathbf{1}}$ are spanned by the two basis vectors $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$ respectively. An example of a non-standard observable on a qubit is $\mathcal{O} = \{E_{\mathbf{0}'}, E_{\mathbf{1}'}\}$ where $E_{\mathbf{0}'}$ and $E_{\mathbf{1}'}$ are spanned by

$$|\mathbf{0}'\rangle = \frac{1}{\sqrt{2}}(|\mathbf{0}\rangle + |\mathbf{1}\rangle) \quad \text{and} \quad |\mathbf{1}'\rangle = \frac{1}{\sqrt{2}}(|\mathbf{0}\rangle - |\mathbf{1}\rangle)$$

respectively. The reader can check that $E_{\mathbf{0}'}$ and $E_{\mathbf{1}'}$ have the correct properties of an observable. The next section will emphasize how the information in a qubit is linked to which observable is used to read them.

2.3 Digression in Quantum Cryptography

If qubits are encoded in the polarization of photons, the two observables $\mathcal{B}$ and $\mathcal{O}$ in section 2.2 have a simple physical implementations. Define $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$ as horizontally and vertically polarized photons. Then $\mathcal{B}$ is a horizontally positioned polarizing filter and $\mathcal{O}$ is a polarizing filter set at 45 degrees from the horizontal. This forms the basic set-up for quantum cryptography. Alice, the sender, wants to send her secret bit to Bob, the receiver. They agree that $\mathbf{0}$ will be encoded as a photon either in state $|\mathbf{0}\rangle$ or $|\mathbf{0}'\rangle$ (based on a coin flip) and a $\mathbf{1}$ is similarly send as either $|\mathbf{1}\rangle$ or $|\mathbf{1}'\rangle$. Bob must read the the photons either with $\mathcal{B}$ or $\mathcal{O}$. Table 1.1 shows the various outcomes for each possible choice of encoding (by Alice) and observable (by Bob). As an example, assume Alice's bit is a zero. Her coin-flip instructs

Alice sends	Bob uses	Alice's state relative to Bob	result and probability	Correctness
$\|\mathbf{0}\rangle$	$\mathcal{B}$	$\|\mathbf{0}\rangle$	$\mathbf{0}$ (prob 1)	correct
	$\mathcal{O}$	$\frac{1}{\sqrt{2}}(\|\mathbf{0}'\rangle + \|\mathbf{1}'\rangle)$	$\mathbf{0}'/\mathbf{1}'$ (prob 50%)	random
$\|\mathbf{0}'\rangle$	$\mathcal{B}$	$\frac{1}{\sqrt{2}}(\|\mathbf{0}\rangle + \|\mathbf{1}\rangle)$	$\mathbf{0}/\mathbf{1}$ (prob 50%)	random
	$\mathcal{O}$	$\|\mathbf{0}'\rangle$	$\mathbf{0}'$ (prob 1)	correct
$\|\mathbf{1}\rangle$	$\mathcal{B}$	$\|\mathbf{1}\rangle$	$\mathbf{1}$ (prob 1)	correct
	$\mathcal{O}$	$\frac{1}{\sqrt{2}}(\|\mathbf{0}'\rangle - \|\mathbf{1}'\rangle)$	$\mathbf{0}'/\mathbf{1}'$ (prob 50%)	random
$\|\mathbf{1}'\rangle$	$\mathcal{B}$	$\frac{1}{\sqrt{2}}(\|\mathbf{0}\rangle - \|\mathbf{1}\rangle)$	$\mathbf{0}/\mathbf{1}$ (prob 50%)	random
	$\mathcal{O}$	$\|\mathbf{1}'\rangle$	$\mathbf{1}'$ (prob 1)	correct

TABLE 1.1. Depending on how Alice will encode her secret bit and what observable will be used by Bob, Bob's read-out of the photon send by Alice will either be correct or completely randomized.

to encode it as a $|\mathbf{0}\rangle$. If Bob chooses the observable $\mathcal{B}$ to observe Alice's incoming photon, he will get a $\mathbf{0}$ outcome with certainty and will know Alice's bit (assume Alice discloses her bases to Bob in a later discussion). However, Bob could choose the $\mathcal{O}$ observable to read the incoming photon.

The reader can check that

$$|\mathbf{0}\rangle = \frac{1}{\sqrt{2}}(|\mathbf{0}'\rangle + |\mathbf{1}'\rangle)$$

Bob thus has 50% probability of getting the $\mathbf{0}'$ outcome and 50% probability of getting the $\mathbf{1}'$ outcome. Similar arguments hold for each case of table 1.1. To learn more about the importance of this situation, see [BBB$^+$92], [BBE92] and [Bra93]. The point of this digression was to demonstrate that the information of a quantum state is a function of the observable used. The same state $|\varphi\rangle$ observed with two different observable can give a definite answer in one case and a totally randomized answer in the other.

2.4 Evolution of a Quantum System

The situations considered up to now were static in the sense that the initial state did not change after being set. Once the initial state vector $|Y\rangle$ was defined, we considered amplitudes of the form $\langle X|Y\rangle$ for some $\langle X|$. But to compute something with quantum states, some transformation of the initial state will have to be performed. Suppose an apparatus A is used to execute this transformation on the initial state $|Y\rangle$. The events of interest are now of the type "what is the amplitude for the final condition X given that the initial condition Y went through apparatus A?". In the bracket notation, this is written as

$$\langle X|A|Y\rangle$$

The next principle gives the mathematical representation of A.

Seventh Principle: State vectors are transformed by unitary matrices. Relative to a set of basis state $\mathcal{B}$, $A_{i,j}$ $(i, j \in \mathcal{B})$ is the amplitude of going from state i to state j.

A matrix U is unitary if $UU^\dagger = U^\dagger U = I$ ($U^\dagger$ is the conjugate transpose of U). In principle, any unitary transformation on a quantum state is allowed but constructing a physical device corresponding to any given matrix U might pose some technological problems.

A simple example of qubit transformation can be made with polarized photons. A polarized photon going through a transparent tank of sugar water will have its polarization slowly rotated[3]. The amount of rotation depends on the length of the tank and the density of sugar. By appropriately setting these parameters, the tank can be made to induce a 45 degree rotation on incoming photons. If $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$ respectively correspond to horizontally and vertically polarized photon, then the tank has the following

[3]Actually, more than just the polarization will be affected, but for simplicity, we will ignore these other effects.

effect:

$$|\mathbf{0}\rangle \quad \text{will be transformed into} \quad \frac{1}{\sqrt{2}}(|\mathbf{0}\rangle + |\mathbf{1}\rangle) \quad \text{and} \tag{1.6}$$

$$|\mathbf{1}\rangle \quad \text{will be transformed into} \quad \frac{1}{\sqrt{2}}(-|\mathbf{0}\rangle + |\mathbf{1}\rangle) \tag{1.7}$$

The transformation induced on the basis states completely determines the matrix A. If $|\varphi\rangle = \alpha|\mathbf{0}\rangle + \beta|\mathbf{1}\rangle$ is shot through the tank, it will come out transformed into state $|\varphi'\rangle$ where

$$\begin{aligned}
|\varphi'\rangle &= A|\varphi\rangle \\
&= A(\alpha|\mathbf{0}\rangle + \beta|\mathbf{1}\rangle) \\
&= \alpha A|\mathbf{0}\rangle + \beta A|\mathbf{1}\rangle && \text{(by linearity)} \\
&= \alpha\left(\frac{1}{\sqrt{2}}(|\mathbf{0}\rangle + |\mathbf{1}\rangle)\right) \\
&\quad + \beta\left(\frac{1}{\sqrt{2}}(-|\mathbf{0}\rangle + |\mathbf{1}\rangle)\right) && \text{(by eqn. 1.6, 1.7)} \\
&= \frac{1}{\sqrt{2}}(\alpha - \beta)|\mathbf{0}\rangle + \frac{1}{\sqrt{2}}(\alpha + \beta)|\mathbf{1}\rangle
\end{aligned}$$

Or, in more familiar matrix-and-vector style:

$$A = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}, \qquad |\varphi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

and

$$|\varphi'\rangle = A|\varphi\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}}(\alpha - \beta) \\ \frac{1}{\sqrt{2}}(\alpha + \beta) \end{pmatrix}$$

Another transformation very similar to that induced by A is "*square root of not*". The name is derived form the fact a qubit going through two identical $\sqrt{\text{Not}}$-apparatuses comes out in a state corresponding to the boolean inverse of its initial value. The $\sqrt{\text{Not}}$ transformation is given below; the reader is encouraged to check that it performs as stated.

$$\sqrt{\text{Not}} = \frac{1}{4}\begin{pmatrix} 1 - i & 1 + i \\ 1 + i & 1 - i \end{pmatrix}$$

A qubit can be set, transformed and observed. But to do serious computation, more than a single qubit is required. The next section introduces the last few mathematical tools needed for quantum computation.

2.5 Quantum Registers

Quantum computations generally use more than just one qubit. The mathematical formalism introduced so far must be adapted to the treatment of groups of qubits.

Definition 2.5 *A* **quantum register** *is an ordered set of a finite number of qubits.*

Definition 2.6 *The* **standard basis** $\mathcal{B}$ *of an* n*-qubit quantum register is*

$$\mathcal{B} = \{|i\rangle \mid i \textit{ is an } n\textit{-bit binary string}\}$$

Let $|\varphi_1\rangle = \alpha_0|\mathbf{0}\rangle + \alpha_1|\mathbf{1}\rangle$ and $|\varphi_2\rangle = \beta_0|\mathbf{0}\rangle + \beta_1|\mathbf{1}\rangle$ be two qubits composing a 2-qubit quantum register. The state vector $|\psi\rangle$ of the register is defined as the *tensor product* of the states $|\varphi_1\rangle$ and $|\varphi_2\rangle$

$$\begin{aligned} |\psi\rangle = |\varphi_1\rangle \otimes |\varphi_2\rangle &= \left(\sum_{i=0}^{1} \alpha_i|i\rangle\right) \otimes \left(\sum_{j=0}^{1} \beta_j|j\rangle\right) \\ &= \sum_{i,j=0}^{1} \alpha_i\beta_j(|i\rangle \otimes |j\rangle) \end{aligned}$$

By definition, the tensor product maps $|i\rangle \otimes |j\rangle$ (where i and j are basis states to $|ij\rangle$. This allows us to write $|\psi\rangle$ as

$$|\psi\rangle = \sum_{i,j=0}^{\alpha_i\beta_j|ij\rangle}$$

Similarly, let A and B be two unitary matrices corresponding to two apparatuses operating on $|\varphi_1\rangle$ and $|\varphi_2\rangle$ separately. The combined action of A and B on the joint state $|\psi\rangle = |\varphi_1\rangle \otimes |\varphi_2\rangle$ is defined as a 4×4 matrix C where

$$C = A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B \\ a_{21}B & a_{22}B \end{pmatrix}$$

It is easily verified that the tensor product has the following property

$$(A \otimes B)(|\varphi_1\rangle \otimes |\varphi_2\rangle) = (A|\varphi_1\rangle) \otimes (B|\varphi_2\rangle)$$

and that it preserves unitarity.

So far, we seem to only complicate the notation for a basically simple situation: two independent qubits are acted upon by two independent apparatuses. But the point in joining two qubits is specifically to allow them to be *dependent*. In fact, not all states of a 2-qubit quantum register can be expressed as the tensor product of single qubit states. An example of such state is

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|\mathbf{00}\rangle + |\mathbf{11}\rangle)$$

If $|\psi\rangle$ is observed with the observable corresponding to the standard basis, the results "**00**" or "**11**" will be seen each with probability 50%, but the results "**01**" or "**10**" will never be observed. When the state of an n-qubit register cannot be expressed as the tensor product of n qubit states, the register is said to be *entangled.* Similarly, not all 4×4 unitary matrices can be expressed as the tensor product of two 2×2 unitary matrices. One such matrix is

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

which effects the following mapping of the basis states of the register: if the register's state is such that the first qubit is $\mathbf{o}$ no action is performed, otherwise the value of the second qubit is negated. The above matrix performs the operation called *controlled-not* on two qubits. As such, it is the first example of a quantum computation introduced here. For the importance of the controlled-not operation, see [BBC+95].

It is a simple matter to generalize what has been presented in this section to represent the state of an n-qubit register. The general state vector $|\psi\rangle$ of an n-qubit quantum register is

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$$

and the 2^n vectors $|i\rangle$ form the set of basis states of the register (note that within $|\cdot\rangle$, the i stands for the binary expansion of the value i). This means that $|\psi\rangle$ is a vector in a 2^n dimensional Hilbert space and operations are defined by $2^n \times 2^n$ unitary matrices. Observables for extracting information from the state vectors are partitions of the 2^n dimensional Hilbert space.

We are now ready to apply these notions of quantum mechanics to computation.

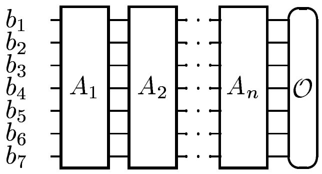
3 Computing with Quantum Registers

The original quantum computing model proposed by Deutsch was essentially a Turing machine, but with the added properties that tape cells and the head's state could be in quantum superposition. Deutsch also constrained the transition function by requiring it to induce a unitary evolution of the Turing machine. This was of course necessary to respect quantum mechanical principles, but it made programming quantum Turing machines even more nightmarish than programming classical ones. Verifying that a given transition function corresponds to a unitary evolution is non-trivial. Bernstein and Vazirani give three rules for verifying that a transition function performs its computation in a unitary fashion [BV93], but even using this method still requires unnatural programming skills.

In classical complexity theory, uniform circuit families are also commonly used as a computing model. Turing machine and uniform circuit families are effectively equivalent in computing power in that they can simulate one another with negligible complexity overhead. This makes the use of one or the other a matter of taste. There exists a quantum equivalent to uniform circuit families: quantum gate arrays. They were introduced by Deutsch ([Deu89]) and studied extensively by many authors (see [BBC+95] for a detailed review of quantum gate arrays). Yao [Yao93] has shown that acyclic quantum gate arrays could simulate quantum Turing machines, thus making the use of one or the other a matter of choice. However, since quantum gate arrays allow a more natural way to introduce unitarity in computation, they are emerging as the standard quantum computing model. In what follows, the diagrams and gate array notation are as in [BBC+95].

3.1 Quantum Gate Arrays

The diagram below represents a general quantum gate array. The initial (basis) state of the register is on the left and time flows from left to right. One might think of the particles composing the register as travelling through the different gates. At the right end is the observable that extracts information from the register after it has gone through all the gates.



The sequence of A_i's with observable $\mathcal{O}$ is what constitutes a quantum program. Formally speaking, the A_i gate should be of some well-defined form corresponding to some definition of elementary steps. For the purpose of this paper, it is sufficient to consider any quantum gate acting on only one or two qubits to be such an elementary step. The reader is encouraged to consult [BBC+95] for more details on the notion of elementary quantum gates. Also, in our gate arrays, we will not always specify *all* the elementary gates: in some cases we will simply convince ourselves that the necessary elementary gates *could be* written down. This procedure is analogous to writing pseudo-code for classical Turing machine and will provide a better intuitive approach.

To illustrate the programming of quantum gate arrays, we will use a variation of the Deutsch-Jozsa Problem [DJ92]. First, we define two properties of functions from $\{\mathbf{0},\mathbf{1}\}^n$ to $\{\mathbf{0},\mathbf{1}\}$:

Definition 3.1 *A function $f : \{\mathbf{0},\mathbf{1}\}^n \to \{\mathbf{0},\mathbf{1}\}$ is* **non-balanced** *if one of the two values of f has majority.*

Definition 3.2 *A function $f : \{0, 1\}^n \to \{0, 1\}$ is* **non-constant** *if there exist $x, y \in \{0, 1\}^n$ such that $f(x) \neq f(y)$.*

Notice that most (but not all) functions from $\{0, 1\}^n$ to $\{0, 1\}$ have both properties simultaneously. The modified Deutsch-Jozsa problem is described as follow:

Modified Deutsch-Jozsa Problem (MDJP):

Input: a computable function $f : \{0, 1\}^n \to \{0, 1\}$

Problem: to answer either "non-balanced" or "non-constant", but the answer must apply to f.

The original Deutsch-Jozsa problem dealt with strings rather than functions and was the first example of a problem which could be solved exponentially faster on a quantum computer than on a Turing machine [DJ92]. By recasting the original problem in the context of promise-problems, Berthiaume and Brassard in [BB94], [BB92a] and [BB92b] proved some early results in relativized quantum complexity theory. These results were improved upon first by Bernstein and Vazirani [BV93] and then by Simon [Sim94] who proved the following theorem.

Theorem 3.1 (Simon) *There exists an oracle relative to which there is a problem solvable in polynomial time (with bounded error probability) on a quantum computer, but any probabilistic Turing machine with bouded error probability claiming to solve this problem (using the oracle) will require exponential time on infinitely many inputs.*

Simon's theorem is the strongest argument in favor of the superiority of quantum computers over Turing machines. Moreover, the quantum gate array used in Simon's proof is similar to the one used by Shor for his factoring algorithm. In this section, we present a solution to the MDJP using quantum gate arrays. This allows us to introduce, in section 4, the gate array used in the proof of theorem 3.1. In section 5, we outline the quantum component of Shor's factoring algorithm.

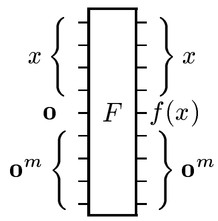
We now present a quantum solution to the MDJP. According to the principles given in section 2, a valid quantum algorithm corresponds to a unitary matrix. But programming in terms of unitary matrices is unnatural to humans who prefer to think in terms of sequential steps. We need to break down the MDJP into a sequence of unitary operations. If each of these sequential steps are simple enough, asserting their unitarity should be a relatively easy task. Just how simple need be these steps? Ideally, they should be broken down to what we defined as elementary gates, but in some cases, this will be unnecessary. The following theorem (Lecerf [Lec63] and Bennett [Ben73]) greatly simplifies quantum thinking:

Theorem 3.2 (Lecerf-Bennett) *For any Turing machine T computing a function f there exists a* reversible *Turing machine T' computing $\langle x, f(x)\rangle$ on input x and whose running time is within a constant factor of the running time of T. The cost in space is also polynomial in $|x|$, but all the tape cells used in the process of computing $\langle x, f(x)\rangle$ will be reset back to zero (reversibly). These tape cells are referred collectively as the* workspace.

Reversible Turing machines are such that at any point in the computation, two operations are possible: continue the computation forward one step or undo the previous step. For a more precise definition, see [Lan61] or the review in [BL85]. Benioff [Ben82] and Deutsch [Deu85] have shown that quantum Turing machines can directly simulate reversible Turing machines. Since quantum Turing machines (and thus also to quantum gate arrays) are reversible[4], we have the following corollary:

Corollary 3.3 *A Turing-computable function f is always computable on a quantum gate array (with a negligible increase in the time complexity).*

Consider the MDJP. The input function is computable so, by the Lecerf-Bennett theorem, there exists a reversible Turing machine that computes $\langle x, f(x)\rangle$ on input x. By definition of the problem, x is an n-bit value and $f(x)$ is a single bit. By corollary 3.3, this implies the existence of a unitary matrix F that computes f on n-bit values in the following sense. Consider the quantum gate corresponding to F:

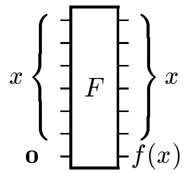


$$F|x, \mathbf{0}, \mathbf{0}^m\rangle \longrightarrow |x, f(x), \mathbf{0}^m\rangle$$

This gate works on an $(n+1+m)$-qubit register; the top n qubits encode the input $x \in \{\mathbf{0}, \mathbf{1}\}^n$. Those qubits must have the same value before and after the gate: if they are changed during the computation itself, they must be returned to their initial value . The next qubit, initially set to $\mathbf{0}$, will have the value of $f(x)$ at the output of the gate. The last m qubits are the "workspace" that comes about in theorem 3.2: before and after the computation, they are set to $\mathbf{0}^m$, but within the gate itself, those qubits will be used and reversibly reset to zero afterwards. We do not specify the

[4]Apart from the observation, which is inherently irreversible.

exact circuitry of elementary gates within the F gate, but by theorem 3.2 and corollary 3.3 we are certain that it can be done in accordance with the quantum principles. Also, for clarity, we will *not* usually display the qubits used as workspace since they serve no purpose outside the gates themselves. Therefore, the above gate F will displayed as follow:



$$F|x, \mathbf{o}\rangle \longrightarrow |x, f(x)\rangle$$

A remark: informally speaking, unitarity means that information cannot be lost. This means that the value $f(x)$ cannot simply overwrite the $\mathbf{o}$ in the last qubit: those values ($f(x)$ and $\mathbf{o}$) must be combined in a way that allows the recovery of both values. The exclusive-or function is commonly used for this purpose. In the Dirac notation, if the initial of the register is $|x, b\rangle$ then the action of the F gate is actually

$$F|x, b\rangle = |x, b \oplus f(x)\rangle$$

Of course, if $b = \mathbf{o}$ then $F|x, \mathbf{o}\rangle = |x, f(x)\rangle$, which is what we wanted. This property of non-destructive writing will be important later on.

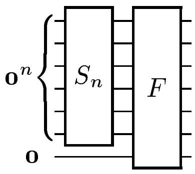
Computing a function on an input is fine, but the rules of quantum mechanics allow much more. Recall that by linearity of quantum operations, if the input state is in quantum superposition $\frac{1}{\sqrt{2}}(|x, \mathbf{o}\rangle + |y, \mathbf{o}\rangle)$ then the F gate will compute the superposition of f on both values.

$$F\left(\frac{1}{\sqrt{2}}(|x, \mathbf{o}\rangle + |y, \mathbf{o}\rangle)\right) = \frac{1}{\sqrt{2}}(|x, f(x)\rangle + |y, f(y)\rangle)$$

Assume there is a way to unitarily generate (through some matrix S_n) a superposition of all possible values of an n qubit register. That is, if the initial state of the register is all zero, S_n transforms it in a superposition of all 2^n values of the first n qubits.

$$S_n|\underbrace{\mathbf{o}\ldots\mathbf{o}}_{n}\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle$$

We can see that by first applying S_n and then F, we can compute in one sweep all possible values for f in quantum superposition.

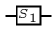


$$FS_n|\underbrace{\mathbf{0}\ldots\mathbf{0}}_{n},\mathbf{0}\rangle \longrightarrow F\left(\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i,\mathbf{0}\rangle\right) \longrightarrow \frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i,f(i)\rangle$$

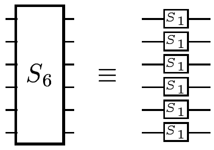
The reader should take careful note of what is meant by the above diagram. While the operator S_n acts on n qubits, its mathematical representation is a $2^n \times 2^n$ unitary matrix. Also, in the expressions below the diagram, it would be more accurate to use $S_n \otimes I$ (where I is the 2×2 identity matrix) since our gate array uses $n+1$ qubits. I trust the reader will be comfortable with this small abuse of notation throughout the text.

We now show how to implement an S_n gate to achieve this form of quantum parallelism. Consider the unitary matrix (and associated gate):

$$S_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad \Rightarrow$$



It is a simple matter to verify that S_1 is indeed unitary. Note also that $S_1^{-1} = S_1$. An S_1 gate is an elementary gate as it acts only on a single qubit: it sends $|\mathbf{0}\rangle$ to $\frac{1}{\sqrt{2}}(|\mathbf{0}\rangle + |\mathbf{1}\rangle)$ and $|\mathbf{1}\rangle$ to $\frac{1}{\sqrt{2}}(|\mathbf{0}\rangle - |1\rangle)$. The desired S_n gate acts on a quantum register by sending each qubit individually into a separate S_1 gate (an example on six qubits is shown here).

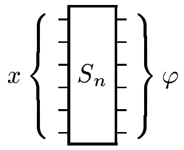


The unitary transformation induced by an S_n gate is given by the formula $S_n = \bigotimes_n S_1$. This has a nice recursive definition[5]: If $n > 1$ then

$$S_n = \begin{pmatrix} S_{n-1} & S_{n-1} \\ S_{n-1} & -S_{n-1} \end{pmatrix}$$

In gate form: for any $x \in \{\mathbf{0},\mathbf{1}\}^n$,

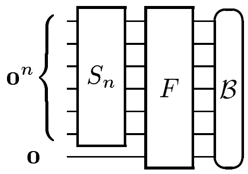
[5]Note: S_n is a special case of Hadamard matrices.



$$S_n|x\rangle \longrightarrow |\varphi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} (-1)^{x \cdot i} |i\rangle$$

where the operation $x \cdot i$ is the XOR of the bitwise AND of the strings x and i. Clearly, if x is set to $\mathbf{o}^n$, S_n performs the desired transformation. When outlining the proof of Simon's theorem, the transformation induced by S_n will be more fully used.

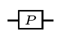
With the conjunction of S_n and F gates, a single computation produces all possible values of the function f for each input. But these values are in quantum superposition and we have seen (by the sixth principle) that only an observable can obtain information from a superposition (and this act destroys the original superposition). If our aim is to compute various outputs $\langle x, f(x)\rangle$ for all x, then the only observable that could be used is the standard one, $\mathcal{B}$ (see definition 2.6).



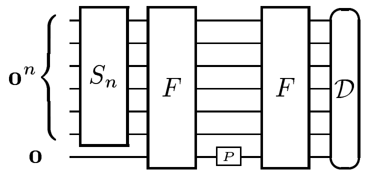
But $\mathcal{B}$ will only produce a single pair $\langle x, f(x)\rangle$ where x is chosen randomly (uniformly). To obtain all values of f in this fashion would require (on average) an exponential number of such runs. This could have been done just as easily using a probabilistic Turing machine by choosing x randomly and computing $f(x)$. Deutsch [Deu85] proved that quantum parallelism *used in this simplistic way* cannot produce that values of f any faster than classical machines. To get some form of benefit from superpositions, a more subtle use of quantum parallelism is needed.

Consider the following unitary transformation (and associated gate):

$$P = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad \Rightarrow$$



If a qubit is set to $\mathbf{o}$ nothing happens but if it is set to $\mathbf{1}$ then the amplitude is multiplied by -1. This gate "encodes" the "value" of the qubit into the sign of the amplitude. Now consider the following gate array



(where the observable $\mathcal{D}$ will be defined shortly). From our gate definitions, we know that the state $|\varphi\rangle$ of the register just after the P gate is:

$$|\varphi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} (-1)^{f(i)} |i, f(i)\rangle$$

When that state goes through the final F gate, the values for f are again computed and non-destructively combined using (in our case) the XOR function. Since $f(i) \oplus f(i) = 0$ for all $i \in \{\mathbf{o}, \mathbf{1}\}^n$, the final state before observation is

$$|\varphi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} (-1)^{f(i)} |i, \mathbf{o}\rangle$$

All the manipulations done so far had only one purpose: to transfer the values of f into the amplitudes relative to each of the basis states. The power of quantum computation resides in the interference of these amplitudes and the observable used to read the quantum states. We now define that observable. Consider $\mathcal{D} = \{E_a, E_b\}$ where the subspace E_a is the one-dimensional space spanned by

$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i, \mathbf{o}\rangle$$

and $E_b = (E_a)^\perp$, the orthogonal complement of E_a. Using $\mathcal{D}$ in the gate array above allows us to answer the MDJP, that is to determine without errors whether f is non-balanced or non-constant. To see this, recall that $\mathcal{D}$ will give the answer a or b with probabilities depending on the amplitudes of $|\varphi\rangle$ in the subspaces E_a and E_b. We must find the expression of $|\varphi\rangle$ in the basis defined by $\mathcal{D}$. This is easy since $\mathcal{D}$ has only two subspaces, one being one-dimensional. Let α and β be the projections of $|\varphi\rangle$ in E_a and E_b, then

$$|\varphi\rangle = \alpha|\psi\rangle + \beta|\psi_b\rangle$$

where $|\psi_b\rangle$ is a vector in E_b and of course, $|\psi\rangle \perp |\psi_b\rangle$. Observing the final state $|\varphi\rangle$ with $\mathcal{D}$ will give the answer a or b with probability $\|\alpha\|^2$ and $\|\beta\|^2$ respectively. Since the observable has only two possible answers, $\|\beta\|^2 = 1 - \|\alpha\|^2$. Also, finding the projection of $|\varphi\rangle$ in the one-dimensional subspace E_a is simple. We now compute the exact expression for α, the projection of $|\varphi\rangle$ along $|\psi\rangle$.

$$\begin{aligned}
\alpha &= \langle\psi|\varphi\rangle \\
&= \left(\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}\langle i,\mathbf{o}|\right)\left(\frac{1}{\sqrt{2^n}}\sum_{j=0}^{2^n-1}(-1)^{f(j)}|j,\mathbf{o}\rangle\right) \\
&= \frac{1}{2^n}\sum_{i=0}^{2^n-1}\sum_{j=0}^{2^n-1}(-1)^{f(j)}\langle i,\mathbf{o}|j,\mathbf{o}\rangle
\end{aligned}$$

But since $\langle i,\mathbf{o}|j,\mathbf{o}\rangle = 1$ if and only $i = j$ and zero otherwise, the expression for α simplifies to

$$\alpha = \frac{1}{2^n}\sum_{i=0}^{2^n-1}(-1)^{f(j)}$$

We now look at the value of α for different functions f. If f is a balanced function, the sum for α will contain exactly as many 1's as -1's, so in this case $\alpha = 0$ and $\mathcal{D}$ will always give a b answer and never a. If f is a constant function, the value for α will either be 1 or -1, so in this case $\mathcal{D}$ always gives the answer a and never b. If f is of any other type, $\mathcal{D}$ will answer a or b with various probabilities.

To demonstrate that the above quantum gate array solves the MDJP, we need to take the above reasoning backwards. If the answer received from $\mathcal{D}$ is a, we know for certain that f could not have been a balanced function (since a is never given in that case), so answering "non-balanced" is correct. Similarly. if $\mathcal{D}$ gives the answer b, then we know for certain that f could not have been a constant function, so answering "non-constant" is correct. For cases where f is neither of these those case, $\mathcal{D}$ might give any of a and b, but this is not a problem since both answers "non-balanced" and "non-constant" are correct.

4 Separating Two Classes of Functions

The solution to the modified Deutsch-Jozsa problem, like most interesting quantum algorithms (or gate arrays) depends on the ability to evolve an n-qubit register in superposition of all 2^n values. In the solution we presented, this operation was performed by the S_n gate. But the transformation induced by S_n is much more subtle. Recall that

$$S_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

and

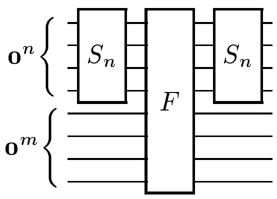
$$S_n = S_1 \otimes S_{n-1} = \begin{pmatrix} S_{n-1} & S_{n-1} \\ S_{n-1} & -S_{n-1} \end{pmatrix}$$

If an n-qubit register, initially set to $x \in \{\mathbf{0},\mathbf{1}\}^n$, goes through an S_n gate, the transformation will be as follow:

$$S_n|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=1}^{2^n-1} (-1)^{x \cdot i} |i\rangle$$

Where $x \cdot i$ is the XOR of the bitwise AND of the two strings. We now show how Simon [Sim94] used this transformation to prove theorem 3.1.

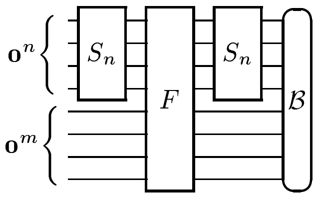
Assume we have a computable function $f : \{\mathbf{0},\mathbf{1}\}^n \rightarrow \{\mathbf{0},\mathbf{1}\}^m$, where $m \geq n$. The Lecerf-Bennett theorem still applies, so there exists a quantum gate F that transforms $|x, b\rangle$ into $|x, b \oplus f(x)\rangle$ for all $x \in \{\mathbf{0},\mathbf{1}\}^n$ and $b \in \{\mathbf{0},\mathbf{1}\}^m$. Consider the following gate array:



$$S_n F S_n |\mathbf{0}^n, \mathbf{0}^m\rangle \longrightarrow \frac{1}{2^n} \sum_{i=0}^{2^n-1} \sum_{j=0}^{2^n-1} (-1)^{i \cdot j} |j, f(i)\rangle$$

The first application of S_n allows all values of f to be computed using quantum superposition (with the F gate). The second application of S_n creates an elaborate entanglement of the states $|j, f(i)\rangle$ whose phases are a function of both i and j. In fact, the output state of the gate array is a form of Fourier spectrum of the function f. With this gate array, Simon was able to distinguish efficiently two classes of function: 1-to-1 versus 2-to-1 with a mask.

A function $f : \{\mathbf{0},\mathbf{1}\}^n \rightarrow \{\mathbf{0},\mathbf{1}\}^m$ is said to be 2-to-1 with a mask s if there exists a non-trivial $s \in \{\mathbf{0},\mathbf{1}\}^n$ such that for all $x \neq x'$, $f(x) = f(x')$ if and only if $x' = x \oplus s$ (where $\oplus$ is the bitwise XOR). Suppose we are given a computable function $f : \{\mathbf{0},\mathbf{1}\}^n \rightarrow \{\mathbf{0},\mathbf{1}\}^m$ with a promise that it is either 1-to-1 or 2-to-1 with a mask. The task is to determine which of these hold for f and, in the second case, produce s. Simon proved that this problem can be solved in expected time $O(nT_f(n) + G(n))$, where $T_f(n)$ is the time to compute f on inputs of size n and $G(n)$ is the the required to solve an $n \times n$ linear system of equations over Z_2. The algorithm will call (on average) n times the following gate array:



To see how this gate array works, we must do a analysis similar to the one for the MDJP in section 3.1. Let $|\varphi\rangle$ be the state of the register just before the observation.

$$|\varphi\rangle = \frac{1}{2^n} \sum_{i=0}^{2^n-1} \sum_{j=0}^{2^n-1} (-1)^{i \cdot j} |j, f(i)\rangle$$

If f is 1-to-1, then all $|j, f(i)\rangle$ configurations are different, each with amplitude $\pm 1/2^n$. The observable $\mathcal{B}$ will yield any of those configurations with probability $1/2^{2n}$ and k repetitions of this subroutine will result in k configurations of $|j, f(i)\rangle$ distributed uniformly and independently.

However, if there exists a non-trivial s such that for all $x \in \{\mathbf{o}, \mathbf{1}\}^n$, $f(x) = f(x')$ if and only if $x' = x \oplus s$, then for all $i, j \in \{\mathbf{o}, \mathbf{1}\}^n$, the configurations $|j, f(i)\rangle$ and $|j, f(i \oplus s)\rangle$ are identical. Therefore, the amplitude $\alpha_{i,j}$ for a particular configuration is:

$$\alpha_{i,j} = \frac{(-1)^{i \cdot j} + (-1)^{(i \oplus s) \cdot j}}{2^n}$$

Two values are possible: if $j \cdot s = 0$ then $i \cdot s = (i \oplus s) \cdot j$ so $\alpha_{i,j} = 1/2^{n-1}$. Otherwise, $\alpha_{i,j} = 0$. This means that when the register is observed, only configurations such that $j \cdot s = 0$ can be seen. Repeating k times this subroutine will result in k configurations of this type chosen uniformly and independently.

In both of these cases, after an expected $O(n)$ repetitions, we can find n configuration $|j_1, f(i_1)\rangle, \ldots, |j_n, f(i_n)\rangle$ such that the equations $j_i \cdot s = 0$ are linearly independent. Solving this linear system yields a non-trivial s'. If f is 1-to-1, this s' is a random string and if f is 2-to-1 with mask, s' is that mask. Computing $f(\mathbf{o}^n)$ and $f(s')$ and comparing the values determines the status of f: if $f(\mathbf{o}^n) \neq f(s')$, then f is 1-to-1, otherwise f is 2-to-1 with $s = s'$ as the mask.

The proof of Simon's theorem rests on the interaction of phases induced by the double application of S_n (with a relativized version of the above problem). Shor's factoring algorithm uses the same trick, but with a refined version of S_n, called the quantum discrete Fourier transform, and more number theory. The next section will go over the quantum component of the factoring algorithm; the reader may consult [Sim94] to see how the relativized version of the above problem is used to prove Simon's theorem.

5 Shor's Factoring Algorithm

Every integer n has a unique decomposition in prime factors. However, finding this decomposition when n is large is a difficult computational problem. All known classical methods are resolutely inefficient (see [Adl94]) and even the best known classical algorithm, the number field sieve (see [LLMP90], [LL93]) requires time $O(e^{c(\log n)^{1/3}(\log\log n)^{2/3}})$, which is exponential in the size (the number of digits, i.e., $\log n$) of n. Whether the factoring problem is polynomial or not (classically) is still unknown. Yet the faith in hardness of this problem is such that the security of many classical cryptographic protocols is based of the impossibility of factoring efficiently.

Number theory offers another interesting problem: finding the order of an element. Given x and n, find r (called the *order*) such that $x^r \equiv 1 \pmod{n}$. As with the factoring problem, no efficient algorithm is known for solving this problem. But while these problems appear very different, they are closely related. Miller [Mil76] has shown that, using randomization, one could solve the factoring problem *if* one had access to an oracle for finding the order of an element. His reduction works as follows: first, make sure that n is odd and not a prime (there are efficient primality testing algorithms). Then, use the following algorithm:

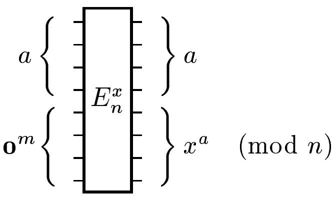
Program *One-Factor* (input: n *odd integer*)
$x \leftarrow$ random$\{0,\ldots,n\}$
$r \leftarrow$ *use the oracle to find the order of* $x \pmod{n}$
Output: if r *is odd* or $x^{r/2} \equiv -1 \pmod{n}$ then *fail*
else *return* $\gcd(x^{r/2}-1, n)$

Choosing a random number in the range $\{0,\ldots,n\}$, doing the modular exponentiation and finding the gcd (greatest common divisor) can all be done in polynomial time (see [Knu81]). Let k be the number of odd prime factors of n. One can prove that, provided n is odd and non-prime, the above algorithm will return a prime factor of n with probability at least $1 - 1/2^{k-1}$. Repeating this algorithm a polynomial number of times will produce a complete factorization of n.

Shor's breakthrough was to discover an efficient quantum algorithm to find the order of an element. The factoring algorithm is simply Miller's reduction where the oracle call is replaced by a call to this quantum algorithm. The next section describes how to find the order of an element using quantum superpositions.

5.1 Finding the Order of an Element

We now describe Shor's algorithm to find the order r of an element $x \pmod{n}$. There are two distinct parts to algorithm: the first is the quantum component, described next, which produces a value c. Thanks to appropriety chosen amplitudes, this c has a relationship to r such that a little (purely classical) post-processing in the second part can efficiently determine r. We describe the quantum component using quantum gate arrays. First, we need to find m such that $n^2 \leq 2^m \leq 2n^2$. The gate array operates on a $2m$-qubit quantum register. Next, we need a gate such that on input $|a, 0\rangle$, it computes $|a, x^a \bmod n\rangle$. We known that modular exponentiation can be done classically in polynomial time. So by the Lecerf-Bennett theorem and corollary 3.3, there exists a quantum gate E_n^x that efficiently implements this operation. This E_n^x gate is shown below.

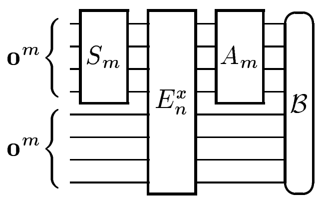


We only need one more quantum operation. Shor refined the S_n transformation used by [BV93] and [Sim94] in the following way: instead of using phases that were $\pm 1/\sqrt{2^m}$, we now make use of the full spectrum of complex amplitudes. The transformation A_m sends a m qubit register in basis state $|a\rangle$ to

$$\frac{1}{\sqrt{2^m}} \sum_{c=0}^{2^m-1} \mathrm{e}^{\frac{2\pi i a c}{2^m}} |c\rangle$$

(Recall that for any $a + bi \in \mathrm{C}$ of norm 1, there exists an angle $\theta \in [0, 2\pi]$ such that $a + bi = \cos\theta + i \sin\theta = \mathrm{e}^{i\theta}$.) This transformation is called the discrete quantum Fourier transform. The fact that one can efficiently implement such a quantum gate is not immediately clear, if only for the fact that the amplitudes seem to require increasing precision as m grows large. However, Deutsch and Coppersmith [Cop94] independently found an efficient solution based on the Fast Fourier Transform algorithm [Knu81], which only requires $O(m^2)$ elementary quantum gates.

The gate array for Shor's algorithm to find the order r of an element x (mod n) is:



The S_n gate was defined in the previous section and only serves to generate a superposition of all possible values for the top half of the register. We then compute in quantum parallel the modular exponentiation of x for all these values and then apply the Fourier Transform A_m. The state of the register just prior the observation: is (omitting the (mod n) in the ket for clarity):

$$\frac{1}{2^m}\sum_{a=0}^{2^m-1}\sum_{c=0}^{2^m-1} \mathrm{e}^{\frac{2\pi iac}{2^m}}|c, x^a\rangle$$

Since we are using the standard observable, the observation will yield any basis state $|c, x^k\rangle$ with probability

$$\left\|\frac{1}{2^m}\sum_{a:x^a\equiv x^k} \mathrm{e}^{\frac{2\pi iac}{2^m}}\right\|^2$$

Shor proves that this probability vanishes everywhere except for basis states $|c, x^k\rangle$ such that there exists an integer d satisfying

$$\left|\frac{c}{2^m}-\frac{d}{r}\right| \leq \frac{1}{2^{m+1}}$$

where the probability is at least $1/3r^2$. This means that reading the final state of the register will yield with high probability a value c such that the fraction $c/2^m$ is close to d/r. Because $2^m > n^2$, there is only one fraction d/r that satisfies the above equality while keeping $r < n$. The algorithm for finding that fraction d/r from $c/2^m$ is the post-processing we referred to earlier and can be done efficiently by continued fraction expansion (see [Knu81]). This produces the r we needed.

For a more detailed study of Shor's algorithm including the necessary number theory which was left out here, see [Sho95] and [EJ96]. Shor's algorithm and Simon's theorem are two of the most important results in quantum complexity theory. Both are strong arguments in favor of the superiority of quantum computing models over classical ones. But if new efficient algorithms are developed on quantum machines, it would be nice to have actual quantum machines on which to run them! The next section considers the obstacles to building quantum mechanical computers.

6 Building a Quantum Computer

Quantum computers offer capabilities unmatched by classical Turing machines. But if quantum computing models could *in principle* be constructed, there are enormous practical issues still to overcome before reaching this goal. The most serious of these obstacles is the preparation and manipulation of macroscopic physical systems in quantum superposition. This section outlines the difficulties and possible solutions to this problems.

Qubits were defined in section 2.2 as any object having two distinct states whose evolution is considered according to the principles of quantum mechanics. Following those principles, it is possible to have this object in quantum superposition, which permits quantum parallelism. But while experimental physicists have been observing and manipulating atomic and sub-atomic particles in quantum superposition, no one has yet claimed to have observed a lamp-post exhibiting a similar behavior. Why? The explanation has to do with *decoherence*: the process by which a system in quantum superposition decays to a classical state because of interaction with the environment.

We illustrate the problem as follow: a qubit in state $|\varphi\rangle = \alpha|\mathbf{0}\rangle + \beta|\mathbf{1}\rangle$ is put inside a black box. If the box is perfectly sealed, sheilding its interior from the rest of the universe, the qubit remains in state $|\varphi\rangle$ indefinitely. But perfect isolation is impossible: some energy in one form or another always leaks through the box, carrying traces of information of the box's content. Consider a very simple case: a stray electron in state $|e_s\rangle$ enters the box and interacts with the qubit. The interaction is such that the electron leaves the box in state either $|e_\mathbf{0}\rangle$ or $|e_\mathbf{1}\rangle$ depending whether the qubit was in basis state $|\mathbf{0}\rangle$ or $|\mathbf{1}\rangle$. In Dirac notation, this sequence of events is described as follows. Initially, we have two independent systems: a qubit in state $|\varphi\rangle$ and the electron in state $|e_s\rangle$. Since they are independent, their joint state is

$$|\varphi\rangle \otimes |e_s\rangle = (\alpha|\mathbf{0}\rangle + \beta|\mathbf{1}\rangle) \otimes |e_s\rangle$$

However, once the electron enters the box, it interacts with the qubits. As it leaves the box, the joint state becomes

$$\alpha(|\mathbf{0}\rangle \otimes |e_\mathbf{0}\rangle) + \beta(|\mathbf{1}\rangle \otimes |e_\mathbf{1}\rangle)$$

The qubit is still in the box and the electron is on its way yonder, but they now form an entangled system. If the state of the electron is now observed in any way (and here any interaction with an object in the lab is considered an observation), the states of the electron will collapse. For simplicity, assume the electron collapses to either $|e_\mathbf{0}\rangle$ or $|e_\mathbf{1}\rangle$. Since the electron and the qubit are entangled, the collapse of one causes the collapse of the other: the electron-qubit system will be in state $|\mathbf{0}\rangle \otimes |e_\mathbf{0}\rangle$ with probability $\|\alpha\|^2$ and in state $|\mathbf{1}\rangle \otimes |e_\mathbf{1}\rangle$ with probability $\|\beta\|^2$. The qubit spontaneously collapses to either $|\mathbf{0}\rangle$ or $|\mathbf{1}\rangle$ (in accordance with the electron's collapse) and the quantum superposition lost.

No matter how well qubits are isolated, random energy exchanges between the environment and the qubits will cause some decoherence on a time scale that depends on the medium used for a qubit and the conditions under which it operates. In the best cases, coherence is kept for some 10^4 seconds and in the worst cases, hardly 10^{-10} seconds. And these figures are for a single qubit only; some decoherence models show the decoherence time dropping exponentially as the number of qubits increases (see [Unr95] and [MSE95]). But keeping a qubit in quantum superposition is only part of the problem. A quantum computer will have to perform operations on that qubit. The time needed to perform an operation also depends on the medium used for a qubit and the conditions under which it operates. Unfortunately, the quick-action qubits are precisely those that interact easily with the environment, i.e. those having the shortest coherence time (see [DiV95]). The faster the operations can be performed, the less time there is to perform them!

Yet hope still remains. Shor's discovery attracted enough attention that more and more breakthroughs are coming from experimental physics. Many proposals for constructing a quantum computer already exist, such as [Fey86],[SW94], [CY94], [Llo93] or [DiV95]. Currently, a proposal by Pellizzari, Gardener, Cirac and Zoller using trapped ions technology appears very promising [PGCZ95] and the authors even suggest a way to control to a certain extent the decoherence in their implementation. An alternative approach proposed by Deutsch could allow computation on a less than perfect quantum state through a stabilizing scheme (the scheme is outlined in [BDJ94] and a preliminary analysis is given in [Ber95]).

In view of this, it seems unlikely that a general purpose quantum computer will be available in the near future. But technological advances in this field are appearing at an increasing rate. Some researchers are already talking about controlling 3 or 4 qubits for a few operations within ten years. This may not be much of a computer, but it would still be quite an achievement! A more reasonable goal could be to have small special purpose quantum machines. For example, considering that cryptography plays such an important role in today's world, a quantum factoring module would have important consequences. History does have a tendency to repeat itself; were not the first classical computers used for code breaking?

Acknowledgments: I would like to thank the many people who attended my seminar on quantum computation at CWI in the spring of 1995. Their questions and comments helped to put together the material for sections 2 and 3. I would also like to thank Janos Simon, Lance Fortnow, Stuart Kurtz, Amber Settle and Sophie Laplante for many interesting discussions as well as providing me with an office during my short visit to the computer science department of the University of Chicago, where part of this article was written. Also many thanks to Gilles Brassard, Paul Vitanyi, Jim Royer, Amber Settle, Sophie Laplante, Harry Buhrman, Jaap-Henk Hoepman, Barbara Terhal and Alain Tapp for their numerous comments and improvements on early drafts.

7 References

[Adl94] L. M. Adleman. Algorithmic number theory — the complexity contribution. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, pages 88–113, 1994.

[BB92a] A. Berthiaume and G. Brassard. Oracle quantum computing. In *Proceedings of the Workshop on Physics and Computation – Physcomp '92*, pages 195–199. IEEE Press, October 1992.

[BB92b] A. Berthiaume and G. Brassard. The quantum challenge to structural complexity. In *Proceedings of the 7th Annual IEEE Conference on Structure in Complexity*, pages 132–137, 1992.

[BB94] A. Berthiaume and G. Brassard. Oracle quantum computing. *Journal of Modern Optics*, 41(12):2521–2535, 1994.

[BBB+92] C. H. Bennett, F. Bessette, G. Brassard, L. Salvail, and J. Smolin. Experimental quantum cryptography. *Journal of Cryptology*, 5(1):3–28, 1992.

[BBC+95] A. Barenco, C. H. Bennett, R. Cleve, D.P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Physical Review Letters A*, 1995.

[BBE92] C. H. Bennett, G. Brassard, and A Ekert. Quantum cryptography. *Scientific American*, pages 50–57, October 1992.

[BDJ94] A. Berthiaume, D. Deutsch, and R. Jozsa. The stabilisation of quantum computations. In *Proceedings of the Workshop on Physics and Computation – Physcomp '94*, pages 60 – 62, 1994.

[Ben73] C. H. Bennett. Logical reversibility of computations. *IBM Journal of Res. Develop.*, 17:525–532, 1973.

[Ben82] P. A. Benioff. Quantum mechanical hamiltonian models of turing machines. *Journal of Statistical Physics*, 29(3):515–546, 1982.

[Ben96] P. A. Benioff. Review of quantum computation. To appear in *Trends in Statistical Physics* by *Council of Scientific Information*, Trivandrum, India, 1996.

[Ber95] A. Berthiaume. *L'ordinateur quantique: complexité et stabilisation des calculs.* PhD thesis, Dept. d'informatique et de recherche opérationelle, Université de Montréal, 1995.

[BL85] C. H. Bennett and R. Landauer. Physical limits of computation. *Scientific American*, page 48, July 1985.

[Bra93] G. Brassard. Cryptology column — quantum cryptography: A bibliography. *Sigact News*, 24(3):16–20, 1993.

[BV93] E. Berstein and U. Vazirani. Quantum complexity theory. In *Proceedings of the 25th Annual ACM Symposium on the Theory of Computation*, pages 11–20, 1993.

[Cop94] D. Coppersmith. An approximate fourier transform useful in quantum computing. Technical report, IBM Research Division, 1994.

[CY94] I. Chuang and Y. Yamamoto. A simple quantum computer, 1994. Submitted to *Physical Review A*.

[Deu85] D. Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society,* London, A400:97–117, 1985.

[Deu89] D. Deutsch. Quantum computational network. *Proceedings of the Royal Society,* London, A425:73–90, 1989.

[DiV95] D. P. DiVincenzo. Two-bit gates are universal for quantum computation. *Physical Review Letters A*, 50(1015), 1995.

[DJ92] D. Deutsch and R. Jozsa. Rapid solutions of problems by quantum computation. In *Proceedings of the Royal Society,* London, volume A439, pages 553–558, 1992.

[EJ96] A. Ekert and R. Jozsa. Shor's quantum algorithme for factorising numbers. *Review of Modern Physics*, 1996. to appear.

[Fey82] R. P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6/7):467–488, 1982.

[Fey86] R. P. Feynman. Quantum mecanical computers. *Foundation of Physics*, 16(6):507–531, 1986.

[FLS64] R. P. Feynman, R. B. Leighton, and M. Sands. *The Feynman Lectures on Physics*, volume 3. Addison-Wesley, 1964.

[Knu81] D. E. Knuth. *The Art of Computer Programming*, volume 2. Addison-Wesley, 1981.

[Lan61] R. Landauer. Irreversability and heat generation in the computing process. *IBM Journal of Research Development*, 5(183), 1961.

[Lec63] Y. Lecerf. Machines de Turing réversibles. Récursive insolubilité en $n \in N$ de l'équation $u = \theta^n$ où θ est un isomorphisme de codes. In *Comptes rendus de l'Académie française des sciences*, volume 257, pages 2597 – 2600, 1963.

[LL93] A. K. Lenstra and H. W. Lenstra, Jr. The devellopment of the number field sieve. *Springer-Verlag's Lecture Notes in Mathematics*, (1554), 1993.

[LLMP90] A. K. Lenstra, H. W. Lenstra, Jr., M. S. Manasse, and J. M. Pollard. The number field sieve. In *Proceedings of the 22nd Annual ACM Symposium on the Theory of Computation*, pages 564–572, 1990.

[Llo93] S. Lloyd. A potentially realizable quantum computer. *Science*, 261:1569 – 1571, September 1993.

[Mil76] G. L. Miller. Riemann's hypothesis and tests for primality. *Journal of Computer Science*, 13:300–317, 1976.

[MSE95] Palma G. M., K.-A. Suominen, and A. Ekert. Decoherence in quantum registers, 1995. preprint.

[PGCZ95] T. Pellizzari, S. A. Gardiner, J. I. Cirac, and P. Zoller. Decoherence, continuous observation and quantum computing: A cavity QED model, 1995. Submitted to *Physical Review Letters*.

[Sch95] B. Schumacher. On quantum coding. *Physical Review Letters A*, 1995. (in press to appear).

[Sho94] P. W. Shor. Algorithms for quantum computation: Discrete log and factoring. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, pages 20–22, 1994.

[Sho95] P. W. Shor. Polynomial-time algorithms for prime factorisation and discrete laogarithms on a quantum computer. *Submitted to SIAM Journal of Computing*, 1995.

[Sim94] D. Simon. On the power of quantum computation. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, pages 116–123, 1994.

[SW94] T. Sleator and H. Weinfurther. Realizable universal quantum logic gates, 1994. preprint.

[Unr95] W. G. Unruh. Maintaining coherence in quantum computers. *Physical Review Letters A*, 51:992–997, 1995.

[Yao93] A. C.-C. Yao. Quantum circuit complexity. In *Proceedings of the 34th IEEE Symposium on Foundations of Computer Science*, page 352, 1993.

Index