Quantum Informatics: A Survey

Carlo Alberto Furia

January 2006

Abstract

We survey the emerging field of quantum computing by showing its motivations and founding principles, and by presenting the most significant results in some selected areas. In particular, we outline the topics of quantum information, quantum algorithms, quantum cryptography, quantum finite automata, quantum error correction, and the physical realizations of quantum computing systems.

Contents

1 Introduction

All of science is a method for reliably answering questions. And it often happens that the deepest and most insightful of science's theories grow out of efforts to give *new, deeper* answers to previously, less satisfactorily tackled, *basic* questions [Kuh96]. Quantum computing fits this framework quite well, as it can be regarded to as a novel answer to the basic question "What does it mean 'to compute'?" [Nie00].

1.1 What Does It Mean 'to Compute'?

Information is physical.
— Rolf Landauer [Lan91].

1.1.1 Defining Computation: the Church-Turing Thesis

Although the idea of computation is a very ancient one, the earliest formal definitions of it were first put forward during the 1930's, with some famous seminal works by Turing [Tur36] and others (such as Church [Chu36, Chu41], Kleene [Kle36], and Post [Pos36]), who founded the field of computation theory. In their works, these pioneers proposed different mathematical formalisms that tried to capture the very idea of 'computing'. All these disparate formalisms (i.e., Turing machines, λ-calculus, recursive functions, etc.) were soon shown to be *equivalent* in expressive power. This fact has cogently suggested that any of them could be elected to be a suitable *formal definition* of the informal idea of 'computing'. This idea — that a Turing machine captures precisely the notion of 'computable' — has been deemed so significant to be taken as a principle under the name of "Church-Turing thesis" (CTT), from the names of the first two persons who formulated the ideas in their works, that is Church [Chu36] and Turing [Tur48]. Let us introduce a "modern" statement of the thesis.

Principle 1 (Church-Turing Thesis [Yao03]). The most general concept of *computability* is captured by the Turing machine model, or an equivalent formalism.

Over the decades, and through the practice and experience of the developing theory of computational complexity, the Church-Turing thesis has been quantitatively refined into an extended version. This strengthening says that not only does the Turing machine model captures completely the intuitive idea of computation, but it also captures its complexity. More precisely, we have the following "Extended Church-Turing thesis" (ECTT).

Principle 2 (Extended Church-Turing Thesis [Yao03]). If a function is computable by some device in time $T(n)$ for input of size n, then it is computable by a Turing machine in time $(T(n))^k$ for some fixed k (dependent on the problem).

That is, everything which is computable is also computable by a Turing machine, with a cost which is equivalent *within a polynomial*.

1.1.2 The Object of the Church-Turing Thesis

Let us now ask ourselves: what is the *object* of the CTT?

Traditionally, the CTT has been considered a statement *about mathematics*. Namely, the CTT defines what is the subset of all mathematics which can be automated (i.e., which is computable), while the ECTT defines what is the subset of all mathematics which can be efficiently automated. For a beautiful discussion on the interpretation of the CTT as a statement about mathematics see [Hof99, Chap. 17].

However, in more recent years, it has first been observed by Deutsch [Deu85] that the CTT should more appropriately be considered a statement *about physics*. Namely, we consider computation as a physical process, that is one which is realized by an evolving physical system. Therefore, the CTT means that every physical process can be *simulated* by a Turing machine. This physical formulation of the CTT is sometimes named "Church-Turing-Deutsch Principle" (CTDP) to include the names of all its contributors.

Principle 3 (Church-Turing-Deutsch Principle [Deu85]). Every finitely realizable physical system can be perfectly simulated by a Turing machine, or an equivalent formalism.

Finally, the extension of the CTDP to the quantitative case (which we will refer to as "Extended CTDP" (ECTDP)) is straightforward: every finitely realizable physical system can be perfectly simulated by a Turing machine, with a cost which is equivalent within a polynomial.

This shift in the perspective by which we consider the CTT has very significant implications. In particular, we are interested in the following consequence. Since the CTDP is a statement about the physical world, it is amenable to validation, or more precisely to refutation by experiment. Conversely, given a set of well-defined physical laws that have been validated through experiments, it should be possible to *derive* the CTDP as a consequence of these laws, by showing explicitly that all the systems described by the laws are (efficiently) simulable with a Turing machine. Notice that these characterizations were not elicited by the mathematical classical interpretation of the CTT.

1.2 Non-Standard Models of Computation

Therefore, an interesting research direction to follow consists in taking some relevant classes of physical systems, described by some known mathematical laws, and consider them as computing systems. Then, it should be possible to validate or refute the CTDP against them.

This has been pursued rather extensively in the last two decades or so. The results have been a variety of models of computations based on physical systems. The paradigm of quantum computing is one of them, and it is the object of this paper. Before focusing on it, we now consider and briefly describe some other physical models of computation.

Analog computing Physical laws usually model quantities by means of continuous variables. Thus, in analog computation some computational models relying on continuous variables are introduced and studied. In particular, there are both proposals for continuous variables still managed in a discrete-time setting (see e.g. [BCSS97]), and fully continuous models, where time is a continuous as well (see [Orp97] for a survey).

DNA computing exploits the molecular processes that involve replication and manipulation of DNA strands to perform computation. The idea was first proposed by Adleman [Adl94]. The present technology of experimental biology is sufficiently developed to implement in practice these microscopic computation devices.

Brane computing In (mem)brane systems [Pău00] (a.k.a. P systems), highly parallel systems of adjacent cells, communicating through their separating membranes, undergo some evolution until they reach a stable (halting) state. The mathematical models of brane systems are strong abstraction of actual biological cells, which should be the actual implementation medium.

Chaos computing [SD99, Sin99] studies dynamical systems that exhibit chaotic behavior and tries to exploit this behavior to perform some meaningful computation. Experiments have shown that these computational models can indeed be implemented in real chaotic systems. Chaos computing can be considered a peculiar form of analog computing, one where the chaotic behavior is explicitly exploited as a computational resource.

All of the above computational models suffer from one of two major shortcomings, with respect to quantum computing: either they are not really implementable, or they are not really more powerful than classical computing. With respect to the CTD principle, it means that either they are not really accurate physical models, or they can be perfectly simulated by a classical Turing machine, so they give no new insights to the very idea of computation.

Brane computing, for instance, falls into the first category: although the theoretical model of branes is demonstrably capable of solving efficiently combinatorial problem that we consider intractable (and in particular NP-complete problems), it is not a really implementable model, since it relies on the assumption that brane systems can relax to reach *global* energy minima. Actual experiments have shown that local minima (i.e., metastable states) are often reached in practice.

On the other hand, DNA computing is fully implementable, as it describes faithfully the actual DNA recombination processes that happen in living cells. Nonetheless, it seems to give no more computational power than ordinary Turing computing. In fact, the promising results in solving quickly combinatorial problems come from the fact that the abundance of biological material permits to exploit a high level of parallelism in solving problems. However, this parallelism is bounded, even if by large constants. Thus, from the complex-theoretical point of view, the model is perfectly equivalent to ordinary classical computing, only with a smart and very efficient implementation.

For a thorough discussion of the relative merits of some of these models of computation, as well as even more exotic ones, see the survey by Aaronson [Aar05].

1.3 Quantum Computing

Let us finally introduce quantum computing. In quantum computing, we use a quantum-mechanical system as computational model. This model promises to be more significant, from the point of view of computational complexity theory, than the other non-standard models of computations (some of which we outlined above). In fact, quantum computing models are:

- *implementable*, since quantum mechanics models accurately real physical systems and the interactions they can have. This has been demonstrated by a vast number of experiments;
- *computationally more powerful* than classical computing. In fact there are some problems which are provably exponentially more efficiently solvable in a quantum computing model than in a classical one;
- *robust*, since we can perform error correction, so to make implementations robust to noise. This is an essential feature for implementability, which several analog computing models cannot achieve.

Thus, quantum computing models *refute* the Extended Church-Turing-Deutsch principle. Therefore, we should change the principle by referring it to quantum computational models.

Principle 4 (Extended Church-Turing-Deutsch Principle, quantum version). Every finitely realizable physical system can be perfectly simulated by a quantum universal computing device, with a cost which is equivalent within a polynomial.

Organization of the paper. The remainder of the paper is organized as follows.

Section 2 presents a general framework for quantum computation, and describes its connections with quantum mechanics, as well as with classical computing models. More precisely, Section 2.2.1 gives some practical motivations for the study of quantum computing; Section 2.3 outlines the historical origins and first developments of the field; Sections 2.4 and 2.5 present a mathematical model of quantum computation, and show its connections with the postulates of quantum mechanics. Finally, Section 2.6 exploits the framework to show a couple of simple but interesting applications to computational and communication problems.

Afterwards, Section 3 presents several different subfields of quantum computation, and for each of them outlines the basic problems to be tackled and the most important practical solutions to them. Therefore, Section 3.1 is about the basics of quantum information. Section 3.2 deals with some interesting results in the design of algorithms for quantum computers. Section 3.3 presents some promising applications of the quantum computing paradigm to cryptographic tasks. Section 3.4 introduces some computational models that combine the quantum model with (finite) automata theory, and explores some relations among them from a language-theoretic point of view. Section 3.5 deals with the important problem of making a quantum computation robust to noise, showing how this may be possible in spite of the limitations imposed by quantum mechanics on the manipulation of states. Section 3.6 briefly outlines some candidate physical systems for the actual implementation of quantum computing devices, and refers to some actual experiments that have demonstrated the feasibility of quantum computing (at least to small scales).

Finally, Section 4 draws some summarizing conclusions.

The field of quantum computation is already developed to such an extent that giving a detailed recount of it is impossible; even treating exhaustively only some selected aspects would require a lot of space and many details. In this report, we have therefore simply tried to give the "big picture" of some interesting aspects within quantum computation, with particular emphasis on the most computer science oriented areas, thus necessarily sacrificing some technical detail. For further particulars, we refer the interested reader to the many references given in the text.

2 Quantum Computing: the Basics

2.1 Naming Issues

Although the field of quantum computing is relatively new (see Section 2.3), it has already developed into a conspicuous number of related subfields. Mirroring these areas, a number of different names denoting the field have appeared. The meanings of these names have substantial overlapping, although they are not fully synonyms.

Under these respects, let us consider some of these different names for quantum computing, and briefly outline what aspects they usually identify.

Quantum computing is probably the most widely used term to denote the field as a whole. Under a more specific meaning, it denotes the subfields centered about algorithms, computational models (**quantum computation**) and computational complexity, as opposed to the information-theoretic aspects.

Quantum information (theory) refers to the information-theoretical aspects of the field, such as the encoding of information with quantum systems, error correction models, the measurement of aspects of quantum systems such as entanglement, etc.

Quantum information sciences usually denotes the field as a whole, sometimes also under the name of **quantum informatics**.

Quantum information processing often refers to the most implementation-oriented aspects of the discipline, such as models for implementations in physical systems, models of noise, etc.

Quantum (something) Whenever one denotes very specifically applications of the quantum paradigm to some aspects or disciplines, one dub them with the word 'quantum'. So, for instance, we have quantum algorithms, quantum automata, quantum grammars, quantum circuits, quantum complexity (theory), quantum control (theory), quantum game theory, etc.

Although the above differences in meanings are indicative, they are not strict, and the naming is often used fairly liberally. Anyway, throughout this paper we will prefer the term *quantum computing* to speak about the subject matter, also mirroring the fact that we will focus on the computational aspects of the field.

2.2 A Definition of Quantum Computing

So, then, "mono" means "one", and "rail" means "rail". And that concludes our intensive three-week course.
— Simpsons Episode 9F10.

Quantum computing can be defined as a paradigm that exploits a *computational model* relying on the principles of *quantum mechanics.*

Quantum mechanics is the most accurate and reliable framework theory of the physical world that we currently have. Notice that quantum mechanics denotes, precisely, a *framework* within which it is possible to develop specific physical theories; it is not a full physical theory itself. In fact, as we will see in the remainder, quantum mechanics is formalized by some mathematical postulates which state abstractly how to describe a system, its evolution over time, the measurements we can perform on it, and the composition of different systems into a larger one. Then, some actual physical systems can be described with laws that conform to these basic axioms. For instance, we can then describe a semi-conductor material by means of a specific quantum-mechanical physical theory of solid state.

To better comprehend this difference, we can compare quantum mechanics to classical (Newtonian) dynamics. Newtonian dynamics describes some general features of physical systems, by defining the notion of force, linking it to cinematic concepts, and introducing the notion of (Galilean) relativity. Then, for instance, gravitation theory is a physical theory which is developed within the framework of Newtonian dynamics, that is which obeys its principles.

Why should we consider the quantum mechanical paradigm interesting to develop a theory of computation? The next subsection proposes some answers to this question.

2.2.1 Motivations for Quantum Computing

Nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy.
— Richard P. Feynman [Fey82].

We can identify at least three sets of motivations to consider quantum mechanics as a framework for a computational paradigm.

Bottom-Up Reasoning bottom-up, that is from a low level of abstraction to a higher one, we consider the current trend in the developments of computer processors implementations. We know that the miniaturization of integrated components seems to follow Moore's law [Moo65]: the capacity of a component of some size doubles every 24 months. Equivalently, the size of a transistor is halved every 24 months. In this strong push towards miniaturization, we will, sooner or later, reach the atomic scale, when a single transistor will be approximately the size of a single atom. At this point, quantum mechanical effects will become prominent, and we will have to deal with them in the construction of electronic components. Therefore, it makes sense to think in advance about the impact of these effects on the behavior of the computing component. Even more, we should consider if and how we can exploit the quantum mechanical effects of the atoms to perform more efficiently some form of elementary computation. Under this perspective, the quantum computing paradigm emerges.

Top-Down Reasoning top-down, that is from a high level of abstraction to a lower one, we consider quantum mechanics as an abstract and precise set of rules that describe some generic systems with a very particular behavior. Thus, it makes sense to consider it as a candidate to represent a notion of computation, in order to explore to what extent its peculiarities have some impact the computing tasks. Then, studying abstractly this paradigm, we realize that it is not only a theoretically stimulating idea, but it is also implementable in practice in real systems. Therefore, we can concretely design implementations that exploit its advantages in terms of computation.

Energy Consumption Finally, a motivation that is tightly connected to the bottom-up considerations is the one about the energy consumption issue.

One of the major obstacles to the miniaturization of digital components is energy consumption. From a purely thermodynamical point of view, a physical process which minimizes the energy consumption for some task is a *reversible* process. Although ordinary computation is not reversible, it has been shown by Landauer [Lan61] and Bennett [Ben73] that all computations can be made reversible exploiting suitable (non-standard) mechanisms.[1] Now, quantum mechanics describes intrinsically reversible processes, this being a fundamental property of its. Therefore, it seems that quantum-mechanical devices are the most natural way to implement reversible, and thus energy efficient, computation.

2.3 Historical Overview

Without any pretense of exhaustiveness, this section outlines the most important contributions to the birth and early development of quantum computing.

The physical and mathematical framework of quantum mechanics has been introduced during the 1930s by the work of many outstanding physicists such as Bohr, Heisenberg, Schrödinger, Dirac, Born, and many others. Moreover, from our perspective, von Neumann's contribution to the development of a mathematical axiomatization of the principles of quantum mechanics [vN55] is particularly significant, as it is a distillation of the very principles of quantum mechanics in an abstract coherent form which serves as basis for the quantum computing models.

The first investigations into the physical nature of computation were put forward in some foreseeing works, during the 1960's and 1970's, mainly by Landauer [Lan61], Bennet [Ben73], and Feynman [Fey60]. Although the idea of a quantum computer was still to come, these investigations were important in focusing the attention on the physical aspects of computation, and in particular on the idea reversibility.

Another key observation that suggested the introduction of quantum mechanical computational models was about the computational difficulty of simulating quantum systems with a classical computer. This was first observed by Manin [Man80] and by Feynman [Fey82] during the early 1980's.

In the immediately following years, Feynman [Fey82] and Benioff [Ben80, Ben82a, Ben82b] were the first to develop this observation into an explicitly quantum mechanical computational model. It was Feynman, however, who was able to really identify and introduce the peculiarities of quantum mechanics in a computational framework. On the contrary, Benioff's models are weaker and completely reducible to the standard Turing model.

It was during the mid 1980's that we see the first models that can be considered fully quantum mechanical computational devices. These were due again to the late Feynman [Fey85b, Fey86], and to Deutsch [Deu85]. Deutsch was also able to exhibit the very first simple examples of problems that can be solved with

[1] For a comprehensive survey of reversible computation see Bennet [Ben88] and Frank [Fra99].

a quantum computer exponentially faster than with a classical one. Deutsch's examples were developed into more complex ones with Jozsa in [DJ92] during the early 1990's.

During the same period — namely the mid 1980's — other important contributions were made in showing the advantages in applying quantum models to cryptography. In particular, Bennet and Brassard [BB84] proposed a protocol to perform *perfect cryptography* between two parties that communicate through a quantum channel. Their protocol was based on some ideas by Wiesner [Wie83] that, although were first put forward *circa* in the year 1970, were not accepted for publication until much later! Bennet and Brassard's protocol has given birth to a set of diverse and improved protocols to perform reliably cryptography.

So, at the beginning of the 1990's the quantum computing inchoate community already had some evidences of the superior power of quantum over classical computation, but was still in search of a true "killer application" for the quantum computer. In fact, the earliest evidences (i.e., mainly Deutsch and Jozsa's examples) were quite "artificial" and contrived, while it would have been significant to find similar results for some "natural" problems.

Such applications were soon found by Shor and Grover during the mid 1990's. Grover discovered a search algorithm [Gro96, Gro97] capable of searching for an element in an unordered array of n elements in time $O\left(\sqrt{n}\right)$, where the (obvious) classical best possible performance is $O\left(n\right)$ (as usual, we are considering worst-case performances). Even if the quantum speedup is polynomial (namely quadratic) and not exponential, this is nonetheless a very significant result since it involves the widespread problem of searching; therefore it has many derived applications.

Even more convincingly, Shor proposed a quantum algorithm to factor an n-bit integer in time $O\left(n^3\right)$ [Sho94]. Although the exact computational complexity of factoring is unknown, the best known classical algorithm takes superpolynomial time [LL94], and the factoring problem is widely considered to be intractable [Pap94, Sch95].

Grover's and Shor's breakthroughs sparkled a revived interest in quantum computing during the 1990's, and a significant enlargement of its research community. So far, however, the main results in quantum computation were mainly, if not solely, theoretical. Indeed, a fundamental obstacle for the concrete implementation of a quantum computing device remained: the spontaneous degeneration process known as *decoherence*. Nonetheless, this hurdle was soon overcome by a new work by Shor that showed how it is possible to perform error correction in a quantum computer [Sho95], thus permitting tolerance to faults caused by decoherence. Shor's new breakthrough was soon improved [Sho96, CS96], and similar results were independently achieved by Steane [Ste96]. Later, during the second half of the 1990's the methods and techniques for fault-tolerant quantum computation were generalized and largely improved by researchers such as Gottesman [Got96], Preskill [Pre01], and many others.

During the same years, the quantum computing theoretical model was perfected and compared against the classical one; among the others, we mention the seminal work by Bernstein and Vazirani [BV93, BV97]. Moreover, several proposals to realize the original ideas of simulation of quantum systems were put forward (see e.g. [AL97]).

Finally, during the late 1990's, thanks to the progresses in the error-correction models, it was finally possible to build the first real implementations of quantum mechanical computing device (see e.g. [CVZ$^+$98]). These implementations, although still very primitive ones, show that quantum computation can indeed be realized. Despite these deep and very promising results, some researchers have put forward some critiques to the very idea of quantum computing and to its concrete realizability. Discussing them is beyond the scope of this paper; we refer the reader to the thorough discussion by Aaronson in [Aarar].

2.4 A Mathematical Model of Quantum Mechanics

Quantum mechanics needs no 'interpretation'.
— Christopher A. Fuchs and Asher Peres [FP00].

The quantum mechanics framework can be completely described by means of four simple postulates. Before showing them, let us briefly introduce some mathematical notions that we are going to use, although we assume that the reader is familiar with linear algebra on complex spaces. For more complete expositions of the basic features of quantum computation, we refer the reader to the articles by Aharonov [Aha98], by Mermin [Mer03], and in particular to the textbook by Nielsen and Chuang [NC00], as well as the one by Gruska [Gru99].

2.4.1 Mathematical Preliminaries

Let us consider finite-dimensional vector spaces over the complex field $\mathbb{C}$. We denote the complex conjugate of a number $c \in \mathbb{C}$ as c^*.

We adopt the Dirac notation, where a generic vector ψ in a space is denoted by a *ket*:

$$|\psi\rangle$$

One can think of $|\psi\rangle$ as a generic column vector. The *dual* of the vector $|\psi\rangle$ is denoted by the *bra* $\langle\psi|$, which indicates a row vector.

Standard orthonormal basis. In a n-dimensional vector space, the *standard orthonormal basis* is denoted by the vectors

$$|0\rangle, |1\rangle, \ldots, |n-1\rangle$$

Thus, every vector $|\psi\rangle$ in the space can be expressed as the linear combination:

$$|\psi\rangle = \sum_{i=0,\ldots,n-1} c_i |i\rangle$$

where $c_i \in \mathbb{C}$ for all i.

Inner products. The *inner product* between two vectors $|\psi\rangle = \alpha_0 |0\rangle + \cdots + \alpha_{n-1} |n-1\rangle$ and $|\phi\rangle = \beta_0 |0\rangle + \cdots + \beta_{n-1} |n-1\rangle$ is defined as:

$$\langle\psi|\phi\rangle \;=\; {\alpha_0}^* \beta_0 + \cdots + {\alpha_{n-1}}^* \beta_{n-1}$$

By means of the inner product, we define the *norm* of a vector $|\psi\rangle$ as $|||\psi\rangle|| = \sqrt{\langle\psi|\psi\rangle}$. We call *unit vector* any vector which has unit norm.

Linear operators. *Linear transformations* on vectors are represented by matrices of complex numbers. If A is a linear operator (i.e., a matrix), we denote by A^* its *conjugate* matrix, by A^{T} its *transpose*, and by $A^\dagger$ its *adjoint* matrix, that is $A^\dagger = (A^*)^{\mathrm{T}}$. The same operations are defined on vectors (and in particular note the equivalence $|\psi\rangle^\dagger = \langle\psi|$).

Tensor product. The *tensor product* between two n-dimensional vectors $|\psi\rangle = \alpha_0 |0\rangle + \cdots + \alpha_{n-1} |n-1\rangle$ and $|\phi\rangle = \beta_0 |0\rangle + \cdots + \beta_{n-1} |n-1\rangle$ is denoted as $|\psi\rangle \otimes |\phi\rangle$, $|\psi\rangle |\phi\rangle$, $|\psi, \phi\rangle$, or simply $|\psi\phi\rangle$. It is defined as the n^2-dimensional vector:

$$|\psi\rangle \otimes |\phi\rangle \;=\; \sum_{\substack{i=0,\ldots,n-1 \\ j=1,\ldots,n-1}} \alpha_i \beta_j |ij\rangle$$

Unitarity and Hermiticity. A linear operator U is *unitary* when it satisfies $U^\dagger U = UU^\dagger = I$, where I is the identity matrix. It is simple to show that a unitary operator maps unit vectors to unit vectors. A linear operator H is *Hermitian* when it satisfies $H^\dagger = H$. Any Hermitian operator H can be written as the sum of projectors onto orthogonal subspaces as $H = \sum_i h_i P_i$, where P_i is the projector onto the subspace (eigenspace) with eigenvalue h_i.

Hilbert spaces. In the finite-dimensional case, a *Hilbert space* is a complex vector space with inner product. We need not consider the infinite-dimensional case.

2.4.2 The Postulates of Quantum Mechanics

We finally present the four postulated that defines quantum mechanics. They deal, respectively with:

1. the definition of the *state* of a quantum system
2. the definition of the *dynamical evolution* of the state over time
3. the definition of the *measurements* that we can perform on a quantum state
4. the definition of the *composition* of subsystems to form a larger system

The exposition follows closely Nielsen and Chuang [NC00, Section 2.2].

Postulate 1 (State of a Quantum System). A quantum system is completely described by a *state vector*, which is a unit vector in the Hilbert space associated to the system.

Postulate 2 (Evolution of the State). The evolution of a *closed* quantum system is completely defined by a *unitary* operator U acting on it.

Postulate 3 (Measurements). Any measurement on a quantum system is defined by a Hermitian operator $M = \sum_i m_i P_i$. The (only) possible outcomes of the measurements correspond to the eigenvalues m_i's. When measuring a generic state $|\psi\rangle$, the probability of getting the result m_i is:

$$p(m_i) \;=\; \langle\psi|P_i|\psi\rangle$$

If the measurement has yielded the result m_i, the state right after the measurement has *collapsed* to:

$$\frac{P_i\,|\psi\rangle}{\sqrt{p(m_i)}}$$

Postulate 4 (Composition of Systems). The state vector of a quantum system made by composing subsystems is given by the *tensor product* of the composed state vectors.

These four postulates completely describes abstractly a quantum system. The complete description of the system is given by a unit vector in complex vector space. This state vector changes over time according to unitary transformations. We can "access" the value of the state only by measuring it; measurements are stochastic events whose probability distributions are as in Postulate 3. After a measurement the state collapses, according to the outcome of the measurement, by following a projection and re-normalization. Finally, the size of a composed system grows as in the tensor product of the component subsystems.

2.4.3 Nature is Really Nonlocal

What is proved by impossibility proofs is lack of imagination.
— John S. Bell [Bel82].

The fact that quantum mechanical systems are composed only with tensor product is strictly linked to a fundamental — many say *the* fundamental — feature on quantum systems: *entanglement.* A state is entangled if it cannot be written as the tensor product of two states of (smaller) spaces. For instance, for a four-dimensional Hilbert space, the following state is entangled.

$$|\psi\rangle \;=\; \frac{|0\rangle\,|0\rangle + |1\rangle\,|1\rangle}{\sqrt{2}} \tag{1}$$

Entanglement has a dramatic impact on how systems behave. For example, consider the entangled state of Equation 1. There, the two composed subsystems *cannot be described in isolation*: the system only admits a global description. This is however completely independent of the spatial location of the two subsystems. In particular, the two subsystems could reside in locations which are arbitrarily distant. Nonetheless, the measurement of the state of one of these two subsystems would *immediately* determine the outcome of the measurement of the state of the other subsystems, despite their being arbitrarily far away.

To understand this, consider the observable $M = 1\left|0\right\rangle\left\langle 0\right| - 1\left|1\right\rangle\left\langle 1\right|$. If we measure the first subsystem according M, we get 1 with probability 1/2 and -1 with probability 1/2. If the outcome of the measurement is 1, the state after the measurement is $\left|0\right\rangle\left|0\right\rangle$, otherwise it is $\left|1\right\rangle\left|1\right\rangle$. Therefore, in the former case a measurement of the same observable on the second subsystem will yield 1 with certainty, while in the latter case it will yield -1 with certainty. This behavior is highly *non local*.

Since the birth of quantum mechanics, a central question has been about whether this non-local behavior is really observed in nature and whether the machinery of quantum mechanics is really necessary to describe it, that is if it is possible or not in classical systems as well. The positive answer to the first question soon came from a very large number of experiments. The positive answer to the second question came instead with the work of Bell who proved, in the mid 1960's [Bel64, Bel66], that no physical theory which is both realistic and local can produce outcome that are in accordance with quantum mechanics. Therefore, since quantum mechanics is experimentally confirmed, any reliable description of the world must be either non-realistic or non-local. Realism is usually considered a fundamental ingredient of a physical theory, so quantum mechanics explains the strange behavior of entangled states in terms of a non-local theory. These non-local features plays an important role in determining the additional power of quantum computation with respect to classical computation.

2.5 A Mathematical Model of Quantum Computation

The qubit. In describing a computational model, we have, first of all, to describe the basic unit of information, that is the computational state. In quantum computing, we model the *basic unit of information* with a quantum state vector in a 2-dimensional Hilbert space, and call it *qubit*. In analogy with the classical case (i.e., classical bits), we denote the standard basis vectors that span the space as $\left|0\right\rangle$ and $\left|1\right\rangle$. Therefore, the generic qubit is a superposition (i.e., a unit-norm linear combination with complex coefficients) of these vectors.

Notice that, although the qubit is the quantum version of the bit, there is a fundamental difference in how one can access the information stored in a qubit with respect to a bit. In fact, accessing the information of a qubit must obey the rules of quantum measurement. Therefore, reading a qubit yields a probabilistic result, and, in general, changes the value of the state.

Multiple qubits. Larger pieces of information can be described by *multiple qubit* systems. In general, an n-qubit systems has a state which has the form:

$$|\psi\rangle \;=\; \sum_{i\in\{0,1\}^n} \alpha_i\,|i\rangle$$

with the constraint that $\sum_{i\in\{0,1\}^n} |\alpha_i| = 1$. Notice that an n-qubit system is in fact a 2^n-dimensional Hilbert space.

Quantum gates. Now that we have defined what kind of information is represented in quantum computing, that is how is the "memory" of a quantum computer, we specify how we can manipulate it, that is how to do the actual computation on the data. Here we consider the so called *quantum circuit* model. Just like classical bits are modified by (Boolean) gates, qubits are modified by *quantum gates*.

A quantum gates modifies an n-qubit state according to a unitary transform. We represent it graphically as a box with the name of the unitary matrix on it, and a number of wires on the left (representing the number of qubits of the state being transformed) and on the right (representing the number of qubits of the resulting state). Since we are dealing with unitary transformation — which are therefore reversible — the number of qubits entering and exiting the gate must be the same. As an example, Figure 1 pictures a single-qubit gate with unitary transform represented by the matrix X. Therefore, the generic qubit $|q_0\rangle$ would take the value $X\,|q_0\rangle$ *after* being transformed through the gate X.

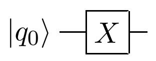

Figure 1: A single-qubit quantum gate.

Let us now define some of the most important single-qubit gates. The first ones are the four *Pauli matrices*.

$$I = \sigma_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad X = \sigma_x = \sigma_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$Y = \sigma_y = \sigma_2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad Z = \sigma_z = \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Another useful gate is the *Hadamard* transform, defined by the matrix

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

We also have the *phase* gate (denoted S), and the $\pi/8$ gate (denoted T).

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \qquad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$$

Finally, we also define an important two-qubit gate, the C-not gate (i.e., Controlled not), represented graphically in Figure 2.

$$C_{\text{not}} = |0\rangle \langle 0| \otimes I + |1\rangle \langle 1| \otimes X$$

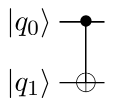

Figure 2: The C-not quantum gate.

Universality. For Boolean operations on "classical" bits, we know that any arbitrarily complex Boolean function can be represented exactly by means of a finite set of two-bit operations. We call such a set *universal*, since it can be used to represent any Boolean function. A well known example of a universal set for classical bits is the one made of the *not* $\neg$ and *and* $\wedge$ operations.

We seek a similar result for quantum gates. The first remarkable difference is that the set of transforms on qubits is a *continuum*, as there is an uncountable number of unitary matrices, and each of them represents a different operation. Therefore, we cannot expect to represent *exactly* any unitary gate by means of a *finite* set of gates. However, we can relax the requirement by seeking a finite set of quantum gates which is capable of representing any unitary operation *within a fixed threshold* of accuracy. More explicitly, we have the following definition.

Definition 5 (Universality). A finite set $U_g = \{G_1, \ldots, G_n\}$ of quantum gates is *universal* iff for any threshold $\epsilon > 0$ and any quantum gate G there exists a quantum circuit G' consisting only of operations from U_g such that:

$$||G - \lambda G'||_2 \leq \epsilon$$

for some complex number λ, where $||\cdot||_2$ is the norm induced by Euclidean distance.

According to the above definition, it can be shown that the set composed by the single-qubit gates H and T, and the two-qubit gate C_{not} is universal. The proof of this important result, which we do not repeat here fully, is carried out in three basic steps.

1. First, it has been shown [RZBB94] that an arbitrary unitary matrix (of any size) can be represented exactly as a (product of) two-level unitary gates, that is gates which act only on two (or fewer) components. This is a basically vector-algebraic proof, where we are able to represent a generic unitary in a sort of "factored" form.

2. Second, it has been shown [DiV95] that an arbitrary two-level gate can be expressed exactly as (a product of) one-qubit operations and (a finite

number of) C-not two-qubit gates. Notice that we are now restricting the number of qubits each gate acts on, but we are still allowing for arbitrary one-qubit operations. In fact, with such restriction we are still able to represent exactly a generic unitary. This proof step is constructive, in that it shows explicitly how to build a circuit out of one- and two-qubit operations to represent a two-level unitary gate.

3. Last, it has been shown [BMP$^+$99] that an arbitrary single-qubit operation can be approximated to arbitrary accuracy by means of the Hadamard H gate and the $\pi/8$ T gate. The proof exploits clever representations of single-qubit operations as *rotations* of the quantum state in the Hilbert space.

Efficiency of universal representation. Now, it is natural to investigate the *efficiency* of the universal representation by means of gates H, T, and C_{not}, which we outlined above. The representation of circuits made of one-qubit generic operations and C-not operations solely by means of the gates in $\{H, T, C_{\text{not}}\}$ can be done efficiently. This corresponds to the "passage" from step 2 to step 3 above. More explicitly, any circuit made of m two-qubit C-not gates, and generic single-qubit gates can be approximated within any fixed $\epsilon > 0$ by means of H, T, and C_{not} gates of size $O\left(m \log^c(m/\epsilon)\right)$, where c is some (small) constant. This means that the size of the circuit made out of gates from a finite set is a poly-logarithm of the size of the circuit which is being approximated to within ϵ. This result was first proved by Solovay [Sol95] and, independently, by Kitaev [Kit97].

However, this nice result does not generalize to the other passage (i.e. that from step 1 to step 2) of the transformation from a generic unitary operator on n qubits. Explicitly, it is possible to show that there exist unitary operators that require $\Omega\left(2^n \log(1/\epsilon)/\log n\right)$ gates to be approximated to within ϵ. (See the proof by Knill [Kni95]). An important question, whose answer is still largely open, is whether there are "interesting" subsets of unitary operations which can be represented efficiently, i.e. with a "small" number of elementary gates.

Quantum parallelism. Let us now present a remarkable effect that goes under the name of *quantum parallelism.*

Let us consider a generic Boolean function $f : \{0,1\}^d \to \{0,1\}^c$ that maps d bits onto c bits. It is always possible to build a $(d + c)$-qubit quantum gate U_f that performs the mapping $|x, y\rangle \to |x, y \oplus f(x)\rangle$, where $\oplus$ denotes the addition modulo 1 (or, equivalently, bitwise exclusive-OR). Informally, U_f performs the mapping defined by f on the bits x, by changing the qubits $|x\rangle$ according to the computational base; the use of an ancillary XOR-ed set of qubits is simply a mechanism required in order to have reversibility. In fact, if we apply U_f to a $(d + c)$-qubit state $|x, 0\rangle$, the result is the state $|x, 0 \oplus f(x)\rangle = |x, f(x)\rangle$ which represents the application of f on the bits x, whose result is stored in the last c qubits of the state.

Now, let us show how to apply simultaneously the function f on all the possible 2^d inputs by means of the gate U_f. Let us start with the initialized d-qubit state $|00\cdots 0\rangle$, and apply the Hadamard transform H to it. Note that H can be applied to d-qubit systems by considering the tensor-product operator $H \otimes \cdots \otimes H$, where H is tensored d times. The effect is to put the state in a uniform superposition of all possible base states $|i\rangle$, for $i \in \{0,1\}^d$. Thus, we get the state:

$$\frac{1}{\sqrt{2^d}}\left(|00\cdots 0\rangle + |00\cdots 1\rangle + \cdots |11\cdots 1\rangle\right) = \frac{1}{\sqrt{2^d}} \sum_{i\in\{0,1\}^d} |i\rangle$$

Then, we add a c-qubit register initialized to $|0\rangle$, thus getting the state $1/\sqrt{2^d}\sum_{i\in\{0,1\}^d} |i,0\rangle$. Now, if we apply U_f to this state, we get:

$$U_f\left(\frac{1}{\sqrt{2^d}} \sum_{i\in\{0,1\}^d} |i,0\rangle\right) = \frac{1}{\sqrt{2^d}} \sum_{i\in\{0,1\}^d} U_f\,|i,0\rangle = \frac{1}{\sqrt{2^d}} \sum_{i\in\{0,1\}^d} |i,f(i)\rangle$$

This means that, for any possible input $i \in \{0,1\}^d$ of f, the quantum state has a component $|i, f(i)\rangle$ that encodes the result of the application $f(i)$ to i, and this component has an amplitude (i.e., a complex coefficient) of $1/\sqrt{2^d}$.

More explicitly, we have computed simultaneously all the possible values of the function f for its possible inputs, and encoded them in the quantum state. That is, we have manipulated an *exponential* amount of classical information in a single operation, and stored it in a quantum system of *linear* size! Naïvely, it may seem that this is the ultimate solution to all computational problems, by performing an exponential amount of parallel computation with linear resources. Of course this is not the end of the story, since we have to take into account the fact that the quantum state can be observed only according to the (strict) rule of Postulate 3. To the point, the effect of measuring the first d qubits of the above state in the computational basis would be to make the system shift into *one* of the states

$$|k, f(k)\rangle$$

for *some random* $k \in \{0,1\}^d$. Which state we measure would be totally random, as the 2^d possible outcomes have the same probability $\left(1/\sqrt{2^d}\right)^2 = 1/2^d$ (since the amplitudes are all the same).

Therefore, one of the most important goals in quantum algorithms is to find ways in which to exploit quantum parallelism, overcoming the limitations imposed by the measurement postulate of quantum mechanics. We will present two (successful) approaches to do so in Section 3.2. In a nutshell, the first of such approaches consists in exploiting an *interference* process between quantum states to *amplify* the outputs of interests. Amplification would augment the amplitudes of such states, so that the probability of getting them in a measurement is higher than that of getting the other ones. This approach is pursued in Grover's algorithm. The second approach consists instead in finding properties which are common to all values of the function being computed. This approach is pursued in Shor's algorithm.

The qubit vs. the bit. The uniqueness of the features of the quantum world can be grasped only if compared against the classical situation, with which we are familiar. To this end, the following table summarizes some of the aforementioned features of quantum computing by comparing the qubit with the bit.

	(classical) **bit**	**qubit**
State s of n (qu)bits	$s \in \{0,1\}^n$	$s = \sum_{x\in\{0,1\}^n} c_x \lvert x\rangle$, with $c_x \in \mathbb{C}$ and $\sum_x \lvert c_x\rvert = 1$
Evolution of the state	any Boolean function $f : \{0,1\}^n \to \{0,1\}^n$	any $2^n \times 2^n$ unitary matrix U
Number of transformations	finitely many	uncountably many
What you get by measuring	the state s	some *classical* state (i.e., n bits)
State after a measurement	unchanged	irreversibly changed
Nature of measurement	deterministic	stochastic
Composition of subsystems	by Cartesian product $\times$	by tensor product $\otimes$
Size of n-(qu)bits systems	n	2^n
Universality is achieved	exactly	with arbitrary precision
State of subsystems	always defined	undefined for entangled states

The quantum gate and Turing completeness. So far, we have presented the whole quantum computational model using the circuit model. However, a quantum circuit, just like a classical one, defines a function only for inputs of some fixed size, whereas we are interested in more general models that represent the computation of some function on inputs of arbitrary size. This is achieved, in the classical world, with models such as the Turing machine. The analogue for the quantum world, the *quantum Turing machine*, was first defined in [Deu85], and later perfected in [BV97].

However, we can still represent the same class of functions computed by quantum Turing machines, by only using the circuit model, but considering *families* of circuits $\{C_n\}_{n\in\mathbb{N}}$ — namely one for each input size — rather than single circuits. Moreover, the size of the circuit is polynomially correlated to the running time of the Turing machine computing on that input. The equivalence was first shown by Yao in [Yao93], and constitutes the analogue of the same result for classical Turing machines and classical Boolean circuits.

A couple of subtleties should be considered when defining a family of circuits $\{C_n\}_{n\in\mathbb{N}}$; this discussion applies to both the classical and the quantum case. First, we must require that each circuit C_n is algorithmically constructible in time polynomial in n. This means that we must be able to provide an implementable description of each circuit in the family, for all n. This is required to rule out the possibility of encoding uncomputable functions in the construction of a circuit family; whenever a circuit family fulfills this requirement we call it *uniform*. A second subtlety regards the fact that, for any given n, we can actually perform a Turing machine computation with a circuit which *has no memory*, since circuits are *acyclic*. This is counterintuitive, and proving it is a highly non-trivial task; for details, we refer the interested reader to [Pap94, Problem 2.8.10, pg. 54].

2.6 Examples of Quantum Computations

Time is ripe to put down a couple of examples that demonstrate the capabilities of quantum information processing. The first example is an application of quantum systems to information transmission, while the second is a simple quantum algorithm that outperforms any classical one for the same problem.

2.6.1 Superdense Coding

The first application we consider is *superdense coding*, which is a technique to transmit two bits of classical information by exchanging just one qubit. It was first proposed by Bennett and Wiesner [BW92].

Let us consider two parties, customarily named Alice and Bob, wishing to communicate. Beforehand, a pair of entangled particles is prepared in the state:

$$|\psi_1\rangle \quad = \quad \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right)$$

and Alice and Bob receive, respectively, the first and second particle. Notice that this preparation does not involve any information exchange yet, as the prepared state is fixed, regardless of the actual values one seeks to communicate.

Afterwards, Alice wants to send Bob two bits of information, that is an integer in the range [1..4]. According on the actual value to be sent, Alice performs one of the transformations U_1, U_2, U_3, U_4 to her qubit, where we take $U_i = \sigma_{i-1}$, the Pauli matrices. Therefore, we have one out of four possible resulting global states (where we apply the identity transformation to Bob's qubit to indicate that he does not change the state of his particle):

message	**transformation**	**new global state**
1	$\sigma_0 \otimes I$	$|\psi_1\rangle = 1/\sqrt{2}(|00\rangle + |11\rangle)$
2	$\sigma_1 \otimes I$	$|\psi_2\rangle = 1/\sqrt{2}(|10\rangle + |01\rangle)$
3	$\sigma_2 \otimes I$	$|\psi_3\rangle = i/\sqrt{2}(|10\rangle - |01\rangle)$
4	$\sigma_3 \otimes I$	$|\psi_4\rangle = 1/\sqrt{2}(|00\rangle - |11\rangle)$

After performing this transformation, Alice sends Bob her qubit: this is when the actual information exchange takes place.

Now, Bob executes the following protocol to retrieve the two bits of information Alice sent him. First, he applies a C-not gate to the two-qubit system and measures the second qubit in the computational basis. Let us summarize the four possible outcomes of this operation in the following table.

message	C-notted state	measure of second qubit
1	$\|\psi'_1\rangle = 1/\sqrt{2}(\|0\rangle + \|1\rangle) \otimes \|0\rangle$	$\|0\rangle$
2	$\|\psi'_2\rangle = 1/\sqrt{2}(\|1\rangle + \|0\rangle) \otimes \|1\rangle$	$\|1\rangle$
3	$\|\psi'_3\rangle = i/\sqrt{2}(\|1\rangle - \|0\rangle) \otimes \|1\rangle$	$\|1\rangle$
4	$\|\psi'_4\rangle = 1/\sqrt{2}(\|0\rangle - \|1\rangle) \otimes \|0\rangle$	$\|0\rangle$

Notice that the outcome of the measurement on the second qubit is certain in all four cases.

Second, Bob applies the Hadamard gate to the first qubit and measures it. The outcomes of these operations are:

message	transformed first qubit	measure of first qubit
1	$\|\psi''_1\rangle = \|0\rangle$	$\|0\rangle$
2	$\|\psi''_2\rangle = \|0\rangle$	$\|0\rangle$
3	$\|\psi''_3\rangle = -i\,\|1\rangle$	$\|1\rangle$
4	$\|\psi''_4\rangle = \|1\rangle$	$\|1\rangle$

where notice that the (phase) factor $-i$ in the third case is not measurable.

All in all, Bob has *two bits of information*, encoding unambiguously the value that Alice sent him according to the following scheme.

first bit	second bit	sent value
0	0	1
1	0	2
1	1	3
0	1	4

Notice that all Bob did was to distinguish between the four possible states for the system; this was possible unambiguously because the four possible values are encoded by orthogonal states, thus distinguishable. This clever encoding scheme has been experimentally demonstrated by Mattle et al. [MWKZ96].

2.6.2 Deutsch Algorithm

Let us now consider a quantum algorithm to solve a very simple problem, which nonetheless outperforms any classical algorithms for the same problem. The problem can be states simply as follows: given a black-box function $f : \{0,1\} \rightarrow \{0,1\}$, determine the value of $f(0) \oplus f(1)$. We will evaluate the complexity of an algorithm solving this problem by means of the number of queries to the function f that the algorithm performs.

To frame the problem, let us first consider for a moment the classical solution. Determining the value of $f(0) \oplus f(1)$ requires to know the value of *both* $f(0)$ *and* $f(1)$ independently, since knowing the value of one operand is not enough to predict with certainty the outcome of the operations. Therefore, it is simple to understand that the minimum possible number of queries required to get the right solution with certainty is two. (More formally, one could apply straightforwardly an adversary argument to prove this lower bound).

Let us now present a quantum algorithm for the same problem, first proposed by Deutsch in his seminal paper [Deu85]. Let us start by preparing a two-qubit state $|\psi\rangle = |0\rangle \otimes |1\rangle$. Then, we apply the Hadamard transform to both qubits, thus getting to the state:

$$|\psi'\rangle \;=\; (H \otimes H)\,|0\rangle \otimes |1\rangle = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = \frac{|00\rangle - |01\rangle + |10\rangle - |11\rangle}{2}$$

Now, recall that, when discussing quantum parallelism in Section 2.5, we have shown that for any Boolean function f it is possible to build a unitary gate U_f that performs the mapping $|x, y\rangle \to |x, y \oplus f(x)\rangle$. Thus, let us consider the gate U_f for the function we are evaluating, and let us apply it to our two-qubit system. Note that this application of the gate U_f counts as one "quantum query" to the function f, in its unitary form. By linearity, we get:

$$|\psi''\rangle \;=\; U_f\,|\psi'\rangle = \frac{|0, f(0)\rangle - |0, \neg f(0)\rangle + |1, f(1)\rangle - |1, \neg f(1)\rangle}{2}$$

Notice that the state $|\psi''\rangle$ can be rewritten in two forms, according to whether $f(0) \oplus f(1)$ equals 1 or 0. Namely, we have:

$$|\psi''\rangle = \begin{cases} \frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes \frac{|f(0)\rangle - |\neg f(0)\rangle}{\sqrt{2}} & \text{if } f(0) \oplus f(1) = 0 \\ \frac{|0\rangle - |1\rangle}{\sqrt{2}} \otimes \frac{|f(0)\rangle - |\neg f(0)\rangle}{\sqrt{2}} & \text{if } f(0) \oplus f(1) = 1 \end{cases}$$

Finally, let us apply one more Hadamard transform on the first qubit only, getting the state:

$$|\psi'''\rangle = H\,|\psi''\rangle = \begin{cases} |0\rangle \otimes \frac{|f(0)\rangle - |\neg f(0)\rangle}{\sqrt{2}} & \text{if } f(0) \oplus f(1) = 0 \\ |1\rangle \otimes \frac{|f(0)\rangle - |\neg f(0)\rangle}{\sqrt{2}} & \text{if } f(0) \oplus f(1) = 1 \end{cases}$$

Therefore, it is immediate to notice that we can retrieve the value of $f(0) \oplus f(1)$ by measuring the first qubit in the computational base. All in all we performed just one evaluation of U_f, against the minimum of two evaluations of f in the classical case. This was made possible by an *interference* between the quantum states, by which a global property of f has emerged into a measurable state. Finding ways to exploit interference processes to emerge interesting properties is one of the key challenges in designing quantum algorithms.

Deutsch-Jozsa algorithm. The above algorithm has been generalized by Deutsch and Jozsa in [DJ92] to a more significant problem. The problem solved by the Deutsch-Jozsa algorithm — of which Deutsch algorithm is a special case — is the following. We are given a function $f : \{0, 1\}^n \to \{0, 1\}$ of n-bits in its "unitary form" U_f. We are promised that f is either *constant*, that is it evaluates to the same value for all inputs, or *balanced*, that is it evaluates to 0 for exactly half of the inputs, and to 1 for the other half. The problem is to determine whether f is constant or balanced.

Let us present the algorithm succinctly; the following steps are also summarized in the circuit in Figure 3.

1. Start with the state $|\psi\rangle = |0^n\rangle\,|1\rangle$;

2. Apply the Hadamard transform to all $n+1$ qubits, thus getting to the superposition:

$$|\psi'\rangle = H^{\otimes(n+1)} |\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

3. Apply the U_f transform, getting to the state $|\psi''\rangle$ which can be conveniently written as:

$$|\psi''\rangle = U_f\, |\psi'\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} |x\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

4. Apply H on the first n qubits. With some work, one can understand that the resulting state is expressed as:

$$|\psi'''\rangle = (H^{\otimes n} \otimes I)\, |\psi''\rangle = \frac{1}{2^n} \sum_{y\in\{0,1\}^n} \sum_{x\in\{0,1\}^n} (-1)^{xy+f(x)} |y\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

   where xy denotes bitwise product (i.e., bitwise and).

5. Measure the first n qubits. If f is constant, then it is easy to check that the amplitude for the state $|0^n\rangle$ must be ± 1, depending on whether f takes constant value 0 or 1 (thus, all other amplitudes are 0 for unitarity). Therefore, the measurement in the computational basis yields the state $|0^n\rangle$ with certainty. Otherwise, f is balanced, and thus the amplitude for the state $|0^n\rangle$ must be 0. Therefore, the measurement in the computational basis yields to a state other than $|0^n\rangle$ with certainty. In any case, we have decided whether f is constant or balanced.

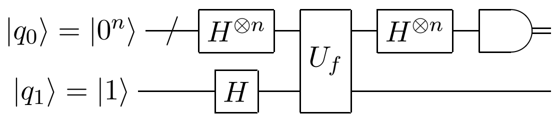

Figure 3: A circuit for the Deutsch-Jozsa algorithm.

The Deutsch-Jozsa is capable of determining the nature of the function f with one query to U_f, whereas it is rather straightforward to show that the best a classical deterministic algorithm can do is performing $2^{n-1}+1$ queries to f (that is more than half of the possible inputs). This shows an *exponential speedup* between the classical and quantum case.

Remarks. A few concluding remarks that put the Deutsch-Jozsa algorithm in the right perspective are worth doing. First, it is clear that the problem solved by the Deutsch-Jozsa algorithm is rather made up, and it is hard to see practical applications for it. Nonetheless, we will be able to demonstrate the possibilities of quantum algorithms also on "natural" problems, of which the Deutsch-Jozsa algorithm constitutes an interesting and important premise.

Second, one may deem more significant to compare the performances of the Deutsch-Jozsa algorithm to those of a *probabilistic*, rather than deterministic, classical algorithm for the same task. In this case, it is not hard to see that we can get to an answer with $\epsilon > 0$ probability of error by performing just $\log_2(2/\epsilon)$ queries to f, so the gap seems to shrink in perspective.

Third, one may feel a mismatch in comparing the number of classical queries to a Boolean function f against the number of application of a quantum gate U_f. Answering satisfactorily to these concerns is a deep task, in the direction of finding ways to evaluate quantum computational resources and comparing them against classical ones. We will see some more of this topic later.

3 Selected Issues in Quantum Computation

This section presents some interesting results in a few of the diverse subfields of quantum computation. Each of these topics is very developed and the object of intense past and present research. While this section simply illustrates some results that characterize the various subfields, we also give some pointers to references for further reading.

3.1 Quantum Information

The field of quantum information aims at building a theory of information about quantum systems, where the information is stored in quantum states, and at commensurating this new kind of information with the classical one.

For the sake of simplicity, we do not present any advanced result in this field, which is very developed and whose results often rely on advanced mathematical techniques. On the contrary, we illustrate a simple example, which is however representative of common trends in quantum information. The result can be simply stated and simply proved; nonetheless it characterizes a striking difference of quantum states with respect to classical ones, from the point of view of information. This deep difference (and lack of intuitiveness) is likely the reason why this result, in spite of its simplicity, has been explicitly discovered as late as the early 80's [WZ82].

The result we consider is usually referred to as *no cloning* theorem. It shows that it is not possible to perform a *copy* operation on arbitrary quantum states. This limitation seems to cast a shadow on the feasibility information-processing tasks which are instead common in the classical world. In particular, we are thinking about duplication of states to achieve fault tolerance. This will be discussed in Section 3.5.

Let us now prove the no cloning theorem. For the sake of contradiction, let us assume that we can clone quantum states, that is there exists some unitary transform U such that

$$U|\psi\rangle|p\rangle = |\psi\rangle|\psi\rangle$$

for all states $|\psi\rangle$ and some initialized state $|p\rangle$. Now, let us take any two quantum states $|\psi\rangle$ and $|\phi\rangle$. Since cloning works for both, we have:

$$\begin{aligned} U|\psi\rangle|p\rangle &= |\psi\rangle|\psi\rangle \\ U|\phi\rangle|p\rangle &= |\phi\rangle|\phi\rangle \end{aligned}$$

Let us take the inner product of these two equations. We get

$$\langle p,\psi|U^{\dagger}U|\phi,p\rangle = (\langle\psi|\,|\phi\rangle)^2$$

that is

$$\langle\psi|\phi\rangle = (\langle\psi|\phi\rangle)^2$$

Considering this as an equation in the variable $\langle\psi|\phi\rangle$, it only admits the solutions $\langle\psi|\phi\rangle = 0$ and $\langle\psi|\phi\rangle = 1$. This means that the either $|\psi\rangle$ and $|\phi\rangle$ are orthogonal, or they are parallel. But this contradicts the fact that we have a general cloning transformation, working for any two states. Therefore, we cannot clone arbitrary states.

3.2 Quantum Algorithms

Hilbert space is a big place!
— Carlton M. Caves [Cav].

All quantum algorithms designed so far fall into one of two categories, according to the basic techniques they use to "harness the quantum power" (and in particular quantum parallelism, hence the opening quote). Algorithms in the first category try to find global properties of a (classical) function by manipulating its *Fourier transform*; they are described in Section 3.2.1. These algorithms are named "Shor-type" since they have their first and most important representative in Shor's factoring algorithm. The algorithms in the second category try instead to perform an interfering manipulations between quantum amplitudes in order to increase those of interest, whose states can therefore be measured with high probability; these algorithms are named "Grover-type" since they are all phrased in the setting of Grover's search algorithm, and are described in Section 3.2.2.

3.2.1 Shor-type Algorithms

All of Shor-type algorithms are based on performing the Fourier transform using quantum gates to determine the global properties of some function applied through quantum parallelism. In this section, we first illustrate briefly what is the quantum Fourier transform, and then detail the steps of Shor's algorithm for factoring integers, which is the most known, and successful, application of these quantum Fourier transform techniques. We do not discuss other applications of the same techniques, and generalizations, such as the algorithms for the discrete logarithm [Sho97], Simon's problem [Sim94, Sim97], the Abelian stabilizer [Kit96], and the general framework of the hidden subgroup problem [ME98].

The quantum Fourier transform. The discrete Fourier transform (DFT) is a discrete analogue of the (continuous) Fourier transform. It takes as input n complex values x_k, and outputs another n complex values X_j given by:[2]

$$X_j \;=\; \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} x_k e^{\frac{2\pi i}{n} jk} \qquad \text{for } j = 0, \ldots, n-1$$

The fast Fourier transform (FFT) is a fast algorithm to perform the DFT when n is a power of 2. The quantum Fourier transform (QFT) performs the same task as the FFT using quantum resources. Namely, for $n = 2^m$ for some m, the QFT operates on the amplitudes of the quantum state, by transforming the states $|j\rangle$, for $j = 0, \ldots, n-1$ to:

$$U_{\mathcal{F}} \, |j\rangle \;=\; \frac{1}{\sqrt{2^m}} \sum_{k=0}^{2^m-1} e^{\frac{2\pi i}{2^m} jk} \, |k\rangle$$

Shor showed [Sho97] explicitly that the above unitary gate $U_{\mathcal{F}}$ can be build efficiently, that is with $O\left(m^2\right)$ standard gates.

Let us now consider a function $g : [0..n-1] \to [0..n-1]$, periodic (modulo n) with respect to some period c, and let $G : [0..n-1] \to [0..n-1]$ be a function such that the values $G(j)$ represent the DFT of the values $g(k)$. If the period c is a factor of n, then the G_j's are non-zero only for those j's which are multiples of n/c. Otherwise, the non-zero values will still concentrate near these multiples, but only approximately. In the quantum case, this means that we will have the largest amplitudes about these multiples, so that these will be the most likely outcomes of a measurement.

Shor-type algorithms try to exploit these properties of the QFT in order to compute efficiently interesting properties of some functions g. In particular, Shor's algorithm efficiently computes the factorization of an integer, and we illustrate it next.

Shor's algorithm. The best known classical algorithm for factoring an n-bit integer is the number field sieve [LL94] that takes time $O\left(\exp(cn^{1/3} \log^{2/3} n)\right)$. In general, the factoring problem is considered to be computationally intractable, so much that the most widely used protocols for public key cryptography rely on its hardness to guarantee security [Pap94, Sch95]. Therefore, Shor's algorithm [Sho94, Sho97] — capable of factoring an n-bit integer in time $O\left(n^2 \log n \log\log n\right)$ — is a great achievement, and constitutes an important breakthrough.

Let us describe Shor's algorithm by detailing its steps. Let M an integer to be factored; let $n = \lceil \log_2 M \rceil$.

[2]The factor $1/\sqrt{n}$ is a normalization factor to have a unitary transform.

1. Pick a random integer p. If p is a factor of M we are done, otherwise choose an m such that $M^2 \leq 2^m \leq 2M^2$. Prepare $m+n$ qubits in the state $|\psi\rangle = |0^{m+n}\rangle$ and apply the Hadamard transform on the first m qubits, thus getting to the superposition:

$$|\psi'\rangle = \left(H^{\otimes m} \otimes I^{\otimes n}\right)|\psi\rangle = \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x, 0^n\rangle$$

2. Now, consider the function $f : [0..2^m-1] \to [0..M-1]$ defined as $f(x) = p^x \bmod M$.[3] Apply the corresponding quantum gate U_f to the state $|\psi'\rangle$. We get:

$$|\psi''\rangle = U_f\,|\psi'\rangle = \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x, p^x \bmod M\rangle$$

3. Measure the last n qubits in the computational basis, obtaining some random state $|u\rangle$. Regardless of the actual measured value $u \in [0..M-1]$, we have the state:

$$|\psi'''\rangle = \frac{1}{S} \sum_{x=0}^{2^m-1} g_x\,|x, u\rangle$$

where S is some proper scale factor we need not worry about, and the coefficients g_x are such that:

$$g_x = \begin{cases} 1 & \text{if } p^x \bmod M = u \\ 0 & \text{otherwise} \end{cases}$$

That is, the only state values x which "survived" are those differing by (multiples of) the period c of the function f.

4. Discard the last n qubits, and only consider the first m qubits. Apply the quantum Fourier transform to the state $|\psi'''\rangle$, obtaining some state $|\psi''''\rangle$. As we discussed in the previous paragraph, if the period c of the coefficients g_x is a factor of 2^m, that is it is a power of two, then the state $|\psi''''\rangle$ is exactly:

$$|\psi''''\rangle = \sum_k x_k\,|k\frac{2^m}{c}\rangle$$

where the actual value of the coefficients x_k is not important, as long as it is non-zero for some integer values k. If the period c is not a factor of 2^m, then $|\psi''''\rangle$ still approximates the exact case to a degree which is sufficient for the remainder of the algorithm to be exact with sufficient precision. For simplicity of exposition, from now on we will only consider the case in which c is a power of 2.

[3]Notice that $f(x)$ is an n-bit value.

5. Measure the qubits in the computational base, obtaining some state $|v\rangle$. It is easy to understand that v must be $v = k2^m/c$ for some integer k. If k and c are relatively prime, then consider the fraction $v/2^m$ and reduce it to its lowest terms. Clearly, the denominator of the reduced fraction is the period c, since $v/2^m = k/c$. Otherwise, i.e. if k and c are not relatively prime, we have found a factor of the period: in such cases, just repeat the whole process.

6. Now, it is likely that the quantity

$$p^{c/2} \pm 1$$

has a non-trivial common factor with M. This can be checked by computing the greatest common divisor (GCD) between M and $p^{c/2} \pm 1$; if the result is neither 1 nor M, we have found a factor of M. Otherwise, repeat the whole process.

Let us explain why the above GCD computation is valid. Since c is the period of $f(x) = p^x \mod M$, then $p^{x+c} = p^x \mod M$, and thus $p^c = 1 \mod M$. Now, since we are assuming that c is even (being a power of 2), we can write:

$$p^c - 1 = 0 \mod M = (p^{c/2} + 1)(p^{c/2} - 1) \mod M$$

Therefore, if $p^{c/2} \pm 1$ are not multiples of M, they have a common factor with it.

Notice that the above algorithm have several steps which may fail, thus forcing one to repeat the whole process. Shor showed [Sho97] that it succeeds with high probability, so that a limited number of repetitions almost inevitably yields a valid outcome. Figure 4 summarizes Shor's algorithm.

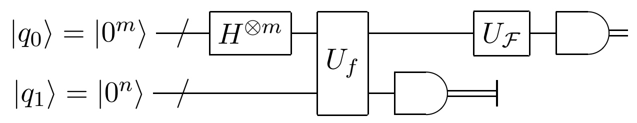

Figure 4: A circuit for Shor's factoring algorithm.

3.2.2 Grover-type Algorithms

All of Grover-type algorithms solve problems which are phrased as *search problems*. Indeed, the search setting is a broad one, in that many different problems can be formulated as search problems. Very generally, we formulate a search problem as following: given a Boolean predicate $P : \{0,1\}^n \to \{0,1\}$, find a value $x \in \{0,1\}^n$ such that $P(x)$ is true. In particular, it is simple to understand that all NP-complete problems can be formulated as search problem. For instance, the satisfiability (SAT) problem is a search problem where we look for a satisfying assignment among the set of all possible truth assignments.

Let us now consider search problems in their purest form, that is as *unstructured* search. This means that we have no extra information to find the values x making the predicate P true. Thus, the only possible search strategy is based on *querying* a black-box that can evaluate P for any given x. It is simple to understand that in such setting, a classical algorithm may have to try all possible assignments. Therefore, the worst-case complexity for the unstructured search problem for classical algorithms is $O\left(2^n\right)$, where 2^n is the number of possible assignment values, Surprisingly, using a quantum algorithm we can perform the same task with a complexity of $O\left(\sqrt{2^n}\right)$, that is with a *polynomial* speedup. Grover's algorithm performs this task and is described in the following paragraph.

Grover's search algorithm. Grover's algorithm [Gro96, Gro97] is fairly simple, at least comparatively to Shor's algorithm. Let us describe its steps in detail. Let $P : \{0,1\}^n \to \{0,1\}$ be the predicate we search for satisfying assignments.

1. Let us start with $n+1$ qubits, prepared in the state $|\psi\rangle = |0^n\rangle \otimes 1/\sqrt{2}\,(|0\rangle - |1\rangle)$.
2. Let us apply the Hadamard transform to the first n qubits, getting to the (usual) superposition:
$$|\psi'\rangle \;=\; (H^{\otimes n} \otimes I)\,|\psi\rangle = \frac{1}{\sqrt{2^{n+1}}}\sum_{x=0}^{2^n-1} |x\rangle \otimes (|0\rangle - |1\rangle)$$
3. Let U_{P} be the unitary gate performing the mapping $|x,y\rangle \to |x, y \oplus P(x)\rangle$, and let us apply it to the state $|\psi'\rangle$. Let X_0 be the set of values $x \in \{0,1\}^n$ such that $P(x) = 0$ and let X_1 be the complement set $\{0,1\}^n \setminus X_0$. As it is relatively straightforward to check, we can write the resulting state as:
$$\begin{aligned}
|\psi''\rangle \;&=\; U_{\mathrm{P}}\,|\psi'\rangle \\
&= \frac{1}{\sqrt{2^{n+1}}}\,U_{\mathrm{P}}\left(\sum_{x\in X_0}|x\rangle\,|0\rangle + \sum_{x\in X_1}|x\rangle\,|0\rangle - \sum_{x\in X_0}|x\rangle\,|1\rangle - \sum_{x\in X_1}|x\rangle\,|1\rangle\right) \\
&= \frac{1}{\sqrt{2^{n+1}}}\left(\sum_{x\in X_0}|x\rangle\,|0\oplus 0\rangle + \sum_{x\in X_1}|x\rangle\,|0\oplus 1\rangle - \sum_{x\in X_0}|x\rangle\,|1\oplus 0\rangle - \sum_{x\in X_1}|x\rangle\,|1\oplus 1\rangle\right) \\
&= \frac{1}{\sqrt{2^{n+1}}}\left(\sum_{x\in X_0}|x\rangle - \sum_{x\in X_1}|x\rangle\right) \otimes (|0\rangle - |1\rangle) \\
&= \left(\left(\sum_{x\in X_0}\frac{1}{\sqrt{2^n}}\,|x\rangle\right) + \left(\sum_{x\in X_1}\frac{-1}{\sqrt{2^n}}\,|x\rangle\right)\right) \otimes \left(\frac{1}{\sqrt{2}}\,(|0\rangle - |1\rangle)\right)
\end{aligned}$$

All in all, after this step we end up in a state $|\psi''\rangle$ where all and only basis states for which $P(x) = 1$ have an amplitude with *inverted sign* with respect to $|\psi'\rangle$. More precisely, if we ignore the last qubit, we have a state where all the states $|x\rangle$ such that $P(x) = 0$ have a positive amplitude of $1/\sqrt{2^n}$, whereas all the states $|y\rangle$ such that $P(y) = 1$ have a negative amplitude of $-1/\sqrt{2^n}$. This is visualized in Figure 5.

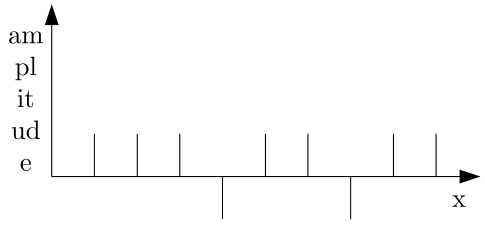

Figure 5: Change of sign in Grover's algorithm.

4. Let us now apply the gate U_{inv} to the first n qubits of $|\psi''\rangle$, where U_{inv} is defined as follows:

$$U_{\text{inv}} = \begin{pmatrix} 2/2^n - 1 & 2/2^n & \cdots & 2/2^n \\ 2/2^n & 2/2^n - 1 & \cdots & 2/2^n \\ \cdots & \cdots & \cdots & \cdots \\ 2/2^n & 2/2^n & \cdots & 2/2^n - 1 \end{pmatrix}$$

It is relatively simple to check that U_{inv} is unitary, and that it can be expressed with $O(n)$ elementary gates. The action on the state that U_{inv} performs can be described as follows. For a generic state $|\phi\rangle = \sum_{k=0}^{n-1} c_k |k\rangle$, let $A = (1/n)\sum_{k=0}^{n-1} c_k$ be the average of the amplitudes c_k's. Then, U_{inv} performs an *inversion about the average*, that is substitutes each amplitude c_k with the value $2A - c_k$. Therefore, we get to the state (ignoring the last qubit)

$$|\psi'''\rangle \quad = \quad \left(\sum_{x\in X_0}\left(2A - \frac{1}{\sqrt{2^n}}\right)|x\rangle\right) + \left(\sum_{x\in X_1}\left(2A + \frac{1}{\sqrt{2^n}}\right)|x\rangle\right)$$

where $A = (1/\sqrt{2^{3n}})\,(|X_0| - |X_1|)$, in this case. The result of this step is the amplification of the amplitudes for which P is true, with respect to the other amplitudes. This is visualized in Figure 6.

5. Repeat steps 3 and 4 for $(\pi/4)\sqrt{2^n}$ times.

6. After successive amplifications, measure the last qubit. If it is 1, then reading the other n qubits yield an x such that $P(x) = 1$; this occurs with high probability, because of the amplification process. If you get a 0, repeat the whole process.

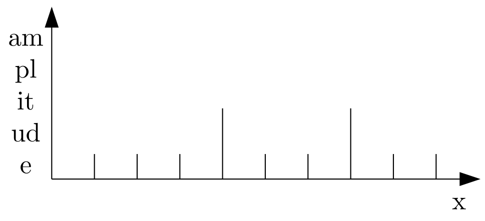

Figure 6: Inversion about average in Grover's algorithm.

The kernel of Grover's algorithm is indeed simple. The main catch lies in the number of repetitions of the steps changing the sign of the "good" amplitudes, and the inversion about the average. Indeed, if we repeat them too few times, or too many times, we get to a state where the amplification of the desired states is not large enough to yield the results we are looking for when measuring. In fact, unitary transforms are rotations in the complex space, and therefore repeating them a certain number of times brings the state back to its original value.

This is a general feature of all Grover-type algorithms. Extensive research has determined how to guess reliably the optimal number of repetitions in various contexts [BHT98]. Grover also extended his techniques to problems other than search, such as computing the mean and median of a function [Gro98].

Structured search. So far, we have only considered unstructured search problems, where no information about the search space is available. Thus all of Grover-type algorithms do is performing a blind search by exploiting "quantum guessing" to reduce the number of queries to be performed.

However, whenever we consider a problem where some information about the structure of the search space is available, it is still possible to exploit techniques like that of Grover's algorithm, but combining them with optimization based on the structure of the specific problem. Among others, Hogg has developed [Hog98] some heuristic techniques that search in structured search spaces, coupling them with Grover-type techniques. The most difficult part in developing these kinds of algorithms lies in evaluating precisely the effectiveness (in terms of computational complexity) of the heuristic techniques. Actually, this happens for classical heuristic algorithms as well. However, while classical algorithms can be simulated to assess their performances (at least experimentally), the same cannot be done with quantum algorithms, since the simulation of quantum systems on classical computers is exponentially expensive. Therefore, most complexity evaluations of these heuristic Grover-type algorithms are partial, limited to some (simple) cases only.

Beyond quadratic speedup? Although the polynomial speedup granted by Grover-type algorithms is practically very appealing, one may speculate about possible modifications of the algorithms that, still in the same vein, would yield an exponential advantage over classical computation. In particular, achieving an exponential speedup in solving generic unstructured search problems would imply the possibility of solving NP-complete problems in polynomial time. However, this exponential speedup is not possible by applying the "quantum guessing" techniques of Grover's algorithms. In fact, it has been shown in [BBBV97, BBHT98, Zal99] that Grover's algorithm is asymptotically optimal for the quantum model of computation. We briefly discuss other (speculative) ways to achieve quantum speedups in the following section.

3.2.3 Other Quantum Algorithms?

Basically of all of quantum algorithms offering better performances (than those of the classical algorithms for the same tasks) can be classified as either Shor-type or Grover-type algorithms. Whether other different techniques for harnessing the quantum features can be developed is an open research question. The task seems particularly difficult as it requires to understand how the peculiarities of quantum mechanics (which are unnatural if compared to the more familiar classical world, which we experience every day) can be exploited while avoiding the inherent limitations given by unitarity and the measurement postulate.

Some research has also been pursued about the highly speculative problem of determining how possible physical theories, derived form quantum mechanics but different than it in some respects, may yield computational models with a larger algorithmic power than quantum computing. We refer the interested reader to the survey by Aaronson [Aar05].

Here, we only hint to the the result by Abrams and Lloyd [AL98], who have shown that any modification to quantum mechanics introducing a non linearity in the dynamics of systems (that is, in other words, where operators need not be unitary), however small, would produce a physical theory where computations of NP-complete problems (as well as $\sharp$P problems) would be feasible in polynomial time. In a nutshell, these techniques would still use a modification of Grover's algorithm, where the amplification of amplitudes would however have exponentially faster effects due to the nonlinearities of the transforms. Notice however that, given the present status of knowledge of the physical world, a non-linear quantum theory is considered highly implausible. In fact, the nonlinearity would imply the possibility of, among other things, *superluminal signaling* [Gis90], that is the possibility of sending faster-than-light information. This in turn would break causality links in the description of physical phenomena. Further discussion of these very speculative issues is out of the scope of the present paper.

3.3 Quantum Cryptography

Quantum cryptography explores what cryptographic tasks can be better performed using quantum systems rather than just classical ones.

We have seen in Section 3.2.1 that it is possible to factor a large integer in polynomial time using Shor's algorithm. Therefore, most used public key encryption schemes (such as RSA [RSA78]) would be broken by a quantum computer. On the other hand, quantum computing is capable of enhancing the security of *private key* encryption schemes, by providing a way to perform *key distribution* between two parties with perfect security.

In the remainder, we describe the famous protocol developed by Bennet and Brassard in the mid 1980's [BB84]. Several other quantum key exchange protocols are variants and refinement of it.

Let us consider the encryption of a message through a key which is entirely made of random bits and is as long as the message itself. This technique is called *one-time pad* or Vernam cipher [Sch95] and is the only encryption scheme that guarantees perfect security: this means that the encrypted message is totally unintelligible without the key. The problem with this scheme is, of course, that, being the key as long as the message, sending it to the receiver becomes the real problem, as complex as the encryption of the original message.

The protocol we now illustrate solves this problem, by providing a mechanism by which:

- the two communicating parties can agree on a string of n random bits:
- if an eavesdropper intercepts some of the exchanged bit, this fact can be detected; therefore in this case the protocol is aborted, without compromising the security of the message

The protocol requires the two communicating parties, named Alice and Bob, to share a classical communication channel, as well as a quantum one, the latter being a channel where quantum particles can be sent and received. We assume that both channels are unprotected, so can be read by an intruder (called Eve).

Let us first describe the protocol, then we will demonstrate its security against eavesdropping.

1. Alice wants to share n random bits with Bob. To this end, she prepares $4n$ random bits, and encodes each of them in a quantum state as follows. For each bit b, she picks randomly one of the following two encoding schemes, and prepares the state accordingly.

bit	**encoding 1**	**encoding 2**
0	$\|0\rangle$	$\|\nearrow\rangle = (\|0\rangle + \|1\rangle)/\sqrt{2}$
1	$\|1\rangle$	$\|\searrow\rangle = (\|0\rangle - \|1\rangle)/\sqrt{2}$

   Afterwards, she sends the $4n$ prepared qubits to Bob.

2. For each received qubit, Bob picks a random encoding of the two Alice used, and measures that qubit accordingly. On the average, Bob will pick the same encoding as Alice half of the time. Therefore, about $2n$ bits will be correctly retrieved (exactly when the right measurement was performed).

3. Alice sends over the classical channel the sequence of $4n$ encoding choices.

4. Bob compares these encodings against the ones he picked, thus knowing which $2n$ bits of the total $4n$ bits are correctly agreed upon between him and Alice. He communicates, over the classical channel, this information so that now Alice and Bob shares exactly the same $2n$ bits B.

5. Alice sends n of the bits in B to Bob for a check.

6. If Bob agrees on them (except for some small fraction due to possible communication errors), the protocol succeeds and the non-exchanged remaining n bits are the string which can be used for one-time pad. Otherwise, it means that something went wrong and the protocol is aborted (since security cannot be guaranteed).

Let us now see what happens if Eve intercepts some of the messages. We assume that all the classical messages are exchanged *in public*, therefore Eve can read them but cannot change them without everybody noticing. Instead, if Eve intercepts a large fraction of the exchanged qubits, all she can do is measure them according to a random encoding, and then send them to Bob. Eve will pick a wrong encoding approximately half of the time, and therefore she will resend a wrongly encoded qubit half of the time to Bob. Accordingly, Bob will now measure the wrong bit approximately a quarter of the times he picks the right basis (with respect to Alice). In the final step of the protocol, Bob will notice that approximately one quarter of the n sent bits are indeed different between Alice and Bob, thus showing that some eavesdropping has taken place. This causes the protocol to abort, and the information retrieved by Eve becomes useless.

3.4 Quantum Finite Automata

So far, we have always considered models of quantum computation which are generalizations of the full classical Turing machine, and thus achieve universal computation. Nonetheless, the classical theory of computation and formal languages has benefited from extensively studying model of computations which are less powerful than the Turing machine, but are nonetheless of interest in defining some classes of languages, which can give the way to interesting practical developments. Just to mention a couple of them, consider the application of the theory of regular languages to string matching algorithms [CLRS01, Chap. 32], and the theory of context-free languages to the construction of parsers and automated translators [App97, ASU86].

In the same vein, it may be useful to study extensions of those restricted models of computations to the quantum case. This might help in two respects, at least. First, studying "simpler" models of quantum computation may help in developing some intuition on how the quantum models may be exploited to perform computation. This intuition is hard to gain, because we are used to classical models. Developing it may help in designing quantum algorithms for new classes of problems [Sho03]. Second, techniques and solutions based on simple quantum models of computations are likely to be physically realized much earlier than full-scale quantum computation [AW02]. Therefore, studying these models of computation may help in developing the first quantum devices to be actually built and used.

Under these respects, this section presents a number of different generalizations to the quantum world of the basic finite state automata. We will briefly present formally the models, outline some of their characteristics, explore some of the relations among them, and compare them against classical models.

3.4.1 Stochastic Functions and Languages

Let us now set the framework in which to present quantum finite state automata.

Stochastic functions on words. Let Σ be an alphabet, that is a finite set of symbols. As usual, Σ^* denotes the free monoid on Σ, that is the set of all (finite-length) strings over the alphabet Σ.

We associate a *probability distribution* over words in Σ^* by specifying a function $\phi : \Sigma^* \to [0,1]$ that associates a real number between 0 and 1 to every word in Σ^*; thus $\phi(w)$ denotes the probability associated to the word $w \in \Sigma^*$. We may have more general definitions, but this will suffice for the present needs.

Bounded probability and languages. We consider (finite) automata as *language acceptors*. That is, they are devices which, given in input a string $w \in \Sigma^*$, accept or reject it after performing some computation. Therefore, we describe the behavior of some automaton A by giving a stochastic function ϕ_A which accepts each word $w \in \Sigma^*$ with probability given by $\phi_A(w)$.

Moreover, given a constant $\lambda \in [0,1]$, we say that A *accepts* a language $L \subseteq \Sigma^*$ *with cut-point* λ, iff:

- for all $w \in L$ the probability of accepting w is greater than λ, that is $\phi_A(w) > \lambda$;
- for all $w \notin L$ the probability of accepting w is at most λ, that is $\phi_A(w) \leq \lambda$.

This provides an acceptable definition of the accepted language L, in that all words in L are accepted with probabilities higher than any word not in L.

In general, however, we seek a stronger definition of language. Thus, we say that a cut-point is *isolated* if $|\phi(w) - \lambda| > \epsilon$ for some real $\epsilon > 0$, for all words $w \in \Sigma^*$. A language accepted with some isolated cut-point is said to be accepted with *bounded probability*.

In fact, whenever a language is accepted with some isolated cut-point, the probabilities of acceptance for string in and out of the language are well separated, and do not "accumulate" about the cut-point. Informally, this implies that we can ascertain the language accepted by the automaton by a "majority vote" over subsequent runs over the same strings. If a string is accepted over all runs, then with high probability it belongs to L, as the probability of always rejecting it decreases exponentially (with the number of runs). It is also not difficult to see that the actual values of λ and ϵ are not important, as long as they are both non-zero.

Using bras. In the remainder, we will represent quantum states using bra vectors, rather than their duals ket vectors. Thus, for instance every automaton will have an initial state represented as $\langle s_0|$. Consequently, operators will be applied to the right of states.

We perform this shift of notation to aid the intuition in representing the automata computations. In fact, in general we will have a unitary operator U_σ associated to every letter $\sigma \in \Sigma$. Then, when performing the computation corresponding to some word $w = \sigma_1\sigma_2 \cdots \sigma_{|w|}$, starting from the initial state $\langle s_0|$, we will apply the operators $U_{\sigma_1}, U_{\sigma_1}, \ldots, U_{\sigma_{|w|}}$ in the same order as the characters in the string w. Thus, by applying them to the right of the start state $\langle s_0|$ we have the advantage of having them in the same order as the letters of the word, rather than reversed.[4]

3.4.2 Quantum Finite Automata Models

In general, a quantum finite automaton is an automaton which can be, at any time, in a state which is a superposition of a *finite* set of base states. In other words, it is an automaton whose state space is a finite-dimensional Hilbert space (or subset thereof).

Quite a large number of finite state quantum automata have been introduced in the literature. Here, we present three variants of the simplest *1-way* automata, that is those were the reading head always moves to the right; these are the most natural extensions of the corresponding classical finite state automata. These variants are:

Measure Once (MO) [MC00, BC01, BP02], where the state is measured just once, after reading the whole input string;

Measure Many (MM) [KW97, AF98, BP02], where successive measurements are performed on the state while reading the input string;

With Control Language (CL) [BMP03b], where the acceptance condition is controlled by specifying an auxiliary control language.

In the remainder of this section, we are going to define them precisely. For simplicity, we never mention the dimension of the state, but we always take it to be finite. Other 1-way quantum automata may be defined, such as reversible quantum automata [GK02, BMP03b], which are similar to MM automata, but with intermediate measurements restricted as to be reversible. For the sake of brevity, we do not consider them in this paper.

Measure once quantum automata (MO). A *measure once quantum automaton* (MO, for short) is defined by a tuple $\langle \langle s_0| , \{U_\sigma\}_{\sigma\in\Sigma}, M\rangle$, where $\langle s_0|$ is the initial state, the U_σ's are unitary transformations, one for each letter in the alphabet Σ, and M is an observable decomposable into the two projectors P_+ and P_- (for acceptance and rejection, respectively).

[4]On this idiosyncrasy in the notation of quantum mechanics Feynman made a jokingly politically-incorrect comment in [Fey85a].

For any word $w = \sigma_1\sigma_2\cdots\sigma_{|w|}$, the behavior of the automaton is described by the stochastic function $\phi : \Sigma^* \to [0,1]$ defined as:

$$\phi(w) \;=\; \left|\left|\langle s_0|\, U_{\sigma_1} U_{\sigma_2} \cdots U_{\sigma_{|w|}} P_+ \right|\right|^2$$

That is, a transformation U_σ is applied to the initial state for every letter in the word w. Finally, the probability of getting a measurement in the acceptance subspace P_+ defines the acceptance probability for w.

We denote by BMO the class of languages accepted by measure once quantum automata with bounded probability.

Measure many quantum automata (MM). A *measure many quantum automaton* (MM, for short) is defined by a tuple $\langle \langle s_0|, \{U_\sigma\}_{\sigma\in\Sigma}, M\rangle$, where $\langle s_0|$ is the initial state, the U_σ's are unitary transformations, one for each letter in the alphabet Σ, and M is an observable decomposable into the three projectors P_+, P_-, and P_0 (for acceptance, rejection, and "continue", respectively).

For any word $w = \sigma_1\sigma_2\cdots\sigma_{|w|}$, the behavior of the automaton is described by the stochastic function $\phi : \Sigma^* \to [0,1]$ defined as:

$$\phi(w) \;=\; \sum_{k=1}^{|w|} \left|\left|\langle s_0|\, U_{\sigma_1} P_0\, U_{\sigma_2} P_0\; \cdots\; U_{\sigma_{k-1}} P_0\, U_{\sigma_k} P_+ \right|\right|^2$$

That is, starting from the initial state, for each letter σ, a transformation U_σ is applied, and a measurement is immediately performed. The process continues only as far as measurement results fall in the subspace P_0. As soon as the measurement yields the result for the subspace P_+, the string is accepted.

We denote by BMM the class of languages accepted by measure many quantum automata with bounded probability.

Quantum automata with control language(CL). A *quantum automaton with control language* (CL, for short) is defined by a tuple $\langle \langle s_0|, \{U_\sigma\}_{\sigma\in\Sigma}, M, L\rangle$, where $\langle s_0|$ is the initial state, the U_σ's are unitary transformation, one for each letter in the alphabet Σ, M is an observable decomposable into s projectors $P_1, \ldots, P_s$ with corresponding measurable eigenvalues $\{v_1, \ldots, v_s\} = V$, and $L \subseteq v^*$ is a regular language over the alphabet V (called the "control language").

For any input word $w = \sigma_1\sigma_2\cdots\sigma_{|w|}$, and control word $c = c_1c_2\cdots c_{|w|}$, the probability of accepting the input word for that control word is given by:

$$p(c;w) \;=\; \left|\left|\langle s_0|\, U_{\sigma_1} P_{c_1}\, U_{\sigma_2} P_{c_2} \cdots U_{\sigma_{|w|}} P_{c_{|w|}} \right|\right|^2$$

That is, starting from the initial state, a transformation U_σ is applied, and a measurement is performed for every letter in w. The process continues only as far as the outcome of the measurement matches the corresponding letter in the control word.

Then, for any input word $w = \sigma_1\sigma_2 \cdots \sigma_{|w|}$, the behavior of the automaton is described by the stochastic function $\phi : \Sigma^* \to [0, 1]$ defined as:

$$\phi(w) \;=\; \sum_{\substack{c \in L \\ |c|=|w|}} p(c; w)$$

That is, a word is accepted iff it is accepted for any control word of conformant size.

We denote by $\mathsf{BCL}(V, L)$ the class of languages accepted with bounded probability by quantum automata with control language L over the alphabet V , and by BCL the class of languages accepted by quantum automata with control language, for any choice of control language and control alphabet.

3.4.3 Properties of the Models

Let us now present some closure properties of the classes of languages accepted by the above automata, as well as some inclusion relations among those classes.

Group languages. A clear-cut characterization of the language class BMO can be done in terms of *group languages*. Group languages are a subclass of regular languages, which we will denote as GRL. More precisely, a regular language accepted by some deterministic finite automaton $D = \langle Q, s_0 \in Q, \delta : Q \times \Sigma \to Q, F \subseteq Q\rangle$, where Q is the set of states, s_0 is the initial state, δ is the (deterministic) transition function, and F are the accepting states, is a *group language* iff for every state $s \in Q$ and every input symbol $\sigma \in \Sigma$, there exists exactly one state $s' \in Q$ such that $\delta(s', \sigma) = s$, that is δ is a complete one-to-one function. In other words, a group language is accepted by a deterministic finite automaton where, for every $\sigma \in \Sigma$, $\delta(\cdot, \sigma) : Q \to Q$ is a permutation of the elements in Q.

Another, equivalent, way of expressing this requirement is obtained by referring to the matrix representation of the transition relation δ of the deterministic finite automaton; thus, if we consider the set $T = \{T_\sigma\}_{\sigma\in\Sigma}$, the accepted language is a group language iff the elements in T generate a group with respect to ordinary matrix multiplication.

Remind that T generating a group means that there exists a set, formed by arbitrary multiplications of the matrices in T, where multiplication is an internal, associative operation, there exists neutral element, and every element has inverse. Note that all the properties but the existence of an inverse are satisfied by any generating set of matrices.

BMO equals GRL. Let us now sketch the proof that the class of languages BMO equals the class of group languages GRL. Note that this implies that MO automata are less powerful than classical deterministic finite automata. We will follow the proofs given in [MC00].

Proving that $\mathsf{GRL} \subseteq \mathsf{BMO}$ is fairly easy, as the transition matrices of any DFA accepting a group language are also valid (unitary) transition matrices of a MO automaton that accepts the same language with certainty (i.e., all probabilities are either 0 or 1).

Let us now show why $\mathsf{BMO} \subseteq \mathsf{GRL}$. First of all, it is possible to show that the class BMO is a proper subset of the regular languages; we omit the proof of this fact. Then, let $L \in \mathsf{BMO}$ be any language accepted by a MO automaton; we now show that any sequence of transition matrices $U = U_{\sigma_1} \cdots U_{\sigma_h}$ has an inverse. As we hinted to above, this is enough to prove that the transition matrices generate a group, and therefore that $L \in \mathsf{GRL}$.

Let $\chi_L : \Sigma^* \to \{0, 1\}$ be the characteristic functions for the language L, defined as $\chi_L(w) = 1$ iff $w \in L$. Then, a pumping lemma for languages in BMO holds (see [MC00]), such that for every word $w \in \Sigma^*$, there exists an integer k such that, for all word u, v, we have $\chi_L(uw^k v) = \chi_L(uv)$. Therefore, the words uw^k and u are equivalent, for all u, since yield to equivalent states whenever followed by the same suffixes (v in this case). Thus, we have $U^k = I$ for some k, since w can be any word, and it returns *any* word u to its original equivalence class. This means that U has inverse U^{k-1}, and thus the transition matrices form a group.

An immediate consequence of the class of languages BMO coinciding with the group languages is that it inherits all the closure properties of the class GRL. In particular, BMO is thus closed with respect to all Boolean operators.

Although the class BMO coincides with a small subset of the regular languages, MO automata do have some advantages over the language equivalent DFAs. In particular, they are often more *succinct* than their classical counterparts. See for instance the results in [MPP01, MP02, BMP03a].

BCL is closed under Boolean operations. As shown by Bertoni et al. in [BMP03b], the class BCL is also closed under all Boolean operations. This can be shown by first proving that the class of the formal power series associated with the stochastic functions defining the CL automata is closed under the operations of f-complement, Hadamard product, and convex linear combination. Then, these properties of the formal series translate into properties of the corresponding languages accepted with bound probability. Namely, f-complement translates to closure under union, Hadamard product translates to closure under intersection, and convex linear combination translates to closure under complement.

Let us just give some definitions. Given a stochastic function $\phi : \Sigma^* \to [0, 1]$, we associate to it the formal power series $\phi^S = \sum_{w \in \Sigma^*} \phi(w)w$. Then, given two formal power series ϕ^S and ψ^S, Hadamard product is defined as the series $\phi^S \odot \psi^S = \sum_{w \in \Sigma^*} \phi(w)\psi(w)w$. Their linear combination with coefficients $a, b \in \mathbb{R}$ is defined as the series $a\phi^S + b\psi^S = \sum_{w \in \Sigma^*} (a\phi(w) + b\psi(w))\, w$. Finally, the f-complement of ϕ^S is defined as the series $1 - \phi^S = \sum_{w \in \Sigma^*} (1 - \phi(w))\, w$.

Closure properties of BMM. Contrarily to the other two language classes, BMM is not closed under all Boolean operations. More precisely, it has been shown [AKV01] that it is closed under complement, but not under union (and therefore not even under intersection, by De Morgan's laws). Although giving an exact characterization of the class BMM has shown to be a very hard task, it is possible to identify some necessary conditions that any language in BMM must satisfy. More precisely, some regular language cannot belong to BMM if the minimal deterministic automaton accepting it contains some characteristics subgraphs in the representation of the transition function. So, for instance, it is possible to show that the two languages $L_1 = (00)^*1(00)^*0$ and $L_2 = 0(00)^*1(00)^*0$ are both accepted by MM automata, but their union $L_1 \cup L_2 = 0^*1(00)^*0$ is accepted by a minimal DFA that contains a "forbidden" subgraph. Therefore $L_1 \cup L_2 \notin \mathsf{BMM}$, which is then not closed under union.

Inclusions among the models. The known relations among the models are summarized in [BMP03b, Theorem 10]. They are:

- $\mathsf{BMO} \subset \mathsf{BCL}(\{a, r\}, a^*)$, that is a CL automaton can simulate a MO automaton by suitable choice of the intermediate observables. Moreover, some trivial languages are accepted by CL automata but not by MO automata (thus showing that the inclusion is strict).
- $\mathsf{BCL}(\{a, r\}, a^*) \subseteq \mathsf{BCL}(\{a, r, g\}, g^*a\{a, r, g\}^*)$, by introducing a sort of generalizations of the control language.
- $\mathsf{BCL}(\{a, r, g\}, g^*a\{a, r, g\}^*) = \mathsf{BMM}$, that is MM automata are in fact a particular instance of CL automata. This suggests that CL automata are a superclass of MM automata, and in fact this is stated next.
- $\mathsf{BMM} \subset \mathsf{BCL}$, as the previous point shows that $\mathsf{BMM} \subseteq \mathsf{BCL}$, and BMM is not closed under Boolean operations while BCL is.

3.5 Robust Quantum Computation

Fight entanglement with entanglement.
— John Preskill [Pre01].

The postulates of quantum mechanic, presented in Section 2.4.2, are (implicitly) referred to *closed* systems. However, in practice, it is impossible to completely isolate a physical system from the rest of the world, that is its environment. Therefore, every physical system is, to some extent, an *open* one; this means that unpredictable interactions with the environment may occur.

These interactions are obviously detrimental to performing computational tasks, where the evolution of the system must be strictly under control. This is true also of classical systems performing classical computations. In quantum computing, however, the problem is all the more crucial, since systems exhibiting quantum-mechanical behavior are usually very small ones, and therefore susceptible of perturbations even for causes of very little magnitude.

The phenomenon of perturbation of the quantum state is referred to as *decoherence*. Decoherence is therefore the main obstacle to the practical realization of a quantum computing device.

Classical computing devices solve the problem of the undesired "faulty" interactions with the environment by exploiting techniques of *error correction*. In extreme summary, the classical theory of error correction [MS98, Bay97] — initiated by von Neumann [vN56] — performs error correction by exploiting *redundancy*. Therefore, information is stored redundantly, and the redundancy is then used to reconstruct the correct state whenever it is corrupted.

This general scheme would seem unappliable to quantum computing models for at least three main reasons. First, as we have shown in Section 3.1, quantum states cannot be cloned, therefore it seems impossible to duplicate them to create redundant encodings. Second, since unitary transforms are uncountably many, we have a whole continuum of possible "transformation errors", while classical error correction assumes a discrete set of possible errors. Third, since measurement changes the quantum states, it seems impossible to even detect errors without altering the computational processes completely and undesirably.

Remarkably, all of these limitations can be overcome, and error correction for quantum states is indeed possible. Therefore, it is possible to realize robust quantum computers in spite of decoherence. In particular, it is possible to exploit the peculiarities of quantum mechanics (and in particular entanglement) to overcome its own limitations for error correction. This is in contrast with what happens with other computational models involving continuous quantities, such as analog computing. There, the effects of noise cannot be corrected, and thus those computations are not realizable in practice, at least in conformance with the theoretical models for them. See the informal comment by Bacon for a sharp further discussion of these aspects [Bac05].

Describing in some detail the results for quantum error correction, and robust quantum computation in general, would require to introduce a lot of material [Pre01]. In the remainder of this section, we simply sketch how a trivial quantum error correcting code works, and demonstrate it with a very simple example, taken from [RP00].

Error correction. Any model of error correction assumes a set $E = \{E_i\}_i$ of errors, where each E_i is an operator on quantum states of, say, n qubits. Therefore, each conceivable error in this model can be expressed as a linear combination $\sum_i e_i E_i$ for some coefficients e_i.

Then, an error correcting scheme consists of the following elements.

- An *encoding* C, which maps each base n-qubit vector to a $(n+s)$-qubit vector, introducing some redundancy. Note that this encoding is performed on the prepared states at the beginning of a computation, and therefore can be realized without altering the computation. All the computing operations are then applied accordingly, exploiting quantum parallelism.

- A *syndrome extractor* operator S that maps, someway, each $(n+s)$-qubit vector to some set of indices $\epsilon = \{j \text{ for some } E_j \in E\}$ of *correctable errors*.

This syndrome extraction must be performed in the quantum state. Notice that it must be, for all j in ϵ and for all states x: $j \in S(E_j(C(x)))$.

- A set of *error recovery* operators, $E^{-1} = \{E_i^{-1}\}_i$, one for each $E_i \in E$. Clearly, it must be, $E_i^{-1}(E_i(C(x)))$ for all used i's.

Let us now summarize how an error-correcting scheme would be employed, illustrating it through a very simple example.

Let us consider the encoding C that maps one qubit to $1 + 2 = 3$ qubits as follows.

base state	**encoded state**
$\|0\rangle$	$\|000\rangle$
$\|1\rangle$	$\|111\rangle$

We consider errors which flip single qubits, according to the Pauli X matrix. Therefore, we have:

$$E = \{E_0 = I \otimes I \otimes I, E_1 = X \otimes I \otimes I, E_2 = I \otimes X \otimes I, E_3 = I \otimes I \otimes X\}$$

The syndrome extractor operator performs the mapping:

$$S: \ |b_0, b_1, b_2, 000\rangle \rightarrow |b_0, b_1, b_2, b_0 \oplus b_1, b_0 \oplus b_2, b_1 \oplus b_2\rangle$$

Notice that S is in this case a mapping over 6 qubit states, since the last 3 qubits are used to store the set of indices of correctible errors. More precisely, we assume that the error E_i is encoded as follows:

encoding	**error**
$\|000\rangle$	E_0
$\|110\rangle$	E_1
$\|101\rangle$	E_2
$\|011\rangle$	E_3

Finally note that $E_i^{-1} = E_i$ for all $i = 0, \ldots, 3$, since double flips correspond to identities.

Let us now assume we end up, after some computational steps, to the state:

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle)$$

in the “error-correction basis”. Let us also consider the error:

$$\overline{E} = \frac{4}{5}E_1 + \frac{3}{5}E_2$$

Therefore, after some simple calculation, we see that the error state can be written as:

$$|\psi_{\overline{\mathrm{E}}}\rangle = \overline{E}\,|\psi\rangle = \frac{4}{5\sqrt{2}}(|100\rangle - |011\rangle) + \frac{3}{5\sqrt{2}}(|010\rangle - |101\rangle)$$

Now, to perform error correction, the syndrome extractor operator is applied to the error state padded with 3 extra qubits initialized to $|000\rangle$, where the result of the syndrome extraction will be stored. Therefore, we end up in a state where the indices of the errors are encoded in the last 3 qubits. In fact, we get:

$$\begin{aligned}
|\psi''\rangle &= S\left(|\psi_{\overline{\mathrm{E}}}\rangle \otimes |000\rangle\right) = S\left(\left(\overline{E}\,|\psi\rangle\right) \otimes |000\rangle\right) \\
&= S\left(\frac{4}{5\sqrt{2}}\left(|100,000\rangle - |011,000\rangle\right) + \frac{3}{5\sqrt{2}}\left(|010,000\rangle - |101,000\rangle\right)\right) \\
&= \frac{4}{5\sqrt{2}}\left(|100,110\rangle - |011,110\rangle\right) + \frac{3}{5\sqrt{2}}\left(|010,101\rangle - |101,101\rangle\right) \\
&= \frac{4}{5\sqrt{2}}\left(|100\rangle - |011\rangle\right)|110\rangle + \frac{3}{5\sqrt{2}}\left(|010\rangle - |101\rangle\right)|101\rangle
\end{aligned}$$

Next, the last three qubits of the state are measured. We get some random result, and precisely either $|110\rangle$ or $|101\rangle$. Assuming that the result is the most likely, that is the former, the resulting states becomes:

$$|\psi'''\rangle \;=\; \frac{1}{\sqrt{2}}\left(|100\rangle - |011\rangle\right)|110\rangle$$

The measured state corresponds to the error E_1. Thus, we apply the $E_1^{-1} = E_1$ to the first three qubits of the state, finally getting:

$$|\psi''''\rangle \;=\; \left(E_1 \otimes I^{\otimes 3}\right)|\psi'''\rangle = \frac{1}{\sqrt{2}}\left(|000\rangle - |111\rangle\right) \otimes |110\rangle = |\psi\rangle \otimes |110\rangle$$

which is the original state, recovered.

3.6 Quantum Physical Realizations

Quantum phenomena do not occur in a Hilbert space. They occur in a laboratory.
— Asher Peres [Per95, Per02].

If quantum computing was to remain a purely speculative technology, it would not deserve most of the efforts that the research community is developing. On the contrary, one of the earliest drives towards theoretical research on quantum computing models was exactly the belief that those models would eventually be implemented.

Choosing and realizing physical implementations of quantum computers is a task for the physicists. Indeed, current experimentation in implementations is a topic of large interest for experimental physicists and engineers. Besides realizing quantum computers, the effort might even have the — extremely important — side effect of helping in discovering new physical theories, while probing the limits of quantum mechanics.

In this section, we will present this important subject under three different perspectives. First, we will list the most important characteristics a physical system should possess in order to be a viable candidate for implementing a quantum computer. Afterwards, we will list some technologies for actual implementations that have been tried experimentally. Finally, we will cite physical realizations of some of the algorithms and protocols we have seen — only theoretically — in the previous sections.

Physical implementation desiderata. DiVincenzo summarizes [DiV00] the most important requirements a physical system should possess in order to be a candidate for implementing physical computations. Let us consider them.

- *Scalable physical system with well characterized qubits.* This means that some physical states of the system must be suitable to represent a two-level quantum state in a well-defined way. For instance, we may have a base state, associated with the value $|0\rangle$ and an excited state, which would represent a state $|1\rangle$. These representations must be scalable, in that it must be possible to couple several qubit systems in such a way that they do not interact unexpectedly but keep their individual qubit states as long as possible.
- *Initializable to a simple fiducial state, such as* $|00\cdots 0\rangle$. This would be the starting point of any computation, and therefore the initialized state must be set with high confidence and precision.
- *Long relevant decoherence time, much longer than the gate operation time.* The decoherence time is the minimum time within which a quantum state can be considered stable and not corrupted by unexpected interaction with the external environment. The gate operation time is instead the maximum time required for a quantum unitary operation to take place. Ideally, we would like a system where the decoherence time is very long, so that error correction must be performed only with reasonably long delays, and where the gate time is very short, so that computations are performed very fast. In practice, we usually require a ratio in the order of 10^4 between the former and the latter times, at least given the current error correcting performances.
- *A universal set of quantum gates.* The system must offer a viable way to implement a universal set of quantum gate operations, that is all of one-qubit operations (within a certain precision) and at least one two-qubit operations (e.g., the C-not).
- *A qubit-specific measurement capability.* This corresponds to the ease of measurement of the system state after a computation has been performed. This “output” capability is in contrast with the requirement of having a low decoherence effect, since they both involve interaction with the environment, being the latter an undesired one, and the former a required one.

Candidate systems for physical implementations. Let us now consider some systems that have been experimented as devices for implementing quantum computation.

Ion traps. Ion traps are lattices of solid material where ions can be trapped and manipulated using interacting laser beams. This is a technology taken from material science, adapted to the need of quantum computation.

Atoms in optical lattices. This technology is, similarly to the ion traps case, a lattice trapping some quantum particles. Here, however, the lattice is made entirely of electromagnetic radiation, and the particles are neutral atoms, rather than charged ions. The main advantage is that neutral particles interact weakly than charged one, thus having a larger decoherence time. Most of the physics for these devices comes from the experience with Bose-Einstein condensates.

Solid state technologies. Examples of solid state technologies are the quantum dots, which are sort of "artificial atoms" where electrons are trapped within matter artificial impurities. The main advantages of solid state technologies over other technologies is their speed at performing unitary operations, and their good controllability. The drawback is having to deal with short decoherence times.

Optic devices. Using photon beams as quantum system is clearly the best technology for implementing communication quantum devices, as photons travel at the speed of light, and have weak interactions with the environment (being massless). On the other hand, it is hard to perform computational tasks with optic devices, as implementing arbitrary unitary operations seems difficult.

Cavity quantum electrodynamics. In this technology, atoms or ions are coupled to photons by means of cavities. This seems a good way to exchange quantum information between optic and material qubits, which is desired in order to be able to build a quantum computer where some tasks (e.g., communication) are performed by optic technologies, and some others (e.g., memorization) by means of solid state ones, thus exploiting the advantages of each one.

Nuclear magnetic resonance. The nuclear magnetic resonance (NMR) technology consists of large amounts of identical atoms which are put into a resonant coherent state by means of very strong magnetic pulses. Historically, NMR has been a very important test bed for many of the ideas of quantum information processing, and several full implementations of quantum algorithms have first been realized using NMR. However, it seems unlikely that it will be a technology for the future, as it has inherent strong limitations for what concerns scalability (using large amounts of atoms in the same state, rather than simple subsystems) and read out of the results.

Some successful experiments and techniques. Some of the algorithms and protocols we have discussed in the previous sections have been actually implemented using some of the technologies we have outlined above. Moreover, some techniques have been studied that should make it possible to try other implementation media in the near future. Although most, if not all, of the realizations have been toy-scaled ones, they are encouraging in showing that the quantum computing paradigm is indeed implementable, and its workings are intimately related to how nature really behaves.

For instance, [BK00] describes how the dense coding protocols can be realized with electromagnetic quantities. Still with reference optical technologies, the Deutsch-Jozsa algorithm has been studied and realized [CY95]. Shor's algorithm has been concretely implementing using a NMR device [VSB$^+$01], factoring the number 15 into the product of 3 and 5 (!). Grover's algorithm has been implemented both using optical technologies [KMSW00] (using simulation techniques), and NMR [CGK98]. Finally, the key exchange protocol described in Section 3.3 has been implemented with success, both through fiber optical cables [HBK$^+$97], and through the atmosphere [HBK$^+$99], over distances of several kilometers. Quantum communication implementations are definitely less problematic than full quantum computing tasks, so much that we already witness some commercial implementations of the key exchange protocol [Mag].

4 Conclusions

We presented motivation and a general framework for the emerging field of quantum computation. In doing this, we showed how the efforts behind quantum computation can be seen as an attempt to develop more satisfactory definitions of the very idea of computation, putting them on a physical — rather than purely mathematical — ground.

We outlined the fundamental steps in the birth of quantum computation, from a historical perspective. We also presented a mathematical model of quantum computation, namely one based on an extension to the quantum case of the classical Boolean circuit model, showing how the new model is linked with the fundamental postulates of quantum mechanics.

After discussing these general features, we presented some issues within the broad field of quantum computation which deserve, in our opinion, some attention. We briefly introduced the idea of quantum information, discussing some very basic, but interesting, results. We presented the two families of quantum algorithms known to date, that is those based on Shor's factoring algorithm, and those based on Grover's search algorithm. We gave some hints as to if other classes of quantum algorithms may be discovered in the future, still in relation with the physical features of quantum theory. Clever applications of the principles of quantum computation to the design of secure cryptographic protocols were discussed; in particular, a key exchange protocol based on communications through exchange of quantum particles was presented. Afterwards, we introduced some theoretical models of finite state automata "enriched" with quantum characteristics. We discussed some features of them, especially for what concerns their expressive power, and their relations to classical finite automata. The important problem of how to build quantum computing circuits that are robust against unwanted noisy interactions with the environment was outlined, and some positive results in these directions were discussed and referenced. Finally, we presented some candidate real physical systems that may be suitable to implement quantum computation, and on which considerable experimental effort is being put toward that goal.

We conclude by noting that all the research in the field of quantum computation — that is being fervently pursued — is and will be profitable for a better understanding of the fundamental limits and capabilities that the physical universe puts on the ability to compute, that is to manipulate information. Given the ubiquity of information processing tasks in today's world, this better understanding is very likely to have a strong impact in several aspects of science and technology, at large.

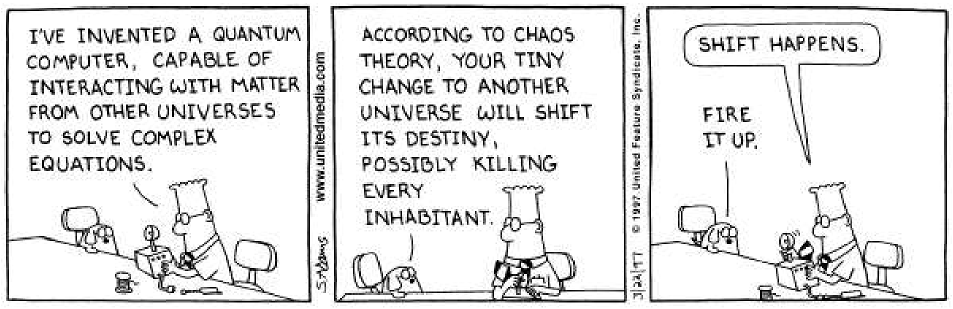
I'VE INVENTED A QUANTUM COMPUTER, CAPABLE OF INTERACTING WITH MATTER FROM OTHER UNIVERSES TO SOLVE COMPLEX EQUATIONS.
www.unitedmedia.com
S. Adams
ACCORDING TO CHAOS THEORY, YOUR TINY CHANGE TO ANOTHER UNIVERSE WILL SHIFT ITS DESTINY, POSSIBLY KILLING EVERY INHABITANT.
© 1997 United Feature Syndicate, Inc.
3/22/97
SHIFT HAPPENS.
FIRE IT UP.

Figure 7: Dilbert comic of 1997/3/22 by Scott Adams

Acknowledgements

This work was carried out as minor research for the Ph.D. program in Computer Science at Politecnico di Milano. I would like to thank Angelo Morzenti, for supporting my research interests and serving as minor advisor, and Carlo "Mere" Mereghetti, for his reviewing and proofreading work, as well as his kind support and expertise in quantum computing.

Note to the References

Several papers in the bibliography are available in electronic form through the `arXiv` archive. To get the abstract (and the full text) of any of them, just point your browser to the URL `http://arxiv.org/abs/category/number`; for instance `http://arxiv.org/abs/quant-ph/0011064` for the reference [Nie00].

References

[Aar04] Scott Aaronson. Multilinear formulas and skepticism of quantum computing. In *Proceedings of 36th Annual ACM Sympotium on Theory of Computing (STOC'04)*, pages 118–127. ACM Press, 2004.

[Aar05] Scott Aaronson. NP-complete problems and physical reality. *SIGACT News*, 2005. Complexity Theory Column.

[Aarar] Scott Aaronson. Multilinear formulas and skepticism of quantum computing. *SIAM Journal on Computing*, to appear. Journal version of [Aar04]. Also `quant-ph/0311039`.

[Adl94] Leonard M. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024, 1994.

[AF98] Andris Ambainis and Rusin Freivalds. 1-way quantum finite automata: Strengths, weakness and generalizations. In *Proceedings of the 39th Symposium on Foundations of Computer Science (FOCS'98)*, pages 332–342. ACM Press, 1998.

[Aha98] Dorit Aharonov. Quantum computation — a review. In Dietrich Stauffer, editor, *Annual Review of Computational Physics*, volume 6. World Scientific, 1998. Also `quant-ph/9812037`.

[AKV01] Andris Ambainis, Arnolds Kikusts, and Maris Valdats. On the class of languages recognizable by 1-way quantum finite automata. In Afonso Ferreira and Horst Reichel, editors, *Proceedings of the 18th Annual Symposium on Theoretical Aspects of Computer Science (STACS'01)*, volume 2010 of *Lecture Notes in Computer Science*, pages 305–316. Springer-Verlag, 2001. Also `quant-ph/0001005`.

[AL97] Daniel S. Abrams and Seth Lloyd. Simulation of many-body Fermi systems on a quantum computer. *Physical Review Letters*, 79:2586–2589, 1997.

[AL98] Daniel S. Abrams and Seth Lloyd. Nonlinear quantum mechanics implies polynomial-time solution for NP-complete and #P problems. *Physical Review Letters*, 81:3992–3995, 1998. Also `quant-ph/9801041`.

[App97] Andrew W. Appel. *Modern Compiler Implementation in ML*. Cambridge University Press, 1997.

[ASU86] Alfred V. Aho, Ravi Sethi, and Jeffrey D. Ullman. *Compilers*. Addison Wesley, 1986.

[AW02] Andris Ambainis and John Watrous. Two-way finite automata with quantum and classical states. *Theoretical Computer Science*, 287(1):299–311, 2002. Also `cs.CC/9911009`.

[Bac05] Dave Bacon. Digital, analog, deterministic, probabilistic. Blog entry `http://dabacon.org/pontiff/?p=882`, May 2005.

[Bay97] John Baylis. *Error Correcting Codes: A Mathematical Introduction.* Chapman & Hall, 1997.

[BB84] Charles H. Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computer Systems and Signal Processing*, pages 175–179. IEEE, 1984.

[BBBV97] Charles H. Bennett, Gilles Brassard, Ethan Bernstein, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997.

[BBHT98] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik*, 46:493–506, 1998. Also `quant-ph/9605034`.

[BC01] Alberto Bertoni and Marco Carpentieri. Regular languages accepted by quantum automata. *Information and Computation*, 165:174–182, 2001.

[BCSS97] Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and Real Computation.* Springer-Verlag, 1997.

[Bel64] John S. Bell. On the Einstein Podolsky Rosen paradox. *Physics*, 1:195, 1964. Reprinted in [Bel87].

[Bel66] John S. Bell. On the problem of hidden variables in quantum mechanics. *Reviews of Modern Physics*, 38:447, 1966. Reprinted in [Bel87].

[Bel82] John S. Bell. On the impossible pilot wave. *Foundations of Physics*, 12(10):989–999, 1982.

[Bel87] John S. Bell. *Speakable and Unspeakable in Quantum Mechanics.* Cambridge University Press, 1987.

[Ben73] Charles H. Bennet. Logic reversibility of computation. *IBM Journal of Research and Development*, 17(6):525–532, 1973.

[Ben80] Paul Benioff. The computer as a physical system: a microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *Journal of Statistical Physics*, 22:563–591, 1980.

[Ben82a] Paul Benioff. Quantum mechanical Hamiltoniam models of Turing machines. *Journal of Statistical Physics*, 29:515–546, 1982.

[Ben82b] Paul Benioff. Quantum mechanical Hamiltoniam models of Turing machines that dissipate no energy. *Physical Review Letters*, 48:1581–1585, 1982.

[Ben88] Charles H. Bennett. Notes on the history of reversible computation. *IBM Journal of Research and Development*, 32(1):16–23, 1988. Reprinted in [LR90, Chap. 4, pp. 281–288].

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum counting. `quant-ph/9805082`, 1998.

[BK00] Samuel L. Braunstein and H. J. Kimble. Dense coding for continuous variables. *Physical Review A*, 61(4), 2000. Also `quant-ph/9910010`.

[BMP$^+$99] P. Oscar Boykin, Tal Mor, Matthew Pulver, Vwani Roychowdhury, and Farrokh Vatan. On universal and fault-tolerant quantum computing. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS'99)*, pages 486–494. IEEE, 1999. Also `quant-ph/9906054`.

[BMP03a] Alberto Bertoni, Carlo Mereghetti, and Beatrice Palano. Golomb rulers and difference sets for succint quantum automata. *International Journal of FOundations of Computer Science*, 14(5):871–888, 2003.

[BMP03b] Alberto Bertoni, Carlo Mereghetti, and Beatrice Palano. Quantum computing: 1-way quantum automata. In Zoltán Ésik and Zoltán Fülöp, editors, *Proceedings of the 7th International Conference on Developments in Language Theory*, pages 1–20, 2003.

[BP02] Alex Brodsky and Nicholas Pippenger. Characterizations of 1-way quantum finite automata. *SIAM Journal on Computing*, 31(5):1456–1478, 2002.

[BV93] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. In *Proceedings of the 25th ACM Symposium on the Theory of Computation (STOC'93)*, pages 11–20. ACM Press, 1993.

[BV97] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26:1411–1437, 1997.

[BW92] Charles H. Bennett and Stephen J. Wiesner. Communication via one- and two-particles operators on Einstein-Podolsky-Rosen states. *Physical Review Letters*, 69(20):2881–2884, 1992.

[Cav] Carlton M. Caves. Brief description of CMC research. Online at `http://info.phys.unm.edu/~caves/research.html`.

[CGK98] Isaac L. Chuang, Neil Gershenfeld, and Mark Kubinec. Experimental implementation of fast quantum searching. *Physical Review Letters*, 80(15):3408–3411, 1998.

[Chu36] Alonzo Church. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58:345–363, 1936.

[Chu41] Alonzo Church. *The Calculi of Lambda-Conversion*, volume 6 of *Annals of Mathematical Studies.* Princeton University Press, 1941.

[CLRS01] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* MIT Press, 2nd edition, 2001.

[CS96] A. Robert Calderbank and Peter W. Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54:1098–1105, 1996.

[CVZ$^+$98] Isaac L. Chuang, Lieven M.K. Vandersypen, Xinlan Zhou, Debbie W. Leung, and Seth Lloyd. Experimental realization of a quantum algorithm. *Nature*, 393:143–146, 1998.

[CY95] Isaac L. Chuang and Y. Yamamoto. A simple quantum computer. *Physical Review A*, 52:3489–3496, 1995. Also `quant-ph/9505011`.

[Deu85] David Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London A*, 400:97–117, 1985.

[DiV95] David P. DiVincenzo. Two-bit gates are universal for quantum computation. *Physical Review A*, 51(2):1015–1022, 1995.

[DiV00] David P. DiVincenzo. The physical implementation of quantum computation. *Fortschritte der Physik*, 48:771, 2000.

[DJ92] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London A*, 439:553–558, 1992.

[Fey60] Richard P. Feynman. There's plenty of room at the bottom: an invitation to open up a new field of physics. *Engineering and Science (California Institute of Technology)*, 23(5):22–36, 1960.

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

[Fey85a] Richard P. Feynman. Seminar at AT&T, 1985. Edited (no jokes) text in [Fey85b].

[Fey85b] Richard P. Feynman. Quantum mechanical computers. *Optics News*, pages 11–21, February 1985. Reprinted in [HA00].

[Fey86] Richard P. Feynman. Quantum mechanical computers. *Foundations of Physics*, 16(6):507–531, 1986.

[FP00] Christopher A. Fuchs and Asher Peres. Quantum mechanics needs no 'interpretation'. *Physics Today*, March 2000.

[Fra99] Michael P. Frank. *Reversibility for Efficient Computing.* PhD thesis, EECS Department, Massachussets Institute of Technology, 1999.

[Gis90] N. Gisin. Weinberg's non-linear quantum mechanics and superluminal communications. *Physical Review A*, pages 1–2, 1990.

[GK02] Marats Golovkins and Maksim Kravtsev. Probabilistic reversible automata and quantum automata. In *Proceedings of the 8th International Computing and Combinatorics Conference (ICCC'02)*, volume 2387 of *Lecture Notes in Computer Science*, pages 574–583. Springer-Verlag, 2002.

[Got96] Daniel Gottesman. A class of quantum error-correcting codes saturating the quantum Hamming bound. *Physical Review A*, 54:1862–1868, 1996.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Annual ACM Symposium on the Theory of Computation (STOC'96)*, pages 212–219. ACM Press, 1996.

[Gro97] Lov K. Grover. Quantum mechanics helps in searching for a needle in a haystack. *Physical Review Letters*, 79:325–328, 1997.

[Gro98] Lov K. Grover. A framework for fast quantum mechanical algorithms. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computation (STOC'98)*, pages 53–62, 1998.

[Gru99] Jozef Gruska. *Quantum Computing.* McGraw-Hill, 1999.

[HA00] Tony Hey and Robin W. Allen, editors. *Feynman Lectures on Computation.* Perseus Books Group, 2000.

[HBK$^+$97] Richard J. Hughes, William T. Buttler, Paul G. Kwiat, Gabriel G. Luther, George L. Morgan, Jane E. Nordholt, C. Glen Peterson, and Charles M. Simmons. Secure communications using quantum cryptography. In Steven P. Hotaling and Andrew R. Pirich, editors, *Photonic Quantum Computing*, volume 3076 of *Proceedings of SPIE*, pages 2–11, 1997.

[HBK$^+$99] Richard J. Hughes, William T. Buttler, Paul G. Kwiat, Steve K. Lamoreaux, George L. Morgan, Jane E. Nordholt, and C. G. Peterson. Practical quantum cryptography for secure free-space communications. Technical Report LA-UR-99-737, Los Alamos National Laboratory, 1999. Also `quant-ph/9905009`.

[Hof99] Douglas R. Hofstadter. *Gödel, Escher, Bach: An Eternal Golden Braid.* Basic Books, 20th anniversary edition, 1999.

[Hog98] Tad Hogg. Highly structured searches with quantum computers. *Physical Review Letters*, 80:2473–2476, 1998.

[Kit96] Alexei Y. Kitaev. Quantum measurements and the Abelian stabilizer problem. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 3, 1996. Also `quant-ph/9511026`.

[Kit97] Alexei Y. Kitaev. Quantum computations: Algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191–1249, 1997.

[Kle36] Stephen C. Kleene. General recursive functions of natural numbers. *Mathematische Annalen*, 112:727–742, 1936.

[KMSW00] P. G. Kwiat, J. R. Mitchell, P. D. D. Schwindt, and A. G. White. Grover's search algorithm: An optical approach. *Journal of Modern Optics*, 47:257–266, 2000.

[Kni95] E. Knill. Approximation by quantum circuits. Technical Report LAUR-95-2225, LANL, 1995. Also `quant-ph/9508006`.

[Kuh96] Thomas S. Kuhn. *The Structure of Scientific Revolutions.* University of Chicago Press, 3rd edition, 1996.

[KW97] Attila Kondacs and John Watrous. On the power of quantum finite state automata. In *Proceedings of the 38th Symposium on Foundations of Computer Science (FOCS'97)*, pages 66–75. ACM Press, 1997.

[Lan61] Rolf Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5:183–191, 1961. Reprinted in [LR90, Chap. 4, pp. 188–196].

[Lan91] Rolf Landauer. Information is physical. *Physics Today*, 44(5):23–29, 1991.

[LL94] Arjen K. Lenstra and Hendrik W. Lenstra, Jr., editors. *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Computer Science.* Springer-Verlag, 1994.

[LR90] Harvey S. Leff and Andrew F. Rex, editors. *Maxwell's Demon: Entropy, Information, Computing.* Princeton series in physics. Princeton University Press, 1990.

[Mag] MagiQ, quantum information solutions for the real world. `http://www.magiqtech.com/`.

[Man80] Yuri I. Manin. Computable and uncomputable. Sovetskoye Radio, Moscow, 1980. In Russian. Partial English translation in [Man99].

[Man99] Yuri I. Manin. Classical computing, quantum computing, and Shor's factoring algorithm. quant-ph/9903008, 1999.

[MC00] Cristopher Moore and James P. Crutchfield. Quantum automata and quantum grammars. *Theoretical Computer Science*, 237(1–2):275–306, 2000.

[ME98] Michele Mosca and Artur Ekert. The hidden subgroup problem and eigenvalue estimation on a quantum computer. In Colin P. Williams, editor, *Proceedings of the 1st NASA International Conference on Quantum Computing and Quantum Communication*, volume 1509 of *Lecture Notes in Computer Science*, pages 174–188. Springer-Verlag, 1998. Also quant-ph/9903071.

[Mer03] N. David Mermin. From Cbits to Qbits: Teaching computer scientists quantum mechanics. *American Journal of Physics*, 71(1):23–30, 2003. Also 0207118.

[MM69] B. Meltzer and D. Michie, editors. *Machine Intelligence*, volume 5. Edinburgh University Press, 1969.

[Moo65] Gordon E. Moore. Cramming more components onto integrated circuits. *Electronics Magazine*, 38(8), 1965.

[MP02] Carlo Mereghetti and Beatrice Palano. On the size of one-way quantum finite automata with periodic behaviors. *Informatique Théorique et Applications (Theoretical Informatics and Applications)*, 36(3):277–291, 2002.

[MPP01] Carlo Mereghetti, Beatrice Palano, and Giovanni Pighizzini. Note on the succinctness of deterministic, nondeterministic, probabilistic and quantum finite automata. *Informatique Théorique et Applications (Theoretical Informatics and Applications)*, 35(5):477–490, 2001.

[MS98] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes.* North Holland, 1998.

[MWKZ96] Klaus Mattle, Harald Weinfurter, Paul G. Kwiat, and Anton Zeilinger. Dense coding in experimental quantum communication. *Physical Review Letters*, 76(25):4656, 4659 1996.

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[Nie00] Michael A. Nielsen. Introduction to quantum information theory. Talk abstract, September 2000. quant-ph/0011064.

[Orp97] Pekka Orponen. A survey of continuous-time computation theory. In Ding-Zhu Du and Ker-I Ko, editors, *Advances in Algorithms, Languages, and Complexity*, pages 209–224. Kluwer, 1997.

[Pap94] Christos Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

[Pău00] Gheorghe Păun. Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143, 2000.

[Per95] Asher Peres. *Quantum Theory: Concepts and Methods.* Kluwer Academic, 1995.

[Per02] Asher Peres. What's wrong with these observables? `quant-ph/0207020`, 2002.

[Pos36] Emil Post. Finite combinatory processes: Formulation I. *Journal of Symbolic Logic*, 1:103–105, 1936.

[Pre01] John Preskill. Fault-tolerant quantum computation. In Hoi-Kwong Lo, Sandu Popescu, and Tim Spiller, editors, *Introduction to Quantum Computation and Information*, pages 213–269. World Scientific, 2001. Also `quant-ph/9712048`.

[RP00] Eleanor G. Rieffel and Wolfgang Polak. An introduction to quantum computing for non-physicists. *ACM Computing Surveys*, 32(3):300–335, 2000. Also `quant-ph/9809016`.

[RSA78] Ronald R. Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[RZBB94] Michael Reck, Anton Zeilinger, Herbert J. Bernstein, and Philip Bertani. Experimental realization of any discrete unitary operator. *Physical Review Letters*, 73(1):58–61, 1994.

[Sch95] Bruce Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* Wiley & Sons, 2nd edition, 1995.

[SD99] Sudeshna Sinha and William L. Ditto. Computing with distributed chaos. *Physical Review E*, 60(1):363–377, 1999.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete log and factoring. In Shafi Goldwasser, editor, *Proceedings of the 35th Annual Symposium on the Foundations of Computer Science (FOCS'94)*, pages 124–134. IEEE Computer Society, 1994.

[Sho95] Peter W. Shor. Scheme for reducing decoherence in quantum memory. *Physical Review A*, 52:2493–2496, 1995.

[Sho96] Peter W. Shor. Fault tolerant quantum computation. In *Proceedings of the 37th Symposium on the Foundations of Computer Science (FOCS'96)*, pages 56–65. IEEE Computer Society, 1996.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[Sho03] Peter W. Shor. Why haven't more quantum algorithms been found? *Journal of the ACM*, 50(1):87–90, 2003.

[Sim94] Daniel R. Simon. On the power of quantum computation. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS'94)*, pages 116–123. IEEE Press, 1994.

[Sim97] Daniel R. Simon. On the power of quantum computation. *SIAM Journal on Computing*, 26(5):1474–1483, 1997.

[Sin99] Sudeshna Sinha. A "computing" chaotic system. In N. Pradhan, P. E. Rapp, and R. Sreenivasan, editors, *Nonlinear Dynamics and Brain Functioning*. Nova Science, 1999.

[Sol95] Robert Solovay. Unpublished manuscript, 1995.

[Ste96] Andrew M. Steane. Error correcting codes in quantum theory. *Physical Review Letters*, 77:793–797, 1996.

[Tur36] Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the Londom Mathematical Society*, 2(42):230–265, 1936. Corrections in no. 43, pp. 544-546, 1937.

[Tur48] Alan M. Turing. Intelligent machinery. Report, National Physical Laboratory, 1948. Published in [MM69].

[vN55] John von Neumann. *Mathematical Foundations of Quantum Mechanics*. Princeton University Press, 1955. Reprinted in 1996.

[vN56] John von Neumann. Probabilistic logics and the synthesis of reliable organisms from unreliable components. In *Automata Studies*, pages 329–378. Princeton University Press, 1956.

[VSB$^+$01] Lieven M. K. Vandersypen, Matthias Steffen, Gregory Breyta, Costantino S. Yannoni, Mark H. Sherwood, and Isaac L. Chuang. Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance. *Nature*, 414:883–887, 2001. Also `quant-ph/0112176`.

[Wie83] Stephen Wiesner. Conjugate coding. *SIGACT News*, 15:78–88, 1983.

[WZ82] William K. Wootters and Wojciech H. Zurek. A single quantum cannot be cloned. *Nature*, 299:802–803, 1982.

[Yao93] Andrew C.-C. Yao. Quantum circuit complexity. In *Proceedings of the 34th IEEE Symposium on Foundations of Computer Science (FOCS'93)*, pages 352–361, 1993.

[Yao03] Andrew C.-C. Yao. Classical physics and the Church-Turing thesis. *Journal of the ACM*, 50(1):100–105, 2003.

[Zal99] Christof Zalka. Grover's quantum searching algorithm is optimal. *Physical Review A*, 60:2746–2751, 1999.