REVIEWS OF MODERN PHYSICS, VOLUME 74, APRIL 2002

Information and computation: Classical and quantum aspects

A. Galindo* and M. A. Martín-Delgado†

Departamento de Física Teórica I, Facultad de Ciencias Físicas, Universidad Complutense, 28040 Madrid, Spain

(Published 8 May 2002)

Quantum theory has found a new field of application in the realm of information and computation during recent years. This paper reviews how quantum physics allows information coding in classically unexpected and subtle nonlocal ways, as well as information processing with an efficiency largely surpassing that of the present and foreseeable classical computers. Some notable aspects of classical and quantum information theory will be addressed here. Quantum teleportation, dense coding, and quantum cryptography are discussed as examples of the impact of quanta on the transmission of information. Quantum logic gates and quantum algorithms are also discussed as instances of the improvement made possible in information processing by a quantum computer. Finally the authors provide some examples of current experimental realizations for quantum computers and future prospects.

CONTENTS

I. INTRODUCTION

The twentieth century opened with Planck's (1900) discovery of quanta, which was followed by the formu-

*Electronic address: agt@fis.ucm.es
†Electronic address: mardel@miranda.fis.ucm.es

lation of quantum theory during the first few decades. As the century went by, we witnessed a continuous increase in the applications of quantum mechanics, beginning with atomic physics and continuing with nuclear and particle physics, optics, condensed matter, and countless other developments. As the century was closing a new field of applications emerged that gave quantum physics a refreshing twist. While it seems inevitable that physics would be affected by the availability of more and more powerful computers, which have revolutionized many areas of science, it is more surprising to find that quantum physics may influence the fields of information and computation in a new and profound way. For instance, fundamental aspects of quantum mechanics such as those entering Einstein, Podolsky, and Rosen (1935) states have found unexpected applications in information transmission and cryptography.

Why has this happened? It began with the realization that information has a physical nature (Landauer, 1961, 1991, 1996). It is imprinted on a physical support (the rocky wall of a cave, a clay tablet, a parchment, a sheet of paper, a magneto-optic disk, and so forth), it cannot be transmitted faster than light in vacuum, and it abides by natural laws. The statement that information is physical does not simply mean that a computer is a physical object, but in addition that information itself is a physical entity. In turn, this implies that the laws of information transmission are restricted or governed by the laws of physics—in particular, those of quantum physics. In fact these laws implying linearity, entanglement of states, nonlocality, and the indetermination principle make possible new and powerful transmission tools and information treatments, as well as a prodigious efficiency of computation.

A typical computation is implemented through an algorithm in a computer. This algorithm is now regarded as a set of physical operations, and the registers of the quantum computer are considered to be states of a quantum system. The familiar operation of initializing data for a program to run is replaced by the preparation of an initial quantum state, and the usual tasks of writing programs and running them correspond, in the new formulation, to finding appropriate Hamiltonians for their time-evolution operators to lead to the desired output. This output is retrieved by a quantum measurement of the register, which has deep implications for the way quantum information must be handled.

We shall see that information and computation blend well with quantum mechanics. Their combination has led to unexpected new ways that information can be transmitted and processed, extending the known capabilities in the field of classical information to unsuspected limits, sometimes entering the realm of science fiction, sometimes surpassing it.

The advances have been especially remarkable in the field of cryptography, where they have provided absolutely secure systems for the quantum distribution of keys. Quantum computation is also one of the hot research fields in current physics, where the challenge is to realize experimentally a computer complex enough to implement the new algorithms exploiting massive parallelism. Such a quantum computer would offer a dramatic improvement for solving hard or classically intractable problems.

We first review the essentials of quantum information theory and then discuss several of their consequences and applications, some specifically quantum, such as quantum teleportation and dense coding, and some with a classical echo, such as quantum cryptography. Next we review the fundamentals of quantum computation, describing the notion of a quantum Turing machine and its practical implementation with quantum circuits. We describe the idea of elementary quantum gates for universal computation and examine how this extends the classical counterpart. We also provide a discussion of the basic quantum algorithms. Finally we give a general overview of some of the possible physical realizations of quantum computers.

In both the information and computation sections we place special emphasis on providing an introduction to the classical aspects of these disciplines in order to better clarify what quantum theory adds to them. Actually, this is also what we do in physics.

II. CLASSICAL INFORMATION

Information is discretized: it comes in irreducible packages. The elementary unit of classical information is the *bit* (or *cbit*, for classic bit), a classical system with only two states, 0 and 1 (false and true, no and yes, . . .). Any text can be coded into a string of bits; for instance, it is enough to assign to each symbol its ASCII code number in binary form and append a parity check bit. For example, *quanta* can be coded as

$$11100010\ \ 11101011\ \ 11000011\ \ 11011101\ \ 11101000\ \ 11000011.$$

Each bit can be stored physically; in classical computers, each bit is registered as a charge state of a capacitor ($0 = \text{discharged}, 1 = \text{charged}$). They are distinguishable macroscopic states and rather robust or stable. They are not spoiled when they are read in (if carefully done) and they can be cloned or replicated without any problem.

Information is not only stored; it is usually transmitted (communication) and sometimes processed (computation).

A. The theorems of Shannon

The classical theory of information is due to Shannon (1948, 1949), who in two seminal works definitively laid down its principles in 1948. With his celebrated noiseless coding theorem he showed how compressible a message can be, or equivalently, how much redundancy it has. Likewise with his coding theorem in a noisy channel, he also found the minimum redundancy that must be present in a message in order for it to be comprehensible when reaching the receiver, despite the noise.

Let $A := \{a_1, \ldots, a_{|A|}\}$ be a finite alphabet, endowed with a probability distribution $p_A : a_i \mapsto p_A(a_i)$, with

$\Sigma_{1\leqslant i\leqslant|A|}p_A(a_i)=1$. Sometimes we shall write this as $A := \{a_i, p_A(a_i)\}_{i=1}^{|A|}$. Let us consider messages or character strings $x_1x_2\cdots x_n \in A^n$, originating from a memoryless source, i.e., a symbol a appears in a given place with probability $p_A(a)$, independently of the symbols entering the remaining sites in the chain.[1] Shannon's first theorem asserts that, if $n \gg 1$, the information supplied by a generic message of n characters [and thus ($n\log_2|A|$) bits long] essentially coincides with that transmitted by another shorter message, of bit length $nH(A)$, where H is Shannon's entropy,

$$H(A) = -\sum_{1\leqslant i\leqslant|A|} p_A(a_i)\log_2 p_A(a_i) \in [0,\log_2|A|]. \tag{1}$$

In other words, each character is compressible up to $H(A)$ bits on average; moreover, this result is optimal (Roman, 1992; Schumacher, 1995; Welsh, 1995; Preskill, 1998).

The basic idea underlying the proof is simple: to take notice only of the typical messages. Let us assume for clarity a binary alphabet ($A=\{0,1\}$). Let $p, 1-p$ be the probabilities of 0,1, respectively. In a long message of n bits ($n \gg 1$), there will be approximately np 0's. Let us call typical messages those with a number of 0's of the order of np. Asymptotically ($n\to\infty$) there are $2^{nH(A)}$ of them, among a total of 2^n messages. The probability $P:(x_1,\dots,x_n)\mapsto p(x_1)\cdots p(x_n)$ of the messages n ($\gg 1$) bits long tends to get concentrated on this reduced ensemble consisting of the typical strings, which explains Shannon's result. The atypical messages can be ignored in probability. It suffices to transmit through the communication channel (assumed to be completely noiseless) the binary number of length $nH(A)$ assigned to each typical message upon common agreement between the sender and the recipient, so that the emitted message can be identified on reception.[2] The optimality of Shannon's first theorem is easily arguable: all $2^{nH(A)}$ typical sequences are asymptotically equiprobable and thus they cannot be represented faithfully with fewer than $nH(A)$ bits.

If the transmission channel is noisy (the common case), the fidelity of the information is lost, since some bits may get corrupted along the way. To counteract the noise of a given channel one resorts to redundancy, by cleverly coding each symbol with more bits than strictly necessary so that the erroneous bits might be easily detected and restored. A price is paid however, since the transmission of essential information is thus made slower. Shannon's wonderful second theorem quantifies this issue.

Let X be the alphabet of the transmitter station (of a memoryless source) and Y be that of the receiver station. Let $[p_{Y|X}(y_j|x_i)]$ be the stochastic matrix for that channel, with entries given by the probabilities that the input symbol $x_i \in X$ appears as $y_j \in Y$ on output. The marginal probability distribution for Y is given by $p_Y(y_j)=\Sigma_i[p_{Y,X}(y_j,x_i) := p_{Y|X}(y_j|x_i)p_X(x_i)]$. The channel's ability to transmit information is measured by its *capacity* $C := \sup_{p_X} I(X:Y) = \max_{p_X} I(X:Y)$, where $I(X:Y)=I(Y:X)$ is the *mutual information*,

$$I(X:Y) := \sum_j \sum_i p_{Y,X}(y_j,x_i)\log_2\frac{p_{Y,X}(y_j,x_i)}{p_Y(y_j)p_X(x_i)}, \tag{2}$$

or the information about X (Y) conveyed by Y (X). The convexity of the log makes $I(X:Y)\geqslant 0$ (knowing Y can never decrease the information about X).

The capacity C may be viewed as the number of output bits per input symbol that are correctly transmitted. Its computation is usually very difficult.

Many channels are binary symmetric: each transmitted bit has the same probability p of being reversed, i.e., of being erroneous upon arrival. These are the channels considered here. For them we have $C=1-H_2(p) =: C(p)$, with $H_2(p) := -p\log_2 p-(1-p)\log_2(1-p)$. Note that a channel with $p=\frac{1}{2}$ has capacity $C(\frac{1}{2})=0$ and would be totally useless for transmission since it would transform any input binary word into a random ouput sequence. Thus we shall assume that $p<\frac{1}{2}$.

In the transmission of a word $w\in\{0,1\}^n$, an error $e \in \{0,1\}^n$ may be produced such that the received word is $w'=w+e$ (addition mod 2). A subset of words $\mathcal{C}_n \subset \{0,1\}^n$ encoding (i.e., in bijective correspondence with) a collection of messages is said to be an *error-correcting classical code* for $e\in\mathcal{E}_n\subset\{0,1\}^n$ if $(w+\mathcal{E}_n)\cap(w'+\mathcal{E}_n)=\varnothing$ for any $w\neq w'\in\mathcal{C}_n$. That is, regardless of the distortion produced by the errors on a code word $w\in\mathcal{C}_n$, there is no overlap between the different sets $w+\mathcal{E}_n$, and decoding is possible without ambiguities. If, upon previous agreement, it is known which specific message corresponds to each code word, it will be enough to send this one word instead of the message; the message can be recovered at the other end of the channel after "cleaning up" the received word from the possible errors that might affect it. In this way the transmitted code word can be identified and decoded. In the practical use of a code $\mathcal{C}_n$, mistakes can occur in the restoration of the message, caused by errors outside $\mathcal{E}_n$, that is, out of the security framework of the code. But as long as the frequency of failures remains very low, the risk will be bearable. It is apparent that, to minimize this risk, the words of the code should be as far apart from each other as possible (in the Hamming sense, i.e., in the number of bits in which they differ) so that errors caused by overlap between two distinct words of code will diminish.

[1]The natural languages are not like these (for instance, in ordinary Spanish there exists no digram like QÑ). Nevertheless, they can be considered, to a good approximation, as a limit of ergodic Markovian languages to which the Shannon theorem can be extended (Welsh, 1995).

[2]There exist very practical methods for classical coding with an efficiency close to the optimal value, such as the Huffman code (Roman, 1992), with multiple applications (facsimile, digital TV, etc.). The essence of this code is to assign shorter binary strings to the most frequent symbols.

One defines the *rate* of the code $\mathcal{C}_n$ as $R:=\log_2|\mathcal{C}_n|/n$. This measures the number of informative bits per transmitted bit. It is easy to argue that in order for the code to be reliable, its rate must not exceed the capacity of the channel: $R\leqslant C$. In fact, when transmitting a code word w with length n, a number of np reversed bits will be produced on average, hence an error e that will likely be one of the $2^{nH_2(p)}$ typical sequences. For the decoding to be reliable, there should be no overlap between the error spheres with centers at the code words, and thus $2^{nH_2(p)}|\mathcal{C}_n|\leqslant 2^n$, thereby $R\leqslant C$. This result suggests that the capacity C is an upper bound to all faithful transmission rates.

Shannon's second theorem closes this issue in the asymptotic limit. Suppose, given a binary symmetric channel, a transmission rate R not exceeding the capacity of the channel $(0<R<C)$, an $\epsilon>0$ arbitrarily small, and any sequence $\{N_n\}_1^\infty$ of integers such that $1\leqslant N_n \leqslant 2^{nR}$. Then the theorem asserts that there exist codes $\{\mathcal{C}_n\subset\mathbb{Z}_2^n\}_1^\infty$ with N_n elements (code words), appropriate decision schemes for decoding, and an integer $n(\epsilon)$, such that the *fidelity* $F(\mathcal{C}_n)$ or probability that a given decoded message coincides with the original is $\geqslant 1-\epsilon$ (that is, the maximum probability of error in the identification of the code word on reception is $\leqslant\epsilon$) for all $n \geqslant n(\epsilon)$ (Roman, 1992; Welsh, 1995). Moreover, it is possible to make the error probabilities tend to 0 exponentially in n.

The theorem is optimal: the capacity C should not be exceeded if the transmission is to be faithful. As a matter of fact, it is known that for each sequence of codes $\{\mathcal{C}_n\}_1^\infty$ with $|\mathcal{C}_n|=\lceil 2^{nR}\rceil$, whose rate exceeds the capacity of the channel $(R>C)$, the average error probability tends asymptotically to 1.

The proof of this theorem relies on codes chosen at random and decoding schemes based on the maximum-likelihood principle; unfortunately it is not constructive but existential, leaving open the practical problem of finding codes that cleverly combine good efficiency in correcting errors, simple decoding, and a high rate of transmission.

B. Classical error correction

Errors in the storage and processing of information are unavoidable. A classical way of correcting them is by resorting to *redundancy* (repetition codes): each bit is replaced by a string of $n\geqslant 3$ bits equal to it,

$$0\mapsto\underbrace{00\cdots00}_{n\ \text{0's}},\quad 1\mapsto\underbrace{11\cdots11}_{n\ \text{1's}}, \tag{3}$$

and, if by any chance an error occurs in such a way that one of the bits in one of those strings gets reversed (for instance $00000\mapsto 01000$), to correct the error it is enough to invoke the majority vote. Let p be the probability of any bit's getting spoiled. In general, several bits of the n-tuple may be reversed. When $p<\frac{1}{2}$, the probability of the majority rule failing can be made as small as desired, if n is sufficiently large. It is apparent that if the n-tuples of bits are systematically and frequently examined, so that it is very unlikely that errors will occur at two or more bits, then the application of this simple method will clean up the n-tuples and their error-free state will be restored. However, the price paid might be too high, since with codes of length n sufficiently large so as to ensure a small degree of error during the detection, the transmission rate can turn out to be prohibitively slow (in our case it is $1/n$ source bits per channel bit).

So far we have been describing correction codes $\mathcal{C}\subset\{0,1\}^n$ for errors in $\mathcal{E}\subset\{0,1\}^n$. More generally, we can consider q-ary alphabets (whose symbols we shall assume to be the elements of the finite field $\mathbb{F}_q$ with $q =p^f$ elements, p being a prime). Given two words $x,y \in\{0,1,\ldots,q-1\}^n$, let $d_{\mathrm{H}}(x,y)$ be the Hamming distance between them (number of locations in which x,y differ). Let $d:=d_{\mathrm{H}}(\mathcal{C}):=\inf_{x\neq y\in\mathcal{C}}d_{\mathrm{H}}(x,y)$ be the minimum distance of the code. Then the code $\mathcal{C}$ allows correction of errors that affect up to a maximum number $t:=\lfloor\frac{1}{2}(d -1)\rfloor$ of positions:[3] it is enough to replace each received word by the closest code word in the Hamming metric.[4] Therefore the most convenient codes are those with a high d, but this is at the expense of decreasing $|\mathcal{C}|$. If M is the number of code words, we shall call it a $(n,M,d)_q$ code. Its rate is defined as $R:=n^{-1}\log_q M$.

When $\mathcal{C}$ is a linear subspace of $\mathbb{F}_q^n$, the code is called *linear*. The linear codes are of the form $(n,q^k,d)_q$, where k is the dimension of the linear subspace $\mathcal{C}$; for them d coincides with the minimal Hamming length of a nonvanishing code word, and the search for the code word nearest to each received word is greatly simplified. It is customary to represent a linear code as $[n,k,d]_q$, or simply as $[n,k]_q$ when d is irrelevant. Its rate is k/n. Given a code $\mathcal{C}$ of type $[n,k]_q$, the matrix G, $k\times n$, with rows given by the components of the vectors in a basis of $\mathcal{C}$, is called a *generator matrix* for $\mathcal{C}$. Defining now a scalar product in $\mathbb{F}_q^n$ in the canonical way, we can introduce the *dual code* $\mathcal{C}^\perp$ of $\mathcal{C}$. A generator matrix H for $\mathcal{C}^\perp$ is known as a *parity-check matrix* for $\mathcal{C}$. Notice that $\mathcal{C}=\{u \in\mathbb{F}_q^n:Hu=0\}$, which justifies in part the name "parity check" given to H, for it allows us to easily check whether a vector in $\mathbb{F}_q^n$ belongs to the subspace $\mathcal{C}$.

The coding applies bijectively and linearly $\mathbb{F}_q^k$ to a code $\mathcal{C}\subset\mathbb{F}_q^n$ of type $(n,q^k,d)_q$, and it is implemented as follows. Let $\{e_1,\ldots,e_k\}\subset\mathbb{F}_q^n$ be a basis of $\mathcal{C}$. Given a source word $w^{\mathrm{t}}=(w_1,\ldots,w_k)\in\mathbb{F}_q^k$, it gets assigned a code word $c(w):=\Sigma_i w_i e_i$. In terms of the generator matrix, $w^{\mathrm{t}}\mapsto w^{\mathrm{t}}G$. Let us call $\pi:w\mapsto c(w)$ this injection. During the transmission, $c(w)$ could get corrupted, becoming $u:=c(w)+e$, where $e\in\mathcal{E}$ is a possible error vector. It is evident that $e\in u+\mathcal{C}$. In order to decode it, the

[3]Notation: $\lfloor x\rfloor$ is the largest and $\lceil x\rceil$ the smallest integer $\leqslant x$ or $\geqslant x$, respectively.

[4]For instance, for the repetition code $\mathcal{C}=\{0\cdots0,1\cdots1,\ldots,(q -1)\cdots(q-1)\}$, with q code words of length n, we have $d =n$. Thus this code exactly corrects $\lfloor(n-1)/2\rfloor$ errors.

criterion of minimal Hamming distance is applied, replacing u by $\pi^{-1}(u-u_0)$, where u_0 is an element of the coset $u+\mathcal{C}$, which minimizes the distance to the origin (such a u_0 is known as a *leader* of $u+\mathcal{C}$). The linearity of the code allows us to economize in this last step. We make a lookup table containing for each coset $v+\mathcal{C} \in \mathbb{F}_q^n/\mathcal{C}$ its *syndrome* Hv (which uniquely characterizes the coset) and a leader v_0. Upon receiving u as a message, we compute the syndrome Hu and search for its corresponding leader u_0 in the table; next, decoding proceeds as stated earlier (Macwilliams and Sloane, 1977; Roman, 1992; Welsh, 1995). The original message is faithfully retrieved if and only if the error coincides with one of the leaders in the table.

Some of the most relevant linear codes are as follows (Macwilliams and Sloane, 1977; Roman, 1992; Welsh, 1995):

(1) The repetition code $\mathcal{C}=\{0\cdots 0,1\cdots 1,\cdots,(q-1)\cdots(q-1)\}$, of type $[n,1,n]_q$. Although its minimum distance is optimal, its rate is dreadful.

(2) The Hamming codes $\mathrm{H}_q(r)$, arguably the most famous. These are codes of the type $[n=1+q+\cdots+q^{r-1},k=n-r,d=3]_q$, and they are *perfect* in the sense that the set of Hamming spheres with radius $\lfloor (d-1)/2 \rfloor$ and center at each code word fills $\mathbb{F}_q^n$. These codes have rates $R=1-r/n$ that tend to 1 as $n\to\infty$, but they correct only one error.

For instance, $\mathrm{H}_2(3)$ is of type $[7,4,3]_2$ and rate 4/7. A parity-check matrix for this code is

$$H=\begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}. \tag{4}$$

Its decoding is particularly simple. Let u be the word received instead of the code word w, and assume that u has only one corrupted bit. The syndrome $s(u):=Hu$ coincides in this case with the binary expression of the position occupied by the erroneous bit. Negating this single bit will thus suffice to clean up the word and get the correct code word. For example, if $u=0110001$, then $s(u)=110$, so that the incorrect bit is the sixth one, and hence $w=0110011$.

(3) The Golay codes G_{24} and G_{23}. These are binary, of type $[24,12,8]_2$ and $[23,12,8]_2$, respectively. They are probably the most important codes.

The code G_{24} is *self-dual*, i.e., $\mathcal{C}=\mathcal{C}^{\perp}$, which simplifies decoding. Its rate is $R=1/2$ and allows the correction of up to three errors; it was used by NASA in 1972–1982 for the transmission of color images of Jupiter and Saturn from the *Voyager* spacecrafts.

The code G_{23} is perfect and it gives rise to G_{24} when augmented with a parity bit.

The Golay codes G_{12} and G_{11} are ternary, of type $[12,6,6]_3$ and $[11,6,5]_3$, respectively. As before, G_{12} is self-dual, while G_{11} is perfect and produces G_{12} when a parity bit is appended.

The codes G_{24} and G_{12} have very peculiar combinatorial properties; their groups of automorphisms are M_{24} and $2.\mathrm{M}_{12}$, where M_{24} and M_{12} are the famous sporadic groups of Mathieu. This latter group is the subgroup of S_{12} generated by two special permutations of 12 cards labeled from 0 to 11: $0,1,2,\ldots,11\mapsto 11,10,9,\ldots,0$ and $0,1,2,\ldots,11 \mapsto 0,2,4,6,8,10,11,9,7,5,3,1$. It is also the group of motions of the form $\tau_i\tau_j^{-1}$ on a "Rubik" icosahedron, where τ_i indicates a rotation of angle $2\pi/5$ degrees around the ith axis of the icosahedron (Conway and Sloane, 1999). As a matter of fact, it was the discovery of the Golay codes that drove further the study of the sporadic groups, which resulted in the complete classification of the finite simple groups with the discovery by Griess in 1983 of the "monster" or "friendly giant" group, finite and simple, an enormous subgroup of $\mathrm{SO}(47\times 59\times 71)$ with about 10^{54} elements.

(4) The Reed-Muller binary codes $\mathrm{RM}(r,m)$, with $0\leq r\leq m$. These are of the type $[n=2^m,k=\Sigma_{j\leq r}\binom{m}{j},d=2^{m-r}]_2$. Their rates, for fixed r, tend to 0 when increasing m. They rank among the oldest codes known. The code RM(1,5), of type $(32,64,16)_2$, is able to correct up to seven errors with a rate of $R=3/16$. It was used in 1969–1972 to transmit from the *Mariner* spacecrafts the black and white photos of Mars.

(5) The Reed-Solomon codes generalize the Hamming codes. They have been heavily employed by NASA in the transmission of information during the *Galileo, Ulysses*, and *Magellan* missions to deep outer space, and currently they are used everywhere, from CD-ROM's to the hard disks of computers.

(6) The algebraic-geometric Goppa codes $\mathrm{G}_q(D,G)$. These interesting generalizations of the Reed-Solomon codes have led to the discovery of families of codes that are *asymptotically good*, that is, families containing infinite sequences $\{[n_i,k_i,d_i]_q\}$ of codes, with $n_i\to\infty$, such that the sequences $\{k_i/n_i,d_i/n_i\}$ of rates and minimum relative distances are bounded from below by certain positive numbers (Macwilliams and Sloane, 1977; Roman, 1992; Stichtenoth, 1993; Blake *et al.*, 1998).

To obtain good encodings it is advisable to use long codes that not only permit sending many different messages but also have a large minimum distance that allows for correcting sufficiently many errors. Given a code $\mathcal{C}=[n,k,d]_q$, let $R(\mathcal{C}):=k/n$ be its rate and $\delta(\mathcal{C}):=d/n$ its minimum relative distance. A theorem of Manin asserts that the set of limit points of $\{(\delta(\mathcal{C}),R(\mathcal{C}))\in[0,1]^2\}$, where $\mathcal{C}$ is a code on $\mathbb{F}_q$ is of the form $\{(\delta,R)\in[0,1]^2:\delta\in[0,1],0\leq R\leq\alpha_q(\delta)\}$, where $\alpha_q(\delta)$ is a continuous function of $\delta\in[0,1]$, decreasing in $[0,1-q^{-1}]$, such that $\alpha_q(0)=1,\alpha_q(\delta)=0$ if $1-q^{-1}\leq\delta\leq 1$ (Stichtenoth, 1993).

Let H_q be the q-ary entropy function $H_q(x\in[0,1-q^{-1}]):=x\log_q(q-1)-x\log_q x-(1-x)\log_q(1-x)$. The following bounds for the function $\alpha_q(\delta)$ in the relevant

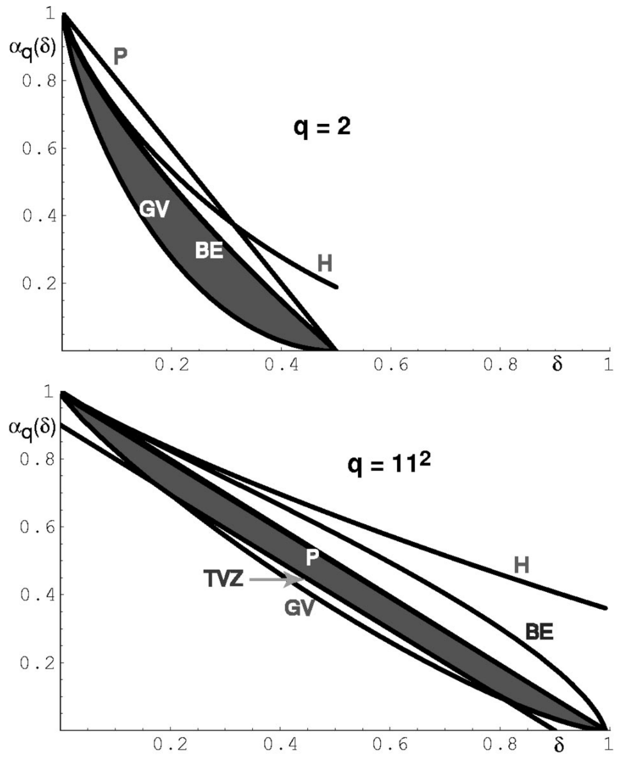


FIG. 1. Asymptotic bounds for $q=2$ (above) and $q=11^2$ (below). The dark zone is limited by the lower and upper bounds given in the text by Eqs. (5)–(9).

interval $\delta\in[0,1-q^{-1}]$ are known (Roman, 1992; Stichtenoth 1993; Blake *et al.*, 1998):

- The Plotkin upper bound:

$$\alpha_q(\delta)\leqslant 1-(1-q^{-1})^{-1}\delta. \tag{5}$$

- The Hamming or sphere-packing upper bound:

$$\alpha_q(\delta)\leqslant 1-H_q(\delta/2). \tag{6}$$

- The Bassaligo-Elias upper bound:

$$\alpha_q(\delta)\leqslant 1-H_q[\theta-\sqrt{\theta(\theta-\delta)}],\ \text{with}\ \ \theta:=(1-q^{-1}). \tag{7}$$

- The Gilbert-Varshamov lower bound:

$$\alpha_q(\delta)\geqslant 1-H_q(\delta). \tag{8}$$

  This last one is very important, since it ensures the existence of codes as long as desired with minimum relative distance δ and rate R, both asymptotically positive.
- The Tsfasman-Vlăduţ-Zink lower bound: if q is a square, then on $[0,1-(\sqrt{q}-1)^{-1}]$ one has

$$\alpha_q(\delta)\geqslant\left(1-\frac{1}{\sqrt{q}-1}\right)-\delta, \tag{9}$$

  which is stronger than the Gilbert-Varshamov bound in some places from $q=7^2$ on.

For an illustration see Fig. 1.

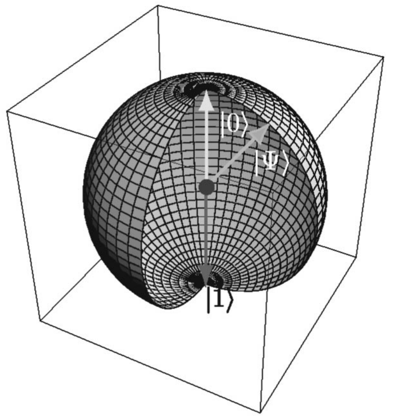


FIG. 2. Parametrization of the states of one qubit: the Bloch sphere.

III. QUANTUM INFORMATION

The quantum information theory, being an extension of the classical theory, is essentially a product of the past decade (Bouwmeester, Ekert, and Zeilinger, 2000; Nielsen and Chuang, 2001).

In quantum information, the analog of the classical bit is the *qubit* or *quantum bit* (Schumacher, 1995). It is a two-dimensional quantum system (for instance, a spin $\frac{1}{2}$, a photon polarization, an atomic system with two relevant states, etc.), with Hilbert space isomorphic to $\mathbb{C}^2$. Besides the two basis states $|0\rangle,|1\rangle$, the system can have infinitely many other (pure) states given by a coherent linear superposition $\alpha|0\rangle+\beta|1\rangle$. The Hilbert space of n qubits is the tensor product $\mathbb{C}^2\otimes\cdots\otimes\mathbb{C}^2=\mathbb{C}^{2^n}$, and its natural basis vectors are $|0\rangle\otimes\cdots\otimes|0\rangle=:|0\cdots 0\rangle$, $|0\rangle\otimes\cdots\otimes|1\rangle=:|0\cdots 1\rangle,\ldots,|1\rangle\otimes\cdots\otimes|1\rangle=:|1\cdots 1\rangle$. For this basis, also known as the computational basis, we shall assume lexicographic ordering. When appropriate, we shall briefly write $|x\rangle$ to denote $|x_{n-1}\cdots x_0\rangle$, with $x:=x_0+2x_1+\cdots+2^{n-1}x_{n-1}$. Thus $|5\rangle=|0\cdots 0101\rangle$.

It is possible to extend two-level qubits to *qudits* or d-dimensional systems ($d\geqslant 2$; Rungta *et al.*, 2001). This leads to an extension of the binary quantum logic. Using d computational levels we can reduce the number n_2 of qubits needed for a computation by a factor of $\lfloor\log_2 d\rfloor$, since the Hilbert space of n_d qudits contains the space of n_2 qubits provided that $d^{n_d}\geqslant 2^{n_2}$.

Given an arbitrary state vector $|\Psi\rangle=c_0|0\rangle+c_1|1\rangle$ of a qubit, the complex coefficients $c_0,c_1\in\mathbb{C}$ amount to four real parameters. However, if we parametrize them as $c_i=r_ie^{i\phi_i}$, $i=0,1$ and factor out a global irrelevant phase, we find $|\Psi\rangle=r_0|0\rangle+r_1e^{i(\phi_1-\phi_0)}|1\rangle$. Imposing $|\Psi\rangle$ to be of unit norm, we can write it as

$$|\psi\rangle=(\cos\tfrac{1}{2}\theta)|0\rangle+e^{i\phi}(\sin\tfrac{1}{2}\theta)|1\rangle, \tag{10}$$

where r_0,r_1 are now parametrized by the angles $\theta,\phi:=\phi_1-\phi_0$.

These two angles represent a point in an S^2 sphere, called the Bloch sphere, as shown in Fig. 2. The (projective) Hilbert space of pure states of a single qubit can be parametrized by the points on this sphere. As a by-product, this construction provides a nice representation of the classical bits as particular points on the sphere. The classical bit 0 (the qubit state $|0\rangle$) marks the north pole and the bit 1 sits at the south pole. Any other point on the sphere amounts to a nontrivial linear superposition of the basis states. The angle θ is related to the proportion of $|1\rangle$ to $|0\rangle$ in the composition of that state, while the angle ϕ is their relative quantum phase.

It is immediately clear from Fig. 2 that the information contained in a qubit is infinite as compared with the information in a classical bit. In other words, at a given time, a classical bit can take on only one of the two values, either 0 or 1, while a qubit can be in any of the infinitely many possible quantum states in Eq. (10). As we shall see later in detail, this fact is basic to what is known as "quantum parallelism," a source of the unprecedented capabilities exhibited by a quantum computer.

A *quantum logic gate*[5] acting on a collection or *quantum register* of k qubits is just any unitary operator in the associated Hilbert space $\mathbb{C}^{2^k}$ (Deutsch, 1989). For instance, in addition to the identity, we have for 1 qubit the unary gates X (or U_{NOT}), Y, Z, given by the Pauli matrices σ_a (in the natural basis $\{|0\rangle,|1\rangle\}$):

$$U_{\mathrm{NOT}}:=X:=\sigma_x,\quad Y:=-i\sigma_y,\quad Z:=\sigma_z. \tag{11}$$

The particular linear combination $U_{\mathrm{H}}:=2^{-1/2}(X+Z)$ is the important *Hadamard gate*.

The unary gates are easy to implement (for instance, on polarized photons, with $\frac{1}{2}\lambda,\frac{1}{4}\lambda$ plates).

On 2 qubits, the most important gate is the *controlled* NOT (U_{CNOT}), or *exclusive* OR (U_{XOR}), gate defined by $U_{\mathrm{CNOT}},U_{\mathrm{XOR}}:|x\rangle|y\rangle\mapsto|x\rangle|x\oplus y\rangle$, where x,y are either 0 or 1, and $\oplus$ means addition mod 2. This gate can be represented by the matrix

$$\begin{aligned}U_{\mathrm{CNOT}}:=U_{\mathrm{XOR}}&:=|0\rangle\langle 0|\otimes 1+|1\rangle\langle 1|\otimes U_{\mathrm{NOT}}\\&=\tfrac{1}{2}(1+\sigma_z)\otimes 1+\tfrac{1}{2}(1-\sigma_z)\otimes\sigma_x.\end{aligned} \tag{12}$$

The physical implementation of this gate is central to the applications of quantum information and will be addressed later in Sec. XI.

The quantum partner of the Shannon entropy is the von Neumann entropy,

$$S(\rho):=-\mathrm{Tr}(\rho\log_2\rho), \tag{13}$$

where ρ is the density operator describing a normal quantum state. Given a convex decomposition $\rho=\Sigma_{i\in I}p_i|\phi_i\rangle\langle\phi_i|$ in pure states, it can be shown that $S(\rho)\le H(I):=-\Sigma_i p_i\log_2 p_i$, equality holding if and only if the state vectors ϕ_i are pairwise orthogonal. The von Neumann entropy has the well-known properties of concavity, strong subadditivity, and triangularity (Thirring, 1983; Galindo and Pascual, 1989, 1990a):

$$\begin{aligned}&\lambda_1 S(\rho_1)+\lambda_2 S(\rho_2)\le S(\lambda_1\rho_1+\lambda_2\rho_2),\\&S(\rho_{ABC})+S(\rho_B)\le S(\rho_{AB})+S(\rho_{BC}),\\&|S(\rho_A)-S(\rho_B)|\le S(\rho_{AB})\le S(\rho_A)+S(\rho_B),\end{aligned} \tag{14}$$

with $\lambda_{1,2}\ge 0$, $\lambda_1+\lambda_2=1$. The subscripts A,B,C denote subsystems.

The first two relations also hold in the classical theory of information. But the third property (whose second part is just the property of simple subadditivity) is peculiar. While in Shannon's theory the entropy of a composite system can never lower the entropy of any of its parts, quantumly this is not the case. The Einstein-Podolsky-Rosen (EPR) states of the form $2^{-1/2}(|aa'\rangle+|bb'\rangle)$,[6] where a,b and a',b' are given orthonormal pairs, provide us with an explicit counterexample.

A basic difference between classical and quantum information is that while classical information can be copied perfectly, quantum cannot. This is relevant to quantum communication protocols because, should a quantum copier exist, then safe eavesdropping of quantum channels would be possible. In particular, we cannot create a duplicate of a quantum bit in an unknown state without uncontrollably perturbing the original. This follows from the no-cloning theorem of Wootters and Zurek (1982). Let $\mathcal{H}:=\mathcal{H}_{\mathrm{orig}}\otimes\mathcal{H}_{\mathrm{copy}}$ be the joint Hilbert space of the original and of the copy, and let U_{QCM} be the linear (unitary) operator in $\mathcal{H}$ representing the action of an alleged quantum copier machine:

$$U_{\mathrm{QCM}}:|\Psi\rangle_{\mathrm{orig}}|\phi_0\rangle\mapsto|\Psi\rangle_{\mathrm{orig}}|\Psi\rangle_{\mathrm{copy}},\ \forall|\Psi\rangle\in\mathcal{H}_{\mathrm{orig}}, \tag{15}$$

where $|\phi_0\rangle$ is the "blank" state of the copy.

We claim that such a machine cannot exist. This is a remarkably simple application of the linearity of quantum mechanics. For a contradiction, suppose it does exist. Assume for simplicity that the object to be copied is just a single qubit, and let $|\Psi\rangle_{\mathrm{orig}}=\alpha_0|0\rangle+\alpha_1|1\rangle$. Then linearity implies

$$U_{\mathrm{QCM}}|\Psi\rangle|\phi_0\rangle=\alpha_0|0\rangle|0\rangle+\alpha_1|1\rangle|1\rangle, \tag{16}$$

whereas the definition of a quantum copier yields

$$\begin{aligned}U_{\mathrm{QCM}}|\Psi\rangle|\phi_0\rangle&=|\Psi\rangle|\Psi\rangle\\&=\alpha_0^2|0\rangle|0\rangle+\alpha_0\alpha_1|0\rangle|1\rangle+\alpha_1\alpha_0|1\rangle|0\rangle\\&\quad+\alpha_1^2|1\rangle|1\rangle.\end{aligned} \tag{17}$$

The results, Eqs. (16) and (17), are in general incompatible, which proves the assertion.

A more general proof of the no-cloning theorem takes into account the environment and makes use of the unitarity of U_{QCM}: now $\mathcal{H}:=\mathcal{H}_{\mathrm{orig}}\otimes\mathcal{H}_{\mathrm{copy}}\otimes\mathcal{H}_{\mathrm{env}}$, and

$$\begin{aligned}&U_{\mathrm{QCM}}|\Psi\rangle_{\mathrm{orig}}|\phi_0\rangle|E_0\rangle\\&\quad=|\Psi\rangle_{\mathrm{orig}}|\Psi\rangle_{\mathrm{copy}}|E_\Psi\rangle,\ \forall|\Psi\rangle\in\mathcal{H}_{\mathrm{orig}},\end{aligned} \tag{18}$$

[5]A more extended study of quantum logic gates and their classical counterparts is presented in Secs. VIII.D and IX.B.

[6]Actually they are EPR states *à la* Bohm, that is, Einstein-Podolsky-Rosen-Bohm states (Bohm, 1951).

where $|E_0\rangle$ is the "rest state" of the "remaning world" (environment) before copying, and $|E_\Psi\rangle$ its state after copying. Let us consider two actions of the quantum copier machine,

$$\begin{aligned} U_{\rm QCM}|\Psi_1\rangle|\phi_0\rangle|E_0\rangle &= |\Psi_1\rangle|\Psi_1\rangle|E_{\Psi_1}\rangle, \\ U_{\rm QCM}|\Psi_2\rangle|\phi_0\rangle|E_0\rangle &= |\Psi_2\rangle|\Psi_2\rangle|E_{\Psi_2}\rangle. \end{aligned} \tag{19}$$

Taking the scalar product of these two actions and using unitarity yields $\langle\Psi_1|\Psi_2\rangle = \langle\Psi_1|\Psi_2\rangle^2\langle E_{\Psi_1}|E_{\Psi_2}\rangle$. Therefore, since all these probability amplitudes have modulus $\leqslant 1$, either $\langle\Psi_1|\Psi_2\rangle = 0$ or it equals 1, and hence copying two different and nonorthogonal states Ψ_1,Ψ_2 is impossible.

However, a known quantum state can be copied at will. Moreover, if one drops the requirement that copies be perfect, approximate quantum copier machines may exist (Buzek and Hillery, 1996). Should it be possible to make close to perfect copies then quantum cryptographic schemes might still be at risk. Quantum copying can also become essential in the storage and retrieval of information in quantum computers.

A. Entanglement and information

A quantum pure state $|\Psi\rangle$ in a Hilbert space $\mathcal{H} = \otimes_{i=1}^n \mathcal{H}_i$ of n qubits is said to be *separable* (with respect to the factor spaces $\{\mathcal{H}_1,\mathcal{H}_2,\ldots,\mathcal{H}_n\}$) when it can be factorized as follows:

$$|\Psi\rangle = \bigotimes_{i=1}^{n} |\psi_i\rangle, \quad |\psi_i\rangle \in \mathcal{H}_i. \tag{20}$$

Otherwise the state $|\Psi\rangle$ is called *entangled*. Famous examples of entangled states include the Einstein-Podolsky-Rosen pairs or Bell states like

$$\begin{aligned} |\Psi^\pm\rangle &:= \frac{1}{\sqrt{2}}[|01\rangle \pm |10\rangle], \\ |\Phi^\pm\rangle &:= \frac{1}{\sqrt{2}}[|00\rangle \pm |11\rangle], \end{aligned} \tag{21}$$

which may be physically represented by a spin-$\frac{1}{2}$ singlet and triplet or by entangled polarized (vertical and horizontal) photons (Kwiat *et al.*, 1995). They also include the GHZ state (Greenberger, Horne, and Zeilinger, 1989),

$$|{\rm GHZ}\rangle := \frac{1}{\sqrt{2}}[|000\rangle + |111\rangle], \tag{22}$$

which has been observed experimentally in polarization entanglement of three spatially separated photons (Bouwmeester *et al.*, 1999).

The concept of entanglement is the distinctive feature that allows quantum information to overcome some of the limitations posed by classical information, as exemplified by the new notions of teleportation, dense coding, etc., to be explained in the following sections. Although it is simple to state mathematically, entanglement leads to profound experimental consequences like nonlocal correlations: when two distant parties A (Alice) and B (Bob) share, say, an EPR pair,[7] the measurement by A of her state univocally determines the state on the B side. Apparently this implies instant transmission of information, in sharp contrast with Einstein's relativity. However, to reconcile both facts we must notice that the only way the B side can know about his state (without measuring it) is by receiving a classical communication from the A side, which propagates no faster than the speed of light.

For these basic reasons, entanglement is considered as a resource in quantum information (Bennett, 1998), something that we must have available if we want to take advantage of the new communication possibilities exhibited by quantum protocols.

When the system has two parts, namely, $\mathcal{H} := \mathcal{H}_{\rm A} \otimes \mathcal{H}_{\rm B}$, it is called *bipartite*. In general, a *multipartite* system is of the form $\mathcal{H} := \otimes_{i=1}^n \mathcal{H}_i$. We may think of entanglement as a manifestation of the superposition principle when applied to bipartite or multipartite systems. Thus genuine multiparticle or many-body states exhibit entanglement properties, which in the theory of strongly correlated systems are known as quantum correlations (Fulde, 1993).[8] We may state that entanglement and quantum correlations are closely linked.

Being a nonlocal concept, entanglement must be independent of local manipulations performed on each of the A and B parties. These operations are represented by unitary operators $U_{\rm A} \otimes U_{\rm B}$, in a factorized form, acting on the states of $\mathcal{H} = \mathcal{H}_{\rm A} \otimes \mathcal{H}_{\rm B}$, or they may be local measurements on either side. Moreover, classical communication is also permitted by the two parties. Entanglement cannot be created by these local operations. However, factorized states can be obtained by local operations, like measurements. Altogether, these types of local operations plus classical communications are known as LOCC transformations. The set LOCC is not a group but a semigroup, for the inverse of a given transformation is not guaranteed to exist, due to possible irreversible measurements by each party.

The characterization of entanglement for general quantum states (pure or mixed, bipartite or multipartite) is very difficult, due in part to the type of transformations allowed in the set LOCC. For entangled pure states of two qubits or general bipartite systems A and B with dimensions $d_{\rm A}, d_{\rm B}$, respectively, entanglement is well understood in terms of the Schmidt (1906) decomposition: given an arbitrary state

$$|\Psi\rangle_{\rm AB} := \sum_{i=1}^{d_{\rm A}} \sum_{j=1}^{d_{\rm B}} C_{ij} |a_i\rangle_{\rm A} |b_j\rangle_{\rm B} \in \mathcal{H} = \mathcal{H}_{\rm A} \otimes \mathcal{H}_{\rm B} \tag{23}$$

[7]It is usual in information theory to introduce a set of characters named Alice (the sender), Bob (the recipient), and Eve (the eavesdropper).

[8]These types of correlations are responsible for novel quantum phase transitions (Sachdev, 1999) in which the transition is driven by quantum fluctuations instead of standard thermal fluctuations.

with $\{|a_i\rangle_{\rm A}\}_1^{d_{\rm A}}, \{|b_i\rangle_{\rm B}\}_1^{d_{\rm B}}$ orthonormal bases of $\mathcal{H}_{\rm A}, \mathcal{H}_{\rm B}$, the state admits a biorthonormal decomposition of the form

$$|\Psi\rangle_{\rm AB} = \sum_{k=1}^{r} \sqrt{w_k}|u_k\rangle_{\rm A}|v_k\rangle_{\rm B}, \quad w_k > 0, \quad \sum_{k=1}^{r} w_k = 1, \tag{24}$$

where $\{|u_k\rangle_{\rm A}\}_1^r$ and $\{|v_k\rangle_{\rm B}\}_1^r$ are sets of orthonormal vectors for subsystems A and B, and $r \leqslant d := \min\{d_{\rm A}, d_{\rm B}\}$ is the so-called *Schmidt rank* of $|\Psi\rangle_{\rm AB}$ (Schmidt, 1906; Hughston, Jozsa, and Wootters, 1993; Ekert and Knight, 1995).[9] The coefficients w_k are called *Schmidt weights*.

The Schmidt decomposition is essentially unique in the following sense: the weights (multiplicities included) are unique (up to order), and hence the rank; given a nondegenerate weight w_k, the state vectors $|u_k\rangle_{\rm A}, |v_k\rangle_{\rm B}$ are unique up to reciprocal phase factors. When the weight w_k is degenerate, the corresponding states on Alice's side are unique up to an arbitrary unitary transformation $U_{\rm A}$ to be compensated by a simultaneous unitary transformation $U_{\rm B} = U_{\rm A}^*$ on the associated vectors on Bob's side.

From the Schmidt decomposition it immediately follows that a bipartite pure state $|\Psi\rangle_{\rm AB}$ is entangled if and only if its Schmidt rank r is greater than 1.

From the point of view of the subsystem A, the description of its quantum properties is realized by means of the reduced density matrix $\rho_{\rm A}$ (and likewise for subsystem B with $\rho_{\rm B}$):

$$\begin{aligned} \rho_{\rm A} &:= {\rm Tr}_{\rm B}|\Psi\rangle_{\rm AB}\langle\Psi|, \\ \rho_{\rm B} &:= {\rm Tr}_{\rm A}|\Psi\rangle_{\rm AB}\langle\Psi|, \end{aligned} \tag{25}$$

where ${\rm Tr}_{\rm B}$ denotes the partial trace over the B subsystem (similarly for ${\rm Tr}_{\rm A}$ and subsystem B). The Schmidt decomposition (24) implies that

$$\begin{aligned} \rho_{\rm A} &= \sum_{k=1}^{r} w_k |u_k\rangle_{\rm A}\langle u_k|, \\ \rho_{\rm B} &= \sum_{k=1}^{r} w_k |v_k\rangle_{\rm B}\langle v_k|. \end{aligned} \tag{26}$$

Another important implication of (24) is that as $r \leqslant d$, when a qubit state $d_{\rm A} = 2$ is entangled with a qudit state $d_{\rm B} \geqslant 2$ then the Schmidt decomposition has at most two terms, no matter how large $d_{\rm B}$ is.

Interestingly enough, the Schmidt decomposition has appeared independently in the field of strongly correlated systems through the density-matrix renormalization-group method (White, 1992, 1993).[10]

Once we know whether or not a given bipartite pure state is entangled, the next task is to get entanglement ordered: given two states $|\Psi_1\rangle_{\rm AB}, |\Psi_2\rangle_{\rm AB}$, which one is more entangled? No sufficiently general answer to this question is known. A tentative simple choice would be to measure entanglement through the partial von Neumann entropies (Bennett, Brassard, *et al.*, 1996):

$$E(|\Psi_{\rm AB}\rangle) := S(\rho_{\rm A}) = S(\rho_{\rm B}). \tag{27}$$

Such entropies do not increase under LOCC, but having $E(|\Phi_{\rm AB}\rangle) < E(|\Psi_{\rm AB}\rangle)$ does not guarantee that an LOCC action may bring $|\Psi_{\rm AB}\rangle$ to $|\Phi_{\rm AB}\rangle$.

The *theory of majorization* provides us with a criterion to ascertain when any two entangled states can be LOCC connected (Nielsen, 1999). Given two vectors $x = (x_1, x_2, \ldots, x_d)$, $y = (y_1, y_2, \ldots, y_d)$ in $\mathbb{R}^d$, decreasingly ordered $x_1 \geqslant x_2 \geqslant \cdots \geqslant x_d, y_1 \geqslant y_2 \geqslant \cdots \geqslant y_d$, we say that x is *majorized* by y, denoted $x \prec y$ (equivalently, y majorizes x) if the following series of relations holds true:

$$\begin{aligned} & x_1 \leqslant y_1, \\ & x_1 + x_2 \leqslant y_1 + y_2, \\ & \vdots \\ & x_1 + x_2 + \cdots + x_{d-1} \leqslant y_1 + y_2 + \cdots + y_{d-1}, \\ & x_1 + x_2 + \cdots + x_d = y_1 + y_2 + \cdots + y_d. \end{aligned} \tag{28}$$

The majorization relation is a partial order in $\mathbb{R}^d$: 1/ $x \prec x$, $\forall x$; 2/ $x \prec y$ and $y \prec x$ if and only if $x = y$; 3/ if $x \prec y$ and $y \prec z$ then $x \prec z$. When the components of the vector x are positive, $x_k \geqslant 0$, and normalized, $\Sigma_k x_k = 1$, they may be thought of as probability distributions as in Sec. II. The central result is the following: a bipartite state $|\Psi\rangle_{AB}$ can be transformed via LOCC operations into another state $|\Phi\rangle_{AB}$ if and only if $w(|\Psi\rangle)$ is majorized by $w(|\Phi\rangle)$,

$$|\Psi\rangle_{AB} \rightarrow |\Phi\rangle_{AB} \Leftrightarrow w(|\Psi\rangle) \prec w(|\Phi\rangle), \tag{29}$$

where $w(|\Psi\rangle)$ is the ordered vector of eigenvalues or weights (multiplicities included) of the reduced density matrix $\rho_{\rm A}$ (25) and (26) associated with $|\Psi\rangle_{AB}$ [similarly for $w(|\Phi\rangle)$].

For example, let us consider the parties A and B sharing this pair of qutrit states in the basis $\{|0\rangle, |1\rangle, |2\rangle\}$:

$$\begin{aligned} |\Psi\rangle_{AB} &= \frac{2}{3}|00\rangle + \frac{2}{3}|11\rangle + \frac{1}{3}|22\rangle, \\ |\Phi\rangle_{AB} &= \sqrt{\frac{2}{3}}|00\rangle + \sqrt{\frac{1}{6}}|11\rangle + \sqrt{\frac{1}{6}}|22\rangle. \end{aligned} \tag{30}$$

[9]The Schmidt decomposition is equivalent to the singular value decomposition of the $d_{\rm A} \times d_{\rm B}$ matrix $C := (C_{ij})$ in linear algebra (Press *et al.*, 1992). Let $d_{\rm A} \leqslant d_{\rm B}$. Then $C = UDV^{\rm t}$, where U is an orthogonal $d_{\rm A} \times d_{\rm A}$ matrix ($U^{\rm t}U = 1_{d_{\rm A}}$), V is a $d_{\rm A} \times d_{\rm B}$ matrix representing a Euclidean isometry from $\mathbb{C}^{d_{\rm A}}$ to $\mathbb{C}^{d_{\rm B}}$ (i.e., $VV^{\rm t} = 1_{d_{\rm A}}$), and D is the $d_{\rm A} \times d_{\rm A}$ diagonal matrix ${\rm diag}(\sqrt{w_1}, \ldots, \sqrt{w_r}, 0, \ldots, 0)$. Using the singular value decomposition $C_{ij} = \Sigma_{k=1}^{d_{\rm A}} U_{ik}\sqrt{w_k}V_{jk}$ in Eq. (23), we inmediately arrive at the Schmidt decomposition, Eq. (24).

[10]The Schmidt weights govern the truncation process inherent to the density-matrix renormalization-group method: the highest weights are retained while the smallest (beyond a certain desired value) are eliminated. This truncation makes an exponentially large problem much more tractable.

Both states are entangled, but $|\Psi\rangle_{AB}$ cannot be transformed into $|\Phi\rangle_{AB}$ or vice versa: they possess different types of entanglement. They are said to be *incomparable* or *incommensurate* (Nielsen, 1999; Vidal, 1999).

However, for general multipartite systems the issue of how to relate the LOCC action with entanglement in a given pure state is an open question (Lewenstein *et al.*, 2000).

A definition of entanglement for finite-dimensional systems with mixed states characterized by a density matrix ρ goes as follows (Werner, 1989): ρ is called separable when it can be written as a convex combination of product states,

$$\rho=\sum_{k=1}^{r}\lambda_k \bigotimes_{j=1}^{n}\rho_k^{(j)},\quad \lambda_k\geqslant 0,\quad \sum_k \lambda_k=1. \tag{31}$$

When ρ is not separable, it is called an entangled mixed state. The situation for quantifying and qualifying entanglement is even worse for mixed quantum states (Horodecki *et al.*, 1996a; Peres, 1996; Dür, Cirac, and Tarrach, 1999; Giedke *et al.*, 2001). There are partial characterizations of entanglement like the Peres criterion (1996): a necessary condition for separability of ρ is that the matrices $\rho^{\mathrm{t},j}$, $j=1,\ldots,r$, obtained by partial transposition[11] of ρ with respect to an arbitrary orthonormal basis of the factor space $\mathcal{H}_j$ of the j component, be non-negative ($\rho^{\mathrm{t},j}\geqslant 0$). The converse is true in the special cases $\mathbb{C}^2\otimes\mathbb{C}^2$ and $\mathbb{C}^2\otimes\mathbb{C}^3$ (Horodecki *et al.*, 1996b).

There are also complete characterizations of entanglement in terms of entanglement witness operators and positive maps (Horodecki *et al.*, 1996a), but their classification turns out to be as complicated as the original problem of entangled mixed states.

B. Quantum coding and Schumacher's theorem

Let $A:=\{|\phi_i\rangle,p_i\}_{i=1}^{|A|}$ be a "quantum alphabet" consisting of a set of distinct pure states (not necessarily orthogonal) and their corresponding probabilities ($\Sigma_i p_i=1$). We assign to it the density operator $\rho(A):=\Sigma_i p_i|\phi_i\rangle\langle\phi_i|$. A message emitted by a source of quantum signals is now a sequence $\phi_{i_1\cdots i_n}:=|\phi_{i_1}\rangle|\phi_{i_2}\rangle\cdots|\phi_{i_n}\rangle$ of "quantum characters" or "quantum symbols," each produced with probability p_{i_j} independently of the others. The collection of messages with n symbols is representable by the density operator $\rho^{\otimes n}$, which lives in a Hilbert space of maximum dimension $|A|^n=2^{n\log_2|A|}$. The question naturally arises whether it is possible to compress the information contained in $\rho^{\otimes n}$. And the answer, found by Schumacher (1995), is similar to Shannon's first theorem: asymptotically ($n\gg 1$) the state $\rho^{\otimes n}$ is compressible to a state in a Hilbert space of dimension $2^{nS(\rho)}$, with a *fidelity* F (the probability that the decoded state coincides with the state prior to coding) arbitrarily close to 1. In other words, it is compressible to $nS(\rho)$ qubits. Then $S(\rho)$ can be thought of as the average number of qubits of essential quantum information, per character of the alphabet.

The idea of the proof follows the same guideline as for the classical theorem (Jozsa and Schumacher, 1994; Schumacher, 1995; Preskill, 1998). Let us diagonalize $\rho=\Sigma_r\lambda_r|r\rangle\langle r|$. The von Neumann entropy $S(\rho)$ clearly coincides with the Shannon entropy $H(D)$ of the classical alphabet $D:=\{r,\lambda_r\}_{r=1}^{|D|}$. Introducing the typical messages as those strings or tensor-product vectors $\psi_{i_1\ldots i_n}:=|\psi_{i_1}\rangle\cdots|\psi_{i_n}\rangle$ in the orthonormal basis that diagonalizes ρ, such that its probability $\lambda_{i_1\cdots i_n}:=\Pi_j\lambda_{i_j}$ satisfies $\lambda_{i_1\cdots i_n}\sim 2^{-nH(D)}$ for $n\gg 1$, it is shown that $\rho^{\otimes n}$ is asymptotically concentrated on the typical subspace T spanned by them: $\mathrm{Tr}(P_T\rho^{\otimes n})\sim 1$. Here P_T is the orthogonal projection onto T. The strategy of compression amounts to making a measurement that projects the original message $\phi_{i_1\cdots i_n}$ onto either T or $T^{\perp}$. If the former is the case, the projected state $P_T\phi_{i_1\cdots i_n}$ is faithfully sent, upon coding it into $nH(D)$ qubits. What one does in the remaining case is irrelevant, for the probability that the result will be $(1-P_T)\phi_{i_1\cdots i_n}$ is asymptotically negligible.

The average fidelity in this procedure is perfect in the limit $n\to\infty$, and as in the classical theory, the quantum compression thus obtained is optimal.

If the alphabet $A:=\{\rho_i,p_i\}_{i=1}^{|A|}$ is made up of mixed states, the issue of message compressibility gets more involved. To properly measure it, the Shannon entropy $S(\rho:=\Sigma_i p_i\rho_i)$ must yield to another more general concept, the so-called *Holevo information* of the alphabet or ensemble $A:=\{\rho_i,p_i\}_{i=1}^{|A|}$ (Levitin, 1969; Holevo, 1973; Preskill, 1998):

$$\chi(A):=S(\rho)-\sum_i p_i S(\rho_i). \tag{32}$$

The Holevo information is similar to the classical mutual information. As $I(X:Y)$ measures how the entropy of X gets reduced when Y is known, $\chi(A)$ represents the reduction of the entropy $S(\rho)$ of ρ when the actual preparation of this state as a convex combination $\rho=\Sigma_i p_i\rho_i$ is known.

Assuming the states ρ_i of the alphabet to be mutually orthogonal, that is, $\mathrm{Tr}(\rho_i\rho_j)=0$ for $i\neq j$, it is not difficult to see that the state $\rho^{\otimes n}$ is asymptotically ($n\gg 1$) compressible to a state of $n\chi(A)$ qubits, with fidelity tending to 1. Moreover, this result is optimal.

When the states are not orthogonal, the results are only partial: it is known that there is no asymptotically faithful compression below $\chi(A)$ per letter of the alphabet, but the problem is still open of whether or not a compression of $\chi(A)$ qubits/character is accessible in the limit $n\to\infty$.

[11]Note that $\rho^{\mathrm{t},j}:=\Sigma_{k=1}^{r}\lambda_k\rho_k^{(1)}\otimes\cdots\otimes\rho_k^{(j),\mathrm{t}}\otimes\cdots\otimes\rho_k^{(n)}\geqslant 0$, since the coefficients and each factor matrix are non-negative, no matter which basis is chosen in $\mathcal{H}_j$ to define the transpose.

C. Capacities of a quantum channel

The capacities of a quantum transmission channel include its capacity C for transmitting classical data, its capacity Q for transmitting quantum states exactly, and its mixed capacities $Q_{1,2}$ for transmitting quantum states, also exactly, but with the assistance of a classical side channel between sender and receiver.

Given a quantum channel $\mathcal{N}$, usually noisy, Shannon's second theorem suggests defining the classical capacity $C(\mathcal{N})$ as the supremum of the transmission rates $R := k/n$ of classical words k cbits long such that (1) transmission is carried out after an appropriate word coding as n-bit words that are sent by n forward uses of the channel $\mathcal{N}$, followed by an associated decoding upon arrival (yielding words of k bits); (2) the fidelity of the transmission is asymptotically 1. The quantum capacity $Q(\mathcal{N})$ is similarly defined by replacing the classical input/output words of k cbits by pure/mixed states of k qubits (Bennett and Shor, 1998).

The assisted quantum capacities $Q_{1,2}(\mathcal{N})$ are defined in a similar fashion as $Q(\mathcal{N})$, but now the coding-decoding protocol may include arbitrary local operations on input and output states and may resort to a classical communication channel in the input-to-output direction (subscript 1) or in both directions (subscript 2).

It is possible to show that $Q = Q_1$ (Bennett, DiVincenzo, *et al.*, 1996; Bennett and Shor, 1998); that is, sending classical messages from origin to destination does not increase the channel capacity. On the other hand, it is evident that $Q \leqslant Q_2$, and using orthogonal states to transmit cbits leads to $Q \leqslant C$. But it is not known whether or not $C < Q_2$ holds. Channels are known for which $Q < Q_2$, and others for which $Q_2 < C$.

It is not surprising that the computation of these capacities, as asymptotically defined, is usually difficult. In some instances they are known, as in the case of the *quantum erasure channel*, in which there is a probability p that the channel replaces the qubit by an erasure symbol orthogonal to the states $\{|0\rangle, |1\rangle\}$, and there is also the complementary probability $1-p$ that the qubit goes through exactly. For this type of channel $C = Q_2 = 1 - p$, and $Q = \max\{0, 1-2p\}$ (Bennett, DiVincenzo, and Smolin, 1997; Bennett and Shor, 1998).

Unlike the classical case, in which the capacity can be computed by maximizing the mutual information between input and output in a single use of the channel, the capacities (whether classical or quantum) of quantum channels do not usually allow for a similar computation. This is because in the quantum case it is permissible to code by entangling several successive states on input, and to decode by means of joint measurements on several states on output. However, for the case C_{cq} (classical capacity with classical encoding and quantum decoding), it is known that $C_{\mathrm{cq}}(\mathcal{N}) = \sup_\rho \chi[\mathcal{N}(\rho)]$ (Bennett and Shor, 1998).

Finally, prior entanglement between sender and receiver improves the transmission capacity. Let $C_{\mathrm{E}}, Q_{\mathrm{E}}$ be the classical and quantum entanglement-assisted capacities of a quantum channel. A direct consequence of dense coding and quantum teleportation, to be discussed later, is the relation $C_{\mathrm{E}} = 2C$ for noiseless quantum channels, and the relation $Q \leqslant Q_{\mathrm{E}} = \frac{1}{2} C_{\mathrm{E}}$ for any quantum channel (Bennett *et al.*, 1999).

D. Quantum error correction

It is not possible in the quantum case simply to imitate the classical methods of error correction, for merely trying to check which qubits have been affected by errors irremediably damages the information content. Nor can we make strings of equal quantum states, for the unitarity of quantum mechanics forbids the cloning of arbitrary unknown quantum states. This explains the initial pessimism about the possible functioning of a quantum computer (Unruh, 1995; Landauer, 1997). Then, what to do? Fortunately, in 1995 Shor provided us with a first solution showing an encoding system (of 9:1 bits) capable of detecting and correcting one erroneous qubit.[12] Soon after, new and more economical codes were discovered, such as the 7:1 code of Steane (1996a, 1996b) and Calderbank and Shor (1996), and the 5:1 code of Bennett, Brassard, *et al.* (1996) and Bennett, DiVincenzo, *et al.* (1996).[13] It is not possible to present here a full account of the many remarkable contributions in this field during the last seven years. It is currently a developing field that, as happened with the classical error correction codes, has also found unexpected connections with pure mathematics (Shor and Sloane, 1998).

The underlying idea of quantum error correction is to hide the information in subspaces of $\mathbb{C}^{2^n}$ in order to protect it against decoherence and errors that affect only a few qubits. To this end, if our system has k qubits (called "logical qubits"), a quantum error-correction code encodes their states by means of a linear isometric embedding $\pi: \mathbb{C}^{2^k} \hookrightarrow \mathbb{C}^{2^n}$, with $n > k$. We shall denote by $\mathcal{Q}$ the image subspace of π, and its states will be called code states (or code words). The additional $n-k$ qubits help us in protecting the information. The map π should disguise the information by delocalizing it, with the aim that errors (which often affect just one or a few qubits locally) may alter it not at all or as little as possible (Aharonov, 1998; Preskill, 1998; Steane, 1998).

A system of n qubits in an initial pure state ψ is not absolutely isolated. Upon interaction with the environment in a state a_{in}, it suffers a transformation of the form $\psi \otimes a_{\mathrm{in}} \mapsto \Sigma_r (E_r \psi) \otimes a_r$, where the operators E_r, $0 \leqslant r \leqslant 2^{2n} - 1$ are Pauli operators (elements of the set

[12]Actually, the very first idea of quantum error correction, at the time called "recoherence," was proposed by Deutsch in 1993 during his talk at the Rank Prize Funds Symposium on Quantum Communication and Cryptography. This idea was later developed further (Berthiaume, Deutsch, and Jozsa, 1994; Barenco *et al.*, 1997). Even the idea of decoherence-free subspaces (Palma, Suominen, and Ekert, 1996) preceded Shor's nine-qubit code.

[13]An n:1 code embeds 1 qubit into the space of n qubits.

$\mathcal{P}^{(n)} := \{1, X, Y, Z\}^{\otimes n}$) and the environment states a_r are not necessarily orthogonal or normalized. Let us call the *weight* of an element in $\mathcal{P}^{(n)}$ the number of its nontrivial (i.e., X, Y, Z) tensor factors. If ψ is a code state, then each term $(E_r\psi)\otimes a_r$ represents a component with a number of errors equal to the weight of E_r.

Given a collection of errors $\mathcal{E}\subset\mathcal{P}^{(n)}$ formed by all the Pauli operators of weight $\leqslant t$, a quantum error-correction code is said to amend up to t errors when it is capable of correcting every error in $\mathcal{E}$. For that to happen it is necessary and sufficient that $\langle\overline{j}|E_s^\dagger E_r|\overline{i}\rangle = m_{sr}\delta_{ji}$ be fulfilled, for any arbitrary orthonormal basis $\{|\overline{i}\rangle\}$ of the code subspace $\mathcal{Q}$ and all $E_r, E_s\in\mathcal{E}$, where m is a self-adjoint matrix. This condition means something quite natural: first, that given any two orthogonal code words $|\overline{i}\rangle, |\overline{j}\rangle$, the sets $\mathcal{E}_r|\overline{i}\rangle$, $\mathcal{E}_r|\overline{j}\rangle$ of corrupted code words must be mutually orthogonal, otherwise the perfect distinguishability of those words might get lost, and second, should $\langle\overline{i}|E_s^\dagger E_r|\overline{i}\rangle$ depend on $|\overline{i}\rangle$, the detection of the error would yield information about the code state, thereby perturbing it. If $m=\mathrm{id}$, the code is called nondegenerate, and the error subspaces $E_r\mathcal{Q}, 1\neq E_r\in\mathcal{E}$ are orthogonal to the code subspace $\mathcal{Q}$ and perpendicular to one another. In this case it suffices to make a measurement, which is possible because of the orthogonality, that determines in which subspace the (n-qubit system)$\otimes$environment lies. If the result of that measurement is $(E_r\psi)\otimes a_r$, by applying to the resulting state of the system the unitary operator $E_r^\dagger$ we shall retrieve the original state ψ free of error. In the degenerate case, an error syndrome does not singularize the error, and the retrieval strategy gets more involved.

The *distance* d of a quantum error-correction code is defined as the lowest weight of a Pauli operator E such that $\langle\overline{j}|E|\overline{i}\rangle\neq c_E\delta_{ji}$. In analogy with the notation for classical error-correcting codes, we shall write $[\![n,k,d]\!]_2$ to denote a binary quantum error-correction code (i.e., with qubits) of parameters n,k,d. It is easy to see that a code $[\![n,k,d]\!]_2$ allows the correction of $t := \lfloor(d-1)/2\rfloor$ errors.

There are also asymptotic bounds for the quantum error-correction codes $[\![n,k,d]\!]_2$ similar to those presented for classical error-correcting codes (Ekert and Macchiavello, 1996; Preskill, 1998).

- Hamming's quantum upper bound:

$$R := k/n \leqslant 1 - H_2(t/n) - (t/n)\log_2 3, \quad n \gg 1. \tag{33}$$

- The Gilbert-Varshamov quantum lower bound:

$$R \geqslant 1 - H_2(2t/n) - (2t/n)\log_2 3, \quad n \gg 1. \tag{34}$$

As in the classical case, there exist quantum error-correction codes that are asymptotically good. A different question (still open) is their explicit construction.

Example: Let $\mathcal{C}_1$ be a linear and binary classical error-correction code of type $[n,k_1,d_1]_2$, and $\mathcal{C}_2\subset\mathcal{C}_1$ a subcode $[n,k_2,d_2]_2$ of $\mathcal{C}_1$, with $k_2<k_1$. Let $\mathcal{C} := \mathcal{C}_1/\mathcal{C}_2$ be the quotient space, of dimension $2^{k_1-k_2}$.

Let us introduce a quantum error-correction code $\mathcal{Q}\subset\mathbb{C}^{2^n}$ of dimension 2^k, with $k=k_1-k_2$, spanned by the vectors

$$|\overline{w}\rangle := 2^{-k_2/2}\sum_{v\in\mathcal{C}_2}|w+v\rangle, \quad w\in\mathcal{C}. \tag{35}$$

Note that this definition does not depend on the element w chosen to represent the class $w+\mathcal{C}$, and that the vectors $|\overline{w}\rangle$ thus constructed form an orthonormal system.

It can be shown that this quantum code recognizes and corrects (up to) $t_\mathrm{b} := \lfloor(d_1-1)/2\rfloor$ bit-flip errors X, and $t_\mathrm{ph} := \lfloor(d_2^\perp-1)/2\rfloor$ phase-flip errors Z, where $d_2^\perp$ is the distance of the code $\mathcal{C}_2^\perp$ dual to $\mathcal{C}_2$. Likewise, the distance d of this quantum code satisfies $d\geqslant\min(d_1,d_2^\perp)$.

The quantum error-correction codes $[\![n,k,d]\!]_2$ thus constructed are called Calderbank-Shor-Steane codes (Calderbank and Shor, 1996; Steane, 1996a, 1996b; Preskill, 1998).

The simplest and most illustrative example of a Calderbank-Shor-Steane code is the $[\![7,1,3]\!]_2$ code of Steane, or a quantum code of seven qubits. It is obtained taking as $\mathcal{C}_1$ the Hamming code $\mathrm{H}_2(1)$ of type $[7,4,3]_2$, and as $\mathcal{C}_2$ its dual ($\mathcal{C}_2=\mathcal{C}_1^\perp$), which is of type $[7,3,4]_2$ and coincides with the even subcode (that is, the code formed by the code words of even weight)[14] of $\mathcal{C}_1$. It corrects one bit-flip error X and one phase-flip error Z. Thus it also corrects a mixed error Y, but not a double bit-flip (or phase-flip) error.

A generator matrix for $\mathrm{H}_2(1)$ is

$$G := \begin{pmatrix} 1&0&1&0&1&0&1\\ 0&1&1&0&0&1&1\\ 0&0&0&1&1&1&1\\ 1&1&1&0&0&0&0 \end{pmatrix} \tag{36}$$

and an associated parity matrix (generator for the dual) is

$$H := \begin{pmatrix} 1&0&1&0&1&0&1\\ 0&1&1&0&0&1&1\\ 0&0&0&1&1&1&1 \end{pmatrix}. \tag{37}$$

Thus a basis of code states is given by

$$\begin{aligned} |\overline{0}\rangle &:= 8^{-1/2}(|1010101\rangle+|0110011\rangle+|0001111\rangle\\ &\quad+|0000000\rangle+|1100110\rangle+|1011010\rangle+|0111100\rangle\\ &\quad+|1101001\rangle),\\ |\overline{1}\rangle &:= 8^{-1/2}(|0100101\rangle+|1000011\rangle+|1111111\rangle\\ &\quad+|1110000\rangle+|0010110\rangle+|0101010\rangle+|1001100\rangle\\ &\quad+|0011001\rangle). \end{aligned} \tag{38}$$

Let us assume that we have a qubit with a state coded as $|\overline{\phi}\rangle := \alpha|\overline{0}\rangle+\beta|\overline{1}\rangle$, in which a bit flip has occurred at

[14]The weight of a binary word is defined as the number of its nonzero coordinates.

the third place (X_3 error). How can we detect and correct it? With the help of an auxiliary system or *ancilla A* that is $(n-k_1=3)$ qubits long we form the state $(X_3|\overline{\phi}\rangle)\otimes|000\rangle_A$, which we transform by the unitary map defined on $\mathbb{C}^{2^7}\otimes\mathbb{C}^{2^3}$ by $|v\rangle\otimes|000\rangle_A\mapsto|v\rangle\otimes|Hv\rangle_A$, with the result $(X_3|\overline{\phi}\rangle)\otimes|He\rangle_A$, where $e:=0010000$ is the binary word that signals the place number 3 at which the bit-flip error occurred. But $He=110$, which is also number 3 in (reversed) binary form. That is, we have marked in the ancilla the syndrome of the error made. It is essential that the ancilla remain in a state depending only on the error, and not on the particular state of the system. Now it is enough to measure the state of the ancilla in order to find out that the error made has been X_3, to apply the operator X_3^{-1} to the system in order to retrieve the state free of error $|\overline{\phi}\rangle$, and to bring back the ancilla to its neutral state $|000\rangle_A$. Finally, suppose instead that the error to detect and correct is a phase flip at the fifth place (Z_5 error). Since $Z_5=U_{\mathrm{H}}^{\otimes 7}X_5U_{\mathrm{H}}^{\otimes 7}$, with U_{H} being the unary Hadamard application, it is enough for the system to go through the operation $U_{\mathrm{H}}^{\otimes 7}$, then to apply the previous strategy, and finally to act with $U_{\mathrm{H}}^{\otimes 7}$ once more.

E. Entanglement distillation

In addition to quantum error-correction codes there is another method for beating decoherence that is especially suitable when communicating over noisy channels. It is based on the notion of *entanglement distillation* or *purification*: two spatially separated parties A and B sharing a collection of entangled pairs, are allowed to perform quantum local operations and classical communication (LOCC; see Sec. III.A) to extract a reduced sample of pairs with a higher purity of entanglement. Entanglement distillation serves as a useful tool for quantum communication, providing us with more powerful protocols for dealing with errors (decoherence) than quantum error correction (Bennett, Brassard, *et al.*, 1996).

First we need a *measure of entanglement* (Vedral and Plenio, 1998). In distillation an appropriate measure of entanglement for a pure bipartite state $|\Psi_{\mathrm{AB}}\rangle$ is $E(|\Psi_{\mathrm{AB}}\rangle)$ [Eq. (27)]. This is because given n pure bipartite states $|\Psi_{\mathrm{AB}}\rangle$, local actions and classical communications are enough to prepare m perfect singlet states with a yield m/n approaching $E(|\Psi_{\mathrm{AB}}\rangle)$ as $n\to\infty$ (Bennett, Brassard, *et al.*, 1996; Bouwmeester, Ekert, and Zeilinger, 2000).

Finding optimal purification procedures in full generality is an open problem. However, explicit examples of entanglement distillation protocols are known to work at least with particular types of mixed states, like the initial entanglement distillation protocol introduced by Bennett, Brassard, *et al.* (1996), which we shall refer to as the *BBPSSW96 protocol*. It is neither optimal nor fully general, but it is the basic protocol from which other generalizations are derived.

In the BBPSSW96 protocol, there are two parties, A and B, Alice and Bob, who communicate over a noisy channel. They share entangled pairs of states and they aim to obtain singlets (21) from them. Their basic strategy is to coordinate their actions through classical messages sacrificing some of the entangled pairs to increase the purity of the remaining ones.

Alice and Bob want to distill some pure entanglement, say in the form of singlet states $|\Psi^-\rangle$ [Eq. (21)], from a given collection of shared entangled pairs in an arbitrary bipartite mixed state ρ. The purity of ρ is measured through the fidelity

$$F:=\langle\Psi^-|\rho|\Psi^-\rangle \tag{39}$$

relative to a perfect singlet. To be specific, in this protocol Alice and Bob share two entangled pairs, each one in the state

$$W_F:=F|\Psi^-\rangle\langle\Psi^-|+\frac{1}{3}(1-F)[|\Psi^+\rangle\langle\Psi^+| \\ +|\Phi^+\rangle\langle\Phi^+|+|\Phi^-\rangle\langle\Phi^-|]. \tag{40}$$

These are called Werner states (Werner, 1989). Note that they are depolarized in the space orthogonal to the singlet. The initial state in $(\mathcal{H}_{\mathrm{A}_1}\otimes\mathcal{H}_{\mathrm{B}_1})\otimes(\mathcal{H}_{\mathrm{A}_2}\otimes\mathcal{H}_{\mathrm{B}_2})$ is therefore

$$\rho_0:=W_F\otimes W_F. \tag{41}$$

We assume that the Werner pairs have fidelity $F>1/2$.

Step 1. Unilaterally, Alice applies the gate Y on each of her two pairs of qubits. This brings ρ_0 to

$$\rho_1:=(Y\otimes 1)\otimes(Y\otimes 1)\rho_0(Y\otimes 1)\otimes(Y\otimes 1). \tag{42}$$

The Pauli operators map the Bell states (21) onto one another in a 1:1 pairwise fashion, leaving no state unchanged (up to irrelevant phase factors, which we shall ignore); in particular, $Y\otimes 1:|\Psi^{\pm}\rangle\leftrightarrow|\Phi^{\mp}\rangle$. Then

$$\rho_1=W'_F\otimes W'_F \tag{43}$$

with

$$W'_F:=F|\Phi^+\rangle\langle\Phi^+|+\frac{1}{3}(1-F)[|\Phi^-\rangle\langle\Phi^-| \\ +|\Psi^-\rangle\langle\Psi^-|+|\Psi^+\rangle\langle\Psi^+|]. \tag{44}$$

The outcome is a new bipartite state with a large component $F>1/2$ of $|\Phi^+\rangle$ and equal components of the other three Bell states.

Step 2. Bilaterally, Alice and Bob apply a CNOT operation (12) to each of their pairs of qubits. Let us denote this joint operation as U_{BCNOT}. Thus

$$\rho_1\mapsto\rho_2:=U_{\mathrm{BCNOT}}\rho_1U_{\mathrm{BCNOT}}. \tag{45}$$

This composite operation acts conditionally on qubits 3 and 4 (target qubits) depending on the states of qubits 1 and 2 (source qubits), namely,

$$U_{\mathrm{BCNOT}}:=(|0\rangle\langle 0|\otimes 1\otimes 1\otimes 1+|1\rangle\langle 1|\otimes 1\otimes U_{\mathrm{NOT}}\otimes 1) \\ \times(1\otimes|0\rangle\langle 0|\otimes 1\otimes 1+1\otimes|1\rangle\langle 1|\otimes 1\otimes U_{\mathrm{NOT}}). \tag{46}$$

TABLE I. The two columns on the right list the states after the action of BCNOT (46) starting from the states on the left two columns. The notation n.c. = no change.

Before		After	
Source	Target	Source	Target
$\|\Phi^{\pm}\rangle$	$\|\Phi^{+}\rangle$	n.c.	n.c.
$\|\Phi^{\pm}\rangle$	$\|\Psi^{+}\rangle$	n.c.	n.c.
$\|\Psi^{\pm}\rangle$	$\|\Phi^{+}\rangle$	n.c.	$\|\Psi^{+}\rangle$
$\|\Psi^{\pm}\rangle$	$\|\Psi^{+}\rangle$	n.c.	$\|\Phi^{+}\rangle$
$\|\Phi^{\pm}\rangle$	$\|\Phi^{-}\rangle$	$\|\Phi^{\mp}\rangle$	n.c.
$\|\Phi^{\pm}\rangle$	$\|\Psi^{-}\rangle$	$\|\Phi^{\mp}\rangle$	n.c.
$\|\Psi^{\pm}\rangle$	$\|\Phi^{-}\rangle$	$\|\Psi^{\mp}\rangle$	$\|\Psi^{-}\rangle$
$\|\Psi^{\pm}\rangle$	$\|\Psi^{-}\rangle$	$\|\Psi^{\mp}\rangle$	$\|\Phi^{-}\rangle$

The possible results of acting with BCNOT on the Bell states as source and target states are summarized in Table I.

Step 3. Alice and Bob measure (with respect to the computational basis) their target qubits, i.e., Alice measures qubit 3 and Bob qubit 4. They then share their results by classical communication. If their results agree, they each keep their unmeasured source qubits, otherwise they discard them.

The source state ρ_s' thereby obtained is a convex combination of the Bell projections, with a weight of $|\Phi^+\rangle\langle\Phi^+|$ given by

$$F' := \frac{F^2 + \frac{1}{9}(1-F)^2}{F^2 + \frac{2}{3}F(1-F) + \frac{5}{9}(1-F)^2}. \tag{47}$$

The rest $1-F'$ is not equally distributed among the other three Bell states.

Step 4. Unilaterally, Alice applies Y on her source qubit in order to convert ρ_s' into a state ρ_s of fidelity F' (relative to $|\Psi^-\rangle$).

Step 5. The state ρ_s is not a Werner state. But there is a depolarizing procedure, called bilateral random operation, that mutates it back into such a state while preserving its fidelity (Bennett, Divincenzo, *et al.*, 1996).

The net result of this protocol is that, with probability greater than $\frac{1}{4}$, one Werner pair of fidelity $F' > F > \frac{1}{2}$ [Eq. (47)] is distilled out of two Werner pairs of fidelity $F > \frac{1}{2}$.

An initial supply of N Werner states of fidelity F is halved by a single run of the above protocol to a sample of Werner states of fidelity $F' > F$. Iterating the procedure as much as necessary, Werner states of purity F_{out} arbitrarily close to 1 can be distilled from a supply of input mixed states ρ of any purity $F_{\text{in}} > \frac{1}{2}$.[15]

The overall result of the BBPSSW96 protocol is to simulate a noiseless quantum channel by a noisy one with the assistance of local actions and classical communication. It assumes tacitly that the quantum channel is shorter than its coherence length; otherwise one might resort to the assistance of quantum repeaters (Dür, Briegel, *et al.*, 1999).

There are also entanglement distillation protocols using one single pair of qubits (Gisin, 1996; Kwiat *et al.*, 2001).

Finding the optimal distillation protocols for a general state and any number of copies is the unsolved *distillability problem*. While it has not been solved, a surprising result has emerged: the existence of entangled states that cannot be distilled, called *bound entangled states* (Horodecki *et al.*, 1998). Explicit examples of such states were found by Horodecki *et al.* (1999). These states are useless for quantum communication protocols, and it is important to distinguish them from distillable states, also known as *free entangled states*. In some general instances, it is possible to conclude that a mixed state is bound entangled: if ρ is entangled and satisfies the Peres criterion $\rho^{t_j} \geq 0$ (Sec. III.A), then ρ is a bound entangled state (Horodecki *et al.*, 1998).

In summary, entanglement is a new resource for computation processing and communication, able to change information theory both qualitatively and quantitatively. The concept of entanglement is a genuinely quantum phenomenon that allows us to extend the theory of information beyond its classical limitations. We have already seen error-correction codes as one essential application of entanglement. Other examples, such as teleportation, dense coding, quantum key distribution, and quantum computation, are addressed in the sections that follow.

IV. QUANTUM TELEPORTATION

Copying a classical system (be it an Etruscan fibula, a Goya painting, or a banknote) has never posed insurmountable difficulties to experts. It suffices to thoroughly observe the original as much as may be required, taking care not to damage it, to retrieve the information needed to make a copy of it. This careful observation does not alter in a noticeable way its state. But if the original to be reproduced is a quantum system in an unknown state ϕ, then any measurement (incompatible with P_ϕ) made on the system to get information on ϕ will uncontrollably perturb the state, destroying the original (Sec. III). Moreover, even if we have an unlimited number of copies of that state, infinitely many measurements will be necessary to determine that unknown state.

For example, let us assume that Alice has a qubit (say one spin-$\frac{1}{2}$ particle) in a pure state. Bob needs it, but Alice does not have any quantum channel to transmit it to him. If Alice knows the precise state of her qubit (for example, if she knows that her spin $\frac{1}{2}$ is oriented in the direction $\mathbf{n}$), it is enough for her to give Bob in a letter (classical channel) that information (the components of $\mathbf{n}$) to enable him to prepare a qubit exactly equivalent to hers. But if she happens not to know the state, she may choose to tell Bob, who would then be obliged to prepare his qubit in a random way, obtaining a 50% fidelity

[15]The map $F \mapsto F'$ is strictly increasing in the interval $[\frac{1}{2},1]$ and has an attractive fixed point at $F=1$.

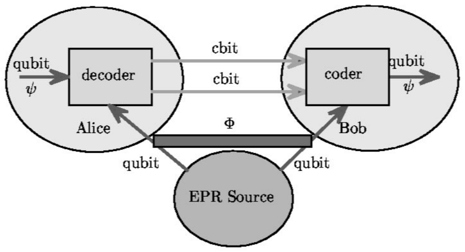


FIG. 3. Scheme for quantum teleportation.

on average. But Alice can also try to be more cooperative, making, for example, a measurement on her qubit of $\mathbf{n}\cdot\boldsymbol{\sigma}$, with $\mathbf{n}$ arbitrarily chosen, and then transmitting to Bob both the components of $\mathbf{n}$ and the result $\epsilon=\pm 1$ thus obtained. Armed with this information, Bob can prepare his qubit in the state $\frac{1}{2}(1+\epsilon\mathbf{n}\cdot\boldsymbol{\sigma})$. The average fidelity so obtained is larger than before: 2/3. However, it is not enough.

If Alice and Bob share an Einstein-Podolsky-Rosen pair, there exists a protocol, devised by Bennett *et al.* (1993), known as *quantum teleportation*, which, resorting to the quantum entanglement of states and the nonlocality of quantum mechanics, allows Bob to reproduce Alice's unknown quantum state with the assistance of only two cbits of information sent by Alice to Bob through a classical channel. This procedure necessarily destroys Alice's state (otherwise it would violate the quantum no-cloning theorem; Sec. III). Let us have a closer look at the aforementioned protocol (see Fig. 3; Rieffel and Polack, 2000).

Let $|\psi\rangle=\alpha|0\rangle+\beta|1\rangle$ be Alice's qubit, with $\alpha=\cos\frac{1}{2}\theta$, $\beta=e^{i\phi}\sin\frac{1}{2}\theta$. And let $|\Phi\rangle:=2^{-1/2}(|00\rangle+|11\rangle)$ be the EPR state shared by Alice and Bob, with Alice having the first of its qubits and Bob the second. The initial state is thus $|\psi\rangle\otimes|\Phi\rangle$, of which Alice can locally manipulate the first two bits and Bob the third one.

Step 1. Alice applies to the initial state the unitary operator $U:=[(U_{\rm H}\otimes 1)U_{\rm CNOT}]\otimes 1$, acting with the CNOT gate on the first two qubits and next with the Hadamard gate H on the first one. The resulting state is

$$\tfrac{1}{2}(|00\rangle\otimes|\psi\rangle+|01\rangle\otimes X|\psi\rangle+|10\rangle\otimes Z|\psi\rangle+|11\rangle\otimes Y|\psi\rangle). \tag{48}$$

Step 2. Alice then measures the first two qubits, obtaining $|00\rangle$, $|01\rangle$, $|10\rangle$, or $|11\rangle$ equiprobably.[16] Alice lets Bob know the result, sending him two cbits: the pair of binary digits 00,01,10,11 thus obtained. As a by-product of Alice's measurement, the first bit ceases to be in the original state $|\psi\rangle$, while the third qubit gets projected onto $|\psi\rangle, X|\psi\rangle, Z|\psi\rangle, Y|\psi\rangle$, respectively.

Step 3. Once Bob receives the classical information sent by Alice, he needs only to apply on his qubit the corresponding gate $1, X, Z, Y$, in order to drive it to the desired state $|\psi\rangle$.

Notice that this teleportation sends an unknown quantum state from one place (whence its vanishes) to another place (where it shows up) without really traversing the intermediate space. It does not violate causality, though. In the first part of the process, quantum correlations get established between the Bell states obtained by Alice and the associated states of Bob's qubit. In the remaining part, to conclude the teleportation, information is transmitted by classical means, in the standard nonsuperluminal fashion. Notice also that in this "noncorporeal" process, it is the information about the quantum state, the qubit, and not the physical state itself, that gets passed from Alice to Bob. There has been no transportation whatsoever of matter, energy, or information at a speed greater than the speed of light.

It is nevertheless surprising in quantum teleportation that all the information needed to reproduce the state $|\psi\rangle=(\cos\frac{1}{2}\theta)|0\rangle+e^{i\phi}(\sin\frac{1}{2}\theta)|1\rangle$ [information that is infinite for it requires fixing a point (θ,ϕ) on the Bloch sphere with infinite precision, thus requiring infinitely many bits] can be accomplished with only two cbits, provided that an EPR state is shared. This state, by itself, generates only potentially an infinite number of random and correlated bit pairs.

An *ebit* is the amount of entanglement in a two-qubit state maximally entangled (usually in a bipartite pure state with entanglement entropy 1; Bennett, Divincenzo, *et al.*, 1996). As an "exchange currency," one ebit is a computing resource made up of a shared EPR pair. Writing $a\lhd b$ to indicate that a resource a is implementable upon spending the resource b, the following relations are quite apparent: 1 cbit$\lhd$1 qubit (to transmit 1 cbit it is enough to send 1 qubit in one out of two orthogonal states), and 1 ebit$\lhd$1 qubit (to have 1 ebit it is enough to produce an EPR pair and to send half of it to the other partner). With this formulation, quantum teleportation allows us to write 1 qbit$\lhd$1 ebit+2 cbits (Bennett, 1995).

Quantum teleportation was realized experimentally with photons for the first time in two laboratories (Bouwmeester *et al.*, 1997; Boschi *et al.*, 1998). At least, this is what these authors claim, although their results have been questioned (Braunstein and Kimble, 1998; Vaidman, 1998; Braunstein, Fuchs, and Kimble, 2000; see, however, Bouwmeester *et al.* 1998, 2000). In the experiment by the Roma group (Boschi *et al.*, 1998), the initial state to be teleported from Alice to Bob was a photon polarization, but not an arbitrary one, for it coincided with that of Alice's photon in the shared EPR photon pair. In the experiments by the Innsbruck group (Bouwmeester *et al.*, 1997), however, the teleported state was arbitrary. Teleportation was reached with a

[16]Steps 1+2 amount to performing a Bell measurement on the initial state, thus correlating the Bell states 00±11,01±10 of Alice's two qubits with the states of Bob's qubit. It suffices to note that

$$\begin{aligned}|\psi\rangle|\Phi\rangle&=\frac{1}{\sqrt{2}}|\psi\rangle(|00\rangle+|11\rangle)\\&=\frac{1}{2\sqrt{2}}[(|00\rangle+|11\rangle)|\psi\rangle+(|01\rangle+|10\rangle)\\&\quad\times X|\psi\rangle+(|00\rangle-|11\rangle)Z|\psi\rangle+(|01\rangle-|10\rangle)Y|\psi\rangle].\end{aligned}$$

high fidelity of 0.80 ± 0.05,[17] but with a reduced efficiency (25% of cases).

It does not seem to be easy to implement the theoretical protocol with 100% effectiveness. The Bell operator (which distinguishes among the four Bell states of two qubits) cannot be measured unless both qubits interact appreciably with each other (as occurs with the CNOT gate used in the protocol explained above), something that is very hard to achieve with photons. However, hopes are high that this will be easier with atoms in electromagnetic cavities.

Teleportation has also been realized in states that are parts of entangled states (Pan *et al.*, 1998).

We should also mention that work has been done on quantum teleportation of states of infinite-dimensional systems (Furuzawa *et al.*, 1998), specifically, the teleportation of coherent optical states based on pairs of EPR squeezed states. In this experiment, whose fidelity is 0.58 ± 0.02 (higher than the maximum $\frac{1}{2}$ expected without resorting to entanglement), a third party, the *verifier* Victor, supplies Alice with one state that is known to him but not to her. After that state is teleported from Alice to Bob, Victor verifies on output if Bob's state is similar to the one he provided to Alice. This experiment, quite different from the others, led the authors to claim priority in the realization of teleportation.

Quantum teleportation, which doubtlessly will be extended to entangled states from different kinds of systems (photons and atoms, ions and phonons, etc.), could lead to remarkable applications for quantum computers and computer networks (for example, combined with prior distillation of good EPR pairs). Moreover, quantum memory records could be created by teleportation of information on systems such as photons to other systems as trapped, well-isolated ions in cavities (Bennett, 1995; Bouwmeester *et al.*, 1997).

V. DENSE CODING

Classical information can also be sent through quantum channels: to transmit the word 10011, it is enough that Alice prepare 5 qubits in the states $|1\rangle, |0\rangle, |0\rangle, |1\rangle, |1\rangle$ and send them to Bob through the quantum channel, and that Bob measure each of them in the basis $\{|0\rangle, |1\rangle\}$. Each qubit carries a cbit, and this is the most it can do in isolation. But if Alice and Bob share beforehand an entangled state, then two cbits of information can be sent from Alice to Bob with a single qubit. This is cast in the formula $2\,\text{cbits} \lhd 1\,\text{ebit} + 1\,\text{qubit}$.

In fact, entanglement is a computing resource that makes possible more efficient ways of coding information (Bennett and Wiesner, 1992). One of these goes by the name of *quantum dense coding* (or superdense coding). Assume, for instance, an entangled state of two

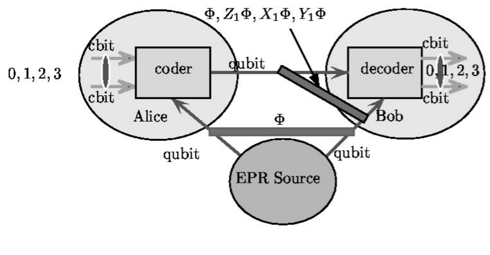


FIG. 4. Scheme for dense quantum coding.

photons. One of the photons goes to Alice, the other one to Bob. She performs one of the following operations on the polarization of her arriving photon: identity, flipping (that is, ↔⇄↕, or ↻⇄↺), change of π in the relative phase, and the product of the last two. Once this is done, she sends the photon back to Bob, who measures in which of the four Bell states the photon pair is. In this fashion they have been able to share two bits of information over one single particle with only two states, that is, by means of a qubit. This is twice what can be accomplished classically, hence the name dense coding. Moreover, if Eve intercepts the qubit, she cannot get from it alone any information whatsoever, for its state is $\frac{1}{2}I$. All the information lies in the entangled state, and Bob possesses half of the pair. Actually, Alice has sent Bob two qubits, but sent the first one long ago, as part of the initial entangled state. This fact has allowed them to communicate more efficiently, using the entangled state they shared.

Dense coding is in some sense the inverse process to teleportation. In the latter the communication of two cbits allows us to reproduce a qubit state, while in the former the communication of a qubit carries along two cbits of information.

The following is a protocol that thoroughly implements what we have just explained (Rieffel and Polack, 2000): an Einstein-Podolsky-Rosen source supplies Alice and Bob with EPR two-particle states like $|\Phi\rangle := 2^{-1/2}(|00\rangle + |11\rangle)$, one of whose particles goes to Alice and the other to Bob, who keep them. Alice is supplied with two cbits, which represent the numbers 0,1,2,3 as 00,01,10,11 (see Fig. 4).

Step 1. Coding. According to the value of that number, Alice effects on her EPR state half the unitary operation $1, Z, X, Y$, which brings the EPR state to $00+11, 00-11, 10+01, 10-01$. Once this is done, she sends her half to Bob.

Step 2. Decoding. Upon reception, Bob first effects on the EPR pair a CNOT operation, such that the state becomes $00+10, 00-10, 11+01, 11-01$. He then measures the second qubit; if he finds 0, he already knows that the message was 0 or 1, and if he finds 1, the message was 2 or 3. That is, he has gotten the first bit of the two-bit message. In order to know the second one, Bob next applies a Hadamard transformation on the first qubit; thereby the state becomes $00, 10, 01, -11$. Measuring the first bit, if he finds 0, he knows that the message was 0 or 2, and if he finds 1, the message was 1 or 3; that is, he has just gotten the second bit of the message.

[17]This fidelity overcomes the value $\frac{2}{3}$ corresponding to the case in which Alice measures her qubit and communicates the result to Bob classically.

An experiment of this nature was performed in Innsbruck (Mattle *et al.*, 1996), using as a source of entangled photons the parametric down conversion that a nonlinear crystal of β-barium borate produces: UV photons are disintegrated (though with low probability) in a pair of softer photons, with polarizations that in a certain geometric configuration are entangled. In that experiment one qutrit/qubit was sent, that is, $\log_2 3 = 1.58$ cbits per qubit.

In a recent experiment, in which the qubits were the spins of ^{1}H and ^{13}C in a chloroform molecule ^{13}CHCl$_3$ marked with ^{13}C, and nuclear magnetic resonance (NMR) techniques were employed to initialize, manipulate, and read out the spins, the authors claim to have reached two cbits per qubit (Fang *et al.*, 2000).

The initial preparation of the entangled pair and the subsequent transmission of the information qubit may be done in the opposite senses; for example, Bob sends Alice one half of the entangled state, keeping the other half for himself, and then Alice uses her qubit to send Bob the desired information. This may be of interest if the cost of transmission in one direction is higher than the cost in the reverse direction. Moreover, distribution of the entangled state prior to the communication, can be scheduled so as to profit from transmission hours at lower charges.

On the other hand, intercepting the message from Alice to Bob does not provide one whit of information to an eavesdropper, for the message is entangled with the part of the EPR system possessed by Bob. Therefore it is automatically an encrypted emission (except if Eve intercepts both the original pair and the message and she replaces them).

VI. CRYPTOGRAPHY

A. Classical cryptography

Cryptography has been a very important part of information theory since 1949 and the pioneering works of Shannon at Bell Labs. He proved that there exist unbreakable codes or *perfectly secret systems* (Shannon, 1949). As a matter of fact, one had been known since 1918 (but not that it was unbreakable). This was the *one-time pad system* (ONETIMEPAD), also named the VERNAM code (Vernam, 1926); it was devised by the young engineer Vernam at AT&T in December 1917 and proposed to the company in 1918 (Kahn, 1967). With Vernam's system both ciphering and deciphering of messages became automatic tasks for the first time.

1. One-time pad

To encode with the one-time pad one starts from the *plain* or *source text* to be ciphered, written as a series $\{p_1, p_2, \ldots, p_N\}$ of integers $p_j \in \mathbb{Z}_B$; then a *key* $\{k_1, k_2, \ldots, k_M\} \in \mathbb{Z}_B^M$, $M \geqslant N$, randomly chosen, is used to produce a *ciphered text* or *cryptogram* $\{c_1, c_2, \ldots, c_N\}$ by combining the key with the plain text in modular arithmetic $c_j := p_j + k_j \bmod B, 1 \leqslant j \leqslant N$. The module B is the maximum number of distinct symbols [2 for binary, 10 for digits, 27 for letters (English text and blank space symbol), etc.].

Both the sender (Alice) and the receiver (Bob) need to have the same key of random numbers, so that upon reception of the cryptogram, Bob undoes the algorithm with that key, thereby recovering the original text.

Possible repetitions in the source text (to which code-breakers resort for decoding) are washed out by the random key. The length of the random sequence must be greater than or equal to that of the source text and must not be employed more than once.[18] Shannon showed that if the key length is smaller than the text length and one reuses the key cyclically to encrypt the message, then it is possible to extract information from the encoded text (Shannon, 1949). These requirements make this procedure very burdensome when there is a large amount of information to encrypt. Moreover, it is not easy to have long series of truly random numbers at our disposal.

This cipher system was used by German and Russian diplomats during the Second World War and by Soviet espionage during the Cold War (Hughes *et al.*, 1995). It is popularly known as "one-time pad" because the keys were written on a notebook or pad, and each time one was used, the corresponding sheet with the key was torn off and destroyed. It is said that the continued use of the same key was what allowed the unmasking of the Rosenberg spy ring and the atom spy Fuchs (Hughes *et al.*, 1995). This system was also used by Che Guevara to communicate secretly with Fidel Castro from Bolivia (Bennett, Brassard, and Ekert, 1992). And it is routinely used for White House and Kremlin communications through the "hot line."

Although invulnerable, the VERNAM cryptosystem has the shortcoming of demanding keys at least as long as the text to be ciphered. This is why it is used only to cipher highly valuable information. For less delicate or sensitive business it is replaced by shorter (though breakable) encryption keys. It was precisely the goal of breaking secret messages that fostered the development of computers.

2. Public-key cryptographic system

The *public-key cryptographic system* is of great interest, since it avoids some of the shortcomings of the Vernam system. It was devised in the middle of the 1970s by Diffie and Hellman at Stanford (Diffie and Hellman, 1976; Hellman, 1979; Diffie, 1992) and later implemented at MIT by Rivest, Shamir, and Adleman (1978).[19] This system is nowadays used worldwide, for instance, on the Internet.

[18]If two binary cryptograms encoded with the same key are intercepted, their sum modulo 2 eliminates the key and makes it possible to decrypt messages with ease (Collins, 1992).

[19]Apparently, some years before Diffie and Hellman, the British Secret Service knew about this system, but kept it a military secret (Ellis, 1970; Ekert, Hayden, and Inamori, 2000).

Two keys are involved: one person X gives away a public key, which anybody can use, and he/she keeps secret a private key, which is the inverse of the former. The public key is used by any sender S to send coded messages to X; on receipt, X decodes them with the private key. Clearly this is of interest only if X alone, but nobody else, knows how to undo the coding at a reasonable cost. How can we accomplish this? In a subtle and cunning way: to encrypt messages, the public-key system uses trapdoor one-way functions. These are injective maps of complexity **P**, i.e., (computationally) tractable functions, the inverse of which are intractable in practice, that is, highly costly to evaluate unless additional information is supplied (a nondeterministic polynomial-time or **NP** problem). See the Appendix for details. Integer factorization stands out in this type of inverse function, as well as discrete logarithms in finite fields and on elliptic curves (Koblitz, 1994; Welsh, 1995).

The public-key cryptographic system allows one to leave wide open both the encryption algorithm and "half" of the total key, namely, the public key, without suffering any extra insecurity; this contrasts sharply with the controversial DES system (*data encryption standard*), which discloses only the algorithm, but whose vulnerability has been shown (Electronic Frontier Foundation, 1998).

3. Rivest-Shamir-Adleman system

One of the most interesting ways of implementing the public-key cryptographic system is the RSA method of Rivest, Shamir, and Adleman (1978) based on the extreme difficulty of factoring large integer numbers. In particular, it is used to protect electronic bank accounts (for instance, against bank transfers electronically requested). The public key of X consists of a pair of integers $(N(X), c(X))$, the first one very large, say 200–300 digits, and the other one in the interval $(1, \varphi[N(X)])$ and coprime to $\varphi[N(X)]$, where φ is Euler's totient function [$\varphi(n)$ is the number of coprimes to n in the interval $[0,n)$]. The sender S, upon transforming his/her message M into an integer following some public bijective prescription upon which both sender and receiver have agreed, partitions it into blocks $B_j < N(X)$ as lengthy as possible, and encodes each block B as

$$B \mapsto C(B) \equiv B^{c(X)} \bmod N(X). \tag{49}$$

The sender then sends the sequence of cryptograms $\{C(B_j)\}$ to X. Let us denote this coding operation as $M \mapsto P_X(M)$, with the symbol P_X meaning that it was done with the public key $c(X)$ of X. The receiver X decodes each $C(B)$ as

$$C(B) \mapsto B \equiv C(B)^{d(X)} \bmod N(X), \tag{50}$$

where the exponent $d(X)$ for decoding is the private key, which is nothing but a solution to

$$c(X) d(X) \equiv 1 \bmod \varphi[N(X)]. \tag{51}$$

That solution is (Koblitz, 1994)

$$d(X) \equiv c(X)^{\varphi\{\varphi[N(X)]\}-1} \bmod \varphi[N(X)]. \tag{52}$$

We shall indicate the decoding as $P_X(M) \mapsto S_X[P_X(M)] = M$, where the symbol S_X refers to the secret key of X.

In principle, since $c(X)$ and $N(X)$ are known, anybody can compute $d(X)$, and hence break the code. But it is here that the shrewdness of X enters the picture. In order to make it extremely difficult for Eve, the eavesdropper, to decode the message, it is better that X abide by certain rules (Salomaa, 1996), among which we highlight the following.

(1) He/she must choose $N(X)$ as the product p_1, p_2 of two large and random prime numbers (with at least 100 digits each), not very close to one another (for this it is enough that the lengths of their expressions differ in a few bits); the numbers must also not be tabulated or have some special form. Algorithms for testing primality, such as the probabilistic Miller and Rabin algorithm (Miller, 1976; Rabin, 1980) or the deterministic Adleman-Pomerance-Rumely-Cohen-Lenstra algorithm, discovered by Adleman, Pomerance, and Rumely (1983) and later simplified and improved by Lenstra and Cohen (Cohen and Lenstra, 1984; Cohen, 1993), facilitate enormously the selection of p_1, p_2.

(2) As X knows p_1, p_2, he/she knows how to compute $\varphi[N(X)]$, namely, $\varphi[N(X)] = (p_1 - 1)(p_2 - 1)$. Now X has to choose an integer $d(X)$ (the private key) randomly in the interval $(1, \varphi[N(X)])$, coprime to $\varphi[N(X)]$, and then compute the public key $c(X)$ by means of

$$c(X) \equiv d(X)^{\varphi(\varphi[N(X)])-1} \bmod \varphi[N(X)], \tag{53}$$

or, much better, by solving $c(X)d(X) \equiv 1 \bmod \varphi[N(X)]$ with the classical Euclidean algorithm.

One should reject small private keys $d(X)$ in order to avoid their discovery by plain trial and error. That is why it is convenient to start by fixing $d(X)$. It is not advisable to have $c(X)$ very small either, for then the interception of the same message sent to several addressees sharing the same public key could lead to its breakup without much effort.

Anybody knowing only $N(X)$ but not its factors should "apparently" first factorize $N(X)$ to compute $\varphi[N(X)]$, and hence find out the exponent for decoding;[20] but factorization of a number 250 digits long would take about 10 million years on a 200-MIPS[21] work station with the best algorithm known today (Hughes, 1998).

The RSA system also allows digital authentication of messages, as well as the appending of an electronic or

[20]We say "apparently," because it is unknown so far whether there exist alternative procedures to decode $C(B)$ that do not go through getting the inverse exponent. Nor is it known whether the computation of this key necessarily requires knowing the prime factors of N.

[21]Million of instructions per second; this gives a general idea of a computer's speed, but refers only to CPU speed—real speed also depends on other factors like input/output speed.

digital signature (Koblitz, 1994; Stinson, 1995; Welsh, 1995; van der Lubbe, 1998).

In 1977 Martin Gardner published an encoded message in his Mathematical Games in *Scientific American* using the RSA method, with the promise of a \$100 reward (payable by the Rivest *et al.* group at MIT) for the first person who could decode it (Gardner, 1977):

96869613754622061477140922254355882905759991124

57431987469512093081629822514570835693147662288

3989628013391990551829945157815154.

This cryptomessage had been obtained using the RSA method starting from an English sentence and the dictionary $\sqcup(\text{blank space})\mapsto 00, a\mapsto 01,\ldots,z\mapsto 26$, and using as a public key (RSA-129,9007), where RSA-129 was the following number 129 digits long:

$$\text{RSA}-129$$
$$=114381625757888867669235779976146612$$
$$010218296721242362562561842935706935245733897830$$
$$5971235639587050589890751475992900268795435 41.$$

Decoding this message required factorizing RSA-129 into two prime factors of 64 and 65 digits each. It was then estimated then that the time needed to reach that goal would be about 4×10^{16} years, at least. In 1994 new factorization algorithms[22] and the combined effort in idle time of a cluster of about a 1000 work stations on the Internet did factorize it in about eight months, after a CPU time of 5000 MIPS years, using the quadratic sieve algorithm. These factors are

$$3490529510847650949147849619903898133417764638 4$$
$$93387843990820577\times$$
$$3276913299326670954996198819083446141317764296 7$$
$$992942539798288533.$$

With this knowledge, it is straightforward to recover the original message: *the magic words are squeamish ossifrage* (Atkins, 1995).

Two years later, RSA-130 was broken with the most powerful factorization algorithm to date (the general number field sieve), and after a computation time almost one order of magnitude lower than that employed for RSA-129. In February 1999, the factorization of the next number in the RSA list was over: the RSA-140, after about 2000 MIPS years and the same general number field sieve method. And in August 1999 the factorization of RSA-155 was achieved, also using the general number field sieve and after about 8000 MIPS years.[23] It has 512 bits and is the product of two prime numbers 78 digits long. To give some idea of the magnitude of this problem, in its solution 35.7 CPU years were employed to perform the sieve, distributed among about 300 work stations and PCs, and 224 CPU hours of Cray C916 operation and 2 Gbytes of central memory in order to find the relations between the rows of a giant sparse matrix of 6.7 million rows and as many columns, with an average of 62.27 nonvanishing elements per row.

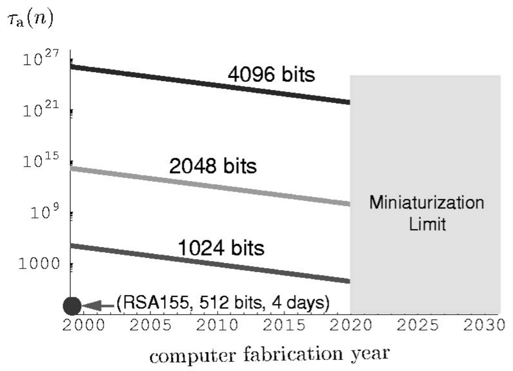


FIG. 5. Factorization with 1000 work stations with increasing power according to Moore's law, starting from 800 MIPS (million instructions per second) in 2000. The vertical axis shows the factorization time $\tau_a(n)$, in years, for an integer number of n bits. The horizontal axis shows the calendar year.

A few years ago, it was considered very safe to use 512-bit modules.[24] The preceding example shows that the general number field sieve factorization algorithm renders this bit length insufficient. Today, the use of (768,1024,2048)-bit modules is recommended for personal, corporate, or high security use. In Fig. 5, the estimated factorization times under the joint use of 1000 work stations is represented, assuming that the processing power follows *Moore's law* (doubling every 18 months; Hughes, 1998). See Sec. VII for more details. We take the RSA-155 time as a reference.[25]

Even though the factorization problem remains a difficult one in computer science, nobody knows for sure whether one day a mathematician may come up with a radically new and faster algorithm such that ordinary classical computers could cope with the task of factorizing large integer numbers in polynomial time. As a mat-

[22]There exist efficient methods, such as those based on the quadratic sieve (QS; Pomerance, 1982; Gerber, 1983; Pomerance, 1996), elliptic curves (EC; Lenstra, 1987), and the general number field sieve (GNFS; Lenstra, 1993; Pomerance, 1996). Their complexities are subexponential, but superpolynomial:

$$\text{QS:}\quad \mathcal{O}(\exp\{[1+o(1)]\sqrt{\log N \log\log N}\}),$$
$$\text{EC:}\quad \mathcal{O}(\exp\{[1+o(1)]\sqrt{\log p \log\log p}\}),$$
$$\text{GNFS:}\quad \mathcal{O}(\exp\{[1.923+o(1)](\log N)^{1/3}(\log\log N)^{2/3}\}),$$

where p is the smallest prime factor of N. From 120–130 digits on, the number field sieve seems to overcome the other methods.

[23]We thank A.K. Lenstra and H. te Riele for sharing with us their information about the latest RSA factorizations.

[24]The number of bits in the integer N is $\lfloor \log_2 N\rfloor+1$.

[25]Miniaturization of classical devices has the atomic/molecular scale as a limit, which at the pace of Moore's will be reached within a couple of decades.

ter of fact, quantum computation has raised high expectations in this regard, with Shor's algorithm (Shor, 1994) to be discussed in Sec. X.D. That is why security agencies closely follow the new advances in number theory and computation.

B. Quantum cryptography

Quantum physics provides us with a secure method for coding, guaranteed by the very laws of physics. The pioneering idea dates back to Stephen Wiesner, who as early as 1969[26] suggested this possibility, as well as the fabrication of forgery-proof, "quantum banknotes" (Wiesner, 1983). In the mid-1980s Bennett and Brassard (1984) devised a quantum cryptosystem based on the Heisenberg uncertainty principle, which soon afterwards was implemented experimentally by sending secret information with polarized photons to a distance of 30 cm apart (Bennett, Bessette, *et al.*, 1992). This system employs quantum states, not all mutually orthogonal, in order to keep them from being cloned by a possible interceptor. Because it uses four distinct states, it is called the *four-state scheme*. The use of nonlocal quantum correlations in pairs of entangled photons (produced, for example, by parametric down conversion) was subsequently proposed by Ekert (1991). Within Ekert's system the Bell inequalities (Bell, 1964, 1966, 1987) are in charge of keeping the security; hence this system is also called the Einstein-Podolsky-Rosen scheme. For a recent detailed review see Gisin *et al.* (2001).

1. Counterfeit-safe "quantum banknotes"

A possible forgery-proof banknote could be provided with a printed ID number and a small collection of (say twenty) photons trapped indefinitely in individual cells of perfectly reflecting walls, and with secret polarizations ↻, ↺, ↕, ↔ randomly distributed, which the issuing bank would keep a record of along with the identification number (see Fig. 6). The bank could therefore at any moment check the legitimacy of the note without ruining it, because it would know beforehand how to place the polarizers to check each photon polarization without destroying it. However, any forger who attempted to copy a note, ignorant of the directions in which the photons were polarized, would perturb the initial polarization, projecting it onto one of two corresponding orientations of the polarizer chosen to measure with (Wiesner, 1983; Bennett, 1992b).

2. Quantum key distribution

Although the quantum notes business may seem like a subject for science fiction, this is not the case for procedures of quantum key distribution. Among the communication protocols, we may highlight the BB84 of Bennett and Brassard (1984), E91 of Ekert (1991), B92 of Bennett (1992a), and an EPR approach without Bell's inequalities, due to Bennett, Brassard, and Mermin (1992). These protocols provide a way for two parties to share keys in absolute secrecy in principle, and thus they are an ideal complement to the Vernam code.

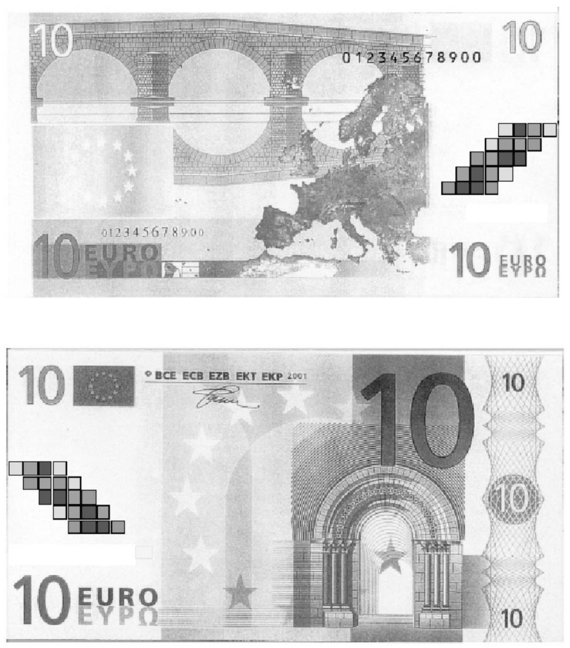


FIG. 6. Counterfeit-safe banknotes: the identification number printed on the bill is correlated with a secret polarization scheme of photons trapped in the individual cells represented by small colored boxes. The allegedly invisible false colors of these boxes are pictured to show the different photon polarizations.

Suppose Alice and Bob want to exchange secret information without recourse to middlemen who bring key pads from one to the other and without fear that someone will break their code. To this end, they must share a key known only to them. They proceed according to a given communication *protocol*, or set of instructions, either to detect any nonauthorized eavesdropper or to determine the secret key that only they will share for coding and decoding.

a. BB84 protocol, or four-state scheme

This is the first protocol devised in quantum cryptography. Alice and Bob are connected by two channels, one quantum and another public and classic. If photons are the vehicle carrying the key, the quantum channel is usually an optical fiber. The public channel can also be so, but with one difference: in the quantum channel, there is in principle only one photon per bit to be transported, while in the public channel, in which eavesdropping by any nonauthorized person does not matter, the intensity is hundreds of times bigger.

Step 1. Alice prepares photons with linear polarizations randomly chosen among the angles $0°$, $45°$, $90°$, and $135°$, which she sends "in a row" through the quan-

[26]His work was finally published in 1983 after being rejected from the journal to which it was first submitted. An unpublished version appeared in 1970.

tum channel, while keeping a record of the sequence of the prepared states and of the associated sequence of 0's and 1's obtained representing by 0 the choices of 0 and 45 degrees, and by 1 otherwise. This sequence of bits is clearly random. For instance, denoting by H, V, D, and A the horizontal, vertical, 45°, and 135° polarizations, respectively, and by $+$, $\times$ the polarization basis {H,V}, {D,A}, Alice's possible sequences are

```
++++x+xx+x++++xx+xx++xxx++x+++x+xxx+xxx++x+++++x...
VVVHAVAAVAHVHHDDVDDHHAAAVHDHVVDVDADVDAAHVDVHHHVA...
111011111101000010000111100011010101011010100011...
```

Step 2. Bob has two analyzers, one "rectangular" ($+$type), the other "diagonal" ($\times$type). Upon receiving each of Alice's photons, he decides at random what analyzer to use, and writes down the aleatory sequence of analyzers used as well as the result of each measurement. He also produces a bit sequence associating 0 to the cases in which the measurement produces a 0° or 45° photon, and 1 in cases of 90° or 135°. With the following analyzers chosen at random a possible result of Bob's action on Alice's previous sequence is

```
x+x+xxxx+++x++x+x+xxxx+++++++xxxx+++x+xxxxxx++x+...
DVAHADAAVVHDHHDHAVDADAHHVHVHVDDADHVVDVAAADADHHDH...
011010111100000011010100101010010011011110100000...
```

Step 3. Next they communicate with each other through the public channel the sequences of polarization basis and analyzers employed, as well as Bob's failures in detection, but never the specific states prepared by Alice in each basis nor the resulting states obtained by Bob upon measuring.

```
Alice to Bob: ++++x+xx+x++++xx+xx++xxx++x+++x+xxx...
Bob to Alice: x+x+xxxx+++x++x+x+xxxx+++++++xxxx++...
```

Step 4. They discard those cases in which Bob detects no photons, and also those cases in which the preparation basis used by Alice and the analyzer type used by Bob differ. After this distillation, both are left with the same random subsequence of bits 0, 1, which they will adopt as the shared secret key:

```
Alice 111011111101000010000111100011010101011010...
      ++++x+xx+x++++xx+xx++xxx++x+++x+xxx+xxx++x...
 Bob  x+x+xxxx+++x++x+x+xxxx+++++++xxxx+++x+xxxx...
      011010111100000011010100101010010011011110...

Alice -1-01-111-0-000---0--1--10-01-0-0--10-1--0...
Bob   -1-01-111-0-000---0--1--10-01-0-0--10-1--0...
```

Therefore the distilled key is

$$10111100000110010010 10\cdots$$

and its length is, on average, and assuming no detection failures, one-half of the length of each initial sequence.

b. Eavesdropping effects

All of this holds in the ideal case in which there are no eavesdroppers, no noises in the transmission, and no defects in the production, reception, or analysis: the distilled keys of Alice and Bob coincide. But let us assume that Eve "taps" the quantum channel and that, having the same equipment as Bob, analyzes the polarization state of each photon, forwarding them next to Bob. Eve, much like Bob, ignorant of the state of each photon sent by Alice, will use the wrong analyzer with probability 1/2 and will replace Alice's photon by another one, so that upon measurement Bob will get Alice's state with probability only 3/8, instead of the probability 1/2 in the absence of eavesdropping. Therefore this intervention of Eve induces on each photon a probability of error 1/4. Returning to the previous example, let us assume that Eve's measurements on Alice's photons produce the following results:

```
Eve x++x++++x++xxx++++++x+xxxx++xx+x+++x+xxx+x...
    DVVAVVVVDVHADAVHVHHHAVAAADHHADHDVVVDHAADVD...
```

Eve's states are now those reaching Bob, who with his sequence of analyzers will obtain, for instance,

```
x+x+xxxx+++x++x+x+xxxx+++++++xxxx+++x+xxxxxx++x+...
DVDVADADHVHAHHDHAHAAAAHHHHHHHDDDAVVVAVADDDAAHHAH...
010110100101000010111100000000000111111000110010...
```

Proceeding as in Step 4:

```
Alice 1110111111010000100001111000110101010111010...
      ++++x+xx+x++++xx+xx++xxx++x+++x+xxx+xxx++x...
Bob   x+x+xxxx+++x++x+x+xxxx+++++++xxxx+++x+xxxx...
      0101101001010000101111000000000001111111000...

Alice -1-01-111-0-000---0--1--10-01-0-0--10-1--0...
Bob   -1-11-100-0-000---1--1--00-00-0-1--11-1--0...
```

We see that the coincidences in the distilled lists get disrupted: in one out of four cases, the coincidence disappears. Sacrificing for verification a piece of the lists taken at random from the final sequences, Alice and Bob can publicly compare them, and their differences will detect Eve's intervention. If the length of that checked partial sequence is N, the probability that Eve's listening has not produced discrepancies is $(3/4)^N$ and is thus negligible for N large enough. Therefore, should they not find any discordance, they can feel safe about the absence of eavesdroppers. But they must clearly disregard the binary string they have made public and not use it for coding.

In practice, the emitting source, the transmission channel, and the receiving equipment all display noise, which will spoil, even without Eve's intervention, the perfect fit of the bit sequences distilled by Alice and Bob. It is necessary then to live with error, so long as this stays within a tolerable limit. In these circumstances, Eve will try to restrain herself, taking care that the effects of her listening stay below a certain threshold and do not sound the alarm.

Cryptanalysts like Eve usually are a good deal more subtle than the previous simple analysis might suggest. Aware as they are of the quantum subtleties, they are not satisfied with incoherently tapping the quantum channel qubit to qubit; they know that a coherent attack on strings of qubits, with probes analyzed after the public exchange of information between Alice and Bob, can be much more rewarding. To test the safety of a protocol such as BB84 under any type of imaginable attack by a malicious and cunning Eve is neither a trivial nor an uninteresting issue, especially bearing in mind that other protocols which were considered to be unconditionally secure have fallen. One such is the *bit commitment* quantum protocol: Alice sends something to Bob under the firm commitment of having chosen a bit b that Bob does not know, but that Alice can later show to him when he claims it. Resorting to entangled EPR states makes it possible for either party of the couple to behave dishonestly (a cheating Alice could change her commitment at the end without Bob's being aware, or an untrustworthy Bob could obtain some information on b without asking Alice; Mayers, 1996, 1997; Brassard *et al.*, 1997).

There exists a proof of the unconditional security of quantum key distribution through noisy channels and up to any distance, by means of a protocol based upon the sharing of EPR pairs and their purification, and under the hypothesis that both parties (Alice and Bob) have fault-tolerant quantum computers (Lo and Chau, 1999). Likewise, the unconditional security of the BB84 protocol is also claimed (Mayers, 1998).

c. B92 protocol

Unlike the previous protocol, which uses a system in four pairwise orthogonal states, in the somewhat simpler B92 protocol, only two nonorthogonal states are involved. We shall not discuss it here, as it is similar to the previous one. The interested reader is referred to the oiginal article of Bennett (1992a).

d. Einstein-Podolsky-Rosen protocols

In 1991 Ekert, relying on earlier ideas of Deutsch (1985), proposed an elegant method for secret key distribution, in which the generalized Bell's inequality safeguards confidentiality in the transmission of pairs of

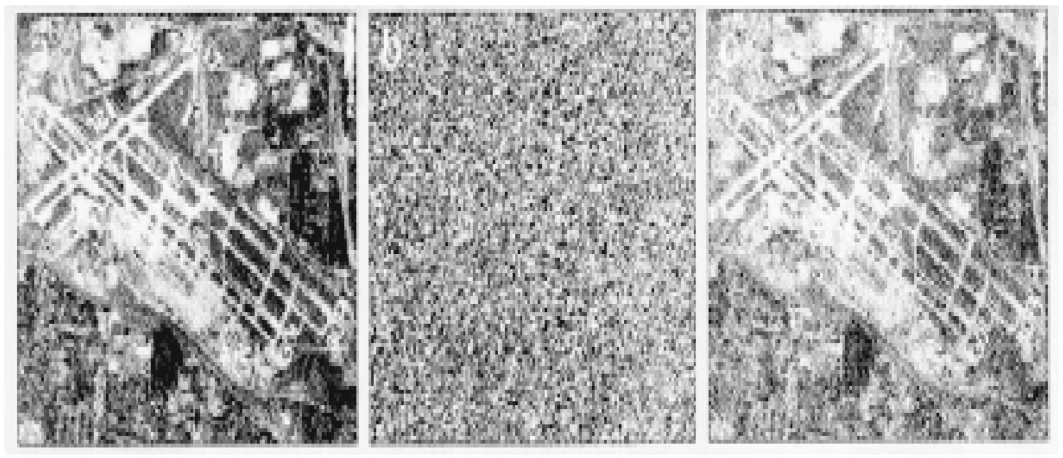


FIG. 7. Aerial view of St. Louis airport (left), image encrypted with a quantum-generated key (center), and decrypted image (right). From Hughes and Nordholt (1999).

spin-$\frac{1}{2}$ particles entangled *à la* Einstein, Podolsky, Rosen, and Bohm (Deutsch, 1985; Ekert, 1991).

Six months after Ekert's work appeared, Bennett, Brassard, and Mermin (1992) presented a very simple scheme for key distribution that still uses Einstein-Podolsky-Rosen-Bohm states in the singlet state $[2^{-1/2}(|01\rangle - |10\rangle)]$, but does not need to invoke Bell's theorem to detect Eve's listening. Alice and Bob measure the spin of their respective subsystems (halves of Einstein-Podolsky-Rosen-Bohm pairs) randomly along Ox or Oz. Through a public channel, they inform each other about their sequences of selected observables, but not of the results $\pm\frac{1}{2}$ obtained. They discard the cases in which their selections differ. They keep the remainder; the results of the latter are evidently anticorrelated. Bob now reverses all his outcomes $(\pm\frac{1}{2} \mapsto \mp\frac{1}{2})$, and then both Alice and Bob add $\frac{1}{2}$ to their results, thereby obtaining the secret key to be shared. Sacrificing as before a piece of the key in the interest of public comparison, they can detect Eve's listening.

Although it can be shown that this protocol is essentially equivalent to the BB84 (Bennett, Brassard, and Mermin, 1992), it offers a potential bonus (Collins, 1992): the users, Alice and Bob, can wait to establish the key until they are about to use it (should they know how to keep the EPR states expectant for a while between their production and use), in this way removing the possibility of Eve's stealing the shared key.

C. Practical implementation of quantum key distribution

The BB84 protocol was implemented for the first time at the IBM T.J. Watson Research Center (1989–1992) with polarized photons sent over 32 cm through air (Brassard, 1989; Bennett, Bessette, *et al.*, 1992). In 1995 the B92 protocol was realized experimentally, also with polarized photons, transmitted this time through optical fiber 22.8 km long in the Swisscom cable connecting the cities of Geneva and Nyon under Lake Leman (Muller, Breguet, and Gisin, 1993; Muller, Zbinden, and Gisin, 1996).

The use of photon polarization states for long distances has a disadvantage: birefringency in the nonstraight parts of the fiber transforms linearly polarized states into states of elliptic polarization, with accompanying losses in transmission, and further produces dispersion of the orthogonal polarization modes. Hence the interest in other ways to codify the states, for example, by means of phases instead of polarizations. A group from British Telecom took this approach in 1994 with optical fiber over a distance of 30 km, using interferometry with phase-encoded photons (Marand and Townsend, 1995). They saw no major obstacles to extending transmission to around 50 km. In 1999 a group from Los Alamos reached 48 km using this procedure (Hughes, Luther, *et al.*, 1996; Hughes, Buttler, *et al.*, 1999; Hughes, Morgan, and Peterson, 2000). The use of phase-encoded photons shows promise for safely connecting diverse government agencies in Washington. To cover distances larger than 100 km would require the use of safe repeaters where key material for rebroadcasting might be generated.

With the B92 protocol, it was possible in 1998 to quantumly transmit the secret key, at a rate of 5 kHz and over 0.5 km in broad daylight and free space, with polarized photons (Hughes, Buttler, *et al.*, 1999; Hughes and Nordholt, 1999). The key was then used to encrypt a photograph (with eight bits per pixel), which the recipient decrypted to reconstruct the primitive image, with the results shown in Fig. 7.

In the near future this procedure will be able to generate secret keys, shared by earth-satellite or satellite-satellite links, that protect the confidentiality of the transmissions.

More recently, quantum key distribution over 360 m has been achieved using variants of E91 and BB84 (Jennewein *et al.*, 2000). Pairs of entangled photons were employed to generate keys at a rate of 0.4–0.8 kHz with an error bit rate of about 3%.

VII. QUANTUM COMPUTATION

A simple and intuitive way to arrive at the notion of quantum computation is through miniaturization.[27] This has been the driving force in the modern upgrade of ordinary computers. As a matter of fact, the computer industry has grown at the same time as integrated circuits have decreasd in size. This rapid growth in the industry will continue as long as it is possible to include more and more circuits in a single chip. However, this pace cannot last forever and at some point we will reach the limits of integrated circuit technology. Even if we can overcome the technological barriers, this trend will lead us to the quantum realm, where the laws of quantum physics will impose fundamental limitations on the size of the circuit components and on their performance. Thus, if the computer industry is to keep growing at the same rate, it will require another technological revolution.

Although such a revolution may seem far in the future, it is estimated that around the year 2020 we shall reach the atomic size for storing one bit. Instead of just waiting for this situation to arrive, some theoretical physicists have decided to move ahead and have already started to wonder about the radical changes and possible advantages that a computer may have if based upon the principles of quantum mechanics.

The estimates for reaching the atomic scale are based on a remarkable observation made by Gordon Moore (1965), later known as Moore's law, that the number of transistors per square inch on integrated circuits had doubled every year since the integrated circuit was invented. Explicitly, the original curve for the density of silicon integrated circuits (transistors per square inch) was $\propto 2^{(t-1962)}$, where t is the calendar year. In subsequent years, the trend slowed down a bit, but chip capacity has doubled approximately every 18–24 months, and this is the current definition of Moore's law (see Fig. 8).

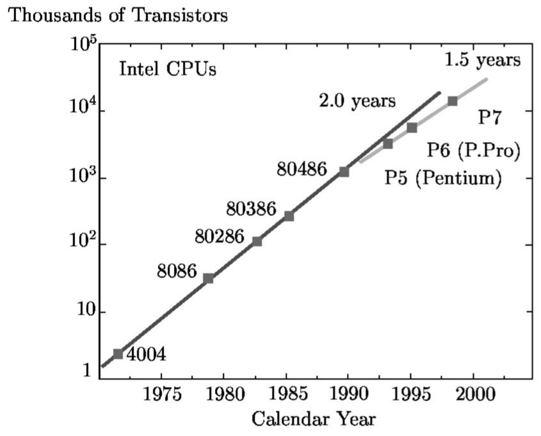


FIG. 8. Moore's law for processor capacity (number of transistors per square inch).

VIII. CLASSICAL COMPUTERS

To pave the way for the concept of quantum computers let us first consider a classical concept, namely, the notion of parallel computation. To properly understand this let us first recall the basic operating principles of the ordinary computers we work with as they were introduced by Turing in 1936 and subsequently developed by von Neumann in 1945 (von Neumann, 1945, 1946), among others.

A. The Turing machine

The concept of a Turing machine was the foundation of the modern theory of computation and computability: the study of what computers can and cannot do. Turing arrived at this concept in 1936 (Turing, 1936) in his attempt to answer one of the questions posed by Hilbert. This was the *problem of decidability* (*Entscheidungsproblem*): Does there exist, at least in principle, a definite method or process by which all mathematical questions can be decided (Hodges, 1992)?

Turing realized that addressing this problem would require a precise and compelling definition of "a definite method," as it appears in the statement of Hilbert's problem. He analyzed what a person does during a methodical process of reasoning. His guiding idea was to translate the human thought process into something purely mechanical. He then went on to map that process into a "theoretical machine" that would operate on symbols on a paper tape according to precisely defined elementary rules. Turing also provided convincing arguments that the capabilities of such a machine would be enough to encompass everything that would amount to "a definite method," which in modern language is what we call an *algorithm*.

We shall see later how Turing answered the question of decidability in the negative using his concept of a Turing machine, which we should first introduce.

A *Turing machine* is a type of machine that has a finite set of states $\mathcal{S}=\{s_1,s_2,\ldots,s_S;s_{S+1}=s_{\rm halt}\}$, a finite alphabet of symbols $\mathcal{A}=\{a_1,a_2,\ldots,a_A;a_{A+1}=\text{blank}\}$, and a finite set of instructions $\mathcal{I}=\{i_1,i_2,\ldots,i_I\}$. In addition, it has an external infinitely long memory tape. This is called an S-state, A-symbol Turing machine.

The states s_i correspond to the functioning modes of the machine, and the Turing machine is exactly in one of these states at any given time. The symbols in the alphabet serve to encode the information processed by the machine: they are used to code input/output data and to store the intermediate operations. The instructions are associated with the states in $\mathcal{S}$, and they tell the machine what action to perform if it is currently scanning a certain symbol, and what state to go into after performing this action. There is a single *halt state* $s_{\rm halt}$ (or halt, for short) from which no instructions emerge, and this halt state is not counted in the total number of states. There

[27]Feynman's famous speech addressing the American Physical Society (Feynman, 1959), with his provocative bets on building microengines and writing on pinheads, signaled the birth of nanotechnology.

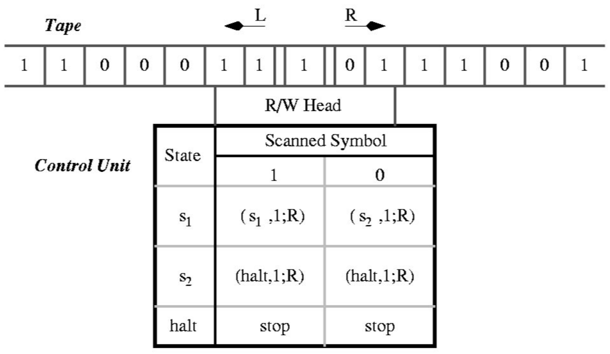


FIG. 9. Components of a Turing machine. The alphabet {1;0} is unary, with 0 denoting blank. "Stop" means that $(s_{\text{halt}},.)$ has no assigned instruction.

is also a blank symbol that serves to separate strings of data coded with the rest of the alphabet symbols.

The elements ($\mathcal{S}$, $\mathcal{A}$, $\mathcal{I}$) are physically arranged as follows. A Turing machine consists of three components (Salomaa, 1989; Papadimitriou, 1994; Welsh, 1995; Li and Vitányi, 1997; Aharonov, 1998; Yan, 2000):

- The *tape*, which is a doubly infinite tape divided into distinct sections or cells. Each cell can hold only one symbol $a_i \in \mathcal{A}$.
- A *read/write (R/W) head* or *cursor*, which can read or write the symbol $a_i \in \mathcal{A}$ in each tape cell.
- A *control unit*, which is a device (or box) that controls the movements of the R/W head based on the current state of the Turing machine and the content of the cell currently scanned by the R/W head, i.e., based on a pair (s_i, a_i).

The read/write head is capable of only three actions:

- Writing on the tape (or erasing from the tape) only the cell being scanned.
- Changing the internal state.
- Moving the head one cell to the left or right. Let us denote this variable as $\gamma \in \{L, R\}$.

The behavior of a Turing machine is governed by the set of instructions $\mathcal{I}$. These are rules that describe the transition from an initial pair (s_i, a_i) to a final pair $(s_f; a_f)$ plus the movement γ of the read/write head. Thus each instruction $j \in \mathcal{I}$ is a 5-tuple $[(s_i, a_i), (s_f, a_f; \gamma)]$ representing the following transition:

$$\mathcal{I} \ni j : (s_i, a_i) \mapsto (s_f, a_f; \gamma). \tag{54}$$

A consistency condition is required: no two instructions $j_1, j_2 \in \mathcal{I}$ have the same initial pair (s_i, a_i).

In Fig. 9 we plot a schematic picture of a Turing machine.

An alternative and efficient way to describe a Turing machine is by means of a flow or state diagram (see Fig. 10). Here each state $s_i \in \mathcal{S}$ is enclosed in a circle, and the instructions associated with a couple of states are represented by arrows also showing the change of symbols on the tape and the head movement.

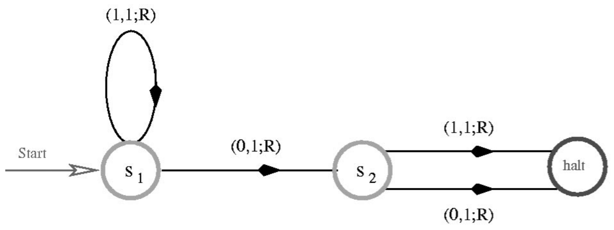


FIG. 10. An example of a flow diagram for a (two-state, one-symbol) Turing machine as shown in Fig. 9.

In Fig. 10 we show a (two-state, one-symbol) Turing machine. It is customary in this case to use 1 for the symbol and 0 for the blank, i.e., $\mathcal{A}=\{1;0\}$. When $A=1$ and $S=2$ we talk of a two-state Turing machine for brevity. This, then, is a unary machine, which should not be confused with a binary system, since each number n is represented as a string of n 1's on the tape, and not by its binary representation. The state set is $\mathcal{S} = \{s_1, s_2; \text{halt}\}$. In this simple example of a Turing machine, when it is in state s_1 scanning a 1, the machine will move right one cell and stay in state s_1 (this is the loop in Fig. 10). When it is in state s_1 scanning a blank symbol, it will change this symbol to a 1 and go to state s_2. When it is in state s_2, it will change both symbols 0,1 to a 1, move right, and stop.

In summary, unless it is in the halt state, this simple Turing machine will march rightward as long as it scans 1's, and when it meets its first blank symbol, it will change this into a 1 and then it will move right twice and stop.

Let us now describe a Turing machine performing a more interesting task like adding two numbers. This is an adding Turing machine. Suppose we want to sum $n_1 + n_2$. The input data on the tape is a string of n_1 1's separated by a 0 from another string of n_2 1's. The output data on the tape must be a string of n_1+n_2 1's. To achieve this output, we need to remove the leftmost 1 in the first string of 1's and convert the intermediate 0 into a 1. Then we can use a two-state Turing machine defined as follows (see Fig. 11). When it is in state s_1 and the R/W head scans a 1, there is a transition to state s_2, the 1 is replaced by 0, and the head moves to the right. Similarly, there are three other instructions that we plot in Fig. 11 in the form of a table of instructions. In Fig. 11 the input is $2+2$ and the output 4.

1. Computability

Despite their simplicity, Turing machines can be devised to compute remarkably complicated functions. In fact, a Turing machine can compute anything that the most powerful ordinary classical computer can compute. Until the formulation of quantum computing, no one had yet proposed a model of computation more powerful than the Turing machine. Thus, if we stick to classical machines and we have to solve problems that a Turing machine cannot solve, it seems that we will have to resort to "supermachines" performing infinitely many

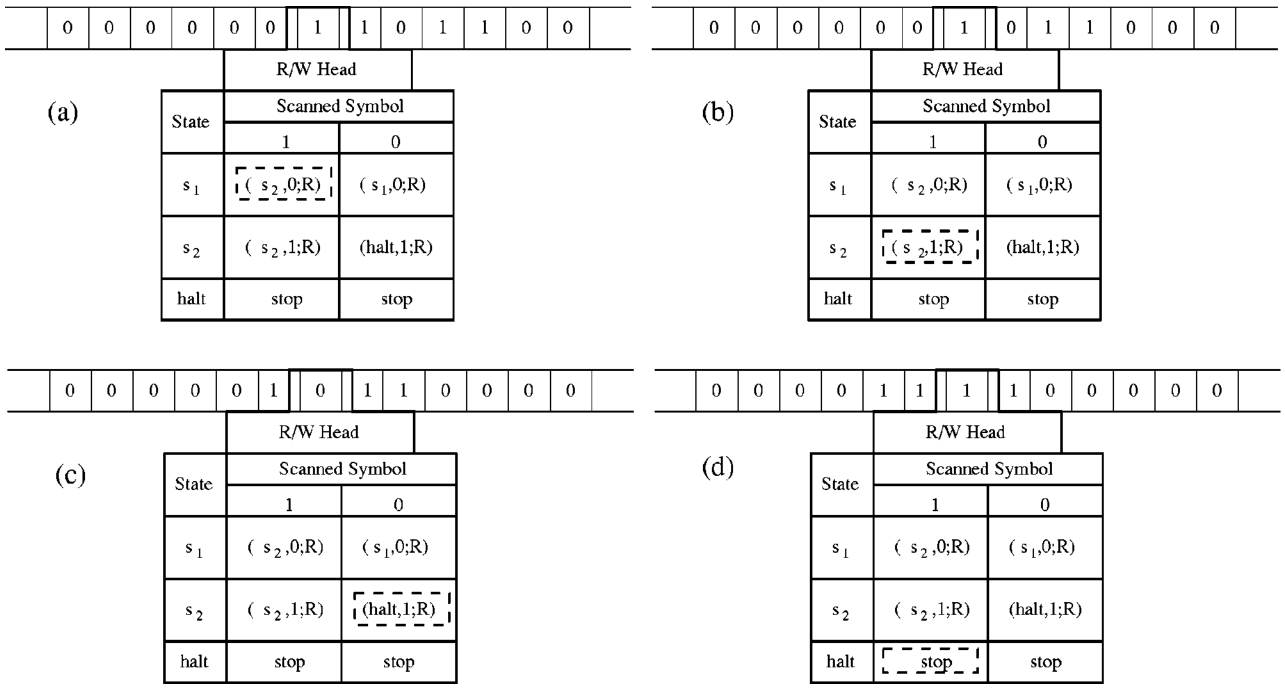


FIG. 11. An example of an adding Turing machine. Following the sequence of instructions in the control unit, the machine performs $2+2=4$.

steps in a finite time or to guess the answer out of the blue or something similar. This idea was formalized into a proposition independently by Church and Turing and goes by the name Church-Turing hypothesis (Church, 1936; Turing, 1936, 1950; Hodges, 1992). Following Turing, it is stated as follows: Every function that would naturally be regarded as computable can be computed by some Turing machine.

This is a hypothesis because it cannot be proved unless we provide a formal definition of what "naturally" means. This hypothesis has not been refuted within the realm of classical physics, but we shall see that the notion of a quantum Turing machine requires the reformulation of the Church-Turing hypothesis.

As a consequence of the Church-Turing hypothesis, a function is called *computable* when it can be computed by a Turing machine, while it is declared a *noncomputable* function otherwise.

2. The universal Turing machine

A further crucial concept introduced by Turing is that of the universal Turing machine (Turing, 1936). So far we have considered Turing machines built for a specific purpose and for that purpose only. The universal Turing machine allows us to run all Turing machines on a general machine. Thus a universal Turing machine is defined as a single machine that comprises all Turing machines and is therefore capable of computing any algorithm.

Just as an ordinary Turing machine T is defined by a set $(\mathcal{S},\mathcal{A},\mathcal{I})$ with the instructions in $\mathcal{I}$ being described by a 5-tuple $[(s_i,a_i),(s_f,a_f;\gamma)]$, a universal Turing machine $T_{\rm U}$ is likewise constructed by providing a set $(\mathcal{S}_{\rm U},\mathcal{A}_{\rm U},\mathcal{I}_{\rm U})$ and a description of its instructions $[(S_i,A_i),(S_f,A_f;\Gamma)]$. These instructions must be general enough to accommodate any possible Turing machine. This is accomplished by supplying it with the information of a Turing machine and the data of its tape.

There are several ways to construct explicitly a universal Turing machine $T_{\rm U}$ (Minsky, 1967; Herken, 1995; Feynman, 1996). For simplicity, let us assume that the alphabet $\mathcal{A}_{\rm U}=\{a_1=0,a_2=1;\mathcal{A}'_{\rm U}\}$ has a binary part corresponding to $\mathcal{A}$. This is not a restriction, since any alphabet $\mathcal{A}$ can be mapped onto a binary alphabet. At any given step in the functioning of $T_{\rm U}$, the initial pair (S_i,A_i) will know about the current description of the tape τ of T, and as it also knows about the set of instructions $\mathcal{I}$, the universal Turing machine will output exactly the same data as the Turing machine T it is simulating. In order to implement this, we need to accommodate a large, but finite, amount of information on the universal Turing machine's tape. Namely, the input data for the tape of $T_{\rm U}$ is precisely all we need to know about the Turing machine it reproduces: $[\tau;(\mathcal{S},\mathcal{A},\mathcal{I})]$. These elements are disposed on the tape of $T_{\rm U}$ consecutively and separated by marks belonging to $\mathcal{A}'_{\rm U}$. The R/W head of $T_{\rm U}$ is positioned at the initial cell of the string encoding the data pair (s_0,a_0). Then the universal Turing machine starts working, resorting to its set of instructions $\mathcal{I}_{\rm U}$. Without going into further detail, this set contains rules specifying how to bring the R/W head to read a pair (s_i,a_i), change it to a new pair (s_f,a_f), and find the movement γ of the tape τ. This is repeated until the given Turing machine T is fully imitated.

The number of states $S_{\rm U}$ and symbols $A_{\rm U}$ is variable in a universal Turing machine. Minsky constructed one with $S_U=7$, $A_U=4$ (Minsky, 1967). In fact, one can in principle always construct a universal Turing machine with only $S_U=2$ and finitely many symbols, or only $A_U=2$ and finitely many states.

The importance of the universal machine is clear. We do not need to have an infinity of different machines doing different jobs. A single one will suffice. The engineering problem of producing various machines for various jobs is replaced by the office work of *programming* the universal machine to do these jobs (Turing, 1948). In summary, a Turing machine is comparable to an algorithm much as the universal Turing machine is to a programmable computer.

3. Undecidability: The halting problem

Turing was able to answer the problem of decidability, rephrased in terms of the Turing machine: is it possible to compute any function by designing an appropriate Turing machine? Turing showed that this is not possible because the set of possible functions is much larger than the set of possible Turing machines. In fact, the set of Turing machines is denumerable (and so is the set of inputs). This is because any Turing machine can be encoded into a finite binary string. However, it is possible to find sets of functions that are uncountable. Turing provided one such example due to Cantor: the set $\mathcal{F}$ of all functions $f:\mathbb{N}\to\mathbb{N}$. Cantor had shown 50 years earlier, with his dilemma of diagonalization, that this set $\mathcal{F}$ was not countable. The proof is simple, by *reductio ad absurdum:* assume $\mathcal{F}$ is denumerable, then label each function $f\in\mathcal{F}$ with an integer: $\mathcal{F}=\{f_0,f_1,\ldots,f_n\cdots\}$. Next construct a function $g:\mathbb{N}\to\mathbb{N}$ by defining $g(k):=f_k(k)+1, \forall k$. This function g is not contained in the initial set $\mathcal{F}$ since it differs for at least one value of the argument from each function in $\mathcal{F}$. Thus the set $\mathcal{F}$ is not complete, which is a contradiction.

This analysis implies that there must be noncomputable functions. Turing provided the first explicit example, known as *the halting problem*: Is it possible to design a Turing machine H that tells us whether or not any Turing machine will halt when executing its procedure for any input? Turing showed that there does not exist such a Turing machine H (Turing, 1936); in other words, the halting problem is undecidable, or equivalently, the predicate ($\{0,1\}$-valued function) $h:\mathbb{N}\times\mathbb{N}\ni(i,j)\mapsto 1$ if the ith Turing machine T_i will halt for input $j, h:(i,j)\mapsto 0$ otherwise, is noncomputable.[28] In fact, suppose that the contrary holds, i.e., that there exists an H, which computes h, and define a function $\bar{h}:x\mapsto 1$ if $h(x,x)=0$, being $\bar{h}(x)$ undefined otherwise.[29] The function $\bar{h}$ is computable by a Turing machine $\bar{H}$ obtained from H just by replacing 0 by 1 when H halts and outputs 0, and by entering an endless loop when H is ready to halt with output 1. Let $\bar{H}=T_{i(\bar{H})}$; if $\bar{h}(i(\bar{H}))=1$, then $h(i(\bar{H}),i(\bar{H}))=0$ and thus $\bar{H}$ should not halt for input $i(\bar{H})$. This is a contradiction. Similarly, if $\bar{h}(i(\bar{H}))$ is not defined, then $h(i(\bar{H}),i(\bar{H}))=1$ and thus $\bar{H}$ should halt for input $i(\bar{H})$: another contradiction. Therefore H cannot exist.

Another example was provided by Rado (1962) with the so-called Rado's Σ function: assume that the Turing machine has S states, $A=1$ symbols and that the input data is a completely blank tape. Then $\Sigma(S)$ is defined as the maximum number of 1's left on the tape after this S-state Turing machine halts. This type of Turing machine is now known as *the busy-beaver machine*. Busy-beaver Turing machines are difficult to find for two reasons (Shallit, 1998): first, the search space is extremely large—there are $[4(S+1)]^{2S}$ Turing machines with S states (for each nonhalting state there are two transitions out, so the total of transitions is $2S$, and each transition has two possibilities for the symbol being written, two possibilities $\gamma=L$, R for the direction to move, and $S+1$ possibilities for what state to go to, including the halting state). Second, due to the halting problem, it is in general not possible to determine whether a particular Turing machine will halt. We have to content ourselves with finding busy beavers for small S by a brute-force approach. In Table II we show the current status of this search. Another Rado's function $\Sigma'(S)$ appears, which is the maximum number of moves performed by the Turing machine before halting. Clearly $\Sigma'(S)\geq\Sigma(S)$.

In Fig. 12 we plot a flow diagram of a three-state busy beaver (Shallit, 1998). When this Turing machine starts

TABLE II. Busy-beaver Turing machines for small-S number of states. For $S=6$, $\Sigma(6)\geq 95\,524\,079$, $\Sigma'(6)\geq 8\,690\,333\,381\,690\,951$ (Marxen, 1997).

S	$\Sigma(S)$	$\Sigma'(S)$
1	1	1[a]
2	4	6[a]
3	6	21[a]
4	13	107[b]
5	≥4098	≥47 176 870[c]

[a]Lin and Rado (1965).
[b]Brady (1983).
[c]Marxen and Buntrock (1990).

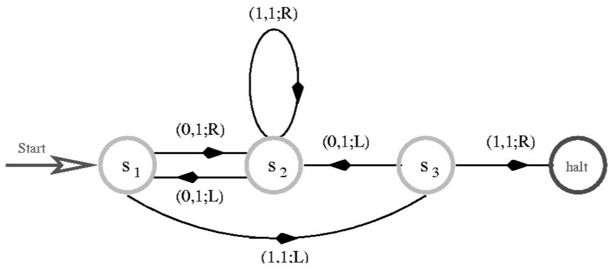


FIG. 12. A three-state "busy-beaver" Turing machine.

[28]Any form of input/output can be encoded into non-negative integers (Salomaa, 1989).

[29]Note that the same integer x singles out here both a Turing machine and an input.

with a tape completely blank of input data it executes 13 moves and writes six 1's. Thus $\Sigma(3)\geqslant 6$ and $\Sigma'(3)\geqslant 13$. Lin and Rado (1965) showed that for $S=3$ the $\Sigma(3)$ lower bound in fact yields the correct solution. From $S=5$ on, only lower bounds are known. For example, $\Sigma(8)>10^{44}$ (Rozenberg and Salomaa, 1994).

The proof that $\Sigma(S)$ is a noncomputable function goes by *reductio ad absurdum*. One shows that $\Sigma(S)$ grows with S faster than any computable function, i.e., if $F(S)$ is an arbitrary computable function, then there exists S_0 such that $\Sigma(S)>F(S)$ for $S\geqslant S_0$ (Shallit, 1998). As a by-product, $\Sigma'(S)$ is not computable either.

4. Other types of Turing machines

The Turing machines considered so far are deterministic: the instructions $i\in\mathcal{I}$ follow the transition rules in Eq. (54). It is possible to design other Turing machines called *nondeterministic* for which, given an initial pair (s_i,a_i), there exists a group of possible final triplets (Yan, 2000). This means that the transition mapping (54) is no longer a function, but a relation given by

$$(\mathcal{S},\mathcal{A})\rightarrow\text{subsets}(\mathcal{S},\mathcal{A};\gamma), \tag{55}$$

where subsets$(\mathcal{S},\mathcal{A};\gamma)$ denote all possible subsets of the Cartesian product $\mathcal{S}\times\mathcal{A}\times\gamma$. A *probabilistic Turing machine* is a type of nondeterministic Turing machine with some distinguished states called *coin-tossing states*. When the machine goes into one of these coin-tossing states, the control unit chooses between two possible legal next triplets in $\mathcal{S}\times\mathcal{A}\times\gamma$. The computation of a probabilistic Turing machine is deterministic except that in coin-tossing states the machine tosses an unbiased coin to decide between two possible legal next moves. The class of nondeterministic Turing machines is more powerful than the class of deterministic machines in the sense that anything computable with a Turing machine is also computable with a nondeterministic Turing machine and is usually faster. A nondeterministic Turing machine is closer to the idea of a quantum computer, but still it is far from one of them, as we shall see in Sec. IX.

The Turing machines introduced so far are irreversible: given the output of a computation we cannot generally reconstruct the input data. A reversible Turing machine is one for which the input determines the output and conversely, the output determines the input. More explicitly, to each Turing machine M we can associate a directed configuration graph $\Gamma(M)$: each node of the graph is a possible configuration $C\in\mathcal{S}\times\mathcal{A}$, and two nodes C,C' are arc connected when there is some instruction $i\in\mathcal{I}$ of M bringing C to C' in a single computation step.

A Turing machine M is reversible if and only if its graph of configurations $\Gamma(M)$ has only nodes with the number of incoming and outgoing lines $\leqslant 1$.

We know that a nonreversible Turing machine has a number of outgoing lines $\leqslant 1$. It is apparent that requiring the number of incoming lines $\leqslant 1$ implies that M can be executed in reverse deterministically, since every configuration has only one possible predecessor.

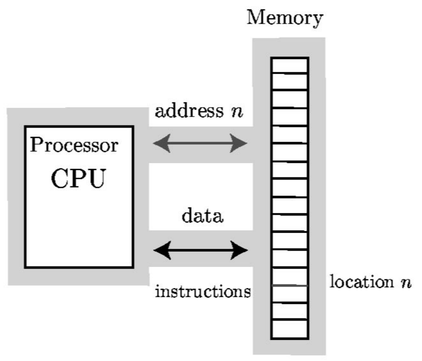


FIG. 13. von Neumann machine.

Lecerf (1963) and Bennett (1973) independently showed that an irreversible Turing machine can be simulated with a reversible Turing machine, at the expense of extra computer space and time. This is a remarkable fact for quantum computing, since a quantum Turing machine must be reversible (see Sec. IX).

Not only did Turing devise a theoretical computer, but he also pursued the practical construction of one of them. At the end of the war Turing was invited by the National Physical Laboratory in London to design a computer. His report proposing the Automatic Computing Engine (ACE) was submitted in March 1946 (Turing, 1946). Turing's design was at that point an original detailed design and prospectus for a computer in the modern sense. The size of storage he planned for the ACE was regarded by most who considered the report as hopelessly overambitious, and there were delays in the project's being approved. In the long run, the National Physical Laboratory design made no advance and other computer plans at Cambridge and Manchester took the lead. One year earlier, von Neumann had pushed forward another project for constructing a computer machine.

B. The von Neumann machine

The foundations of von Neumann's work on computers were laid down in the "First Draft of a Report on the EDVAC," written in the spring of 1945 and distributed to the staff of the Moore School of Engineering at the University of Pennsylvania, where the Electronic Discrete Variable Automatic Computer (EDVAC) was originally developed, in late June (Aspray, 1990). It presented the first written description of the *stored-program* concept and explained how a stored-program computer processes information. von Neumann collaborated with Mauchly and Eckert on the design for the EDVAC.

We can summarize the functioning of an ordinary computer by saying that it does one single thing at a time. von Neumann was the first to formalize the principles of a "program-registered calculator" based in the sequential execution of the programs registered in the memory of the computer. This is called a *von Neumann machine* (VNM). A VNM has the following parts, which are depicted in Fig. 13.

- *Processor*: The active part of the computer in which the information contained in the programs is processed step by step. It is in turn divided into three main parts:

  (i) *Control Unit*: The unit that controls all the parts of the computer in order to carry out all the operations requested by other parts, such as extracting data from the memory, executing and interpreting instructions, etc.
  (ii) *Registers*: A very fast memory unit inside the processor, which contains that part of the data currently being processed.
  (iii) *Arithmetic and logic unit*: This unit is devoted to the real computations such as sums, multiplications, logic operations, etc., executed on the data supplied by the registers or memory upon demand by the control unit.

- *Memory*: The part of the computer devoted to the storage of the data and instructions to be processed. It is divided into individual cells, which are accesible by means of a number called the *address*.

The functioning of a von Neumann machine is cyclic. One of these cycles contains the following operations: the control unit reads one program instruction from the memory, which is executed after being decoded. Depending on the type of instruction, a piece of data can either be read from or written into the memory, or an instruction can be executed. In the next cycle to be performed, the control unit reads another program instruction, which is precisely next in the memory to the one processed in the previous cycle.

It is the simplicity of this sequentially operating model that makes it advantageous for many purposes because it facilitates the design of machines and programs.

C. Classical parallelism

There are complex problems that demand a very large number of operations to be performed as well as a large amount of computer resources. These problems include image processing such as satellite images, meteorological predictions, scientific calculations arising in strongly correlated many-body systems, computation of the hadronic spectrum in quantum chromodynamics on the lattice, real-time calculations in plasma physics, turbulence in fluids, and many more. It was soon noticed that an ordinary computer based on the VNM architecture would take a very long time to cope with problems in which a massive number of operations must be performed.

A classical parallel computer is the natural way to address these problems. The idea of parallelism may be simply summarized as doing many things at a time. We shall see that a quantum computer would realize this goal at the highest possible degree of parallelism.

Although the idea of parallelism is very simple to state, its practical implementation has faced many obstacles for several reasons that we shall briefly describe. This will be quite illustrative later when we refer to the principles of quantum computation.

The way to extend the sequential von Neumann machine into a parallel computer is not unique. The components entering a parallel machine are already present in the VNM, but their number and organization differ. One way to understand the various possibilities is by recalling the organization of a program in any computer. A program is divided into instructions and data. These are its building blocks. This distinction means that we may have several degrees of parallelism depending on how many instructions and/or data the parallel machine handles at a time. This leads to a first classification of parallel machines known as *Flynn's classification* (Flynn, 1966, 1972), which describes in four categories how a computer functions without reference to the details of its architecture.

(i) *Single instruction stream, single data stream* (*SISD*): This executes one instruction at a time (single instruction stream) and fetches/stores one data value at a time (single data stream). It has only one CPU. Example: the von Neumann machine (specifically, processors like Motorola, Intel, and AMD).
(ii) *Multiple instruction stream, single data stream* (*MISD*): This corresponds to multiple programs operating on the same data (performing different computations) Example: none is available. This category does not seem to be useful.
(iii) *Single instruction stream, multiple data stream* (*SIMD*): This executes one instruction at a time (single instruction stream) and the same operation is performed on many data values at the same time (multiple data stream). Example: the vector machines like Thinking Machine's Connection Machine CM-2. A vector operation with n elements can be executed by one instruction cycle on a SIMD parallel machine.
(iv) *Multiple instructions stream, multiple data stream* (*MIMD*): These are multiprocessor systems, with each processor executing a different program on its own data. Thus there are multiple instruction streams (programs) and multiple data streams. Example: most distributed memory parallel processors, like Thinking Machine's Connection Machine CM-5, Cray T3D, IBM SP-2, and workstation clusters fit in this category.

Of these machines, those of the SIMD and MIMD types are parallel machines, the latter having a higher degree of parallelism. In Fig. 14 we show a schematic representation of Flynn's classification. Only processors and memory units are represented, without going into

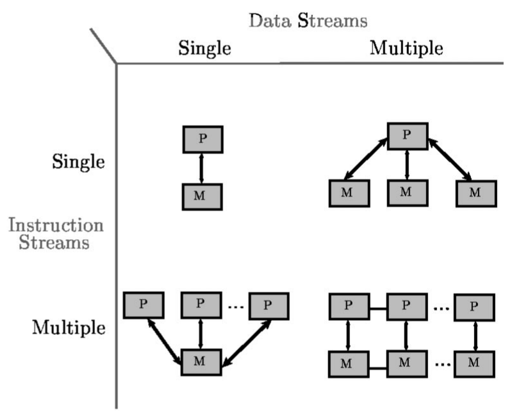


FIG. 14. Flynn's classification of parallel machines (P = processor, M = memory).

finer details about the interconnection network, types of memories (shared, distributed, cached, . . .), pipelines, etc.[30]

At first glance it may appear that what counts in a parallel machine is simply the number of processors. However, what really matters is the way the many processors are organized and how the information is exchanged among them. This is because for two processors to intercommunicate, it is necessary that they be synchronized, and consequently they have to wait for each other. Thus, the functioning of a parallel machine is slowed down if only the number of processors is increased without taking care of their organization.

We therefore arrive at the conclusion that to scale up a parallel machine one has to multiply the number of processors and to provide as well interconnecting structures for them. These structures or networks need be regular, efficient, and low cost. The determination of the best interconnecting network for the processors in a parallel machine is especially crucial when their number increases considerably.

For an interconnecting network (or lattice) to be good it has to minimize at the same time the total number of physical connections (or links) and the average distance between processors. This average distance is measured in terms of the number of connections to be traversed. Furthermore, the network has to be regular enough to allow scalability when more processors are added.

In order to understand these requirements let us enumerate and analyze some typical networks.

- *Fully connected lattice*: This is an extreme case that is made up of, say, N processors in such a way that all of them are connected to one another, as shown in Fig. 15. The number of connections is $\frac{1}{2}N(N-1)$, and thus it is of order $O(N^2)$. This fact makes it impractical because there are other more economical alternatives for connections.

[30]Flynn's classification is too coarse for classifying multiprocessor systems, and there exist modifications to it (Hwang and Briggs, 1985) and new ones, as well, like *Händler's classification* (Händler, 1982) and others.

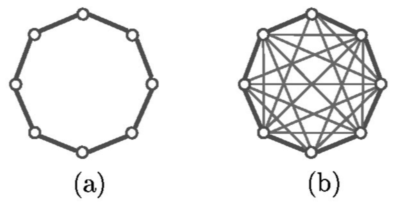


FIG. 15. Ring vs fully connected processor lattices.

- *Ring lattice*: The network of processors forms a ring (see Fig. 15), which has the advantage of needing only two connections per processor, no matter their number. In this sense it is the opposite of the full lattice. However, it has a very important disadvantage, because in the worst case a message has to traverse $N/2$ processors (half of the lattice) to reach its destination. This is also impractical when N is large.
- *Binary tree*: The processors are organized such that each node is connected to three nodes, namely, one *parent* and two *children* (Fig. 16). The problem with this type of lattice is that the inner nodes deep inside the tree are very badly connected among themselves.
- *Hypercube*: This is the solution that has turned to be optimal in meeting the desired requirements (Fig. 17). In the simplest possibility, one processor is installed at each vertex of the cube, which can be of any dimension D. In the familiar case of a $D=3$ cube, each processor is connected to three others and, more importantly, each one is at a maximum distance of three connections from any other. For a D-dimensional hypercube the number of processors is 2^D; each one is connected to D neighbor processors and is at most a distance D apart from any other. The most famous parallel machine based on this hypercube architecture is the original Connection Machine and the Crays. It is not surprising that Feynman, who played a paramount role in the beginning of quantum computers, worked on the design of this parallel machine and made some notable contributions (Hillis, 1998).

The interconnecting networks of processors considered so far are called static because the structure is fixed by construction. There also exists the possibility of dynamic networks where the configuration is changeable.

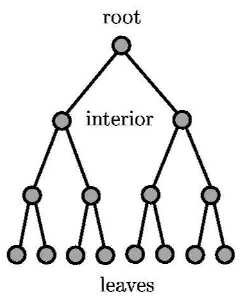


FIG. 16. Binary tree processor lattice.

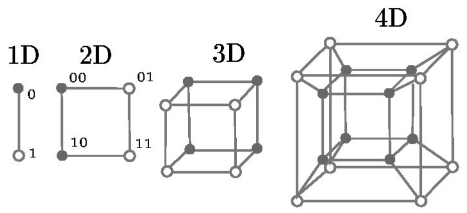


FIG. 17. Hypercube networks.

In this case the processors are connected not directly but through commuters that can be switched in different ways.

One of the fundamental problems posed by the parallel computer is its control. There are several strategies for addressing this issue. One possibility is to have a central processor working as a control unit for the rest of processors, as in the SIMD. This is a model of centralized control in which the control unit sends instructions to the other processors which never interfere with the central processor. In order to simplify their working, generally the same instruction is sent to all the processors, which in turn operate on different sets of data. This mode of control has the same disadvantage as the original von Neumann machine: it is slow. The control unit has to send many electrical pulses to perform the control task.

An alternative to centralized control consists in allowing each processor to make its own decisions, usually consulting only its nearest-neighbor processors. This solution also has difficulties because the programs must be written in a way very different from the standard. Moreover, such decentralized control can become very inefficient because the processors might spend most of their time exchanging messages rather than making computations.

The problem of organizing and controlling the parallelism in a classical computer very much resembles organization problems in human societies, which is as open a problem there as for networks of computers. We shall see in Sec. IX that in a quantum computer one also faces similar synchronization problems, and we shall discuss how they are solved in terms of physical principles.

D. Classical logic gates and circuits

A Turing machine is by no means a practical computer, despite being a powerful theoretical machine. In practice, computers are made of electronic circuits, which in turn contain *logic gates*. A logic gate is a device that implements a classical logic operator like the AND operator. A logic operator or function f is an application $f:\{0,1\}^n \mapsto \{0,1\}^m$, which maps an input of n bit-valued operands into an m-bit-valued output. When the target space of f is $\{0,1\}$, one usually says that f is a *Boolean operator* or function. Boolean calculus is useful for elucidating situations that can be true or false, making appropriate reasonings to draw conclusions correctly. They are therefore helpful in building practical computers and in programming. Furthermore, it is possible to show that classical Turing machines are equivalent to classical logic circuits. This means that they both have the same complexity classes. This is a mathematical result that legitimates the use of electronic circuits in the construction of real computers.

TABLE III. Truth tables for the basic logic operators: NOT ($\overline{\ }$), AND ($\wedge$), OR ($\vee$).

x	$\overline{x}$	x	y	$x\wedge y$	$x\vee y$
0	1	0	0	0	0
1	0	0	1	0	1
		1	0	0	1
		1	1	1	1

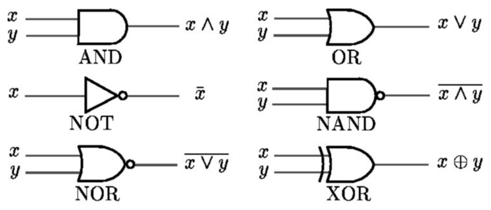


FIG. 18. Basic classical logic gates.

Before stating this important result as a theorem, let us take a closer look at some rudiments of Boolean logic that will also help in understanding the peculiarities of quantum logic gates (see Sec. IX).

An operator with one operand is called a *unary operator*, with two operands is a *binary operator*. There are three basic Boolean or logic operators: (1) The unary operator NOT: $x \mapsto \text{NOT}x := \overline{x} := 1-x$, also denoted by overlining the argument ($\overline{\ }$); (2) the binary operator AND: $(x,y) \mapsto x$ AND $y := x\wedge y := xy$, also denoted by $\wedge$; and (3) the binary operator OR, $(x,y) \mapsto x$ OR $y := x\vee y := x+y-xy$, denoted also by $\vee$. As usual, Boolean arithmetic is done in the field $\mathbb{Z}_2$: $1+1=0$.

The action of a logic operator is represented by a *truth table*. A truth table contains as many columns as input operands and ouput bits, and $2^{\#\text{operands}}$ rows. The inputs are shown on the left, and the outputs are shown on the right. The truth tables for the basic operators are shown in Table III. An important Boolean expression involving two variables x,y is $r=(\overline{x}\wedge y)\vee(x\wedge\overline{y})$, i.e., $r(x,y)=x+y$.[31] Boolean expressions can be represented by *logic circuits*. A logic circuit is a directed acyclic graph with incoming lines carrying input Boolean variables $x_1,x_2,\ldots,x_n$ and outgoing lines carrying the output variables $y_1,\ldots,y_m$ of the circuit. Every node in the graph is a logic gate that represents a logic operator of Boolean algebra. In real computers, circuits consist of electronic devices such as switches and wires.

To each logic operator we can associate a logic gate with a specific form. That logic gate has a number of incoming lines, one per input operand, and outgoing lines for the output result. In Fig. 18 we show the con-

[31]This r corresponds to the XOR operation.

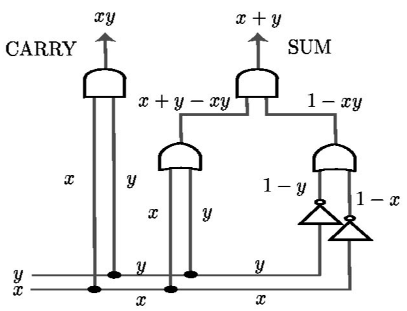


FIG. 19. A classical logic circuit: adder for two bits x,y. The bifurcating wires at the nodes are achieved with FANOUT gates.

vention for the basic logic gates. In the same way as the basic operators of algebra make up more complicated expressions, the basic gates are combined to construct complex circuits.

Additional gates that duplicate the input values on wires are frequently needed. These are called FANOUT or COPY gates and are schematically represented by —•< (see Fig. 19). In classical computation, these are obvious gates, for they simply correspond to splitting the wire into two or more leads, which is an easy operation. This is why they are usually taken for granted throughout classical computing. Nevertheless, these FANOUT gates are logically necessary when discussing the important issue of the universality of classical gates. By contrast, these duplicating gates have no place in the insides of a quantum circuit due to the linearity of quantum mechanics (no-cloning theorem; see Sec. III).

A logic circuit computes a logic function in a natural way by following its directed path (usually from left to right) upon application of its constituent gates. The size of a circuit C is its number of gates, and the depth of C is the length of the longest directed path in it. A typical circuit is depicted in Fig. 19.

Suppose that we are given a tractable decision problem, i.e., a problem in class **P** (see the Appendix). This means that there exists a Turing machine M able to decide it $[M(x_n)=0,1]$ for initial data x_n of arbitrary length n, in polynomial time. This problem is said to have *polynomial circuits* when there is a family $\{C_1,\ldots,C_n,\ldots\}$ of logic circuits, of polynomial size in the input length n, such that $M(x_n)=0,1$ iff $C_n(x_n)=0,1$.

It can be shown that all problems in class **P** have polynomial circuits. The converse, however, is not true: there exist intractable decision problems that have polynomial circuits (Papadimitriou, 1994). This shortcoming is remedied by restricting the circuit family to be a *uniform circuit family*: for each n, the description of each C_n is an output of an auxiliary Turing machine in polynomial time when entered with an appropriate input of length n.[32]

TABLE IV. Truth tables for the logic operators NAND, NOR, XOR.

x	y	x NAND y	x NOR y	x XOR y
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	0	0	0

The equivalence between classical Turing machines and logic circuits is stated in the following theorem (Savage, 1972; Schnorr, 1976; Pippenger and Fisher, 1979; Papadimitriou, 1994).

Turing machines and uniform circuit families: A decision problem is in class **P**, i.e., it can be solved for inputs of length n by a Turing machine in polynomial time $p(n)$, if and only if it has a uniform family of polynomial circuits. Moreover, the minimum size of C_n is $O[p(n)\log p(n)]$.

This theorem legitimates the simulation of Turing machines by logic circuits. Dealing with gates and circuits is simpler and more practical than with Turing machines. Actually, gates are packaged into hardware chips.

So far we have introduced a set of three basic logic operators (NOT, AND, OR). It will also prove useful to introduce three additional gates: NAND, NOR, and XOR. The gates NAND and NOR are the negation of AND and OR, respectively. The gate XOR is called exclusive OR and is also denoted by $\oplus$. Their truth tables are shown in Table IV.

With the basic set {NOT, AND, OR} one can build any logic function, provided that FANOUT gates and ancilla or work bits are freely used. Because of this property, {NOT, AND, OR} is called a *universal set* of logic gates. However, this set is not minimal. To see this we use de Morgan's laws, which are the following Boolean identities:

$$\overline{(x\vee y)}=\bar{x}\wedge\bar{y},$$
$$\overline{(x\wedge y)}=\bar{x}\vee\bar{y}. \tag{56}$$

These two algebraic equations are dual each other. Negation of the first produces $x\vee y=\overline{(\bar{x}\wedge\bar{y})}$. This is telling us that OR gates are not essential: the AND and NOT gates can by themselves reproduce the functionality of the OR gate. Similarly, the second relation in Eq. (56) leads to $(x\wedge y)=\overline{(\bar{x}\vee\bar{y})}$, that is, AND gates can be implemented with OR and NOT gates. Then the set {AND, NOT} is universal, and so is the set {OR, NOT}.

Can we further reduce the number of elements in a universal set? The answer is yes. The surprising result is that NAND gates alone (or, similarly, NOR gates alone) are sufficient for constructing any circuit (up to FANOUT and work bits). We know this from the following simple laws:

$$\bar{x}=1 \ \text{NAND}\ x,$$
$$x\wedge y=\overline{(x\ \ \text{NAND}\ y)}=1\ \text{NAND}\ (x\ \ \text{NAND}\ y). \tag{57}$$

[32]Actually the auxiliary Turing machine should be $(\log n)$-space bounded, which implies polynomial time boundedness.

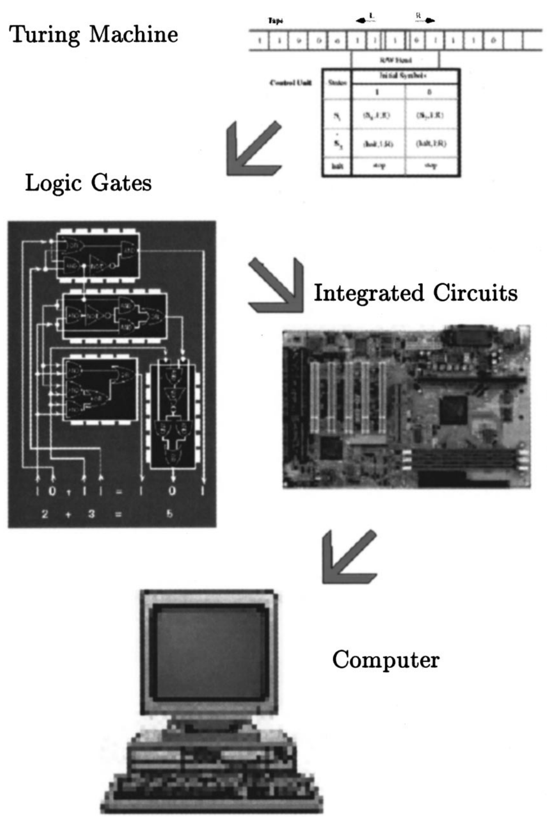


FIG. 20. From a Turing machine to a real computer.

Therefore we see that {NAND} (or {NOR}) can do everything that the set {AND, NOT} does, and hence {NAND} and {NOR} are also universal sets.

IX. PRINCIPLES OF QUANTUM COMPUTATION

In the previous section we described some basic aspects of Turing machines and their practical implementations by means of the von Neumann architecture. Yet it is a long way from there towards the construction of a real computer like those we have on our desks. In Fig. 20 we provide a visualization of the route we have to follow. This long route starts with the abstract notion of a classical computer embodied in a Turing machine. No real computer has a Turing machine inside. Instead, the operations carried out by a Turing machine can be replaced by logic gates. These logic gates can do sums, multiplications, logic operations, and the like. With just a few logic gates we can do almost none of the daily tasks we are used to nowadays. To get the power and speed of an ordinary computer we need millions of logic gates interconnected and integrated into tiny circuits or chips. Finally, these integrated circuits are arranged into the computer motherboard with other components, and along with a screen, keyboard, and mouse we have a universal machine capable of doing many tasks, like writing this article.

All four stages pictured in Fig. 20 have been accomplished in the case of classical computers. What is the current state of the art in the case of quantum computers? The first step in Fig. 20 has also been achieved for quantum computers. This is the topic of Sec. IX.A, where we discuss the notion of quantum Turing machines, the quantum version of the classical Turing machines introduced thus far. Moreover, the second step regarding the design of quantum logic gates has also been accomplished, as we shall explain in Sec. IX.B. These quantum gates are used as the basic components of a quantum computer to design quantum algorithms that surpass certain very important classical algorithms (see Sec. X). More important is the fact that, in recent years, an experimental realization of these quantum gates has been made (see Sec. XI), which allow us to cherish the possibility of building a real operative quantum computer on equal footing with the current classical precursors. However, to achieve this goal we need to move farther, to find the quantum equivalent of an integrated circuit (third step). This step amounts to the problem of scalability in a quantum computer: so far, the experimental realizations mentioned previously are made of only a few gates and, although a quantum gate is more powerful than a classical one, we also need a large number of them to perform nontrivial tasks. We need to scale up our current quantum technology. Finally, the fourth step will be to have a real operative quantum computer in our hands, with all the external devices to communicate with it. Although there is still a long way ahead to achieve this goal, the fact that the fundamental first and second steps have been already taken is very encouraging. In the following we shall describe these two steps for quantum computers.

From a fundamental point of view, a quantum computer is a quantum Turing machine, and this is a concept that we shall next define.

A. The quantum Turing machine

Several achievements led to the concept of a quantum Turing machine, and it is not our purpose to give a full account of all of them; instead we shall mention some of the most representative constructions or machines. The first of these was the model introduced by Benioff (1980, 1981, 1982). Benioff's goal was to use quantum-mechanical systems to construct reversible Turing machines. His motivation was that the unitary evolution of an isolated quantum system provides a way to implement reversible computations. The issue of reversibility had attracted much attention since Bennett (1973) constructed a classical model of a reversible computing machine equivalent to a Turing machine. Landauer (1961) had shown that reversible operations dissipate no energy, while a Turing machine as described in Sec. VIII generally performs irreversible changes during computations. Although Benioff's machine is a quantum machine, it is not a quantum computer, for it is equivalent to a reversible Turing machine. Feynman (1982) went one step further towards the notion of the quantum computer with his "universal quantum simulator." He proposed using quantum systems to simulate quantum

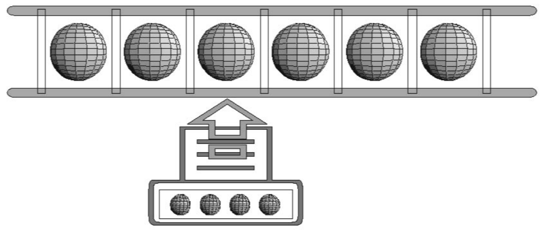


FIG. 21. Pictorial view of a quantum Turing machine. There are qubits (Bloch spheres, Fig. 2) in the tape and in the control unit.

mechanics more efficiently than do classical computers.[33] He showed (Feynman, 1985) that classical Turing machines exponentially slow down when simulating quantum phenomena, while a universal quantum simulator would do the job efficiently. However, Feynman's machine is not fully a quantum computer in the sense described below because it cannot be programmed to perform an arbitrary task.

Deutsch (1985) took the final step in the quest for a sensible definition of a quantum computer. His starting point is a critique of the Church-Turing hypothesis (see Sec. VIII.A), which he considers very vague as compared to physical principles such as the gravitational equivalence principle. Deutsch proposes to make more concrete the statement "functions that would naturally be regarded as computable" in the Church-Turing hypothesis. He identifies such functions as those which can be computed by a real physical system. This is quite apparent, since it is hard to believe that something can be naturally computable if it cannot be computed in Nature. Thus Deutsch goes on to transform the Church-Turing hypothesis into a physical principle, which he states as the *Church-Turing Principle*: Every finitely realizable physical system can be perfectly simulated by a universal model computing machine operating by finite means.

The content of this principle is more physical than the corresponding hypothesis, since it appeals to objective concepts such as measurement and physical systems instead of the subjective notion of "naturally computable." The "finite means" in the Church-Turing principle are more general and expand the role of the Turing machine in the corresponding hypothesis (Sec. VIII.A).

Deutsch follows a natural way to introduce the definition of a quantum Turing machine. Starting from the knowledge we have of its classical counterpart (see Sec. VIII.A), he replaces some of the classical components of an ordinary Turing machine, like bits, by quantum elements, like qubits (see Fig. 21).

A *quantum Turing machine* is a finite-state machine that has three components: a finite processor, an infinite memory unit (of which only a finite portion is ever used), and a cursor. The description of these components is as follows:

(i) *Finite Processor*: This is the control unit as in a Turing machine but it consists of a finite number P of qubits. Let us denote the Hilbert space of these processor states as

$$\mathcal{H}_{\mathrm{P}} := \mathrm{span}\{\otimes_i |p_i\rangle : p_i = 0,1\}_{i=0}^{P-1}. \tag{58}$$

(ii) *Memory Tape*: This has a similar functionality to that in a Turing machine but it consists of an infinite number of qubits.[34] Let us denote the Hilbert space of these memory states as

$$\mathcal{H}_{\mathrm{M}} := \mathrm{span}\{\otimes_i |m_i\rangle : m_i = 0,1\}_{i=-\infty}^{+\infty}. \tag{59}$$

(iii) *Cursor*: This is the interacting component between the control unit and the memory tape. Its position is scanned by a variable $x \in \mathcal{H}_{\mathrm{C}} = \mathbb{Z}$, and the associated Hilbert space is

$$\mathcal{H}_{\mathrm{C}} := \mathrm{span}\{|x\rangle : x \in \mathbb{Z}\}. \tag{60}$$

Therefore there is a Hilbert space of states associated with a quantum Turing machine that altogether takes the form

$$\mathcal{H}_{\mathrm{QC}} := \mathcal{H}_{\mathrm{C}} \otimes \mathcal{H}_{\mathrm{P}} \otimes \mathcal{H}_{\mathrm{M}}. \tag{61}$$

The basis vectors in the Hilbert space $\mathcal{H}_{\mathrm{QC}}$ of the quantum Turing machine are of the form

$$|x;\mathbf{p};\mathbf{m}\rangle := |x;p_0,p_1,\ldots,p_{\mathrm{P}};\ldots,m_{-1},m_0,m_1,\ldots\rangle, \tag{62}$$

and are called the computational basis states.

We may wonder about the relationship between the defining features of a classical Turing machine (see Sec. VIII.A) and those of a quantum Turing machine. The set of states $\mathcal{S}$ corresponds to the Hilbert space $\mathcal{H}_{\mathrm{P}}$ of states in a quantum Turing machine. The alphabet $\mathcal{A}$ is just the qubit space $\mathbb{C}^2$. As for the set of instructions $\mathcal{I}$ of a Turing machine, we need to specify the way a quantum Turing machine works.

A quantum Turing machine operates in steps of fixed duration T, and during each step only the processor and a finite part of the memory unit interact via the cursor. We stress that a quantum Turing machine, much like a Turing machine, is a mathematical construction; we shall present explicit experimental realizations of equivalent quantum circuits in Sec. XI.

The set of instructions $\mathcal{I}$ of a Turing machine is replaced by the unitary time evolution of the quantum states $|\Psi\rangle \in \mathcal{H}_{\mathrm{QC}}$. After a number $n \in \mathbb{N}$ of computational steps, the state will be transformed into

$$|\Psi(nT)\rangle = U^n |\Psi(0)\rangle, \tag{63}$$

with U a unitary evolution operator, $UU^\dagger = U^\dagger U = 1$. A valid quantum program takes a finite number of steps n. With each quantum Turing machine there is associated a unitary evolution operator U to perform a certain job or

[33]Manin (1980) had already envisaged that the complexity of quantum systems surpassed the capabilities of classical computers.

[34]Even if ideally there is a qubit per cell, only a finite number of them are active during each running of the quantum Turing machine.

program, much as a Turing machine has a unique set of instructions $\mathcal{I}$, and each Turing machine performs a certain task. To specify the initial state $|\Psi(0)\rangle$, we set to zero both the cursor position $x=0$ and the prepared processor states $\mathbf{p}=\mathbf{0}$. The memory states $\mathbf{m}$ are prepared allocating the input data and other program instructions, conveniently encoded into a finite number of qubit strings, with the rest of the memory qubits set to $|0\rangle$. The initial state is then

$$|\Psi(0)\rangle=\sum_{\mathbf{m}} c_{\mathbf{m}}|0;\mathbf{0};\mathbf{m}\rangle, \text{ with } \sum_{\mathbf{m}} |c_{\mathbf{m}}|^2=1. \tag{64}$$

The notion of a quantum Turing machine operating "by finite means" according to the Church-Turing principle means that the machine cannot do infinitely many operations at a given time nor at arbitrary positions along the memory tape. This notion suggests the following constraint on the matrix elements of the evolution operator:

$$\begin{aligned}\langle x';\mathbf{p}';\mathbf{m}'|U|x;\mathbf{p};\mathbf{m}\rangle \\ &=[\delta_{x',x+1}U^{+}(\mathbf{p}',m'_{x'}|\mathbf{p},m_x) \\ &\quad+\delta_{x',x-1}U^{-}(\mathbf{p}',m'_{x'}|\mathbf{p},m_x)]\prod_{x'\neq x\pm1}\delta_{m_{x'},m_x}.\end{aligned} \tag{65}$$

In these matrix elements, the infinite product guarantees that only a finite number of memory qubits participate in a single computational step. Once the qubit at the xth cursor position is singled out, the two deltas appearing in the brackets guarantee that the cursor position cannot change by more than one unit, either backward, forward, or both. This operating mode amounts to locality in the tape space. We call the parts $U^{\pm}(\mathbf{p}',m'_{x\pm1}|\mathbf{p},m_x)$ of U forward and backward matrices at x, respectively. They represent the operators $P_{x\pm1}UP_x$ in the computational basis, where P_x is the projection onto the Hilbert subspace of $\mathcal{H}_{\rm QC}$ consisting of the states with the cursor at the xth position. Unitarity of U is equivalent to $U^{\pm\dagger}U^{\mp}=0, U^{+\dagger}U^{+}+U^{-\dagger}U^{-}=1$. Each unitary operator $U\{U^{-},U^{+}\}$ defines a quantum Turing machine.

As with any other computer, we need a mechanism to cause the machine to halt when the computation ends. In a quantum machine there is a severe constraint to do this because the principles of quantum mechanics do not allow us to observe or measure the machine's operation until it terminates. To know when this happens, we may set aside one of the qubits of the processor to signal the end. Let us choose the first qubit $|q_0\rangle$ to acquire the value 1 when the computation is over while it is 0 during the operations. The program does not interact with $|q_0\rangle$ until it has reached the end. Thus the state $|q_0\rangle$ can be monitored periodically from the outside without affecting the operation of a quantum Turing machine.

So far we have set up several analogies between the components of quantum and classical Turing machines. To complete this comparison, we can also think about the relationships concerning their functioning. Does a quantum Turing machine somehow extend the notion of a classical Turing machine? Yes, and this relation turns out to be very physical and will sound familiar to us. First, not all classical Turing machines are closely related to a quantum Turing machines, only reversible classical Turing machines will be, as follows from the discussion above. It is possible for a quantum Turing machine to reproduce the functioning of a reversible classical Turing machine (Deutsch, 1985) if we choose its unitary evolution operator to have the following form:

$$\begin{aligned}U^{\pm}(\mathbf{p}',m'_{x\pm1}|\mathbf{p},m_x)&=\delta_{\mathbf{p}',\mathbf{A}(\mathbf{p},m_x)}\delta_{m'_{x\pm1},B(\mathbf{p},m_x)} \\ &\quad\times\tfrac{1}{2}[1\pm C(\mathbf{p},m_x)],\end{aligned} \tag{66}$$

where $\mathbf{A},B,C$ are maps of $\mathbb{Z}_2^P\times\Pi_{-\infty}^{+\infty}\mathbb{Z}_2$ into $\mathbb{Z}_2^P,\mathbb{Z}_2$ and $\{-1,1\}$, respectively.

This form of dynamics guarantees that this particular quantum Turing machine will remain in a computational basis state (62) at the end of each time step. This is precisely the way a classical Turing machine operates. The requirement of reversibility is guaranteed by demanding that the mapping $(\mathbf{p},m)\mapsto(\mathbf{A}(\mathbf{p},m), B(\mathbf{p},m),C(\mathbf{p},m))$ be bijective.

Therefore there is a particular limiting case in which a quantum Turing machine becomes a reversible classical Turing machine. This fact is somewhat reminiscent of the familiar correspondence principle of quantum mechanics to recover classical mechanics. This principle played a fundamental role in the development of the old quantum theory and the beginnings of modern quantum mechanics. Here we are following a similar path by starting with a revision of the classical fundamentals of information and computation to develop their quantum versions.

1. Quantum parallelism

The ability of a quantum Turing machine to be in several computational basis states at the same time is called *quantum parallelism* and is one of the defining features of a quantum Turing machine. The classical counterpart of this is the notion of classical parallelism introduced in Sec. VIII.C. The quantum version of doing many things at a time in a classical parallel computer is the possibility of being in many states at a time in a quantum computer. Furthermore, in a classical computer it is not enough to have a large number of processors connected in parallel in order to perform computations efficiently. It is also necessary to have all of them appropriately synchronized to avoid message jams and disruptive functioning of the several processors, which would not operate coherently. Likewise, quantum parallelism is not enough for a successful quantum computation. Recall that the result of a quantum computation is probabilistic. There is not a 100% certainty that after measuring the final output state it will contain the correct result for which we are searching. We need to repeat the measurement several times in order to retrieve the correct value of the function or procedure for which the computer was devised. If we program the quantum computer carelessly, this number of measurements would be exponentially large, and all the potential advantages of quantum parallelism would be spoiled. What do we need to make

TABLE V. Principles of quantum computation.

Computer science		Quantum physics
1st quantum parallelism	=	superposition principle
2nd quantum programming	=	constructive interference

good quantum programs? We need to reduce the number of trials to just a few. This fact will depend on how the evolution operator $U\{U^+,U^-\}$ and the initial memory states $|\mathbf{m}\rangle$ are prepared. In order to become good quantum programmers we must be smart enough to devise them in such a way that the maxima of the probability distribution in the output state correspond to the desired result, while the rest of the possible results, which are useless for the purpose of our computation, must be somehow damped. We recognize this pattern of behavior for the unitary operator $U\{U^+,U^-\}$ as the phenomenon of constructive interference of amplitudes in quantum mechanics. The typical example is the two-slit experiment.

We shall present explicit examples of how quantum parallelism and constructive interference work together when we deal with quantum algorithms in Sec. X. Here, we summarize these correspondences between classical parallel and quantum computers as follows:

Classical parallel computers

(i) many things at a time

(ii) synchronization of many processors

$$\updownarrow$$

Quantum Computer

(i) many states at a time

(ii) constructive interference of many states

The quantum version of parallelism exceeds the classical one, for whereas in a quantum computer it is possible to have an exponentially large number of available states within a reduced space, this capacity seems unreachable in any known classical parallel computer.

In quantum mechanics there are some basic principles, such as the correspondence principle, Heisenberg's uncertainty principle, or Pauli's principle, that encode the fundamentals of that theory. The knowledge of those principles provides us with the essential understanding of quantum mechanics at a glance, without going into the complete formalism of that subject. A similar thing happens with other areas in physics. In computer science there are guiding principles for the architecture of a computer (hardware) and the programs to be run (software). Likewise, in quantum computing we have seen that there are basic principles associated with the ideas of quantum parallelism and quantum programming. It is useful to synthesize the relationships between quantum computation and the principles of quantum physics in the form of basic principles, as shown in Table V.

By principles of quantum computation we mean those rules that are specific to the act of computing according to the laws of quantum mechanics. In this table we indicate that the quantum version of parallelism is realized through the superposition principle of quantum mechanical amplitudes; likewise the act of programming a quantum computer should be closely related to constructive interference of those amplitudes involved in the superposition of quantum states in the registers of the computer. We shall see these principles in action when studying quantum algorithms (see Sec. X) that supersede their classical counterparts. The superposition principle, when applied to multipartite quantum systems like those of a quantum register [see Eq. (71) below], yields the notion of entanglement (see Secs. III.A and III.E).

2. Universal quantum Turing machine

The notion of a universal Turing machine can also be extended to quantum Turing machines. A standard quantum Turing machine is capable of performing only the job for which it has been set up. This is so because the unitary operator $U\{U^+,U^-\}$ and the memory quantum states $|\mathbf{m}\rangle$ are chosen to do one specific task. Deutsch (1985) has shown that the elements $U\{U^+,U^-\}$ and $|\mathbf{m}\rangle$ of a quantum Turing machine can be devised to simulate with arbitrary precision any other quantum computer. This is the concept of a *universal quantum Turing machine* or programmable quantum computer. We now give more explicit details about how such a machine is programmed.

Let f be any function that we want to compute with the universal quantum Turing machine and let $\pi(f)$ be a quantum program to do the job. The quantum computer will take the program $\pi(f)$ and a given input value i and then compute the desired value $f(i)$. This process is implemented as follows. There exists an integer n_{fin} such that

$$U^{n_{\text{fin}}}|0;\mathbf{0};\pi(f),i,\mathbf{0}\rangle=|0;1,\mathbf{0};\pi(f),i,f(i),\mathbf{0}\rangle, \qquad (67)$$

where the halting qubit is set to $|1\rangle$ after the computation ends. In this expression we assume that the initial quantum memory states are

$$|\mathbf{m}_{\text{in}}\rangle=|\pi(f),i,\mathbf{0}\rangle, \qquad (68)$$

while the final memory states contain the answer $f(i)$:

$$|\mathbf{m}_{\text{fin}}\rangle=|\pi(f),i,f(i),\mathbf{0}\rangle. \qquad (69)$$

If in Eq. (67) we focus only on the memory states, then we can use a shorthand notation for the unitary evolution,[35] namely,

$$|\pi(f),i,j\rangle\mapsto|\pi(f),i,j\oplus f(i)\rangle. \qquad (70)$$

Although a quantum Turing machine has an infinite-dimensional memory space, much like a classical Turing machine, we remark that only a finite-dimensional unitary transformation need be applied at every step of the computation to simulate the associated evolution.

[35]See Sec. IX.C for more on quantum function evaluation.

The concept of a quantum Turing machine has many implications, which we shall continue to explore. Most of these implications amount to a revision of the typical areas of classical computation in light of the new principles of computation. For instance, we can now address how the theory of complexity will be affected. In Sec. VIII.A we mentioned that this theory deals with the issue of what a computer can do. Namely, it studies not only which function can be computed, but also how fast and how many memory resources are needed. This scheme must be modified for conversion into a quantum theory of complexity. In this new theory we must pose another question: with what probability can a quantum computer achieve a certain task? See the Appendix for details.

B. Quantum logic gates

The quantum Turing machine is a basic model for quantum computation. However, it is not a practical starting point for designing a quantum computer, in much the same way as a classical Turing machine is not a handy computer.

A key step towards the realization of a practical quantum computer is to decompose its functioning into the simplest possible primitive operations or gates. The identification of universal logic gates, such as NAND, in classical computers (see Sec. VIII.D) was of great help in the development of the field. A universal gate such as NAND operates locally on a very reduced number of bits (actually, two). However, by combining NAND gates in the appropriate number and sequence we can carry out arbitrary computations on arbitrarily many bits. This was very useful in practice for it allowed device engineers to focus on creating only a few devices, leaving the rest to the circuit designer. The same rationale applies to a quantum computer and the relation of a quantum Turing machine to quantum circuits.

When a quantum computer is working, it is a unitary evolution operator that is effecting a predetermined action on a series of qubits. These qubits form the memory register of the machine, or a *quantum register*. A quantum register is a string of qubits with a predetermined finite length. The space of all the possible register states makes up the Hilbert space of states associated with the quantum computer. If $\mathcal{H}$ is the Hilbert space of a single qubit and $|\Psi_i\rangle \in \mathcal{H}$, $i=1,2$, a given basis state, then a basis vector $|\Phi\rangle$ for the states of the quantum register is a tensor product of qubit states

$$|\Phi\rangle = |\Psi_1\rangle \otimes |\Psi_2\rangle \otimes \cdots \otimes |\Psi_n\rangle \in \mathcal{H}^{\otimes n}. \tag{71}$$

A quantum memory register can store multiple sequences of classical bits in superposition. This is a manifestation of quantum parallelism.

A *quantum logic gate* is a unitary operator acting on the states of a certain set of qubits. If the number of such qubits is n, the quantum gate is represented by a $2^n \times 2^n$ matrix in the unitary group $\mathrm{U}(2^n)$. It is thus a reversible gate: we can reverse the action, thereby recovering the initial quantum state from the final one. Generically a quantum logic gate can have any finite number of input qubits, but in practice we shall be interested in gates that are elementary for quantum computation, and those have a small number of input qubits. Diagrammatically a quantum gate is represented by a "black box," wherein the operation takes place, and a number of input (output) lines, used to wire up a set of gates, equal to the number of qubits involved in the computation (see Fig. 22). Let us see more explicitly how quantum gates look by considering some representative gates in increasing order of complexity.

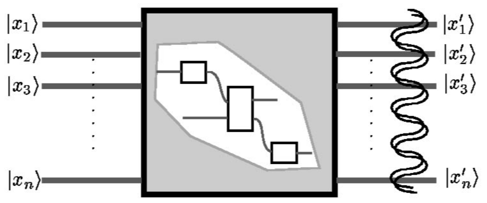


FIG. 22. A generic quantum logic gate. The wavy lines indicate that the output state is a generic superposition of product quantum states.

1. One-qubit gates

These are the simplest possible gates because they take one input qubit and transform it into one output qubit. The quantum NOT gate is a one-qubit gate. Its unitary evolution operator U_{NOT} is [Eq. (11)]

$$U_{\mathrm{NOT}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{72}$$

The truth table and the diagram representing this gate are shown in Table III and Fig. 23, respectively. We see that this quantum NOT gate coincides with its classical counterpart. However, there is a basic underlying difference: the quantum gate acts on qubits while the classical gate operates on bits. This difference allows us to introduce a truly quantum one-qubit gate: the $\sqrt{\mathrm{NOT}}$ gate. Its matrix representation is

$$U_{\sqrt{\mathrm{NOT}}} := \frac{1}{\sqrt{2}} e^{\mathrm{i}\pi/4}(1 - \mathrm{i}\sigma_x). \tag{73}$$

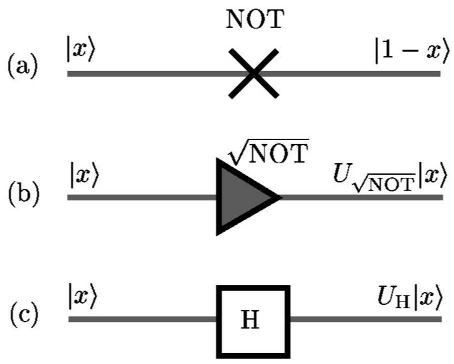


FIG. 23. Quantum unary gates: (a) NOT gate; (b) $\sqrt{\mathrm{NOT}}$ gate; (c) Hadamard gate.

TABLE VI. Truth table for the quantum CNOT gate.

x	y	x'	y'
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

This gate, when applied twice, gives NOT. Explicitly

$$U_{\sqrt{\mathrm{NOT}}}U_{\sqrt{\mathrm{NOT}}}=\begin{pmatrix}\frac{1+\mathrm{i}}{2} & \frac{1-\mathrm{i}}{2}\\ \frac{1-\mathrm{i}}{2} & \frac{1+\mathrm{i}}{2}\end{pmatrix}\cdot\begin{pmatrix}\frac{1+\mathrm{i}}{2} & \frac{1-\mathrm{i}}{2}\\ \frac{1-\mathrm{i}}{2} & \frac{1+\mathrm{i}}{2}\end{pmatrix}$$
$$=\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}=U_{\mathrm{NOT}}. \qquad (74)$$

This gate has no counterpart in classical computers since it implements nontrivial superpositions of basis states.

Another one-qubit gate without analog in classical circuitry and heavily used in quantum computations is the so-called Hadamard gate H (see Sec. III). It is defined as

$$U_{\mathrm{H}}=\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}. \qquad (75)$$

2. Two-qubit gates

The XOR (exclusive-OR), or CNOT (controlled-NOT) gate, is an example of a quantum logic gate on two qubits [Eq. (12)]. It is instructive to give the unitary action $U_{\mathrm{XOR,CNOT}}$ of this gate in several forms. Its action on the two-qubit basis states is

$$U_{\mathrm{CNOT}}|00\rangle=|00\rangle,\quad U_{\mathrm{CNOT}}|10\rangle=|11\rangle,$$
$$U_{\mathrm{CNOT}}|01\rangle=|01\rangle,\quad U_{\mathrm{CNOT}}|11\rangle=|10\rangle. \qquad (76)$$

From this definition we see that the name of this gate is quite apparent, for it means that it executes a NOT operation on the second qubit conditioned to have the first qubit in the state $|1\rangle$. Its matrix representation is

$$U_{\mathrm{CNOT}}=U_{\mathrm{XOR}}=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}. \qquad (77)$$

The action of the CNOT operator (76) immediately translates into a corresponding truth table as in Table VI. The diagrammatic representation of the CNOT gate is shown in Fig. 24.

We shall see how this quantum CNOT gate plays a paramount role in both the theory and experimental realization of quantum computers. It allows the implementation of conditional logic at a quantum level.

Unlike the CNOT gate, there are two-qubit gates with no classical analog. One example is the *controlled-phase gate* or CPHASE:

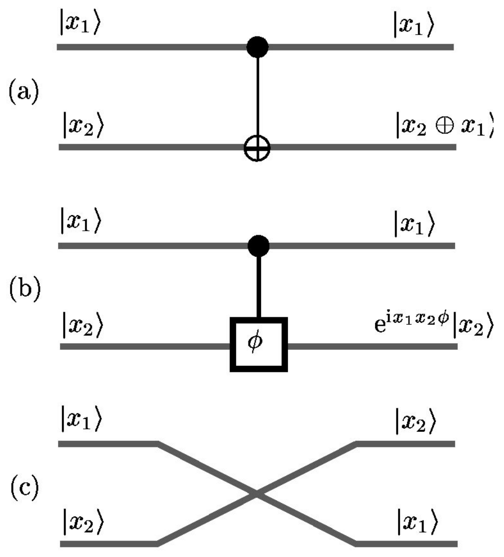


FIG. 24. Quantum binary gates: (a) CNOT gate; (b) CPHASE gate; (c) SWAP gate.

$$U_{\mathrm{CPh}(\phi)}:=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{i\phi}\end{pmatrix}. \qquad (78)$$

It implements a conditional phase shift $e^{i\phi}$ on the second qubit.

An important result is that we can reproduce the CNOT gate with a controlled-phase gate of $\phi=\pi$ and two Hadamard transforms on the target qubits as shown in Fig. 25. This is a simple consequence of the relation

$$U_H\sigma_z U_H=\sigma_x. \qquad (79)$$

Other interesting two-qubit gates are the SWAP gate, which interchanges the states of the two qubits, and the $\sqrt{\mathrm{SWAP}}$ gate, whose matrix representations are

$$U_{\mathrm{SWAP}}:=\begin{pmatrix}1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\end{pmatrix},$$

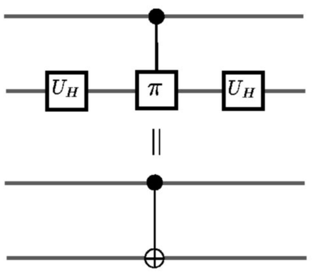


FIG. 25. Relation between CNOT gate and controlled phase using Hadamard gates.

TABLE VII. Truth table for the Toffoli gate.

x	y	z	x'	y'	z'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

$$U_{\sqrt{\mathrm{SWAP}}}:=\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1+\mathrm{i}}{2} & \frac{1-\mathrm{i}}{2} & 0 \\ 0 & \frac{1-\mathrm{i}}{2} & \frac{1+\mathrm{i}}{2} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{80}$$

3. Three-qubit gates

An immediate extension of the CNOT construction to three qubits yields the CCNOT gate (or C^2NOT),[36] which is also called Toffoli gate T (Toffoli, 1981). The matrix representation is a one-qubit extension of the CNOT gate, namely,

$$U_{\mathrm{CCNOT}}=U_{\mathrm{T}}:=\begin{pmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&1&0 \end{pmatrix}. \tag{81}$$

The associated truth table is shown in Table VII. The first two input qubits x,y are copied to the first two output qubits x',y' (see Fig. 26), while the third output qubit z' is the XOR of the third input z and the AND of the first two inputs x,y.

The Deutsch gate $\mathrm{D}(\theta)$ (Deutsch, 1989) is also an important three-qubit gate. It is a controlled-controlled-S or C^2S operation (see Fig. 26), where

$$U_{S(\theta)}:=ie^{-\mathrm{i}1/2\,\theta\sigma_x}=i\cos\tfrac{1}{2}\theta+\sigma_x\sin\tfrac{1}{2}\theta \tag{82}$$

is a unitary operation that rotates a qubit about the x axis by an angle θ and then multiplies it by a factor i. We require θ to be incommensurate to π, that is, not a rational multiple of π. Two properties follow: (1) Let $|q\rangle$ be a given qubit. Then for any fixed value of $\alpha\in\mathbb{R}$ we can get arbitrarily close to $e^{i\alpha\sigma_x}|q\rangle$ by successive application of $U_{S(\theta)}$ to $|q\rangle$ a finite number of times. (2) The Deutsch gate generates as closely as needed the Toffoli gate. This is because the C^2S^n gate is just the D^n gate. Since we can make $\frac{1}{4}(n\theta/\pi-1)$, with $n=4k+1$, as near to a given arbitrary integer as desired, D^n will thereby closely approach the Toffoli gate.

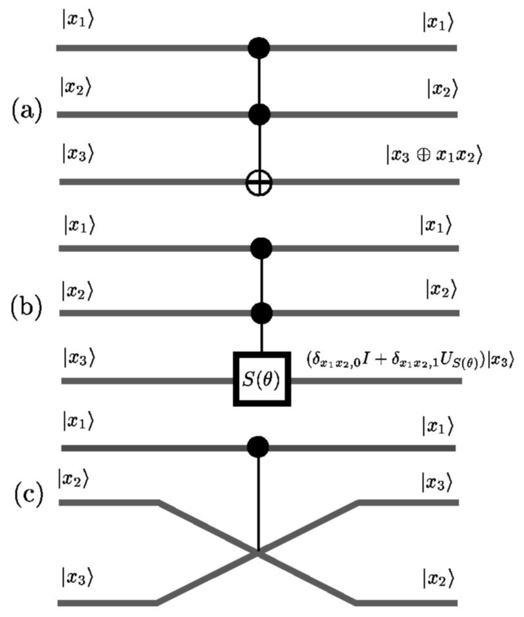


FIG. 26. A set of three-qubit gates: (a) Toffoli gate; (b) Deutsch gate; (c) Fredkin gate.

Another instance of a three-qubit gate is the Fredkin gate F (Fredkin and Toffoli, 1982). It is a controlled SWAP operation, schematically shown in Fig. 26 and represented by the matrix

$$U_{\mathrm{F}}=\begin{pmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&1 \end{pmatrix}. \tag{83}$$

Needless to say, these unitary linear gates act not only on the individual basis states, but also on any linear combination of them.

We have enumerated a series of quantum logic gates whose use and importance will be explained in the following sections. We shall address the experimental implementation of some of these quantum gates in Sec. XI.

C. Quantum circuits

The simple gates introduced in the previous section can be assembled into a networklike arrangement that

[36]Controlled-controlled-not gate.

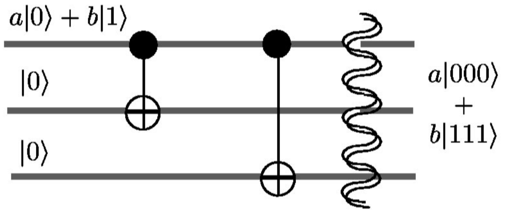


FIG. 27. An example of a quantum circuit implementing a Greenberger-Horne-Zeilinger state.

enables us to perform more complicated quantum operations than those initially carried out by those gates. This is the basic idea of a *quantum circuit*. Deutsch (1989) generalized the classical reversible circuit model to produce the idea of quantum circuits. A quantum circuit is a computational network composed of interconnected elementary quantum gates.

The following example illustrates a simple use of a quantum circuit. Let us prepare initially a one-qubit state as an arbitrary superposition of the logical states $|0\rangle,|1\rangle$, namely,

$$|\psi_0\rangle = a|0\rangle + b|1\rangle. \tag{84}$$

We want to obtain a final state of the Greenberger-Horne-Zeilinger type [Eq. (22)]:

$$|\psi_f\rangle = a|000\rangle + b|111\rangle. \tag{85}$$

To this end, instead of writing a sequence of algebraic operations, we can simply arrange the following quantum circuit using the CNOT gate as pictured in Fig. 27.

Quantum circuits are widely used in quantum computation, where most of the problems can be formulated in terms of them. Indeed, standard quantum mechanics might be flooded with quantum circuits in the future, something similar to what happened with Feynman diagrams in quantum field theory. This is because quantum circuits are able to condense graphically much more information than the use of several formulas. Besides, this form of presenting and reasoning is closer to what experimental physicists really do with their devices.

In Sec. VIII.D we presented the basic result that a classic Turing machine is equivalent to a classical logic circuit. In quantum computing there is a similar result due to Yao (1993) showing that a quantum Turing machine is equivalent to a quantum circuit. This theorem justifies replacing the more complicated study of quantum Turing machines by that of quantum circuits, which are simpler to analyze and design. In fact, experimental approaches to quantum computers are presented in terms of quantum circuits (see Sec. XI).

Let K be a quantum logic circuit with n input qubits. Suppose that $|\Psi_x\rangle = \Sigma_{y\in\{0,1\}^n} c_x(y)|y\rangle$ is the final quantum state of K for an input $x\in\{0,1\}^n$. The distribution generated by K for the input x is defined as the map $p_x : y\in\{0,1\}^n \mapsto |c_x(y)|^2$. The quantum circuit K is said to (n,t)-*simulate* a quantum Turing machine Q if the family of probability distributions p_x, $x\in\{0,1\}^n$, coincides with the probability distributions of the Q configurations after t steps with input x.[37] Yao's theorem is the following statement.

Quantum Turing machines and quantum circuits: Let Q be a quantum Turing machine and n,t positive integers. There exists a quantum logic circuit K of polynomial size in n,t, that (n,t)-simulates Q.

This result implies that quantum circuits can mimic quantum Turing machines in polynomial time and vice versa. Thus quantum circuits provide a sufficient model for quantum computation that is easier to implement and manipulate than quantum Turing machines. This situation goes in parallel with similar results about classical logic circuits and Turing machines (Sec. VIII.D). From now on when talking about a quantum computer we shall usually refer to an underlying equivalent quantum circuit.

1. Universal quantum gates

After the works of Deutsch (1989) and Yao (1993) the concept of a universal set of quantum gates became central in the theory of quantum computation. A set $\mathcal{G} := \{G_{1,n_1}, \ldots, G_{r,n_r}\}$ of quantum gates G_{j,n_j} acting on n_j qubits, $j=1,\ldots,r$, is called universal if any unitary action U_N on N input quantum states can be decomposed into a product of succesive actions of G_{j,n_j} on different subsets of the input qubits. More explicitly, given any U_N acting unitarily on N qubits, there exists a sequence $S_1,S_2,\ldots,S_s$ of subsets of $\{1,2,\ldots,N\}$, with $n_{S_1},\ldots,n_{S_s}$ elements, and a map $\pi:\{1,2,\ldots,s\}\to\{1,2,\ldots,r\}$ such that $n_{\pi(j)} = n_{S_j}$, $\forall j$, and

$$U_N = U_{N,G_{\pi(s)},S_s} \cdots U_{N,G_{\pi(1)},S_1}. \tag{86}$$

Here

$$U_{N,G_{\pi(j)},S_j} := I_{\{1,2,\ldots,N\}-S_j} \otimes U_{G_{\pi(j)},S_j}, \tag{87}$$

where $I_{\{1,2,\ldots,N\}-S_j}$ is the identity on the qubits not in S_j, and $U_{G_{\pi(j)},S_j}$ stands for the unitary action of the gate $G_{\pi(j)}$ on the Hilbert space of the n_{S_j} qubits in the set S_j.

For instance, a generic unitary $k\times k$ matrix of dimension $k\geqslant 2$ can be represented as the product of $k(k-1)/2$ two-level unitary matrices (Reck *et al.*, 1994).

This notion of a universal set of gates is exact because the generic transformation U_N is reproduced exactly in terms of a finite number of elements in $\mathcal{G}$. We denote this situation by writing the universal set as $\mathcal{G}_{\text{ex}}$. However, this notion is too strong. Dealing with practical quantum devices, it is not conceivable to work with a set of gates implementing any other gate with perfect accuracy. Thus we are inevitably led to work with approximate simulations of gates. Underlying this idea is the concept of distance between two unitary gates.

[37]We assume that a given configuration is encoded as a list of the tape symbols from cell $-t$ to t, followed by the state and the position of the cursor, all encoded as strings of qubits (see Sec. IX.A).

A quantum gate U_N is said to be approximated by another gate U'_N with error $<\epsilon$ when the distance $d(U_N, U'_N) := \inf_{\theta\in\mathbb{R}} ||U_N - e^{i\theta}U'_N||$ between both matrices as projective operators is $<\epsilon$.[38,39] This means that if the gate U_N is replaced by gate U'_N in a quantum circuit K, then the unit rays of the associated output states will differ in norm by at most ϵ.[40]

With this definition, we also introduce the notion of an *approximate set* of universal quantum gates as before but with the weaker requirement that it simulates any other quantum gate in an approximate sense. We denote these sets as $\mathcal{G}_{\rm ap}$, and by universality we shall mean it in this sense henceforth, unless the exact notion is explicitly indicated.

Some examples of universal sets of quantum gates, to be discussed next, are the following (for a more mathematical and general approach, see Brylinski and Brylinski, 2001):

(1) $\mathcal{G}^{\rm I}_{\rm ex} := \{U_2 : U_2 \in {\rm U}(2^2)\}$ (DiVincenzo, 1995).

(2) $\mathcal{G}^{\rm II}_{\rm ex} := \{U_1, \textsc{cnot} : U_1 \in {\rm U}(2)\}$ (Barenco, Bennett, *et al.*, 1995).

(3) $\mathcal{G}^{\rm III}_{\rm ap} := \{{\rm D}\}$, Deutsch gate [Eq. (82)] (Deutsch, 1989).

(4) $\mathcal{G}^{\rm IV}_{\rm ap} := \{{\rm C}^2U, {\rm C}^2W\}$, with $U(\alpha) := R_y(4\pi\alpha) = e^{-i2\pi\alpha\sigma_y}, W(\alpha) := {\rm diag}(1, e^{i2\pi\alpha})$, α an irrational root of a degree-2 polynomial (Aharonov, 1998).

(5) $\mathcal{G}^{\rm V}_{\rm ap} := \{{\rm H}, {\rm CPh}(\pi/2)\}$, Eqs. (75) and (78) (Solovay, 1995; Kitaev, 1997; Cleve, 1999).

(6) $\mathcal{G}^{\rm VI}_{\rm ap} := \{{\rm H}, W, \textsc{cnot}\}$, with $W := {\rm diag}(1, e^{i\pi/4})$ (Cleve, 1999).

Of these examples, (1) and (2) correspond to infinite sets of universal gates. However, a practical quantum computer must have a set with a finite number of universal gates. Examples (3)–(6) are finite suitable cases. Although with a finite set of gates we are limited to simulating a countable subset of all possible quantum gates, it is possible to reproduce an arbitrary gate within a given small error ϵ. Moreover, a finite universal set $\mathcal{G}_{\rm ap}$ is closer to the spirit of the Church-Turing principle stating that a computing machine must operate by finite means (Sec. IX.A).

A first example of a three-qubit universal gate is the Deutsch gate (Deutsch, 1989),[41] which is an extension of the Toffoli gate $U_{\rm CCNOT}$ [Eq. (81); Toffoli, 1981] for classical logic circuits. Toffoli gates are exactly universal for reversible (classical) circuits.[42] Deutsch showed that his gate $D(\theta_0)$ with a fixed angle θ_0 that is an irrational multiple of π is universal.

A further improvement in the analysis of universal quantum gates was provided by DiVincenzo (1995) who showed that the set of two-qubit gates is exactly universal for quantum computation. This is a remarkable result, since it is known that its classical analog is not true: classical reversible two-bit gates are not sufficient for classical computation. The NAND gate, although binary, is not reversible.

After DiVincenzo's result it was shown that a large subclass of two-qubit gates are universal (Barenco, 1995) and moreover that almost any two-qubit gate is universal.

The reduction from three to two qubits amounts to a large simplification in the analysis of quantum circuits and in their experimental implementation. It is much simpler to deal with two-body quantum interactions than with a three-body problem.

The race towards reducing the number of necessary qubits in the elementary gates culminated with the joint work of Barenco, Bennett, *et al.* (1995), in which it is shown that even one-qubit gates are enough for quantum computation (in the exact sense) provided they are combined with the CNOT gate. This result, another manifestation of the superposition principle, is quite surprising, since in classical computation the classical CNOT is not universal.

We shall refer to this important result as the *universality theorem of elementary quantum gates*. The proof of this result (Barenco, Bennett, *et al.*, 1995) can be simply stated in terms of quantum circuits and it has three parts. First, we need to prove that with one-qubit gates plus CNOT it is possible to generate any controlled-unitary two-qubit gate. Second, this result is extended to a controlled-unitary gate with an arbitrary number of qubits. And third, one applies these results to construct any unitary gate with one-qubit and CNOT gates.

Part 1. The proof of the first part is contained in the identity between quantum circuits shown in Fig. 28. In the lower part we show a controlled-unitary CU gate of two qubits associated with a unitary 2×2 matrix U. The upper part shows its decomposition in terms of one-qubit gates U_1, U_2, U_3, E and CNOT's. The rationale of this decomposition comes from group theory: any unitary 2×2 matrix U can be decomposed as

[38]The norm $\|A\|$ of the (finite) matrix A is usually defined as $\sup_{x:\|x\|=1}\|Ax\|$. Other norms are topologically equivalent to it.

[39]A compactness argument shows that the infimum in the definition of d is attainable, i.e., $\exists\,\theta_0$ such that $d(U_N, U'_N) := \|U_N - e^{i\theta_0}U'_N\|$. From now on, we shall assume that the phase factor is included in the approximating unitary operator U'_N.

[40]The unit ray of a state vector $|\phi\rangle$ is the set $[\phi] := \{e^{i\theta}|\phi\rangle : \theta \in \mathbb{R}\}$. A distance between unit rays can be defined as ${\rm dist}([\phi_1],[\phi_2]) = \inf_{\theta\in\mathbb{R}}\|\phi_1 - e^{i\theta}\phi_2\|$, which justifies the presence of a phase factor in the notion of an appproximate gate.

[41]Previously Deutsch (1985) had already given a universal set of eight 2×2 matrices.

[42]To see that C^{2}NOT is classically universal, notice that: (1) $\textsc{not}(x_3) = [\textsc{ccnot}(1,1,x_3)]_3$; (2) $\textsc{and}(x_1,x_2) = [\textsc{ccnot}(x_1,x_2,0)]_3$. Now apply the result (Sec. VIII.D) that $\{\textsc{and},\textsc{not}\}$ is a classical universal set. See in addition that the COPY operation is also reproduced as $\textsc{copy}(x_2) = [\textsc{ccnot}(1,x_2,0)]_{2,3}$.

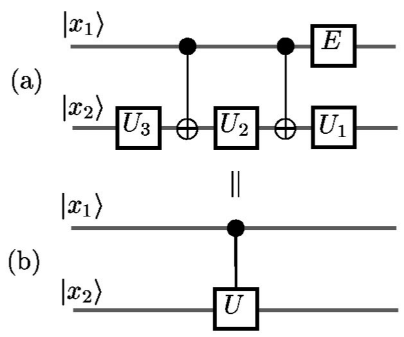


FIG. 28. Decomposition of an arbitrary two-qubit CU gate into one-qubit gates and CNOT's. The symbol E denotes the gate $E{:}|0\rangle\mapsto|0\rangle,|1\rangle\mapsto e^{i\delta}|1\rangle$.

$$U=\mathrm{Ph}(\delta)\overline{U},\quad \overline{U}:=R_z(\alpha)R_y(\beta)R_z(\gamma)\in\mathrm{SU}(2), \tag{88}$$

where δ is the phase (mod π) of the U(1) factor of U(2), and α,β,γ are the Euler angles parametrizing the SU(2) matrix $\overline{U}$. More explicitly,

$$\mathrm{Ph}(\delta)=\begin{pmatrix} e^{i\delta} & 0\\ 0 & e^{i\delta}\end{pmatrix},\quad R_z(\alpha)=\begin{pmatrix} e^{-i(\alpha/2)} & 0\\ 0 & e^{i(\alpha/2)}\end{pmatrix},$$

$$R_y(\beta)=\begin{pmatrix} \cos\frac{\beta}{2} & -\sin\frac{\beta}{2}\\ \sin\frac{\beta}{2} & \cos\frac{\beta}{2}\end{pmatrix},$$

$$R_z(\gamma)=\begin{pmatrix} e^{-i(\gamma/2)} & 0\\ 0 & e^{i(\gamma/2)}\end{pmatrix}. \tag{89}$$

With the help of this decomposition we can further show that for any unitary matrix $\overline{U}$ in SU(2) there exist matrices U_1,U_2,U_3 in SU(2) such that

$$U_1U_2U_3=1,$$
$$U_1\sigma_xU_2\sigma_xU_3=\overline{U}. \tag{90}$$

The proof for this is by construction, namely,

$$U_1=R_z(\alpha)R_y(\tfrac{1}{2}\beta),$$
$$U_2=R_y(-\tfrac{1}{2}\beta)R_z[-\tfrac{1}{2}(\alpha+\gamma)], \tag{91}$$
$$U_3=R_z[\tfrac{1}{2}(-\alpha+\gamma)].$$

Now the equivalence between the quantum circuits of Fig. 28 proceeds by considering the two possibilities for the first qubit.

(i) $|x_1\rangle=|0\rangle$. In this case the CNOT gates are not operative and using Eq. (90) we find that the second qubit $|x_2\rangle$ is not altered.

(ii) $|x_1\rangle=|1\rangle$. In this case the CNOT gates do act on the second qubit producing the chain of operations $\mathrm{Ph}(\delta)U_1\sigma_xU_2\sigma_xU_3|x_2\rangle$, which using Eq. (90) turns out to be $U|x_2\rangle$. Recall that the controlled-σ_x gate is CNOT.

Part 2. The proof of the second part is represented in Fig. 29 by another identity between quantum circuits. The proof is by induction on the number of qubits. We illustrate the simplest case. In the lower part we show a controlled-controlled-unitary C$^2U^2$ gate of three qubits associated with the square of an arbitrary unitary 2×2 matrix U. The upper part shows its decomposition in terms of controlled two-qubit gates (which in turn were already decomposed into one-qubit gates and CNOT's in the first part) and CNOT's.

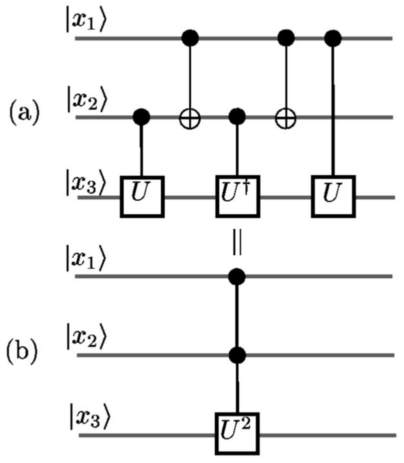


FIG. 29. Building up a controlled-controlled-U^2 three-qubit gate from elementary gates.

The proof of this equivalence proceeds by considering the possible actions on the third qubit depending on the state of the other two qubits.

(i) $|x_1\rangle=|0\rangle$. In this case, the two CNOT gates become inactive and so does the second controlled-U gate. We have two possibilities: (a) if $|x_2\rangle=|0\rangle$ then neither of the remaining controlled gates operate and the net result is to leave $|x_3\rangle$ unchanged; (b) if $|x_2\rangle=|1\rangle$ then the effect is now $U^\dagger U|x_3\rangle=|x_3\rangle$, as before.

(ii) $|x_1x_2\rangle=|10\rangle$. Now the CNOT gates do operate on the second qubit $|x_2\rangle$, and the second controlled-U gate acts on the third qubit. However, the first U gate is inactive. Thus the first CNOT gate changes the state of $|x_2\rangle$ to $|1\rangle$ and this makes the $U^\dagger$ gate become operative. Later the action of the second CNOT brings the second qubit back to $|0\rangle$. Altogether, the final effect on $|x_3\rangle$ is to yield $UU^\dagger|x_3\rangle=|x_3\rangle$ and remains unchanged again.

(iii) $|x_1x_2\rangle=|11\rangle$. In this case we need to produce the action of U^2 on the third qubit. Now all the gates in Fig. 29 become operative and we make a sequential counting of their effects. As $|x_2\rangle=|1\rangle$, the first U gate does operate on the third qubit. Next, the action of the first CNOT gate sets $|x_2\rangle=|0\rangle$ so that the $U^\dagger$ gate becomes inactive. Then the second CNOT gate puts the second qubit back to $|1\rangle$. Altogether, the final effect on $|x_3\rangle$ is to yield $UU|x_3\rangle=U^2|x_3\rangle$, as required.

Finally, we can always choose the initial matrix U as the square root of a unitary matrix, say $U^2=V$, such that the output in Fig. 29 is a C^2V gate. For instance, if we choose $U=e^{i\pi/4}R_x(\frac{1}{2}\theta)$ we reproduce the Deutsch gate [Eq. (82)].

Moreover, we can go on and provide a construction of an arbitrary C^nV transformation (useful in quantum algorithms) by extending the construction in Fig. 29 to an arbitrary number of qubits. For instance, for a C^3U^2 gate of four qubits we would have another qubit line on Fig. 29(b); the construction then holds by adding only a similar line to Fig. 29(a) so that the two CNOT gates become CCNOT (C^2NOT) gates and the last CU gate also picks up another control qubit gate. In general, for an n-qubit $C^{n-1}U^2$ gate that has $n-1$ control qubits and one target qubit where U^2 acts, the construction in Fig. 29 is generalized by simply using generalized C^{n-2}NOT gates with $n-2$ control qubits and a last $C^{n-2}U$ gate with $n-2$ control qubits. The proof of this generalized construction follows straightforwardly.

Part 3. Combining the results of Parts 1 and 2 with the previously known construction of an arbitrary unitary matrix U as a product of two-level (not necessarily one-qubit) unitary matrices of Reck *et al.* (1994), one can easily represent U through one-qubit and CNOT gates, in this way concluding the proof that one-qubit gates plus CNOT is a set of elementary gates for exact universal computation (Barenco, Bennett, *et al.*, 1995).

So far we have cared only about the possibility of reconstructing a generic quantum gate from a given set of gates. The complexity of these constructions, measured by the number of basic gates necessary to achieve a certain gate simulation, is of great interest.

As an example of this issue, it is also interesting to count how many elementary gates in $\mathcal{G}_{\rm ex}^{\rm II}$ are needed to simulate a general C^nU gate. For instance, for a C^2U gate the first part of the proof yields four one-qubit gates and two CNOT's. For a generic controlled gate of n control qubits C^nU, the second part of the proof yields a quadratic dependence on n. To see this, let us denote by C_n the cost (in number of gates) of simulating a C^nU gate. From the first part of the proof we know that the cost of simulating the U and $U^\dagger$ gates in Fig. 29 is order $\Theta(1)$;[43] moreover, it is not difficult to show that the cost of the two C^{n-1}NOTs is $\Theta(n+1)$ (Barenco, Bennett, *et al.*, 1995). The cost of the generalized $C^{n-1}U$ gate is C_{n-1}. Altogether, the cost of a gate satisfies a recursion relation like this:

$$C_n = C_{n-1} + \Theta(n+1), \tag{92}$$

whose solution yields $C_n = \Theta[(n+1)^2]$.

What is the size (number of gates) for exactly simulating an arbitrary gate of n qubits in $U(2^n)$? Barenco, Bennett, *et al.* (1995) showed that using the universal set $\mathcal{G}_{\rm ex}^{\rm II}$ this cost is $O(n^3 4^n)$;[44] Knill (1995) reduced this bound to $O(n4^n)$.

However, we are also interested in the efficiency of the approximate simulation of a generic gate. The universality property of a set of gates $\mathcal{G}_{\rm ap}$ means that, given an arbitrary quantum gate $U \in \mathrm{U}(2^n)$ and $\epsilon > 0$, we can always devise an approximate quantum gate U' generated by $\mathcal{G}_{\rm ap}$ such that $d(U,U') < \epsilon$. The errors scale up linearly with the number of gates: given N gates U_i and their approximations U_i', then the telescopic identity

$$U_1 \cdots U_N - U_1' \cdots U_N' = \sum_{1 \le k \le N} U_1' \cdots U_{k-1}' (U_k - U_k') U_{k+1} \cdots U_N$$

yields immediately $\| U_1 \cdots U_N - U_1' \cdots U_N' \| < N\epsilon$.

This construction can be done efficiently using poly$(1/\epsilon)$ gates from the universal set (Lloyd, 1995; Preskill, 1998). Although we shall not prove it, the underlying reasons are simple: (1) any universal set generates unitary matrices having eigenvalues with phases incommensurate relative to π; (2) if $\theta/\pi \in \mathbb{R}$ is irrational, then the integral powers $e^{ik\theta}, k \in \mathbb{Z}$ are dense in the unit circle S_1, and given $\epsilon > 0$, any $e^{i\alpha} \in S_1$ is within a distance ϵ of some $e^{in\theta}$ with $n = O(1/\epsilon)$.

As a matter of fact, we can do much better than approximating a given n-qubit gate with circuits of size poly$(1/\epsilon)$ in the universal set $\mathcal{G}_{\rm ap}$. A theorem of Solovay and Kitaev shows that an exponentially improved approximation is possible (Solovay, 1995; Kitaev, 1997): Let $\mathcal{G}_{\rm ap}$ be an arbitrary finite universal set of gates, i.e., $\mathcal{G}_{\rm ap}$ generates a dense subset in $\mathrm{U}(2^n)$. Then any matrix $U \in \mathrm{U}(2^n)$ can be approximated within an error ϵ by a product of $O\{\mathrm{poly}[\log(1/\epsilon)]\}$ gates in $\mathcal{G}_{\rm ap}$ (more precisely, $O\{\mathrm{poly}[\log(1/\epsilon)]\} = O[\log^c(1/\epsilon)]$, with $c \approx 2$). The idea of the proof is to construct thinner and thinner nets of points in $\mathrm{U}(2^n)$ by taking group commutators of unitaries in previous nets. It turns out that in this way the width of the resulting nets decreases exponentially.

Finally, when the above Solovay-Kitaev theorem is combined with the complexity for exactly simulating gates with $\mathcal{G}_{\rm ex}^{\rm II}$ and the linearity of the error propagation with the number of gates, it immediately follows that any unitary gate $U \in \mathrm{U}(2^n)$ can be approximated to within error ϵ with $O[n4^n \log^c(n4^n/\epsilon)]$ gates in any $\mathcal{G}_{\rm ap}$. Note that this represents an exponential complexity in the number of qubits, i.e., most gates will be hard to simulate.

2. Arithmetic with quantum computers

The universality theorem of elementary quantum gates is a central result in the theory of quantum computation because it reduces the implementation of conditional quantum logic to a small set of simple operations. However, with a computer we are typically interested in doing arithmetic operations and thus we need to know how to perform quantum arithmetic with universal quantum gates. Vedral, Barenco, and Ekert (1996) provided efficient ways of doing arithmetic operations such as addition, multiplication, and modular exponentiation building on the Toffoli gate. The key point in their constructions is that we have to preserve the coherence of quantum states and make those operations reversible, unlike in a classical computer. For in-

[43]One writes $y = \Theta(x)$ to denote that both $y = O(x)$ and $x = O(y)$ hold simultaneously.

[44]The factor n^3 arises from the cost $O(n)$ to bring a generic two-level matrix to a C^{n-1}-unitary matrix which in turn costs $O(n^2)$. The dominant factor 4^n counts asymptotically the maximum number of two-level unitary factors in the Reck *et al.* decomposition.

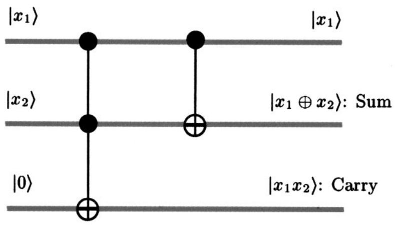


FIG. 30. The quantum addition from a Toffoli gate.

stance, the AND operation of Sec. VIII.D can be made reversible by embedding it into a Toffoli gate (Ekert, Hayden, and Inamori, 2000): setting the third qubit to zero in Eq. (81) we get

$$U_{\text{CCNOT}}|x_1,x_2,x_3=0\rangle = |x_1,x_2,x_1\wedge x_2\rangle. \tag{93}$$

Similarly, the quantum addition can be embedded into a Toffoli gate as shown in Fig. 30 with the help of a CNOT gate for the first two qubits. The result of the addition mode 2 is stored in the second qubit whereas, the third qubit carries the bit necessary to complete addition in base 2.

Quantum multiplication can be implemented in a similar fashion, as can exponentiation modulo N (Vedral, Barenco, and Ekert, 1996). This latter operation is central in the Shor algorithm (Sec. X.D).

Another important operation that must be implemented in a quantum circuit is the evaluation of a function f. This must again comply with the requisite of reversibility, which is accomplished with a U_f gate as shown in Fig. 31, where U_f is a unitary transformation that implements the action of f on certain qubits of the circuit. In this figure the box representing the evaluation of the gate is a kind of black box, also called a *quantum oracle*, which represents the way in which we call or evaluate the function f. These evaluations are also called queries.

Reversible implementation of f requires splitting the quantum register storing an initial state $|\Psi_0\rangle$ into two parts: the *source register* and the *target register*, namely,

$$|\Psi_0\rangle = |\Psi_s\rangle \otimes |\Psi_t\rangle, \tag{94}$$

where $|\Psi_s\rangle$ stores the input data for the computation and $|\Psi_t\rangle$ stores the output data, that is, the results of the quantum evolution or application of logic gates.

Thus, in order to implement a Boolean function $f:\{0,1\}^m \to \{0,1\}$ in a quantum circuit, we need the action of a unitary gate U_f acting on the target register as follows:

$$U_f|x_1x_2\cdots x_m\rangle_s|x_{m+1}\rangle_t = |x_1x_2\cdots x_m\rangle_s|x_{m+1}\oplus f(x_1,x_2,\ldots,x_m)\rangle_t. \tag{95}$$

Why is it not possible to evaluate directly the action of f by a unitary operation that evolves $|x\rangle$ into $|f(x)\rangle$? The answer lies in unitarity of computation: we know that orthonormality is preserved under unitary transformations; thus if f is not a one-to-one mapping then two states $|x_1x_2\cdots x_m\rangle$ and $|x'_1x'_2\cdots x'_m\rangle$ that are initially orthonormal could evolve into two nonorthonormal states, say $|f(x_1,x_2,\ldots,x_m)\rangle = |f(x'_1,x'_2,\ldots,x'_m)\rangle$.

In the following we shall omit for simplicity the subscripts denoting source and target registers.

X. QUANTUM ALGORITHMS

In this section we present a survey of the most representative quantum algorithms to date, named after Deutsch-Jozsa, Simon, Grover and Shor, without discussing the many spinoffs and ramifications that they have led to (for example, Bernstein and Vazirani, 1993; Kitaev, 1995; Hogg, 1998; etc.). We also use these quantum algorithms to emphasize and show in action the main ideas concerning the principles of quantum computation introduced in Sec. IX.

Due to space constraints, we have left out some interesting developments, including quantum clock synchronization[45] (Chuang, 2000; Jozsa *et al.*, 2000) and quantum games (Eisert, Wilkens, and Lewenstein, 1999; Meyer, 1999).[46]

The merging of quantum mechanics and information theory has proved to be very fruitful. One of the products of this merger is the discovery of quantum algorithms that outperform classical ones. It is appealing to think that we can take classical algorithms and devise quantization processes in order to discover new modified quantized versions of classical algorithms. By quantizing a classical algorithm is simply meant the possibility of using quantum bits in a quantum computer as opposed to the classical bits, and all the consequences thereof. This way of thinking reflects the well-known procedure of studying a quantum system by starting with its classical analog and making a quantization of it, using, for instance, Dirac's prescription. One instance of this approach is Shor's algorithm (Sec. X.D). In fact, Shor's algorithm relies on its ability to find the period of a simple function in number theory. The known classical algorithms for this task are inefficient because, as mentioned in Sec. VI, they have subexponential complexity in the input length (unless hard information is supplied). However, when qubits are used to implement the com-

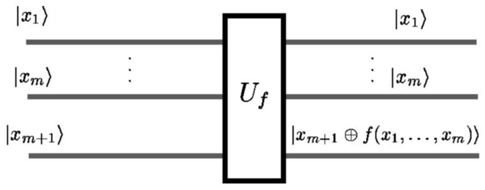


FIG. 31. A gate for function evaluation.

[45]A way to make two atomic clocks start ticking at once. This can also be considered as an application of the quantum Fourier transform (see Sec. X.D) for quantum phase estimation (Cleve *et al.*, 1998).

[46]Quantum games appear so far to be more related to quantum communication protocols (Sec. III) or to applications of the above quantum algorithms.

mon algorithm (we quantize it in our language), then the principles of quantum computation shorten the task to polynomial time. For this drastic improvement are liable the peculiar properties of the discrete quantum Fourier transform (Sec. X.D).

Shor's algorithm also illustrates another common feature of the quantum algorithms known so far: they are best suited for studying global properties of a function or a sequence as a whole, such as finding the period of a function, the median of a sequence, etc., and not individual details. When the value of the function is needed for a particular choice of the argument, no real advantage is gained: one has to extract it from the quantum superposition and this may generally require measuring many times on the output to compensate for the low probability, exponentially small in the register length, of getting the desired result.

Let us point out that it is possible to give a unified picture of most of the forthcoming algorithms in terms of the *hidden subgroup problem*: to find a generating set for a subgroup K of a finitely generated group G, given a function $f:G\to X$, where X is a finite set and f is constant and distinct on the K cosets. Some instances of this problem are the Deutsch-Jozsa, Simon, and Shor algorithms (Boneh and Lipton, 1995; Mosca and Ekert, 1999). Likewise, one may profitably view the quantum computation process as a multiparticle quantum interference (Cleve *et al.*, 1998). However, we have adhered to a more traditional and historical pathway of presenting these quantum algorithms.

A. Deutsch-Jozsa algorithm

This is the quantum algorithm first introduced by Deutsch (1985), providing an explicit and concrete example of how a quantum computer can beat a classical computer. It was later extended to more complex situations by Deutsch and Jozsa (1992). We shall present first an improved version (Cleve *et al.*, 1998) of this algorithm for the simplest case of a Boolean function of a single qubit.

Suppose we are given an oracle that upon request computes a function $f:\{0,1\}^n\to\{0,1\}$. No other information on f is available, just the promise or assumption that f is either *constant* [i.e., $\forall x_1,x_2\in\{0,1\}^n, f(x_1)=f(x_2)$] or *balanced* [in the sense that $\#f^{-1}(0)=\#f^{-1}(1)$, i.e., the numbers of arguments mapping to 0 and to 1 are equal]. The problem is to ascertain whether f is constant or balanced with as few queries to the oracle as possible.

The result of the Deutsch-Jozsa algorithm is that we need only one query or function evaluation to determine the nature of f, while classically $2^{n-1}+1$ consultations would be necessary in the worst case.

Let us see this first when $n=1$. Now f is balanced if and only if $f(0)\neq f(1)$, and thus the promise is worthless. The quantum circuit in Fig. 32 implements the Deutsch-Jozsa algorithm and embodies the following steps.

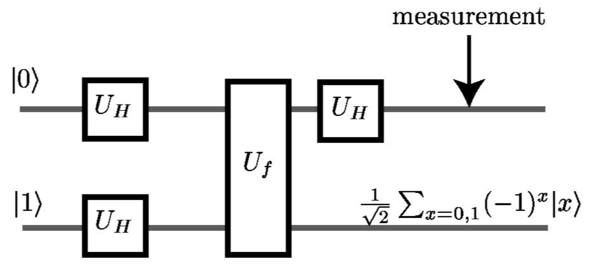


FIG. 32. Quantum circuit for the Deutsch-Jozsa algorithm.

Step 1. An initial quantum register is prepared with two qubits in the state $|\Psi_1\rangle:=|01\rangle$.
Step 2. The Hadamard gate [Eq. (75)] is applied bitwise to this quantum register, producing the state

$$|\Psi_2\rangle:=U_H|0\rangle\otimes U_H|1\rangle=\tfrac{1}{2}(|0\rangle+|1\rangle)\otimes(|0\rangle-|1\rangle). \tag{96}$$

Step 3. We query the f oracle with the state $|\Psi_2\rangle$ and get the answer $|\Psi_3\rangle:=U_f|\Psi_2\rangle$. Using Eq. (95) we readily find

$$\begin{aligned}|\Psi_3\rangle&=U_f\tfrac{1}{2}\sum_{x=0,1}|x\rangle(|0\rangle-|1\rangle)\\&=\tfrac{1}{2}\sum_{x=0,1}(-1)^{f(x)}|x\rangle(|0\rangle-|1\rangle).\end{aligned} \tag{97}$$

Step 4. The Hadamard gate is applied again to the first qubit, which yields

$$\begin{aligned}|\Psi_4\rangle&:=\frac{1}{2}\sum_{x=0,1}(-1)^{f(x)}(U_H|x\rangle)(|0\rangle-|1\rangle)\\&=\frac{1}{2^{3/2}}\sum_{x=0,1}[(-1)^{f(x)}|0\rangle+(-1)^{x+f(x)}|1\rangle]\\&\quad\otimes(|0\rangle-|1\rangle).\end{aligned} \tag{98}$$

Step 5. Finally we measure (in the computational basis) the first qubit (the second qubit no longer plays a role). There are two possibilities: (i) either f is constant, and then the first-qubit amplitude of $|1\rangle$ in Eq. (98) vanishes and we measure $|0\rangle$ with certainty; or (ii) f is not constant and consequently it is balanced, in which case it is the amplitude of $|0\rangle$ in Eq. (98), which vanishes and we measure $|1\rangle$ with certainty.

Therefore with this Deutsch-Jozsa algorithm we need only query the function once in order to determine whether it is constant or balanced.

Let us point out how the peculiarities of quantum mechanics enter in the algorithm and provide its power. In Step 2 it is possible to prepare a superposition of all the basis states using the Hadamard gates, which have no classical analog. In Step 3 we evaluate the function on all the basis states at one go. However, this is not enough and we need to use interference of the quantum amplitudes in Step 5 to discriminate between the two possibilities for which we were searching. This is a simple manifestation of the idea of using constructive interference to distill the desired results, as was advanced in Sec. IX.A (see Table V).

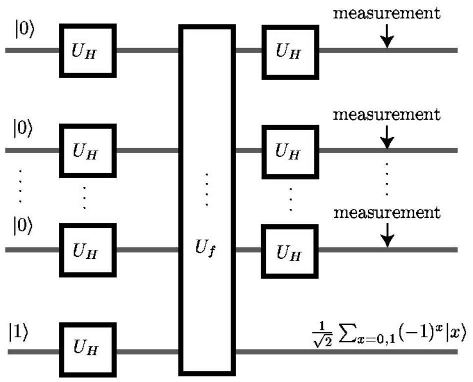


FIG. 33. Extended Deutsch-Jozsa algorithm.

The extension of the Deutsch-Jozsa algorithm to a function of n qubits $f:\{0,1\}^n \to \{0,1\}$ constrained to be either constant or balanced can be done with the help of the quantum circuit shown in Fig. 33. Following this circuit we can extend the previous five steps immediately. We prepare a source register with n qubits initialized to $|0\rangle$ and a target register with one qubit initialized to $|1\rangle$. With x we denote the integer $x := \sum_{i=0}^{n-1} x_i 2^i$ associated with the string of bits $x_{n-1}\cdots x_1 x_0$, and $|x\rangle := |x_{n-1}\cdots x_1 x_0\rangle$.

Let $|\Phi_1\rangle := |0\rangle|1\rangle$. After the bitwise application of the Hadamard gate to $|\Phi_1\rangle$ we find

$$\begin{aligned}|\Phi_2\rangle &:= U_H^{\otimes(n+1)}|\Phi_1\rangle \\ &= (U_H|0\rangle)(U_H|0\rangle)\cdots(U_H|0\rangle)(U_H|1\rangle) \\ &= \frac{1}{2^{n/2}}\sum_{x=0}^{2^n-1}|x\rangle\frac{1}{\sqrt{2}}\sum_{y=0,1}(-1)^y|y\rangle. \qquad (99)\end{aligned}$$

Using Eq. (95), we find that the function evaluation on $|\Phi_2\rangle$ yields the following state:

$$|\Phi_3\rangle := \frac{1}{2^{n/2}}\sum_{x=0}^{2^n-1}(-1)^{f(x)}|x\rangle\frac{1}{\sqrt{2}}\sum_{y=0,1}(-1)^y|y\rangle. \quad (100)$$

In the next step we again apply the Hadamard gates but only on the n source qubits. After some algebra we arrive at the final state $|\Phi_4\rangle$, given by

$$\begin{aligned}|\Phi_4\rangle &:= (U_H^{\otimes n}\otimes 1)|\Phi_3\rangle \\ &= \frac{1}{2^n}\sum_{x=0}^{2^n-1}\sum_{x'=0}^{2^n-1}(-1)^{x\cdot x'+f(x)}|x'\rangle\frac{1}{\sqrt{2}}\sum_{y=0,1}(-1)^y|y\rangle, \qquad (101)\end{aligned}$$

where $x\cdot x' := \sum_{i=0}^{n-1} x_i x_i' \in \mathbb{Z}_2$.

If f is constant, then it produces an overall sign factor in Eq. (101), and after the double summation only the state $|x'\rangle = |0\rangle$ survives. Conversely, if f is balanced, then the same reasoning shows that such a state has zero amplitude in $|\Phi_4\rangle$. In summary, only when all the final source qubits are $|0\rangle$ is the function constant; otherwise it is balanced.

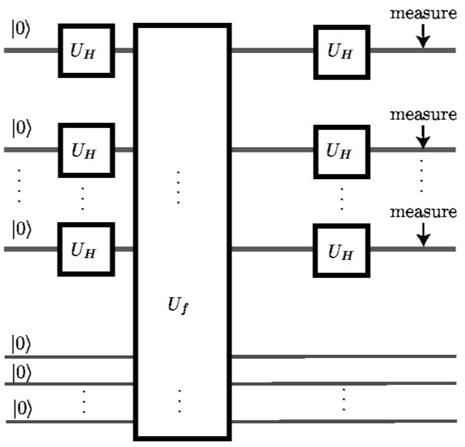


FIG. 34. Quantum circuit for Simon's algorithm.

Thus by measuring the state of the source qubits we can determine the nature of f with certainty. This final measurement step allows us to take advantage of the interference among amplitudes obtained in previous stages.

A single query to the function black box has proved sufficient. However, with the classical algorithms known so far we would require a number of $2^{n-1}+1$ function evaluations (in the worst case) to determine with certainty which type of function f is. This represents an exponential speedup for this quantum algorithm.

Let us point out that classically, given any $1>\epsilon>0$, it is also possible to devise an efficient probabilistic algorithm such that running it a large enough number of times M (independent of the input length n) will determine whether any given function f is constant or balanced, with error probability $<\epsilon$. This is the procedure: the function f is evaluated for M random choices of the argument. When any two of the values differ, then we know that f is balanced. However, when all values are equal then the error probability in claiming that f is constant will be less than 2^{-M}. Thus it suffices to choose M such that $2^{-M}<\epsilon$. In this sense, the quantum Deutsch-Jozsa algorithm is not such an impressive improvement over classical algorithms.

B. Simon algorithm

Simon's algorithm (Simon, 1994) uses several tools of the Deutsch-Jozsa algorithm. It deals with a vector-valued Boolean function $f:\{0,1\}^n \to \{0,1\}^n$ which is constrained by the following condition or promise: There exists a non-null vector $p \in \{0,1\}^n$, called the *period* of f, such that $f(x)=f(y)$ if and only if either $x=y$ or $x=y\oplus p$. Note that such an f is forcefully a 2-to-1 function.

This algorithm finds the period p after a number $O(n)$ of function evaluations, while the known classical algorithms would require an exponential number of queries.

The steps in Simon's algorithm can be seen in Fig. 34.

Both the source and the target registers have n qubits each. The algorithm proceeds as follows.[47]

Step 1. The quantum registers are initialized to the state $|\Psi_1\rangle := |0\rangle|0\rangle = |00\cdots0\rangle|00\cdots0\rangle$.

Step 2. The Hadamard gate [Eq. (75)] is applied bitwise to the source register, producing the state

$$|\Psi_2\rangle := (U_H|0\rangle)\cdots(U_H|0\rangle)|0\rangle = \frac{1}{2^{n/2}}\sum_{x=0}^{2^n-1}|x\rangle|0\rangle. \tag{102}$$

Step 3. The vector-valued function f is evaluated on the target qubits by applying the gate U_f. Using Eq. (95) we readily find the entangled state (Sec. III)

$$|\Psi_3\rangle := U_f|\Psi_2\rangle = \frac{1}{2^{n/2}}\sum_{x=0}^{2^n-1}|x\rangle|f(x)\rangle. \tag{103}$$

Step 4. A further application of the Hadamard gate to the source qubits results in the state

$$\begin{aligned}|\Psi_4\rangle &:= \frac{1}{2^n}\sum_{x=0}^{2^n-1}\sum_{y=0}^{2^n-1}(-1)^{x\cdot y}|y\rangle|f(x)\rangle \\ &= \frac{1}{2^{n+1}}\sum_{x,y=0}^{2^n-1}[(-1)^{x\cdot y}+(-1)^{(x\oplus p)\cdot y}]|y\rangle|f(x)\rangle.\end{aligned} \tag{104}$$

Note that only those qubit states $|y\rangle$ such that $p\cdot y=0$ enter with nonvanishing amplitudes in $|\Psi_4\rangle$. The remaining states are washed out by destructive interference.

Step 5. An ideal measurement of the source qubits (in the computational basis) will necessarily yield a state $|y\rangle$ such that $p\cdot y=0$ with probability $2^{-(n-1)}$.

Step 6. Repeating the previous steps M times we will get M vectors $y_{(i)}$ such that

$$p\cdot y_{(i)}=0, \quad i=1,\ldots,M. \tag{105}$$

Solving this linear system with the Gaussian elimination algorithm will yield the period p with probability large enough provided $M=O(n)$.

The cost in time of Simon's algorithm is $O[n^2 + nC_f(n)]$, where $C_f(n)$ is the cost of evaluating the function f on inputs of length n. The term n^2 is just the cost of the Gaussian elimination over $\mathbb{Z}_2$.

However, a classical blind search would require $2^{n-1}+1$ calls to the oracle in the worst case, and on average a number $O(2^{n/2})$ of function evaluations (Shor, 2000). Thus Simon's algorithm represents an exponential speedup.

We note in passing that Simon's algorithm resorts to a classical algorithm (Gaussian elimination) to finish off the job. We shall find another interesting collaboration between quantum and classical procedures in Shor's algorithm.

[47]Sometimes one introduces, for didactical purposes, a further step in which the target qubits are measured (Jozsa, 1998).

C. Grover algorithm

The previous quantum algorithms show explicitly some instances in which a quantum computer beats a classical computer, as was advanced in Sec. VIII.A devoted to quantum Turing machines. However, they also present several drawbacks:

(i) *utility*: it is not clear what they are useful for in practical applications.

(ii) *structure*: the searched functions are constrained to comply with certain promises. Thus we may feel as if those constraints quantumly conspire in favor of the Deutsch-Jozsa and Simon algorithms.

Grover's algorithm (Grover, 1996, 1997) represents an example of an *unstructured problem*: one in which no assumptions are made about the function f under scrutiny. Thus we can contrast classical and quantum algorithms on equal footing. Although it came after Shor's algorithm (Shor, 1994), we present it first because it is quite related to the previous algorithms.

The algorithm by Grover solves the problem of searching for an element in a list of N unsorted elements, similar to searching a database like a telephone directory when we know the number but not the person's name. When the size of the database becomes very large, this is known to be one of the basic problems in computational science (Knuth, 1975). The utility of such an algorithm is guaranteed. Classically one may devise many strategies to perform this search, but if the elements in the list are randomly distributed, then we shall need to make $O(N)$ trials in order to have a high confidence of finding the desired element. Grover's quantum searching algorithm takes advantage of quantum-mechanical properties to perform the search with an efficiency of order $O(\sqrt{N})$ (Grover, 1996, 1997).

Let us state the searching problem in terms of a list $\mathcal{L}[0,1,\ldots,N-1]$ with a number N of unsorted elements. We shall denote by x_0 the marked element in $\mathcal{L}$ that we are looking for. The quantum-mechanical solution to this searching problem goes through the preparation of a quantum register in a quantum computer to store the N items of our list. This will allow exploit quantum parallelism. Let us assume that a quantum register is made of n source qubits so that $N=2^n$. We shall also need another register with a target qubit to store the output of function evaluations or calls.

To implement the quantum search we need to construct a unitary operation that discriminates between the marked item x_0 and the rest. The following function,

$$f_{x_0}(x) := \begin{cases} 0 & \text{if} \quad x\neq x_0 \\ 1 & \text{if} \quad x=x_0, \end{cases} \tag{106}$$

and its corresponding unitary operation (95),

$$U_{f_{x_0}}|x\rangle|y\rangle = |x\rangle|y\oplus f_{x_0}(x)\rangle, \tag{107}$$

will do the job. We shall need to count how many applications of this operation or oracle calls are needed to find the item. The rationale behind the Grover algorithm is (1) to start with a quantum register in a state in

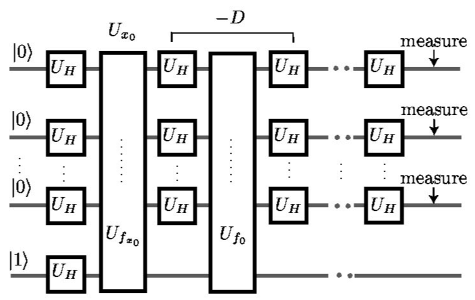


FIG. 35. Quantum circuit (up to an irrelevant global sign factor) for Grover's algorithm.

which all the computational basis states are equally present; and (2) to apply several unitary transformations to produce an output state in which the probability of catching the marked state $|x_0\rangle$ is large enough.

We now present the steps in Grover's algorithm, with the quantum circuit shown in Fig. 35.

Step 1. Initialize the quantum registers to the state $|\Psi_1\rangle := |00\cdots 0\rangle|1\rangle$.
Step 2. Apply bitwise the Hadamard one-qubit gate [Eq. (75)] to the source register so as to produce a uniform superposition of basis states in the source register, and also to the target register:

$$|\Psi_2\rangle := U_{\rm H}^{\otimes(n+1)}|\Psi_1\rangle = \frac{1}{2^{(n+1)/2}}\sum_{x=0}^{2^n-1}|x\rangle\sum_{y=0,1}(-1)^y|y\rangle. \tag{108}$$

Step 3. Apply the operator $U_{f_{x_0}}$:

$$|\Psi_3\rangle := U_{f_{x_0}}|\Psi_2\rangle = 2^{-(n+1)/2}\sum_{x=0}^{2^n-1}(-1)^{f_{x_0}(x)}|x\rangle\sum_{y=0,1}(-1)^y|y\rangle. \tag{109}$$

Let U_{x_0} be the operator defined by

$$U_{x_0}|x\rangle := (1-2|x_0\rangle\langle x_0|)|x\rangle = \begin{cases} -|x_0\rangle & \text{if } x=x_0 \\ |x\rangle & \text{if } x\neq x_0, \end{cases} \tag{110}$$

that is, it flips the amplitude of the marked state, leaving the remaining source basis states unchanged. Grover

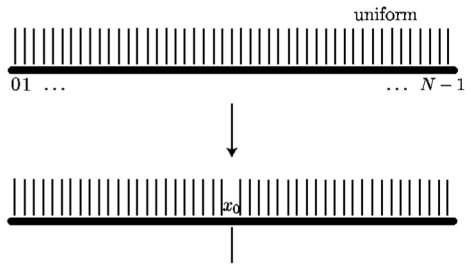


FIG. 36. Schematic representation of Grover's operator U_{x_0} in Eq. (110).

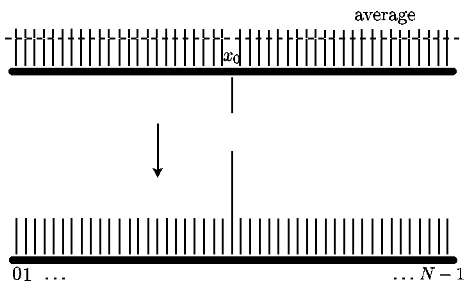


FIG. 37. Schematic representation of Grover's operator D in Eq. (112). The dashed line represents the mean amplitude.

presents this operator graphically as in Fig. 36, with a sort of "quantum comb" in which the spikes denote the uniform amplitudes of state (108) and the action of U_{x_0} is to flip over the spike corresponding to the marked item. We realize that the state in the source register of Eq. (109) equals precisely the result of the action of U_{x_0}, i.e.,

$$|\Psi_3\rangle = ([1-2|x_0\rangle\langle x_0|]\otimes 1)|\Psi_2\rangle. \tag{111}$$

Step 4. Apply the operation D known as *inversion about the average* (Grover, 1996, 1997). This operator is defined as follows:

$$D := -(U_H^{\otimes n}\otimes I)U_{f_0}(U_H^{\otimes n}\otimes I), \tag{112}$$

where U_{f_0} is the operator in Eq. (109) for $x_0=0$. The effect of this operator on the source qubits is to transform $\Sigma_x\alpha_x|x\rangle \mapsto \Sigma_x(-\alpha_x+2\langle\alpha\rangle)|x\rangle$, where $\langle\alpha\rangle := 2^{-n}\Sigma_x\alpha_x$ is the mean of the amplitudes, so its net effect is to enhance the amplitude of $|x_0\rangle$ over the rest. This is graphically represented in Fig. 37 (Grover, 1996, 1997).
Step 5. Iterate Steps 3 and 4 a number of times m.
Step 6. Measure the source qubits (in the computational basis). The number m is determined such that the probability of finding the searched item x_0 is maximal.

The basic component of the algorithm is the quantum operation encoded in Steps 3 and 4, which is repeatedly applied to the uniform state $|\Psi_2\rangle$ in order to find the marked element. Although this procedure resembles the classical strategy, Grover's neatly designed operation enhances by constructive interference of quantum amplitudes (see Table V) the presence of the desired marked state.

It is possible to give a more general formulation to the operators entering Steps 3 and 4 of the algorithm (Galindo and Martin-Delgado, 2000). To this end it is sufficient to focus on the source qubits and introduce the following definitions.

(i) A *Grover operator G* is any unitary operator with at most two different eigenvalues, i.e., G a linear superposition of two orthogonal projectors P and Q:

$$G=\alpha P+\beta Q,\quad P^2=P,\quad Q^2=Q,\quad P+Q=1, \tag{113}$$

where $\alpha,\beta\in\mathbb{C}$ are complex numbers of unit norm.
(ii) A *Grover kernel K* is the product of two Grover

operators:

$$K = G_2 G_1. \tag{114}$$

Some elementary properties follow immediately from these definitions:

(a) Any Grover kernel K is a unitary operator.

(b) Let the Grover operators G_1, G_2 be chosen such that

$$G_1 = \alpha P_{x_0} + \beta Q_{x_0}, \quad P_{x_0} + Q_{x_0} = 1,$$
$$G_2 = \gamma \overline{P} + \delta \overline{Q}, \quad \overline{P} + \overline{Q} = 1, \tag{115}$$

with $P_{x_0} = |x_0\rangle\langle x_0|$, and $\overline{P}$ given by the rank 1 matrix

$$\overline{P} := \frac{1}{N}\begin{pmatrix} 1 & \cdots & 1 \\ \vdots & & \vdots \\ 1 & \cdots & 1 \end{pmatrix}. \tag{116}$$

This is clearly a projector $\overline{P} = |k_0\rangle\langle k_0|$ on the subspace spanned by the state $|k_0\rangle = (1/\sqrt{N})(1,\ldots,1)^{\mathrm{t}}$, where the superscript denotes the transpose. Then, if we take the following set of parameters,

$$\alpha = -1, \quad \beta = 1, \quad \gamma = -1, \quad \delta = 1, \tag{117}$$

the Grover kernel (114) reproduces the original Grover's choice (1996, 1997). This property follows immediately by construction. In fact, we have in this case $G_1 = 1 - 2P_{x_0} =: G_{x_0}$ while the operator $G_2 = 1 - 2\overline{P}$ coincides (up to a sign) with the diffusion operator D (112) introduced by Grover to implement the inversion about the average of Step 4.

The iterative part of the algorithm in Step 5 corresponds to applying m times the Grover kernel K to the initial state $|x_{\mathrm{in}}\rangle := 2^{-n/2}\Sigma_x |x\rangle$, which describes the source qubits after Step 2, searching for a final state $|x_{\mathrm{f}}\rangle$ of the form

$$|x_{\mathrm{f}}\rangle := K^m |x_{\mathrm{in}}\rangle, \tag{118}$$

such that the probability $p(x_0)$ of finding the marked state is above a given threshold value. We may take this value to be 1/2, meaning that we choose a probability of success of 50% or larger. Thus we are seeking under which circumstances the condition

$$p(x_0) = |\langle x_0|K^m|x_{\mathrm{in}}\rangle|^2 \geqslant 1/2 \tag{119}$$

holds true.

The analysis of this probability gets simplified if we realize that the evolution associated with the searching problem can be mapped onto a reduced 2D space spanned by the vectors

$$\left\{|x_0\rangle, |x_\perp\rangle := \frac{1}{\sqrt{N-1}} \sum_{x \neq x_0} |x\rangle. \tag{120}$$

Then we can easily compute the projections of the Grover operators G_1, G_2 in the reduced basis with the result

$$G_1 = \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix}, \tag{121}$$

$$G_2 = \begin{pmatrix} \delta & 0 \\ 0 & \gamma \end{pmatrix} + (\gamma - \delta)\begin{pmatrix} \dfrac{1}{N} & \dfrac{\sqrt{N-1}}{N} \\ \dfrac{\sqrt{N-1}}{N} & \dfrac{-1}{N} \end{pmatrix}. \tag{122}$$

From now on, we shall fix two of the phase parameters using the freedom we have to define each Grover factor in Eq. (114) up to an overall phase. Then we decide to fix them as follows:

$$\alpha = \gamma = -1. \tag{123}$$

With this choice, the Grover kernel (112) takes the following form in this basis:

$$K = \frac{1}{N}\begin{pmatrix} 1 + \delta(1-N) & -\beta(1+\delta)\sqrt{N-1} \\ (1+\delta)\sqrt{N-1} & \beta(1+\delta-N) \end{pmatrix}. \tag{124}$$

The source state $|x_{\mathrm{in}}\rangle$ has the following components in the reduced basis:

$$|x_{\mathrm{in}}\rangle = \frac{1}{\sqrt{N}}|x_0\rangle + \sqrt{\frac{N-1}{N}}|x_\perp\rangle. \tag{125}$$

In order to compute the probability amplitude in Eq. (119), we introduce the spectral decomposition of the Grover kernel K in terms of its eigenvectors $\{|\kappa_1\rangle, |\kappa_2\rangle\}$, with eigenvalues $e^{i\omega_1}, e^{i\omega_2}$. Thus we have

$$a(x_0) := \langle x_0|K^m|x_{\mathrm{in}}\rangle$$
$$= \frac{1}{\sqrt{N}} \sum_{j=1}^{2} \{|\langle x_0|\kappa_j\rangle|^2 + \sqrt{N-1}\langle x_0|\kappa_j\rangle\langle\kappa_j|x_\perp\rangle\} e^{im\omega_j}. \tag{126}$$

This in turn can be cast into the following closed form:

$$\langle x_0|K^m|x_{\mathrm{in}}\rangle = e^{im\omega_1}\left(\frac{1}{\sqrt{N}} + (e^{im\Delta\omega} - 1)\langle x_0|\kappa_2\rangle \times \langle\kappa_2|x_{\mathrm{in}}\rangle\right), \tag{127}$$

with $\Delta\omega := \omega_2 - \omega_1$.

In terms of the matrix invariants

$$\mathrm{Det}K = \beta\delta, \quad \mathrm{Tr}K = -(\beta + \delta) + (1+\beta)(1+\delta)\frac{1}{N}, \tag{128}$$

the eigenvalues $\zeta_{1,2} := e^{i\omega_{1,2}}$ are given by

$$\zeta_{1,2} = \tfrac{1}{2}\mathrm{Tr}K \mp \sqrt{-\mathrm{Det}K + \tfrac{1}{4}(\mathrm{Tr}K)^2}. \tag{129}$$

The corresponding un-normalized eigenvectors are

$$|\kappa_{1,2}\rangle \propto \begin{pmatrix} \dfrac{A \mp \sqrt{-4(\mathrm{Det}K)N^2 + A^2}}{2(1+\delta)\sqrt{N-1}} \\ 1 \end{pmatrix}, \tag{130}$$

with

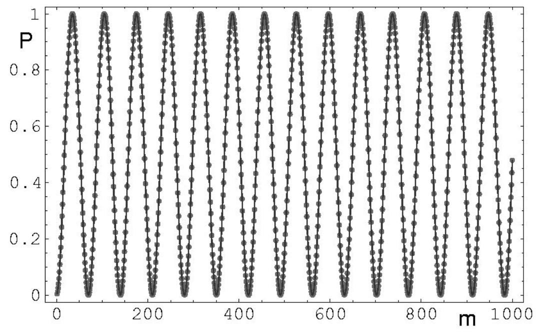


FIG. 38. Probability of success p as a function of the time step for $N=1000$ and $\beta=\delta=e^{i\pi/2}$.

$$A:=(\beta-\delta)N+(1-\beta)(1+\delta). \tag{131}$$

Although we could work out all the expressions for a generic value N of elements in the list, we shall restrict our analysis to the case of a large number of elements, $N\to\infty$ (see Fig. 38). Thus, in this asymptotic limit, we need to know the behavior for $N\gg 1$ of the eigenvector $|\kappa_2\rangle$, which turns out to be

$$|\kappa_2\rangle\propto\begin{pmatrix}\frac{\beta-\delta}{1+\delta}\sqrt{N}+O\left(\frac{1}{\sqrt{N}}\right)\\ 1\end{pmatrix}. \tag{132}$$

For generic values of β,δ we observe that the first component of the eigenvector dominates the second one, meaning that asymptotically $|\kappa_2\rangle\sim|x_0\rangle$ and then $\langle x_0|\kappa_2\rangle\langle\kappa_2|x_{\text{in}}\rangle=O(1/\sqrt{N})$. This implies that the probability of success in Eq. (127) will never reach the threshold value (119). Then we are forced to tune the values of the two parameters in order to have a well-defined and nontrivial algorithm, and we require

$$\beta=\delta\neq-1. \tag{133}$$

Now the asymptotic behavior of the eigenvector changes and is given by a balanced superposition of marked and unmarked states, as

$$|\kappa_2\rangle\sim\frac{1}{\sqrt{2}}\begin{pmatrix}i\delta^{1/2}\\ 1\end{pmatrix}. \tag{134}$$

This is normalized, and we see that none of the components predominates. When we insert this expression into Eq. (127) we find

$$|\langle x_0|K^m|x_{\text{in}}\rangle|\sim\tfrac{1}{2}|\delta||e^{im\Delta\omega}-1|\sim|\sin(\tfrac{1}{2}m\Delta\omega)|. \tag{135}$$

This result means that we have succeeded in finding a class of algorithms that are appropriate for solving the quantum searching problem. Now we need to find out how efficient they are. To do this let us denote by M the smallest value of the time step m at which the probability becomes maximum; then, asymptotically,[48]

$$M\sim[|\pi/\Delta\omega|]. \tag{136}$$

As it happens, we are interested in the asymptotic behavior of this optimal period of time M. From Eq. (129) we find the following behavior as $N\to\infty$:

$$\Delta\omega\sim\frac{4}{\sqrt{N}}\operatorname{Re}\sqrt{\delta}. \tag{137}$$

If we parametrize $\delta=e^{i\phi}$, we finally obtain the expression

$$M\sim\left[\frac{\pi}{4\cos\frac{\phi}{2}}\sqrt{N}\right]. \tag{138}$$

Therefore we conclude that Grover's algorithm of the class parametrized by ϕ is a well-defined quantum searching algorithm with an efficiency of order $O(\sqrt{N})$.

There have been many applications of Grover's work to quantum searching: for example, finding the mean and median of a given set of values (Grover, 1996), searching the maximum/minimum (Durr and Hoyer, 1996), searching more than one marked item (Boyer *et al.*, 1998), and quantum counting, i.e., finding the number of marked items without caring about their location (Brassard, Hoyer, and Tapp, 1998). There is also a nice geometrical interpretation of the Grover kernel $K=-G_2G_1$ in terms of two reflections G_1 and $-G_2$, one about $|x_\perp\rangle$ and the other about $|x_{\text{in}}\rangle$, producing a simple rotation of the initial state (Jozsa, 1999) by an angle $\theta=2\arcsin(1/\sqrt{N})$ in the plane spanned by $|x_0\rangle$ and $|x_\perp\rangle$. With this construction it is straightforward to arrive at the following exact condition for the optimal value m of iterations:

$$m=\left[\frac{1}{2}\left(\frac{\pi}{2\arcsin\frac{1}{\sqrt{N}}}-1\right)\right]. \tag{139}$$

Finally, it has been shown that Grover's algorithm is optimal (Bennett, Bernstein, *et al.*, 1997; Zalka, 1999), that is, its quadratic speedup cannot be improved for unstructured lists.

D. Shor algorithm

Shor's algorithm (1994) came as a wake-up call for cryptographers working with codes based on the difficulty of factoring large integer numbers[49] (see Sec. VI.A), and now it represents a Damocles's sword hanging over this type of cryptosystem.

The algorithm of Shor has several parts that make it somewhat involved. It may be useful to keep in mind the main ingredients of this algorithm:

(i) A periodic function.

(ii) Quantum parallelism.

[48]The symbol $[x]$ stands for the closest integer to x.

[49]"The problem of distinguishing prime numbers from composite numbers and of resolving the latter into their prime factors is known to be one of the most important and useful in arithmetic" (Gauss, 1801).

(iii) Quantum Fourier transform.
(iv) Quantum measurement.
(v) Euclid's classical algorithm for finding the greatest common divisor $\gcd(n_1,n_2)$ of two integers n_1,n_2.

Quantum computation opens the door to a new factorization method in polynomial time (Shor, 1994). This is why, although the technological difficulties of succeeding in their construction are enormous,[50] it is highly interesting to find systems for key distribution whose security (see Sec. VI.B) does not rely upon the practical difficulty of factoring large integers. Quite ironically, quantum physics provides both a fast factorization method and a secure key distribution (Sec. VI.B).

Let $N \geqslant 3$ be an odd integer to factorize. Let a be an integer in $(1,N)$. Let us assume that $\gcd(N,a)=1$, that is, N and a are coprimes; otherwise $\gcd(N,a)$ would be a nontrivial factor f of N and we would restart with N/f. The integral powers a^x of a form a cyclic group in $\mathbb{Z}_N := \mathbb{Z}/N\mathbb{Z}$, and there exists a smallest integer $r \in (1,N)$, called the order of a mod N, such that $a^r = 1$ in $\mathbb{Z}_N$. Several cases may arise:

(1) r is odd;
(2) r is even and $a^{r/2} = -1$ in $\mathbb{Z}_N$;
(3) r is even and $a^{r/2} \neq -1$ in $\mathbb{Z}_N$.

Only case (3) is of interest because then $\gcd(N,a^{r/2} \pm 1)$ are nontrivial factors of N.

It can be shown that, for any given odd N, the probability of picking up at random an integer $a \in [1,N]$ coprime to N and fulfilling case (3) is $\geqslant 1/(2\log N)$, provided that N is not a pure prime power (Ekert and Jozsa, 1996).[51] Therefore it will be enough to analyze $O[\log(1/\epsilon)\log N]$ randomly chosen values of a to succeed in obtaining a nontrivial factor of N with a probability larger than $1-\epsilon$. For example, if $N = 21\,823$ and $a = 12\,083$, the order of a mod N is $r = 3588$, and $12\,083^{1794} \equiv 4866 \bmod 21\,823$, thereby $\gcd(12\,083^{1794} \mp 1, 21\,823) = \{139,157\}$ are factors of 21 823. By contrast, although the order of $a = 14\,335$ mod N is also even, namely, $r = 1794$, we have $14\,335^{897} \equiv -1 \bmod 21\,823$, and $\gcd(14\,335^{897} \mp 1, 21\,823) = \{1, 21\,823\}$, so that no nontrivial factor of N is now obtained.

The big problem lies in computing the order r of a mod N for large N. And here is where the Shor algorithm comes in to quantumly search for the order r of an integer x in the multiplicative group $\mathbb{Z}_N^*$ of integers modulo N, by producing a state with periodicity r.

[50]As Preskill (1997) recalls, it is quite risky to make guesses in this field; 50 years ago it was foreseen that "*Where a calculator on the ENIAC is equipped with 18,000 vacuum tubes and weighs 30 tons, computers in the future may have only 1,000 tubes and perhaps only weigh 1 1/2 tons*" (*Popular Mechanics*, March 1949). The "future" has surpassed these expectations amply.

[51]There are fast power tests to detect whether N is a prime power, say $N = p^s$, and to find p in that case (Cohen, 1993). A rudimentary transcendental and not very efficient procedure consists in trying with the integers $\lfloor N^{1/k} \rfloor, \lceil N^{1/k} \rceil, k = 2,3,\ldots,\lceil \log_2 N \rceil$, until hopefully finding one being a divisor of N.

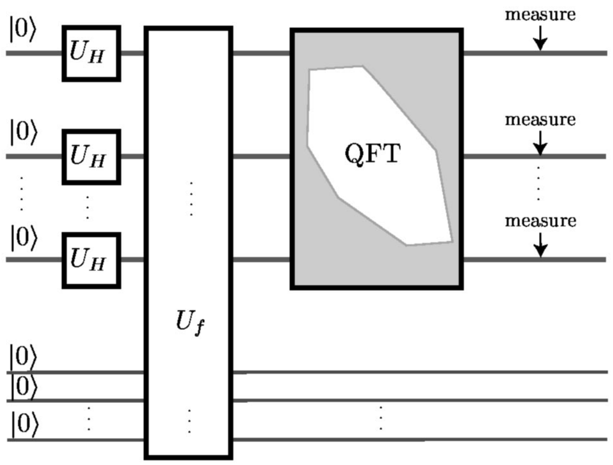


FIG. 39. A quantum circuit representing the Shor algorithm. QFT = quantum Fourier transform.

As usual, we need two quantum registers: a source register with K qubits such that $Q := 2^K \in (N^2, 2N^2)$, and a target register with at least N basis states (i.e., with $\lceil \log_2 N \rceil$ qubits).

These are the main steps of Shor's algorithm (see Fig. 39).

Step 1. Initialize the source and target qubits to the state $|\Psi_1\rangle := |0\rangle \otimes |0\rangle$.

Step 2. Apply on the source register the quantum Fourier transform (which is just the discrete Fourier transform F_Q in $\mathbb{Z}_Q$):[52]

$$U_{F_Q}: |q\rangle \mapsto \frac{1}{\sqrt{Q}} \sum_{q'=0}^{Q-1} e^{2\pi i q q'/Q} |q'\rangle. \tag{140}$$

Here, as usual, $q := \sum_{j=0}^{Q-1} q_j 2^j$, $q_j = 0,1$, and $|q\rangle := |q_{Q-1} \cdots q_1 q_0\rangle$. The following output state is produced:

$$|\Psi_2\rangle := (U_{F_Q} \otimes 1)|\Psi_1\rangle = Q^{-1/2} \sum_{q=0}^{Q-1} |q\rangle \otimes |0\rangle. \tag{141}$$

This particular case of the quantum Fourier transform corresponds to the Hadamard gate acting bitwise on the source qubits.

Step 3. Next apply the gate U_a implementing the modular exponentiation function $q \mapsto a^q \bmod N$:

$$|\Psi_3\rangle := U_a|\Psi_2\rangle = Q^{-1/2} \sum_{q=0}^{Q-1} |q\rangle \otimes |a^q \bmod N\rangle. \tag{142}$$

This operation computes at one go $a^q \bmod N$ for all q as a manifestation of the quantum parallelism (see Sec. IX. A).

Step 4. Again apply the Fourier transform U_{F_Q} on the source register. Then the state becomes

[52]This is especially fast when $Q = 2^K$.

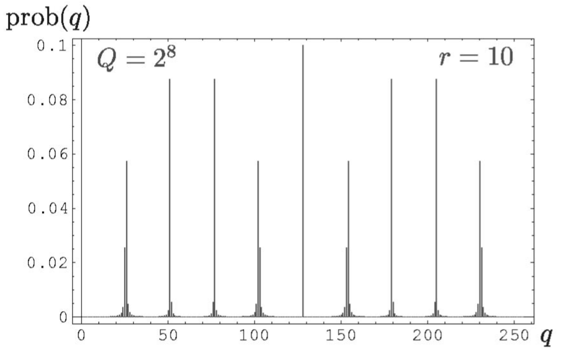


FIG. 40. The probability prob(q) for the case $Q=2^8, r=10$. It becomes concentrated around the integers $\lfloor sQ/r \rfloor$, with s integer.

$$|\Psi_4\rangle := (U_{\mathrm{F}_Q}\otimes 1)|\Psi_3\rangle = \frac{1}{Q}\sum_{q=0}^{Q-1}\sum_{q'=0}^{Q-1} e^{2\pi i q q'/Q}|q\rangle\otimes|a^{q'} \bmod N\rangle. \tag{143}$$

Step 5. Measure the source qubits in the computational basis. The probability of finding them in the state $|q\rangle$ is prob$(q)=\Sigma_{j=0}^{r-1}$prob$_j(q)$, where

$$\mathrm{prob}_j(q) := \frac{1}{Q^2}\left|\sum_{k=0}^{B_j-1}(e^{2\pi i q r/Q})^k\right|^2, \tag{144}$$

with $B_j := 1+\lfloor (Q-1-j)/r \rfloor$.

To simplify the algebra, an intermediate step is introduced in most discussions of Shor's algorithm in which the target qubits are measured prior to the second application of the quantum Fourier transform (Shor, 1995; Ekert and Jozsa, 1996). If $|b\rangle$ is the result, the source register will be projected onto a state $B^{-1/2}\Sigma_{k=0}^{B-1}|d_b + kr\rangle$, superposition of basis states with the periodicity r of a^q. Here d_b is the minimum non-negative integer such that a^{d_b} mod $N=b$, and $B:=1+\lfloor (Q-1-d_b)/r \rfloor$ is the length of the series. After applying the quantum Fourier transform and measuring the source qubits, the probability to obtain $|q\rangle$ now is just (Q/B_{d_b})prob$_{d_b}(q)$.

Let us see how to pull out the order r of a from the study of the above probability prob(q). The analysis of the geometrical series in Eq. (144) shows that prob(q) peaks around those qs for which all the complex numbers in the sum fall on a semicircle, and thus they enhance each other constructively. It can be shown that such q's are characterized by $|(qr \bmod Q)| \leq \frac{1}{2}r$, they number r, and satisfy prob$(q) \geq (2/\pi)^2 r^{-1}$; therefore the probability of hitting upon any one of them is $\geq (2/\pi)^2 = 0.405\ldots$. In Fig. 40 the form of prob(q) is shown.

The condition of constructive interference (see Table V) for each $q>0$ amounts to the existence of an integer $q' \in (0,r)$ such that $|(q/Q)-(q'/r)| \leq \frac{1}{2}Q^{-1}$. As we have chosen $Q>N^2$, and $r<N$, there exists a unique q' such that the fraction q'/r satisfies that inequality. This rational number q'/r can be easily found as a convergent to the (finite simple) continued fraction expansion of q/Q. If this convergent is the irreducible fraction q_1/r_1, it may happen that $a^{r_1}\equiv 1$ mod N, which implies $r=r_1$, and we are done. Otherwise, we would only know that r_1 is a divisor of r, and we would have to carry on, choosing another q with constructive interference, to see if this time we are luckier. It can be shown that the probability of finding an appropriate q is of order $\mathcal{O}(1/\log\log r)$, and therefore with a number $\mathcal{O}(\log\log N)$ of trials it is highly probable to obtain r.

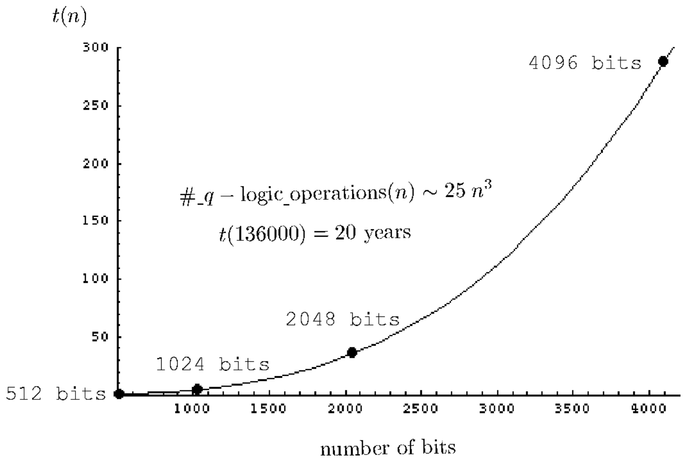


FIG. 41. Factorization times with a hypothetical quantum computer at a nominal clock frequency of 100 MHz. The time $t(n)$, in minutes, is shown as a function of the number of bits.

For example, let $N=15$ (this is a sort of "toy model") and $a=7$. We can effortlessly see that $r=4$. Suppose, however, that we insist on following the Shor way (quite a luxury in this case, but a necessity if N had half a thousand digits). We would take $Q=2^8$ to comply with $N^2<Q<2N^2$. After Step 5 we would obtain the state $|q\rangle$ of the source qubits, where, for instance, $q=0,64,128,192$ with probabilities 0.25,0.25,0.25,0.25. The first value is useless, for q/Q does not allow us to determine r if $q=0$. From the continued fraction series expansion $\{a_0,a_1,a_2,\ldots\}:=a_0+1/[a_1+1/(a_2+\cdots)]$ of q/Q $(64/256=\{0,4\},128/256=\{0,2\},192/256=\{0,1,3\})$ we see that for $q=64$ (respectively, 128,192), the fraction 1/4 (respectively, 1/2,3/4) approximates q/Q with an error less than $1/2Q$. Thus 4 is a divisor of r, i.e., $r=4,8,12$, and so on. A direct check selects $r=4$ as the order of 7 mod 15. And since $7^{4/2}\not\equiv -1$ mod 15, then gcd$(49 \pm 1,15)=\{5,3\}$ are factors of 15.

As a little more complicated example, take $N=25\,397, a=71$. Then $Q=2^{30}=1\,073\,741\,824$. There are many values of q for which the probability is appreciable and similar. One of those is $q=6\,170\,930$, for which prob(q) is about 2×10^{-3}. The approximation 1/174 to q/Q is the only convergent with denominator $<N$ provided us by the continued fraction expansion $\{0,174,1\,542\,732,2\}$ of q/Q. Therefore the order r of 71 mod 25 397 is a multiple of 174, say $r=174$, 348, 522, etc. A direct check shows that $r=522$. Also in this case $a^{r/2}\not\equiv -1$ mod N, and gcd$(71^{261}\pm 1,25\,397)=\{109,233\}$ are divisors of 25 397.

In Fig. 41 the factorization time with a hypothetical quantum computer at 100 MHz is represented as a func-

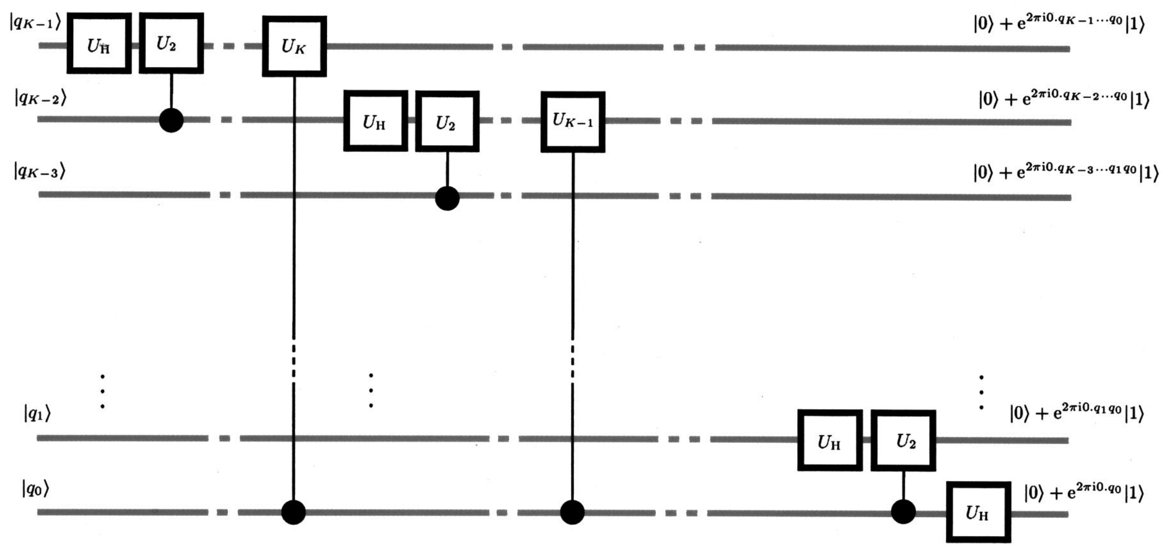

$|q_{K-1}\rangle$
$|q_{K-2}\rangle$
$|q_{K-3}\rangle$
$|q_1\rangle$
$|q_0\rangle$
U_H
U_2
U_K
U_{K-1}
$|0\rangle + e^{2\pi i 0.q_{K-1}\cdots q_0}|1\rangle$
$|0\rangle + e^{2\pi i 0.q_{K-2}\cdots q_0}|1\rangle$
$|0\rangle + e^{2\pi i 0.q_{K-3}\cdots q_1 q_0}|1\rangle$
$|0\rangle + e^{2\pi i 0.q_1 q_0}|1\rangle$
$|0\rangle + e^{2\pi i 0.q_0}|1\rangle$

FIG. 42. Implementation of the quantum Fourier transform with Hadamard and controlled-phase gates (up to a reversion of output qubits). By U_j we denote the unary gate $U_j := |0\rangle\langle 0| + e^{2\pi i/2^j}|1\rangle\langle 1|$. For typographical reasons a factor of $2^{-1/2}$ has been omitted in each output qubit.

tion of binary length of the integer to be factorized. The spectacular efficiency of the Shor algorithm stands out, with a time of 20 years for an integer of about 40 000 digits (Hughes, 1998).

Shor's algorithm may seem a bit miraculous after those several "manipulations" or steps. The rationale is the same as we described in Sec. IX: to drive the system into an appropriate outcome state that upon measurement yields the desired result with high probability. Where does the constructive interference ingredient (Table V) come into the algorithm? It is by means of the second quantum Fourier transform operation. This is designed to produce the interference among qubit amplitudes in such a way as to enhance those aspects of the output that favor the determination of the order r.

1. The quantum Fourier transform

Let us take a closer look at the discrete Fourier transform U_{F_Q} when $Q=2^K$. It is at the core of Shor's algorithm and is responsible for its exponential speedup. To analyze the efficiency of the Shor algorithm it proves convenient to implement the quantum Fourier transform by means of one- and two-qubit gates. The result, shown in Fig. 42, will follow from Eq. (140), duly worked out.

The phase factor $e^{2\pi i q q'/2^K}$ in Eq. (140) is a periodic function of q, and of q' as well, with period 2^K. The numbers q and q' have the following binary decompositions: $q=\Sigma_{j=0}^{K-1} q_j 2^j$, $q_j=0,1$ and $q'=\Sigma_{l=0}^{K-1} q'_l 2^l$, $q'_l=0,1$. Their product can then be written as

$$qq' = \sum_{j,l=0}^{K-1} q_j q'_l 2^{j+l} = \sum_{0\leqslant j+l<K} q_j q'_l 2^{j+l} \bmod \mathbb{Z}_Q . \tag{145}$$

By entering this expression into Eq. (140) and defining $\overline{q'_l} := q'_{K-1-l}, l=0,\dots,K-1$, $0.abc\cdots := 2^{-1}a + 2^{-2}b + 2^{-3}c + \cdots$, we find

$$\begin{aligned} U_{\mathrm{F}_Q}|q\rangle &= \frac{1}{\sqrt{Q}} \sum_{q'=0}^{Q-1} \exp(2\pi i q q'/2^K)|q'\rangle \\ &= \frac{1}{\sqrt{Q}} \sum_{q'=0}^{Q-1} \exp\left(2\pi i \sum_{0\leqslant j+l<K} q_j q'_l 2^{j+l-K}\right)|q'\rangle \\ &= \frac{1}{\sqrt{Q}} \sum_{\overline{q'}=0}^{Q-1} \exp\left(2\pi i \sum_{0\leqslant j\leqslant l<K} q_j \overline{q'_l} 2^{j-l-1}\right)|\overline{q'}\rangle, \end{aligned} \tag{146}$$

and hence

$$\begin{aligned} U_{\mathrm{F}_Q}|q\rangle &= \frac{1}{\sqrt{Q}} \sum_{\overline{q'}=0}^{Q-1} \bigotimes_{l=0}^{K-1} \exp\left(2\pi i \sum_{0\leqslant j\leqslant l} q_j 2^{j-l-1}\overline{q'_l}\right)|\overline{q'_l}\rangle \\ &= \frac{1}{\sqrt{Q}} \bigotimes_{l=0}^{K-1} \sum_{\overline{q'_l}=0}^{1} \exp\left(2\pi i \sum_{0\leqslant j\leqslant l} q_j 2^{j-l-1}\overline{q'_l}\right)|\overline{q'_l}\rangle \\ &= \frac{1}{\sqrt{Q}} \bigotimes_{l=0}^{K-1} [|0\rangle + \exp(2\pi i 0.q_l q_{l-1}\cdots q_0)|1\rangle]. \end{aligned} \tag{147}$$

In particular, the transformed state $U_{\mathrm{F}_Q}|q\rangle$ is separable. The quantum Fourier transform gate U_{F_Q} can be explictly written as a product of Hadamard, controlled-phase, and SWAP gates:

$$U_{\mathrm{F}_Q}=\left(\prod_{i=0}^{\lfloor K/2\rfloor-1} U_{\mathrm{SWAP},i,K-1-i}\right)\times\prod_{l=K-1,\dots,1,0}\left[\left(\prod_{0\leq j\leq l-1} U_{j,l}(\theta_{l-j})\right)U_{\mathrm{H},l}\right], \tag{148}$$

where $\theta_j := \pi/2^j$, $U_{\mathrm{SWAP},i,j}$ exchanges the qubit states labeled by i,j, and

$$U_{\mathrm{H},l}|\cdots q_l\cdots\rangle := 2^{-1/2}\sum_{\overline{q}'_l=0,1} e^{i\pi q_l \overline{q}'_l}|\cdots \overline{q}'_l\cdots\rangle,$$
$$U_{j,l}(\theta)|\cdots q_l\cdots q_j\cdots\rangle := e^{iq_l q_j\theta}|\cdots q_l\cdots q_j\cdots\rangle \tag{149}$$

are the Hadamard gate action of the one-qubit $|q_l\rangle$ and the controlled-phase gate action on the two-qubit state $|q_l q_j\rangle$, respectively. From the factorization (148) we can read off the quantum circuit (see Fig. 42) implementing the quantum Fourier transform (up to a reversion of the output qubits).

The number of Hadamard gates in this implementation of the quantum Fourier transform is K, and that of the controlled-phase gates is $\frac{1}{2}K(K-1)$. Altogether this implies that the size of the quantum circuit for Shor's algorithm is of order $O(K^2)$ regardless of the SWAP gates for the final reversion (Coppersmith, 1994).[53]

The quantum Fourier transform can be extended to deal with qubits with a number of states d not necessarily equal to 2 (see Sec. III). In this case the dimension of the Hilbert space of K source qubits is $Q=d^K$, and Eqs. (140) and (149) for the quantum Fourier transform, the Hadamard and the controlled-phase gates hold true provided the phase angle is taken to be

$$\theta_j=\frac{2\pi}{d^{j+1}}. \tag{150}$$

For instance, for qubits with $d=3$ state or qutrits, the Hadamard gate takes the following explicit form:

$$U_{\mathrm{H}}^{(3)}|0\rangle=\frac{1}{\sqrt{3}}[|0\rangle+|1\rangle+|2\rangle],$$
$$U_{\mathrm{H}}^{(3)}|1\rangle=\frac{1}{\sqrt{3}}[|0\rangle+\omega|1\rangle+\omega^2|2\rangle],$$
$$U_{\mathrm{H}}^{(3)}|2\rangle=\frac{1}{\sqrt{3}}[|0\rangle+\omega^2|1\rangle+\omega|2\rangle], \tag{151}$$

with $\omega := e^{2\pi i/3}$.

In this general case, the sequence of one- and two-qubit gates for the decomposition of the quantum Fourier transform as well as their counting, remains valid. This implies that using qudits for the quantum Fourier transform does not spoil its superb performance, and retains the advantage of reducing by a factor of $\lfloor \log_2 d\rfloor$ the length of the quantum registers (see Sec. III).

2. Cost of Shor's algorithm

We finally evaluate the complexity of Shor's algorithm. The first quantum Fourier transform (Step 2) is just a Hadamard operation applied bitwise, and its cost is $O(\log_2 N)$. The modular exponentiation in step 3 consumes $O(\log_2^2 N \log_2\log_2 N \log_2\log_2\log_2 N)$ time (Shor, 1994). The second quantum Fourier transform gate (Step 4) is, according to the results just mentioned, $O(\log_2^2 N)$. Therefore the total cost to determine the order r of a mod N, with a probability of success $O(1)$, is $O(\log_2^{2+\epsilon} N)$, any $\epsilon>0$.

Once r is determined, it remains to calculate gcd($a^{r/2}\pm 1, N$) in order to find a factor of N. This arithmetical operation is more resource demanding, since it takes $O(\log_2^3 N)$ time steps when Euclid's celebrated algorithm is applied.[54]

Altogether we end up with a total cost $O(\log_2^3 N)$ for the complete factorization algorithm with high probability,[55] which represents in practice a subexponential gain over the best classical algorithms (quadratic sieve, general number field sieve) known nowadays.

E. On the classification of algorithms

One of the most important issues in quantum computing is the design of quantum algorithms. Very few of them are known. Apparently we lack the basic principles underlying the quantum version of algorithm problem solving. We want in part to address this question and we believe that one approach is via a comparison with the known strategies of designing classical algorithms in computational science. This is suggested by the relationships between classical and quantum computations presented in Secs. VIII and IX.A. In this regard, we need to distinguish between fundamentals of quantum computation and strategies for designing algorithms. Although the latter are still unknown, the former have been described in Table V. The fact that we can understand the fundamentals of quantum computation does not mean in principle that we know the keys to setting up quantum algorithms, although it can be of great help.

To analyze the classical strategies of algorithm design from the point of view of quantum computation, let us first consider the classification introduced by Levitin (1999), who has done a reformulation that includes and categorizes in a nice fashion other classification schemes (Brassad and Bratley, 1996). According to Levitin, there are four classical general design techniques, which we shall describe briefly, with a simple example to illustrate

[53]In contrast, the classical fast Fourier transform requires order $O(K2^K)$ elementary operations to transform a K-bit vector (Press *et al.*, 1992).

[54]A more refined implementation of the gcd algorithm (Knuth, 1981) reduces its cost to $O[\log N(\log\log N)^2\log\log\log N]$.

[55]Or, better, $O(\log_2^{2+\epsilon} N)$, if the previous footnote is considered.

TABLE VIII. Classification of classical algorithms.

Classical technique	Algorithm example
Brute force	Searching the largest
Divide-and-conquer	Quicksort
Decrease-and-conquer	Euclid's algorithm
Transform-and-conquer	Gaussian elimination

them. This example is the problem of computing a^n mod p, which is of great importance in public-key encryption algorithms (Secs. VI and X.D). We have the following generic types.

(1) *Brute-force algorithms*. This amounts to solving a problem by directly applying its crude formulation. Example: $a^n = a \cdot a \cdots a, n$ times.
(2) *Divide-and-conquer algorithms*. The original problem is partitioned into a number of smaller subproblems, usually of the same kind. These in turn are solved and their solutions combined to get a solution to the bigger problem. This strategy usually employs recursivity in order to obtain a greater profit. Example: $a^n = a^{\lfloor n/2 \rfloor} \cdot a^{\lfloor n/2 \rfloor} \cdot a^{n-2\lfloor n/2 \rfloor}$.
(3) *Decrease-and-conquer algorithms*. The original problem is reduced to a smaller one, which is usually solved by recursion and the solution so obtained is applied to find a solution of the original problem. Examples: (a) $a^n = a^{n-1} \cdot a$ (decrease-by-one variety); (b) $a^n = (a^{\lfloor n/2 \rfloor})^2$ if n even, $a^n = (a^{\lfloor n/2 \rfloor})^2 \cdot a$ if n odd (decrease-by-half variety).
(4) *Transform-and-conquer algorithms*. The original problem is transformed into another equivalent problem that is more amenable to solution with simpler techniques. Example: a^n is computed by exploiting the binary representation of n.

These four types of strategies in turn have several subtypes that we shall not go into.

Table VIII contains these classical strategies with some well-known and less trivial examples of representative algorithms. There are important algorithms built upon a mixture of these basic techniques; for example, the fast Fourier transform employs both divide-and-conquer and transform-and-conquer techniques.

It can be quite revealing to set up the quantum version of Table VIII by classifying the most useful of the quantum algorithms known to date. This we do in Table IX. Several remarks are in order.

TABLE IX. Classification of quantum algorithms.

Quantum technique	Algorithm example
Brute force	Grover's algorithm Deutsch-Jozsa's algorithm Simon's algorithm
Divide-and-conquer	$\varnothing$
Decrease-and-conquer	$\varnothing$
Transform-and-conquer	Shor's algorithm

First, we have placed Grover's algorithm in the category of brute-force algorithms. The strategy is similar to its classical counterpart, which is of the brute-force type. The difference lies in the fact that the quantum operation is realized through a unitary operator that implements the reversible quantum computation.[56] Although the brute-force technique usually produces low-efficiency algorithms, it is very important for several reasons. One is that there are important cases, like the searching problem, in which it outperforms more sophisticated strategies like divide-and-conquer. We find Grover's algorithm as a realization of the brute-force technique at the quantum level and this is why it is so simple and of general utility at the same time.

Second, we have included Shor's algorithm in the category of transform-and-conquer algorithms. As we have explained in Sec. X.D, Shor solves the factorization problem by reducing it to the problem of finding the period of a certain function in number theory, which in turn is solved with the aid of the fundamentals of quantum computation. Having realized this, we point out that classical versions of transform-and-conquer algorithms are very rare (Levitin, 1999). This may explain why Shor's algorithm, although more powerful than Grover's, has a more reduced range of applications.

Third, the most notable aspect of Table IX is the absence of quantum algorithms based on the divide-and-conquer technique, which is by far the most general and widely used strategy in classical computation. This may partly account for the shortness of the list of quantum algorithms. Moreover, if we consider the basic features of quantum computation (Table V) we may better understand why this entry is empty in Table IX. We know that a quantum register supports the superposition of many states at the same time. This implies that the qubits of the quantum registers are strongly correlated (entangled) and that their joint state is not separable into a product of states of smaller subregisters. Thus quantum parallelism and entanglement render unnatural any attempt to implement the strategy of divide-and-conquer in a quantum register, at least in a straightforward and naive fashion.[57]

XI. EXPERIMENTAL PROPOSALS FOR QUANTUM COMPUTERS

The great challenge of quantum computation is to build real quantum computers capable of implementing the quantum logic operations of Sec. IX and of performing the quantum algorithms of Sec. X. In this section we present some of the experimental proposals to this end. Some of these proposals have actually been carried out, and this is already a significant advance, for it means that the theoretical constructs can be checked experi-

[56]By a similar rationale, we have placed the Deutsch-Jozsa and Simon algorithms in the same class.

[57]A blend of classical and quantum algorithms might make room for a divide-and-conquer strategy.

mentally. However, these devices are very modest in size and the real breakthrough will be to scale them up to sizes capable of doing tasks not yet done with classical computers, like code breaking with Shor's algorithm or database searching with Grover's algorithm.

Before giving an overview of a few experimental proposals, it is convenient to summarize what they all have in common. There is a generic foundation for building a quantum computer.[58] We basically need

(i) any two-level quantum system,
(ii) interaction between qubits, and
(iii) external manipulation of qubits.

The two-level system is used as a qubit and the interaction between qubits is used to implement the conditional logic of the quantum logic gates (Sec. IX). The system of qubits must be accessible to external manipulations, to read in the input state and read out the output, as well as during the computation if the quantum algorithm requires it.

Interestingly enough, some of the possible qubits and quantum logic gates have been with us since the time of Bohr. For example, the quantum NOT gate is obtained, at least in principle, either by exciting an atomic ground state to an upper level with a photon of apppropriate frequency and duration, or by induced emission. If the length of light pulses is halved, a Hadamard-like gate will result.[59] Quantum computation has provided us with a new insight on these operations.

There are several settings in which one can fulfill the above three requirements. We shall not go into all the technical details of the experimental proposals below but instead present the basic physical foundations underlying these ideas for quantum computers. We shall choose as our qubit system a spin-$\frac{1}{2}$ massive particle with magnetic moment, whose translational motion will be ignored.[60] Placing this qubit in a suitably oscillating external magnetic field will allow us to theoretically implement the unary quantum gates.

A. One- and two-qubit logic gates with spin qubits

This is one of the few examples in which one can follow exactly the evolution of the quantum system, and it is versatile enough to allow building some of the basic logic gates. We present it as a preparation for more complex setups.

Suppose that our qubit, a spin-$\frac{1}{2}$ particle, has a magnetic moment $\boldsymbol{\mu}=\gamma\mathbf{S}$, where $\mathbf{S}=\frac{1}{2}\hbar\boldsymbol{\sigma}$ is the spin operator. In the presence of a uniform but time-dependent magnetic field $\mathbf{B}(t)$ the qubit state $|\psi(t)\rangle$ will evolve with the Hamiltonian $H(t)=-\gamma\mathbf{S}\cdot\mathbf{B}(t)$ (Rabi, 1937):

$$i\hbar\frac{d}{dt}|\psi(t)\rangle=-\gamma\mathbf{S}\cdot\mathbf{B}(t)|\psi(t)\rangle. \tag{152}$$

When the magnetic field rotates uniformly around a fixed axis (say Oz),

$$\mathbf{B}(t)=(B_1\cos\omega t, B_1\sin\omega t, B_0), \tag{153}$$

then Eq. (152) can be solved explicitly, with the result (Galindo and Pascual, 1990b)

$$\begin{aligned}\psi(t)\rangle&=U(t)|\psi(0)\rangle,\\ U(t)&:=e^{-i\omega t\sigma_z/2}e^{-i[(\omega_0-\omega)\sigma_z+\omega_1\sigma_x]t/2}\\ &=[\cos\tfrac{1}{2}\omega t-\mathrm{i}(\sin\tfrac{1}{2}\omega t)\sigma_z][\cos\tfrac{1}{2}\Omega t-i(\sin\tfrac{1}{2}\Omega t)\sigma'],\end{aligned} \tag{154}$$

where $\omega_0:=-\gamma B_0$, $\omega_1=-\gamma B_1$, $\Omega:=[(\omega_0-\omega)^2+\omega_1^2]^{1/2}$ is the so-called *Rabi frequency* and $\sigma':=\Omega^{-1}[(\omega_0-\omega)\sigma_z+\omega_1\sigma_x]$.

As the computational basis (Sec. IX.A) we shall take the eigenvectors of σ_z: $|0\rangle:=|\uparrow\rangle$ (spin-up state) and $|1\rangle:=|\downarrow\rangle$ (spin-down state).[61]

The probability of spin flip $\uparrow\leftrightarrow\downarrow$ is one if and only if $\omega=\omega_0$ (resonance condition), hence $\Omega=|\omega_1|$, and $t\Omega\in 2\pi(\mathbb{Z}+\frac{1}{2})$. When the oscillating part of the magnetic field (153) is resonant, i.e., it satisfies $\omega=\omega_0$, then such a field is known as a *Rabi pulse*.

Let us see how to induce one-qubit operations using Rabi pulses of appropriate durations. In view of Eq. (88), and up to the global phase factor represented by $\mathrm{Ph}(\delta)$ in Eq. (89), it suffices to do it for the rotations $R_z(\alpha), R_y(\beta)$.

(a) The rotation $R_z(\alpha)$ is emulated by taking a constant field along the z axis and setting to zero the oscillating part ($B_1=0$, i.e., $\Omega=0$). The angle is simply $\alpha=\frac{1}{2}\omega_0 T$, T being the pulse length. The rotation $R_z(\gamma)$ is obtained similarly.

(b) To reproduce the rotation $R_y(\beta)$ in the decomposition (88), note that $R_y(\beta)=R_z(\frac{1}{2}\pi)R_x(\beta)R_z(-\frac{1}{2}\pi)$, and that $U(t)=R_z(\omega t)R_x(\Omega t)$. Therefore to build $R_y(\beta)$ it suffices to compose with suitable rotations around Oz, implemented as above, the action of a Rabi pulse with $\Omega T=\beta$.

For instance, a π pulse, i.e., a pulse with duration $T=\pi/\Omega$, reproduces in the interaction picture a quantum NOT gate (up to a global factor $-i$).[62] Similarly, a $\pi/2$ pulse produces essentially a Hadamard gate.

So far we have manipulated externally the spins $\frac{1}{2}$ to produce one-qubit gates. To generate two-qubit gates we need a pair of interacting qubits at sites 1,2. For simplicity's sake, let us assume the simplest possible type of interaction between them, namely, an Ising interaction:

$$H_{12}=-(\gamma_1 S_1^z+\gamma_2 S_2^z)B^z+2(J/\hbar)S_1^z S_2^z. \tag{155}$$

[58]At least with our present knowledge.

[59]Strictly speaking, this halved pulse produces the action of the so-called pseudo-Hadamard gate.

[60]Other simple choices might be the polarization of a photon, an atomic system with just two relevant levels, etc.

[61]With this choice, $|0\rangle$ will be the ground state of the magnetic Hamiltonian provided that the spin corresponds to a positively charged particle ($\gamma>0$).

[62]At resonance, the time-evolution operator $U(t)$ factorizes as $U(t)=e^{-i\omega_0 t\sigma_z/2}e^{-i\Omega t\sigma_x/2}$. The first factor represents the evolution operator $U_0(t)$ under the static magnetic field, whereas the second factor is just the total unitary propagator $U_{\mathrm{I}}(t):=U_0^{-1}(t)U(t)$ in the interaction picture.

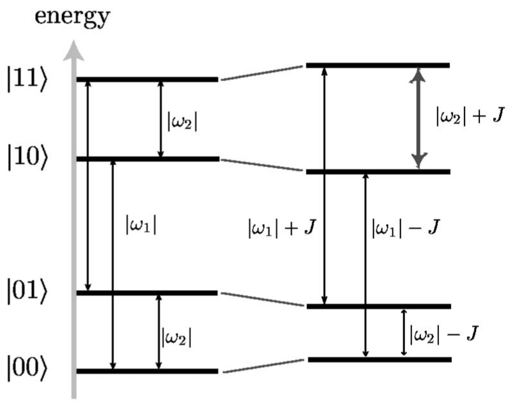


FIG. 43. Energy levels of a two-qubit spin system with Ising interaction (units $\hbar = 1$): On the left are the noninteracting Zeeman levels; on the right, the levels perturbed by the Ising term (when $\omega_1 < \omega_2 < -J < 0$).

The origin of the single spin terms may be the presence of an external magnetic field. In Eq. (155), this field is constant and directed along Oz, and the two spins may have different magnetic moments. The coupling constant J measures the spin-spin interaction. Defining the frequencies $\omega_i := -\gamma_i B^z, i = 1,2$, the eigenvalues of this Hamiltonian are

$$E_{x_1 x_2} = \tfrac{1}{2}\hbar[(-1)^{x_1}\omega_1 + (-1)^{x_2}\omega_2 + (-1)^{x_1+x_2}J], \tag{156}$$

where $x_i = 0,1$, $i = 1,2$.

These energy levels are represented in Fig. 43 for $\omega_1 < \omega_2 < -J < 0$. We can clearly see that if we apply a π pulse with frequency $\omega = |\omega_2| + J$, the states $|11\rangle$ and $|10\rangle$ are interchanged while the rest are not excited. This is precisely what a CNOT gate does, with the first spin acting as control qubit and the second spin as a target qubit (Berman *et al.*, 1997).

Other useful two-qubit gates such as the controlled-phase gate [Eq. (78)], which enters Shor's algorithm, can be built up similarly using the Ising interaction. An explicit construction of this gate is the following (Jones, Hansen, and Mosca, 1998):

$$U_{\rm CPh}(\phi) = \exp(-i\tfrac{1}{2}\phi[-\tfrac{1}{2} + \overline{S_1^z} + \overline{S_2^z} - 2\overline{S_1^z}\,\overline{S_2^z}]), \tag{157}$$

where $\overline{S_k^z} := S_k^z/\hbar = \tfrac{1}{2}\sigma_k^z$. Of particular interest is the case $\phi = \pi$ for, as remarked in Sec. IX.B, with this controlled gate plus two Hadamard gates (on the target qubit) we can reconstruct the important CNOT gate [Eq. (79)].

B. The ion-trap quantum computer

The ion-trap quantum computer was introduced by Cirac and Zoller (1995) and since then many other potential and actual realizations of quantum computers have been pursued by many groups. The quantum hardware is the following: a qubit is a single ion held in a trap by laser cooling and the application of appropriate electromagnetic fields; a quantum register is a linear array of ions; operations are effected by applying laser Rabi pulses; information transmission is achieved as a result of the Coulomb interaction between ions and the exchange of phonons from collective oscillations. We see again, at a very fundamental level, that information is physical. Using the Cirac-Zoller (CZ) technique, Monroe *et al.* (1995) were soon able to construct a single quantum gate.

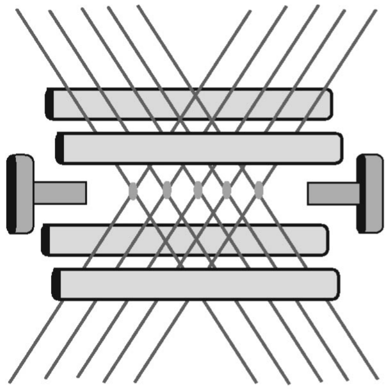


FIG. 44. Schematic geometry of a radio-frequency quadrupole linear ion trap. Laser beams address a string of ions in the middle of the setup with four linear rods and two end caps.

The ion-trap proposal has several advantages: it calls for manipulation of quantum states that is already known from precision spectroscopy techniques; it has low decoherence rates due to the decay of excited states and the heating of the ionic motion; and it takes advantage of existing very efficient experimental methods for retrieving the information from the quantum computer, such as the mechanism of quantum jumps.

1. Experimental setup

The geometry of a radio-frequency (RF) ion trap or Paul trap is schematically shown in Fig. 44. An RF Paul trap uses static and oscillating electric potentials to confine particles within small ($\sim 1\ \mu$m) regions. To obtain a string of ions for forming the quantum register we need a quadrupole ion trap with a cylindrical geometry. The confining mechanism of ions is twofold:

(i) A strong radial confinement, achieved by RF potentials generally produced with four rod electrodes.
(ii) An axial confinement, achieved by applying a quadrupolar electrostatic potential through two end caps.

The ions lie along the trap axis and their oscillations are controlled by the axial potential. The collective oscillations of the string center of mass are used as a sort of computational bus, transferring information from one ion to another by phonon exchange. The dimensions of the ion traps used by the Los Alamos group are typically 1 cm long and 1–2 mm wide (Hughes *et al.*, 1998).

Before any computation takes place, the center of mass of the ion string must be set to its ground state. This is accomplished by a laser cooling process that

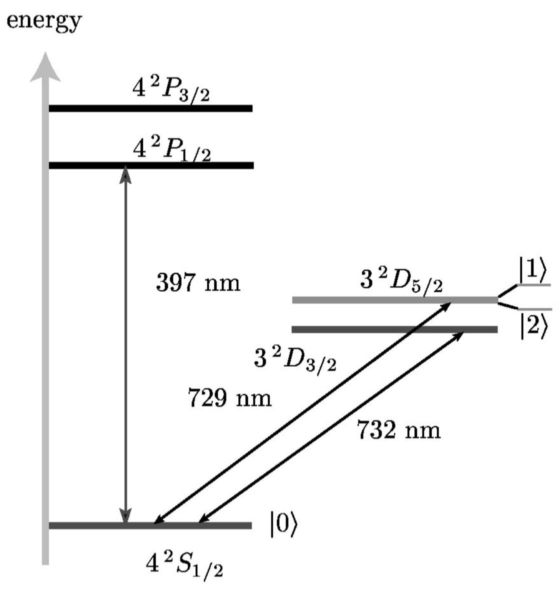


FIG. 45. Relevant energy levels in Ca$^+$ ions.

brings the ions to the ground state of their vibrational motion. The result is an ion string configuration as shown in Fig. 44, crystallizing into a linear array that makes it possible to address each ion individually by lasers. The inter-ion spacing can be controlled by balancing the Coulomb repulsion of the ions and the axially confining potential (Wineland *et al.*, 1998).

Several kinds of ions (Be$^+$, Ca$^+$, Ba$^+$, Mg$^+$, Hg$^+$, Sr$^+$) and qubit schemes have been proposed. The Cirac-Zoller qubit $\{|0\rangle,|1\rangle\}$ is built using some appropriate electronic ion states. For instance, the Los Alamos group (Hughes *et al.*, 1998) has chosen Ca$^+$ ions, whose most relevant levels are shown in Fig. 45. The state qubits $\{|0\rangle,|1\rangle\}$ and one extra auxiliary level $|2\rangle$ (to be described below) are identified as follows (see Fig. 45):

$$\begin{aligned}|0\rangle&=|4\ ^2S_{1/2},M_J=\tfrac{1}{2}\rangle,\\|1\rangle&=|3\ ^2D_{5/2},M_J=\tfrac{3}{2}\rangle,\\|2\rangle&=|3\ ^2D_{5/2},M_J=-\tfrac{1}{2}\rangle.\end{aligned}\tag{158}$$

The level $(4\ ^2S_{1/2},M_J=\frac{1}{2})$ is the ground state, while $(3\ ^2D_{5/2},M_J=\frac{3}{2})$ is a metastable level with a long lifetime (1.06 s). Both the electric dipole transition $4\ ^2S_{1/2}\rightarrow 4\ ^2P_{1/2}$ at 397 nm wavelength and the electric quadrupole transition $4\ ^2S_{1/2}\rightarrow 3\ ^2D_{3/2}$ at 732 nm are suitable for Doppler and sideband laser cooling, respectively. In Doppler cooling the laser radiation pressure slows down the axial motion of the ions until temperatures $T\sim$a few mK. To further reduce the temperature ($T\sim$a few μK) until no phonons are present, one resorts to sideband cooling (Hughes, 1998).

The interaction between Cirac-Zoller qubits is achieved using two types of degrees of freedom: internal (the electronic states of the ions) and external (the vibrational states of their collective excitations). Thus an active state for information processing is the tensor product of an electronic state and a quantum oscillator state of the axial potential, namely,

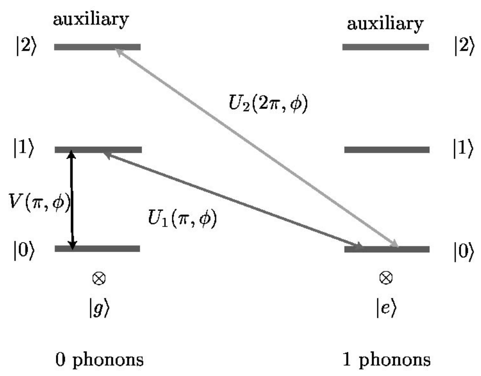


FIG. 46. Schematic representation of the transitions generated by the V and U pulses.

$$|\Psi\rangle=|x\rangle|\alpha\rangle,\quad x=0,1;\ \ \alpha=g,e,\tag{159}$$

where $|x\rangle$ refer to the electronic levels and $|g\rangle,|e\rangle$ denote the ground state and first excited state of the vibrational motion, respectively. In $|g\rangle$ there are no phonons present in the system, while there is one phonon in $|e\rangle$ (see Fig. 46).

2. Laser pulses

With this structure of states one can apply two types of laser Rabi pulses to the ions in order to achieve quantum logic operations. These are called V and U pulses.

The V pulse implements one-qubit operations. Its frequency is tuned to resonate with the optical transition between the qubit states. It swaps the electronic states $|0\rangle\leftrightarrow|1\rangle$ and leaves the vibrational mode in the ground state $|g\rangle$. The unitary evolution operator induced by this pulse is

$$\begin{aligned}&V(\theta,\phi):=e^{-itH_V/\hbar},\\&H_V:=\tfrac{1}{2}\hbar\Omega[e^{-i\phi}|1\rangle\langle 0|+e^{i\phi}|0\rangle\langle 1|],\end{aligned}\tag{160}$$

where $\theta:=\Omega t$, H_V is the V-pulse Hamiltonian, Ω is the Rabi frequency (proportional to the square root of the laser intensity), and ϕ is the laser phase. This pulse then produces the following action on the electronic states:

$$V(\theta,\phi):\begin{cases}|0\rangle\mapsto\cos\frac{\theta}{2}|0\rangle-ie^{-i\phi}\sin\frac{\theta}{2}|1\rangle\\|1\rangle\mapsto\cos\frac{\theta}{2}|1\rangle-ie^{i\phi}\sin\frac{\theta}{2}|0\rangle.\end{cases}\tag{161}$$

The U pulse is used to implement two-qubit operations. The laser frequency is now adjusted to induce simultaneously both an electronic and a vibrational transition. To help perform the desired logic gates, an auxiliary electronic state $|2\rangle$ (see Fig. 46) is available. The time-evolution operator led by this pulse is

$$\begin{aligned}&U_{\hat{x}}(\kappa,\phi):=e^{-itH_U(\hat{x})/\hbar},\quad \hat{x}=1,2,\\&H_U(\hat{x}):=\tfrac{1}{2}\hbar\,\eta\Omega[e^{-i\phi}|\hat{x}\rangle\langle 0|a+e^{i\phi}|0\rangle\langle\hat{x}|a^\dagger],\end{aligned}\tag{162}$$

where H_U is the U-pulse Hamiltonian, $\kappa := \eta\Omega t$, η is the Lamb-Dicke parameter,[63] and $a^\dagger, a$ are phonon creation and annihilation operators satisfying

$$a^\dagger|g\rangle = |e\rangle,\ \ a|e\rangle = |g\rangle,\ \ [a,a^\dagger] = 1. \tag{163}$$

Several physical constraints on these parameters in a linear ion trap must be fulfilled for it to function stably and as required (Cirac and Zoller, 1995).

The U pulse acts as follows:

$$U_{\hat{x}}(\kappa,\phi): \begin{cases} |0\rangle|g\rangle \mapsto |0\rangle|g\rangle \\ |0\rangle|e\rangle \mapsto \cos\frac{\kappa}{2}|0\rangle||e\rangle - ie^{-i\phi}\sin\frac{\kappa}{2}|\hat{x}\rangle|g\rangle \\ |\hat{x}\rangle|g\rangle \mapsto \cos\frac{\kappa}{2}|\hat{x}\rangle|g\rangle - ie^{i\phi}\sin\frac{\kappa}{2}|0\rangle|e\rangle. \end{cases} \tag{164}$$

3. Building logic gates

By controlling the duration of the laser pulses in Eqs. (161) and (164) we can perform logic operations in a fashion akin to those for spin qubits with Rabi pulses. The nice thing about the ion-trap quantum computer is that the same Rabi pulses can drive conditional logic when phonons are suitably put to work.

For instance, a CNOT gate can be constructed using a series of V and U pulses. To this end, we first reproduce a π controlled-phase gate [Eq. (78)] between qubits at sites i,j as follows:

$$U_{\mathrm{CPh}}^{(i,j)}(\pi) = U_1^{(i)}(\pi,0)\,U_2^{(j)}(2\pi,0)\,U_1^{(i)}(\pi,0). \tag{165}$$

The explicit action of this sequence of operations is shown in Fig. 47. This two-bit gate is constructed only out of U pulses.

In order to construct CNOT from this gate [see Eq. (79) and Fig. 25] we need to employ V pulses,

$$U_{\mathrm{CNOT}}^{(i,j)} = V^{(j)}(\tfrac{1}{2}\pi,\tfrac{1}{2}\pi)\,U_{\mathrm{CPh}}^{(i,j)}(\pi)\,V^{(j)}(-\tfrac{1}{2}\pi,\tfrac{1}{2}\pi), \tag{166}$$

where these V pulses correspond to Hadamard gates. Other logic gates involving a larger number of qubits can be constructed similarly using these basic pulse operations (Cirac and Zoller, 1995).

Let us note that the 2π auxiliary rotations in Eq. (165) do not produce any population of the auxiliary atomic levels nor of the center-of-mass levels. Thus a variation in the population of these levels when the gate is operated would indicate a faulty experimental realization.

Upon completion of the quantum operations in the ion-trap quantum computer, we need to read out the result (see Sec. IX). This is done by measuring the state

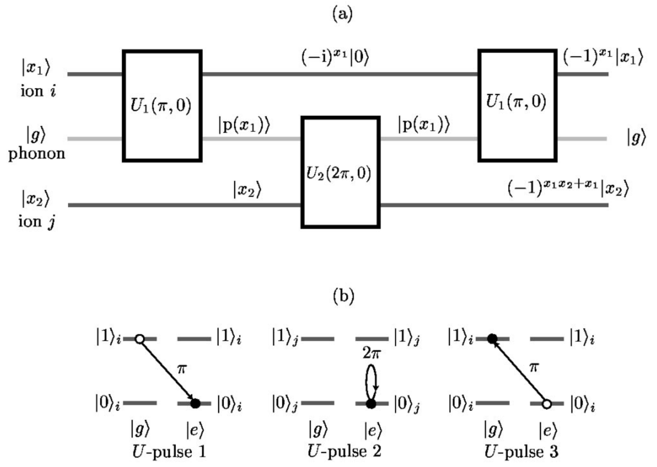


FIG. 47. Sequence of operations for a controlled-phase gate: (a) Quantum circuit for the controlled-phase gate in an ion-trap quantum computer. We denote by $|p(x_1)\rangle$ the phonon states $p(0) := g, p(1) := e$. Note also that the overall final phase is $(-1)^{x_1x_2}$, as it corresponds to a controlled phase $\phi = \pi$. (b) Evolution of a state under the sequence of U pulses in Eq. (165).

of each qubit in the quantum register using the quantum-jump technique (Bergquist *et al.*, 1986; Nagourney *et al.*, 1986; Sauter *et al.*, 1986). For instance, for the Ca$^+$ qubits (158), the laser is tuned to the dipole transition $4\ {}^2S_{1/2} \to 4\ {}^2P_{1/2}$ at 397 nm (see Fig. 45). There are two possibilities for the ion being addressed with the laser: (i) if the ion radiates (fluoresces), this means that its state is $|0\rangle$; (ii) if the ion does not radiate (remains dark), then it was in the $|1\rangle$ state. Therefore just by observing which ions fluoresce and which remain dark we can retrieve the bit values of the register. Actually, there is a third possibility in which $4\ {}^2P_{1/2} \to 3\ {}^2D_{3/2}$. In order to prevent this metastable level from being populated, a pump-out laser is also required.

4. Further applications

The ion-trap technique has also found applications in the preparation of entangled states (Molmer and Sorensen, 1999). This has been experimentally realized by the NIST group (Sackett *et al.*, 2000), who generated entangled states of two and four trapped ions. In Fig. 48 a four-qubit quantum register used in these experiments is shown.

Unavoidable errors impose computational limits on ion-trap quantum computers. Sources of these constraints are the spontaneous decay of the metastable state, laser phase decoherence, ion heating, and other kinds of errors. Using simple physical arguments it is possible to place upper bounds on the number of laser pulses N_U sustained by the ion trap before it enters a decoherence regime (Hughes, James, *et al.*, 1996),

$$N_U L^{1.84} < \frac{2Z(\tau/1\ \mathrm{s})}{A^{1/2}F^{3/2}(\lambda/1\ \mathrm{m})^{3/2}}, \tag{167}$$

where Z is the ion degree of ionization, τ is the lifetime of the metastable state, L is the number of ions, A is their atomic mass, F parametrizes the focusing capabil-

[63]This quantity is the ratio between the width of the ion oscillation in the vibrational ground state of the register and the (reduced) laser wavelength $\lambda_L/2\pi$: $\eta := (\hbar/2NM_{\mathrm{ion}}\omega_z)^{1/2} \times (2\pi/\lambda_L)$, where N is the number of cold ions and ω_z is the vibrational frequency of the register's center of mass along the trap axis. The Lamb-Dicke criterion $\eta \ll 1$ is required for Eq. (162) to be a good approximation (Cirac and Zoller, 1995). For the Ca$^+$ trap, with $N \sim 10$, $\omega_z \sim 100$ kHz, then $\eta \sim 0.2$.

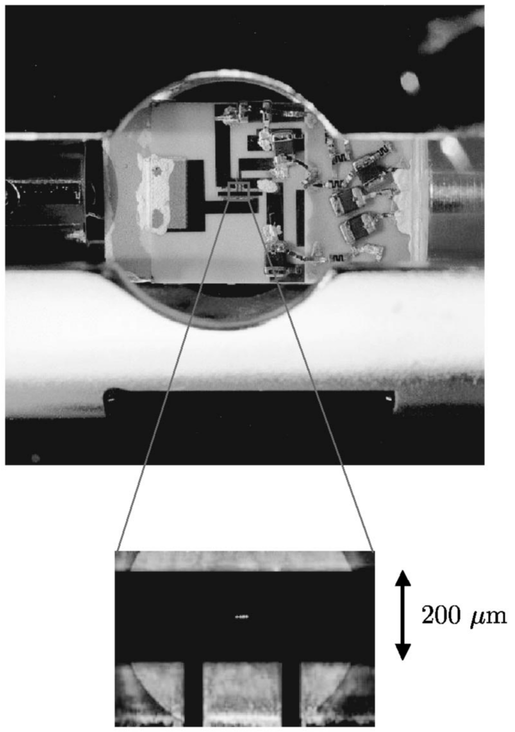


FIG. 48. Micromachined ion trap showing a four-qubit register in the inset. From Sackett *et al.*, 2000.

ity of the laser, and λ is the laser wavelength. This bound depends on the ion parameters A and τ, making some ion species more suitable than others.[64] With this bound it is possible to estimate the number of ions needed to factorize a 438-bit number using ytterbium [with the transition $4f^{14}6s\ {}^2S_{1/2}\leftrightarrow 4f^{13}6s^2\ {}^2F_{1/2}$, which has a very long lifetime (1533 days) and a wavelength of 467 nm]. Around 2200 trapped ions and 4.5×10^{10} pulses would be required to perform the desired factorization, in about 100 hours of computation time (Hughes, James, *et al.*, 1996).

Scalability of the ion-trap quantum computer is a central issue if we want to have a useful machine for number factoring and the like. With current techniques, it is believed that prospects for reaching a few tens of qubits are good (Hughes *et al.*, 1998). Cirac and Zoller (2000) have proposed an ion-trap-based quantum computer with a two-dimensional array of independent ion traps and a different ion (head) that moves above this plane. This setup is still conceptually simple and it is believed to be within reach of present experimental technologies.

C. NMR liquids: Quantum ensemble computation

We have seen that spin qubits and spin resonance are natural choices for performing quantum computations. Nuclear spins are good candidates for spin qubits but they pose both theorical and experimental challenges. There have been independent proposals for overcoming these difficulties, namely, the *logical labeling formalism* of Gershenfeld, Chuang, and Lloyd (1996) and Gershenfeld and Chuang (1997), and the *spatial averaging formalism* of Cory, Fhamy, and Havel (1997). A *time-averaging formalism* was introduced by Knill, Chuang, and Laflamme (1997). Several groups have attempted to realize these ideas experimentally.

The quantum hardware in this case consists of a liquid containing a large number of molecules of a certain type. The qubit is the spin of a nucleus in a molecule, and the quantum register is a molecule as a whole, i.e., each molecule is an independent quantum computer. Operations are effected using nuclear magnetic resonance techniques (Rabi oscillations), and information transmission between nuclei is based on the spin interactions within each molecule.

1. Spins at thermal equilibrium

The choice of nuclear spins as qubits has several pros and cons. On the one hand, nuclear spins in a molecule of a liquid are very robust quantum systems, for they are well screened from other sources of magnetic field by the electron cloud that surrounds them. This results in decoherence times of the order of seconds, long enough to let quantum computations take place. On the other hand, in a liquid at finite temperature the nuclear spins form a highly mixed state, not a pure state as we have been assuming in the formalism for quantum computation introduced so far. Such a formalism needs to be modified accordingly, by using density matrices to describe the mixed states of spins and their evolution.

A consequence of the finite temperature is that the precise initial conditions of a particular nuclear spin are not known, as would be required for standard quantum computation. Instead, we can only know the probability of finding the spin in one of the two states $|0\rangle=|\uparrow\rangle$ or $|1\rangle=|\downarrow\rangle$. In the following, we shall assume that the molecules in the solution are in thermal equilibrium at some temperature T. Hence the density matrix describing the quantum state of the relevant nuclear spins in each single molecule is

$$\rho := \frac{e^{-\beta H}}{\mathrm{Tr}[e^{-\beta H}]}, \tag{168}$$

where H is the Hamiltonian of the system, $\beta=1/k_B T$ is the inverse temperature, and the trace is over any orthonormal basis of the Hilbert space. Let us take the simplest case of a single spin qubit with a Zeeman-splitting Hamiltonian $H=\omega S^z$, $\omega=-\gamma B_0$. Equation (168) then becomes

$$\rho_{00}=\frac{e^{-\beta\hbar\omega/2}}{e^{\beta\hbar\omega/2}+e^{-\beta\hbar\omega/2}}, \tag{169}$$

$$\rho_{11}=\frac{e^{\beta\hbar\omega/2}}{e^{\beta\hbar\omega/2}+e^{-\beta\hbar\omega/2}},$$

[64] The number N_U refers only to the number of U pulses because they last much longer than the V pulses, which are thus neglected.

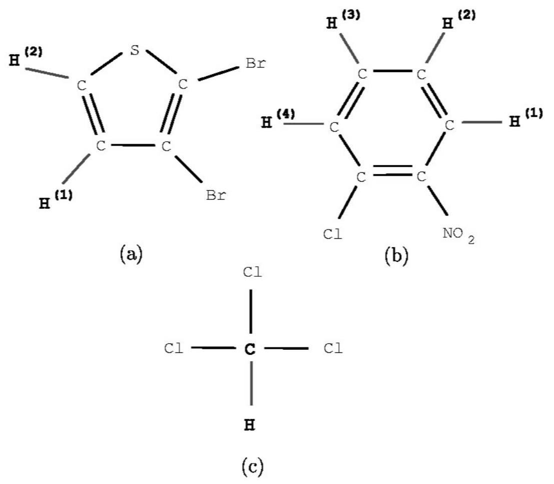


FIG. 49. Some examples of molecules used in NMR liquid quantum computation: (a) 2,3-dibromo-thiophene (homonuclear); (b) 1-chloro-2-nitro-benzene (homonuclear); (c) chloroform (heteronuclear).

$$\rho_{01}=0=\rho_{10}.$$

The diagonal terms of ρ represent the probability of finding the spin in the state $|0\rangle$ or $|1\rangle$. In contrast, the density matrix of a pure state $|\psi(t)\rangle := \alpha_0(t)|0\rangle + \alpha_1(t)|1\rangle$ is

$$\rho_\psi := |\psi\rangle\langle\psi| = \begin{pmatrix} |\alpha_0|^2 & \alpha_0\alpha_1^* \\ \alpha_0^*\alpha_1 & |\alpha_1|^2 \end{pmatrix}. \tag{170}$$

Therefore we see that at finite temperature and thermal equilibrium, the off-diagonal elements of the density matrix average to zero, while they are nonvanishing for a generic pure quantum state.

2. Liquid-state NMR spectroscopy

To overcome these difficulties, the proposal for a NMR quantum computer takes advantage of techniques that have been developed in liquid-state NMR spectroscopy over the past 50 years (Ernst *et al.*, 1987).

In a NMR liquid the molecules are in solution. Only some of the nuclei in each molecule are active for doing quantum computation. When the qubits consist of atomic nuclei of the same chemical element, the molecules are called *homonuclear*, when they are of a different element they are called *heteronuclear*. Figure 49 shows examples of homonuclear molecules, like 2,3-dibromo-thiophene, in which the active nuclear spins are those of the two hydrogen atoms; or 1-chloro-2-nitro-benzene with four active hydrogen atoms. An example of a heteronuclear molecule is the ^{13}C-labeled chloroform[65] in which the two active qubits come from

[65]The nucleus of the most common isotope ^{12}C is spinless. Adding one extra neutron endows it with an overall operative spin $\frac{1}{2}$.

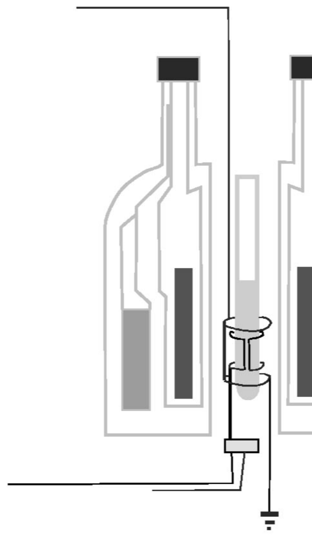


FIG. 50. Schematic setup of a NMR experiment. The liquid sample is in the middle tube surrounded by a radio-frequency cavity that produces a strong, homogeneous magnetic field. The apparatus is connected to electronic control devices not shown. From Cory *et al.* (2000).

the atoms of hydrogen and carbon. The number of qubits in the working register narrows the choice of the molecule structure.

An appropriate experimental setup for NMR computation is much like any other instrumentation used in NMR spectroscopy. In Fig. 50 the basic structure of a NMR spectrometer is shown. The liquid sample is held in a probe inside a radio-frequency cavity subjected to a strong homogeneous magnetic field of around 10 T, usually produced by a superconducting magnet. The RF cavity is tuned to the resonance frequencies of the active nuclear spins.

In a typical sample, the number of molecules N in solution is $\sim 10^{18}$. The dipole-dipole interactions between the spins in different molecules as well as other intermolecular interactions average to zero due to the random rotational motion of the molecules in the usual time scale for controlling the spin dynamics and the measurement (Slichter, 1990). Hence only interactions within each molecule are observable, and the sample can be regarded as an ensemble of independent and mutually incoherent quantum computers. This reasonable approximation yields a huge reduction in the large density matrix of dimension $\sim 2^{O(N)}$ describing the whole ensemble of active nuclear spins. Such a matrix may be replaced by a much smaller density matrix of dimension 2^n, where n is the number of active nuclei in a single molecule.

Within each molecule, the total Hamiltonian $H(t)$ of the active spins has two parts (Cory *et al.*, 2000), one internal and another external:

$$H(t) := H_{\text{int}} + H_{\text{ext}}(t). \tag{171}$$

The internal Hamiltonian describes the interactions among spins within the molecule, while the external Hamiltonian controls the spin dynamics under Rabi pulses. The operator $H_{\rm int}$ embodies (a) the molecule interaction energy with a strong homogeneous magnetic field that causes a Zeeman splitting of the nuclear-spin levels; and (b) the spin-spin interactions between active nuclei, modeled by a magnetic exchange interaction $2(J_{ij}/\hbar)\mathbf{S}_i\cdot\mathbf{S}_j$ mediated by electrons in molecular orbitals that overlap both nuclear spins i,j. In most cases this interaction can be further simplified using the weak-coupling approximation $|J_{ij}|\ll|\omega_i-\omega_j|$, which assumes that the spin-spin coupling is much smaller than the Zeeman splitting. This simplification produces a scalar coupling of Ising type between the spins and yields the following good approximation to the internal Hamiltonian:

$$H_{\rm int}\approx\sum_{i=1}^{n}\omega_i S_i^z+2\sum_{i\neq j=1}^{n}(J_{ij}/\hbar)S_i^zS_j^z, \tag{172}$$

where J_{ij} measures the coupling between the active spins at sites i,j,[66] and ω_i are the resonance frequencies for each spin. They are different even for homonuclear molecules due to the unlike screening of each nuclear spin from the surrounding electrons. This effect is called a *chemical shift*. Thus in Eq. (172) one-body terms may be used to distinguish qubits, while two-body terms serve to implement the conditional logic of two-qubit gates. The values of the parameters ω_i and J_{ij} are determined by standard NMR spectroscopy techniques prior to the computation. Standard NMR spectroscopy and NMR quantum computation share the means but differ in goals: in the former we aim to determine the parameters of the Hamiltonian (172) to study the chemistry and dynamics of the molecules in solution, while in the latter the form of Eq. (172) is already known and we set out to use it to perform controlled logic operations.

The external time-dependent Hamiltonian $H_{\rm ext}(t)$ helps to control the evolution of the spins. These form an ensemble of systems, initially described by the thermal density matrix ρ [Eq. (169)] and its time evolution is

$$\rho(t)=U(t)\rho(0)U^\dagger(t), \tag{173}$$

where $U(t)$ is the unitary propagator generated by the total Hamiltonian in Eq. (171) and $\rho(0)$ is the thermal density matrix (169).

3. High-temperature regime: pseudopure states

The evolution of the density matrix (168) is simplified in the high-temperature limit $k_{\rm B}T\gg\hbar\omega_i$, where the Zeeman splittings are much smaller than the Bolzmann energy. We can then approximate Eq. (168) as follows:

$$\rho\approx\frac{1-\beta H}{\mathrm{Tr}(1-\beta H)}\approx\rho_n:=\frac{1}{2^n}-\frac{\beta H}{2^n}. \tag{174}$$

Thus in NMR quantum computing there is no need to cool down the system until it reaches its ground state as in other types of quantum computers.

[66]In NMR spectroscopy J_{ij} are typically $\sim$100 Hz.

Let us analyze step by step the approximation (174) for quantum computing. First, let us consider the case of a single spin. The density matrix is simply given by

$$\rho_1:=\tfrac{1}{2}-\epsilon_1\delta_1,$$
$$\delta_1:=\overline{S_1^z},\quad \epsilon_1:=\tfrac{1}{2}\hbar\omega_1/k_{\rm B}T, \tag{175}$$

where δ_1 is called the *deviation density matrix*[67] and $|\epsilon_1|\sim10^{-5}$ at room temperature for conventional NMR liquids. The factor ϵ_1 gives the strength of the NMR signal relative to background noise. This expression can be further simplified by dropping out the unit term, which does not change under time evolution (173): in a NMR experiment the expectation value of an observable O is given by

$$\langle O\rangle=\mathrm{Tr}(O\rho), \tag{176}$$

and, as it happens, all NMR observables are traceless. Thus all the information is in $\epsilon_1\delta_1$. As ϵ_1 enters only as an overall scale factor, we can also drop it from this description and write the effective thermal density matrix simply as

$$\rho_1\sim\overline{S_1^z}. \tag{177}$$

Now let us recall that for a qubit in the ground state or excited state the density matrices are

$$\rho_{|0\rangle}=|0\rangle\langle0|=\tfrac{1}{2}+\overline{S^z},$$
$$\rho_{|1\rangle}=|1\rangle\langle1|=\tfrac{1}{2}-\overline{S^z}. \tag{178}$$

Discarding the unit terms, we see that for NMR purposes the one-qubit states $|0\rangle,|1\rangle$ are equivalent to $\overline{S^z}$, $-\overline{S^z}$, respectively. The spin operators representing one-qubit states in this correspondence are called *pseudopure* or *effective pure states*. This is also the case for a superposition state; for instance, the pure state $|\Psi\rangle=2^{-1/2}(|0\rangle+|1\rangle)$ has a density matrix

$$\rho_{|\Psi\rangle}=\tfrac{1}{2}+\overline{S^x}, \tag{179}$$

equivalent to $\overline{S^x}$. Actually, the correspondence is one-to-one in the case of one-qubit states, because the density matrix of a single pure state (170) is a Hermitian operator that can be expanded as a real linear combination of the Pauli matrices $\{1,\sigma^x,\sigma^y,\sigma^z\}$.

The time evolution of a NMR density matrix is that of the spin-$\frac{1}{2}$ operators. When the external Hamiltonian corresponds to a Rabi pulse, the transformation laws are simple. The evolution operator for a single spin with Zeeman Hamiltonian $H_1:=\hbar\omega_1\overline{S_1^z}$ is

$$U_{\rm Z}(t):=e^{-it\omega_1\overline{S_1^z}}=\cos(\tfrac{1}{2}\omega_1t)-2i\sin(\tfrac{1}{2}\omega_1t)\overline{S_1^z}, \tag{180}$$

whence the evolution of the one-qubit effective pure states:

$$U_{\rm Z}(t)\overline{S_1^x}U_{\rm Z}^\dagger(t)=\cos(\omega_1t)\overline{S_1^x}+\sin(\omega_1t)\overline{S_1^y},$$

[67]Sometimes it is also called a reduced density matrix.

$$U_Z(t)\overline{S_1^y}U_Z^\dagger(t) = -\sin(\omega_1 t)\overline{S_1^x} + \cos(\omega_1 t)\overline{S_1^y},$$
$$U_Z(t)\overline{S_1^z}U_Z^\dagger(t) = \overline{S_1^z}. \tag{181}$$

The Zeeman propagator $U_Z(t)$ rotates the spin around the z axis through an angle $\varphi := \omega_1 t$. It is customary to use the spectroscopic notation to denote the unitary action of the RF pulses in the rotating frame or interaction picture:

$$[\varphi]_i^\alpha := e^{-i\varphi\overline{S_i^\alpha}}, \quad \alpha = x,y, \quad i = 1,2,\ldots,n, \tag{182}$$

where φ is the rotation angle, α is the rotation axis, and i is the index labeling the rotating qubit. Thus the effect of a $[\pi]_1^x$ pulse

$$[\pi]_1^x = e^{-i\pi\overline{S_1^x}} = \begin{pmatrix} 0 & -i \\ -i & 0 \end{pmatrix} \tag{183}$$

is

$$\overline{S_1^z} \overset{[\pi]_1^x}{\rightarrow} -\overline{S_1^z}, \quad \text{i.e., } |0\rangle\langle 0| \leftrightarrow |1\rangle\langle 1|. \tag{184}$$

Therefore, with a $[\pi]_1^x$ pulse effected on a noninteracting ensemble of single spins in thermal equilibrium, we can simulate the quantum transition between the qubit states $|0\rangle$ and $|1\rangle$. In the thermal equilibrium ensemble, there are more populated ground states than populated excited states. After applying the pulse, the populations are reversed. Likewise, a $[\frac{1}{2}\pi]_1^x$ pulse produces off-diagonal terms in the density matrix at finite temperature that simulate quantum superpositions of pure states.

For multiqubit states, the correspondence between pure states and spin density matrices is not so simple. Let us consider the case of two-qubit states. It is possible to extend the description of a multispin density matrix using the *product operator formalism* of the NMR spectroscopists. Thus the density matrix for the pure ground state $|\Psi\rangle = |00\rangle$ is

$$\rho_{|\Psi\rangle} := |00\rangle\langle 00| = \tfrac{1}{2}\left(\tfrac{1}{2} + \overline{S_1^z} + \overline{S_2^z} + 2\overline{S_1^z}\,\overline{S_2^z}\right). \tag{185}$$

In general, any density matrix can be expanded in a tensor product basis of one-spin operators $\{\overline{S_i^x}, \overline{S_i^y}, \overline{S_i^z}\}_{i=1,\ldots,n}$. For n qubits,

$$\rho = \sum_{\alpha_1,\ldots,\alpha_n} c_{\alpha_1,\ldots,\alpha_n}\sigma_1^{\alpha_1}\cdots\sigma_n^{\alpha_n},$$
$$c_{\alpha_1,\ldots,\alpha_n} := 2^{-n}\mathrm{Tr}(\rho\,\sigma_1^{\alpha_1}\cdots\sigma_n^{\alpha_n}), \tag{186}$$

where $\alpha_i = 0,x,y,z$, and $\sigma_i^0 := 1$.

This has the advantage that the evolution of the ensemble density matrix is then simply determined through the evolution rules for single spin operators. The problem that we face now is that the thermal equilibrium matrix in the high-temperature limit $k_B T \gg \hbar\omega_i$ for the Hamiltonian (172) is

$$\rho_2 = \frac{1}{4} - \frac{1}{8}\hbar\beta\,\mathrm{diag}(\omega_1 + \omega_2 + J_{12}, \omega_1 - \omega_2 - J_{12},$$
$$-\omega_1 + \omega_2 - J_{12}, -\omega_1 - \omega_2 + J_{12}), \tag{187}$$

which is further approximated assuming a weak-coupling regime $|\omega_1 - \omega_2|, |J_{1,2}| \ll |\omega_1 + \omega_2|/2$ to

$$\rho_2 \approx \tfrac{1}{4} - \epsilon_2(\overline{S_1^z} + \overline{S_2^z}), \quad \epsilon_2 := \frac{1}{8}\hbar(\omega_1 + \omega_2)/k_B T, \tag{188}$$

and the corresponding deviation matrix $\delta_2 := \overline{S_1^z} + \overline{S_2^z}$ is not equivalent to the initial quantum ground state (185) we want to simulate. This is the *initialization problem* in NMR computing.

4. Logic gates with NMR

To prepare the ensemble of spins in the reference state (185) as well as to implement the logical operations for quantum processing, we use a series of well-known techniques in NMR liquid spectroscopy to carry out controlled time evolution of spins.

(i) *Rabi pulses*. The associated external Hamiltonian (171) corresponds to a harmonically oscillating magnetic field perpendicular to the Zeeman axis. It is applied at resonance and its effect on a single spin in the z direction is

$$[\varphi]_1^x:\ S_1^z \mapsto \cos(\varphi)S_1^z - \sin(\varphi)S_1^y,$$
$$[\varphi]_1^y:\ S_1^z \mapsto \cos(\varphi)S_1^z + \sin(\varphi)S_1^x, \tag{189}$$

where $\varphi := \Omega t$, t is the time duration, and Ω is the Rabi frequency.

(ii) *Chemical-shift pulses*. These pulses act as the propagator generated by the Zeeman part of the internal Hamiltonian (171). Their effect on the spin operators is given by Eq. (181).

(iii) *Scalar pulses*. These pulses induce the time evolution under the scalar coupling (two-spin) part of the internal Hamiltonian (171). For two qubits labeled 1,2, the scalar coupling propagator is also diagonal in the computational basis:

$$U_J(t) = e^{-i2J_{12}t\overline{S_1^z}\,\overline{S_2^z}} = \cos(\tfrac{1}{2}J_{12}t) - 4i\sin(\tfrac{1}{2}J_{12}t)\overline{S_1^z}\,\overline{S_2^z}, \tag{190}$$

and its effect on single spin operators is

$$U_J(t)\overline{S_1^x}U_J^\dagger(t) = \cos(J_{12}t)\overline{S_1^x} + 2\sin(J_{12}t)\overline{S_1^y}\,\overline{S_2^z},$$
$$U_J(t)\overline{S_1^y}U_J^\dagger(t) = \cos(J_{12}t)\overline{S_1^y} - 2\sin(J_{12}t)\overline{S_1^x}\,\overline{S_2^z}, \tag{191}$$
$$U_J(t)\overline{S_1^z}U_J^\dagger(t) = \overline{S_1^z}.$$

The NMR spectroscopic notation for these pulses is

$$[\varphi]_{12}^J := e^{-i2J_{12}t\overline{S_1^z}\,\overline{S_2^z}}, \tag{192}$$

where the rotation angle is $\varphi = J_{12}t$ and the subscript denotes the spins involved in the scalar pulse.

(iv) *Gradient pulses*. This is the technique used in the spatial averaging formalism of Cory, Fhamy, and Havel (1996, 1997). It consists in applying an external Hamiltonian (171) in the form of a field gradient along the liquid sample:

$$H_{\rm grad} = -\sum_{i=1}^{n} \gamma_i (z\,\partial_z B^z)_{z=z_i} S_i^z , \tag{193}$$

where z_i is the coordinate of the ith spin in the sample along the direction of the applied field gradient. This produces a spatially varying distribution of states throughout the sample. Its effect is to create a position-dependent phase shift with zero average, causing the vanishing of nondiagonal elements of the density matrix. The notation for these pulses is $[\text{grad}]^z$.

This gradient method is used to selectively turn off the transverse (x,y) spin factors in the product operator expansion of the density matrix, while leaving the rest untouched. For example, it is possible to induce the following transformation:

$$[\text{grad}]^z : \overline{S_1^z} + \overline{S_2^x} \mapsto \overline{S_1^z} . \tag{194}$$

The combined effect of the following series of pulses (Jones, 2000) produces the reference state (185) starting from the thermal ensemble of spins (188):[68]

$$\begin{aligned}
\overline{S_1^z} + \overline{S_2^z} &\overset{[\pi/3]_2^x}{\mapsto} \overline{S_1^z} + \frac{1}{2}\overline{S_2^z} - \frac{\sqrt{3}}{2}\overline{S_2^y} \\
&\overset{[\text{grad}]^z}{\mapsto} \overline{S_1^z} + \frac{1}{2}\overline{S_2^z} \\
&\overset{[\pi/4]_1^x}{\mapsto} \frac{1}{\sqrt{2}}\overline{S_1^z} - \frac{1}{\sqrt{2}}\overline{S_1^y} + \frac{1}{2}\overline{S_2^z} \\
&\overset{[\pi/2]_{12}^J}{\mapsto} \frac{1}{\sqrt{2}}\overline{S_1^z} + \frac{1}{\sqrt{2}} 2\overline{S_1^x}\,\overline{S_2^z} + \frac{1}{2}\overline{S_2^z} \\
&\overset{[-\pi/4]_1^y}{\mapsto} \frac{1}{2}\overline{S_1^z} - \frac{1}{2}\overline{S_1^x} + \frac{1}{2} 2\overline{S_1^x}\,\overline{S_2^z} + \frac{1}{2}\overline{S_2^z} + \frac{1}{2} 2\overline{S_1^z}\,\overline{S_2^z} \\
&\overset{[\text{grad}]^z}{\mapsto} \frac{1}{2}\overline{S_1^z} + \frac{1}{2}\overline{S_2^z} + \frac{1}{2} 2\overline{S_1^z}\,\overline{S_2^z} .
\end{aligned} \tag{195}$$

Once we have the reference state available, we can proceed to simulate other quantum states, applying a series of pulses to produce the desired ensemble of spin states. For instance, the density matrix of the Bell state $|\Psi\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$ in the product operator formalism is

$$\rho_{|\Psi\rangle} = \frac{1}{2}\left(\frac{1}{2} + 2\overline{S_1^z}\,\overline{S_2^z} + 2\overline{S_1^x}\,\overline{S_2^x} - 2\overline{S_1^y}\,\overline{S_2^y}\right), \tag{196}$$

which can be reached from the ground state $|00\rangle$ with the unitary operator

$$U = e^{-i\pi \overline{S_1^x}\,\overline{S_2^y}} . \tag{197}$$

This propagator, in turn, can be simulated with the following series of NMR pulses (from right to left):

$$[\tfrac{1}{2}\pi]_2^x [-\tfrac{1}{2}\pi]_1^y [\tfrac{1}{2}\pi]_{12}^J [\tfrac{1}{2}\pi]_1^y [-\tfrac{1}{2}\pi]_2^x : \rho_{|00\rangle} \mapsto \rho_{|\Psi\rangle} . \tag{198}$$

Likewise, the controlled-NOT gate is simulated by the following sequence:

$$[-\tfrac{1}{2}\pi]_2^y [-\tfrac{1}{2}\pi]_2^z [\tfrac{1}{2}\pi]_1^z [\tfrac{1}{2}\pi]_{12}^J [\tfrac{1}{2}\pi]_2^y . \tag{199}$$

In a similar fashion, one can implement other quantum states and logic gates. Actually, this NMR pulse technique has been so highly developed that it is possible to simulate the propagator of a set of interacting spins with any desired couplings, even turning on and off certain spin couplings at will. For this reason, this capability for controlling the NMR dynamics is referred to as *spin choreography* (Freeman, 1998).

The logical labeling formalism of Gershenfeld and Chuang (1997) uses a different strategy to prepare pseudopure states. It is based on the appropriate embedding of a set of spin states into a larger system. It does not resort to field gradients but instead uses these auxiliary spin states to implement the quantum computation with several qubits. There are also experimental realizations of this scheme (Vandersypen *et al.*, 1999).

5. Measurements

Once the NMR computation is completed, we have to read out the result from the spectrometer. This is done by measuring the macroscopic magnetization of the liquid sample with a detection coil (see Fig. 50). This bulk magnetization induces currents in the transverse RF coil, which is tuned to the resonance frequency. The RF coil generates a dipole field, and only the dipolar components of the density matrix oriented along the transverse magnetic field will couple to the measurement device.

In computing with NMR ensembles, measuring an observable (176) entails a perturbation milder than for pure states, where measurement is a strong projective process. The measured currents are proportional to the trace (Cory, Laflamme, *et al.*, 2000)

$$\text{Tr}\left(\sum_{i=1}^{n} \overline{S_i^+}\,\rho\right), \tag{200}$$

with $\overline{S_i^+} := \overline{S_i^x} + i\overline{S_i^y}$. For instance, Fig. 51 shows the signal (200) due to the precession induced on S_i^x, $i=1,2$, by chemical-shift and scalar-coupling pulses acting on a two-qubit molecule such as the 2,3-dibromo-thiophene of Fig. 49(a). This is the Fourier-transformed real part of the signal (Cory, Price, and Havel, 1997) and clearly shows the population peaks corresponding to the four states of a two-spin system depicted in Fig. 43. This is called an *in-phase doublet* because both peaks have the same sign. For different series of pulses the pattern of the signal changes accordingly, and this allows us to retrieve the information contained in the ensemble of states. When implementing simple quantum algorithms with NMR liquid spectroscopy, the output retrieval is performed by analyzing a subset of resonances, but in more general situations the technique of quantum-state

[68]This sequence is not necessarily unique.

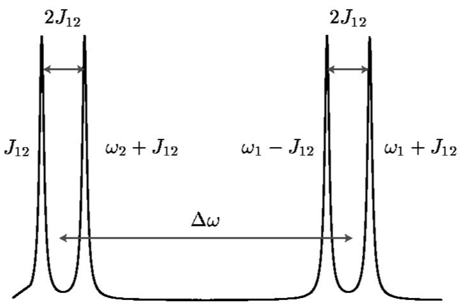


FIG. 51. Schematic signal from a NMR liquid spectrometer corresponding to an in-phase doublet for a two-spin system with energy levels as in Fig. 43. Notice that here the frequencies are positive.

tomography is used to systematically obtain the final quantum state (Knill, Chuang, and Laflamme, 1998).

6. Achievements and limitations

There is an extensive list of experimental achievements in NMR quantum computing (Cory *et al.*, 2000). To cite only a few of them, two-qubit gates have been constructed by several groups (Cory, Fahmy, and Havel, 1996; Chuang *et al.*, 1998; Collins *et al.*, 1999), the Toffoli gate has been implemented by Price *et al.* (1999), the quantum Fourier transform by Weinstein, Lloyd, and Cory (1999), quantum teleportation by Nielsen, Knill, and Laflamme (1998), etc. There are also NMR experiments involving seven qubits (Knill *et al.*, 2000). An alternative approach to implementing NMR quantum computation uses geometric phase-shift gates (Jones *et al.*, 2000) in which the controlled phases are Berry phases.

Despite the list of successes in NMR quantum computing, there are currently strong limitations in the scalability of the pseudopure-state preparation: it is clear from Eq. (174) that the deviation density matrix used in high-temperature NMR scales down exponentially with the factor 2^{-n}. This is a severe limitation that reduces the ratio of the observable signal to the background noise. To overcome this inefficiency we would need an exponentially large system.[69] It is currently estimated that it is not possible to go much beyond ten qubits using NMR liquid-state methods. This and other shortcomings have led to the pursuit of other NMR-like proposals, but this time based on solid-state samples (Cory *et al.*, 2000), with the aim of using true pure states. The goals set for these proposals are to reach 10–30 qubits, still not enough for competitive purposes.

The use of mixed states in NMR computing and the fact that they are exponentially inefficient have raised doubts about the truly quantum nature of the computations carried out by NMR liquid spectroscopy. The main objection comes from the results of Braunstein, Caves, *et al.* (1999) showing that all the pseudopure states used so far in NMR are separable, with no entanglement. This does not invalidate the speedup obtained with the NMR implementation of quantum algorithms (Chuang, Gershenfeld, and Kubinec, 1998; Jones and Mosca, 1998; Jones, Mosca, and Hansen, 1998).[70]

[69]This is something that happens in classical DNA computing (Adleman, 1994), where there is a tradeoff between exponential computing time for solving a problem and exponential space for molecular states.

D. Solid-state quantum computers

There are several proposals for building a quantum computer with some sort of solid-state device. We have just mentioned that a possible cure for the shortcomings of bulk NMR liquid computation is precisely resorting to solid NMR techniques. One type of proposal uses macroscopic superconducting devices with a radio-frequency superconducting quantum interference device (SQUID) as the qubit (Averin, 1998). The presence of 0 or 1 quantum of flux is the two-state system. Several ways exist to couple the SQUID's to make logic circuits, such as using Josephson tunnel junctions (Makhlin, Schön, and Shnirman, 2001). Another type of design would rely on quantum-dot nanotechnology. Barenco, Deutsch, *et al.* (1995) proposed using both charge and spin degrees of freedom for qubits in quantum dots, addressed, respectively, with electric and magnetic fields. Loss and DiVincenzo (1998) have developed in depth the theory of spin-based quantum computing. For details see the recent review of Burkard, Engel, and Loss (2000).

The list of experimental proposals is too long to be covered in detail here. Instead we shall focus on one of the most original proposals for solid-state quantum computation, Kane's idea (Kane, 1998) of building a silicon-based quantum computer. This is an appealing program, for Kane envisages the possibility of using the same semiconductors now used in most conventional computer electronics. The challenges to achieving this goal are still enormous, but the belief is that silicon technology is a very rapidly developing field and has some chance of overcoming those challenges.

The quantum hardware in Kane's proposal is an array of nuclear spins located on donors in silicon. The qubit is an individual nuclear spin of phosphor ^{31}P atoms; the quantum register is the whole array of ^{31}P dopants in silicon ^{28}Si; operations are carried out using a combination of magnetic-resonance techniques (Rabi pulses) with static electric fields; information is exchanged between nearby ^{31}P nuclear spins by means of the surrounding electrons.

1. Semiconductors for quantum computation

The choice of nuclear spins in this case is again motivated by their extreme isolation from the environment, as in the NMR proposal. A further requirement now is

[70]Whether working with separable states in NMR spectroscopy is a truly quantum computation is still a controversial issue (Jones, 2001).

that the dopant spins must not interact appreciably with the spins of the host semiconductor. To guarantee this we require that the chemical elements of the host have zero nuclear spin, $S=0$, to avoid undesired spin couplings. This singles out the semiconductor group V as a host candidate and removes other groups like III (with Ga) and IV (with As). Silicon ^{28}Si is an example of a stable isotope in group V.

In contrast to NMR liquid spectroscopy, computation following Kane's approach neither uses bulk spin nor resorts to macroscopic magnetization measurements. Instead, it truly needs to address spins individually for initialization and readout, and this is precisely one of the open challenges.

The basic ingredient in Kane's proposal is to replace direct nuclear-spin interactions by electronic detections, which are likely to be easier to handle. Thus the spin state of an individual nucleus dopant on a semiconductor will not be detected directly, but through its hyperfine interaction with the surrounding electrons. The hyperfine interaction is proportional to the probability density of the electrons at the nucleus. The electronic cloud is sensitive to electric voltages and can in principle be externally manipulated. Moreover, in certain cases the electronic wave functions extend far enough so as to overlap with those of a neighboring atom, thereby producing an indirect coupling between nuclear spins mediated by the atomic electrons. This indirect electron coupling can also be enhanced by applying external electric fields.

These conditions are met by shallow-level donors like ^{31}P, for which the range of the electron wave function is of order 10–100 Å. In addition, within the group V, the only shallow donor in Si with nuclear spin $S=\frac{1}{2}$ is precisely ^{31}P. Therefore the ^{31}P:Si system is a good candidate for a silicon-based quantum computer. For instance, at low ^{31}P concentrations and low temperature, $T=1.5$ K, the electron-spin relaxation time is of order 10^3 s, and the nuclear-spin relaxation time is over 10 h. If the temperature is further reduced to $T\sim 1$ mK, the phonon-limited ^{31}P relaxation time is likely of the order of 10^{18} s (Kane, 1998).

2. External control fields

We see that in Kane's idea the electrons play a role similar to that of phonons in the Cirac-Zoller gate: they mediate the conditional interactions between the real qubits. Likewise, we also need external electric fields to bring dopant nuclei close enough to interact. In all, we need to control three types of external fields:

(1) Electric gates above the donors to control individual electronic states (see Fig. 52).
(2) Electric gates between the donors to control interactions between qubits.
(3) Constant B and oscillating $B_{\rm ac}$ magnetic fields to execute operations on the individual spins much akin to those we have described for nuclear-spin resonance.

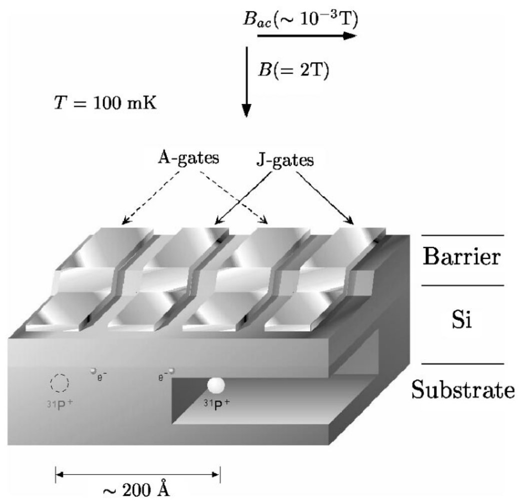


FIG. 52. Schematic design of a silicon-based quantum computer under study by the group at the University of New South Wales.

The scenario for replacing a Si vacancy by a P dopant atom is possible because both elements have similar sizes. Of the five outer $(3p)$ electrons in a ^{31}P atom (one more than in Si), four of them will form covalent bonds with neighoring Si atoms, while the remaining fifth electron is loosely bound to the ^{31}P atom. This outer electron and the rest of the dopant atom behave in first approximation as a hydrogenlike atom embedded in a Si environment. At low temperatures, the electron state is $1s$, and this yields a large hyperfine interaction. The effective Bohr radius is estimated at 30 Å. To proceed with the quantum computation we need this electron to remain in its ground state and to apply an external constant magnetic field to break the spin degeneracy. These conditions are met if $2\mu_{\rm B}B \gg k_{\rm B}T$, as for the typical values $B \geqslant 2$ T and $T \leqslant 100$ mK.

3. Logic gates

The description of the basic gate operations is the following.

(i) *One-qubit A gate*. The terminology is due to the A coupling constant of the hyperfine interaction between nuclear and electron spins. Single spin control is achieved by externally changing the voltage on a gate electrode (A gate) located on top of each nucleus (see Fig. 52); spin flips are then driven by a Rabi pulse tuned to the resonance frequency for the particular spin.

The one-qubit Hamiltonian H_1 modeling the interaction between the nuclear spin (denoted by n) and the electronic spin (denoted by e) in the presence of a constant magnetic field B is

$$H_1 := H_{1,\rm Z} + (A/\hbar^2)\mathbf{S}_{\rm n,1}\cdot\mathbf{S}_{\rm e,1},$$

$$H_{1,\rm Z} := -\gamma_{\rm n}S^z_{\rm n,1}B - \gamma_{\rm e}S^z_{\rm e,1}B, \tag{201}$$

where $\mathbf{S}_{\rm n,1}, \mathbf{S}_{\rm e,1}$ are the nuclear and electron spins, $\gamma_{\rm n}\mathbf{S}_{\rm n,1}, \gamma_{\rm e}\mathbf{S}_{\rm e,1}$ their corresponding magnetic moments, and

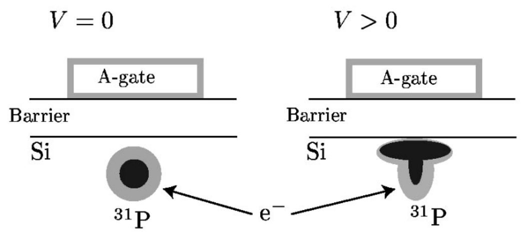


FIG. 53. Pictorial representation of an A gate that controls the nucleus-electron system (201). An externally applied electric field shifts the electron wave function from the donor ^{31}P, reducing the contact hyperfine interaction (202).

$$A:=-\frac{8\pi}{3}\overline{\gamma}_{\rm n}\overline{\gamma}_{\rm e}|\Psi(0)|^2 \text{ with } \overline{\gamma}_{\rm n}:=\hbar\gamma_{\rm n}, \overline{\gamma}_{\rm e}:=\hbar\gamma_{\rm e} \tag{202}$$

is the contact hyperfine interaction energy, with $|\Psi(0)|^2$ the probability density of the electron wave function at the nucleus position. Note that $\overline{\gamma}_{\rm e}=-g_{\rm e}\mu_{\rm B}, \overline{\gamma}_{\rm n}=g_{\rm n}\mu_{\rm N}$, where $g_{\rm e}=2.00$, is the relevant electron Landé g factor and $g_{\rm n}\approx 2\times 1.13$ is the nuclear gyromagnetic factor in ^{31}P:Si. Under operating conditions the electron remains in its ground state, and the separation of the nuclear spin levels is, to second order in the hyperfine coupling $A \ll \overline{\gamma}_{\rm n}B$,[71]

$$\hbar\omega_A=\overline{\gamma}_{\rm n}B+\frac{A}{2}-\frac{A^2}{4\overline{\gamma}_{\rm e}B}. \tag{203}$$

In ^{31}P:Si, $A/2h=58$ MHz and therefore $A>\overline{\gamma}_{\rm n}B$ for $B<3.5$ T. We can have control over this energy gap with the static electric field applied with the A gate (see Fig. 52). This shifts the electron wave function away from the nucleus (see Fig. 53) and reduces the hyperfine interaction A in Eq. (202). Thus the frequency (203) of the nuclear spins is controlled externally and this allows us to bring them into resonance with the oscillating pulse $B_{\rm ac}$ in order to effect arbitrary one-spin rotations.

(ii) *Two-qubit J gate*. The name is suggested by the J spin-exchange coupling between electron spins. Conditional logic operations are possible because of electron-mediated interactions between the nuclear spins of two Kane's qubits when brought sufficiently close by an externally applied voltage (J) gate (see Fig. 52). The two-qubit Hamiltonian is then

$$H_{12}=\sum_{i=1}^{2}(H_{i,{\rm Z}}+A_i\overline{\mathbf{S}}_i^{\rm n}\cdot\overline{\mathbf{S}}_i^{\rm e})+J\overline{\mathbf{S}}_1^{\rm e}\cdot\overline{\mathbf{S}}_2^{\rm e}, \tag{204}$$

where $H_{i,{\rm Z}}$ are the Zeeman Hamiltonians for each qubit (201), A_i are the hyperfine couplings for each nucleus-electron system, and J is the exchange coupling interaction between electron spins. This exchange energy depends on the overlap of the electron wave functions. Treating the ^{31}P dopants as hydrogenlike atoms in first approximation, we can estimate the J coupling for well-separated donors as (Herring and Flicker, 1964)

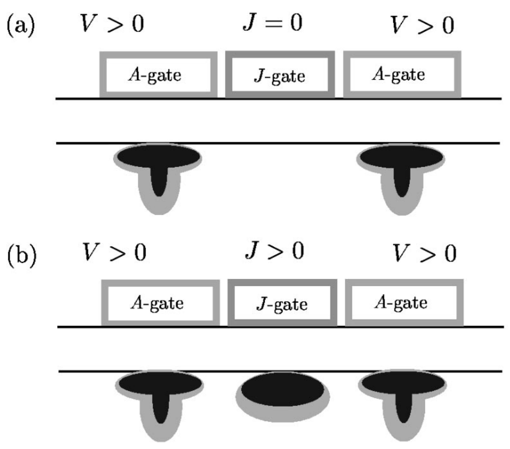


FIG. 54. Pictorial representation of a J gate that controls the nucleus-electron-nucleus system (204). When the electrostatic potential of the J gate is (a) off or (b) on, the J-exchange coupling in Eq. (204) is reduced or enhanced, respectively.

$$J(r)\approx 1.6\frac{e^2}{\epsilon a_{\rm B}}\left(\frac{r}{a_{\rm B}}\right)^{5/2}e^{-2r/a_{\rm B}}, \tag{205}$$

with r the interdonor distance, $\epsilon=11.7$ the Si dielectric constant, and $a_{\rm B}$ the Bohr radius of the atom. As the J coupling depends on the electron overlap, we can again use a voltage gate between donors to distort the electron clouds in order to control their coupling strength (see Fig. 54). This coupling will be significant when $J\sim|\overline{\gamma}_{\rm e}|B/2$, and this corresponds to a donor separation of order 100–200 Å (Kane, 1998), which is not far from the current limits of atom-scale lithography.

The relevant energy levels for doing quantum computation with a two-qubit Hamiltonian (204) are easily found (Berman *et al.*, 2000). This Hamiltonian is a 16×16 matrix. We shall label the basis states with the z components of the nuclear and electron spins at each donor site, with $|0\rangle_{\rm n},|1\rangle_{\rm n}$ denoting nuclear spins (up and down) and $|\uparrow\rangle_{\rm e},|\downarrow\rangle_{\rm e}$ for the electron spins; for instance,

$$|11\rangle_{\rm n}|\downarrow\downarrow\rangle_{\rm e} \tag{206}$$

represents a state with both nuclear and electron spins down.

In the presence of a static magnetic field and for low temperatures ($k_{\rm B}T\ll|\overline{\gamma}_{\rm e}|B$), the electrons remain with the spins down polarized $|\downarrow\downarrow\rangle_{\rm e}$. For example, $B=2$ T, $T=100$ mK meet this requirement. However, we shall see that switching on the J gate may change such a state, which will be the basis for doing spin measurements.

The essence of the functioning of the J gate is to enhance the overlap between the electron wave functions of two nearest ^{31}P donors. In this way, the ^{31}P nuclear spins (Kane qubits) can be indirectly coupled to one another through the electron-mediated interaction J. To operate two-qubit quantum logic gates, we need to address individually the four nuclear spin states $\{|00\rangle_{\rm n},|01\rangle_{\rm n},|10\rangle_{\rm n},|11\rangle\}_{\rm n}$. For simplicity, we assume A_1

[71]We have also approximated $-\overline{\gamma}_{\rm e}B+\overline{\gamma}_{\rm n}B$ by $-\overline{\gamma}_{\rm e}B$ in the denominator of Eq. (203).

$=A_2=A$. In the absence of J coupling the states $|01\rangle_n|\downarrow\downarrow\rangle_e, |10\rangle_n|\downarrow\downarrow\rangle_e$ are degenerate. These states belong to the sector of total z component of spin $\overline{S}^z_{\rm tot} := (\overline{S}^z_{1,n}+\overline{S}^z_{2,n})+(\overline{S}^z_{1,e}+\overline{S}^z_{2,e})=-1$. The role of the J gate is precisely to control this energy splitting, which we now try to estimate.

Let us consider the Kane implementation of the CNOT gate (Goan and Milburn, 2000). There are four steps involved.

(1) We start with $J=A_2-A_1=0$, so that the states $\{|00\rangle_n|\downarrow\downarrow\rangle_e, |01\rangle_n|\downarrow\downarrow\rangle_e, |10\rangle_n|\downarrow\downarrow\rangle_e, |11\rangle_n|\downarrow\downarrow\rangle_e\}$ have energies

$$E_{|00\rangle_n|\downarrow\downarrow\rangle_e} = -\sqrt{(-\overline{\gamma}_e+\overline{\gamma}_n)^2B^2+A^2}-\tfrac{1}{2}A,$$

$$E_{|01\rangle_n|\downarrow\downarrow\rangle_e} = E_{|10\rangle_n|\downarrow\downarrow\rangle_e} = \tfrac{1}{2}[(\overline{\gamma}_e+\overline{\gamma}_n)B-\sqrt{(-\overline{\gamma}_e+\overline{\gamma}_n)^2B^2+A^2}], \tag{207}$$

$$E_{|11\rangle_n|\downarrow\downarrow\rangle_e} = (\overline{\gamma}_e-\overline{\gamma}_n)B+\tfrac{1}{2}A.$$

(2) Next we introduce a bias between the two A gates by adiabatically switching on a difference $\triangle A := A_1 - A_2$ in their couplings, while keeping $J=0$. This splits the degeneracy of the $|01\rangle_n|\downarrow\downarrow\rangle_e, |10\rangle_n|\downarrow\downarrow\rangle_e$ states, allowing us to choose one as a control qubit and the other as a target qubit. The energies in Eq. (207) become

$$E_{|00\rangle_n|\downarrow\downarrow\rangle_e} = -\tfrac{1}{2}[\sqrt{(-\overline{\gamma}_e+\overline{\gamma}_n)^2B^2+A_1^2} + \sqrt{(-\overline{\gamma}_e+\overline{\gamma}_n)^2B^2+A_2^2}]-\tfrac{1}{4}(A_1+A_2),$$

$$E_{|01\rangle_n|\downarrow\downarrow\rangle_e} = -\tfrac{1}{4}\triangle A+\tfrac{1}{2}[(\overline{\gamma}_e+\overline{\gamma}_n)B - \sqrt{(-\overline{\gamma}_e+\overline{\gamma}_n)^2B^2+A_1^2}],$$

$$E_{|10\rangle_n|\downarrow\downarrow\rangle_e} = \tfrac{1}{4}\triangle A+\tfrac{1}{2}[(\overline{\gamma}_e+\overline{\gamma}_n)B - \sqrt{(-\overline{\gamma}_e+\overline{\gamma}_n)^2B^2+A_2^2}],$$

$$E_{|11\rangle_n|\downarrow\downarrow\rangle_e} = (\overline{\gamma}_e-\overline{\gamma}_n)B+\tfrac{1}{4}(A_1+A_2), \tag{208}$$

and the corresponding eigenstates are still $\{|00\rangle_n|\downarrow\downarrow\rangle_e, |01\rangle_n|\downarrow\downarrow\rangle_e, |10\rangle_n|\downarrow\downarrow\rangle_e, |11\rangle_n|\downarrow\downarrow\rangle_e\}$, predominantly.

(3) Once the two qubits are distinguished energetically it is time to introduce, again adiabatically, the J coupling to bring the states $|10\rangle_n$ and $|01\rangle_n$ to the symmetric and antisymmetric combinations,

$$|10\rangle_n \mapsto |s\rangle_n := 2^{-1/2}(|01\rangle_n+|10\rangle_n),$$

$$|01\rangle_n \mapsto |a\rangle_n := 2^{-1/2}(|01\rangle_n-|10\rangle_n). \tag{209}$$

For this purpose it is necessary to keep J at full strength before adiabatically switching off $\triangle A$.

The energies of the new eigenstates, in the presence of both A and J couplings, with $\triangle A=0$, can be computed exactly by diagonalizing H_{12} in the sectors of a fixed total third component $S^z_{\rm tot}$ of the spin, since this is a conserved quantity. Only the values $S^z_{\rm tot}=-2,-1,0$ are relevant for our discussion, since our initial states lie there. First we need to know the energy splitting $\hbar\omega_J$ between the symmetric and antisymmetric qubit states in the sector $S^z_{\rm tot}=-1$. Second, to control the Rabi pulse in the coming step, we must also know the gap energy $\hbar\omega_{\rm ac}$ between $|s\rangle_n|\downarrow\downarrow\rangle_e$ and $|11\rangle_n|\downarrow\downarrow\rangle_e$.

To calculate $\hbar\omega_J$ we use the reduced basis

$$\{|01\rangle_n|\downarrow\downarrow\rangle_e, |10\rangle_n|\downarrow\downarrow\rangle_e, |11\rangle_n|\downarrow\uparrow\rangle_e, |11\rangle_n|\uparrow\downarrow\rangle_e\} \tag{210}$$

to express the Hamiltonian H_{12} in the sector $S^z_{\rm tot}=-1$ as the following matrix:

$$H_{(-1)} = \begin{pmatrix} \frac{1}{4}J+\overline{\gamma}_eB & 0 & 0 & \frac{1}{2}A \\ 0 & \frac{1}{4}J+\overline{\gamma}_eB & \frac{1}{2}A & 0 \\ 0 & \frac{1}{2}A & -\frac{1}{4}J+\overline{\gamma}_nB & \frac{1}{2}J \\ \frac{1}{2}A & 0 & \frac{1}{2}J & -\frac{1}{4}J+\overline{\gamma}_nB \end{pmatrix}. \tag{211}$$

As $A_1=A_2=A$, the two-qubit Hamiltonian is symmetric under the site labels and its eigenvectors can be either symmetric or antisymmetric under this exchange. The two symmetric (unnormalized) eigenstates are given by

$$|s,\pm\rangle := (\overline{\gamma}_nB+\tfrac{1}{4}J-E_{s,\pm})|s\rangle_n|\downarrow\downarrow\rangle_e+\tfrac{1}{2}A|00\rangle_n|s\rangle_e, \tag{212}$$

where

$$|s\rangle_e := \frac{1}{\sqrt{2}}(|\downarrow\uparrow\rangle_e+|\uparrow\downarrow\rangle_e),$$

$$E_{s,\pm} := \tfrac{1}{2}(\overline{\gamma}_e+\overline{\gamma}_n)B+\tfrac{1}{4}J\pm\tfrac{1}{2}\sqrt{(-\overline{\gamma}_e+\overline{\gamma}_n)^2B^2+A^2}. \tag{213}$$

Similarly the two antisymmetric (un-normalized) eigenstates are

$$|a,\pm\rangle := -(-\overline{\gamma}_eB-\tfrac{1}{4}J+E_{a,\pm})|00\rangle_n|a\rangle_e - \tfrac{1}{2}A|a\rangle_n|\downarrow\downarrow\rangle_e, \tag{214}$$

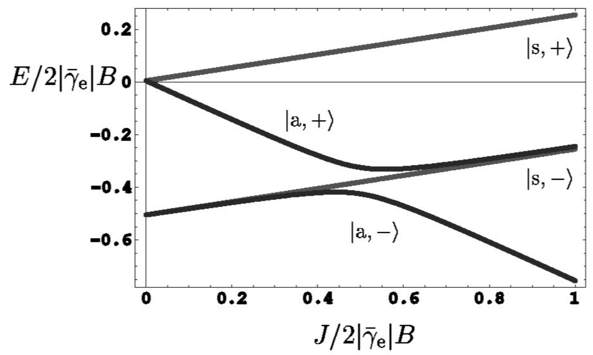


FIG. 55. Energy levels for a two-donor interacting system as a function of the exchange coupling J, for $A=0.2|\overline{\gamma}_e|B$.

with

$$|a\rangle_e := \frac{1}{\sqrt{2}}(|\downarrow\uparrow\rangle_e - |\uparrow\downarrow\rangle_e),$$

$$E_{a,\pm} := \tfrac{1}{2}(\overline{\gamma}_e + \overline{\gamma}_n)B - \tfrac{1}{4}J \pm \tfrac{1}{2}\sqrt{[(-\overline{\gamma}_e+\overline{\gamma}_n)B - J]^2 + A^2}. \tag{215}$$

In Fig. 55 the energies $E_{s,\pm}, E_{a,\pm}$ are plotted against the exchange coupling constant J. For a two-electron spin system with antiferromagnetic coupling ($J>0$), the exchange interaction lowers the energy of the spin singlet with respect to the triplet. When a static magnetic field is applied, the electron ground state is $|\downarrow\downarrow\rangle_e$ for $J<|\overline{\gamma}_e|B$. The exchange coupling can be increased adiabatically by external manipulation of the J voltage gate. For $J>|\overline{\gamma}_e|B$, the electron ground state is a singlet. The value $J=|\overline{\gamma}_e|B$ corresponds to the case in which levels $E_{a,+}$ and $E_{s,-}$ avoid their crossing (Fig. 55). The energy splitting to be controlled with the J gate is $\hbar\omega_J := E_{s,-} - E_{a,-}$, which can be estimated using the exact formulas (213) and (215) and treating the hyperfine interaction as a small perturbation (assuming $J<|\overline{\gamma}_e|B$):

$$\hbar\omega_J \simeq \frac{A^2}{4}\left(\frac{1}{|\overline{\gamma}_e|B-J} - \frac{1}{|\overline{\gamma}_e|B}\right). \tag{216}$$

For the ^{31}P:Si system at $B=2$ T and $J/h=30$ GHz, Eq. (216) gives $\nu_J=75$ kHz as the nuclear-spin exchange frequency. This is roughly the rate at which binary operations can be performed in the purported quantum computer. Recall that the speed for individual spin operations is determined by the oscillating field B_{ac}, and this speed is comparable to 75 kHz when $B_{ac}\sim 10^{-3}$ T.

Finally, to calculate the gap $\hbar\omega_{ac}$, we just need the energy of the state $|11\rangle_n|\downarrow\downarrow\rangle_e$, which lies in the trivial sector $S^z_{tot}=-2$:

$$E_{|11\rangle_n|\downarrow\downarrow\rangle_e} = (\overline{\gamma}_e + \overline{\gamma}_n)B + \tfrac{1}{4}J + \tfrac{1}{2}A. \tag{217}$$

(4) Now we can carry out the CNOT operation. This amounts to swapping the states $|s\rangle_n$ and $|11\rangle_n$, which are well separated in energies by previous steps, while leaving the two other states untouched. To this end, it suffices to apply a Rabi pulse $H_{ac}(t) := -\gamma_n(S^x_{n,1} + S^x_{n,2})B_{ac}\sin\omega_{ac}t$ resonant with the separation energy between the states to be exchanged. Although the gaps $E_{|11\rangle_n|\downarrow\downarrow\rangle_e} - E_{|s\rangle_n|\downarrow\downarrow\rangle_e}$ and $E_{|a\rangle_n|\downarrow\downarrow\rangle_e} - E_{|00\rangle_n|\downarrow\downarrow\rangle_e}$ are very close to each other, the spin part of the magnetic interaction $H_{ac}(t)$ couples only in first order the states $|s\rangle_n$ and $|11\rangle_n$ and thus it does not essentially affect the states $|a\rangle_n$ and $|00\rangle_n$. To complete the CNOT gate one applies in reverse order steps (3), (2), and (1) (see Fig. 56).

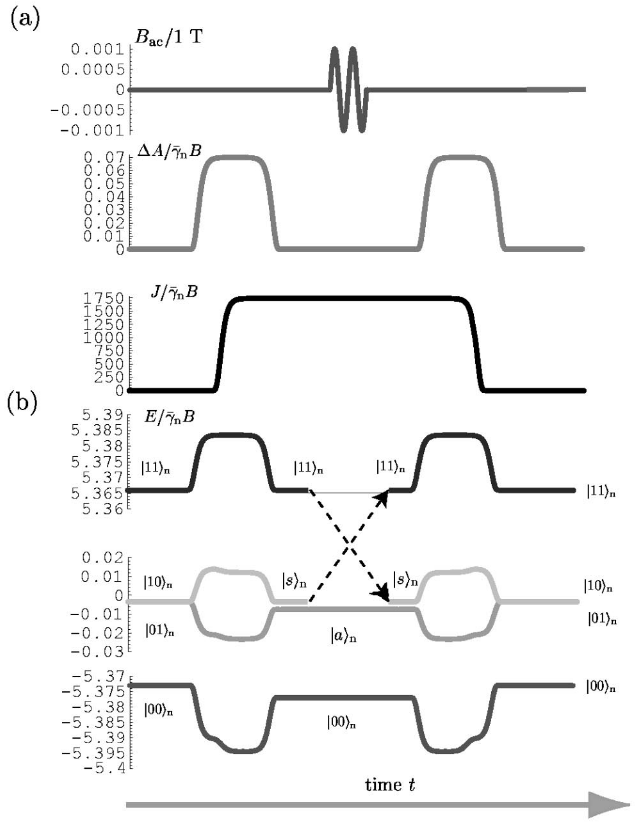


FIG. 56. Implementation of the CNOT gate in a Kane quantum computer as described in steps (1)–(4) in the text (time t runs along the horizontal axis): (a) externally driven couplings are shown; (b) qubit energies are plotted, conveniently shifted by $E \mapsto E - \overline{\gamma}_e B - \tfrac{1}{4}J$.

Other computer operations such as spin measurement and initialization of the quantum register are also based on the adiabatic manipulation of the A and J voltages. The underlying idea has been to correlate nuclear-spin states adiabatically with states of electron spins, which in turn affect the symmetry of the electron orbital wave function (Kane, 2000).

Unlike the quantum computer proposals based on ion traps or NMR spectroscopy, the silicon-based quantum computer has not yet been implemented experimentally.[72] This will require nanofabrication at the

[72]There is a funded project in the Semiconductor Nanofabrication Facility of New South Wales University (Australia) for building a Kane's quantum computer.

atomic scale involving specialized techniques such as quantum electronic measurements with single-electron transistors for addressing individual qubits, atom-scale lithography to place phosphorus donors in a silicon crystal with near-atomic precision, and electron-beam lithography for building the quantum array of qubits. (Kane, 2000). It remains an open issue whether the current technologies will be up to the challenge of building a Kane quantum computer.

XII. CONCLUSIONS

Although this may look like an extensive review, the field has grown at such a pace that it is not possible to cover in detail all the interesting developments going on, and many have been left out. To mention just a few of them: universal sets of fault-tolerant quantum gates, a thorough study of decoherence problems, quantum erasure, and further experimental proposals for quantum computers.

We share a belief in the mutual benefit of the link between quanta and information. The very knowledge of the foundations of physics can benefit from the theory of information and computation (Landauer, 1991, 1996). We have reviewed some aspects of the idea that information is physics. We could further speculate the other way around: physics is also information. It is even conceivable that a fundamental theory of physics could be based on the notion of the qubit, from which all the rest would be derived (Wheeler, 1990; Zeilinger, 1999).

We have made an effort to present both classical and quantum aspects of information and computation. Classical aspects have been traditionally associated with computer science, of interest to computer and electronic engineers and to mathematicians addressing the fundamentals of information theory. Quantum information, by contrast, has so far been almost exclusively of interest to quantum physicists. Each community faces its own barriers in entering the field of quantum computation: an engineer frequently lacks the necessary training in quantum theory, while most physicists are not used to dealing with the insides of a real computer. Our work is aimed in part at setting up a bridge between the two communities. We are confident that in the coming age of quantum information it will become more commonplace for quantum mechanics to be taught at engineering schools and for information theory to figure among background courses in physics. Moreover, as is evident from the proposals we have discussed for quantum computers, other fields of physics are likely to be involved, like condensed matter and its many branches, especially the area of strongly correlated systems.

There is currently widespread interest in building real quantum computers, capable of doing nontrivial tasks. Many proposals have been presented and more are likely. Each physical system or interaction in nature is being scrutinized as a possible realization of a quantum computer. In the past, marvelous machines, like aircraft, were envisaged by Leonardo da Vinci. He described them on paper but they were not actually built until hundreds of years later. We hope that in the case of quantum computers this process will not take that long. In any case, there is no doubt that quantum physics has already influenced in depth the theory of information.

ACKNOWLEDGMENTS

We would like to thank I. Cirac and P. Zoller for their enthusiasm in embracing this project and for pushing us to carry through this long process. We have benefited from discussions and correspondence with I. Cirac, H.-S. Goan, L. Grover, P. Hoyer, B. King, A. K. Lenstra, A. Levitin, H. te Riele, A. Trill, and P. Zoller.

APPENDIX: COMPUTATIONAL COMPLEXITY

There are countless unsolvable problems like the halting problem connected with the Turing machine (Sec. VIII.A). On the other hand, solvable problems can be classified according to their difficulty. Easy ones, like computing the determinant of any $n \times n$ matrix, are referred to as computationally *tractable*, and difficult ones, like computing the permanent of the same matrix,[73] are called computationally hard or *intractable*.

The complexity classes have been devised to group solvable problems according to their degree of difficulty. Three features are addressed (Nielsen and Chuang, 2000): (1) time or space resources required for solution; (2) the machine used for solution (deterministic Turing machine, nondeterministic Turing machine, probabilistic Turing machine, or quantum Turing machine); and (3) the type of problem (decision, number of solutions, optimization, etc.).

1. Classical complexity classes

When the computation is done with deterministic or nondeterministic Turing machines, the relevant classes are the following (Salomaa 1989; Papadimitriou, 1994; Welsh, 1995; Li and Vitányi, 1997; Yan, 2000).[74]

(i) Class **P** (polynomial), containing those problems that can be solved by a deterministic Turing machine in polynomial time, i.e., the time for the machine to find the solution increases at most polynomially with the length n (in bits) of the initial data.

Examples: (1) arithmetic operations such as the addition and multiplication of integers; (2) Euclid's algo-

[73]The definition of the permanent is similar to the determinant. In fact the only difference is the missing sign of the permutations.

[74]Although the complexity classes **P**, **NP**, etc., that we shall consider here usually contain only decision problems [problems whose solution is either YES (1) or NO (0)], we shall implicitly enlarge them by including other computational problems, like searching, which are defined in a similar fashion to decision problems by means of the costs in time or space invested in the solution.

rithm; (3) modular exponentiation; (4) computation of determinants; (5) sorting a list; and (6) multiplication of points on elliptic curves by integers.

(ii) Class **NP** (nondeterministic polynomial), containing those problems that a nondeterministic Turing machine can solve in polynomical time.[75]

As the nondeterministic Turing machines look impractical, it is convenient to know that the **NP** class also can apply when only deterministic Turing machines are involved: a problem is **NP** if, given an arbitrary initial datum x of binary length n, it admits a *succinct certificate* or *polynomial witness* y (i.e., of polynomial length in n), such that there exists a deterministic Turing machine which, given x,y, can solve the problem in polynomial time in n.

Clearly, $\mathbf{P}\subseteq\mathbf{NP}$. A central conjecture in computation theory is $\mathbf{P}\subsetneq\mathbf{NP}$.

Examples: (1) the *discrete logarithm* problem (computation in $\mathbb{Z}_N$ of the solution x to $a^x=b \bmod N$); (2) the *primality* problem (given N, is it prime?); (3) the *compositeness* problem, complement to *primality* (given N, is it composite?); (4) the *factorization* problem (find the decomposition of N into prime factors); (5) the *satisfiability* problem [check whether a given Boolean expression ϕ in conjunctive normal form $\phi=\wedge_1^n C_i$, $C_i \coloneqq z_{i1}\vee z_{i2}\vee\cdots\vee z_{ir_i}$, with $z_{ij}\in(x_{ij},\neg x_{ij})$ Boolean variables or their negations, is satisfiable, that is, there exists a choice of variables that make ϕ true]; and (6) the *traveling salesman* problem (given n cities, their mutual distances $d_{ij}\geqslant 0$, and a cost or "travel budget" C find whether there exists a cyclic permutation π of order n, such that $\Sigma_{i=1}^n d_{i,\pi(i)}\leqslant C$).

Factorization is **NP** since it is apparent that given N, and the succinct certificate consisting of its prime divisors, the decomposition of N into primes is trivial and of polynomial cost.

(iii) Class **PSPACE** (polynomial space) or **NSPACE** (nondeterministic polynomial space), containing those problems that some deterministic (nondeterministic) Turing machine can solve in polynomial space, i.e., using a number of cells that grows at most polynomially with the length (in bits) of the initial data.

It is known that $\mathbf{NP}\subseteq\mathbf{PSPACE}=\mathbf{NSPACE}$.

Examples: (1) In the two-player game Geography, player A chooses the name of a city, say Madrid, and B has to name another city, like Dublin, starting with the last letter D of the previous city; then it is A's turn to name another city starting with N, like New York; B says next Kyoto, and so on. The cities' names must not be repeated. The loser is the player who cannot name another city because there are no more names left. The *Geography* problem is: given an arbitrary set of cities (strings, all different, of alphabet symbols), and A's initial choice of one of them, can A win? It can be shown that Geography is **PSPACE** complete.[76] (2) The game Go suggests a *Go* problem on $n\times n$ boards and the associated question of whether there exists some winning strategy for the starting player. This *Go* problem is likewise **PSPACE** complete.

(iv) Class **EXP** (exponential) or **NEXP** (nondeterministic exponential), containing those problems that some deterministic (nondeterministic) Turing machine can solve in exponential time, i.e., a time that grows at most exponentially with the length (in bits) of the initial data.

Examples: Consider the problems related to the games Go, Checkers, and Chess on $n\times n$ fields: are there always winning strategies for the first player? Since the number of movements to analyze grows exponentially with the board size, such problems are in the class **EXP**. Furthermore, it is believed that they are not in the class **NP**.

The following inclusions among the previous classes hold:

$$\mathbf{P}\subseteq\mathbf{NP}\subseteq\mathbf{PSPACE}\subseteq\mathbf{EXP}\subseteq\mathbf{NEXP}.$$

Moreover, it is also known that $\mathbf{P}\subsetneq\mathbf{EXP}$. Thus at least one of the first three inclusions in the long previous chain must be proper. But it is not known which one.

The classification does not end here. There are even more "monstrous" problems as far as complexity is concerned. For instance, pertaining to the Presburger arithmetic there exists a problem that is at least doubly exponential [time complexity $O(2^{2^n})$ in the size n of the initial data].

Let us now assume that our computers are probabilistic Turing machines. The corresponding classes are called *random*, and some of them stand out.

(i) Class **RP** (randomized polynomial), consisting of those decision problems that a probabilistic Turing machine T, always working in polynomial time (for every initial datum), can decide with error $\leqslant\frac{1}{2}$. These problems are called polynomial Monte Carlo. In other words, if L denotes the set of input data having answer YES, i.e., 1, then

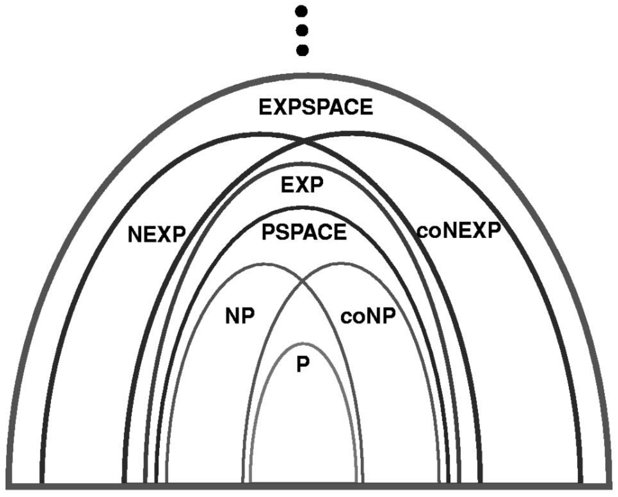
$$x\in L\Rightarrow\text{prob}[T(x)=1]\geqslant\tfrac{1}{2},$$

$$x\notin L\Rightarrow\text{prob}[T(x)=1]=0.$$

This means that all computational pathways that a probabilistic Turing machine T can take from data $x\notin L$ end up with rejection [$T(x)=0$, i.e., NO], while if $x\in L$, then at least a fraction $\frac{1}{2}$ of the possible paths end up with acceptance [$T(x)=1$]. Therefore there cannot be false positives, and at most a fraction $\frac{1}{2}$ of false negatives can happen (that is, cases in which $x\in L$ and the followed path ends with rejection). Repeating the computation with the same $x\in L$ a number of times n

[75]As there may be several computational pathways leading to the solution, the one of shortest duration marks the cost (Salomaa, 1989).

[76]Given a complexity class $\mathbf{X}$, a decision problem $P\in\mathbf{X}$ is called $\mathbf{X}$ complete when any $Q\in\mathbf{X}$ is polynomially reducible to P, i.e., $\exists$ a polynomial-time map $f{:}x\mapsto f(x)$ from the inputs of Q to the inputs of P such that $Q(x)=0,1$ iff $P(f(x))=0,1$.



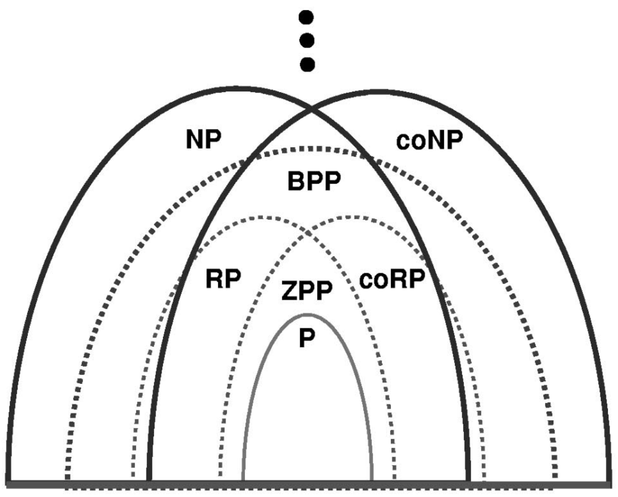


FIG. 57. Different classical complexity classes. On the right, we provisionally accept that the **BPP** class is not a subset of **NP**.

$\gtrsim \lceil \log_2 \delta^{-1} \rceil$, where $0 < \delta < 1$, we will find that the probability of n consecutive false negatives is $\leqslant \delta$ and thus can be made as small as desired by appropriately choosing δ. Equivalently, the probability of obtaining in that series of n trials some acceptance of x will turn out to be $\geqslant (1-\delta)$ and thus can be made as close to 1 as we wish. In cases of real "bad luck" it might happen that very long series would not contain any acceptance of x; that is why it is often said that a probabilistic Turing machine T decides the problem, in the average case, in polynomial time.

(ii) Class **ZPP**$:=$**RP**$\cap$**coRP** (zero-error probabilistic polynomial), where the class **coRP** is the complement of **RP**, that is, it contains those decision problems that answer (YES, NO) to an input if and only if there exists a problem in **RP** that answers (NO, YES) to the same input.

The class **ZPP** thus contains those decision problems for which there exist two probabilistic Turing machines T_{RP} and T_{coRP}, always working in polynomial time and satisfying

$$x \in L \Rightarrow \mathrm{prob}[T_{\mathrm{RP}}(x)=1] \geqslant \tfrac{1}{2}, \mathrm{prob}[T_{\mathrm{coRP}}(x)=0]=0,$$

$$x \notin L \Rightarrow \mathrm{prob}[T_{\mathrm{RP}}(x)=1]=0, \mathrm{prob}[T_{\mathrm{coRP}}(x)=0] \geqslant \tfrac{1}{2}.$$

These are called polynomial Las Vegas problems: they are Monte Carlo, and so are their complements. In other words, they have two Monte Carlo algorithms, one without false positives and another without false negatives. Most likely any input data will be decidable with certainty: it is enough that the algorithm without false positives says YES, or the one without false negatives says NO. In a case of real bad luck, we shall have to repeat both until one of them yields a conclusive answer.

Example: Primality is in **ZPP**. The Miller-Selfridge-Rabin algorithm (pseudoprimality strong test, 1974) is of co Monte Carlo type, that is, primality is in **coRP** (in fact, the probability of false positives, i.e., that one probable prime is composite, is $\leqslant 1/4$). That primality is also in **RP** is a harder issue and was proved by Adleman and Huang (1987), with the theory of Abelian varieties (generalization of elliptic curves to higher dimensions).[77]

(iii) Class **BPP** (bounded-error probabilistic polynomial). This class contains those decision problems for which there exists a probabilistic Turing machine T always working in polynomial time and satisfying

$$x \in L \Rightarrow \mathrm{prob}[T(x)=1] \geqslant \frac{3}{4},$$

$$x \notin L \Rightarrow \mathrm{prob}[T(x)=1] \leqslant \frac{1}{4}.$$

BPP problems are perhaps those best representing the notion of realistic computations. They are accepted or rejected by a probabilistic Turning machine with the possibility of error. But the error probability is $\leqslant \frac{1}{4}$ on both the acceptance and the rejection. Repetition of the algorithm with the same input allows us to amplify the probability of success and, using the majority rule, to decide within polynomial time (average case time, except in bad-luck instances) and with an error as small as required. It is not known whether **BPP**$\subseteq$**NP**, although it is believed that **NP**$\not\subseteq$**BPP**. It is clear that **RP**$\subseteq$**BPP**, and likewise **BPP**$=$**coBPP**. Generically,

$$\mathbf{P} \subseteq \mathbf{ZPP} \subseteq \mathbf{RP} \subseteq (\mathbf{BPP}, \mathbf{NP}) \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP} \subseteq \mathbf{NEXP}.$$

Figure 57 shows the inclusions among the classical complexity classes (Papadimitriou, 1994).

[77]Given an integer N, there exists a deterministic primality-testing algorithm, due to Adleman, Pomerance, and Rumely (1983) and Cohen and Lenstra (1984), with complexity $O[(\log_2 N)^{c \log_2 \log_2 \log_2 N}]$, where c is a constant. A current typical computer takes about 30 s for N with 100 decimal digits, about 8 min if N has 200 digits, and a reasonable time for 1000 digits.

2. Quantum complexity classes

When the computers employed in the computations are quantum Turing machines, the associated complexity classes are called quantum. We list here some of the most relevant:

(i) Class **QP** (quantum polynomial), containing those (decision) problems solvable in polynomial time with a quantum Turing machine.

(ii) Class **BQP** (bounded-error quantum polynomial), containing those problems solvable with error $\leqslant 1/4$ in polynomial time by a quantum Turing machine.

(iii) Class **ZQP** (zero-error probability quantum polynomial), containing problems solvable with zero-error probability in expected polynomial time by a quantum Turing machine.

The following relations among the classical and the quantum complexity classes hold:

$$\mathbf{P} \subsetneq \mathbf{QP}, \quad \mathbf{BPP} \subseteq \mathbf{BQP} \subseteq \mathbf{PSPACE}.$$

The proper inclusion of **P** in **QP**, shown by Berthiaume and Brassard (1992), is remarkable. It means that quantum computers can efficiently solve more problems than their classical kin. This amounts to the first clear victory in the strict separation of classical and quantum complexities.

The second chain of inclusions is due to Bernstein and Vazirani (1993). The crucial question of whether or not $\mathbf{BPP} \subsetneq \mathbf{BQP}$ remains open. That is, are there "tractable" quantum problems that are classically hard? Simon's algorithm (Sec. X.B) is the first positive indication in the presence of a quantum oracle. Further support comes from Shor's algorithm (Sec. X.D), showing that the factorization and discrete logarithm problems are in **BQP**, whereas the current state of the art does not allow us to assert that they are in **BPP**. The inclusion of **BQP** in **PSPACE** implies that it is possible to classically simulate, and with as good an approximation as desired, quantum problems with reasonable memory resources, although the simulation would be exponentially slow in time. That is why there are not solvable problems with quantum Turing machines escaping the domain of deterministic Turing machines. Stated in a different way, quantum computation does not contradict the Church-Turing hypothesis (Sec. VIII.A). Only by invoking efficiency might classical Turing machines yield to quantum Turing machines.

Even though we do not know whether **BPP** is a proper subset of **BQP**, we do know particular cases of classical algorithms (not complexity classes as a whole) that can be speeded up quantumly with respect to their classical running time. Simon's algorithm shows an exponential gain $O(2^n) \rightarrow O(n)$ (Sec. X.B), and Grover's shows a quadratic improvement $O(N) \rightarrow O(N^{1/2})$ (Sec. X.C). But it is not always possible to speed up an algorithm substantially. There are oracle problems that do not admit an essential quantum speed-up; at most it is possible to go from N classical queries down to $N/2$ quantum queries. An example is the *parity* problem [to find the parity of the number of nonzero bits of a string in $\{0,1\}^n$ (Farhi *et al.*, 1998)].

REFERENCES

Adleman, L., C. Pomerance, and R. Rumely, 1983, Ann. Math. **117**, 173.

Adleman, L.M., 1994, Science **266**, 1021.

Aharonov, D., 1998, in *Annual Reviews of Computational Physics*, edited by Dietrich Stauffer (World Scientific, Singapore), Vol. VI.

Aspray, W., 1990, *John von Neumann and the Origins of Modern Computing* (MIT, Cambridge, MA)

Atkins, D., M. Graff, A.K. Lenstra, and P.C. Leyland, 1995, in *Advances in Cryptology—Asiacrypt'94: Proceedings of the 4th International Conference on the Theory and Applications of Cryptology*, Lecture Notes in Computer Science No. 917 (Springer, New York), p. 263.

Averin, D.V., 1998, Solid State Commun. **105**, 659.

Barenco, A., 1995, Proc. R. Soc. London, Ser. A **449**, 679.

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

Barenco, A., A. Berthiaume, D. Deutsch, A. Ekert, R. Jozsa, and Ch. Macchiavello, 1997, SIAM J. Comput. **26**, 1541.

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

Bell, J.S., 1964, Physics (Long Island City, N.Y.) **1**, 195.

Bell, J.S., 1966, Rev. Mod. Phys. **38**, 447.

Bell, J.S., 1987, *Speakable and Unspeakable in Quantum Mechanics* (Cambridge University Press, Cambridge).

Benioff, P.A., 1980, J. Stat. Phys. **22**, 563.

Benioff, P.A., 1981, J. Math. Phys. **22**, 495.

Benioff, P.A., 1982, Phys. Rev. Lett. **48**, 1581.

Bennett, C.H., 1973, IBM J. Res. Dev. **17**, 525.

Bennett, C.H., 1992a, Phys. Rev. Lett. **68**, 3121.

Bennett, C.H., 1992b, Science **257**, 752.

Bennett, C.H., 1995, Phys. Today **48** (10), 24.

Bennett, C.H., 1998, Phys. Scr. **T76**, 210.

Bennett, C.H., E. Berstein, G. Brassard, and U. Vazirani, 1997, SIAM J. Comput. **26**, 1510.

Bennett, C.H., F. Bessette, G. Brassard, L. Savail, and J. Smolin, 1992, J. Cryptology **5**, 3.

Bennett, C.H., and G. Brassard, 1984, in *Proceedings of the International Conference on Computers, Systems & Signal Processing*, Bangalore, India (Indian Institute of Science, Bangalore, India), p. 175.

Bennett, C.H., G. Brassard, C. Crepeau, R. Jozsa, A. Peres, and W.K. Wootters, 1993, Phys. Rev. Lett. **70**, 1895.

Bennett, C.H., G. Brassard, and A. Ekert, 1992, Sci. Am. (Int. Ed.) (10), 50.

Bennett, C.H., G. Brassard, and N.M. Mermin, 1992, Phys. Rev. Lett. **68**, 557.

Bennett, C.H., G. Brassard, S. Popescu, and B. Schumacher, 1996, Phys. Rev. Lett. **76**, 722.

Bennett, C.H., D.P. DiVincenzo, and J. Smolin, 1997, Phys. Rev. Lett. **78**, 3217.

Bennett, C.H., D.P. DiVincenzo, J. Smolin, and W.K. Wootters, 1996, Phys. Rev. A **54**, 3824.

Bennett, C.H., and P.W. Shor, 1998, IEEE Trans. Inf. Theory **44**, 2724.

Bennett, C.H., P.W. Shor, J.A. Smolin, and A.V. Thapliyal, 1999, Phys. Rev. Lett. **83**, 3081.

Bennett, C.H., and S.J. Wiesner, 1992, Phys. Rev. Lett. **69**, 2881.

Bergquist, J.C., R.G. Hulet, W.M. Itano, and D.J. Wineland, 1986, Phys. Rev. Lett. **57**, 1699.

Berman, G.P., D.K. Campbell, G.D. Doolen, G.V. López, and V.I. Tsifrinovich, 1997, Physica B **240**, 61.

Berman, G.P., D.K. Campbell, G.D. Doolen, and K.E. Nagaev, 2000, J. Phys. Condens. Mater. **12**, 2945.

Bernstein, E., and U. Vazirani, 1993, *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing* (Association for Computing Machinery, New York), p. 11.

Berthiaume, A., and G. Brassard, 1992, *Proceedings of the 7th IEEE Conference on Structure in Complexity Theory*, Boston, MA (IEEE Computer Society, Los Alamitos, CA), p. 132.

Berthiaume, A., D. Deutsch, and R. Jozsa, 1994, *Third Workshop on Physics of Computation* (IEEE Computer Society, Los Alamitos, CA), p. 60.

Blake, I., C. Heegard, T. Høholdt, and V. Wei, 1998, IEEE Trans. Inf. Theory **44**, 2596.

Bohm, D., 1951, *Quantum Theory* (Prentice-Hall, Englewood Cliffs, NJ).

Boneh, D., and R.J. Lipton, 1995, in *Advances in Cryptology—CRYPTO'95, Proceedings of the 15th Annual International Cryptology Conference*, edited by D. Coppersmith (Springer, Berlin), p. 424.

Boschi, D., S. Branca, F. De Martini, L. Hardy, and S. Popescu, 1998, Phys. Rev. Lett. **80**, 1121.

Bouwmeester, D., A. Ekert, and A. Zeilinger, 2000, Eds., *The Physics of Quantum Information* (Springer, Berlin).

Bouwmeester, D., J.-W. Pan, M. Daniell, H. Weinfurter, and A. Zeilinger, 1999, Phys. Rev. Lett. **82**, 1345.

Bouwmeester, D., J.-W. Pan, M. Daniell, H. Weinfurter, M. Zukowski, and A. Zeilinger, 1998, Nature (London) **394**, 841.

Bouwmeester, D., J.-W. Pan, K. Mattle, M. Eibl, H. Weinfurter, and A. Zeilinger, 1997, Nature (London) **390**, 575.

Bouwmeester, D., J.-W. Pan, H. Weinfurter, and A. Zeilinger, 2000, J. Mod. Opt. **47**, 279.

Boyer, M., G. Brassard, P. Hoyer, and A. Tapp, 1998, Fortschr. Phys. **46**, 493.

Brady, A.H., 1983, Math. Comput. **40**, 647.

Brassard, G., 1989, SIGACT News **20** (4), 78.

Brassard, G., and P. Bratley, 1996, *Fundamentals of Algorithmics* (Prentice-Hall, Englewood Cliffs, NJ).

Brassard, G., C. Crépeau, D. Mayers, and L. Salvail, 1997, e-print quant-ph/9712023.

Brassard, G., P. Hoyer, and A. Tapp, 1998, in *Proceedings of the 25th International Colloquium on Automata, Language, and Programming: ICALP*, Lecture Notes in Computer Science No. 80 (Springer, Berlin), Vol. 1443, p. 820.

Braunstein, S.L., C.M. Caves, R. Jozsa, N. Linden, S. Popescu, and R. Schack, 1999, Phys. Rev. Lett. **83**, 1054.

Braunstein, S.L., C.A. Fuchs, and H.J. Kimble, 2000, J. Mod. Opt. **47**, 267.

Braunstein, S.L., and H.J. Kimble, 1998, Nature (London) **394**, 840.

Brylinski, J.-L., and R. Brylinski, 2002, *Mathematics of Quantum Computation* (CRC Press, Boca Raton, FL, in press).

Burkard, G., H.-A. Engel, and D. Loss, 2000, Fortschr. Phys. **48**, 9–11, 965. This is a special issue on "Experimental Proposals for Quantum Computation" edited by S.L. Braunshein and H.K. Lo.

Buzek, V., and M. Hillery, 1996, Phys. Rev. A **54**, 1844.

Calderbank, A.R., and P.W. Shor, 1996, Phys. Rev. A **54**, 1098.

Chuang, I.L., 2000, Phys. Rev. Lett. **85**, 2006.

Chuang, I.L., N. Gershenfeld, and M. Kubinec, 1998, Phys. Rev. Lett. **80**, 3408.

Chuang, I.L., N. Gershenfeld, M. Kubinec, and D. Leung, 1998, Proc. R. Soc. London, Ser. A **454**, 447.

Church, A., 1936, Am. J. Math. **58**, 345.

Cirac, J.I., and P. Zoller, 1995, Phys. Rev. Lett. **74**, 4091.

Cirac, J.I., and P. Zoller, 2000, Nature (London) **404**, 579.

Cleve, R., 2000, in *Collected Papers on Quantum Computation and Quantum Information Theory*, edited by C. Macchiavello, G. M. Palma, and A. Zeilinger (World Scientific, Singapore), pp. 103–127.

Cleve, R., A. Ekert, C. Macchiavello, and M. Mosca, 1998, Proc. R. Soc. London, Ser. A **454**, 339.

Cohen, H., 1993, *A Course in Computational Algebraic Number Theory*, Graduate Texts in Mathematics No. 138 (Springer-Verlag, Berlin).

Cohen, H., and H.W. Lenstra, 1984, Math. Comput. **42**, 297.

Collins, D., K.W. Kim, W.C. Holton, H. Sierzputowska-Gracz, and E.O. Stejskal, 1999, e-print quant-ph/9910006.

Collins, G.P., 1992, Phys. Today **45** (11), 21.

Conway, J.H., and N.J.A. Sloane, 1999, *Sphere Packings, Lattices and Groups*, 3rd ed., Grundlehren der Mathematischen Wissenschaften No. 290 (Springer-Verlag, New York).

Coppersmith, D., 1994, IBM Res. Rep. RC 19642.

Cory, D.G., A.F. Fahmy, and T.F. Havel, 1996, in *PhysComp96: Proceedings of the 4th Workshop on Physics and Computation*, Boston, MA, 1996, edited by T. Toffoli, M. Biafore, and J. Leao (New England Complex Systems Institute, Cambridge, MA), p. 87.

Cory, D.G., D.G. Fhamy, and T.F. Havel, 1997, Proc. Natl. Acad. Sci. U.S.A. **94**, 1634.

Cory, D.G., R. Laflamme, E. Knill, L. Viola, T.F. Havel, N. Boulant, G. Boutis, E. Fortunato, S. Lloyd, R. Martinez, C. Negrevergne, M. Pravia, Y. Sharf, G. Teklemariam, Y.S. Weinstein, and W.H. Zurek, 2000, Fortschr. Phys. **48**, 875.

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

Deutsch, D., 1989, Proc. R. Soc. London, Ser. A **425**, 73.

Deutsch, D., 1993, talk given at the Rank Prize Funds Symposium on Quantum Communication and Cryptography, Broadway, UK.

Deutsch, D., and R. Jozsa, 1992, Proc. R. Soc. London, Ser. A **439**, 553.

Diffie, W., 1992, in *Contemporary Cryptology: The Science of Information Integrity*, edited by G. Simmons (IEEE, Piscataway, NJ), p. 135.

Diffie, W., and M.E. Hellman, 1976, IEEE Trans. Inf. Theory **22**, 644.

DiVincenzo, D., 1995, Phys. Rev. A **51**, 1015.

Dür, W., H.-J. Briegel, J.I. Cirac, and P. Zoller, 1999, Phys. Rev. A **59**, 169.

Dür, W., J.I. Cirac, and R. Tarrach, 1999, Phys. Rev. Lett. **83**, 3562.

Durr, Ch., and P. Hoyer, 1996, e-print quant-ph/9607014.

Einstein, A., B. Podolsky, and N. Rosen, 1935, Phys. Rev. **47**, 777.

Eisert, J., M. Wilkens, and M. Lewenstein, 1999, Phys. Rev. Lett. **83**, 3077.

Ekert, A., 1991, Phys. Rev. Lett. **67**, 661.

Ekert, A., P. Hayden, and H. Inamori, 2000, e-print quant-ph/0011013.

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

Ekert, A., and P. Knight, 1995, Am. J. Phys. **63**, 415.

Ekert, A., and C. Macchiavello, 1996, Phys. Rev. Lett. **77**, 2585.

Electronic Frontier Foundation, 1998, *Cracking DES. Secrets of Encryption Research, Wiretap Politics & Chip Design. How federal agencies subvert privacy*, foreword by W. Diffie (EFF, San Francisco, CA).

Ellis, J.H., 1970, CESG (Communications-Electronics Security Group) Report, January.

Ernst, R.R., G. Bodenhausen, and A. Wokaum, 1987, *Principles of Nuclear Magnetic Resonance in One and Two Dimensions* (Oxford University Press, New York).

Fang, X., X. Zhu, M. Feng, X. Mao, and F. Du, 2000, Phys. Rev. A **61**, 022307.

Farhi, E., J. Goldstone, S. Gutmann, and M. Sipser, 1998, Phys. Rev. Lett. **81**, 5442.

Feynman, R.P., 1982, Int. J. Theor. Phys. **21**, 467.

Feynman, R.P., 1985, Optics News **11**, 11.

Feynman, R.P., 1996, in *Feynman Lectures on Computation*, edited by A. Hey and R. Allen (Addison-Wesley, Reading, MA).

Flynn, M.J., 1966, Proc. IEEE **54**, 1901.

Flynn, M.J., 1972, IEEE Trans. Comput. **21**, 948.

Fredkin, E., and T. Toffoli, 1982, Int. J. Theor. Phys. **21**, 219.

Freeman, R., 1998, *Spin Choreography* (Oxford University Press, New York).

Fulde, P., 1995, *Electron Correlations in Molecules and Solids*, Springer Series in Solid-State Sciences No. 100 (Springer-Verlag, Berlin).

Furuzawa, A., J.L. Sørensen, S.L. Braunstein, C.A. Fuchs, H.J. Kimble, and E.S. Polzik, 1998, Science **282**, 706.

Galindo, A., and M.A. Martin-Delgado, 2000, Phys. Rev. A **62**, 062303.

Galindo, A., and P. Pascual, 1989, *Problemas de Mecánica Cuántica* (EUDEMA, Madrid).

Galindo, A., and P. Pascual, 1990a, *Quantum Mechanics I* (Springer-Verlag, Berlin).

Galindo, A., and P. Pascual, 1990b, *Quantum Mechanics II* (Springer-Verlag, Berlin).

Gardner, M., 1977, Sci. Am. **237** (8), 120.

Gauss, K. F., 1801, *Disquisitiones Arithmeticae* (G. Fleischer, Leipzig), English translation by A.A. Clark (Yale University, New Haven, 1966), revised English translation by W.C. Waterhouse (Springer-Verlag, New York, 1975).

Gerber, J., 1983, Math. Comput. **41**, 287.

Gershenfeld, N.A., and I.L. Chuang, 1997, Science **275**, 350.

Gershenfeld, N.A., I.L. Chuang, and S. Lloyd, 1996, in *PhysComp96: Proceedings of the Fourth Workshop on Physics and Computation*, edited by T. Toffoli, M. Biafore, and J. Leão (New England Complex Systems Institute, Cambridge, MA), p. 136.

Giedke, G., B. Kraus, M. Lewenstein, and J.I. Cirac, 2001, e-print quant-ph/0104050.

Gisin, N., 1996, Phys. Lett. A **210**, 151.

Gisin, N., G. Ribordy, W. Tittel, and H. Zbinden, 2001, Rev. Mod. Phys. **74**, 145.

Goan, H.-S., and G.J. Milburn, 2000, unpblished.

Greenberger, D.M., M. Horne, and A. Zeilinger, 1989, in *Bell's Theorem, Quantum Theory and Conceptions of the Universe*, edited by M. Kafatos (Kluwer, Dordrecht).

Grover, L.K., 1996, *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing*, Philadelphia, PA (Association for Computing Machinery, New York), p. 212.

Grover, L.K., 1997, Phys. Rev. Lett. **79**, 325.

Händler, W., 1982, in *Parallel Processing Systems—An Advanced Course*, edited by David J. Evans (Cambridge University Press, Cambridge, England), p. 1.

Hellman, M.E., 1979, Sci. Am. **241** (8), 146.

Herken, R., 1995, Ed., *The Universal Turing Machine: A Half-Century Survey* (Springer-Verlag, Vienna).

Herring, C., and M. Flicker, 1964, Phys. Rev. **134**, 362.

Hillis, W.D., 1998, in *Feynman and Computation*, edited by Anthony J.G. Hey (Addison-Wesley, Reading, MA), p. 257.

Hodges, A., 1992, *Alan Turing: The Enigma* (Vintage Random House, London).

Hogg, T., 1998, Physica D **120**, 102.

Holevo, A.S., 1973, Probl. Peredachi Inf. **9**, 3 [Probl. Inf. Transm. **9**, 177].

Horodecki, R., P. Horodecki, and M. Horodecki, 1996a, Phys. Lett. A **210**, 377.

Horodecki, R., P. Horodecki, and M. Horodecki, 1996b, Phys. Lett. A **223**, 1.

Horodecki, R., P. Horodecki, and M. Horodecki, 1998, Phys. Rev. Lett. **80**, 5239.

Horodecki, R., P. Horodecki, and M. Horodecki, 1999, Phys. Rev. Lett. **82**, 1056.

Hughes, R.J., 1998, Philos. Trans. R. Soc. London **356**, 1713.

Hughes, R.J., D.M. Alde, P. Dyer, G.G. Luther, G.L. Morgan, and M. Schauer, 1995, Contemp. Phys. **36**, 149.

Hughes, R.J., W.T. Buttler, P.G. Kwiat, S.K. Lamoreaux, G.L. Morgan, J.E. Nordholt, and C.G. Peterson, 1999, e-print quant-ph/9905009.

Hughes, R.J., D.F.V. James, J.J. Gomez, M.S. Gulley, M.H. Holzscheiter, P.G. Kwiat, S.K. Lamoreaux, C.G. Peterson, V.D. Sandberg, M.M. Schauer, C.M. Simmons, C.E. Thorburn, D. Tupa, P.Z. Wang, and A.G. White, 1998, Fortschr. Phys. **46**, 329.

Hughes, R.J., D.F.V. James, E.H. Knill, R. Laflamme, and A.G. Petschek, 1996, Phys. Rev. Lett. **77**, 3240.

Hughes, R.J., G. Luther, G. Morgan, G. Peterson, and C. Simmons, 1996, Lect. Notes Comput. Sci. **1109**, 329.

Hughes, R.J., G.L. Morgan, and C.G. Peterson, 2000, J. Mod. Opt. **47**, 533.

Hughes, R.J., and J.E. Nordholt, 1999, Phys. World **12** (15), 31.

Hughston, L.P., R. Jozsa, and W.K. Wootters, 1993, Phys. Lett. A **183**, 14.

Hwang, K., and F.A. Briggs, 1985, *Computer Architecture and Parallel Processing* (McGraw-Hill International, New York).

Jennewein, T., C. Simon, G. Weihs, H. Weinfurter, and A. Zeilinger, 2000, Phys. Rev. Lett. **84**, 4729.

Jones, J.A., 2001, Prog. Nucl. Magn. Reson. Spectrosc. **38**, 325.

Jones, J.A., R.H. Hansen, and M. Mosca, 1998, J. Magn. Reson. **135**, 353.

Jones, J.A., and M. Mosca, 1998, J. Chem. Phys. **109**, 1648.

Jones, J.A., M. Mosca, and R.H. Hansen, 1998, Nature (London) **392**, 344.

Jones, J.A., V. Vedral, A. Ekert, and G. Castagnoli, 2000, Nature (London) **403**, 869.

Jozsa, R., 1994, J. Mod. Opt. **41**, 2315.

Jozsa, R., 1998, Proc. R. Soc. London Ser. A **454**, 323.

Jozsa, R., 1999, e-print quant-ph/9901021.

Jozsa, R., D.S. Abrams, J.P. Dowling, and C.P. Williams, 2000, Phys. Rev. Lett. **85**, 2010.

Jozsa, R., and B. Schumacher, 1994, J. Mod. Opt. **41**, 2343.

Kahn, D., 1967, *The Codebreakers, the Story of Secret Writing* (MacMillan, New York).

Kane, B.E., 1998, Nature (London) **393**, 133.

Kane, B.E., 2000, Fortschr. Phys. **48**, 1023.

Kitaev, A.Yu., 1995, L. D. Landau Institute for Theoretical Physics, Moscow report; e-print quant-ph/9511026.

Kitaev, A.Y., 1997, Russ. Math. Surveys **52**, 1191.

Knill, E., 1995, e-print quant-ph/9508006.

Knill, E., I. Chuang, and R. Laflamme, 1998, Phys. Rev. A **57**, 3348.

Knill, E., R. Laflamme, R. Martinez, and C.-H. Tseng, 2000, Nature (London) **404**, 368.

Knuth, D.E., 1975, *The Art of Computer Programming, Vol. 3: Sorting and Searching* (Addison-Wesley, Reading, MA).

Knuth, D.E., 1981, *The Art of Computer Programming 2: Seminumerical Algorithms*, 2nd ed. (Addison-Wesley, Reading, MA).

Koblitz, N., 1994, *A Course in Number Theory and Cryptography*, 2nd ed. (Springer-Verlag, Berlin).

Kwiat, P., S. Barraza-López, A. Stefanov, and N. Gisin, 2001, Nature (London) **409**, 1014.

Kwiat, P., K. Mattle, H. Weinfurter, A. Zeilinger, A.V. Sergienko, and Y. Shih, 1995, Phys. Rev. Lett. **75**, 4337.

Landauer, R., 1961, IBM J. Res. Dev. **5**, 183.

Landauer, R., 1991, Phys. Today **44** (5), 23.

Landauer, R., 1996, Phys. Lett. A **217**, 188.

Landauer, R., 1997, in *Proceedings of the Drexel-4 Symposium on Quantum Nonintegrability-Quantum-Classical Correspondence*, Philadelphia, PA, 1994, edited by D.H. Feng and B.-L. Hu (International Press, Boston).

Lecerf, Y., 1963, C. R. Acad. Fran. Sci. **257**, 2597.

Lenstra, H.W., 1987, Ann. Math. **126**, 649.

Lenstra, A., and H.W. Lenstra, 1993, Eds., *The Development of the Number Field Sieve*, Lecture Notes in Math. No. 1554 (Springer-Verlag, Berlin).

Levitin, L.B., 1969, *Proceedings of the 4th All-Union Conf. Information and Coding Theory*, Tashkent, 1969, p. 111 (in Russian).

Levitin, A., 1999, in *Proceedings of SIGCSE'99*, pp. 179–183.

Lewenstein, M., D. Bruss, J.I. Cirac, B. Kraus, M. Kus, J. Samsonowicz, A. Sanpera, and R. Tarrach, 2000, J. Mod. Opt. **47**, 2841.

Li, M., and P. Vitányi, 1997, *An Introduction to Kolmogorov Complexity and Its Applications*, 2nd ed. (Springer-Verlag, New York).

Lin, S., and T. Rado, 1965, J. ACM **12** (2), 196.

Lloyd, S., 1995, Phys. Rev. Lett. **75**, 346.

Lo, H-K., and H. F. Chau, 1999, Science **283**, 2050.

Loss, D., and D.P. DiVincenzo, 1998, Phys. Rev. A **57**, 120.

Macwilliams, F.J., and N.J.A. Sloane, 1977, *The Theory of Error-Correcting Codes* (North-Holland, Amsterdam).

Makhlin, Y., G. Schön, and A. Shnirman, 2001, Rev. Mod. Phys. **73**, 357.

Manin, Yu., 1980, *Vychislimoe i Nevychislimoe [Computable and Uncomputable]* (Sovetskoye Radio, Moscow).

Marand, Ch., and P.D. Townsend, 1995, Opt. Lett. **20**, 1695.

Marxen, H., 1997, Usenet newsgroup comp.theory, October 5.

Marxen, H., and J. Buntrock, 1990, Bull. Eur. Assoc. Theor. Comput. Sci. **40**, 247.

Mattle, K., H. Weinfurter, P.G. Kwiat, and A. Zeilinger, 1996, Phys. Rev. Lett. **76**, 4656.

Mayers, D., 1996, in *PhysComp96: Proceedings of the Fourth Workshop on Physics and Computation*, Boston, edited by T. Toffoli, M. Biafore, and J. Leão (New England Complex Systems Institute, Cambridge, MA), p. 226.

Mayers, D., 1997, Phys. Rev. Lett. **78**, 3414.

Mayers, D., 1998, e-print quant-ph/9802025.

Meyer, D.A., 1999, Phys. Rev. Lett. **82**, 1052.

Miller, G.L., 1976, J. Comput. Syst. Sci. **13**, 300.

Minsky, M., 1967, *Computation: Finite and Infinite Machines* (Prentice-Hall, Englewood Cliffs, NJ).

Molmer, K., and A. Sorensen, 1999, Phys. Rev. Lett. **82**, 1835.

Monroe, C., D.M. Meekhof, B.E. King, W.M. Itano, and D.J. Wineland, 1995, Phys. Rev. Lett. **75**, 4714.

Moore, G., 1965, unpublished.

Mosca, M., and A. Ekert, 1999, in *Quantum Computing and Quantum Communications*, edited by C.P. Williams (Springer, Berlin), p. 174.

Muller, A., J. Breguet, and N. Gisin, 1993, Europhys. Lett. **23**, 383.

Muller, A., H. Zbinden, and N. Gisin, 1996, Europhys. Lett. **33**, 335.

Nagourney, W., J. Sandberg, and H. Dehmelt, 1986, Phys. Rev. Lett. **56**, 2797.

Nielsen, M.A., 1999, Phys. Rev. Lett. **83**, 436.

Nielsen, M.A., and I.L. Chuang, 2000, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge).

Nielsen, M.A., E. Knill, and R. Laflamme, 1998, Nature (London) **396**, 52.

Palma, G.M., K.A. Suominen, and A.K. Ekert, 1996, Proc. R. Soc. London, Ser. A **452**, 567.

Pan, J-W., D. Bouwmeester, H. Weinfurter, and A. Zeilinger, 1998, Phys. Rev. Lett. **80**, 3891.

Papadimitriou, Ch. H., 1994, *Computational Complexity* (Addison-Wesley, Reading, MA).

Peres, A., 1996, Phys. Rev. Lett. **77**, 1413.

Pippengger, N., and M. Fisher, 1979, J. ACM **26**, 361.

Planck, M., 1900, Verh. Dtsch. Phys. Ges. **2**, 237.

Pomerance, C., 1982, in *Computational Methods in Number Theory*, edited by H.W. Lenstra, Jr., and R. Tidjeman (Mathematisch Centrum, Amsterdam), p. 89.

Pomerance, C., 1996, Not. Am. Math. Soc. **43**, 1473.

Preskill, J., 1997, Talk, 15 January 1997, www.theory.caltech.edu/~preskill

Preskill, J., 1998, lecture notes, www.theory.caltech.edu/~preskill/ph229

Press, W.H., S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, 1992, *Numerical Recipes in C*, 2nd ed. (Cambridge University Press, Cambridge).

Price, M-D., S.S. Somaroo, C.-H. Tseng, J.C. Gore, A.F. Fahmy, T.F. Havel, and D.G. Cory, 1999, J. Magn. Reson., Ser. A **140**, 371.

Rabi, I.I., 1937, Phys. Rev. **51**, 652.

Rabin, M.O., 1980, J. Number Theory **12**, 128.

Rado, T., 1962, Bell Syst. Tech. J. **61**, 877.

Reck, M., A. Zeilinger, H.J. Bernstein, and P. Bertani, 1994, Phys. Rev. Lett. **73**, 58.

Rieffel, E., and W. Polack, 2000, ACM Comput. Surv. **32**, 300.

Rivest, R.L., A. Shamir, and L. Adleman, 1978, Commun. ACM **21**, 120.

Roman, S., 1992, *Coding and Information Theory* (Springer-Verlag, New York).

Rozenberg, G., and A. Salomaa, 1994, *Cornerstones of Undecidability* (Prentice Hall, Englewood Cliffs, NJ).

Rungta, P., W.J. Munro, K. Nemoto, P. Deuar, G.J. Milburn, and C.M. Caves, 2001, in *Directions in Quantum Optics: A Collection of Papers Dedicated to the Memory of Dan Walls*, edited by H.J. Carmichael, R.J. Glauber, and M.O. Scully (Springer, Berlin), pp. 149–164.

Sachdev, S., 1999, *Quantum Phase Transitions* (Cambridge University Press, New York).

Sackett, C.A., D. Kielpinski, B.E. King, C. Langer, V. Meyer, C.J. Myatt, M. Rowe, Q.A. Turchette, W.M. Itano, D.J. Wineland, and C. Monroe, 2000, Nature (London) **404**, 256.

Salomaa, A., 1989, *Computation and Automata*, Encyclopedia of Mathematics and Its Applications No. 25 (Cambridge University Press, Cambridge).

Salomaa, A., 1996, *Public-Key Cryptography*, 2nd enlarged ed. (Springer-Verlag, Berlin).

Sauter, Th., W. Neuhauser, R. Blatt, and P.E. Toschek, 1986, Phys. Rev. Lett. **57**, 1696.

Savage, J., 1972, J. ACM **19**, 660.

Schmidt, E., 1906, Math. Ann. **63**, 433.

Schnorr, C., 1976, Acta Inf. **7**, 95.

Schumacher, B., 1995, Phys. Rev. A **51**, 2738.

Shallit, J., 1998, University of Waterloo report.

Shannon, C.E., 1948, Bell Syst. Tech. J. **27**, 379; **27**, 623 (1948).

Shannon, C.E., 1949, Bell Syst. Tech. J. **28**, 656.

Shor, P.W., 1994, *Proceedings of the 35th Annual Symposium on the Foundations of Computer Science* (IEEE Computer Society Press, Los Alamitos, CA, 1994), p. 124; e-print quant-ph/l9508027.

Shor, P.W., 1995, Phys. Rev. A **52**, R2493.

Shor, P., 2000, e-print quant-ph/0005003.

Shor, P.W., and J.A. Sloane, 1998, J. Algebr. Combinatorics **7**, 157.

Simon, D.R., 1994, *Proceedings of the 35th Annual IEEE Symposium on the Foundations of Computer Science* (IEEE Computer Society, Los Alamitos, CA), extended abstract p. 116. Full version, in SIAM J. Comput. **26**, 1474 (1997).

Slichter, C.P., 1990, *Principles of Magnetic Resonance* (Springer-Verlag, New York).

Solovay, R., 1995, unpublished.

Steane, A.M., 1996a, Phys. Rev. Lett. **77**, 793.

Steane, A.M., 1996b, Proc. R. Soc. London, Ser. A **452**, 2551.

Steane, A.M., 1998, Rep. Prog. Phys. **61**, 117.

Stichtenoth, H., 1993, *Algebraic Function Fields and Codes* (Springer Verlag, Heidelberg).

Stinson, D.R., 1995, *Cryptography: Theory and Practice* (CRC Press, Boca Raton, FL).

Thirring, W., 1983, *A Course in Mathematical Physics 4: Quantum Mechanics of Large Systems* (Springer-Verlag, New York).

Toffoli, T., 1981, Math. Syst. Theory **14**, 13.

Turing, A., 1936, Proc. London Math. Soc. **42**, 230; **43**, 544(E) (1936).

Turing, A., 1946, unpublished, reprinted in *A.M. Turing's ACE Report of 1946 and Other Papers*, edited by B.E. Carpenter and R.W. Doran (MIT, Los Angeles, CA, 1986).

Turing, A.M., 1948, "Intelligent Machinery." National Physical Laboratory Report, reprinted 1969 in *Machine Intelligence 5*, edited by B. Meltzer and D. Michie (Edinburgh University Press, Edinburgh).

Turing, A., 1950, Mind **49**, 433.

Unruh, W.G., 1995, Phys. Rev. A **51**, 992.

Vaidman, L., 1998, *Talk at the Conference: Mysteries, Puzzles and Paradoxes in Quantum Mechanics*, Gargano, Italy; e-print quant-ph/9810089.

van der Lubbe, J.C.A., 1998, *Basic Methods of Cryptography* (Cambridge University Press, Cambridge).

Vandersypen, L.M.K., C.S. Yannoni, M.H. Sherwood, and I.L. Chuang, 1999, Phys. Rev. Lett. **83**, 3085.

Vedral, V., A. Barenco, and A. Ekert, 1996, Phys. Rev. A **54**, 147.

Vedral, V., and M.B. Plenio, 1998, Phys. Rev. A **57**, 1619.

Vernam, G.S., 1926, J. Am. Inst. Electr. Eng. **45**, 109.

Vidal, G., 1999, Phys. Rev. Lett. **83**, 1046.

von Neumann, J., 1945, *First Draft of a Report on the EDVAC (June 1945)*, reprinted with corrections in Ann. Hist. Comput. **15**, 25 (1993).

von Neumann, J., 1946, *The Principles of Large-Scale Computing Machines*, reprinted in the Ann. Hist. Comput. **3**, 263 (1981).

Weinstein, Y.S., S. Lloyd, and D.G. Cory, 1999, e-print quant-ph/9906059.

Welsh, D., 1995, *Codes and Cryptography* (Oxford University Press, Oxford).

Werner, R. F., 1989, Phys. Rev. A **40**, 4277.

Wheeler, J.A., 1990, in *Complexity, Entropy and the Physics of Information*, edited by W.H. Zurek (Addison-Wesley, Redwood City, CA).

White, S.R., 1992, Phys. Rev. Lett. **69**, 2863.

White, S.R., 1993, Phys. Rev. B **48**, 10345.

Wiesner, S., 1983, SIGACT News **15** (1), 78.

Wineland, D.J., C. Monroe, W.M. Itano, D. Leibfried, B.E. King, and D.M. Meekhof, 1998, J. Res. Natl. Inst. Stand. Technol. **3**, 259.

Wootters, W.K., and W.H. Zurek, 1982, Nature (London) **299**, 802.

Yan, S.Y., 2000, *Number Theory for Computing* (Springer-Verlag, Berlin).

Yao, A., 1993, *Proceedings of the 34th IEEE Symposium on the Foundations of Computer Science* (IEEE Computer Society, Los Alamitos, CA), p. 352.

Zalka, Ch., 1999, Phys. Rev. A **60**, 2746.

Zeilinger, A., 1999, Found. Phys. **29**, 631.