Computing with quanta—impacts of quantum theory on computation

Mika Hirvensalo

TUCS—Turku Centre for Computer Science, Department of Mathematics, University of Turku, FIN-20014, Turku, Finland

Abstract

This is a survey article about quantum computing. We begin with a brief introduction on theories of computing and represent the Hilbert space formalism of quantum physics. We study some devices for quantum computing, and finally mention some important achievements and restrictions of quantum computing.

Keywords: Quantum computing; Computability; Hilbert space; Linear operators

1. Introduction

There are many ways to look at the theory of computing. Over 60 years the well-known work of Alan Turing has been the basis onto which the mathematical theory of computing has been (mainly) established, and there is a good reason for this: Turing abstracted the idea of a computational device into a mathematical form nowadays known as a Turing machine. In fact, Turing's work was a huge step when standardizing the notion of a computational device in mathematical terms.

However, it has been pointed out by Benioff [3] and Feynman [10], that any realization of computation is eventually a physical process and, therefore it is worth studying the opportunities as well as the restrictions that the physics gives for computations.

An analysis of the computation models shows that the traditional models of computing are well explainable by devices that can be described by using classical physics, whereas computational machines built on the principles of quantum physics have not been deeply examined until Feynman's article [10]. In his article, Feynman was the first to propose that to simulate quantum physical phenomena efficiently (that is, with...
a polynomial slowdown in the simulation) by using a traditional computer is an impossible task. On the other hand, Feynman also argued that by using a quantum computer running according to the laws of quantum physics, an efficient simulation would be possible. Therefore, Feynman was indeed the first one to propose, that a quantum computer could work much more efficiently than any classical one.

The theory exploring the possibilities and the restrictions given by quantum physics for computations seems to be divided at least in two parts: the theory of quantum computation and quantum information theory. As it is usual for many mathematical theories, also the above division is rough: both theories emerge from the assumption that the physical systems representing the basic elements of a computation or information processing are described by using quantum physics.

A great part of the research articles on quantum computation are published in journals directed to researchers familiar with quantum physics. Many people agree with me when I say that for readers oriented to mathematics or computer science, those articles seem quite cryptic, mainly because the terminology and the notational systems used in physical sciences differ so much from notations conventionally used in mathematics and computer science.

In this article, we will represent, in an introductory way, the very basic notions of quantum physics needed to understand quantum computation and quantum information theory. The very purpose of this article is to serve a reader who is already familiar with computer science and wants to learn about quantum computing. We, however, assume here that the reader oriented to computer science is also aware of the basic notions of linear algebra: vector spaces, inner products and matrices.

2. Basics on computation

Let $A$ be a finite set, referred as to an alphabet hereafter. We write $|A|$ to mean the cardinality of $A$, and notation $A^*$ stands for the free monoid generated by $A$. Hence an element $w \in A^*$ can be uniquely represented as $w = a_1a_2\ldots a_k$, where each $a_i \in A$. An element $w \in A^*$ is called a word or a string over alphabet $A$. A subset $L \subseteq A^*$ is called a language over $A$.

A (discrete) computational device $M$ has usually been viewed as a facility for computing a function

$$f_M : A^* \rightarrow B^*$$

(1)

where $A$ and $B$ are both finite alphabets, referred as to input alphabet and output alphabet, respectively. In this article, we will not change this premise, but we will only consider computational devices capable for representing functions (1). Using a suitable encoding, one can even assume that $A = B = \{0, 1\}$ is the binary alphabet. When studying the representations of the alphabets, we will also learn how the above notion suits to probabilistic algorithms.

In connection with the practical means, it has also been assumed that the computational device $M$ has a finite description. In other words, it has usually been assumed that $M$ has finitely many internal states and finitely many transition rules that instruct
the machine how to proceed on. In this article, we are not going to change these assumptions. However, we may study devices analogous to Boolean circuits, which, however, can be efficiently simulated by finite-state machines.

Anyone familiar with the classical models of computation such as Turing machines, Random Access Machines, etc., knows that quite a usual assumption is that the act of computation, i.e., the act of applying the transition rules, takes place at discrete times steps. In this article, we are not going to change this assumption, either. We will not study devices associated to analogue computation, but only devices similar to finite-state machines. Therefore, we will not pay much more attention on examining the deep meaning of word “computation”.

On the other hand, we will closely examine the nature of representing the elements of $A^*$ as “inputs”, as well as “computational rules”, and the nature of “output”. In fact, studying the representations of these concepts will serve as a starting point on the way to quantum computation, as well as to quantum information processing.

3. Representing the alphabet—classical physics

It is not hard to agree with famous physicists Paul Benioff and Richard Feynman, who have stated that any realization of computation is ultimately a physical process, see [3,10], respectively. Since, in this article, we study computational devices realizing functions $f: A^* \rightarrow B^*$, it is worth studying the physical systems which are used to represent the elements of $A$ and $B$.

For a physical system, one can associate the notion of a state. Here we will not enter into details, but merely outline the basic features. For more details, see [18,16] or [8], for instance. In this section, we study the alphabet representations according to the classical physics, and the following section is devoted to the study of representing the alphabets, having quantum physics in mind.

3.1. State set

A physical system $C_A$ which is capable to represent a finite alphabet $A = \{a_1, \ldots, a_n\}$ reliably must have $n$ basis states, which are denoted by $[a_1], \ldots, [a_n]$ hereafter. The notion “system is in state $[a_i]$” intuitively means that the system is currently representing letter $a_i$: when the system is in state $[a_i]$, then observing the system will yield outcome $a_i$ with a probability of 1.

We can also introduce the notion of a mixed state: a mixed state $[P]$ of the system $C_A$ can be uniquely represented as

$$[P] = p_1[a_1] + \cdots + p_n[a_n],$$

where $p_i \geq 0$ and $p_1 + \cdots + p_n = 1$. The intuitive meaning of a mixed state (2) is the following: observation of the system $C_A$ in state (2) will yield $a_i$ as outcome with a probability of $p_i$. State (2) can be interpreted to reflect our ignorance: one may as well say that the system really is in one of the basis states, in $[a_i]$ with a probability of $p_i$. This interpretation for state (2) is called ignorance interpretation.
It is worth noticing that we required the basis states \( [a_i] \) to be reliably distinguishable: by performing an observation, one should be able to tell without any error possibility, the state of the system. When introducing quantum mechanics, we are not going to change this requirement. Instead, we are going to rewrite the mathematical structure behind the state set.

For the further study, it is worth listing some properties of the state set, as understood in classical physics.

Above we mentioned that when we are representing an alphabet \( A \), we use a physical system \( CA \) having \( n = |A| \) basis states \( [a_1], \ldots, [a_n] \). Other states can be composed from the basis states by mixing them. Thus, a general state of system \( CA \) can be expressed as

\[
p_1[a_1] + \cdots + p_n[a_n],
\]

which, naturally enough, suggests that the state set should actually be identified with probability distributions over an \( n \)-element set \( A \). The state set thus becomes a convex set\(^2\) with extremals \(^3\) \( [a_1], \ldots, [a_n] \). Moreover, a representation of each state \( (3) \) as a convex combination of basis states \( [a_1], \ldots, [a_n] \) is clearly unique. This is a property which is going to change when we introduce quantum mechanics. Another fundamental property—that there is no other way but mixing for introducing new states—will also change.

3.2. Dynamics

In a physical system, there is one important thing that we should be able to describe: the dynamics of the system, i.e., how the state of the system changes as the time passes on. In connection with physics, one usually adopts a so-called causality principle (from past to future), which intuitively speaking, states, that when all circumstances are known, then the system state at time \( t_2 \) is fully determined of a system state at time \( t_1 \), for any \( t_2 > t_1 \). We will use this point of view, suitably formulated for the computational aspects: in fact, we will be interested in observing the system \( S_i \) representing the alphabet at some time points \( t_1, t_2, \ldots \), and regard the time \( t_{i+1} - t_i \) as the time needed to perform an elementary computational step. It is also often assumed that, for each \( i \), \( t_{i+1} - t_i = \tau \) is a constant.

To describe the state transformation

\[
p_1[a_1] + \cdots + p_n[a_n] \mapsto p'_1[a_1] + \cdots + p'_n[a_n]
\]

between time points \( t_i \) and \( t_{i+1} \), we will adopt, besides the causality principle, an assumption that mapping \( (4) \) should be linear. This condition is easily understood by using the ignorance interpretation for the mixed state

\[
p_1[a_1] + \cdots + p_n[a_n]
\]

\(^2\) A subset \( C \) of a real vector space is convex, if \( x = \lambda x_1 + (1 - \lambda) x_2 \in C \), whenever \( x_1, x_2 \in C \) and \( \lambda \in [0, 1] \). We say that \( x = \lambda x_1 + (1 - \lambda) x_2 \) is a convex combination of \( x_1 \) and \( x_2 \).

\(^3\) An element \( x \) of a convex set \( C \) is called an extremal, if it cannot be represented as a convex combination of two distinct elements \( x_1, x_2 \in C \).
since the ignorance interpretation says that a system in state (5) actually is in some of the pure states \([a_i]\), in any such with a probability of \(p_i\). In addition to the linearity, we will not introduce more restrictions than those ones arising from the structure of the state set itself: if \(p_i \geq 0\) and \(p_1 + \cdots + p_n = 1\), then it must also hold that \(p'_i \geq 0\) and that \(p'_1 + \cdots + p'_n = 1\).

Using these restrictions, it easily follows that the state change (4) is induced by a *Markov matrix*.\(^4\) That is, denoting \(p = (p_1, \ldots, p_n)^T\) and \(p' = (p'_1, \ldots, p'_n)^T\),\(^5\) then equation

\[ p' = M_i p \]  

must hold for some Markov matrix \(M_i\). We will adopt Eq. (6) as the basic model for connecting the system states at time \(t_i\) and \(t_{i+1}\). It is also assumed that the matrix \(M_i\) is fully determined by the conditions affecting the system.

**Remark 1.** Using the previous notations, a probabilistic algorithm as computational device (1) is understood as a device which, given an input in \(A^*\), gives the output as a mixed state describing some elements in \(B^*\).

4. **Representing the alphabet—quantum physics**

4.1. **Developmental aspects**

In the end of the 19th, and in the beginning of the 20th century, observations not explainable according to the theory of physics developed so far, gave reason enough to introduce a new *quantum theory*. In fact, even much earlier there had been discussion within the scientific community about the nature of some physical phenomena, including *light*: should we regard light as a particle flow or as an undulatory phenomenon? Some observations, like the reflection, supported the particle theory, but other ones, like interference, suggested that light should in fact be regarded as a wave-like action.

It is somehow simplifying, but also well illustrative to say that quantum mechanics actually unified the two apparently different points of view: light, as well as matter, can be seen as particles and also as waves. It should be emphasized that this summary ignores the deep philosophical difficulties included even in the most recent developments of quantum theory, but it also somehow explains the form of the *mathematical machinery* used in modern quantum physics.

Physics itself, is ultimately an empirical science: a theory cannot be a good one unless it is supported by observations. On the other hand, a deep analysis of the mathematical tools beyond quantum physics may lead, and in fact, has lead, to new experiments that may be useful for further developments of the theory. As a consequence, the most widespread mathematical machinery used to describe quantum physics, so called *Hilbert space formalism* of quantum mechanics is not easy to derive deductively from

---

\(^4\) A Markov matrix is a matrix with nonnegative entries such that the entries in each column sum up to 1.

\(^5\) Notation \(x^T\) stands for the transpose of a vector \(x\).
the observations. The theory itself has originally emerged from the need to introduce mathematical tools to explain the strange probability distributions observed in experiments. The formalism has later been developed by mathematically analysing its core features.

4.2. State set

Previously, we have harnessed a physical system $C_A$ for representing an alphabet $A = \{a_1, \ldots, a_n\}$, but only seen as a system of classical physics, and we have depicted the state set of the system as probability distributions over an $n$-element set $A$. That is, a state of the system can be seen as a vector in $\mathbb{R}^n$, whose all components are nonnegative and sum up to 1. The basis states $[a_1], \ldots, [a_n]$ which we considered were the extremals of the state set in the sense that each other (mixed) state can be expressed as a convex combination of the basis states.

In quantum physics, there should necessarily be another method for composing new states from the given ones—superposition. This requirement, which originates from the description of particle systems as waves, essentially results in the Hilbert space formalism, where the states of the quantum systems are represented as self-adjoint, positive, unit-trace operators. We will now study the concepts mentioned in this cryptic-sounding notion.

4.2.1. Finite-dimensional Hilbert spaces

The aim here also is to represent an alphabet $A = \{a_1, \ldots, a_n\}$ by using a physical system, but now we will describe how the quantum physical description works. The very basic component of the description is an $n$-dimensional complex vector space $H_n$, called Hilbert space.\[6\]

We fix a basis $A = \{a_1, \ldots, a_n\}$ of $H_n$, which we can use to represent any $x \in H_n$ as

$$x = x_1 a_1 + \cdots + x_n a_n$$

with unique coefficients $x_1, \ldots, x_n \in \mathbb{C}$. Moreover, writing

$$y = y_1 a_1 + \cdots + y_n a_n,$$

we can introduce the inner product by

$$\langle x \mid y \rangle = x_1^* y_1 + \cdots + x_n^* y_n \tag{7}$$

The basis $A = \{a_1, \ldots, a_n\}$ is orthonormal with respect to the inner product (7). We will associate the basis vector $a_i$ to the element $a_i \in A$. Because of our purposes, this basis will also be referred as to a computational basis.

\[6\]For a general definition of a Hilbert space, see [15], for example.

\[7\]Notation $x^*$ means the complex conjugate of $x \in \mathbb{C}^*$. Usually, an inner product of a vector space over complex numbers is required to be linear with respect to the first component, but here we follow the notion more widely used in the literature of physics.
In order to introduce the notion of the states of quantum systems, we have to study linear mappings $H_n \rightarrow H_n$, also called operators. Such mappings also form a vector space, denoted by $L(H_n)$, when the addition and scalar multiplication are defined pointwise.

We will now introduce a useful notation for elements of $L(H_n)$. For any vectors $x, y \in H_n$, we define mapping $\langle x \rangle \langle y \rangle$ by setting

$$\langle x \rangle \langle y \rangle(z) = \langle y | z \rangle x.$$  

By using the properties of inner product (7), it is plain to check that $\langle x \rangle \langle y \rangle$ is linear mapping. Moreover, it is not hard to see that mappings $\langle a_i \rangle \langle a_j \rangle$, where $a_i, a_j \in A$ are linearly independent and that all linear mappings $T \in L(H_n)$ can be represented as linear combinations of mappings $\langle a_i \rangle \langle a_j \rangle$. In fact, if $T \in L(H_n)$, then we can, after short calculations, see that

$$T = \sum_{i=1}^{n} \sum_{j=1}^{n} \langle a_i | T a_j \rangle \langle a_i \rangle \langle a_j \rangle.$$  

(8)

If $\langle a_i | T a_j \rangle = 0$ for each $i, j \in \{1, \ldots, n\}$, then $T a_j$ is orthogonal to each $a_i$. But since the vectors $a_i$ span the whole space $H_n$, this is possible only if $T a_j = 0$. Because this is true for each $j$, we must conclude that $T = 0$. Thus, we have obtained.

**Proposition 2.** Vector space $L(H_n)$ has basis $\{ \langle a_i \rangle \langle a_j \rangle | i, j \in \{1, \ldots, n\} \}$. Therefore, the dimension of $L(H_n)$ is $n^2$.

The above proposition is well known from elementary linear algebra. In fact, representation (8) is merely a slight reformulation of the familiar matrix representation of linear mapping $T$ with respect to basis $A$.

The trace of a mapping $T \in L(H_n)$ is defined to be

$$\text{Tr}(T) = \sum_{i=1}^{n} \langle a_i | T a_i \rangle.$$  

It is easy to verify that the trace is independent of the orthonormal basis chosen. In matrix representation (8), the trace is clearly the sum of the diagonal elements.

An important subclass of linear mappings $H_n \rightarrow H_n$ are self-adjoint mappings. For each

$$T = \sum_{i=1}^{n} \sum_{j=1}^{n} \langle a_i | T a_j \rangle \langle a_i \rangle \langle a_j \rangle$$

there exists the adjoint mapping of $T$ defined by

$$T^* = \sum_{i=1}^{n} \sum_{j=1}^{n} \langle T a_i | a_j \rangle \langle a_i \rangle \langle a_j \rangle,$$

which has the property that $\langle x | T y \rangle = \langle T^* x | y \rangle$ for each $x, y \in H_n$ (in fact, this property could have been used to define $T^*$, as well). Notice that $(T^*)^* = T$. Mapping $T$ is called self-adjoint if $T^* = T$. 
Self-adjoint mappings have important structural properties, see [15] for the proofs of the following propositions:

**Proposition 3.** If \( T \in L(H_n) \) is self-adjoint, then \( T \) has real eigenvalues. Moreover, there is an orthonormal basis of \( H_n \) consisting of the eigenvectors of \( T \).

**Proposition 4 (Spectral representation).** Any self-adjoint mapping \( T \in L(H_n) \) can be represented as

\[
T = \lambda_1 |x_1 \rangle \langle x_1| + \cdots + \lambda_n |x_n \rangle \langle x_n|,
\]

where \( \lambda_1, \ldots, \lambda_n \) are the eigenvalues of \( T \) (not necessarily distinct), \( x_i \) is an eigenvector of \( T \) belonging to \( \lambda_i \), and \( \{x_1, \ldots, x_n\} \) is an orthonormal set.

Notice that in the above proposition, each mapping \( |x_i \rangle \langle x_i| \) is a projection onto the one-dimensional subspace of \( H_n \) spanned by \( x_i \).

**Remark 5.** For the continuation, it is even more important to notice that the spectral representation (9) is not unique in general. It is however true that the eigenvalues of a mapping \( T \) are uniquely determined, but if \( T \) has \(< n \) distinct eigenvalues, then, according to Proposition 3, there are \( k \geq 2 \) orthonormal eigenvectors \( x_{i_1}, \ldots, x_{i_k} \) belonging to some eigenvalue \( \lambda_i \). In that case, a partial sum

\[
\lambda_i (|x_{i_1} \rangle \langle x_{i_1}| + \cdots + |x_{i_k} \rangle \langle x_{i_k}|)
\]

of representation (9) can be represented in many different ways. In fact, vectors \( x_{i_1}, \ldots, x_{i_k} \) generate a \( k \)-dimensional subspace of \( H_n \), so-called eigenspace of \( \lambda_i \). But any other orthonormal basis \( x'_i, \ldots, x'_k \) of that eigenspace would do as well. That is, for such a basis, mapping

\[
\lambda_i (|x'_{i_1} \rangle \langle x'_{i_1}| + \cdots + |x'_{i_k} \rangle \langle x'_{i_k}|)
\]

is identical with (10). Therefore, in representation (9), the one-dimensional projections \( |x_i \rangle \langle x_i| \) are not uniquely determined, unless \( T \) has \( n \) distinct eigenvalues. Notice that even in the case that \( T \) has \( n \) distinct eigenvalues, we are not claiming that vectors \( x_i \) are unique, only that the mappings \( |x_i \rangle \langle x_i| \) are.

On the other hand, if \( x_{i_1}, \ldots, x_{i_k} \) is an orthonormal basis of the eigenspace of \( \lambda_i \), then one can see that the mapping

\[
|x_{i_1} \rangle \langle x_{i_1}| + \cdots + |x_{i_k} \rangle \langle x_{i_k}|
\]

is a projection onto the eigenspace of \( \lambda_i \). We can therefore introduce another formulation of Proposition 4

---

\( ^8 \) If \( T \) has \( n \) distinct eigenvalues, then \( T \) is called nondegenerate, otherwise degenerate.
Proposition 6 (The spectral representation). Any self-adjoint mapping \( T \in L(H_n) \) can be uniquely represented as
\[
T = \lambda'_1 P_1 + \cdots + \lambda'_n P_n',
\]
where \( n' \leq n \), \( \lambda'_1, \ldots, \lambda'_n \) are distinct eigenvalues of \( T \), and \( P_i \) is a projection onto the eigenspace of \( \lambda'_i \).

To conclude this section, we introduce one more notion: a self-adjoint mapping \( T \) is said to be positive, if \( \langle x | Tx \rangle \geq 0 \) for each \( x \in H_n \). It turns out that \( T \) is positive if and only if all of its eigenvalues are nonnegative.

4.2.2. Interpretations

Finally, we are ready for introducing the mathematical description for the state of a quantum system. After the formal description, we will learn how to interpret the notion of the state and study some properties of states.

If a quantum system \( Q_A \) should be capable for representing alphabet \( A = \{a_1, \ldots, a_n\} \) reliably, i.e., for each \( a_i \) there should be a state such that an observation gives \( a_i \) with a probability of 1, we associate to such a system a Hilbert space \( H_n \), referred as to the state space of the system.

Postulate 1. The states of the system \( Q_A \) are identified with unit-trace, self-adjoint positive mappings in \( L(H_n) \).

To interpret the abstract notion of state, we will also associate the notion of observable to system \( Q_S \).

Definition 7. A (sharp)\(^9\) observable \( O \) of a quantum system \( Q_A \) with state space \( H_n \) is a collection
\[
O = \{P_1, \ldots, P_k\},
\]
where each \( P_i \) is a projection onto a subspace \( E_i = P_i(H_n) \subseteq H_n \), spaces \( E_i \) are mutually orthogonal and \( P_1 + \cdots + P_k = I \) is the identity mapping on \( H_n \).

Since there is a one-to-one correspondence between subspaces and the projections onto the subspaces, we could equivalently represent an observable as a collection \( \{E_1, \ldots, E_k\} \), where \( E_i \) are mutually orthogonal subspaces such that \( E_1 + \cdots + E_k = H_n \). Thus, an observable \( O \) is simply a decomposition of \( H_n \) into mutually orthogonal subspaces.

The intuitive interpretation of an observable \( O = \{P_1, \ldots, P_k\} \) is that each subspace \( P_i(H_n) \) represents a property which the system \( Q_A \) can have. For instance, when representing alphabet \( A = \{a_1, \ldots, a_n\} \), we have fixed an orthonormal basis, so-called computational basis \( \{a_1, \ldots, a_n\} \), which shatters the space \( H_n \) into \( n \) orthogonal one-dimensional subspaces spanned by vectors \( a_i \). This collection defines an observable,

\(^9\) There is also a notion of “unsharp observable” [8], which we will not treat here.
and the intuitive meaning of the subspace $A_i$ spanned by $a_i$ is that if the state of the system is “near” to the projection onto $A_i$,\(^{10}\) then it is likely that observing the system will outcome the result that the system was representing letter $a_i$.

**Remark 8.** As everyone can easily understand, that when observing a physical system, the outcome does not look like “a subspace $E_i$”, but rather like “12.475 m”, “5.2 J” or “the particle has been detected” (which can be viewed as a $\{0,1\}$-valued experiment), etc. In fact, traditionally in quantum physics, the notion of an observable has not been defined as a set of projections subspaces $O = \{P_1, \ldots, P_k\}$, onto mutually orthogonal subspaces, but rather as a set $O$ together with a set of $k$ distinct real numbers $\lambda_1, \ldots, \lambda_k$ such that $\lambda_i$ is associated to $P_i$. Thus, the notion “state of the system is close to $P_i$”\(^{11}\) intuitively means that, when observing $O$, the outcome will be $\lambda_i$ with high probability.

Associating a set of distinct real numbers to projections opens another way to envisage an observable: we could as well treat an observable $O = \{P_1, \ldots, P_k\}$ with associated real numbers $\lambda_1, \ldots, \lambda_k$ as a self-adjoint operator

$$A_O = \lambda_1 P_1 + \cdots + \lambda_k P_k. \tag{12}$$

In fact, because projections $P_i$ are mutually orthogonal and the values $\lambda_i$ real, it is trivial to see that (12) always defines a self-adjoint operator, and, by Proposition 6, any self-adjoint operator $A$ has unique representation as in (12). Representation (12) is usually used to formulate the so-called *uncertainty relations* of quantum physics.

In this article, we will however mainly consider the real numbers as “labels” of the subspaces and ignore them. On the other hand, these labels are handy in formulating the following postulate.

**Postulate 2.** Let $Q_A$ be a quantum system and $O = \{P_1, \ldots, P_k\}$ an observable with associated “labels” $\{\lambda_1, \ldots, \lambda_k\}$. Then the probability that measuring observable $O$ will give $\lambda_i$ as the outcome when the system state is $T$, is given by

$$\text{Prob}(\lambda_i) = \text{Tr}(P_i T).$$

The above postulate is frequently referred as to the *minimal interpretation* of quantum physics. We should also verify that the Postulate 2 defines a probability distribution in a reasonably way. This is verified in the following proposition, whose proof can be found in [15].

**Proposition 9.** Let $T \in \mathcal{L}(H_n)$ be a unit-trace, self-adjoint positive mapping and $O = \{P_1, \ldots, P_k\}$ a set of mutually orthogonal projections such that $P_1 + \cdots + P_k = I$. Then

---

\(^{10}\)Notice that $|a_i\rangle\langle a_i|$, which is a projection onto $A_i$, is a unit-trace, self-adjoint positive operator. That is, $|a_i\rangle\langle a_i|$ is a potential state of the system $Q_A$.

\(^{11}\)Notice here that if $P_i$ is a projection onto a subspace of dimension more than 1, then $P_i$ is not a potential state of system $Q_A$. Although it is easy to verify that all projections are positive and self-adjoint, they do not have unit trace, if the dimension exceeds 1.
Example 10. Let us again study the alphabet $A = \{a_1, \ldots, a_n\}$ represented by a quantum system $Q_A$ with computational basis $A = \{a_1, \ldots, a_n\}$. Then $O = \{|a_1\rangle\langle a_1|, \ldots, |a_n\rangle\langle a_n|\}$ is a collection of projections onto mutually orthogonal subspaces, and

$$|a_1\rangle\langle a_1| + \cdots + |a_n\rangle\langle a_n| = I$$

is the identity mapping. In other words, $O$ defines an observable of system $Q_A$. Let us equip projection $|a_i\rangle\langle a_i|$ with label $i$. Now $T_j = |a_j\rangle\langle a_j|$ is a self-adjoint, positive, unit-trace mapping in $H_n$, and therefore it is a potential state of the system $Q_A$. The probability that measuring the observable $O$ will give $i$ as outcome (meaning that the system was found to represent letter $a_i$) is given by

$$\text{Tr}(|a_i\rangle\langle a_i| T_j) = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases}$$

Hereafter, we will usually identify the notions “system is in state $|a_i\rangle\langle a_i|$” and “system is representing letter $a_i$”—for this “letter observable” $O$ there exist $n$ states $T_1, \ldots, T_n$ such that whenever the system is in state $T_j$, then, by measuring the observable $O$, we find, with a probability of 1, that the system is representing letter $a_j$. That is, the system is capable of faithfully representing alphabet $A$.

Now, if $T$ is a given state of system $Q_A$ representing alphabet $A$, we can find the probabilities $p_i$ for each letter $a_i$ to occur, and then introduce a classical system having state

$$p_1[a_1] + \cdots + p_n[a_n],$$

giving exactly the same behaviour. One may now wonder why to introduce the quantum systems at all? To that question we answer that even though it is possible to imitate any instantaneous description of a quantum system by a classical one, we will see, that the time evolution of quantum systems cannot be directly imitated by classical ones.

4.2.3. Structural properties of the state set

Let us recall that a state $T$ of a system $Q_A$ is here represented as a self-adjoint, positive, unit-trace mapping $H_n \to H_n$. According to Proposition 4, there is a representation

$$T = \lambda_1|x_1\rangle\langle x_1| + \cdots + \lambda_n|x_n\rangle\langle x_n|,$$

where $\{x_1, \ldots, x_n\}$ are the eigenvectors of $T$ forming an orthonormal basis of $H_n$, and $\lambda_i$ are the eigenvalues of $T$. Self-adjointness of $T$ implies that each $\lambda_i$ is real, positivity of $T$ implies that each $\lambda_i$ is nonnegative, and, finally, the condition on the trace implies that $\lambda_1 + \cdots + \lambda_n = 1$. This means, that any self-adjoint, positive unit-trace operator
can be expressed as a *convex combination* of one-dimensional mutually orthogonal projections $|x_i\rangle\langle x_i|$. Unfortunately, representation (13) is not necessarily unique, as we have seen, cf. Remark 5.

However, representation (13) is worth investigating. If $T = |x\rangle\langle x|$ originally is some one-dimensional projection, it turns out that then $T$ is not expressible as convex combination of other one-dimensional projections [16]. In other words:

**Proposition 11.** The set of states of system $Q_A$ is a convex set having one-dimensional projections as extremals.

**Definition 12.** A state $T$ of system $Q_A$ is **pure**, if $T$ is a projection onto a one-dimensional subspace. Otherwise, $T$ is **mixed**.

**Remark 13.** Let $T = \lambda T_1 + (1 - \lambda) T_2$ be a representation of $T$ as a convex combination of states $T_1$ and $T_2$ (here we do not assume that $T_1$ and $T_2$ must be pure). If $O = \{P_1, \ldots, P_k\}$ is an observable, then the probabilities induced by state $T$ are of form

$$\text{Tr}(P_i T) = \lambda \text{Tr}(P_i T_1) + (1 - \lambda) \text{Tr}(P_i T_2),$$

which shows that the statistical properties of a mixture $\lambda T_1 + (1 - \lambda) T_2$ behave exactly as in the classical case.

Very typically in the theory of quantum computation it is assumed that the state of a computational device can always be described as a pure state. Therefore, we will now investigate the pure states with the computational basis $A = \{a_1, \ldots, a_n\}$ and the “letter observable” $O = \{|a_1\rangle\langle a_1|, \ldots, |a_n\rangle\langle a_n|\}$ in mind: if $T = |x\rangle\langle x|$ for some unit-length vector $x$, we can find a representation

$$x = x_1 a_1 + \cdots + x_n a_n$$

for some complex numbers $x_1, \ldots, x_n$. Because $x$ has unit length, these numbers must satisfy

$$|x_1|^2 + \cdots + |x_n|^2 = 1.$$

If we now assume that our system is in state $|x\rangle\langle x|$, then the probability that measuring observable $O$ will give letter $a_i$ as outcome, is, according to Postulate 2, given by

$$\text{Tr}(|a_i\rangle\langle a_i| |x\rangle\langle x|) = \langle x | |a_i\rangle\langle a_i| |x\rangle\langle x|$$

$$= \langle x | |a_i\rangle\langle a_i| x \rangle$$

$$= \sum_{k=1}^{n} \sum_{l=1}^{n} x_k^* x_l \langle a_k | |a_i\rangle\langle a_i| a_l \rangle$$

$$= \sum_{k=1}^{n} x_k^* x_l \langle a_i | a_k \rangle \langle a_i | a_l \rangle$$

$$= |x_i|^2.$$
The two first equalities are simply due to the fact that trace can be computed by using any orthonormal basis, so we can use one which includes $x$. Then, mapping $|x⟩⟨x|$, as a projection onto a one-dimensional subspace spanned by $x$, annihilates all members of that basis but leaves $x$ untouched. The remaining equalities are obtained by only using representation (14) and the orthonormality of basis $\{a_1, \ldots, a_n\}$. This gives us into the following proposition:

**Proposition 14.** If $T = |x⟩⟨x|$ is a pure state of quantum system $Q_A$ representing alphabet $A = \{a_1, \ldots, a_n\}$, and

$$x = x_1 a_1 + \cdots + x_n a_n,$$

and $O = \{|a_1⟩⟨a_1|, \ldots, |a_n⟩⟨a_n|\}$ the “letter observable”, then the probability that measuring $O$ will give “letter $a_i$” as outcome, is $|x_i|^2$.

**Remark 15.** Notice that if we replace the observable $O$ above with $O’ = \{|x_1⟩⟨x_1|, \ldots, |x_n⟩⟨x_n|\}$ such that $\{x = x_1, \ldots, x_n\}$ is an orthonormal basis (such a basis always exists), then notation (15) would become

$$x = 1 \cdot x_1 + 0 \cdot x_2 + \cdots + 0 \cdot x_n.$$

Then the probability that in state $|x⟩⟨x|$, a measurement yields result “subspace spanned by $x_1$” is 1. This means that for any pure state $T = |x⟩⟨x|$ there is an observable with $n$ potential values, but one of them will be seen with a probability of 1, when the system is in state $T$.

For a pure state $T = |x⟩⟨x|$ with $x$ as in (15) we use also a conventional notation

$$|x⟩ = x_1 |a_1⟩ + \cdots + x_n |a_n⟩.$$  

(16)

Notation (16) is referred as to vector state notation. Recall that a pure state is a projection onto a one-dimensional subspace of $H_n$, so a generating vector of that subspace is enough to describe the state. It should however be noted that a unit-length vector $x$ spanning a one-dimensional subspace of $H_n$ is not unique, but any vector of form $e^{iθ} x$ with real $θ$ would do as well.

Notice carefully that (16) does not mean a linear combination of pure states $|a_i⟩⟨a_i|$, but it is essentially the same as (15). So-called “ket”-notation $|·⟩$ originally due to P. Dirac is here to emphasize that instead of merely speaking about vectors in $H_n$, we are referring to the pure states generated by the unit-length vectors. When $x$ can be represented as in (15), we say that pure state $|x⟩$ is a superposition of pure states $|a_1⟩, \ldots, |a_n⟩$ with amplitudes $x_1, \ldots, x_n$.

To be more precise, one could figure out the properties of notation $|·⟩⟨·|$ to write

$$|x⟩⟨x| = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j^* |a_i⟩⟨a_j|$$

$$= \sum_{i=1}^{n} |x_i|^2 |a_i⟩⟨a_i| + \sum_{i̸=j} x_i x_j^* |a_i⟩⟨a_j|. $$

(17)
Eq. (17) reveals that the representation $|x\rangle\langle x|$ as a superposition (16) in fact consists of a convex combination

$$\sum_{i=1}^{n} |x_i|^2 |a_i\rangle\langle a_i|$$

of states $|a_i\rangle\langle a_i|$ (recall that $\sum_{i=1}^{n} |x_i|^2 = 1$) together with so called interference term

$$\sum_{i\neq j} x_i x_j^* |a_i\rangle\langle a_j|.$$  

If we are only interested in the statistics associated to the “letter observable” $O = \{|a_1\rangle\langle a_1|, \ldots, |a_n\rangle\langle a_n|\}$, i.e., finding out the probabilities to see letter $a_i$ as the outcome of a measurement, then it is clear that the state (18) alone would yield exactly the same probabilities as state (16). On the other hand, we will see that the states (16) and (18) can behave in an essentially different manner, when considering the time evolution of the system.

The possibility of introducing new pure states as a superposition of other pure states has no counterpart in classical physics.

4.3. Dynamics

Earlier, when considering the time evolution of classical systems (see Section 3.2), we restricted to time evolution at discrete time steps, i.e., how to describe the state of the system at certain time points $t_1, t_2, \ldots$. Here we will follow the same outlining.

Recall that in classical systems we assumed the evolution mapping to obey the causality principle: we must be able to determine the state of the system at time $t_{i+1}$ from the state at time $t_i$. The additional requirement was the linearity: if $T_{i+1}$ and $T_i$ are the states of the system at times $t_{i+1}$ and $t_i$, respectively, and $T_{i+1} = V_i(T_i)$, where $V_i$ is the mapping implementing the time evolution $t_i \to t_{i+1}$, then $V_i$ should, because of the ignorance interpretation, always work as a linear operator.

Another requirement for the form of each mapping $V_i$ is that for each state $T$, $V_i(T)$ should also be a state. In classical systems, these requirements together were sufficient to imply the model where state $T_{i+1}$ is obtained from the previous state $T_i$ by multiplying via using a Markov matrix $M_i$. When determining the form of the quantum time evolution operators $V_i$, we will use essentially the same requirements as we used for classical systems.

Since we know that all states of quantum system can be represented as convex combinations of pure states, we could use a strategy of finding the time evolution of pure states, and then try to extend that for mixed states. It should, however, be emphasized here that now it is not so straightforward to justify the requirement that the time evolution mapping should be linear: we have seen that the decomposition of a mixed state into pure states is not necessarily unique, which means that the ignorance interpretation simply does not work here. In fact, the problem of justifying the linearity of the time evolution of quantum systems has not been resolved so far.  

12 Experiments have long supported the idea of linear evolution in quantum systems.
Here we will simply assume that the time evolution mappings are linear, and find their form.

Unfortunately, we should here introduce another requirement, associated to compound systems. It is not enough to assume that whenever $T_i$ is a state (i.e., unit-trace, self-adjoint positive operator) of a quantum system $Q_A$, but we must also take into account the potential environment system. Intuitively speaking, we will assume, that whenever $T_i$ is a state of quantum system consisting of $Q_A$ and some environment, then the time evolution would transform $T_i$ into a state of the larger system. Formally speaking, we will require that each time evolution mapping should be a completely positive mapping (we will not define the notion here, see [15] for details) $L(H_n) \rightarrow L(H_n)$. For the proof of the following proposition, see [15].

**Proposition 16** (Quantum time evolution). Let $Q_A$ be a quantum system with state space $H_n$. Let also $V : L(H_n) \rightarrow L(H_n)$ be a completely positive linear mapping such that $V(T)$ is a state whenever $T$ is. Then there exist $n^2$ linear mappings $V_j \in L(H_n)$ such that $V$ can be represented as

$$V(T) = \sum_{j=1}^{n^2} V_j TV_j^*, \quad (20)$$

where mappings $V_j$ satisfy

$$\sum_{j=1}^{n^2} V_j^* V_j = I.$$

We consider Eq. (20) as a general form of time evolution of a quantum system. Notice that it emerges from the facts that we assumed “from past to future”-causality principle, that evolution should be linear, that evolution mapping should map states to states, and finally from the fact that the mapping should be completely positive.

**Remark 17.** In the literature, it is often stated that quantum time evolution should always be reversible, i.e., to obey also form “future to past”-causality also. In other words, the state at time $t_i$ should be recoverable from the state at time $t_{i+1}$. However, the conditions mentioned above are not sufficient to imply reversibility, but mapping (20) can be irreversible as well.

On the other hand, if we restrict to operators the system $Q_A$ which cannot change a pure state into a mixed one (we say that a quantum system with such a time evolution is closed), then the time evolution is reversible. See [15] for the proof of the following proposition.

**Proposition 18** (Closed quantum time evolution). Let $Q_A$ be a quantum system with state space $H_n$. Let also $V : L(H_n) \rightarrow L(H_n)$ be a linear mapping such that $V(T)$ is a pure state whenever $T$ is. Then there exists a linear mapping $U \in L(H_n)$ such that
U^*U = I and V can be represented as

\[ V(T) = UTU^*. \]  \hspace{1cm} (21)

**Definition 19.** If \( U \in L(H_n) \) satisfies \( U^*U = I \), then \( U \) is called unitary.

A unitary mapping \( U \in L(H_n) \) is always invertible—\( U^* \) is the inverse of \( U \) by the very definition. It follows that if \( V \) is a time evolution operator of system \( Q_A \) defined by \( V(T) = UTU^* \), then the inverse of \( V \) is defined by \( V^{-1}(T) = U^*TU \). Thus, the time evolution of a closed quantum system obeys also “from future to past”-causality. For further properties of unitary operators, we refer to [15].

**Remark 20.** If \( A, B, |x\rangle\langle y| \in L(H_n) \), then a short computation reveals that \( |Ax\rangle\langle By| = A|x\rangle\langle y|B^* \). Thus a time evolution (21) for a pure state \( T = |x\rangle\langle x| \) can also be written as

\[ V(|x\rangle\langle x|) = U|x\rangle\langle x|U^* = |Ux\rangle\langle Ux|, \]

which, in turn, by using the vector state notation (16) means that the time evolution operator \( V \) transfers the vector \( x \) describing the state into \( Ux \). If

\[ |x\rangle = x_1|a_1\rangle + \cdots + x_n|a_n\rangle \]

and

\[ |Ux\rangle = x'_1|a_1\rangle + \cdots + x'_n|a_n\rangle \]

respectively are the representations of states \( |x\rangle \) and \( |Ux\rangle \) as a superposition of basis states \( |a_1\rangle, \ldots, |a_n\rangle \), then the transformation \( |x\rangle \mapsto |Ux\rangle \) can be written as

\[ (x'_1, \ldots, x'_n)^T = U(x_1, \ldots, x_n)^T, \]

where \( U \) is a unitary matrix. Notice the apparent similarity between closed quantum system evolution (22) and the dynamics of a classical probabilistic system (6): the probabilities are replaced with the amplitudes \( x_i \) and the Markov matrices by unitary matrices. In most applications of quantum computing, only a closed system evolution (22) is considered, but the time evolution may be quite different, as revealed by the following example.

**Example 21.** Consider a binary alphabet \( A = \{0, 1\} \). The Hilbert space associated to this system is two-dimensional space \( H_2 \). A quantum system representing a binary alphabet is called a *quantum bit*, or *qubit*, for short. We fix an orthonormal computational basis \( \{0, 1\} \), \(^\text{13}\) and if there is no danger of confusion, we sometimes also

\(^{13}\) Notice that here does not refer to zero vector, but is just a notation which associates logical 0 to a basis vector.
identify basis states $|0\rangle$ and $|1\rangle$ with vectors $0$ and $1$, respectively.\footnote{Using notation $|x\rangle$ for a vector also is somewhat misleading, since the pure state $|x\rangle$ does not uniquely determine the vector $x$, but states $|x\rangle$ and $|e^{i\theta}x\rangle$ are identical for each $\theta \in \mathbb{R}$. On the other hand, this notation leads to quite useful mnemonics: for instance, $|y\rangle\langle z| x\rangle = \langle z | x \rangle |y\rangle$. When using notation $|x\rangle$ for vectors also, we usually use a controversial identification system: notation $|x\rangle$ is first fixed to mean some particular vector, and then extended to mean also to the pure state determined by that vector.} For the matrix representation of the time evolution operator, we use coordinate representations

$$
0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad 1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}
$$

for the vectors which determine the basis states. Now, matrix

$$
W_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}
$$

is unitary, easily verified. It thus determines a closed time evolution in system $H_2$. Because the system remains closed, and the time evolution is linear, it suffices to know the effect of $W_2$ on basis states $|0\rangle$ and $|1\rangle$. An easy computation gives

$$
W_2|0\rangle = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle,
$$

$$
W_2|1\rangle = \frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle.
$$

Recall that the notations above denote superposition of basis states, not a mixture. Consider now our system in a pure state

$$
T = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle.
$$

Then, measuring the “letter observable”, i.e., finding out whether the system represents 0 or 1 will give 0 with a probability of $|1/\sqrt{2}|^2 = 1/2$, and 1 with a probability of $1/2$, as well.

Assume then, that the system undergoes a time evolution determined by $W_2$. The state $T$ transforms then into

$$
W_2T = \frac{1}{\sqrt{2}} \left( \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) + \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right)
$$

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

$$
= |0\rangle,
$$

meaning that in state $W_2T$, the system is found to represent 0 with a probability of 1.

A remarkable feature is now seen, if one considers the evolution $W_2$ itself: if the system is initially in any of states $|0\rangle$ or $|1\rangle$, then, after evolution $W_2$, 0 or 1 are seen...
both with a probability of 1/2 (notice that state (27) is reached from $|0\rangle$ by applying $W_2$ once). But when the state is a superposition (27), when both 0 and 1 can be seen, then an evolution determined by $W_2$ cancels out the possibility of seeing 1. This kind of behaviour is obviously impossible for classical probabilistic systems.

The strengthening and cancellation of coefficients of $|0\rangle$ and $|1\rangle$ in (28) are called constructive and destructive interference, respectively.

5. Compound quantum systems

In the previous section we learned how the quantum system representing alphabet $A = \{a_1, \ldots, a_n\}$ is treated. Postulate 1 established the representation of the system states as self-adjoint, unit-trace positive operators of a Hilbert space $H_n$. Postulate 2 established the connection between the state and the probability of seeing a particular letter, and finally, we used several restrictive conditions to introduce the dynamics in Proposition 16.

It is worth repeating here, that most of the research on the theory of quantum computation is made by assuming that the system remains closed, leading to somewhat simpler formalism utilizing only pure states, which essentially can be described as unit-length vectors in the state space $H_n$. In a closed system, the probabilities associated to measurements are also more clearly visible: they are found by representing the vector describing the state in the computational basis: the probabilities are the squared absolute values of the basis vector coefficients. Finally, the time evolution of a closed system is easy to describe as a unitary operator in $L(H_n)$.

On the other hand, so far we have introduced only how to treat a quantum system representing a single letter, but for computational purposes we want to represent systems of many letters. Therefore, we will introduce the representation of compound quantum systems. It turns out that the description will depend on whether the subsystems are distinguishable or indistinguishable. Here, we consider only systems which are distinguishable.

5.1. Upwards—to the compound system

We are not entering into a deep analysis, but will represent the compound system description as a postulate:

**Postulate 3.** Let $Q_A$ and $Q_B$ be two distinguishable quantum systems and $H_n$ and $H_m$ the Hilbert spaces associated to them. Then the Hilbert space associated to the compound system $Q_{AB}$ consisting of $Q_A$ and $Q_B$ is the tensor product $H_{nm} = H_n \otimes H_m$.

Again, we are not entering into details, but will only briefly explain the notion of a tensor product. For more details, consult [15], for example.

If $H_n$ and $H_m$ are some Hilbert spaces representing alphabets $A$ and $B$ with computational bases $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_m\}$, we define $H_n \otimes H_m$ as a vector space with basis consisting of all possible pairs $(a_i, b_j)$, but instead of notation $(a_i, b_j)$,
we write $a_i \otimes b_j$. Hence the dimension of $H_n \otimes H_m$ is $mn$. Notation $\otimes$ is then extended to all other pairs $(x, y) \in H_n \times H_m$ by expressing $x$ and $y$ in bases $A$ and $B$, respectively: if $x = x_1 a_1 + \cdots + x_n a_n$ and $y = y_1 b_1 + \cdots + y_m b_m$, then

$$x \otimes y = \sum_{i=1}^n \sum_{j=1}^m x_i y_j a_i \otimes b_j.$$  \hspace{1cm} (29)

Because of this definition, the tensor product of vectors is clearly bilinear

$$(a_1 x_1 + a_2 x_2) \otimes (b_1 y_1 + b_2 y_2)$$

$$= a_1 b_1 x_1 \otimes y_1 + a_1 b_2 x_1 \otimes y_2 + a_2 b_1 x_2 \otimes y_1 + a_2 b_2 x_2 \otimes y_2.$$ 

Introducing the inner product in $H_n \otimes H_m$ in the same manner as we introduced in $H_n$, one can utilize (29) to see that

$$\langle x_1 \otimes y_1 | x_2 \otimes y_2 \rangle = \langle x_1 | x_2 \rangle \langle y_1 | y_2 \rangle,$$ \hspace{1cm} (30)

where the inner products on the right-hand side are the inner products of $H_n$ and $H_m$, respectively.

If $A \in L(H_n)$ and $B \in L(H_m)$, then mapping $A \otimes B$ defined by

$$(A \otimes B)(a_i \otimes b_j) = Aa_i \otimes Bb_j$$

determines a unique linear mapping $H_n \otimes H_m \rightarrow H_n \otimes H_m$. We also say that mapping $A \otimes B \in L(H_n \otimes H_m)$ is the tensor product of $A$ and $B$. We will shortly see that there are also linear mappings in $L(H_n \otimes H_m)$, which are not expressible as $A \otimes B$. On the other hand, using basis $a_i \otimes b_j$ and formula (30) to calculate the trace, \hspace{1cm} 15 we see that

$${\rm Tr}(A \otimes B) = {\rm Tr}(A) {\rm Tr}(B).$$

It turns out that $H_n \otimes H_m$ can be defined (yielding into the same definition as we used here) independently of the basis, see [13], for example.

All other concepts associated to a compound system with distinguishable subsystems (state, observables, time evolution) are treated exactly in the same fashion as we treated them in the previous section. In fact, a quantum system representing both alphabets $A$ and $B$ can be treated as a single system representing alphabet $A \times B$. One may therefore wonder why to handle compound systems in details at all, but one thing for compound systems is not so straightforward: how to recover the states of the subsystems, when the state of the compound system is known. Later, we will also use a compound system description to express general quantum time evolution (20) in a form of closed time evolution (21).

Example 22. Let us consider a system $Q_{2 \times 2}$ consisting of two qubits, for representing one qubit, we use space $H_2$ with computational basis $\{|0\rangle, |1\rangle\}. \hspace{1cm} 16$ Thus, for a system

\hspace{1cm} 15 Notice that the basis $\{a_i \otimes b_j | i \in \{1, \ldots, n\}, j \in \{1, \ldots, m\}\}$ is orthonormal.

\hspace{1cm} 16 Notice that here we use notations $|0\rangle$ and $|1\rangle$ primarily for the vectors, and secondarily for the pure states they determine.
of two qubits, we use a computational basis
\[ \{|0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle \}, \]
as well as its shortened notations
\[ \{|00\rangle, |01\rangle, |10\rangle, |11\rangle \} \]
A general pure state of system \( Q_2 \times Q_2 \) is then determined by a vector
\[ c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle, \tag{31} \]
which has unit length, i.e.,
\[ |c_{00}|^2 + |c_{01}|^2 + |c_{10}|^2 + |c_{11}|^2 = 1. \]
The measurement probabilities are easily recovered from pure state (31): the probabilities of seeing 00, 01, 10, and 11 as the values of the quantum bits are given by |c_{00}|^2, |c_{01}|^2, |c_{10}|^2, and |c_{11}|^2, respectively.

A pure state
\[ \frac{1}{\sqrt{2}} (|00\rangle + |01\rangle + |10\rangle + |11\rangle) = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \tag{32} \]
can be expressed as a (tensor) product of two one-qubit pure states. In this case, we say that a state with that property is decomposable. On the other hand, a pure state
\[ \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle \tag{33} \]
cannot be expressed as a (tensor) product of two one-qubit pure states, for any such attempt
\[ \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle = (a_0|0\rangle + a_1|1\rangle)(b_0|0\rangle + b_1|1\rangle) \]
\[ = a_0 b_0 |00\rangle + a_0 b_1 |01\rangle + a_1 b_0 |10\rangle + a_1 b_1 |11\rangle \]
leads into a system of equations \( a_0 b_0 = a_1 b_1 = 1/\sqrt{2} \), \( a_0 b_1 = a_1 b_0 = 0 \), which clearly cannot have any solution. A pure state of a compound system, such as (33), which cannot be expressed as a tensor product of two subsystem states, is said to be entangled.

**Remark 23.** Notice that when a two-qubit system in state (33) is observed, then one can see outcomes 00 and 11, both with a probability of 1/2. Especially, outcomes 01 and 10 cannot be observed, i.e., the qubits are perfectly correlated. Interestingly, the experiments have shown that this correlation can be detected even if the two qubits are spatially separated more than 10 km apart [24]. In [5] it was proposed that this nonlocal correlation could be used to generate a key for cryptographic purposes, in such

\[ ^{17} \text{A vector always determines a pure state uniquely.} \]
a manner, that any attempt of eavesdropping could be detected. For an experimental realization of such communication, see [17] for instance.

We now investigate the pure states (32) and (33). In the first case, it seems plain to say that the subsystem states are both pure states
\[
\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle).
\]
On the other hand, since pure state (33) cannot be expressed as a tensor product of pure states, it is not so straightforward to express the subsystem states.

5.2. Downwards—to the subsystems

When determining the subsystem states \( T_A \) and \( T_B \) from a known compound system state \( T \), we will use, as usual, a statistical approach. An observable as we defined it, is a collection of mutually orthogonal projections whose sum is the identity mapping. Let \( H_{nm} = H_n \otimes H_m \) be the state space of system that consists of subsystems \( Q_A \) and \( Q_B \), and consider an observable \( O_A = \{P_1, \ldots, P_k\} \) of system \( Q_A \). Thus each \( P_i \in L(H_n) \), and we can define another set of projections by \( O = \{P_1 \otimes I_B, \ldots, P_k \otimes I_B\} \), where \( I_B \) is the identity mapping in \( L(H_m) \), so each \( P_i \otimes I_B \in L(H_n \otimes H_m) \). Notice that \( \{I_B\} \) alone defines an observable of subsystem \( Q_B \), but this observable is trivial: there is only one possible value to be observed, and it will be seen with a probability of \( \text{Tr}(IT_B) = \text{Tr}(TB) = 1 \), when the state of system \( Q_B \) is \( T_B \).

Therefore, if the state of the whole system is \( T \), it is natural, because of Postulate 2, to define the state of system \( Q_A \) as a unit-trace, positive, self-adjoint mapping \( T_A \in L(H_n) \) which satisfies
\[
\text{Tr}(P_A T_A) = \text{Tr}((P_A \otimes I_B)T)
\]
for any projection \( P_A \in L(H_n) \). That is, state \( T_A \) should induce exactly the same probability to see the property that \( P_A \) refers to that state \( T \) induces for the property that \( P_A \otimes I_B \) refers to. Similarly, the state \( T_B \) should be the unit-trace, self-adjoint positive mapping in \( L(H_m) \) which satisfies
\[
\text{Tr}(P_BT_B) = \text{Tr}(I_A \otimes P_B T)
\]
for any projection \( P_B \in L(H_m) \). The following proposition guarantees the existence of subsystem states as defined earlier. For the proof, see [15].

**Proposition 24.** Let \( T \in L(H_n \otimes H_m) \) be a unit-trace, self-adjoint operator. There exists a unique self-adjoint, unit-trace positive operator \( T_A \in L(H_n) \) such that
\[
\text{Tr}(P_A T_A) = \text{Tr}((P_A \otimes I_B)T)
\]
for each projection \( P_A \in L(H_n) \).

We say that \( T_A \) is obtained from \( T \) by tracing over \( H_m \) and \( T_B \) from \( T \) by tracing over \( H_n \). We also write \( T_A = \text{Tr}_{H_m}(T) \) and \( T_B = \text{Tr}_{H_n}(T) \). Explicit expressions for subsystem states \( T_A \) and \( T_B \) can also be found, see [15], for instance. Here we will represent the
expressions in the case that the state $T$ is pure. See [15] for the proof of the following proposition.

**Proposition 25.** Let all notions be as before, $T = \ket{z}ra{z}$ be a pure state of a compound system with state space $H_n \otimes H_m$, and

$$z = \sum_{i=1}^{n} \sum_{j=1}^{m} z_{ij} a_i \otimes b_j = \sum_{i=1}^{n} \left( \sum_{j=1}^{m} z_{ij} a_i \right) \otimes b_j.$$ 

Then the mapping $T_A$ mentioned in Proposition 25 is given by

$$T_A = \sum_{j=1}^{m} \left( \sum_{i=1}^{n} z_{ij} a_i \right) \left( \sum_{i=1}^{n} z_{ij} a_i \right).$$ (34)

Notice that representation (34) is not necessarily the spectral representation of $T_A$.

**Example 26.** Consider a pure state $T = \ket{z}ra{z}$ of Example 22 determined by vector

$$z = \frac{1}{\sqrt{2}} \ket{00} + \frac{1}{\sqrt{2}} \ket{11}.$$ 

Expression (34) gives

$$T_A = \text{Tr}_{H_2}(T) = \frac{1}{\sqrt{2}} \left( \frac{1}{\sqrt{2}} \ket{0} \bra{0} + \frac{1}{\sqrt{2}} \ket{1} \bra{1} \right) = \frac{1}{2} \left( \ket{0} \bra{0} + \ket{1} \bra{1} \right),$$

as the state of the qubit $A$ (as well as the state of the qubit $B$: by symmetry, $T_A = T_B$).

An attempt to recover $T$ from $T_A$ and $T_B$ gives

$$T_A \otimes T_B = \frac{1}{4}(\ket{00}\bra{00} + \ket{01}\bra{01} + \ket{10}\bra{10} + \ket{11}\bra{11}),$$

which is a mixed state and different from $T$. But $T_A \otimes T_B$ is a state of a two-qubit system which also gives $T_A$ and $T_B$ as subsystem states. Therefore, the subsystem states alone are not enough to reconstruct the state of the whole system.

The following proposition, whose proof can be found in [15], offers another tool for expressing completely positive mappings (20).

**Proposition 27.** Let $Q_A$ be a quantum system with state space $H_n$. Let also $V : L(H_n) \rightarrow L(H_n)$ be a completely positive linear mapping such that $V(T)$ is a state whenever $T$ is. Then there exists a unitary mapping $U \in L(H_n \otimes H_{n^2})$ and a pure state $S = \ket{s} \bra{s} \in L(H_{n^2})$ such that for each state $T$,

$$V(T) = \text{Tr}_{H_{n^2}}(U(T \otimes S)U^*).$$ (35)

**Remark 28.** Notice that $W(T \otimes S) = U(T \otimes S)U^*$ determines a closed time evolution operator in the compound system consisting of $Q_A$ and another system with $n^2$ basis
Example 29. To represent a pure state of an \( n \)-qubit system, we use an \( n \)-fold tensor product \( H_2 \otimes \cdots \otimes H_2 \), which is isomorphic to \( 2^n \)-dimensional space \( H_{2^n} \). The elements of the computational basis can be naturally labelled by strings of length \( n \) over the binary alphabet: We take \( |x\rangle \{ x \in \{0,1\}^n \} \) as an orthonormal basis.

A general pure state of the system is then determined by

\[
\sum_{x \in \{0,1\}^n} c_x |x\rangle,
\]

where

\[
\sum_{x \in \{0,1\}^n} |c_x|^2 = 1.
\]

As usual, a closed time evolution mapping of the system can be expressed by using a unitary mapping in \( L(H_{2^n}) \). It is straightforward to see that if \( U_1 \) and \( U_2 \) are unitary, then \( U_1 \otimes U_2 \) also is. For instance, an \( n \)-fold tensor product of mapping (29)

\[
W_{2^n} = W_2 \otimes \cdots \otimes W_2
\]

determines a unitary mapping in \( L(H_{2^n}) \). To recover the effect of \( W_{2^n} \), we can simply rewrite Eqs. (25) and (26) as

\[
W_2 |x\rangle = \frac{1}{\sqrt{2}} (|0\rangle + (-1)^x |1\rangle)
\]

to see that

\[
W_{2^n} |x\rangle = \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle,
\]

where \( x \cdot y = x_1 y_1 + \cdots + x_n y_n \).

6. Devices for quantum computation

In a simplified manner, we can say that in order to introduce a device for quantum computation, one can just consider a classical computational device, especially which kind of system can be used to express all possible configurations of the device, and then just regard the system expressing the configurations as a quantum system instead of a classical one.

However, if we assume, as is usually done, that the time evolution of a quantum computing device should be closed, then the device is necessarily reversible. Therefore, it is not always so straightforward how to convey the concepts of traditional theory of computation into the domain of quantum computation. In this section, we will represent, without many details, three examples of quantum computing devices. For more details, we recommend to consult [12,15], for instance.
6.1. Quantum finite automata

A quantum finite automaton (QFA) consists of an alphabet $A$, state set $Q$, and the transition function $\delta$. Moreover, there is a fixed initial state $s$. A configuration of the system is simply a state $q \in Q$, so a representation of a configuration is based on a quantum system capable of representing $Q$. We fix $\{|q\rangle | q \in Q\}$ as a computational basis.

For any letter $a \in A$, and any state $q \in Q$, the transition function $\delta$ determines the time evolution of the state set

$$|q\rangle \mapsto \sum_{r \in Q} \delta(q,a,r)|r\rangle$$  \hspace{1cm} (36)

in such a way that the time evolution determined by (36) is a closed quantum system evolution, i.e., for each $a \in A$, $\delta$ defines a unitary mapping in the state space. Hence, mapping $\delta$ can be as well defined as a collection of $|A|$ unitary matrices.

The computation of QFA with an input word $w = a_1 \ldots a_k$ can then be defined as a consecutive application of time evolutions (36) with letters $a_1, \ldots, a_k$, starting at state $|s\rangle$. In order to introduce various computational behaviours, we can also fix accepting states and rejecting states, similarly as we can do for probabilistic finite automata. For more work on quantum finite automata, see [1,20].

6.2. Quantum Turing machines

A quantum Turing machine (QTM for short) was first introduced by David Deutsch in 1985. In his seminal paper [9] also gave a description of a universal quantum Turing machine, which is capable of simulating all other QTM’s, but in [9] he did not pay much attention to the simulation efficiency. The work of D. Deutsch was improved by Bernstein and Vazirani in 1997 [7], where the authors showed how to construct a universal QTM which can simulate any other QTM with polynomial efficiency.

Quantum Turing machine (with one tape), as introduced by Bernstein and Vazirani [7], looks like a straightforward generalization of a probabilistic Turing machine. It consists of a tape alphabet $A$, set of internal states $Q$, and of an amplitude transition function $\delta$.

The model is built on a quantum system capable of representing any configuration of an ordinary Turing machine. The configuration consists of

- the contents of the tape,
- the position of the read–write-head, and
- the internal state of the machine.

Thus, a configuration can be represented as $(w, q, i)$, where $w \in A^*$ is the word on the tape, $q \in Q$ is the internal state, and $i \in \mathbb{Z}$ is the number of the tape cell the machine is currently scanning. There are infinitely but only countably many different configurations, so the quantum system representing the configuration is infinite dimensional.

A configuration

$$(a_1 \ldots a_i \ldots a_k, q, i)$$  \hspace{1cm} (37)$$
can be represented by using $k$ systems each capable of representing $A$, one system capable of representing $Q$, and one system representing the integers. Configuration (37) is interpreted in such a way that the contents of the tape is word $a_1 \ldots a_l$, machine is in state $q_i$, and $a_i$ is the currently scanned symbol.

Locality now means that only the subsystem

$$(a_i, q, i)$$

of (37) is altered during a computational step. In connection with deterministic Turing machines, the transition function

$$\delta: Q \times A \to Q \times A \times \{-1, 0, 1\}$$

uniquely determines the behaviour of the machine: If $\delta(q, a_i) = (r, a'_i, d)$, then the machine performs transformation

$$(a_i, q, i) \mapsto (a'_i, r, i + d)$$

and leaves other components of (37) unchanged. In other words, $a_i$ is replaced with $a'_i$, state $q$ is replaced with state $r$, and the read–write-head moves one step to direction $d \in \{-1, 0, 1\}$.

A QTM works exactly in the same fashion, but now the transition $\delta$ is not necessarily determined uniquely, but there may be many potential actions that the machine can do, any such with an amplitude

$$\delta(q, a_i, r, a'_i, d).$$

Thus, a partial configuration (38) is transformed (and all other components of system (37) are left unchanged) as

$$|a_i\rangle |q\rangle |i\rangle \mapsto \sum_{r, a'_i, d} \delta(q, a_i, r, a'_i, d) |a'_i\rangle |r\rangle |i + d\rangle,$$

(39)

in such a way, that the entire time evolution of the system remains closed, i.e., $\delta$ determines a unitary mapping in the infinite-dimensional quantum system representing the configurations.

It turns out that the unitarity of the time evolution determined by $\delta$ can be seen already locally, see [7,14,21].

There are at least three problematic points in the concept of QTM. First, the requirement of a closed quantum time evolution implies that the time evolution is also reversible, but, on the other hand, the transition function of an ordinary Turing machine can be irreversible, as well. This means, that there is no way to straightforwardly simulate an arbitrary Turing machine by a QTM. This difficulty can be won by first simulating an arbitrary Turing machine by a reversible Turing machine, which is always possible by a result of Bennett [4]. (In article [19], Lecerf had already demonstrated that it is possible to simulate any Turing machine by using an irreversible Turing machine. The simulation time in Lecerf’s construction grows quadratically, whereas Bennett’s construction gives only linear time growth.)
Second, allowing arbitrary complex numbers as amplitudes (the values of $\delta$) lead to the problematic extreme cases familiar from probabilistic computation, also: one can imagine, that, for instance, the decimal expansion of some probability encodes some uncomputable language $L$. Therefore, such a machine could be, in principle, used to decide, with arbitrarily high correctness probability, whether a given word $w$ is in $L$. On the other hand, a physical realization of such a machine is highly questionable. A way out of this problem is, for instance, that one requires the amplitudes to be expressible by rational numbers, or more generally, by numbers whose digits can be “rapidly” computed by using a deterministic Turing machine.

The third problem is that when should we consider a QTM had finished its computation? In an ordinary Turing machine some states are considered as halting states, and this concept works as well for a probabilistic Turing machine: potential computations on an input $w$ may take different number of steps. For a QTM this does not fit so well any more, mainly because the superposition of potential configurations reached so far does not admit ignorance interpretation. So far, there is no uniform way to circumvent this difficulty, but for machines realizing some algorithm whose computation time is known in advance, the problem can simply be solved by letting the machine run the known computation time, and then to observe the outcome.

6.3. Quantum circuits

Perhaps the most convenient way to express quantum algorithms is offered by quantum circuits, which are quite straightforward analogues to Boolean circuits. Boolean circuits are usually represented as acyclic directed graphs computing a function $f : \{0,1\}^n \rightarrow \{0,1\}$. Briefly, a Boolean circuit can be represented as follows (for details, see [22]): in the circuit, there are $n$ input nodes which represent the input word $w \in \{0,1\}^n$. There are also inner nodes called gates labelled with logical and (in symbols, $\wedge$), logical or ($\vee$), and logical not ($\neg$), which compute elementary Boolean functions\(^{18}\) from the bit values given by the preceding nodes, and finally, one node (the sink) is give a special status as the output node, giving the value of function $f$ computed by the circuit. The complexity of a Boolean circuit is simply defined to be the number of the gates in the circuit.

Classical $\wedge$ and $\vee$-gates have two input bits but only one output bit. Especially, gates $\wedge$ and $\vee$ are not reversible: the input bits cannot be recovered from the output in general. On the other hand, it turns out that, by introducing some ancilla bits (also called dummy bits) having constant input values one can replace the $\wedge$ and $\vee$-gates with reversible gates having equally many input and output bits [15].

So called Toffoli gate is defined on three bits by

$$T(b_1, b_2, b_3) = (b_1, b_2, b_1 \cdot b_2 + b_3),$$

\(^{18}\)The gates $\wedge$, $\vee$, and $\neg$ can be replaced by any other gate set capable of representing all Boolean functions. A set of gates capable of representing all Boolean functions is called universal.
where $b_1 \cdot b_2 + b_3$ is computed modulo 2. Thus, a Toffoli gate leaves bits $b_1$ and $b_2$ untouched, and flips the value of $b_3$ if $b_1 = b_2 = 1$. It turns out, that Toffoli gate alone is capable of simulating $\land$, $\lor$, and $\lnot$-gates, when suitable ancilla bits with constant values are introduced [15]. Thus, one can build any Boolean circuit by using only Toffoli gates and some constant ancilla bits. A gate with this property is called universal.

Let $B = \{B_0, B_1, B_2, \ldots, \}$ be a collection of Boolean circuits such that $B_n$ has $n$ input nodes, i.e., $B_n$ represents a function $f_n : \{0, 1\}^n \rightarrow \{0, 1\}$. Since $\{0, 1\}^* = \{0, 1\}^0 \cup \{0, 1\}^1 \cup \{0, 1\}^2 \ldots$, we can regard collection $B$ as a representation of a function $f : \{0, 1\}^* \rightarrow \{0, 1\}$. In fact, it can be shown that each function $f : \{0, 1\}^* \rightarrow \{0, 1\}$ has a representation as an infinite collection of Boolean functions [15]. On the other hand, the collection $B$ does not offer a method for computing function $f$, unless we know how to produce a representation of $B_n$, when $n$ is given.

A quantum gate on $k$ qubits is simply a closed time evolution operator on Hilbert space $H_{2^k} = H_2 \otimes \cdots \otimes H_2$ representing $k$ qubits. Thus a quantum gate on $n$ qubits can be represented as $2^n \times 2^n$ unitary matrix. Note that a quantum gate as defined here has equally many input and output qubits, and is reversible (the output always defines the input). In fact, any reversible classical gate can be seen as a special case of a quantum gate.

A quantum circuit $Q$\textsuperscript{19} with $n$ input qubits consists of a Hilbert space $H_{2^n} = H_2^{(1)} \otimes \cdots \otimes H_2^{(n)}$\textsuperscript{20} representing $n$ qubits, of a finite set $G$ of quantum gates, and a finite sequence $V_1, \ldots, V_i$ of closed quantum time evolutions on these $n$ qubits, i.e., each $V_i$ is a unitary mapping in $H_{2^n}$. In addition to that, each $V_i$ must be expressible in terms of some quantum gate $G_i \in G$, meaning that if $G_i$ has $k$ input qubits, there is a set $\{i_1, \ldots, i_k\} \subseteq \{1, 2, \ldots, n\}$ of qubits such that $Q_i$ restricted to subspace $H_{2^{(i_1)}}(i) \otimes \cdots \otimes H_{2^{(i_k)}}(i)$ is exactly $G_i$ and $V_i$ acts as the identity mapping on the other qubits. In other words, each $V_i$ is essentially a quantum gate $G_i \in G$ acting on some qubits.\textsuperscript{21}

The complexity of a quantum circuit $Q$ (with respect to the set $G$) is the number of quantum gates in the circuit. Notice that, as a sequence of unitary mappings on $n$ qubit, a quantum circuit itself can be seen as a unitary mapping on $n$ qubits, that is, as a quantum gate on $n$ qubits. Thus, there is no fundamental differences between quantum gates and circuits, but the difference is contextual: notion “quantum circuit” refers to a unitary mapping which can be represented as a sequence of simpler unitary mappings chosen from a finite set.\textsuperscript{22} However, the contextual difference between quantum gates and quantum circuits becomes more important when considering an infinite family $\{Q_0, Q_1, Q_2, \ldots, \}$ such that $Q_i$ has $i$ input qubits and any $Q_i$ must be built using only a finite collection $G$ of quantum gates. For a comparison of the computational powers of quantum circuits and quantum Turing machines, see [25].

\textsuperscript{19}Also called quantum network.

\textsuperscript{20}We use superscript notation $H_2^{(i)}$ in order to address to the $i$th qubit.

\textsuperscript{21}Usually it is also allowed that each $V_i$ may consist of several quantum gates $G_{i_1}, \ldots, G_{i_l}$ which act on disjoint sets of qubits. This leads essentially to the same concept of a quantum circuits as we have here.

\textsuperscript{22}The same holds for Boolean circuits: a circuit consists of gates which compute some simple Boolean functions, but the circuit itself also computes a Boolean function.
7. The power of quantum computing

7.1. The discrete Fourier transform

For the definitions and further explanations of the notions in this section, we refer to [15].

We will first explain the idea behind a (discrete) Fourier transform. If \( G \) is a finite abelian group having \( N \) elements, then all the functions \( f : G \to \mathbb{C} \) form an \( N \)-dimensional complex vector space (addition and scalar multiplication is defined pointwise), which is isomorphic to \( H_N \), hence we will denote that vector space by \( H_N \) hereafter. This vector space has the so-called natural basis \( B = \{ T_g \mid g \in G \} \), where \( T_g \) is defined as \( T_g(g) = 1 \) and \( T_g(g') = 0 \) whenever \( g' \neq g \). It easy to verify that the basis \( B \) is orthonormal with respect to the standard inner product defined by

\[
\langle f \mid h \rangle = \sum_{g \in G} f(g)^* h(g).
\]

Name “natural basis” can be justified as follows: clearly each function \( f \in H_N \) as a unique representation as

\[
f = \sum_{g \in G} f(g) T_g
\]

so the coordinates of the function \( f \) with respect to the natural basis are indeed the values of \( f \).

On the other hand, it turns out that an abelian group \( G \) has \( N = |G| \) characters, \(^{23}\) which form an orthogonal basis of \( H_N \). By using a suitable normalization (multiplying the characters with constant \( 1/\sqrt{N} \)), we can get another orthonormal basis for space \( H_N \). Let us denote that basis by \( C = \{ B_g \mid g \in G \} \). Any \( f \in H_N \) has then also a unique representation

\[
f = \sum_{g \in G} \hat{f}(g) B_g
\]

for some coefficients \( \hat{f}(g) \in \mathbb{C} \). Then for any function \( f \in H_N \), the coefficients \( \hat{f}(g) \) in representation (40) also define a function in \( H_N \) by \( g \mapsto \hat{f}(g) \). Function \( \hat{f} \) is called the Fourier transform of \( f \). By using the orthonormality of functions \( B_g \) we can easily extract any coefficient \( \hat{f}(g) \) in representation (40)

\[
\hat{f}(g) = \langle B_g \mid f \rangle = \sum_{h \in G} B_g(h)^* f(h).
\]

In what follows, we will concentrate on a special group. We can equip the binary alphabet \( \{0, 1\} \) with a commutative addition defined by \( 0 + 0 = 1 + 1 = 0 \), and \( 0 + 1 = 1 + 0 \) to create the abelian group structure for the binary alphabet. We denote the

\(^{23}\text{The characters } \chi \text{ of a finite abelian group } G \text{ are mappings } \chi : G \to \mathbb{C} \text{ such that } \chi(g_1 + g_2) = \chi(g_1) \chi(g_2) \text{ whenever } g_1, g_2 \in G.\)
outcoming binary group by $C_2$. Defining the addition componentwise, an $n$-fold Cartesian product $C_2^n$ of the binary group can also be given an abelian group structure. Clearly $N = |C_2^n| = 2^n$.

It turns out that for each $y = (y_1, \ldots, y_n) \in C_2^n$, mapping $\chi_y$ defined by $\chi_y(x) = (-1)^{y_1 x_1 + \cdots + y_n x_n}$ is a character of $C_2^n$, and that all of the characters of $C_2^n$ are of that form [15]. Consequently, so-called Walsh functions $W_y = (1/\sqrt{2^n}) \chi_y$, where $y \in C_2^n$, form an orthonormal basis of $H_N$. Clearly, the values of the Walsh functions are always in set $\{-1/\sqrt{2^n}, 1/\sqrt{2^n}\}$ (especially the values are real) and the Walsh functions are symmetric with respect to index $y$ and argument $x$: $W_y(x) = W_x(y)$.

The Fourier transform in $C_2^n$ thus takes the following form: if $f \in H_N$ is a function, then

$$\hat{f}(y) = \langle W_y | f \rangle = \sum_{x \in C_2^n} W_y(x)^* f(x) = \frac{1}{\sqrt{2^n}} \sum_{x \in C_2^n} f(x) \chi_y(x).$$ (41)

7.2. Quantum Fourier transform

By computing the Fourier Transform in $C_2^n$ we understand the following: given the values of a function $f \in H_N$ (2$^n$ values), output the values $\hat{f}$ (also 2$^n$ values). By directly utilizing (41), we should use 2$^n$ multiplications and 2$^n$ − 1 additions for computing a single value of $\hat{f}(y)$, thus resulting in $\Theta(2^{2n})$ arithmetic operations altogether to compute $\hat{f}$.

An improvement to complexity $\Theta(2^{2n})$ can be obtained by representing (41) as

$$\hat{f}(y) = \frac{1}{\sqrt{2^n}} \sum_{x' \in C_2^{n-1}} f(0x') \chi_{x'}(y').$$

(42)

In the above representation, $x'$ and $y'$ are obtained from $x$ and $y$ by deleting the first coordinate. Representation (42) essentially states that the Fourier transform in $C_2^n$ can be computed by first computing two Fourier transforms in $C_2^{n-1}$, and then combining the results. The number of arithmetic operations required by computing the Fourier transform by recursively applying the decomposition (42) is $\Theta(n2^n)$, an improvement having practical significance over the naive method. The recursive method thus obtained for computing the Fourier transform is called fast Fourier transform, FFT for short.

By quantum Fourier transform we understand transforming a superposition

$$\sum_{x \in \{0,1\}^n} f(x) |x \rangle$$

(43)

into

$$\sum_{y \in \{0,1\}^n} \hat{f}(y) |y \rangle.$$ (44)
Notice that a vector (43) describing a pure state is of unit length, but because of the Parseval’s identity [15]

$$\sum_{x \in \{0,1\}^n} |f(x)|^2 = \sum_{y \in \{0,1\}^n} |\hat{f}(y)|^2$$

also (44) has unit length. By Example 29, we have that

$$W_2 \otimes \cdots \otimes W_2 \sum_{x \in \{0,1\}^n} f(x)|x\rangle = \sum_{y \in \{0,1\}^n} \hat{f}(y)|y\rangle.$$  \hspace{1cm} (45)

Eq. (45) reveals quite an interesting fact: in order to compute a quantum Fourier transform \( f \rightarrow \hat{f} \), it is sufficient to use only \( n \) qubit operations (cf. classical complexity \( O(n^2) \)). This is however related to the exponential packing density of quantum systems itself: for a description of a (pure) state of a quantum system consisting of \( n \) qubits, we need \( 2^n \) amplitudes to describe the state.

In the above example we showed how to compute quantum Fourier transform in \( C_2^n \) by using only \( n \) qubit operations. When regarding the bits strings in \( \{0,1\}^n \) as elements of \( \mathbb{Z}_{2^n} \), one can understand to coefficients of a superposition (43) as a function \( f: \mathbb{Z}_{2^n} \rightarrow \mathbb{C} \), and it turns out that the corresponding quantum Fourier transform can be computed by using \( n^3 \) quantum bit operations. In his article which appeared in 1994 [23] Peter W. Shor has demonstrated how to find, with a high probability, a factorization of given number \( N \) only by using \( O(\log^3 N) \) elementary quantum operations (quantum gates), and the very core of Shor’s factorization algorithm is indeed the quantum Fourier transform in \( \mathbb{Z}_{2^n} \). Shor’s quantum algorithm for factorizing numbers offers a substantial improvement over the known traditional techniques: the complexity of Shor’s algorithm is polynomial with respect to the representation size of \( N \) (proportional to \( \log N \)), whereas all known classical algorithms (including the probabilistic ones) require a superpolynomial (with respect to \( \log N \)) amount for finding the factors of a given number. For the details of Shor’s algorithm, see [15].

The work of Peter W. Shor can be seen as a starting point of more intensive research on quantum computation. The problem of factoring is of great theoretical but also of practical interest: the security of a famous RSA cryptosystem is based on the assumption that the factorizing of integers will remain as an untractable problem, but Shor demonstrated that this is not the case, if a large-scale quantum computer can be built some day.

After Shor’s work, it is natural to ask whether there are some other that can be rapidly solved by using a quantum computer. In particular, we may ask whether there is polynomial-time solution (when using a quantum computer) to all \( \text{NP} \)-problems? By now, it seems that this is not the case, and that question we shall discuss in the following sections.

---

24 Peter W. Shor won the Nevanlinna Prize 1998 for the work on quantum computation.
7.3. Grover search

Mainly by using the fact that a pure quantum state is indeed different from a probability distribution (in a time evolution, we can utilize the interference phenomena), it is possible to devise quantum algorithms that work essentially faster than any classical ones. The following problem is substantial in computer science: given a function $f : \{0,1\}^n \to \{0,1\}$, decide, if there exists a $x \in \{0,1\}^n$ such that $f(x) = 1$ or not. Clearly the problem can be solved by exhausting: compute all the values of $f(x)$, and then give the decision. But the exhaustive search requires $2^n$ values to be computed, which will be too time consuming for a large $n$.

Lov Grover has devised an algorithm for quantum computers [11], which operates on superpositions to compute all values of function $f$ simultaneously,\(^{25}\) and then using the interference phenomena to cancel out the “unwanted” configurations. For a function $f : \{0,1\}^n \to \{0,1\}$, Grover’s algorithm makes the decision mentioned above (with a high probability) by using only $O(\sqrt{2^n})$ computations of values of $f$. For details, see [15].

7.4. Restrictions

Grover’s result mentioned in the above section essentially states that a property “does there exist an element $x \in \{0,1\}^n$ such that $f(x) = 1$ for some (given) function $f : \{0,1\}^n \to \{0,1\}$” can be solved (with a high probability) by using a quantum computer which computes only $O(\sqrt{2^n})$ times the value of $f$. Thus, by using a quantum computer, we can reach a quadratic speed-up over the traditional ones. Naturally, we can ask if the above problem could have even a faster solution than that one provided by Grover’s algorithm.

Based on polynomial representations of Boolean functions, it was shown in article [2] how to obtain a general lower bound for the number of how many times the value of a function $f : \{0,1\}^n \to \{0,1\}$ must be evaluated in order to get some particular information about the values (e.g., whether there exists some $x \in \{0,1\}^n$ such that $f(x) = 1$). It turns out (see [2,15]) that for unstructured functions $f$ (meaning that the algorithm for computing function $f$ is not given, but the values of $f$ can be obtained by just calling the subroutine (oracle) computing $f$), the decision whether there is such a $x \in \{0,1\}^n$ that $f(x) = 1$, cannot be made (with a high probability), unless $\Theta(\sqrt{2^n})$ calls for function $f$ are made. See also [6].

The result mentioned above also implies, that for (unstructured functions), the algorithm devised by L. Grover works optimally. On the other hand, it implies also that it is very hard to find any polynomial-time solution to $\text{NP}$-problems even by using a quantum computer.

\(^{25}\) Recall that this property can, in principle, used also in probabilistic algorithms, when a superposition is simply seen as a distribution over the potential configurations of the computational machine.
References