Universitat Oberta de Catalunya

UNIVERSITAT ROVIRA i VIRGILI

Máster en Ingeniería Computacional y Matemática

Trabajo Fin de Máster

Introduction to post-quantum cryptography

Student: Ángela Capel Cuevas

Advisor: Dr. Oriol Farràs Ventura (Universitat Rovira i Virgili)

Date of delivery: September 2018

Signature of the advisor authorizing the final delivery of the TFM:

Index Card of the Final Master Project

Title of the FMP:	Introduction to post-quantum cryptography
Name of the author:	Ángela Capel Cuevas
Name of the tutor:	Montserrat Fernández Barta
Name of the PRA:	Oriol Farràs Ventura
Date of delivery:	September 2018
Degree:	Master's Degree in Computational Engineering and Mathematics
Area of Final Work:	Cryptography
Language of the Work:	English
Keywords:	Cryptography, post-quantum, algorithms

Summary of the Work (maximum 250 words): With the purpose, context of application, methodology, results and conclusions of the work.

The main purpose of this project is to give an introduction to post-quantum cryptography. For that, we have decided to split the work into three well-differentiated parts.

In the first one, we provide a general introduction to quantum information theory and some basic concepts that are necessary to understand quantum algorithms. Indeed, we present concepts such as qubits and superposition, the notion of measurements and the postulates of quantum mechanics in two different (and dual) settings, among other things.

In the second part, we present a summary of the main results that exist in quantum cryptography. For that, we give before some preliminaries in both classical and quantum circuits, and we also introduce some situations and interesting phenomenon that appear in quantum mechanics. Afterwards, we present the first important quantum algorithms: Deutsch-Jozsa's and Simon's algorithms. From them and using the quantum Fourier transform, we are able to present the most important quantum algorithms so far: Shor's and Grover's algorithms.

Finally, we conclude our work with a chapter concerning the emergence of post-quantum cryptography. In the first part, we notice how the previous quantum algorithms can be used to attack security systems, and, thus, how this motivates the appearance of new secure schemes. In the last part of this chapter, we give a short introduction to post-quantum cryptography, mentioning the different classes of systems that are under study in this field and their current development.

Abstract (in English, 250 words or less):

In this project, we give an introduction to post-quantum cryptography, which refers to cryptographic algorithms (usually public-key algorithms) that are thought to be secure against an attack by a quantum computer.

This new field has appeared and evolved in the last few years due to two main reasons: The first one, most popular public-key algorithms can be efficiently broken by a sufficiently strong hypothetical quantum computer; and the second one, the hypothetical quantum computer seems to be likely to appear in the next few years.

The main purpose of this work is to give a well-motivated and self-contained introduction to post-quantum cryptography. To show the necessity of the development of this field, we present some attacks based on some well-known quantum algorithms, which we previously introduce in much detail, and to make the text self-contained, we introduce all the basic concepts that are necessary to understand these quantum algorithms, and, in general, some basic concepts and situations that appear in quantum information theory and arise from quantum mechanics.

Contents

0.1 Context and justification of the Project . . . 11
0.2 Objectives of the Project . . . 11
0.3 Approach and method followed . . . 12
0.4 Work planning . . . 12
0.5 Brief summary of the products obtained . . . 13
0.6 Brief description of the other chapters of the memory . . . 13

1 Preliminaries **15**
1.1 Qubits and superposition . . . 15
1.2 Measurements . . . 17
1.2.1 Distinguishability . . . 19
1.2.2 Projective Measurements . . . 20
1.2.3 POVM Measurements . . . 20
1.3 Unitary evolution . . . 21
1.4 Density operators . . . 22
1.5 Postulates of quantum mechanics . . . 23
1.5.1 Heisenberg picture . . . 23
1.5.2 Schrödinger picture . . . 26

2 Quantum algorithms **29**
2.1 Preliminaries . . . 30
2.1.1 Classical circuits . . . 30
2.1.2 Quantum circuits . . . 31
2.1.3 No-cloning theorem . . . 34
2.1.4 Quantum teleportation . . . 36
2.1.5 Superdense coding . . . 38
2.1.6 Universality of quantum gates . . . 38
2.2 Introduction to quantum algorithms . . . 39
2.3 Deutsch-Jozsa's algorithm . . . 41
2.3.1 2-bit functions . . . 42
2.3.2 n-bit functions . . . 44
2.3.3 Bernstein-Vazirani problem . . . 46
2.4 Simon's algorithm . . . 47
2.5 Fourier transform . . . 48
2.6 Shor's quantum factoring algorithm . . . 50
2.6.1 Reduction of factoring to order-finding . . . 51
2.6.2 The order-finding problem . . . 53
2.7 Grover's algorithm . . . 60
2.7.1 Algorithm . . . 60
2.7.2 Geometrical proof of the algorithm . . . 62

2.7.3 Amplitude amplification 64

3 Introduction to post-quantum cryptography 65
3.1 Quantum attacks 65
3.1.1 A quantum attack based on the FX construction based on Grover's and Simon's algorithms 67
3.1.2 Hidden Shift Quantum Cryptanalysis 70
3.2 A brief overview of post-quantum cryptography 71
3.2.1 Code-based encryption 74
3.2.2 Lattice-based encryption 75
3.2.3 Lattice-based signatures 77
3.2.4 Multivariate-quadratic-equation signatures 78
3.2.5 Hash-based signatures 79
3.2.6 Isogeny-based cryptography 80

4 Conclusions 83
4.1 General conclusions and attainment of the aims initially proposed 83
4.2 Follow-up of the planning and methodology 83
4.3 Future lines of work 84

List of Figures

1 Time spent on each task. . . . 13

2.1 Some classical gates for two qubits. . . . 30
2.2 Circuit used to turn $|00\rangle$ into $\frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$. . . . 34
2.3 Classical way to clone a bit. . . . 35
2.4 f is given as an oracle. . . . 42
2.5 f is given as an oracle. . . . 42
2.6 Quantum oracle. . . . 43
2.7 2-bit Deutsch-Jozsa model. . . . 43
2.8 n-bit Deutsch-Jozsa model. . . . 44
2.9 Another scheme for the n-bit Deutsch-Jozsa model. . . . 45
2.10 Quantum circuit to compute the quantum Fourier transform. . . . 50
2.11 β makes a small angle with the real line. . . . 57
2.12 At least r values of kr lie in the range $[Q - r/2, r/2]$. . . . 58
2.13 At least $r/2$ values of kr lie in the range $[Q - r/2, r/2]$. . . . 58
2.14 Graphical representation of Grover's algorithm. . . . 61
2.15 First iteration of the third step of Grover's algorithm. In the first picture, it starts with $|U\rangle$; in the second one, it reflects through $|B\rangle$ to get $O_{x,\pm}|U\rangle$; and in the last one, it reflects through $|U\rangle$ to get $\mathcal{G}|U\rangle$. . . . 63

3.1 Even-Mansour construction. . . . 66
3.2 FX construction. . . . 67
3.3 Examples of some cryptographic systems and their conjectured security levels. 73
3.4 Merkle tree with public key Y_{15} to sign eight messages. . . . 80
3.5 Qualitative overview of the described post-quantum systems. . . . 81

Introduction

0.1 Context and justification of the Project

In this work, we give an introduction to the field of post-quantum cryptography. Throughout the whole text, we intend to give a proper motivation to this field, as well as a brief introduction to all the concepts that are necessary to understand the development of the work.

In the chapter of Quantum Algorithms (Chapter 2), among many other things, we present Shor's algorithm (Section 2.6), which is one of the most important examples that quantum computers would have a great impact on the security of many cryptographic scheme. Indeed, this algorithm, used on a quantum computer, would allow to factor numbers and compute discrete logarithms inside abelian groups in polynomial time. Since many public key schemes nowadays are built upon the assumption that an enormous number cannot be factorized efficiently, the existence of quantum computer would be an important problem for the security of almost all our digital communications.

Post-quantum cryptography is a new line of research that has appeared due to this situation, and its main objective is to develop new cryptographic schemes that would resist attacks from quantum computers (and algorithms implemented on them). Due to the appearance of the first prototypes of quantum computers of a small number of qubits and the possibility of (hopefully) commercializing functional quantum computers in the next decades, the NIST has announced a competition to promote fundamental research on this new field by looking for an standard (or several) quantum-resistant public-key cryptographic algorithms. Indeed, the development of these new cryptographic schemes is a long-time process, and that is the reason to start the development of the field soon enough, even though quantum computers are still far from becoming an actual threaten.

With this project, we aim to give an introduction to this field of post-quantum cryptography, mentioning the main classes of systems under study in this field, after presenting a couple of works where we can see how quantum cryptography could actually attack public-key security. These works are based on some quantum algorithms that we will have defined in previous chapters.

0.2 Objectives of the Project

The main objective of this work is to give a well-motivated and self-contained introduction to post-quantum cryptography. For that, before presenting the field of post-quantum cryptography itself, we need to introduce some quantum attacks to motivate its necessity, as well as the main basic concepts to understand the background in quantum algorithms.

More specifically, the main objectives of the project are the following:

- Introduce some basic concepts of quantum information theory (such as qubits and measurements), and quantum mechanics (such as its postulates).

- Give a brief introduction to classical and quantum circuits.
- Show the evolution of quantum algorithms in the last years, presenting the evolution from the first algorithms of Deutsch-Jozsa and Simon, to the well-known algorithms of Shor and Grover, respectively.
- Use these last algorithms to construct quantum attacks and, hence, motivate the necessity of post-quantum cryptography.
- Give a brief introduction to post-quantum cryptography, mentioning the most important classes of systems that are under study in this field.

0.3 Approach and method followed

In this project, we have used a direct strategy: We have started from the most basic concepts of quantum information theory, and quantum mechanics, studying them and some of their properties, to move afterwards to the introduction and development of some of the most well-known quantum algorithms, to state them, and quantum attacks built upon them, as the motivation for the emergence of post-quantum cryptography.

We remark that the main difference between our project and some other texts in either quantum or post-quantum cryptography is essentially the focus. In most of the texts in post-quantum cryptography, the authors usually focus specifically on it, with the different possible classes of systems that are currently under development to get quantum-computer-attacks resistant systems and the future perspective of this field. However, we considered that it was more interesting to actually develop the motivation for the emergence of this field, instead of just mention the fact that quantum computers break RSA and some other public key schemes.

Moreover, we wanted to present a self-contained text. Since the motivation for post-quantum cryptography enforces us to introduce some quantum algorithms, the self-contained aspect imposes the condition of including a whole chapter with some basic concepts on quantum information theory and quantum mechanics. This method makes it possible to read the whole manuscript, understanding most of the concepts therein, with no need of a strong background on the field.

0.4 Work planning

As we have mentioned in the previous section, the approach followed has been a direct one. Hence, the path followed to read the background material and write the manuscript has been a straight line, which we have devided into pieces to split the amount of time devoted to each part. This construction can be seen in Figure 1 below.

In this figure, we can see that most of the time has been spent on the parts of Quantum Algorithms and Introduction to Post-Quantum Cryptography (as well as the part of reading bibliography), as we could have expected. Indeed, these parts are the hardest ones that appear on this work (and also the largest), and more time has been necessary to understand properly the concepts and schemes associated to them.

The material that we have used in the ellaboration of this project is mainly formed of several textbooks and papers on topics related to quantum algorithms, post-quantum cryptography, or related topics. A complete list of these texts can be consulted at the end of the project.

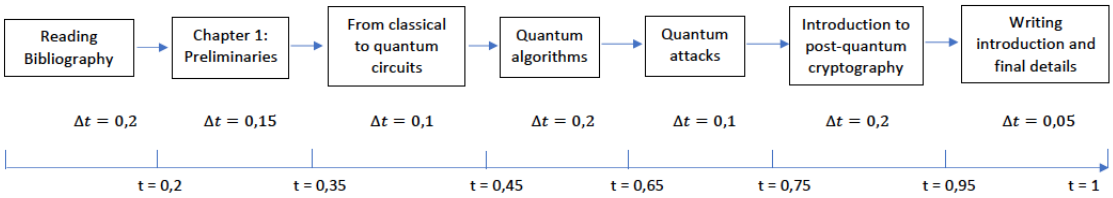

Figure 1: Time spent on each task.

0.5 Brief summary of the products obtained

In this work, we have focused on a theoretical exposition of a certain topic of interest. Hence, besides from the current memory of the work, no other product has been obtained.

0.6 Brief description of the other chapters of the memory

The main purpose of this project is to give an introduction to post-quantum cryptography, motivated by some quantum attacks that are constructed using previous quantum algorithms. For that, we have decided to split the work into three well-differentiated important chapters.

In the first one, we give a general introduction to quantum information theory and provide some basic concepts of quantum mechanics that are necessary to understand quantum algorithms. Indeed, we present concepts such as qubits and superposition, the notion of measurements and density operators, as well as the postulates of quantum mechanics in two different (and dual) settings, among other things.

In the second part, we present a summary of the main results that exist in quantum cryptography. For that, we give before some preliminaries in both classical and quantum circuits, and we also introduce some situations and interesting phenomenon that appear in quantum mechanics, such as quantum teleportation and the no-cloning theorem. Afterwards, we present the first important quantum algorithms: Deutsch-Jozsa's and Simon's algorithms, as well as a slight modification of Deutsch-Jozsa's algorithm known as Bernstein-Vazirani's algorithm. From them, and after introducing the quantum Fourier transform, we are able to present the most important quantum algorithms so far: Shor's and Grover's algorithms.

The third chapter concerns the emergence of post-quantum cryptography. In the first part, we remark how the previous quantum algorithms can be used to attack security systems, and, thus, how this motivates the emergence of new secure schemes. In the last part of this chapter, we give a short introduction to post-quantum cryptography, mentioning the different classes of systems that are under study in this field and their current development.

Finally, we conclude our work with some remarks and conclusions.

Chapter 1

Preliminaries

In this chapter, we are going to provide a brief introduction to quantum mechanics and its formalism, from a mathematical perspective. The concepts that we will introduce below are essential for the postulates that we will present in the following sections.

Throughout the whole chapter, we will denote the n-dimensional complex Hilbert space by $\mathbb{C}^n$, and we will denote vectors of this space in the *ket* notation, i.e., $|\varphi\rangle \in \mathbb{C}^n$ [1]. Given an element $|\varphi\rangle$ in the n-dimensional Hilbert space, its corresponding dual vector will be denoted in the *bra* notation by $\langle\varphi|$, and the action of the bra $\langle\varphi|$ on the ket $|\varphi\rangle$ is expressed by the standard inner product $\langle\varphi|\varphi\rangle$, whereas the element $|\varphi\rangle\langle\varphi|$ provides a rank-one operator.

Moreover, in general, every rank-one operator can defined in $|\varphi\rangle\langle\psi| : \mathbb{C}^n \to \mathbb{C}^n$ as

$$|\varphi\rangle\langle\psi|(|\xi\rangle) = \langle\psi|\xi\rangle|\varphi\rangle \qquad \text{for every } |\xi\rangle \in \mathbb{C}^n.$$

All the concepts and results that will be introduced in this chapter can be found in some basic texts of quantum information theory, such as the courses [4], [5] and [21], and the books [6] and [3], although one of the most fundamental texts in this field is [2]. We refer the reader to any of those texts for further knowledge on the topic.

1.1 Qubits and superposition

The simplest quantum mechanical system is the *qubit*. In fact the qubit plays the same role in quantum information theory as the *bit* in classical information theory, which can be 0 or 1; it is the basic unit of information. Indeed, it is a *superposition* of 0 and 1. Formally, it is the system whose associated vector space is a two dimensional Hilbert space.

We will use notation

$$\{|0\rangle, |1\rangle\},$$

for the canonical basis of $\mathbb{C}^2$, where we are denoting

$$|0\rangle \equiv \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |1\rangle \equiv \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

This basis is usually called *computational basis*. Then, while a classical bit can be in the state 0 or in the state 1, an arbitrary state for a qubit is a vector

[1] When it is not necessary to express explicitly the dimension of a finite-dimensional Hilbert space, we will just denote it by $\mathcal{H}$.

$$|\varphi\rangle = a\,|0\rangle + b\,|1\rangle, \text{ with } a, b \in \mathbb{C} \text{ and } |a|^2 + |b|^2 = 1.$$

When $a \neq 0 \neq b$, we say that the state is in *superposition* of the situations $|0\rangle$ and $|1\rangle$. Notice that this fact leads to the essential difference between the possible states of a bit, which are just two, 0 or 1, and the possible states of a qubit, which, in principle, are infinite. This new situation allows us to perform new protocols for quantum information processing. Indeed, this principle constitutes the basis of the theoretical *quantum computer*, as we will see in the following chapter (Chapter 2) in much more detail. Here let us just briefly mention the main idea behind quantum computing in a nutshell:

- If we consider one bit, we can perform **1** operation at a time, whereas we can perform **2** operations simultaneously with one qubit. This is due to the superposition phenomenon mentioned above, since now an arbitrary state is of the form

  $$|\varphi\rangle = a\,|0\rangle + b\,|1\rangle, \text{ with } a, b \in \mathbb{C} \text{ and } |a|^2 + |b|^2 = 1,$$

  where one can see two basic bits (thus, operations) being performed at the same time.

- If we now have two bits, we can perform **2** operations at a time, while, if we consider two qubits, **4** operations can be performed at the same time (we will see that when we consider a superposition of the four elements of the Bell basis).

- In general, with n qubits, one can perform **n** operations simultaneously (one per each bit), whereas with n qubits one can perform 2^n operations at the same time.

Hence, theoretically, a quantum computer can give an exponential improvement to the amount of operations performed in parallel compared to a classical computer.

Let us go back now to the definition and basic properties of qubits. It is important to remark that, even though a given qubit can be in any superposition state $a\,|0\rangle + b\,|1\rangle$, if we *measure* the state of such a qubit, we will obtain either the value $|0\rangle$ or $|1\rangle$ for the state of the qubit (these states can be seen as classical states), with certain probabilities. Hence, we cannot "see" the superposition phenomenon, although we are able to use it, as we will see later and this chapter and with greater applications in the following one.

Apart from the states $|0\rangle$ and $|1\rangle$, the following two states will be of use in the following sections:

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

A single qubit lives in $\mathbb{C}^2$. However, one can consider systems of more qubit to have richer spaces. For example, if we consider 2 qubits, the 2-qubit system that we get has four elements in a possible basis:

$$\{|0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle\},$$

where, in each case, the qubit in the left part denotes the first qubit (and associated to the first system), and the right one denotes the second qubit. If one considers $|0\rangle \otimes |1\rangle$, for instance, this element can be also expressed by $|0\rangle\,|1\rangle$ or $|01\rangle$, and the structure of tensor product implies that in $\mathbb{C}^4$ can be written as:

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

More generally, as mentioned previously, if one considers a system of n qubits, a basis of such system has 2^n elements (it is equivalent to saying that, with n qubits, one can perform 2^n operations simultaneously). In particular, one can always consider for such elements of the basis the elements $|a_1\rangle \otimes |a_2\rangle \otimes \ldots \otimes |a_n\rangle$, with $a_i \in \{0, 1\}$ for all $i = 1, \ldots, n$. Since there are 2^n elements in this basis, we can change this previous notation to $|0\rangle, |1\rangle, \ldots |2^{n-1}\rangle$, to simplify it.

Therefore, a quantum state on n qubits, because of superposition, is given by

$$\alpha_0 |0\rangle + \alpha_1 |1\rangle + \ldots + \alpha_{2^n-1} |2^n - 1\rangle, \quad \sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1.$$

Moreover, as in the case of a single qubit, if one measures this in the computational basis, one just gets a "classical" n-bit state, $|i\rangle$, with probability $|\alpha_i|^2$.

In a more general setting, consider a physical system that can be in N different, mutually exclusive classical states (in the case of the qubit, $N = 2$, and for n qubits, $N = 2^n$). A *pure quantum state* $|\varphi\rangle$ is a superposition of classical states in the following form:

$$|\varphi\rangle = \alpha_1 |1\rangle + \alpha_2 |2\rangle + \ldots + \alpha_N |N\rangle.$$

The elements α_i in the previous expression are complex numbers that are called *amplitudes*, and, in this expression, it is easy to read the superposition phenomenon as the possibility of a quantum system to be in N classical states states at the same time (or perform N operations simultaneously, as mentioned above).

1.2 Measurements

In general, there are a couple of things that one can do with a quantum state: Measure it, or let it evolve unitarily without measuring it. In this subsection, we explain the first one.

Formally, we can define a measurement in the following form:

Definition 1. Let $\{M_n\}_n \subset \mathcal{B}(\mathcal{H})$[a] be a collection of operators verifying

$$\sum_n M_n^* M_n = \mathbb{1},$$

where $\mathbb{1}$ denotes the identity operator (we drop the subindex with the dimension when there is no possible confusion) and M_n^* denotes the dual of the operator M_n. This collection of operators is called **quantum measurements** when the following holds: Given a state of a quantum system $|\varphi\rangle$ before performing this operation to measure it, the probability that result $|n\rangle$ occurs is given by

$$p(n) = \langle \varphi | M_n^* M_n | \varphi \rangle$$

and the state of the system after this operation is given by

$$\frac{M_n |\varphi\rangle}{\sqrt{p(n)}}.$$

[a]Let $\mathcal{H}$ be a Hilbert space (in general, we will just consider finite-dimensional spaces) and let $T : \mathcal{H} \to \mathcal{H}$ be a linear operator on it. Since $\mathcal{H}$ is a Hilbert space, in particular it is a normed space with an associated norm $\|\cdot\|_{\mathcal{H}}$, which comes from a scalar product $\langle \cdot, \cdot \rangle$.

We say that T is a bounded operator if

$$\|T\|_{\mathcal{H}\to\mathcal{H}} < \infty,$$

where

$$\|T\|_{\mathcal{H}\to\mathcal{H}} := \sup_{x\in\mathcal{H}} \frac{\|T(x)\|_{\mathcal{H}}}{\|x\|_{\mathcal{H}}},$$

and denote by $\mathcal{B}(\mathcal{H})$ the space of bounded linear operators on $\mathcal{H}$.

Finally, if $T : \mathcal{H} \to \mathcal{H}$, we can define its dual operator, and denote it by T^*, as the operator that satisfies

$$\langle y, T(x)\rangle = \langle T^*(y), x\rangle \qquad \text{for every } x, y \in \mathcal{H}.$$

Consider again the state

$$|\varphi\rangle = \alpha_1 |1\rangle + \alpha_2 |2\rangle + \ldots + \alpha_N |N\rangle.$$

and assume that we measure it. As we have already mentioned, we will obtain the classical state $|i\rangle$, with probability $|\alpha_i|^2$, thus we cannot "see" the superposition itself. Among some other things, this means that the probability to get specifically the state $|i\rangle$ when we measure, and not another one, is $|\alpha_i|^2$. Hence, since the quantum state induces a probability distribution on the classical states, this implies

$$\sum_{i=1}^{N} |\alpha_i|^2 = 1.$$

Notice that when we measure $|\varphi\rangle$ and get a classical state, $|\varphi\rangle$ dissapears, and all that is left is the classical state itself. We say then that $|\varphi\rangle$ has *collapsed* to the classical state that we got, and the information encoded in the amplitudes α_i is now gone.

In general, we will measure in the computational basis. However, there are several ways to perform these measurements, that we will present throughout this section. Let us begin with the easiest one, the measurement of a qubit in its computational basis. It is defined by the measurement operators:

$$M_0 = |0\rangle\langle 0| \qquad \text{and} \qquad M_1 = |1\rangle\langle 1|.$$

Notice that both operators are *selfadjoint*, i.e., they coincide with their dual operators (actually, they are projections), and they verify $M_i^* M_i = M_i^2 = M_i$, for $i = 1, 2$, where M_i^* denotes the dual of the operator M_i, and $M_0 + M_1 = \mathbb{1}$. Also, when we measure

$$|\varphi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle,$$

the probability to obtain the outcome $|i\rangle$ is $|\alpha_i|^2$, and the state after measurement in that case is $\frac{\alpha_i}{|\alpha_i|} |i\rangle$. Actually, we can see that this state is equivalent to $|i\rangle$ (since it is just a rotation of the latter).

Indeed, consider $|\varphi\rangle$ and $e^{i\theta} |\varphi\rangle$ (which is a more general expression for the element mentioned above) and assume that we measure both states with a measurement $\{M_n\}_n$. Then, the probability of getting outcome n for the second element is

$$\left\langle \varphi e^{-i\theta} | M_n^* M_n | e^{i\theta}\varphi \right\rangle = \langle \varphi | M_n^* M_n | \varphi \rangle,$$

the same that for the first element. Hence, both states are operationally identical.

1.2.1 Distinguishability

One typical problem in quantum information is to distinguish between two (or more) quantum states. Namely, among several possible states, we have a particle in one of them and we want to find out in which one the particle actually is. We study this problem now in the simplest case: to distinguish between two possible states.

First, let us assume that the states that we want to distinguish, $|\varphi_1\rangle$ and $|\varphi_2\rangle$ are orthogonal. Then, if we choose the measurement operators $M_i = |\varphi_i\rangle \langle\varphi_i|$ for $i = 1, 2$ and define $M_0 = \mathbb{1} - \sum_i M_i$, it is easy to see that

$$M_0 + M_1 + M_2 = \mathbb{1}.$$

Therefore, given $i \in \{1, 2\}$, if $|\varphi\rangle$ is prepared in the state $|\varphi_i\rangle$, we have

$$p(i) = \langle \varphi | M_i | \varphi \rangle = 1, \qquad \text{and} \qquad p(j) = 0 \text{ for } j \neq i,$$

thus both states can be unambiguously distinguished.

Now, suppose that we want to distinguish two non-orthogonal states $|\varphi_1\rangle$ and $|\varphi_2\rangle$. We will see that there is no way to do that.

Proposition 1. *Let $|\varphi_1\rangle$ and $|\varphi_2\rangle$ be two non-orthogonal states. Then, we cannot distinguish between them.*

Proof. By reduction to the absurd, let as assume that there is a measurement $\{M_n\}_{n\in I}$ capable of doing that, i.e., distinguishing both states. Then, we can consider a partition of the set I ($I_1, I_2 \subset I$ such that $I_1 \neq \emptyset \neq I_2$, $I_1 \cup I_2 = I$ and $I_1 \cap I_2 = \emptyset$) and this allows us to decide that if the result of the measurement is $|m\rangle$, with the index $m \in I_i$, then the state is $|\varphi_i\rangle$.

Consider, for $i = 1, 2$, the operations $\mathbb{E}_i = \sum_{j\in I_i} M_j^* M_j$, which satisfy by construction $\mathbb{1} = \mathbb{E}_1 + \mathbb{E}_2$. It is clear that we have

$$\langle \varphi_i | \mathbb{E}_i | \varphi_i \rangle = 1 \ \text{ for } i = 1, 2,$$

and

$$\langle \varphi_1 | \mathbb{E}_2 | \varphi_1 \rangle = \langle \varphi_2 | \mathbb{E}_1 | \varphi_2 \rangle = 0.$$

Consider the first term in the previous equality. Since $\mathbb{E}_2$ is positive, one can write:

$$0 = \langle \varphi_1 | \mathbb{E}_2 | \varphi_1 \rangle = \left\langle \varphi_1 | \sqrt{\mathbb{E}_2}\sqrt{\mathbb{E}_2} | \varphi_1 \right\rangle,$$

so we get $\sqrt{\mathbb{E}_2}\,|\varphi_1\rangle = 0$. By assumption, $|\varphi_1\rangle$ and $|\varphi_2\rangle$ are not orthogonal, thus there exist $\alpha_1 \neq 0 \neq \alpha_2$ and another state $|\phi\rangle$, orthogonal to $|\varphi_1\rangle$, so that

$$|\varphi_2\rangle = \alpha_1 |\varphi_1\rangle + \alpha_2 |\phi\rangle.$$

Applying $\sqrt{\mathbb{E}_2}$ to this expression, we get:

$$\sqrt{\mathbb{E}_2}\,|\varphi_2\rangle = \alpha_1 \sqrt{\mathbb{E}_2}\,|\varphi_1\rangle + \alpha_2 \sqrt{\mathbb{E}_2}\,|\phi\rangle = \alpha_2 \sqrt{\mathbb{E}_2}\,|\phi\rangle.$$

However, this is a contradiction, since this implies $|\alpha_2| = 1$ and, by assumption, we have $|\alpha_2| < 1$ (since $\alpha_1 \neq 0$).

□

1.2.2 Projective Measurements

In this subsubsection, we are going to introduce *projective measurements*, which play an special role in Postulate III of quantum mechanics (as we will see in the following section). They can be defined in the following form.

Definition 2. Consider a collection $\{M_n\}_n \subset \mathcal{B}(\mathcal{H})$ of measurements, as described in Definition 1. Assume that they have the additional property that the M_n are orthogonal projections, i.e., they are self-adjoint and verify

$$M_n M_m = \delta_{mn} M_n,$$

where $\delta_{mn} = 1$ iff $m = n$ and 0 otherwise. These measurements are called **projective measurements**.

It is clear that each one of these operators M_n projects on a subspace $\mathcal{H}_n \subset \mathcal{H}$ of the global Hilbert space. Hence, an *observable* M can be defined as the Hermitian operator

$$M = \sum_n \lambda_n M_n,$$

where the term in the right hand-side is, in fact, the spectral decomposition of M. Moreover, the possible outcomes of the measurement correspond to the eigenvalues λ_n of the observable, and when we measure the state $|\varphi\rangle$, the probability of getting state $|n\rangle$ is:

$$p(n) = \langle\varphi|M_n|\varphi\rangle.$$

With this notation, the average value of the measurement, with respect to the state $|\varphi\rangle$, is

$$\sum_n np(n) = \sum_n n \langle\varphi|M_n|\varphi\rangle = \langle\varphi|M|\varphi\rangle.$$

1.2.3 POVM Measurements

In many situations, we will not be as interested in the post measurement state of our particle itself as in the probabilities of the different possible measurement outcomes. In this case, we can reduce to the formalism of the so called *Positive Operator Valued Measurements (POVM's)*.

Definition 3. Consider a measurement $\{M_n\}_n \in \mathcal{B}(\mathcal{H})$ as in the Definition 1. Then, we can define the *positive*[a] operators

$$E_n = M_n^* M_n.$$

This family of operators $\{E_n\}_n$ is called a **POVM**.

[a] An operator $T \in \mathcal{B}(\mathcal{H})$ is said to be *positive* (shortened form of *positive semidefinite*) if

$$\langle x, T(x)\rangle \geq 0 \qquad \forall x \in \mathcal{H}.$$

The operators mentioned in the definition of POVM are clearly positive, since

$$\langle x, E_n(x)\rangle = \langle M_n(x), M_n(x)\rangle = \|M_n(x)\| \geq 0 \qquad \forall x \in \mathcal{H}.$$

The operators presented in the definition of POVM clearly satisfy

$$\sum_n E_n = \mathbb{1}$$

and their probability of obtaining outcome m is

$$p(m) = \langle\varphi|E_m|\varphi\rangle.$$

Conversely, if we have a collection of positive operators $\{E_n\}_n$ verifying $\sum_n E_n = \mathbb{1}$, we can define a measurement $\{M_n\}_n$ from them just by considering $M_n = \sqrt{E_n}$.

1.3 Unitary evolution

Recall that we mentioned at the beginning of the previous subsection that there are a couple of things that one can do with a quantum state: Measure it, or let it evolve unitarily without measuring it. Now we are going to focus on the second one.

Consider the state

$$|\varphi\rangle = \alpha_1 |1\rangle + \alpha_2 |2\rangle + \ldots + \alpha_N |N\rangle.$$

Then, we want to change it to another state of the form:

$$|\psi\rangle = \beta_1 |1\rangle + \beta_2 |2\rangle + \ldots + \beta_N |N\rangle.$$

Quantum mechanics only allows linear operations to be applied to quantum states. This means that, after a change of notation (identifying $|\varphi\rangle$ with an n-dimensional vector), applying an operation that changes $|\varphi\rangle$ to $|\psi\rangle$ corresponds just to a multiplication by an $N \times N$ complex-valued matrix. With the previous expressions for $|\varphi\rangle$ and $|\psi\rangle$, one has

$$U \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_N \end{pmatrix}.$$

Since $|\psi\rangle$ should be a state, its coeficients should give a probability distribution, so the following has to hold:

$$\sum_{i=1}^{N} |\beta_i|^2 = 1.$$

This implies that U has to preserve the norm of vectors and, thus, be a *unitary* transformation. Since it is unitary, then, in particular, $U^{-1} = U^*$, and this inverse always exists, what can be translated in the quantum setting to the fact that every non-measuring operation on a quantum state must be reversible (in contrast with measurements, which were clearly non-reversible).

In the particular case of qubits, there are distinguished unitaries called *the Pauli matrices*, which are denoted by $\sigma_0, \sigma_x, \sigma_y, \sigma_z$, or just $\mathbb{1}, X, Y, Z$, respectively. They are defined by:

$$\sigma_0 = \mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}; \qquad \sigma_x = X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix};$$

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

From a quantum computational point of view we can think of unitary matrices as quantum logical gates. We will deepen in this connection in the first section of the following chapter.

1.4 Density operators

In this subsection, we introduce the density operators formalism that will be necessary to present the Postulates of the Quantum Mechanics in the Schrödinger picture. Before moving to the definition of density operators themselves, let us start by recalling some basic concepts.

For every operator $T : \mathcal{H} \to \mathcal{H}$, represented in a certain basis by a matrix M, one can define the *trace* of M, $\mathrm{tr}[M]$, as the sum of the elements of the diagonal of M. We don't emphasize the choice of basis to represent T on, since the trace is invariant under change of basis. Indeed, it is easy to see that the trace is linear and cyclic, i.e., for A and B matrices,

$$\mathrm{tr}(AB) = \mathrm{tr}(BA).$$

From this last property, one also gets unitarily invariance: For every unitary operator U,

$$\mathrm{tr}(UAU^*) = \mathrm{tr}(U^*UA) = \mathrm{tr}(A).$$

Finally, another useful and interesting property concerning the trace is the following. Let $|\varphi\rangle \in \mathcal{H}$ be a state (or unit vector), and consider the rank-one operator $|\varphi\rangle\langle\varphi| : \mathcal{H} \to \mathcal{H}$, which projects in the direction of $|\varphi\rangle$. Consider now an arbitrary operator $T \in \mathcal{B}(\mathcal{H})$ and suppose that we want to compute $\mathrm{tr}(A|\varphi\rangle\langle\varphi|)$. To do that, before we express $|\varphi\rangle$ in a basis $\{|i\rangle\}$ of $\mathcal{H}$ where the first element is exactly $|\varphi\rangle$, i.e., $|\varphi\rangle = |1\rangle$. Then, we get:

$$\mathrm{tr}(A|\varphi\rangle\langle\varphi|) = \sum_i \langle i|A|\varphi\rangle\langle\varphi|i\rangle = \langle\varphi|A|\varphi\rangle$$

Now, let us move to the formalism of density operators. In the previous subsections, we have described the state of a physical system identifying it with a unit vector in the Hilbert space $\mathcal{H}$. However, there is an equivalent description with trace class operators on the Hilbert space. One of the main advantages of this description with respect to certain problems appears, for example, when dealing with real experimental systems where noise is present.

A motivation for this formalism comes from the following situation: Sometimes, we do not know whether our system is in a specific state $|\varphi\rangle$, but rather that it is in each one of the states $|\varphi_i\rangle$ with probability p_i, respectively. Hence, we would like to be able to consider the element

$$\sum_i p_i |\varphi_i\rangle,$$

with the constants p_i verifying

$$\sum_i p_i = 1,$$

and work with it as a state. However, it is not a state anymore, since it is not a unit vector. To avoid this difficulty, one can associate each state $|\varphi_i\rangle$ to the rank-one projector $|\varphi_i\rangle\langle\varphi_i|$. Hence, the state in the previous scenario can be described, instead, in the following form

$$\rho = \sum_i p_i |\varphi_i\rangle\langle\varphi_i|,$$

where ρ is a Hermitian, positive semidefinite, and trace one operator. Indeed, it is clear from its description that ρ is Hermitian and has trace one (because of the linearity of the trace and the fact that $\mathrm{tr}(|\varphi_i\rangle\langle\varphi_i|) = 1$). To see that it is positive semidefinite, notice that for any $|\phi\rangle \in \mathcal{H}$,

$$\langle\phi|\rho|\phi\rangle = \sum_i p_i \langle\phi|\varphi_i\rangle\langle\varphi_i|\phi\rangle = \sum_i p_i |\langle\phi|\varphi_i\rangle|^2 \geq 0.$$

These operators are called *density operators* or *density matrices* and the set of such elements is usually denoted by $\mathcal{S}(\mathcal{H})$.

1.5 Postulates of quantum mechanics

The postulates of quantum mechanics were derived after a long process of trial and error, which involved a considerable amount of guessing and fumbling by the originators of the theory. The motivation for them is not always clear; even to experts the basic postulates of quantum mechanics appear surprising.

In this section, we mostly focus on the mathematical formulation for the postulates of quantum mechanics in two different (and dual) settings, Heisenberg and Schrödinger picture. These two descriptions will help us ot understand the topics presented in the following chapter.

1.5.1 Heisenberg picture

The postulates in the Heisenberg picture can be enunciated as follows.

Postulate 1. Given an isolated physical system, there is a complex Hilbert space $\mathcal{H}$ associated to it, which is known as the **state space** of the system.

Moreover, the physical system is completely described by its **state vector**, which is a unitary vector in the state space.

Quantum mechanics does provide, for a given physical system, the explicit form of the state space of that system, nor does it provide the state vector of the system. Finding these two elements for a specific system is a difficult problem, for which physicists have developed many intricate and beautiful tools, and many new fields are devoted to that.

In general, the state space $\mathcal{H}$ of the system under study will depend on the specific physical system, but we know that it is a *separable* Hilbert space. Frequently, one restricts to finite dimensional Hilbert spaces[2].

Postulate 2. Given an isolated physical system, its evolution is described by a **unitary transformation** in the Hilbert space. More specifically, if the state of the system at time t_1 is described by $|\varphi_1\rangle$, and at time $t_2 > t_1$ by $|\varphi_2\rangle$, then there exists a unitary operator $U \in \mathcal{B}(\mathcal{H})$, which depends on both instants t_1 and t_2, such that

$$|\varphi_2\rangle = U_{1,2}\,|\varphi_1\rangle.$$

From a more physical point of view, Postulate 2 can be rewritten as follows.

Postulate 2 bis. *Given a closed quantum system (with no interaction with an environment), the time evolution of a state on such system is described by the Schrödinger equation:*

$$i\hbar\frac{d\,|\varphi_t\rangle}{dt} = H\,|\varphi_t\rangle,$$

where $\hbar$ is the Planck's constant and H is a Hermitian operator associated to the physical closed system which is called Hamiltonian.

Let us consider the spectral decomposition of the Hamiltonian (since it is a Hermitian operator):

$$H = \sum_{E_i} E_i\,|E_i\rangle\,\langle E_i|,$$

where we denote by E_i the eigenvalues and by $|E_i\rangle$ the corresponding normalized eigenvectors, to emphasize the fact that these eigenvalues represent some energies of the physical system. Indeed, the states $|E_i\rangle$ are usually called *energy eigenstates* or *stationary states*, with associated energy E_i.

The lowest energy is known as *ground state energy*, and its associated eigenstate is known as the *ground state*, a fundamental element in the theory of quantum systems. Moreover, when the difference between the two smallest eigenvalues is strictly positive, this difference is called *spectral gap*, and we say in that case that the system is *gapped*. Determining whether a physical system has or not a spectral gap is a really important problem in Quantum Physics.

The states $|E_i\rangle$ mentioned above are called stationary because their only change in time is of the form

$$|E_i\rangle \mapsto \exp(-iE_it/\hbar)\,|E_i\rangle.$$

Let us see now the connection between the two formulations for this postulate. If we consider the Schrödinger equation, we can see:

[2]Since every separable Hilbert space has the *approximation property*, every compact operator in such space can be approximated by finite-rank ones. Hence, if one considers a density matrix, for instance, in the infinite dimensional separable Hilbert space, it is a compact operator, so theoretically one will be able to approximate this operator by a sequence of finite-rank ones, leading to the possibility to restrict to the study of finite-dimensional Hilbert spaces.

$$|\varphi(t_2)\rangle = \exp\left[\frac{-iH(t_2-t_1)}{\hbar}\right]|\varphi(t_1)\rangle = U(t_1,t_2)\,|\varphi(t_1)\rangle^{3},$$

where we are defining:

$$U(t_1,t_2) := \exp\left[\frac{-iH(t_2-t_1)}{\hbar}\right].$$

This operation is easily seen to be unitary, and, furthermore, one can see that any unitary operator U can be written in the form

$$U = \exp(iK),$$

for some Hermitian operator K.

Postulate 3. Given a physical system, with associated Hilbert space $\mathcal{H}$, the quantum measurements over such system are described by a collection $\{M_n\}_n \subset \mathcal{B}(\mathcal{H})$ of measurements as defined in Definition 1.

More specificaly, the index n refers to the measurement outcomes that may occur in the experiment, and given a state of a quantum system $|\varphi\rangle$ before a measurement, the probability that result $|n\rangle$ occurs is given by

$$p(n) = \langle\varphi|M_n^* M_n|\varphi\rangle$$

and the state of the system after the measurement is given by

$$\frac{M_n\,|\varphi\rangle}{\sqrt{p(n)}}.$$

Finally, measurement operators satisfy:

$$\sum_n M_n^* M_n = \mathbb{1}.$$

Measurement operators were already introduced in Subsection 1.2 and several kinds of them were presented there.

Finally, the fourth postulate can be stated as follows.

Postulate 4. Given a composite physical system, its state space is also composite, and corresponds to the tensor product of the state spaces of the component physical systems. Moreover, if each system i is prepared in the state $|\varphi_i\rangle$, then the composite system is in the state $|\varphi_1\rangle \otimes \ldots \otimes |\varphi_n\rangle$.

[3] In this expression, we are considering the exponential of a matrix. Analogously to the scalar case, given a matrix A, the matrix $\exp(A)$ is given by:

$$\exp(A) := \sum_{j=0}^{\infty} \frac{A^j}{j!},$$

where $j!$ denotes the factorial number of j, and, by convention, $0! = 1$.

After introducing the fourth postulate, it is necessary to make the following remark, which leads to introducing the concept of *entanglement*. A composite Hilbert space, i.e., a Hilbert space of the form $\mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \ldots \otimes \mathcal{H}_n$[4], contains elements which are not tensor products of elements of each one of the components. In other words, if $|\varphi\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \ldots \otimes \mathcal{H}_n$, there not exist, in general, $|\varphi_i\rangle \in \mathcal{H}_i$ for all i so that

$$|\varphi\rangle = |\varphi_1\rangle \otimes |\varphi_2\rangle \otimes \ldots \otimes |\varphi_n\rangle.$$

A standard example of a non trivial two qubit state is the *EPR pair* [16], or *Bell state* is the following state:

$$\left|\phi^+\right\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}.$$

This structure of tensor products leads to the definition of *quantum entanglement*, a behavior that seems to be at the root of many of the most surprising phenomena in quantum mechanics.

Definition 4. Given a state $|\varphi\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \ldots \otimes \mathcal{H}_n$, we say that $|\varphi\rangle$ is **entangled** if it cannot be written as an elementary tensor product.

Notice that, in particular, one needs to have more than one system to talk about entangled states.

1.5.2 Schrödinger picture

In the Schrödinger picture, we consider density matrices instead of states.

Postulate 1. Given an isolated physical system, there is a complex Hilbert space $\mathcal{H}$ which is known as the state space of the system. This system is completely described by its **density operator**, which is a Hermitian, positive semidefinite and trace one operator $\rho \in \mathcal{S}(\mathcal{H})$.

Moreover, if we know the probability of the system in every state (for each state ρ_i, the probability that the system is in that state is p_i), then the state ρ can be written as

$$\sum_i p_i \rho_i.$$

Since the Heisenberg and Schrödinger picture are duals, there is an identification between observables in the Heisenberg picture and density matrices in the Schrödinger one. This leads to directly calling by *states* the density matrices, in a slight abuse of notation. With this notation, we denote by *pure states* the density matrices of the form

[4] Consider two Hilbert spaces $\mathcal{H}_1$ and $\mathcal{H}_2$. Since these two Hilbert spaces have inner products (resp. $\langle\cdot,\cdot\rangle_1$ and $\langle\cdot,\cdot\rangle_2$), it is a natural question whether one can introduce an inner product, and therefore a topology, on the tensor product that arise naturally from those of the factors. This can be done by defininining the inner product as:

$$\langle \varphi_1 \otimes \varphi_2, \psi_1 \otimes \psi_2 \rangle = \langle \varphi_1, \psi_1 \rangle_1 \langle \varphi_2, \psi_2 \rangle_2$$

for every $\varphi_1, \psi_1 \in \mathcal{H}_1$ and $\varphi_2, \psi_2 \in \mathcal{H}_2$, and extending by linearity. Finally, we take the completion under this inner product, and we get as the resulting Hilbert space the tensor product of $\mathcal{H}_1$ and $\mathcal{H}_2$.

This can be generalized to the tensor product of n Hilbert spaces.

$$\rho = |\varphi\rangle \langle\varphi|$$

and by *mixed states* the ones of the form

$$\rho = \sum_i p_i |\varphi_i\rangle \langle\varphi_i|.$$

For the second postulate, concerning evolution of systems, we have the following formulation.

Postulate 2. Given an isolated physical system, with associated Hilbert space $\mathcal{H}$, its evolution is described by a **unitary transformation**. More specifically, if the state of the system t_1 is described by the density matrix ρ_1 and the state of the system at instant $t_2 > t_1$ is described by ρ_2, then there exist a unitary operator U, which depends only on t_1 and t_2, such that

$$\rho_2 = U\rho_1 U^*.$$

As in the Heisenberg picture, the evolution of a density matrix is given by a unitary. To explain the form of the statement of the second postulate, consider

$$\rho = \sum_i p_i |\varphi_i\rangle \langle\varphi_i|.$$

Notice that, since the system initially is in the state $|\varphi_i\rangle$ with probability p_i, then after the evolution given by a unitary U it will be in state $U|\varphi_i\rangle$ with probability p_i. Therefore, the associated density operator will be given by

$$\sum_i p_i U |\varphi_i\rangle \langle\varphi_i| U^* = U \left(\sum_i p_i |\varphi_i\rangle \langle\varphi_i| \right) U^* = U\rho U^*.$$

Moving to the third postulate and the relation with quantum measurements, we have the following formulation for it.

Postulate 3. Given an isolated physical system, with associated Hilbert space $\mathcal{H}$, any quantum measurements on it are described by a collection of measurement operators $\{M_n\}_n$ as the ones described in Definition 1. As in the case of the Heisenberg picture, each index n refers to the different outcomes that may occur when measuring. Indeed, If the state of the quantum system is ρ before the measurement, the probability that we get result n is given by

$$p(n) = \mathrm{tr}(M_n^* M_n \rho),$$

and the state that we get after the measurement is given by:

$$\frac{M_n \rho M_n^*}{p(n)}.$$

Moreover, since probabilities need to sum one, these operators have to satisfy

$$\sum_n M_n^* M_n = \mathbb{1}.$$

Suppose that we measure with the measurement $\{M_n\}_n$ a mixed state of the form

$$\rho = \sum_i p_i \left|\varphi_i\right\rangle \left\langle\varphi_i\right|.$$

Then, if the initial state is $|\varphi_i\rangle$, for instance, the probability of having outcome n is

$$p(n|i) = \langle\varphi_i|M_n^* M_n|\varphi_i\rangle = \mathrm{tr}(M_n^* M_n \left|\varphi_i\right\rangle \left\langle\varphi_i\right|).$$

Hence, the total probability of this outcome is

$$p(n) = \sum_i p(n|i)p_i = \sum_i p_i \,\mathrm{tr}(M_n^* M_n \left|\varphi_i\right\rangle \left\langle\varphi_i\right|) = \mathrm{tr}(\rho M_n^* M_n),$$

because of the definition of ρ. And analogously, one can see that the post-measurement state is given by:

$$\frac{M_n \rho M_n^*}{p(n)}.$$

Finally, concerning the state space of a composite physical system, we get the following postulate, due to the linearity of tensor products.

Postulate 4. Given a composite physical system, its state space is the tensor product of the state spaces of the component physical systems.

Moreover, if each system i is initially prepared in state ρ_i, then the state in which the composite system is prepared is given as the tensor product of the ρ_i, i.e., $\rho_1 \otimes \rho_2 \otimes \ldots \otimes \rho_n$.

These reformulations of the postulates of quantum mechanics in terms of the density operator are, clearly, mathematically equivalent to the description in terms of the state vector. However, as a way of thinking about quantum mechanics, the density operator approach has advantages with respect to two main facts: the description of quantum systems whose state is not known, and the description of subsystems of a composite quantum system.

Chapter 2

Quantum algorithms

In the future, quantum computers will have the potential to solve problems exponentially faster than classical computers. However, which problems can be solved exponentially faster? And which problems would do just as well as being solved by a classical computer? These are some of the main questions for quantum algorithm researchers.

To approach an answer to this question, it makes sense to develop the connection between quantum algorithms and computer hardware. A computer is a programmable machine which is based on some fundamental concepts of Physics, whereas an algorithm is a sequence of steps that leads to the solution of a problem. Their connection comes then into stage when one notices that for implementing an algorithm in a computer it is necessary to check that this machine can actually run it.

Quantum computers can run algorithms that we cannot run on classical computers. Those algorithms make use of quantum effects, such as non-locality, entanglement, superposition, etc. When one designs a quantum algorithm, one of their main aims is to exploit these concepts to make the algorithm get the solution to the problem faster. In general, we can say that a quantum algorithm provides opportunities to solve problems that are difficult in the classical context, i.e., problems that we cannot solve efficiently with the known classical algorithms.

A quantum algorithm might work as follows: It starts with a classical input, and turns it into a quantum state by obtaining the superposition of an exponential number of classical states. Then, it transforms the quantum state that encodes the problem into a quantum state that encodes the solution. From that quantum state, one can measure and obtain the solution.

Most of these quantum algorithms are designed for the future quantum computers, which will have many qubits. Nowadays, only some small prototipes of these devices already exist, counting on, at most, approximately 50 qubits. It is then an interesting research area to study what can one do with these devices before we actually have the ones with a great number of qubits.

The key to the security of public key cryptosystems is that it should be difficult to invert the encryption stage if only the public key is available. For example, inverting the encryption stage of RSA is in fact a problem closely related to factoring. Hence, much of the presumed security of RSA comes from the belief that factoring is a problem hard to solve on a classical computer. However, as we will see in this chapter, Shor's fast algorithm for factoring on a quantum computer could be used to break RSA. In an analogous way, there are other public key cryptosystems that can be broken using a fast algorithm for solving the discrete logarithm problem, like Shor's quantum algorithm for discrete logarithm. This practical application of quantum computers to the break codes

has excited much of the interest in quantum computation and quantum information.

The two main quantum algorithms that we will study in this chapter are *Shor's quantum factoring algorithm* and *Grover's algorithm*, which will be explained, respectively, in Section 2.6 and Section 2.7. First, we describe some of the earlier quantum algorithms that preceded them.

All quantum algorithms work with queries in some form or another, which we will explain below. The query complexity model differs from the standard model described in the previous chapter, as the input in this case is given as a *black-box*. However, before that, we need to introduce some basic notions about *quantum circuits* [9], [18], since they constitute one of the two models that are more commonly used to explain how a quantum computer can apply computational steps to its qubits. The other model, the *quantum Turing machine*, will not be discussed in this project.

Many of the topics discussed in this chapter are based in some of the following books or notes: [4], [5].

2.1 Preliminaries

In this section, first, we will present a brief survey on classical Boolean circuits, and, then, we will introduce some notions of quantum circuits, by outlining the diference with respect to the first ones.

2.1.1 Classical circuits

In a nutshell, we can say that circuits represent functions from $\{0,1\}^n$ into $\{0,1\}$. It is a computational model that consists of decomposing each function in some elemental operations, so that this procedure allows us to represent all the possible functions in the domain. This model has good properties in general and is fundamental in computational theory.

In classical complexity theory, one can define a *Boolean circuit* as a finite directed acyclic graph with AND, OR and NOT gates, whose shape can be seen in Figure 2.1. The rest of the gates that appear on that figure can be constructed from those three.

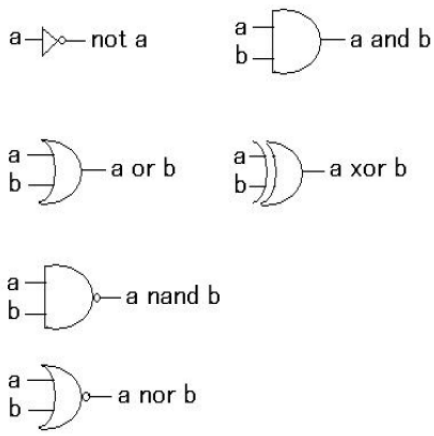

Figure 2.1: Some classical gates for two qubits.

An important theoretical result is that any function on bits can be computed from the

composition of NAND gates alone, which is thus known as a *universal gate*. By contrast, the XOR alone or even with NOT is not universal (one can notice that just by taking a look at the parity).

The idea of classical circuits lays on the following facts:

- Every circuit has n input nodes, which contain n input bits.
- The circuit is made of those three gates (AND, OR and NOT), and combinations of then, as well as some output nodes.
- The initial input bits are fed into combinations of the previous gates, so that eventually the output nodes assume some value.

Let $f : \{0,1\}^n \to \{0,1\}^m$ be a Boolean function. Then, we say that a circuit *computes* it if the output nodes get the right value $f(x)$ for every $x \in \{0,1\}^n$.

Now we can introduce some concepts related to the complexity of some circuits. Let us denote a *circuit family* by a set $\mathcal{C} = \{C_n\}$, each one of them of *input size* n (which means that the number of input nodes, and hence bits, is exactly n). We assume that each one of these circuits has one output bit. Then, we say that this family *recognizes* a certain *language* $L \subseteq \bigcup_{n\geq 0} \{0,1\}^n$ (which we denote hereafter by $\{0,1\}^*$) if, for every $x \in \{0,1\}^n$, the circuit C_n outputs:

- 1 if $x \in L$.
- 0 if $x \notin L$.

We say that this circuit family is *uniformly polynomial* if there exists a deterministic Turing machine that outputs C_n, given n as an input, in some time that depends polynomially in n. The *size* of a circuit is its number of gates, and it can grow, at most, polynomially in n. This kind of circuit families have equivalent power to polynomial time deterministic Turing machines, since it is known that a language L can be decided by a uniformly polynomial circuit family if, and only if, $L \in \mathbf{P}$, where $\mathbf{P}$ is the complexity class of languages that are decidable by polynomial-time Turing machines.

2.1.2 Quantum circuits

Let us move now to quantum circuits, which generalize the idea of classical circuit families. In this case, we replace the AND, OR and NOT gates by elementary *quantum gates*.

We define a quantum gate as a unitary transformation in a small number of qubits, usually 1, 2 or 3. Let us recall from the previous chapter that a unitary matrix satisties that its inverse coincides with its conjugate transpose. Some of the most important 1-qubit gates are the following:

- **Bitflip** gate (**X**): It negates the bit, i.e., swaps $|0\rangle$ and $|1\rangle$. It can be represented by:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

- **Phaseflip** gate (**Z**): It puts a - in front of $|1\rangle$. It can be represented by:

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

- **Phase** gate (R_ϕ): It rotates the phase of the $|1\rangle$-state by an angle ϕ:

$$R_\phi |0\rangle = |0\rangle,$$
$$R_\phi |1\rangle = e^{i\phi} |1\rangle.$$

  It is clear that the previous gate is a particular case of this family of gates. They can be represented by:

$$R_\phi = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}.$$

- **Hadamard** gate (**H**): It is specified by:

$$H |0\rangle = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle,$$
$$H |1\rangle = \frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle.$$

  As a unitary matrix, it can be represented by:

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

The last one, the Hadamard gate, is possibly the most important 1-qubit gate. If we apply H to an initial state $|0\rangle$ and then measure, we have the same probability of observing $|0\rangle$ or $|1\rangle$, and analogously if we apply it to initial $|1\rangle$. However, when applied to the superposition state

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

the Hadamard gate provides the value $|0\rangle$. The effect that we get in this case (both positive and negative amplitudes for $|1\rangle$ cancelling out) is called *interference*. It is completely analogous to the interference patterns that one can notice in light or sound waves.

Let us present now an example of a 2-qubit gate, the CNOT (*controlled-not*) gate.. Given two input bits, this gate is used to negate the second bit if the first one is 1, and to leave it invariant if the first bit is 0, i.e.,

$$\text{CNOT} |0\rangle |b\rangle = |0\rangle |b\rangle,$$
$$\text{CNOT} |1\rangle |b\rangle = |1\rangle |1 - b\rangle.$$

We can represent this action in the following matrix form:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

In this scenario, the first qubit is called the *control* qubit, since it is the one that determines the effect of the gate, and the second one is called the *target* qubit, as it is the one that receives the effect.

In general, if U is a 1-qubit gate (as the ones that we have defined above), then the we can define the 2-qubit controlled-U gate analogously to the previous one, i.e., if the first bit is 0 it does nothing, and, if it is 1, the gate applies the unitary to the second bit. We can represent it in the following matrix form:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{11} & U_{12} \\ 0 & 0 & U_{21} & U_{22} \end{pmatrix}.$$

Another way to understand the quantum CNOT gate is as a generalization of the classical XOR gate. Note, however, that there are some classical gates, like NAND or XOR, which cannot be understood as unitary gates in a sense similar to the way the quantum NOT gate represents the classical NOT gate. The reason is because these two gates are essentially irreversible.

We can see that in the following example: Given an output $A \oplus B$ of a XOR gate, it is not possible to determine what the inputs A and B were. This can be also stated by saying that there is a loss of information associated with the irreversible action of the XOR gate. On the other hand, since quantum gates are described by unitary matrices, it is important to remark that they can always be inverted by another quantum gate.

Let us now introduce an example of a 3-qubit gate, the Toffoli gate. This gate can be seen as a controlled-controlled not gate, as it negates the third bit of its input if both the first two bits are 1. It is quite important, since it is complete for classical reversible computations, in the sense that any classical computation can be implemented by a circuit of Toffoli gates (further discussion about this topic can be found in Subsection 2.1.6).

All those gates mentioned above can be composed into bigger unitary operations in the following ways:

- By taking *tensor products*, if the gates are applied *in parallel*.
- By taking *matrix products*, if the gates are applied *sequentially*.

We show now an example of these operations. If we apply a Hadarmard gate H to each bit in a register of n zeros, we get

$$\frac{1}{\sqrt{2^n}} \sum_{j \in \{0,1\}^n} |j\rangle,$$

a superposition of all n-bit strings, whereas applying $H^{\otimes n}$ to an initial state $|i\rangle$, with $i \in \{0,1\}^n$ gives us

$$H^{\otimes n} |i\rangle = \frac{1}{\sqrt{2^n}} \sum_{j \in \{0,1\}^n} (-1)^{i \cdot j} |j\rangle,$$

with $i \cdot j = \sum_{k=1}^{n} i_k j_k$ the inner product of the n-bit strings $i, j \in \{0,1\}^n$. In this case, one can also notice that the Hadamard is its own inverse. Thus, if we apply it again on the right-hand side of the previous expression, we get the initial $|i\rangle$. This makes the Hadarmard gate quite useful for the development of algorithms, as we will see in the following section.

To sum up, as in the case for classical circuits, one can define a quantum circuit in the following form.

Definition 5. A *quantum circuit* is a finite directed acyclic graph composed by:

- **Input nodes.** Some of these nodes (n nodes) contain the input, and some more nodes are initially $|0\rangle$ (they are called the *workspace*).
- **Quantum gates.** Each of them operates on, at most, two or three qubits of the state.
- **Output nodes.** The previous gates transform the initial state vector into a final state, which will generally be a superposition.

Let us see now how one can draw these circuits. We usually consider that time progresses from left to right. As briefly mentioned above, each qubit is represented as a wire, and the circuit prescribes which gates are applied to each wire.

With this notation of wires, it is clear that 1-qubit gates act on just one wire, whereas 2-qubit and 3-qubit gates act, respectively, on 2 or 3 wires. Moreover, when a gate acts on more than 1 qubit, and one of them is the *control* one, its wire is drawn with a dot linked vertically to the *target* qubits, i.e., the qubits where this effect is applied.

We show an example of this notation in the following figure.

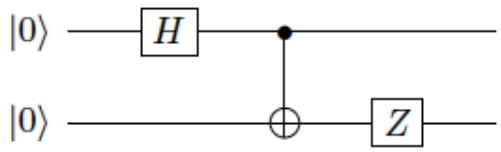

Figure 2.2: Circuit used to turn $|00\rangle$ into $\frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$.

In this example, and in general, we denote the quantum CNOT by $\oplus$. If we study every step separately, and taking into account the definition for every gate mentioned above, we can see that, after each step, the resulting state is:

- **Step 0.** We start with $|\varphi_0\rangle = |00\rangle$.
- **Step 1.** After the Hadarmard gate, we have $|\varphi_1\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$.
- **Step 2.** When we apply the CNOT gate, we get $|\varphi_2\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$.
- **Step 3.** Finally, after the Z gate, we have $|\varphi_3\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$.

2.1.3 No-cloning theorem

In this subsection, we will present an application of the previously defined gates. In particular, we will give an answer to the following question: 'Can we clone a classical/quantum bit?'.

In the classical case, it is easy to see that the answer is affirmative. Indeed, it is a trivial application of the classical CNOT gate, as we can see in the following figure.

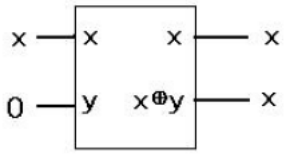

Figure 2.3: Classical way to clone a bit.

The explanation for this circuit is the following. Let us assume that we start with the bit x that we want to clone, and we take it as a *control* bit, as well as the 0 bit, which we take as the *target* bit. Then, applying a classical CNOT gate, one automatically gets an output given by two copies of x.

In the quantum case, one could be tempted to think that a similar argument can be followed using instead a quantum CNOT gate. However, it does not work [12], as we show below.

Consider a quantum machine with two inputs labeled by A and B, as in the classical case (for x and 0). We will assume that A is the *control* input, whereas B constitutes the *target* one. Suppose that A starts in an unknown pure quantum state $|\psi\rangle$, and we want to copy it into input B, which is also initiated in a pure state $|\phi\rangle$. Hence, the initial state is given by

$$|\psi\rangle \otimes |\phi\rangle.$$

Now, let us assume that cloning is possible, i.e., that there exists a unitary U such that

$$U\left(|\psi\rangle \otimes |\phi\rangle\right) = |\psi\rangle \otimes |\psi\rangle.$$

And let us further assume that this procedure works for two particular pure states $|\psi\rangle$ and $|\varphi\rangle$. Then, we get:

$$\begin{aligned} U\left(|\psi\rangle \otimes |\phi\rangle\right) &= |\psi\rangle \otimes |\psi\rangle, \\ U\left(|\varphi\rangle \otimes |\phi\rangle\right) &= |\varphi\rangle \otimes |\varphi\rangle. \end{aligned}$$

Since the application of unitaries respects inner products, we must have

$$\langle \psi, \varphi \rangle = |\langle \psi, \varphi \rangle|^2,$$

and this happens only if they are either the same state or orthogonal states. Thus, cloning devices can only clone states which are orthogonal to one another and, therefore, a general quantum cloning device is impossible.

Remark 1. In this subsection, we have focused on the possibility of constructing a cloning device with a unitary evolution. However, it can be proved that even if one allows non-unitary cloning devices this perfect cloning of non-orthogonal pure states remains impossible, but the techniques for that are much deeper.

2.1.4 Quantum teleportation

In this subsection, we present another example of how the gates of the previous subsections can be used to get really interesting results. In this particular case, we will explain *quantum teleportation* [14] as a combination of some elementary gates from the ones above.

Quantum teleportation is one of the most representative communication protocols of quantum information theory. Suppose there are two parties, Alice and Bob, which are spatially separated, and Alice wants to send a qubit of information to Bob by just sending two classical bits via a classial channel. For that, they need first to share an EPR-pair, given by:

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

The fact that a qubit can be sent by means of this procedure is usually expressed by writing:

$$1 \text{ EPR} + 2 \text{ bits} \geq 1 \text{ qubit}.$$

In fact, it is important to remark that Alice does not need to know her own qubit in order to send it to Bob. Let us see how this works. Assume that Alice's qubit is of the form:

$$|\varphi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle,$$

thus, because of what we have just mentioned, we can assume that Alice does not know the values of α and β. We can then split the procedure in the following steps:

1. Since Alice and Bob share the EPR-pair mentioned above (we can say that Alice holds the first qubit and Bob the second one), their initial joint state is of the form:

$$\begin{aligned}|\varphi_0\rangle &= |\varphi\rangle \otimes |\psi\rangle \\ &= (\alpha\,|0\rangle + \beta\,|1\rangle) \otimes \left(\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)\right) \\ &= \frac{1}{\sqrt{2}}\left[\alpha\,|0\rangle_A (|00\rangle_{A'B'} + |11\rangle_{A'B'}) + \beta\,|1\rangle_A (|00\rangle_{A'B'} + |11\rangle_{A'B'})\right],\end{aligned}$$

   where we are writing subindices in the last line to outline whom each bit belongs to.

2. Alice now applies a CNOT on her part (the elements marked with A and A' in $|\varphi_0\rangle$). She then gets:

$$\begin{aligned}|\varphi_1\rangle &= U_{A,A'} \otimes \mathbf{1}_B(|\varphi_0\rangle) \\ &= \frac{1}{\sqrt{2}}\left[\alpha\,|0\rangle_A (|00\rangle_{A'B'} + |11\rangle_{A'B'}) + \beta\,|1\rangle_A (|10\rangle_{A'B'} + |01\rangle_{A'B'})\right].\end{aligned}$$

   Notice that only the elements of the second term in the sum have suffered some change.

3. Later, Alice applies a Hadamard operaton on her first qubit, getting:

$$\begin{aligned}|\varphi_2\rangle &= H_A \otimes \mathbf{1}_{A'B'}(|\varphi_1\rangle)\\ &= \frac{1}{\sqrt{2}}\left[\alpha\left(\frac{|0\rangle_A + |1\rangle_A}{\sqrt{2}}\right)(|00\rangle_{A'B'} + |11\rangle_{A'B'}) + \right.\\ &\left.\beta\left(\frac{|0\rangle_A - |1\rangle_A}{\sqrt{2}}\right)(|10\rangle_{A'B'} + |01\rangle_{A'B'})\right]\\ &= \frac{1}{2}\left[|00\rangle_{A,A'}(\alpha|0\rangle_B + \beta|1\rangle_B) + |01\rangle_{A,A'}(\alpha|1\rangle_B + \beta|0\rangle_B)\right.\\ &\left. + |10\rangle_{A,A'}(\alpha|0\rangle_B - \beta|1\rangle_B) + |11\rangle_{A,A'}(\alpha|1\rangle_B - \beta|0\rangle_B)\right].\end{aligned}$$

This last expression has four different terms, and each one of them can be seen as the product one of the elements of the 2-qubit computational basis for Alice and another qubit (in four different forms) for Bob. Thus, if Alice measures her pair in the computational basis, i.e.,

$$\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\},$$

she gets one of the previous four elements and, thus, she can read off Bob's postmeasurement from this value, given her measurement, in the following way:

$$\begin{cases} |00\rangle_A \mapsto |\varphi_3(00)\rangle_B := \alpha|0\rangle_B + \beta|1\rangle_B\\ |01\rangle_A \mapsto |\varphi_3(01)\rangle_B := \alpha|1\rangle_B + \beta|0\rangle_B\\ |10\rangle_A \mapsto |\varphi_3(10)\rangle_B := \alpha|0\rangle_B - \beta|1\rangle_B\\ |11\rangle_A \mapsto |\varphi_3(11)\rangle_B := \alpha|1\rangle_B - \beta|0\rangle_B. \end{cases}$$

Hence, the way of proceeding is the following: Alice measures her two bits in the computational basis, sends the result to Bob over a classical channel, and Bob knows which transformation he must do on his qubit to regain the desired qubit $|\varphi\rangle$. In each case, Bob has to do the following:

$$\begin{cases} \text{Alice sends } |00\rangle_A \mapsto & \text{Bob's qubit is } \alpha|0\rangle_B + \beta|1\rangle_B \mapsto & \text{He does nothing to get } |\varphi\rangle\\ \text{Alice sends } |01\rangle_A \mapsto & \text{Bob's qubit is } \alpha|1\rangle_B + \beta|0\rangle_B \mapsto & \text{He applies an X gate to get } |\varphi\rangle\\ \text{Alice sends } |10\rangle_A \mapsto & \text{Bob's qubit is } \alpha|0\rangle_B - \beta|1\rangle_B \mapsto & \text{He applies a Z gate to get } |\varphi\rangle\\ \text{Alice sends } |11\rangle_A \mapsto & \text{Bob's qubit is } \alpha|1\rangle_B - \beta|0\rangle_B \mapsto & \text{He applies X and Z gate to get } |\varphi\rangle. \end{cases}$$

Remark 2. It is important to remark that we are not transmiting states instantaneously. Indeed, this would be a contradiction with the theory of relativity, which states that nothing can travel faster than light.

Quantum teleportation does not allow for faster than light communication, as, to complete the protocol, in step 3 above, Alice must transmit her measurement's result to Bob over a classical communication channel, and this is clearly limited by the speed of light. Moreover, without this classical communication, teleportation does not convey any information at all.

Remark 3. We also need to remark that we are not creating a copy of $|\varphi\rangle$ being teleported (and hence, there is no contradiction with the previous subsection). Although it seems that this could be the case, after the teleportation process, only the target qubit is left in the state $|\varphi\rangle$, and the original control qubit ends up in one of the computational basis states. In fact, Alice's side has been destroyed in this protocol.

2.1.5 Superdense coding

Now we present a third example of use of the previous quantum gates for a certain communication protocol. More specifically, in this subsection we discuss superdense coding [15], which is a communication protocol that allows to transmit 2 classical bits of information by just sending 1 qubit, assuming that Alice and Bob share an EPR pair.

Analogously to what we stated in the previous example, we can express the fact that two bits can be sent by means of this procedure by writing:

$$1 \text{ EPR} + 1 \text{ qubit} \geq 2 \text{ bits}.$$

The idea of this protocol is that Alice wants to send two bits to Bob just by sending a qubit. The algorithm is then given by the following steps:

1. Alice and Bob share an EPR state of the form:

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

2. Depending on the qubit Alice wants to send, she has to do something different in each one of the following cases:

$$\begin{cases} \text{Alice wants to send } |00\rangle_A \mapsto & \text{She does nothing to } |\psi\rangle \mapsto & |\psi_2\rangle := \frac{|00\rangle+|11\rangle}{\sqrt{2}} \\ \text{Alice wants to send } |01\rangle_A \mapsto & \text{She applies a } Z \text{ gate to } |\psi\rangle \mapsto & |\psi_2\rangle := \frac{|00\rangle-|11\rangle}{\sqrt{2}} \\ \text{Alice wants to send } |10\rangle_A \mapsto & \text{She applies a } X \text{ gate to } |\psi\rangle \mapsto & |\psi_2\rangle := \frac{|10\rangle+|01\rangle}{\sqrt{2}} \\ \text{Alice wants to send } |11\rangle_A \mapsto & \text{She applies a } iY \text{ gate to } |\psi\rangle \mapsto & |\psi_2\rangle := \frac{|01\rangle-|10\rangle}{\sqrt{2}}. \end{cases}$$

   Since these four last elements for the *Bell basis* of $\mathbb{C}^2 \otimes \mathbb{C}^2$, each state can be perfectly distinguished by an appropriate quantum measurement.

3. Alice sends her part of the state to Bob, so that he is now in possession of the whole state. By measurement the whole state in the Bell basis, Bob can then determine which of the four possible bit strings Alice sent him.

2.1.6 Universality of quantum gates

In this final subsection concerning circuits, we are going to discuss universality of certain sets of elementary gates.

Let us start with the classical case. For classical gates, one can see that AND and NOT together are universal, in the sense that any classical Boolean circuit can be implemented just using AND and NOT gates.

In subsection 2.1.2, we mentioned that Toffoli gates are complete for classical reversible computation. We explain this in more detail now. One can get an AND and a NOT classical gates from a Toffoli gate in the following way:

- Toffoli gate $\mapsto$ Fix the 3rd ingoing wire to 0 $\mapsto$ Classical AND gate.
- Toffoli gate $\mapsto$ Fix the 1st and 2nd ingoing wires to 1 $\mapsto$ Classical NOT gate.

Hence, if we apply Toffoli gates, we can implement any classical computation in a reversible manner.

Now, if we move to the quantum case, there are also several possibilities for universal sets of elementary gates. Let us see some examples.

- **All 1-qubit operations + 2-qubit CNOT.** This set is universal, in the sense that any other unitary transformation can be built from these gates.

  However, it is difficult to consider this set, as 'all' possible 1-qubit gates are difficult to be described (there are continuously many of them). Also, we cannot expect that experimentalists can implement these gates with infinite precision. Hence, the practical model that is usually considered allows just a small finite set of 1-qubit gates from which the rest can be efficiently approximated.

- **CNOT, Hadamard and $R_{\pi/4}$.** It is universal concerning approximation. It means that any other unitary can be arbitrarily well approximated using circuits, and it is a consequence of the *Solovay-Kitaev theorem*.

  If we restrict to real numbers, then we also have the following set.

- **Hadamard and Toffoli.** This set is universal for all unitaries tieh real entries, again in the sense of approximation.

2.2 Introduction to quantum algorithms

As we have mentioned in the introduction of this chapter, the two main quantum algorithms that we will study here are *Shor's quantum factoring algorithm* and *Grover's algorithm*, which will be explained, respectively, in Section 2.6 and Section 2.7. However, before, we will introduce and describe some of the earlier quantum algorithms that appeared previously.

All quantum algorithms work with queries in some form or another. The query complexity model differs from the standard model described in the previous chapter, as the input in this case is given as a *black-box*, what implies that the exponential separation between quantum and classical that we describe in the following algorithms does not lead to an exponential separation between quantum and classical in the standard model.

The phenomenon that appears in the query setting can be further explained as follows. Consider an N-bit input $x \in \{0,1\}^N$, for $N = 2^n$. Hence, we can address a bit x_i using an n-bit index i. The input can be seen as an N-bit memory which we can access at any point of our choice (a Random Access Memory), and this access is done via a black-box, which is designed to output the bit x_i on input i.

As a quantum operation, it must always be a unitary mapping on $n+1$ qubits, given by

$$O_x : |i, 0\rangle \to |i, x_i\rangle,$$

and, in general,

$$O_x : |i, b\rangle \to |i, b \oplus x_i\rangle,$$

where we recall that $i \in \{0,1\}^n$ and $b \in \{0,1\}$. Also, by $\oplus$ we denote *exclusive-or*, which is nothing but addition modulo 2. The first n qubits of the state are called the *address bits* (in previous sections have been callen *control bits*), while the qubit in position $(n+1)$ is called the *target bit*.

In the setting of matrix representation, O_x is a permutation matrix, and, as we have mentioned above, it is unitary. Here we can notice a difference between a quantum computer and a classical one, since the first one can apply O_x on a superposition of several i, something that the latter cannot do (we will elaborate this some paragraphs below).

One application of this black-box is called a *query*, and counting the required number of queries to compute a certain function of x is something essential in quantum complexity. Assume now that we make a query of the type mentioned above. Thus, we can also make a query of the form

$$|i\rangle \mapsto (-1)^{x_i} |i\rangle$$

by setting the target bit to the state

$$|-\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) = H |1\rangle.$$

Hence, we obtain:

$$O_x (|i\rangle |-\rangle) = |i\rangle \frac{1}{\sqrt{2}} (|x_i\rangle - |1 - x_i\rangle) = (-1)^{x_i} |i\rangle |-\rangle.$$

The aim of term $(-1)^{x_i}$ is to put the output variable in the phase of the state:

1. If x_i is 1, then we get a -1 in the phase of basis state.
2. If x_i is 0, then the phase of basis state remains invariant.

This "phase-oracle" is sometimes more convenient than the standard type of query. We sometimes denote the corresponding n-qubit unitary transformation by $O_{x,\pm}$.

Before introducing the first proper quantum algorithm in the next section, we need to present a quantum-mechanical effect that appears exclusively in quantum mechanics and we can (and will) use for building quantum algorithms, and which is called *quantum parallelism*. Assume that we have a Boolean function $f : \{0,1\}^n \to \{0,1\}^m$ and a classical algorithm that computes it. It is clear then that we can construct a quantum circuit of Toffoli gates (because of the universality mentioned in the previous) that maps:

$$|x\rangle |0\rangle \mapsto |x\rangle |f(x)\rangle \qquad \text{for every } x \in \{0,1\}^n.$$

Let us denote this quantum circuit by U. Then, if we apply U to a superposition of all possible inputs, we have:

$$U \left(\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |0\rangle \right) = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |f(x)\rangle.$$

Even though we have applied U just once, the final superposition state that we get contains all possible inputs. However, to observe the final superposition just gives one random $|x\rangle |f(x)\rangle$, so, by itself, this effect is not very useful, since it does not provide any improvement to classical randomization. It has to be combined with some other effects of interference or entanglement to improve the classical case.

In the following sections, we will use the concepts that we have just introduced to present several quantum algorithms. In general, one can classify quantum algorithms in three different classes which provide an advantage over known classical algorithms. The first one is the class of algorithms *based upon quantum versions of the Fourier transform*, a

tool which is also widely used in classical algorithms. The *Deutsch–Jozsa* algorithm is an example of this type of algorithm, although the use of such transform in the algorithm is a bit vague. The main exponent of this class of algorithms is *Shor's algorithm* for factoring and discrete logarithm.

The second class of algorithms is *quantum search algorithms*. The main algorithm from this class that we will present is *Grove's algorithm*. Finally, the third class of algorithms is *quantum simulation*, in which a quantum computer is used to simulate a quantum system.

In this project, we just focus on a brief description of the first two classes of algorithms mentioned above. This description will be presented in the following sections.

2.3 Deutsch-Jozsa's algorithm

In the early 1980s, Richard Feynman, in the US, and Yuri Manin, in the Soviet Union, suggested using quantum computers to simulate quantum mechanics. Some years later, in 1985, David Deutsch tried to actually formulize this. For that, he defined a quantum Turing machine and explained how to solve what later was going to turn out to be the 2-bit Deutsch-Jozsa problem.

Solving the 2-bit Deutsch-Jozsa problem did not really impress anyone. However, in 1992, David Deutsch and Richard Jozsa generalized his construction to solve the n-bit Deutsch-Jozsa problem [7].

Problem 1. *For $N = 2^n$, we are given an element $x \in \{0,1\}^N$ such that one of the following holds:*

1. *Every x_i has the same value.*
2. *$N/2$ of the x_i are 0 and $N/2$ are 1.*

In the first case, we call x constant, whereas in the second case we call it balanced. The goal is to find out whether x is constant or balanced.

This is what computer scientists call a *promise problem*, since we get the promise that we are always going to receive an n-bit in one of the previous two situations.

We shall now put this problem in words. Suppose that Alice, in Amsterdam, selects a number x from 0 to $2^n - 1$, and mails it in a letter to Bob, in Boston. Bob then calculates some function $f(x)$ and replies with the result he gets, which is either 0 or 1. The function that Bob uses is of one of two kinds: either $f(x)$ is constant for all values of x, or else $f(x)$ is balanced. Alice's goal is to determine with certainty whether Bob has chosen a constant or a balanced function, exchanging with him as little information as possible. How fast can she succeed?

More formally, we define a function $f : \{0,1\}^N \to \{0,1\}$, i.e., it takes N-bits as input and produces either a 0 or a 1 as output for each such value. We are promised that the function is either constant (0 on all outputs or 1 on all outputs) or balanced (returns 1 for half of the input domain and 0 for the other half).

The most natural question one can think of is: How is f given? In this sense, f is given as an oracle. Hence, we are given a black box, we input an N-bit x and it outputs $f(x)$, as we show in Figure 2.4.

It is clear that this black box is not really a quantum circuit, since it has an N-bit as input and one bit as output, and we have mentioned in previous sections that a quantum

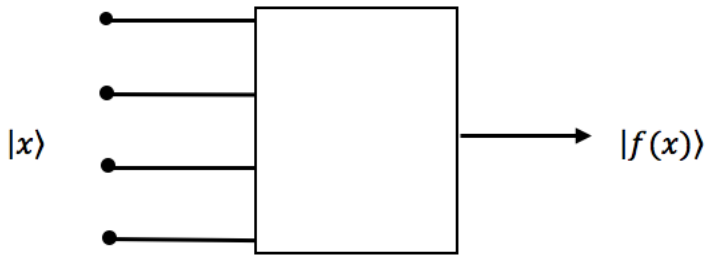

Figure 2.4: f is given as an oracle.

oracle has to be reversible, which means that the number of inputs should be the number of outputs. Hence, it can only be a classical oracle.

A way to get an actual quantum circuit is by recalling that any classical function can be made reversible as long as you keep the input around. In this situation, this reads as we can see in Figure 2.5:

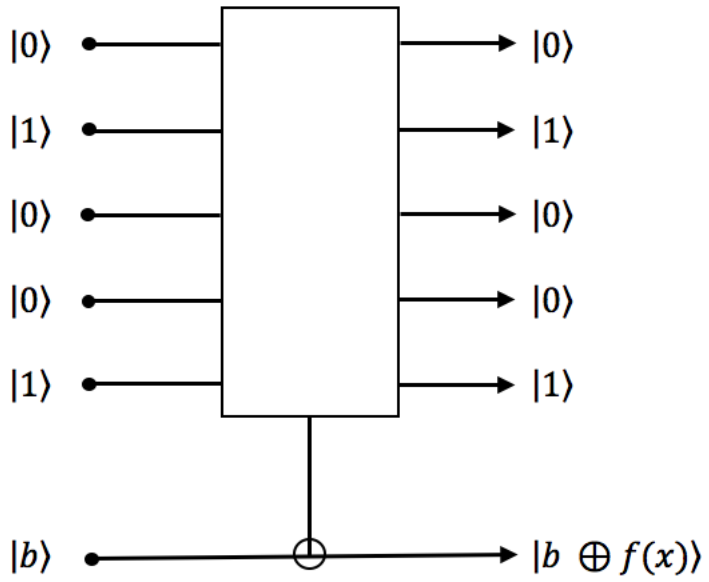

Figure 2.5: f is given as an oracle.

Moreover, one can notice that it is its own inverse.

2.3.1 2-bit functions

Let us show how we can solve the Deutsch-Jozsa problem for the 2-bit qubit case. For that, consider the quantum oracle Figure 2.6, so that it changes the phase of the input if $f(x) = 1$, and leaves it untouched if $f(x) = 0$.

Since we are studying the 2-bit case, it is clear that $|x\rangle \in \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. Let us consider the input

$$\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle).$$

Figure 2.6: Quantum oracle.

For that input, we can have, if f is constant and equal to 0, the output

$$\frac{1}{2}\left(|00\rangle + |01\rangle + |10\rangle + |11\rangle\right)$$

and for f equal to 1

$$-\frac{1}{2}\left(|00\rangle + |01\rangle + |10\rangle + |11\rangle\right).$$

On the opposite, if f is balanced, the output would be

$$\frac{1}{2}\left(|00\rangle - |01\rangle + |10\rangle - |11\rangle\right),$$

or something like that.

If we recall that solving the Deutsch-Jozsa problem consists on distinguishing between these two possibilities, we need to develop a tool to do that. In this situation, we can just reduce to making a measurement that obtains a *yes* if the function is constant and a *no* if it is balanced. We can construct this measurement applying two Hadamard gates and testing for 0:

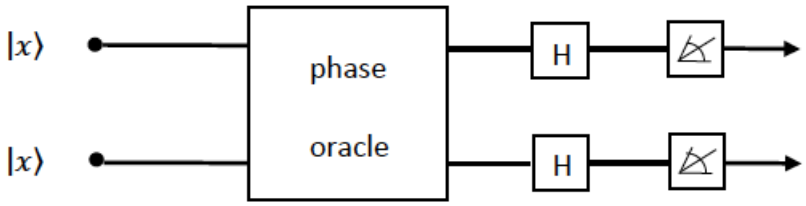

Figure 2.7: 2-bit Deutsch-Jozsa model.

We will present in more detail this phase oracle in the next subsection. If output equals $|00\rangle$, then f is constant. On the contrary, if output is different from $|00\rangle$, f is balanced. However, it is important to remark that there are many cases in which f is neither constant nor balanced. For example, consider

$$f\,|{+}{+}\rangle = \frac{1}{2}\left(|00\rangle + |01\rangle + |10\rangle - |11\rangle\right).$$

Then, the probability of seeing $|{+}{+}\rangle$ is

$$\text{Prob}(|{+}{+}\rangle) = \frac{1}{2},$$

which is neither 0 nor 1.

2.3.2 n-bit functions

We can generalize the solution of the previous subsection to the n-bit case. Although the solution is almost exactly the same, it took seven years for Deutsch to generalize it. It is due to the fact that in Deutsch's original paper [8], it looked nothing like this circuit, and, thus, it was hard to generalize.

The generalization of the algorithm presented in the previous subsection is the following:

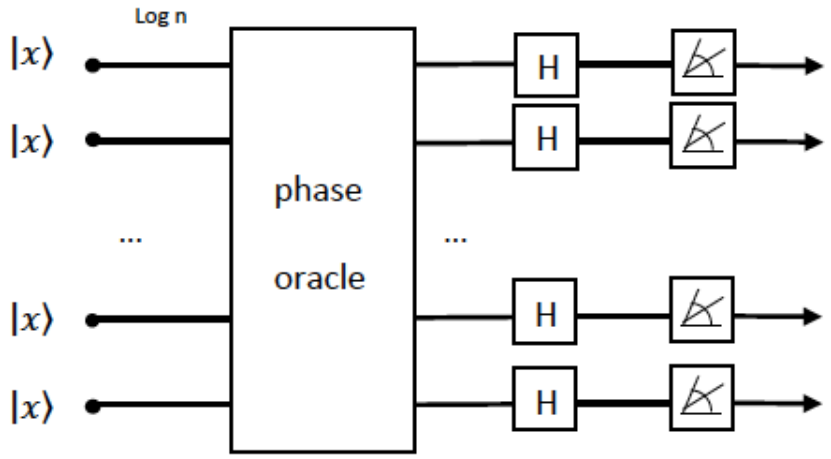

Figure 2.8: n-bit Deutsch-Jozsa model.

Now, analogously to the previous case, if output equals $|00\ldots0\rangle$, then f is constant. On the contrary, if output is different from $|00\ldots0\rangle$, f is balanced.

Let us explain this algorithm in some detail. We start with the zero state for n-qubits, $|00\ldots0\rangle = |0^n\rangle$, we apply a Hadamard gate to each qubit (in the previous picture encoded in the phase oracle), apply a query to everything (the phase oracle itself), then apply another Hadamard gate to each qubit, and, finally, measure the final state. Hence, this algorithm could be summarized as something like:

$$H^{\otimes n} O_{x,\pm} H^{\otimes n}.$$

Now we go step by step, seing what we get in each one of them. Initially, we have the state $|0^n\rangle$. After the first Hadarmard gate applied to each qubit, we get,

$$\frac{1}{\sqrt{2^n}} \sum_{i\in\{0,1\}^n} |i\rangle,$$

which is a uniform superposition of all possible i.

When we apply the query, the last term becomes:

$$\frac{1}{\sqrt{2^n}} \sum_{i\in\{0,1\}^n} (-1)^{x_i} |i\rangle,$$

and after the second Hadarmard gates, we have the final superposition

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

where we are denoting $i\cdot j=\sum_{k=1}^{n} i_k j_k$, as we have already mentioned before.

Finally, notice that $i\cdot 0^n=0$ for all $i\in\{0,1\}^n$. Thus, we can rewrite the first part of the last term as:

$$\frac{1}{2^n}\sum_{i\in\{0,1\}^n}(-1)^{x_i}=\begin{cases}1 & \text{if } x_i=0 \text{ for all } i,\\ -1 & \text{if } x_i=1 \text{ for all } i,\\ 0 & \text{if } x \text{ is balanced .}\end{cases}$$

Thus, the final term will yield $|0^n\rangle$ if x is constant and some other state is x is balanced. Therefore, the Deutsch-Jozsa problem can be solved using only 1 query and in $O(n)$ other operations, but this solution is not the original one of Deutsch and Jozsa, in which they used 2 queries instead of one.

Another equivalent way to present the Deutsch-Jozsa algorithm, extracted from [2] is Figure 2.9.

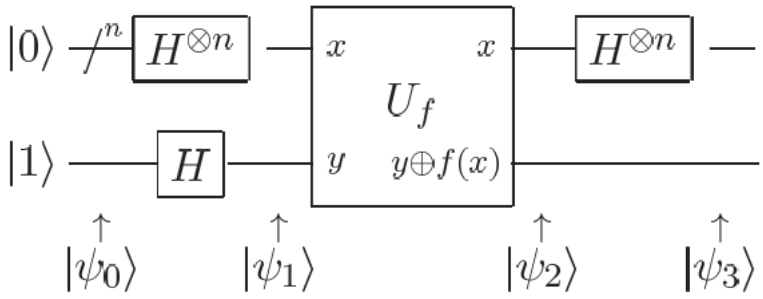

Figure 2.9: Another scheme for the n-bit Deutsch-Jozsa model.

In this case, the initial state is $|\psi_0\rangle=|0^n\rangle\,|1\rangle$. With this notation, the black box U_f performs the transformation $|x\rangle\,|y\rangle\to|x\rangle\,|y\oplus f(x)\rangle$ for $x\in\{0,1\}^n$ and $f(x)\in\{0,1\}$.

After the application of the first Hadarmard gates, one gets:

$$|\psi_1\rangle=\frac{1}{\sqrt{2^n}}\sum_{i\in\{0,1\}^n}|i\rangle\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right],$$

where the only difference with respect to the analogous term in the previous approach is the last term.

Now, after the application of the query, which is given by $U_f:|x,y\rangle\to|x,y\oplus f(x)\rangle$, one gets

$$|\psi_2\rangle=\frac{1}{\sqrt{2^n}}\sum_{i\in\{0,1\}^n}(-1)^{f(i)}\,|i\rangle\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right].$$

Then, if we apply the second Hadamard gates, we get:

$$|\psi_3\rangle=\frac{1}{2^n}\sum_{j\in\{0,1\}^n}\sum_{i\in\{0,1\}^n}(-1)^{i\cdot j+f(i)}\,|i\rangle\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right].$$

Finally, performing the same measurement as in the previous case, one gets the final output as required.

To see the improvement with respect to classical algorithms, notice that any classical deterministic algorithm needs, at least, $N/2 + 1$ queries. On the other hand, a classical algorithm can solve this problem efficiently if we allow a small error probability: Query x at two random positions, and output *constant* if they are the same and *balanced* otherwise. Then, the algorithm gets the correct answer with probability 1 if x is constant, and with probability $1/2$ when x is balanced.

Therefore, the difference between quantum and classical for this problem only appears if we consider algorithms without error probability.

2.3.3 Bernstein-Vazirani problem

In this subsection, we introduce the Bernstein-Vazirani problem.

Problem 2. *For $N = 2^n$, we are given an element $x \in \{0,1\}^N$ such that there is some $a \in \{0,1\}^n$ verifying:*

$$x_i \equiv i \cdot a \pmod 2 \qquad \text{for every } i,$$

where $i \cdot a$ denotes the usual scalar product between vectors, i.e.,

$$i \cdot a = \sum_{k=1}^{n} i_k a_k.$$

The goal is to find a.

Compared to the algorithms that we have just presented, it is easy to notice that, essentially, this algorithm is the same as the Deutsch-Jozsa algorithm. The main difference yields in the fact that, in this case, the final observation yields a.

By assumption,

$$x_i \equiv i \cdot a \pmod 2 \qquad \text{for every } i,$$

so

$$(-1)^{x_i} = (-1)^{(i \cdot a)}$$

and we can write the state obtained after the query in the following form:

$$\frac{1}{\sqrt{2^n}} \sum_{i \in \{0,1\}^n} (-1)^{x_i} |i\rangle = \frac{1}{\sqrt{2^n}} \sum_{i \in \{0,1\}^n} (-1)^{i \cdot a} |i\rangle.$$

We recall now that the Hadamard gate is its own inverse, since this implies that applying a Hadamard to each qubit of the above expression will turn it into $|a\rangle$. Hence, this solves the Bernstein-Vazirani problem with 1 query and $O(n)$ other operations.

Differently from the Deutsch-Jozsa algorithm, every classical algorithm needs n queries for information purposes, and the final answer consists of n bits and a classical query which contains, at most, 1 bit of information.

Remark 4. Bernstein and Vazirani also presented a recursive version of the problem stated above, which can be solved by a quantum algorithm in a polynomial number of steps. However, for this version, every classical randomized algorithm needs $n^{\Omega(\log n)}$ steps.

To finish this section, notice that the Deutsch-Jozsa problem has showed an exponential quantum improvement over the best deterministic classical algorithms, whereas the Bernstein-Vazirani problem has showed a polynomial improvement over the best randomized classical algorithms that have probability error $\leq 1/3$. In the following section, we will combine this two features, to obtain a problem where quantum computers are exponentially more efficient than bounded-error randomized algorithms.

2.4 Simon's algorithm

In this section, we change slightly the previous notation for convenience. Consider again $N = 2^n$ and denote by $[N] = \{1, \ldots, N\}$, which can be identified with $\{0,1\}^n$. Remember also that by $\oplus$ we are denoting the addition modulo 2 (or addition in $\mathbb{F}_{2^n}$). Then, Simon's problem can be stated as follows:

Problem 3. *For $N = 2^n$, we are given an element $x = (x_1, \ldots, x_N)$ with $x_i \in \{0,1\}^n$ for every i, such that there is some $s \in \{0,1\}^n$, non null and unknown, verifying:*

$$x_i = x_j \quad \textit{iff } i = j \textit{ or } i = j \oplus s \qquad \textit{for every } i.$$

The goal is to find s.

Notice that x can be seen as a function $x : [N] \to [N]$ 2-to-1 (two values of the domain are projected onto the same value in the codomain), where this fact is determined by s. Differently from the previous examples, the queries now are of the same form: The input $x = (x_1, \ldots, x_N)$ has variables that are not scalars, but strings of n numbers themselves, and one query produces completely such string $|i, 0^n\rangle \mapsto |i, x_i\rangle$.

This problem can also be seen in the following form: We can consider that we have $n2^n$ binary variables that we can query individually. Moreover, we can simulate one x_i-query using only n binary queries. This alternative view does not affect the number of queries too much.

The algorithm is quite similar to the Deutsch-Jozsa's one. Now, we start with $2n$ zero qubits of the following form:

$$|0^n\rangle\,|0^n\rangle$$

and apply Hadamard transforms just to the first n qubits, i.e., to $|0^n\rangle$, getting:

$$\frac{1}{\sqrt{2^n}} \sum_{i \in \{0,1\}^n} |i\rangle\,|0^n\rangle.$$

The second n qubits are still zero, but when we apply a query to the whole term, we get

$$\frac{1}{\sqrt{2^n}} \sum_{i \in \{0,1\}^n} |i\rangle\,|x_i\rangle.$$

At this point, the measurement algorithm is only applied to the second n collection of qubits. This measurement is not really necessary, but simplifies notably the posterior analysis. The outcome will be some value x_i, what implies that the first part also collapses to the superposition of the two indices having the x_i-value:

$$\frac{1}{\sqrt{2}}\left(|i\rangle + |i \oplus s\rangle\right)|x_i\rangle.$$

Once we have this term, we apply Hadamard gates just to the first n qubits again.

$$\frac{1}{\sqrt{2^{n+1}}}\left(\sum_{j\in\{0,1\}^n}(-1)^{i\cdot j}|j\rangle + \sum_{j\in\{0,1\}^n}(-1)^{(i\oplus s)\cdot j}|j\rangle\right),$$

and taking into account that the following property holds for the exclusive-or operation

$$(i \oplus s)\cdot j = (i\cdot j)\oplus(s\cdot j),$$

we can write the resulting state as:

$$\frac{1}{\sqrt{2^{n+1}}}\left(\sum_{j\in\{0,1\}^n}(-1)^{i\cdot j}(1+(-1)^{s\cdot j})|j\rangle\right).$$

For the final measurement, notice that $|j\rangle$ has non-zero amplitude iff $s\cdot j = 0 \bmod 2$, so measuring the state gives an element from the set $\{j|s\cdot j = 0 \bmod 2\}$. Hence, we are getting a linear equation that gives some information about s.

We can repeat now this algorithm until we get $n-1$ independent linear equations with information on s. The solutions to these equations will be 0^n and the right s (which can be computed by a classical algorithm).

All of this can be done by means of a classical circuit of size $O(n^3)$. However, Simon's algorithm finds s using $O(n)$ operations in the form of x_i queries and a polynomial amount of many other operations.

2.5 Fourier transform

In this section, we are going to present and develop the quantum Fourier transform, which is the key ingredient for quantum factoring and many other interesting quantum algorithms, as we will see in the following sections. The quantum Fourier transform is essentialy an efficient quantum algorithm for performing a Fourier transform of quantum mechanical amplitudes, and it appears in many different versions throughout classical computing, in areas ranging from signal-processing to data compression in complexity theory.

In its usual mathematical notation, the *discrete Fourier transform* takes as input a vector of complex numbers $x_0, \ldots x_{N-1}$, where the length of this vector, N, is a fixed parameter, and outputs another vector of complex numbers $y_0, \ldots, y_{N-1}$ defined by

$$y_k := \frac{1}{\sqrt{N}}\sum_{j=0}^{N-1} x_j e^{2\pi i j k/N}.$$

Notice that, in the previous expression, i stands for the imaginary unit.

The quantum Fourier transform is essentialy the same transformation, although the conventional notation for the quantum Fourier transform is a bit different. Consider an orthonormal basis $\{|0\rangle, \ldots, |N-1\rangle\}$. Then, the *quantum Fourier transform* is defined as the linear operator with the following action on the basis states:

$$|k\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i j k/N} |j\rangle.$$

Equivalently, by considering the action on an arbitrary state, we can write:

$$\sum_{j=0}^{N-1} x_j |j\rangle \mapsto \sum_{k=0}^{N-1} y_k |k\rangle,$$

where the *amplitudes* y_k are the discrete Fourier transform of the amplitudes x_j. Moreover, this definition is a unitary transformation, which implies that can be implemented as the dynamics for a quantum computer.

We can derive an equivalent expression for the quantum Fourier transform. For that, let $N = 2^n$ for some integer n and consider the basis $\{|0\rangle, \ldots, |2^n - 1\rangle\}$, which is, as mentioned previously, the computational basis for an n qubit quantum computer. For purposes of simplification of notation in the derivation of the aforementioned expression, let us write the state $|j\rangle$ using the binary representation $j = j_1 j_2 \ldots j_n$[1].

Then, after some calculations that we omit in this text for the sake of simplicity in the reading (we refer the reader to [2] for the specific algebra), the quantum Fourier transform can be given the following useful product representation:

$$|j_1, \ldots, j_n\rangle \mapsto \frac{\left(|0\rangle + e^{2\pi i 0, j_n} |1\rangle\right) \left(|0\rangle + e^{2\pi i 0, j_{n-1} j_n} |1\rangle\right) \left(|0\rangle + e^{2\pi i 0, j_1 \ldots j_n} |1\rangle\right)}{2^{n/2}}.$$

This product representation allows to construct an efficient quantum circuit to compute the Fourier transform (see Figure 2.10, extracted from [2]) and to prove that the quantum Fourier transform is unitary. It also provides insight into algorithms based upon the quantum Fourier transform.

An important fact to compare the quantum algorithm to compute the quantum Fourier transform with their classical analogues is the number of gates that the circuit of Figure

[1] Given a state $|j\rangle$, we say that its binary representation is given by

$$j = j_1 j_2 \ldots j_n,$$

or, more formally,

$$j = j_1 2^{n-1} + j_2 2^{n-2} + \ldots + j_n 2^0$$

if $|j\rangle$ is of the form

$$|j\rangle = |j_1\rangle \otimes |j_2\rangle \otimes \ldots |j_n\rangle.$$

Following this convention, it is also convenient to adopt the notation $0, j_1 \ldots j_n$ to represent the binary fraction

$$\frac{j_1}{2} + \ldots \frac{j_n}{2^n}.$$

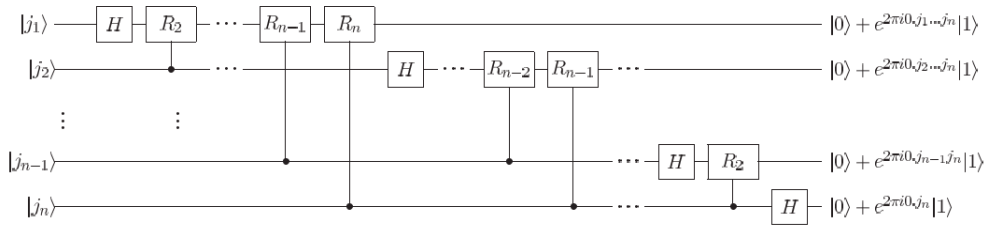

Figure 2.10: Quantum circuit to compute the quantum Fourier transform.

2.10 uses. The different steps of this circuit have the following numbers of gates, respectively:

1. One Hadamard gate and $n-1$ phase gates on the first qubit $\mapsto n$ gates.
2. One Hadamard gate and $n-2$ phase gates on the second qubit $\mapsto n-1$ gates.
3. ...

Hence, we see that

$$n + (n-1) + \ldots + 1 = \frac{n(n+1)}{2}$$

gates are required, plus the swap ones. Indeed, at most $n/2$ swaps are required, and each swap is accomplished using three CNOT gates.

Therefore, this circuit provides a $O(n^2)$ algorithm to perform the quantum Fourier transform. In contrast, the best classical algorithm for computing the discrete Fourier transform on 2^n elements are elements such as the *Fast Fourier Transform (FFT)*. This algorithm computes the discrete Fourier transform using $O(n2^n)$ gates. This implies that it is exponentially "worse" than its quantum analogue, meaning that it requires exponentially more operations to compute the Fourier transform on a classical computer than to implement the quantum Fourier transform on a quantum computer.

To finish this section, let us remark that, even though the previous comparison between classical and quantum algorithms to compute Fourier transforms, and the exponential improvement of the latter with respect to the first one, might allow us to think that this could have huge applications in real-world data processing applications (for example, in computer speech recognition, the first step in the recognition of phonemes is to perform the Fourier transform to the digitalized sound), in general one cannot use the quantum Fourier transform to speed up the computation of these Fourier transforms. This is due to the fact that amplitudes in a quantum computer cannot be directly accesed by a measurement and, thus, it is not possible to determine the Fourier transformed amplitudes of the original state. However, the quantum Fourier transform is still quite useful for some purposes related with some specific algorithms, as we will see in the following two sections.

2.6 Shor's quantum factoring algorithm

The most important quantum algorithm so far is probably Shor's algorithm. If we could use a quantum computer with a sufficiently large number of qubits and without succumbing to noise and other quantum decoherence phenomena, Shor's algorithm would break public-key cryptography schemes such as the widely used RSA scheme, which is based on the

assumption that factoring large numbers is computationally intractable. Currently it is known that this assumption is valid for classical (non-quantum) computers, since no classical algorithm is known that can factor in polynomial time. However, Shor's algorithm shows that factoring is efficient on an ideal quantum computer, so it may be feasible to defeat RSA by constructing a large quantum computer. Hence, it constitutes a powerful motivation for the design and construction of quantum computers and for the study of new quantum computer algorithms. Moreover, a variation of this algorithm can also be used to attack the schemes based on the discrete logarithm problem (in particular, the ones associated to elliptic curves).

We will devote this section to explain this algorithm and to introduce the necessary tools to understand the soluiton to the problem that it solves. Most of the images that appear on this section have been extracted from [21], a text that we will also follow in the proofs that appears here.

Roughly speaking, the statement of the problem of factoring integer, in which Shor's algorithm can be applied, is as follows.

Problem 4. *Let N be a natural number. The goal is to find $e_1, \dots e_n$ natural numbers and $p_1, \dots, p_n$ prime numbers so that*

$$N = p_1^{e_1} \cdot \ldots \cdot p_n^{e_n}.$$

Notice that this decomposition of N in prime factors is unique.

We can do two initial simplifications to the problem without loss of generality. The first one is that, given N, it is enough to split it into two other integers N_1 and N_2 such that $N = N_1 \cdot N_2$. Hence, after a linear number in size of the input of such steps, we are guaranteed to reach prime factors, thus the second simplification is that we can assume that N is a product of two primes, $N = p \cdot q$.

Classically, there exist some naive algorithms for the factoring problem that work in time $O(\sqrt{N})$. The fastest known algorithm for this problem is called *Field Sieve algorithm*, and it works in time $2^{O(\sqrt[3]{\log N})}$.

Shor's result yields the fact that one can do better with a quantum computer.

Theorem 1. *There exists quantum algorithm that solves the factoring problem with bounded error probability in polynomial time.*

We will devote most of this section to prove this result, following several steps for that. First, we will show that the factoring problem is equivalent to the *order-finding problem*, since a fast algorithm for order-finding problem implies a fast algorithm for factoring problem. Later, we will present a fast quantum algorithm for order-finding.

2.6.1 Reduction of factoring to order-finding

First, notice that the set of modulo N coprime numbers with N, i.e.,

$$\{1 \leq x \leq N \, : \, \gcd(x, N) = 1\}$$

where $\gcd(x, N)$ denotes the *greatest common divisor* of x and N, forms a group under multiplication modulo N. Moreover, given x and N such that $\gcd(x, N) = 1$, we define the order of x by the minimum positive r such that $x^r \equiv 1 (\text{mod } N)$, and denote it by $\text{ord}(x)$.

We can now state the order-finding problem.

Problem 5. *Given x and N such that $gcd(x, N) = 1$, the **order-finding problem** consists of finding $ord(x)$.*

We will devote the rest of the subsection to prove that Problem 4 can be reduced to Problem 5.

For that, let us first introduce and prove three technical lemmata. The first and last one will be essential to obtain the desired result, whereas the second one will be used in the proof of the third one.

Lemma 2. *Given a composite number N and a number x that is a nontrivial square root of 1 modulo N, i.e., verifying $x^2 \equiv 1 (mod\ N)$ with neither $x \equiv 1 (mod\ N)$ nor $x \equiv -1 (mod\ N)$ holding, we can efficiently compute a nontrivial factor of N.*

Proof. Since, by assumption, $x^2 \equiv 1 (\text{mod } N)$, we have:

$$x^2 - 1 \equiv (x-1)(x+1) \equiv 0 (\text{mod } N).$$

Now, as none of

$$x \equiv 1 (\text{mod } N), \qquad x \equiv -1 (\text{mod } N)$$

hold, we know that $1 < x < N - 1$, and, hence, both $\gcd(x-1, N)$ and $\gcd(x+1, N)$ are nontrivial factors of N.

Finally, since there exists a fast algorithm for computing the gcd of two numbers, *Euclid's algorithm*, the efficiency follows from there.

□

Now, the next lemma reads as follows:

Lemma 3. *Let p be an odd prime and let x be a uniformly random element verifying $0 \leq x < p$. Then, $ord(x)$ is even with probability, at least, $1/2$.*

Proof. In virtue of *Fermat's little theorem*, it is clear that, for every x, the following holds:

$$x^{p-1} \equiv 1 (\text{mod } p).$$

Another well-known result in mathematics is that the multiplicative group modulo a prime number is a cyclic group, i.e., there is an element g (from '*generator*') that generates all the elements of the group, in the sense that any element can be written by

$$x \equiv g^k (\text{mod } p)$$

for some k. Since x is chosen uniformly at random, k is odd with probability $1/2$.

Assume now that k is odd. Then, since $x \equiv g^k (\text{mod } p)$, we have

$$x^{\text{ord}(x)} \equiv g^{k\ \text{ord}(x)} \equiv 1 (\text{mod } p).$$

From here, we can deduce

$$p - 1 | k\ \text{ord}(x),$$

and, thus, since both p and k are odd (and $p-1$ is even), $\text{ord}(x)$ has to be even.

To sum up, if k is odd, something that happens with probability $1/2$, $\text{ord}(x)$ is even. If k is even, however, we cannot say anything on $\text{ord}(x)$ in general. Putting both facts together, $\text{ord}(x)$ is even with probability, at least, $1/2$.

□

The last lemma that we present in this section, and in whose proof we use the previous one, is the following:

Lemma 4. *Let $N = p \cdot q$, with p and q prime numbers, and assume that N is an odd number. Let us further assume that x is taken uniformly at random from $0, \ldots, N-1$.*

If $gcd(x, N) = 1$, then with probability at least $3/8$, we have that $ord(x) = r$ is even and $x^{r/2} \neq \pm 1 (mod\ N)$.

Proof. In virtue of the *Chinese remainder theorem*, choosing x uniformly at random from $0, \ldots, N-1$ is the same that choosing x_1 uniformly at random from $0, \ldots p-1$ and, independently, x_2 uniformly at random from $0, \ldots q-1$. Moreover, if we denote $r_1 = \text{ord}(x_1)$ and $r_2 = \text{ord}(x_2)$, we can see that $r_1 | r$ and $r_2 | r$.

Now, we can first see that the probability that r is even is, at least $3/4$. Since we are assuming that N is an odd number, it is clear that both p and q also have to be odd. If x_1 is odd, then r_1 has to be even, and the same happens for x_2 and r_2. Hence, since r_1 even or r_2 even imply that r is even, and x_1 and x_2 are chosen uniformly at random, applying Lemma 3 we get that the probability that r is even is at least $3/4$.

Finally, we can prove that the probability of $x^{r/2} \equiv \pm 1 (\text{mod } N)$ is, at most, $1/2$ when r is even. Indeed, notice that, under this assumption, $x^r \equiv 1 (\text{mod } p)$ and there are only two square roots of 1 modulo a prime number, namely ± 1. Again in virtue of the Chinese remainder theorem, it follows that there are only four roots of 1 modulo N. Only two of them make $x^{r/2} \neq \pm 1 (\text{mod } N)$.

□

In the last part of the subsection, we can see that the reduction from Problem 4 to Problem 5 follows directly from Lemma 2 and 4. Indeed, if someone computes the function $\text{ord}(\cdot)$ for us, the prime factors of N can be found classically. By checking the answer, something that can be done efficiently easily, and repeating the procedure several times, we can increase the probability of success.

2.6.2 The order-finding problem

Now that we have seen that the problem of efficiently factoring a number can be reduced to the problem of efficiently finding $\text{ord}(x) = r$, we are going to find an example for the second problem. This example is Shor's algorithm, and we devote this subsection to analyze it.

First, we will start with a simplified case, before going to the more general case.

Simplified case

In this case, we use an auxiliary number Q, which is sufficiently large, and which verifies $Q >> N^2$, and assume that $r | Q$. The case when $r \nmid Q$ does not differ drastically to the one we are studying, so we will restrict for the moment to that for the simplicity of notation, and we will proceed to the more difficult case in the next section.

The algorithm that we are going to present uses two registers:

- *Register 1* stores a number (mod $Q = 2^q$).
- *Register 2* stores a number (mod N).

It also has several steps, that we are going to present individually:

1. The registers are initially in the state $|0\rangle \otimes |0\rangle$.
2. We apply the Fourier Transform modulo Q to the first qubit, to get the state
$$\frac{1}{\sqrt{Q}} \sum_{a=0}^{Q-1} |a\rangle \otimes |0\rangle.$$
3. Consider now the function $f(a) = x^a (\text{mod } N)$, a function that we can easilly compute classically, and has r as its smallest order (or *period*). Notice that this function can be computed in $\log a$ multiplications, and also that f is different in $[0, r-1]$ (otherwise, it would have a smaller period). Applying then f to the state obtained for the first qubit in the previous step, we get
$$\frac{1}{\sqrt{Q}} \sum_{a=0}^{Q-1} |a\rangle |f(a)\rangle.$$
4. At this point, we measure the second qubit, and when we perform this measurement, the second register collapses to some value, $f(k)$, for k uniformly random over $0, \dots r-1$. Then, all superposed states which are inconsistent with the measured value must dissapear. Hence, the state of the two registers must be given by
$$\frac{1}{\sqrt{\frac{Q}{r}}} \sum_{a=0}^{\frac{Q}{r}-1} |ar + k\rangle |f(k)\rangle.$$
5. We have set up a periodic superposition of period r in the first register (the value that we wanted to compute), so we can drop the second register. At this moment, Shor's algorithm comes into play, by Fourier sampling modulo Q.

   As we are going to perform Fourier sampling, we can drop the shift value k by the properties of Fourier Transforms seen in Section 2.5. This allows us to move k to phase. Then, applying the Fourier sample to the state
$$\frac{1}{\sqrt{\frac{Q}{r}}} \sum_{a=0}^{\frac{Q}{r}-1} |ar + k\rangle,$$
   we get
$$\frac{1}{\sqrt{r}} \sum_{a=0}^{r-1} \omega^{ak} \left| a\frac{Q}{r} \right\rangle,$$
   where ω is a primite qth root of unity, i.e.,

$$\omega = e^{\frac{2\pi i}{Q}}.$$

6. Now, we measure this register. The measurement provides $\left|a\frac{Q}{r}\right\rangle$, where a is a random variable uniformly from $0, \ldots, r-1$. It is easy to see then that with great probability we have $\gcd\left(a, \frac{Q}{r}\right) = 1$. If that is the case, then by computing $\gcd\left(a\frac{Q}{r}, Q\right)$ we shoul get $\frac{Q}{r}$. Since we already know Q, it is clear that, from $\frac{Q}{r}$, computing r is straightforward.

General case

In the previous subsubsection we have assumed $r|Q$. As we said in its introduction, we will devote this subsubsection to the case $r \nmid Q$, which is a bit more difficult in notation, but the same in spirit.

Let us perform the same algorithm than before up to Step 3. From Step 4 on, the situation is a bit different. After applying the first measurement, we get the state:

$$\frac{1}{\sqrt{\lfloor \frac{Q}{r} \rfloor}} \sum_{a=0}^{\lfloor \frac{Q}{r} \rfloor - 1} |ar + k\rangle.$$

In this case, this is not a coset of a subgroup, so we cannot proceed as in the previous case. However, we can anyway take the Fourier transform, and we will show that we get a constructive interference primarily at the points which are close to multiples of $\frac{Q}{r}$, so close that they can be 'rounded' to the nearest multiple. This is, in summary, the fact that will allow us to calculate r with reasonable probability.

Let us explain that step by step. First, if we apply a Fourier transform to the previous expression, we get

$$\sum_{k=0}^{Q-1} \alpha_k |k\rangle,$$

where the coeficient α_k is given by:

$$\alpha_k = \frac{1}{\sqrt{Q}} \cdot \frac{1}{\sqrt{\lfloor \frac{Q}{r} \rfloor}} \sum_{a=0}^{\lfloor \frac{Q}{r} \rfloor - 1} \left(\omega^{rk}\right)^a.$$

One can notice that if Q is small, then terms in the sum cover only a small angle of the complex plane, and hence the magnitude of the sum is almost the sum of the magnitudes. We will see this fact explicit in a couple of lemmas that appear at the end of this section, where we will prove that, with probability more than $1/16$, we can sample a k such that

$$-\frac{r}{2} \leq kr (\text{mod } Q) \leq \frac{r}{2}.$$

Meanwhile, let us just assume that this fact is true and continue with the algorihtm. First, notice that the previous expression is equivalent to

$$|kr - lQ| \leq \frac{r}{2}$$

for a certain integer l, or what is the same,

$$\left|\frac{k}{Q}-\frac{l}{r}\right| \leq \frac{1}{2Q}.$$

This condition implies that $\frac{k}{Q}$ is a $\frac{1}{2Q}$-approximation of $\frac{l}{r}$. Moreover, if we measure k, we get to know Q, so we know the quotient, which constitutes the good approximation of $\frac{l}{r}$.

Furthermore, notice that l is randomly chosen from $[0, r-1]$, which implies that, with probability at least $1/\log l$, l and r are coprimes, which allows us to compute r from l/r. Then, in principle, if we are able to choose Q much larger than N, this way of reasoning provides a good approximation. To see how much larger than N needs to be Q we use *continued fractions*[2]. Indeed, we just have to compute continued fractions until we get precision of at least $\frac{1}{2Q}$. Since l/r is rational, we know that, for a certain integer n, the continued fractions verify $CF_m(l/r) = l/r$ for every $m \geq n$. Hence, if we assume that the approximation is some rational number l'/r', clearly $r = r'$. Otherwise, we would have

$$\left|\frac{l}{r}-\frac{l'}{r'}\right| \geq \frac{1}{rr'} \geq \frac{1}{N^2},$$

and this would be a contradiction, since both $\frac{l}{r}$ and $\frac{l'}{r'}$ are $\frac{1}{2Q} \leq \frac{1}{2N^2}$ close to the same rational fraction (the one that approximates $\frac{k}{Q}$ from the beginning).

Therefore, $r = r'$, so by using these continued fractions we have finished the algorithm.

In the last part of this section, we will state and prove a couple of lemmas that have been used in the development of the algorithm.

Lemma 5. *If $-\frac{r}{2} \leq kr(mod\ Q) \leq \frac{r}{2}$ for some kr, then*

$$|\alpha_k| \geq \frac{1}{2^{2/3}\sqrt{r}}.$$

Proof. Let us denote

$$\beta = e^{\frac{2\pi i r k}{Q}a} = \omega^{rk}.$$

[2]The idea for this part of the proof lies in the use of continued fractions to approximate real numbers using finite numbers of integers.

A *continued fraction* is defined in the following way: A real number α can be approximated by a set of positive integers $a_0, a_1 \ldots a_n$ by

$$CF_n(\alpha) = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ldots + \cfrac{1}{a_n}}}} = \frac{A_n}{B_n},$$

where the numbers a_i are chosen in a specific way, and A_n and B_n are always integers. To explain the procedure to chose these numbers, let us consider π and suppose that we want to approximate it with four decimals. Then, we get:

$$\pi \approx 3,1415 = 3 + \frac{1415}{10000} = 3 + \cfrac{1}{7 + \frac{95}{1415}} = 3 + \cfrac{1}{7 + \cfrac{1}{14 + \frac{85}{95}}}.$$

Moreover, continued fractions satisfy the following two important properties, which we are essential for the last step of the algorithm:

1. $CF_n(\alpha)$ is the best rational approximation of α with denominator $\leq B_n$.
2. If α is rational, then it coincides, from some n on, with the approximations $CF_n(\alpha)$.

Finally, it is easy to notice that continued fractions are easily computable for any rational number.

It is clear that this expression corresponds to a vector of the complex plane. Also, the sum that appears in the term α_k, i.e.,

$$\sum_{a=0}^{\lfloor \frac{Q}{r} \rfloor - 1} \beta^a$$

is a geometric series of ratio β.

Because of the assumption of the statement of the lemma $\left(-\frac{r}{2} \leq kr(\text{mod } Q) \leq \frac{r}{2}\right)$, the terms of the series cover less than or equal to an angle π on the complex plane (since β makes a small angle with the real line). Then, as we can see in Figure 2.11,

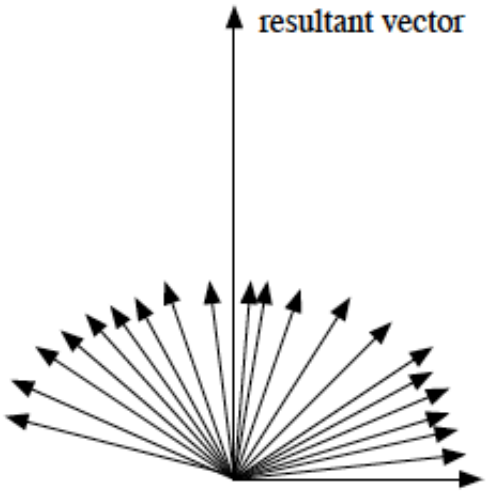

Figure 2.11: β makes a small angle with the real line.

half of the terms on the previous series make an angle which is less than or equal to $\frac{\pi}{4}$ with the resultant vector of the addition of the terms in the series. Then, since the cosinus is a decreasing function from 0 to $\pi/4$, each term contributes with a fraction that is at least:

$$\cos\left(\frac{\pi}{4}\right) = \frac{1}{\sqrt{2}}$$

of its length to the resultant vector. Hence, the magnitude of the resultant is, at least:

$$\frac{1}{2} \cdot \frac{1}{\sqrt{2}} \cdot \frac{Q}{r} \cdot \frac{1}{\sqrt{Q}} \frac{1}{\sqrt{\lfloor \frac{Q}{r} \rfloor}} = \frac{1}{2^{3/2}} \cdot \frac{1}{\sqrt{r}}.$$

□

Lemma 6. *The fact of the statement of the previous lemma, i.e.,*

$$-\frac{r}{2} \leq kr(mod\ Q) \leq \frac{r}{2}$$

happens with probability $P(X \leq 1)$, where X is a random variable that follows a distribution $\mathcal{N}(0,1)$.

Proof. If $\gcd(r, Q) = 1$, then the element $r^{-1} (\text{mod } Q)$ exists. Thus, as the variable k varies in the range $[0, Q-1]$, $k \cdot r$ must take values that constitute a permutation of $\{0, 1, \ldots, Q-1\}$. As we can see in the following picture,

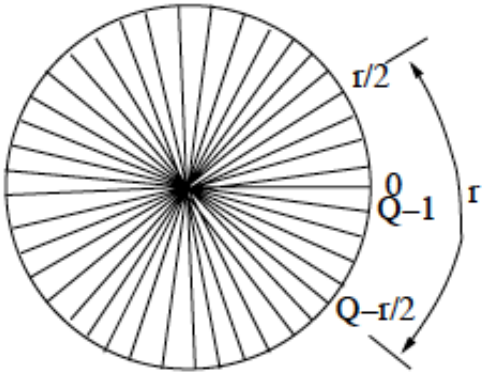

Figure 2.12: At least r values of kr lie in the range $[Q - r/2, r/2]$.

at least r values of kr lie in the range $[Q - r/2, r/2]$.

Now, assume that $\gcd(r, Q) \neq 1$. In this case, the distribution of $k \cdot r \pmod{Q}$ is a bit different, and can be shown in the following picture:

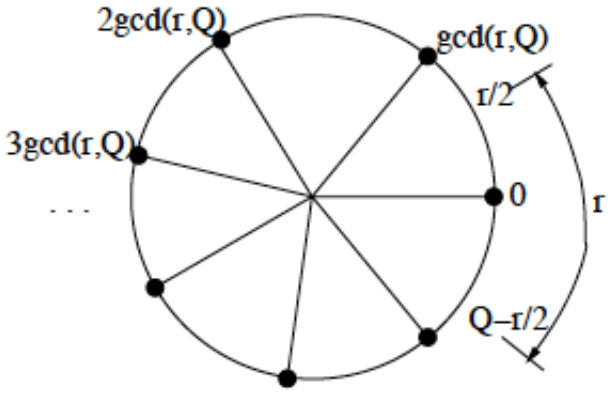

Figure 2.13: At least $r/2$ values of kr lie in the range $[Q - r/2, r/2]$.

In this case, as we can see, at least $r/2$ values of $k \cdot r$ lie in the desired range $[Q - r/2, r/2]$.

Hence, if we consider both cases together, we have that, in any case, at least $r/2$ values of kr lie in the range $[Q - r/2, r/2]$, thus satisfy the condition:

$$-\frac{r}{2} \leq kr (\text{mod } Q) \leq \frac{r}{2}.$$

In virtue of Lemma 5, since each one of them has an amplitude which is at least

$$\frac{1}{2^{3/2} r^{1/2}},$$

we get that the probability to get such a k when sampling is, at least,

$$\frac{r}{2} \left(\frac{1}{2^{3/2} r^{1/2}} \right)^2$$

and this is greater than 1/16. Therefore, the condition on the statement of the lemma happens with probability greater than $P(X \leq 1)$ for a normal random variable.

□

To conclude this section, let us just recall that, to factor and integer N, Shor's algorithm runs in polynomial time. More specifically, it takes quantum gates of order $O((\log N)^2(\log\log N)(\log\log\log N))$ using fast multiplication, and, thus, the integer factorization problem can be efficiently solved on a quantum computer (and belongs to the complexity class **BQP**). This proves Theorem 1.

For the sake of simplicity, we can briefly summarize the whole factorization of N in the following steps:

1. A reduction of the factoring problem to the problem of order-finding (this can be done on a classical computer).
2. A quantum algorithm to solve the order-finding problem.

   Let r be the period that we want to find.

   (a) Initialize the registers in the state $|0\rangle \otimes |0\rangle$.

   (b) Apply the Fourier Transform modulo Q to the first qubit, to get the state

   $$\frac{1}{\sqrt{Q}} \sum_{a=0}^{Q-1} |a\rangle \otimes |0\rangle.$$

   (c) Consider $f(a) = x^a (\text{mod } N)$ and apply it to the state:

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

   (d) Measure the second qubit:

   $$\frac{1}{\sqrt{\lfloor \frac{Q}{r} \rfloor}} \sum_{a=0}^{\lfloor \frac{Q}{r} \rfloor - 1} |ar + k\rangle.$$

   (e) Apply Fourier transform:

   $$\sum_{k=0}^{Q-1} \alpha_k |k\rangle, \text{ where } \alpha_k = \frac{1}{\sqrt{Q}} \cdot \frac{1}{\sqrt{\lfloor \frac{Q}{r} \rfloor}} \sum_{a=0}^{\lfloor \frac{Q}{r} \rfloor - 1} \left(\omega^{rk}\right)^a.$$

   (f) With probability more than 1/16, we can sample a k such that

   $$-\frac{r}{2} \leq kr (\text{mod } Q) \leq \frac{r}{2}.$$

   (g) The previous fact implies that $\frac{k}{Q}$ is a $\frac{1}{2Q}$-approximation of $\frac{l}{r}$. Using continued fractions, we get r from l/r.

This is much faster than the most efficient known classical factoring algorithm, the *Field Sieve algorithm*, that works in time $2^{O(\sqrt[3]{\log N})}$. The efficiency of Shor's algorithm, as seen before, is due to the quantum Fourier transform and the modular exponentiation.

2.7 Grover's algorithm

The second most important quantum algorithm, after Shor's, is Grover's quantum search problem [10]. Attacks based on this algorithm can be used to break cryptographic schemes in symmetric key, such as AES, as well as schemes based on hash functions, such as SHA2 or SHA3. We will discuss this in more detail in the next chapter.

In general, as mentioned in previous sections, much of the excitement concerning quantum computation comes from the fact that quantum algorithms can provide improvements over classical algorithms. In this particular case, Grover's algorithm exhibits a quadratic speedup with respect to the classical case. Although it does not provide exponential speedup, as Shor's did, it is much more applicable.

Problem 6. *For $N = 2^n$, consider an arbitrary $x \in \{0,1\}^N$. The goal is to find $i \leq N$ such that $x_i = 1$, and to output 'no solutions' if such i does not exist.*
This problem is called the unstructured search problem.

It can be formulated equivalently as a database search problem, in which we consider a database and we want to find an item in it which fulfills certain specifications. One example of this can be a database of N names and we want to find the position of a specific name in it.

We call this search "unstructured" because we are not given any information about how the database is ordered. If, for example, we knew in advance that the database was sorted, then we could perform a binary search and find any element in logarithmic time. However, since that is not the case, with classical circuits one cannot do better than performing a linear number of queries to find the target element.

However, in the quantum setting, Grover's algorithm leads to the following theorem:

Theorem 7. *The unstructured search problem can be solved in $O(\sqrt{N})$ queries using quantum computation (and $O(\sqrt{N}\log N)$ other gates).*

In fact, it was shown in [13] that, up to a constant factor, this is the best that one can do for this problem under the quantum computational model, since the query complexity is $\Theta(\sqrt{N})$. In particular, since the unstructured search problem cannot be solved in logarithmic time, this problem cannot be used as a way to solve NP problems in polynomial time. However, there is still a quadratic improvement with respect to the classical case.

2.7.1 Algorithm

Let us consider the following notation introduced in Section 2.2:

$$O_{x,\pm}\,|i\rangle = (-1)^{x_i}\,|i\rangle$$

to denote the $\pm$-type oracle for the input x. Let us also denote by R the unitary transformation that puts a phase -1 in front of all basis states which are different from $|0\rangle$:

$$R\,|i\rangle = \begin{cases} -\,|i\rangle & \text{if } |i\rangle \neq |0\rangle \\ |i\rangle & \text{if } |i\rangle = |0\rangle \end{cases}$$

This transformation R is clearly independent of the input x for the algorithm, and can be implemented using $O(n)$ elementary gates.

Now, let us define by *Grover iterate* the following quantity:

$$\mathcal{G} = H^{\otimes n} R H^{\otimes n} O_{x,\pm},$$

where we recall that H denotes the Hadamard gate. Notice that 1 Grover iterate provides 1 query.

Let us develop now each one of the steps of the algorithm:

1. Grover's algorithm starts with the n-bit state $|0^n\rangle$.
2. Later, it applies a Hadamard transformation to all the qubits, and gets the uniform superposition

$$|U\rangle = \frac{1}{\sqrt{N}} \sum_i |i\rangle$$

of all the indices N.

3. Now, it applies the Grover iterate $\mathcal{G}$ to this state k times, where this k will be conveniently chosen later.
4. Finally, it measures the final state.

We can see this algorithm represented graphically in Figure 2.14.

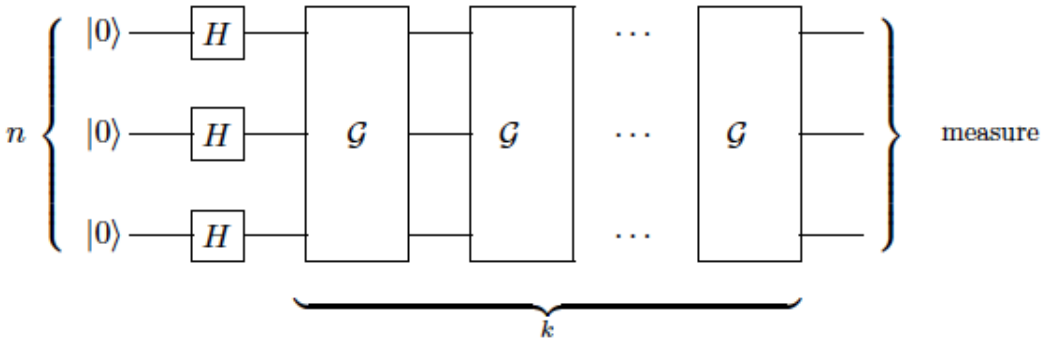

Figure 2.14: Graphical representation of Grover's algorithm.

Intuitively, the idea behind this algorithm is that in each iteration some amplitude is moved from the indices of the 0-bits to the ones of the 1-bits. Hence, the algorithm stops when nearly all the amplitude is on the indices of the 1-bits, and in this case the measurement of the final state will probably give the index of a 1-bit.

Let us now analyze this explicitly. For that, we define the following states:

$$|G\rangle := \frac{1}{\sqrt{t}} \sum_{i\,:\,x_i=1} |i\rangle\,, \qquad |B\rangle := \frac{1}{\sqrt{N-t}} \sum_{i\,:\,x_i=0} |i\rangle,$$

where $t = \#\{i : x_i = 1\}$. The first one provides a superposition of the basis states for whose index i we have $x_i = 1$, and the second one the same when $x_i = 0$. Then, it is clear that we can write the uniform superposition of the second state of the algorithm as:

$$|U\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle = \sin(\theta)\,|G\rangle + \cos(\theta)\,|B\rangle\,,$$

for a certain θ that is given by

$$\theta = \arcsin\left(\sqrt{\frac{t}{N}}\right).$$

To determine the value of k, notice that Grover iterate $\mathcal{G}$ is actually the product of two reflections in the 2-dimensional space spanned by $|G\rangle$ and $|B\rangle$. Indeed, since a *reflection* in a subspace $W \subset V$ is a unitary A such that $Av = v$ for every $v \in V$, and $Aw = -w$ for every w orthogonal to V, we can easily see that

- $O_{x,\pm}$ is a reflection through $|B\rangle$.
- $H^{\otimes n}RH^{\otimes n}$ is a reflection through $|U\rangle$.

  Indeed, the following is satisfied:

$$H^{\otimes n}RH^{\otimes n} = H^{\otimes n}(2\,|0^n\rangle\,\langle 0^n| - \mathbb{1})H^{\otimes n} = 2\,|U\rangle\,\langle U| - \mathbb{1}.$$

Hence, we can restate Grover's algorithm as follows, assuming that we know that the fraction of solutions is $\varepsilon = t/N$:

1. Grover's algorithm starts with the n-bit state $|0^n\rangle$.
2. Later, it applies a Hadamard transformation to all the qubits, and gets the uniform superposition

$$|U\rangle = \frac{1}{\sqrt{N}}\sum_i |i\rangle$$

   of all the indices N.
3. Now, it repeats the following k times, where $k = O(1/\sqrt{\varepsilon})$:
   (a) It reflects through $|B\rangle$, i.e., it applies $O_{x,\pm}$.
   (b) It reflects through $|U\rangle$, i.e., it applies $H^{\otimes n}RH^{\otimes n}$.
4. Finally, it measures the first register and checks that the resulting i is a solution.

2.7.2 Geometrical proof of the algorithm

In this subsection, we are going to show that the algorithm described above indeed works. This geometrical argument has been extracted from [5].

Let us consider the 2-dimensioanl real plane spanned by $|G\rangle$ and $|B\rangle$. Consider

$$|U\rangle = \sin(\theta)\,|G\rangle + \cos(\theta)\,|B\rangle.$$

Then, if we consider the two reflections mentioned on the third step of the algorithm above, we can see that the initial angle increases from θ to 3θ, moving us towards $|G\rangle$, as we can see in Figure 2.15.

For the next two reflections, the angle increases with another factor 2θ. Hence, after k applications of the two reflections, the initial state has been transformed to

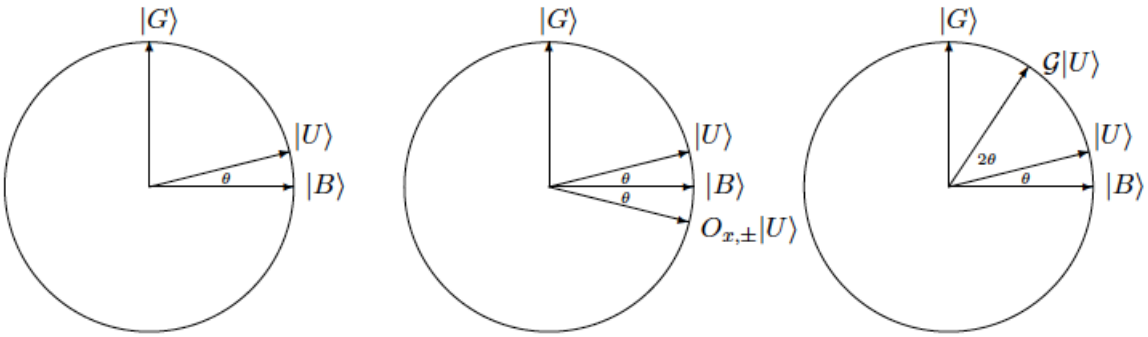

Figure 2.15: First iteration of the third step of Grover's algorithm. In the first picture, it starts with $|U\rangle$; in the second one, it reflects through $|B\rangle$ to get $O_{x,\pm}|U\rangle$; and in the last one, it reflects through $|U\rangle$ to get $\mathcal{G}|U\rangle$.

$$\sin((2k+1)\theta)\,|G\rangle + \cos((2k+1)\theta)\,|B\rangle.$$

If we now measure this state, it is clear that the probability of seeing a solution is

$$P_k = \sin((2k+1)\theta)^2,$$

and we are interested on this P_k being as close to 1 as possible. Notice that, for example, if we choose $\tilde{k} = \pi/4\theta - 1/2$, so $(2\tilde{k}+1)\theta = \pi/2$ and, thus, $P_{\tilde{k}} = \sin(\pi/2)^2 = 1$.

One example for these quantities can be:

$$t = N/4, \qquad \theta = \pi/6, \qquad \tilde{k} = 1.$$

Unfortunately, in general, $\tilde{k} = \pi/4\theta - 1/2$ is not an integer, and we can only consider an integer number of applications of the Grover iterate. Therefore, our aim will be to choose k as the closest possible to $\tilde{k}$ integer, and in this scenario our final state will still be close to $|G\rangle$.

The failure probability is still small, as we can see below (assuming $t << N$):

$$\begin{aligned}
1 - P_k &= \cos((2k+1)\theta)^2 \\
&= \cos\Big((2\tilde{k}+1)\theta + 2(k-\tilde{k})\Big)^2 \\
&= \cos\Big(\pi/2\theta + 2(k-\tilde{k})\Big)^2 \\
&= \sin\Big(2(k-\tilde{k})\Big)^2 \\
&\leq \sin(\theta)^2 \\
&= \frac{t}{N},
\end{aligned}$$

where we have used $\left|k - \tilde{k}\right| \leq 1/2$ in the inequality. Finally, since $\arcsin(\theta) \geq \theta$, the number of queries is

$$k \leq \frac{\pi}{4\theta} \leq \frac{\pi}{4}\sqrt{\frac{N}{t}}.$$

2.7.3 Amplitude amplification

The analysis worked above can be extended to a much more general setting. To see that, consider $\xi : \mathbb{Z} \to \{0, 1\}$ any Boolean function. We will call *solutions* the inputs $z \in \mathbb{Z}$ satisfying $\xi(z) = 1$.

Suppose now that we have an algorithm that checks whether z is a solution or not. Let us denote this by:

$$O_\xi(|z\rangle) = (-1)^{\xi(z)} |z\rangle.$$

We further assume that we have some quantum or classical algorithm, denoted by $\mathcal{A}$, that uses no intermediate measurements and, when applied to $|0\rangle$, has probability p of finding a solution. Hence, the *amplitude amplification* algorithm only needs to run the algorithm $\mathcal{A}$ $O(1/\sqrt{p})$ times. This algorithm is implemented as:

1. Consider the starting state $|U\rangle = \mathcal{A}|0\rangle$.
2. It repeats the following $O(1/\sqrt{p})$ times:
   (a) It reflects through $|B\rangle$, i.e., it applies O_ξ.
   (b) It reflects through $|U\rangle$, i.e., it applies $\mathcal{A}R\mathcal{A}^{-1}$.
3. It measures the first register and checks that the resulting element x is marked.

We can now define $\theta = \arcsin\left(\sqrt{p}\right)$ and $|G\rangle$ and $|B\rangle$ as we did before, within the geometric argument for Grover's algorithm. Hence, an analogous reasoning will show that the amplitude amplification finds a solution with probability close to 1.

Therefore, we can speedup many heuristic classical algorithms by this procedure. In general, any algorithm that has non-trivial probability of finding a solution can have this probability amplified to nearly 1.

Finally, notice that Grover's algorithm is a special case of amplitude amplification, where $O_\xi = O_x$ and $\mathcal{A} = H^{\otimes n}$.

Chapter 3

Introduction to post-quantum cryptography

In the previous chapter, we presented Shor's algorithm (Section 2.6), which is one of the most important examples that quantum computers would have a great impact on the security of many cryptographic scheme. Indeed, this algorithm, used on a quantum computer, would allow to factor numbers and compute discrete logarithms inside abelian groups in polynomial time. Since many public key schemes nowadays are built upon the assumption that an enormous number cannot be factorized efficiently, the existence of quantum computer would be an important problem for the security of almost all our digital communications.

In this chapter, we are going to present an introduction to *post-quantum cryptography*, a new line of research that has appeared due to the situation aforementioned. It is also known as *quantum-resistant cryptography*, and its main objective is to develop new cryptographic schemes that would resist attacks from quantum computers (and algorithms implemented on them). Due to the appearance of the first prototypes of quantum computers of a small number of qubits ([36], [37], [38], etc) and the possibility of (hopefully) commercializing functional quantum computers in the next decades, the NIST [35] has announced a competition to promote fundamental research on this new field by looking for an standard (or several) quantum-resistant public-key cryptographic algorithms. Indeed, the development of these new cryptographic schemes is a long-time process, and that is the reason to start the development of the field soon enough, even though quantum computers are still far from becoming an actual threaten.

This chapter will consist of two well-differentiated parts. In the first one, we will study "quantum attacks", as an extension of the last sections of the previous chapter, presenting a couple of works where we can see how quantum algorithms could actually attack public-key security. In the last part of the chapter, and of the project, we will show a brief introduction to the field of post-quantum cryptography, presenting a basic description of the different kinds of schemes that are studied in this field and a summary on the competition promoted by NIST.

3.1 Quantum attacks

This section is mainly based in the works [19] and [20], as well as in the quantum algorithms presented in the previous chapter. As mentioned in the previous chapter, the existence of quantum computers could have a major impact on the security of many cryptographic

schemes that we currently use. In Section 2.6, we saw that those computers could factor integer numbers in polynomial time, and since almost all public key schemes are based on the assumption that such problems are intractable, the development of quantum computers is compromising the security of most our digital transactions and communications, for instance.

This situation has motivated the appearance of post-quantum cryptography, and the NIST competition, to start developping new cryptographic schemes (since it is a long-time process) before quantum computers are available. In the case of symmetric cryptography, however, the situation seems a bit less critical (although it has also been less studied). For many years, people believed that the only advantage that an attacker could have by using a quantum computer to attack symmetric cryptography would be due to Grover's algorithm (Section 2.7), just to speed up brute force search. Since Grover's algorithm reduces the effective key-length of any cryptographic scheme by a factor of two, to counter the attack it would be sufficient to double the key-length of the block cipher, to have again the same security against quantum attackers than before. There are two main techniques to do so: Using whitening keys, or multiple encryptions.

However, in some recent works [32] [33], followed by [29], the idea that not only Grover's algorithm might be a threat for symmetric cryptography was presented. In the first two papers, the authors showed that the Even-Mansour construction [28] can be broken in polynomial time in the quantum *CPA-setting* [26] (Chosen Plaintext Attack), a setting in which the attacker is allowed to make quantum queries (to the encryption function in quantum superposition).

A scheme of the aforementioned *Even-Mansour construction* can be seen in the Figure 3.1.

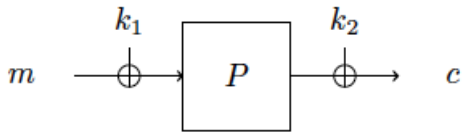

Figure 3.1: Even-Mansour construction.

It consists of a public permutation P on n bits, along with two secret keys k_1 and k_2 that are used as pre-whitening and post-whitening keys for the encryption of a certain message m, respectively. We will denote the encryption of a message m by means of this procedure as $\text{Enc}_{EM}(m)$. With this construction, they showed that if P is randomly chosen amongst all possible permutations, the advantage that an attacker might have to distinguish between the encryption and a random permutation is bounded by $q^2/2^n$, with q the number of queries to P of to the encryption oracle.

The main idea of a CPA is that the adversary is allowed to ask for encryptions of multiple messages chosen in an adaptative form, and this can be formalized just by allowing the adversary to interact freely with an encryption oracle, a black-box that encrypts its messages. When the adversary queries the oracle, by providing a plaintext message as an input, the oracle returns a ciphertext as a reply.

The definition of security under CPA requires that the adversary should not distinguish the encryption of two arbitrary messages (even if we give access to an encryption oracle to the adversary). Hence, in the quantum CPA-setting, the previous scheme is completely insecure. This can be seen following this procedure:

1. Consider the function

$$f(x) := \text{Enc}_{EM}(x) \oplus P(x) = P(x \oplus k_1) \oplus k_2 \oplus P(x),$$

where $\oplus$ is the XOR, or addition modulo 2.

2. Since this function satisfies

$$f(x) = f(x \oplus k_1) \text{ for all } x,$$

Simon's algorithm can be used to compute the unknown period k_1 of the function f in linear time.

3. Once k_1 is computed, computing k_2 is trivial even on a classical computer.

The same idea was also used in [29] to construct polynomial time quantum CPA-attacks on many modes of operations. Since both Grover's and Simon's algorithms have been mentioned in the previous scheme, the natural question then is whether the *FX construction* [30] [31] is secure against quantum adversaries.

That construction is an elegant way of, given a block cipher, extending its key-length. It is the natural combination of the Even-Mansour construction and a generic cipher. A scheme of the construction can be seen in Figure 3.2.

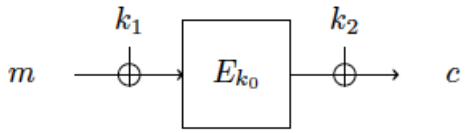

Figure 3.2: FX construction.

For this construction, we are given a secure block cipher E, encrypting n bit messages under an m bit key k_0, and we introduce two more n bit keys k_1 and k_2 as pre-whitening and post-whitening keys, respectively. The new block cipher is then given by

$$\text{Enc}(x) = E_{k_0}(x \oplus k_1) \oplus k_2.$$

3.1.1 A quantum attack based on the FX construction based on Grover's and Simon's algorithms

In an idealized model, one can see that, to attack the FX construction scheme, the probability of success of an attacker is bounded by $\frac{q^2}{2^{n+m}}$, with q the number of queries to the encryption scheme and to the underlying block cipher. If one considers only Grover's algorithm, this scheme seems pretty resistant to quantum computers, and Simon's algorithm cannot be used in this case either, since the encryption function is not periodic in general.

However, in [19], the authors show that the FX construction, indeed, can be broken in the quantum CPA-setting in basically the same time without the whitening keys, i.e., in $O(m+n)2^{\frac{m}{2}}$ quantum steps, and, thus, whitening keys are useless against quantum CPA-attackers.

For that, they combine Simon's and Grover's algorithms, but, in contrast with the previous works mentioned above, they need to define a new quantum algorithm based

on those techniques, and not just apply the known ones to new problems. They remark in their work that the main difficulty that appears when merging Simon's and Grover's algorithms lies in the technical obstacle that appears when considering both algorithms in their original form, since Simon's extracts information on the period bit by bit, whereas Grover's requires all the information to be available at once. They hence outcome this situation by running several instances of the quantum circuit that appears in Simon's algorithm in parallel, which can be seen to scale linearly with the size of the quantum computer. Another key point is that they measure in Simon's algorithm at the very end of the quantum algorithm that they design, using the deferred measurement principle of quantum computation, something that we will further explain below.

We will outline now the main ideas of this work. Since we are going to attack the FX-construction, let us recall that the block cipher is given by

$$\mathrm{Enc}(x) = E_{k_0}(x \oplus k_1) \oplus k_2.$$

Consider the function

$$f(k, x) := \mathrm{Enc}(x) \oplus E_k(x) = E_{k_0}(x \oplus k_1) \oplus k_2 \oplus E_k(x).$$

For a certain key $k = k_0$, the periodicity property in the second argument is satisfied, i.e., $f(k, x) = f(k, x \oplus k_1)$ for all x. However, in general, for $k \neq k_0$, the function $f(k, \cdot)$ is not periodic with high probability.

Taking this into account, we present the main idea of the paper:

Idea 1. *Define a Grover search over $k \in \mathbb{F}_2^m$, where we can test periodicity, for every $f(k, \cdot)$, via Simon's algorithm.*

Hence, Grover is considered as an outer loop, with running time $2^{\frac{m}{2}}$, and Simon is an inner loop, with polynomial complexity.

Remark 5. We are using the notation $\mathbb{F}_2^m$ for the set $\{0,1\}^m$ to be consistent with the notation of paper [19], but notice that both notations represent the same set.

Recall now the process of amplitude amplification explained in Subsection 2.7.3, which we mentioned that could be seen as a generalization of the original Grover search. In the preivous idea, it plays a role in the following sense: Classically, we can define an outer loop that guesses $k = k_0$ correctly with probability 2^{-m}, and this would require expectedly 2^m iterations until we hit the correct key, each of which would increase the probability of success by an amount of $1/p$. However, quantumly, each iteration roughly increases the amplitude of success by a constant, and we would only have to repeat the loop approximately $2^{\frac{m}{2}}$.

A more accurate result of amplitude amplification, extracted from [22], is the following.

Theorem 8. *Let $\mathcal{A}$ be a quantum algorithm on q qubits in which no measurement is used. Consider the function $\mathcal{B} : \mathbb{F}_2^q \to \{0,1\}$ that classifies the outcomes of $\mathcal{A}$ as good or bad. Let $p > 0$ be the initial success probability that a measurement of $\mathcal{A}|0\rangle$ is good. Fix $k = \lceil \frac{\pi}{4\theta} \rceil$, for θ defined via $\sin^2(\theta) = p$.*

Define also the unitary operator $Q = -\mathcal{A}S_0\mathcal{A}^{-1}S_{\mathcal{B}}$, for an operator $S_{\mathcal{B}}$ that changes the sign of the good state:

$$|x\rangle \mapsto \begin{cases} -|x\rangle & \text{if } \mathcal{B}(x) = 1 \\ |x\rangle & \text{if } \mathcal{B}(x) = 0 \end{cases},$$

and an operator S_0 that changes the sign of the amplitude only for the zero state $|0\rangle$.

Then, after the computation of $Q^k\mathcal{A}|0\rangle$, a measurement yields good with probability at least $max\{1-p, p\}$.

Following the idea that appears in 1, we would like to choose $\mathcal{A}$ as Simon's algorithm and apply the previous theorem to this setting. Although, explicitly, the previous theorem excludes the use of measurement (and Simon's algorithm uses measurements to extract information about the period), this technical problem can be easily resolved by the quantum principle of deferred measurement, which postpones all measurements until the very end of the computation (something that we mentioned before that was one of the key tools in this procedure).

However, there are still three problems to overcome, that can be addressed in the following way:

1. **Classifier.** *We need to define a classifier $\mathcal{B}$ that identifies states as good if, and only if, they correspond to the correct key $k = k_0$. Without the knowledge of k_1, it is difficult to do that efficiently, and this is a problem, since Simon's algorithm computes some information on k_1 iteratively, but we need the complete k_1 to be able to classify states as good or bad.*

   Simon's algorithm computes k_1 after $O(n)$ iterations. Each one of these iterations gives a random vector that belongs to a certain subspace, so the main idea of this step is to parallelize the computation of these vectors (and increase the number of input bits from $O(n)$ to $O(n^2)$). By means of this, the classifier identifies the states $|x\rangle$ with $k = k_0$ adequately.

2. **Simon's promise.** *Simon's algorithm is originally defined for 2-to-1 functions, but $f(k_0, \cdot)$ does not fulfill that condition, since some values have more than two preimages.*

   To overcome this problem, the authors show that any function $f(k_0, x)$ has only two preimages with probability at least $1/2$. Then, they only have to argue about the proper function values $f(k_0, x)$.

3. **Success probability.** *If we are able to define a suitable classifier $\mathcal{B}$, it is possible that we are only capable of lower bounding the initial success probability p instead of exactly determining it, which might cause problems in properly setting the number of iterations k.*

   For this, we can choose the number k of iterations assuming that we would classify all states with $k = k_0$ as good. This implies that the choice of k might be too small to fully rotate towards the subspace of good states, but using Theorem 8 it is possible to show success with probability at least $2/5$.

Once these three problems have been resolved, and some technical results are proven (to see them, consult [19]), the following result can be proven:

Theorem 9. *Consider the function $f : \mathbb{F}_2^m \times \mathbb{F}_2^{3n} \to \mathbb{F}_2^n$ verifying*

$$(k_0, k_1, k_2, x) \mapsto g(k_0, x \oplus k_1) \oplus k_2,$$

where $g : \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $g(k, \cdot)$ is a random function $\mathbb{F}_2^n \to \mathbb{F}_2^n$ for any fixed

$k \in \mathbb{F}_2^m$. Then, if we denote $f_{k_0,k_1,k_2}(\cdot) := f(k_0, k_1, k_2, \cdot)$ and give quantum oracle access to $f_{k_0,k_1,k_2}(\cdot)$ and $g(\cdot,\cdot)$, the tuple (k_0, k_1, k_2) can be computed with success probability at least $2/5$ using $m + 4n(n + \sqrt{n})$ qubits and

$$2^{\frac{m}{2}} \cdot O(m + n) \text{ oracle queries.}$$

This result show that the FX construction can be broken in the quantum CPA-setting in $O(m + n)2^{\frac{m}{2}}$ quantum steps (the same time without the whitening keys), and, thus, whitening keys are useless against quantum CPA-attackers.

3.1.2 Hidden Shift Quantum Cryptanalysis

In this subsection we are going to summarize the results of another paper on quantum attacks [20]. The attacks mentioned in the previous subsection apply in scenarios of superposition quantum queries, i.e., when the adversary is allowed to perform local computations on a quantum computer and to perform superposition queries to a remote quantum cryptographic oracle to obtain the superposition of outputs. This is a strong model for the attacker.

However, there are some arguments in the community to defend the interest of the study of the security of symmetric primitives in this setting:

1. The model is simple. Using another model would imply measures that would be artificial and hard to respect with respect to cryptographic oracles, with complex manipulations and uncertain outcomes.

2. Safety in this model implies safety in any other scenario, even advanced ones.

3. The model is not trivial: Not every primitive is broken in it. Several resistant constructions to this model have been proposed.

In a very recent work [25], the key idea that the authors present is to replace the common addition over $(\mathbb{Z}/(2))^n$, mentioned in the previous sections (in this chapter and the previous one), since it is vulnerable to Simon's algorithm, and use instead another operatios that implies a harder problem to solve. For that, they consider addition over $\mathbb{Z}/(2^n)$, what they call modular addition, and claim that the quantum hardness of the hidden shift problem proves that their proposal is secure against quantum chosen-plaintext attacks.

With this new idea, the attacks are no longer $O(n)$, and Simon's algorithm does not hold anymore, but attacks that are faster than the generic ones can be described using Kuperberg's algorithm [34]. For example, one can get a speedup from $O(\sqrt{2^n})$ to $2^{O(\sqrt{n})}$.

Classically, we say that a symmetric primitive is secure when no attack is better than the generic attack that exists. While the complexity of the generic exhaustive search is $2^{n/2}$, the complexity of quantum attacks on primitives with modular additions is subexponential. This induces a necessity for a new definition of security, and new tools to build secure primitives with the countermeasures mentioned above, since the best generic attacks, such as those based on Kuperberg's, are better than exhaustive search.

The main contribution of [20] can be summarized in the following points:

1. **Improvement and generalization of Kuperberg's algorithm.** They study Kuperberg's quantum algorithm for hidden shifts in $\mathbb{Z}/(N)$ and its possible applications in symmetric cryptography. Later, they focus on $\mathbb{Z}/(2^n)$, which have been commonly used in symmetric cryptography, and propose a variant for the original

algorithm that performs better by getting all the bits in one step (since the original algorithm gets one bit at a time and uses a reducibility property to get the next one), which provides a great cost reduction of the attack on Poly1305 [27].

Moreover, they also propose a way to solve the hidden shift problem in non-abelian groups.

2. **Simon meets Kuperberg.** They propose a new quantum algorithm that considers a generalization for products of groups of the form $\mathbb{Z}/(2^p)^\omega$ and their subgroups (cyclic groups), which are commonly used in symmetric primitives. Then, they notice that the problem is easier to solve in this case than in $\mathbb{Z}/(2^{\omega p})$. Their analysis shows that it reduces to Simon when $\omega = 1$ and Kuperberg when $p = 1$.

3. **Simulation of the algorithms.** They implement the classical parts of these algorithms (the three mentioned above: Kuperberg, the improved Kuperberg and the Simon-meets-Kuperberg one) and simulate them to estimate the asymptotic query complexity and to get values for parameters of interest verifying the expected complexity.

4. **Attack on Poly1305 in the superposition model.** The authors propose a quantum attack on Poly1305. A classical and quantum security of 128 bits has been previously claimed for Poly1305. Their attack works in the superposition model, has a complexity of 2^{38} and uses their improved version of Kuperberg's algorithm.

5. **Attack on the FX variants.** In this step, they answer a question raised in the paper discussed in the previous subsection, concerning the quantum security of the FX construction with any group law. If the inner key addition is done with a commutative group law, the improvement in the security of the construction is marginal, and, thus, the best one can hope to get with non-abelian groups is an improvement of around $n/3$ bits of security for an n-bit inner key.

6. **Evaluate the proposed countermeasures from [25].** Finally, they try to determine the right size of the symmetric primitives to offer a certain desired security, and to decide if the proposed countermeasure is sufficient.

As a conclusion, they mention that the new use of modular additions in vulnerable constructions instead of xor-addition for key addition increases the complexity of the corresponding quantum key-recovery attack. However, they show that the proposal from [25] does not seem practical, and would require an internal state size of a few thousand bits, to be compared with the size of the internal state of AES-256, which is 128 bits.

3.2 A brief overview of post-quantum cryptography

In this section, the last one of the manuscript, we are going to present an introduction to post-quantum cryptography. This section is mainly based in the works [23], [24] and [39].

An important part of the current possibilities in communication nowadays is protected by cryptographic systems, such as the Rivest-Shamir-Adleman (RSA) system, or elliptic-curve cryptography (ECC), among others. However, as we have seen in the previous chapters, if quantum computing scales as expected, it will break both of them. Hence, the necessity of the development of post-quantum cryptographic systems is huge.

In the popular RSA public-key system, the public key is a product $N = pq$ of two secret prime numbers p and q, and the security of this system relies on the difficulty to find the

factors p and q. However, Shor's algorithm, introduced in Section 2.6, finds quickly the prime factorization of any positive integer N.

Optimizing the exact cost of this algorithm is a popular problem in quantum cryptography, but deep research on this topic has been conducted in the last years. This algorithm, when applied to widely deployed public-key sizes for RSA and ECC, needs billions of operations on thousands of logical qubits to work. Hence, fault-tolerant attacks seem likely to require trillions of operations on millions of physical qubits.

Thus, there is still a chance that quantum computing will encounter a fundamental obstacle that prevents it from ever scaling successfully to the enormous sizes mentioned above. However, so far, such obstacles have not been identified yet, so we cannot rely on their possible existence. Since an attack on that system would imply general loses and chaos in society, prudent risk management is necessary, and it requires defending against the possibility that these attacks will be successful.

Some other of these cryptographic systems may be affected by Grover's algorithm, introduced in Section 2.7. This algorithm constitutes the foundation for most (although not all) of the positive applications that have been identified for quantum computing. The 'Advanced Encryption Standard' (AES) is an example of a symmetric encryption algorithm. To see how Grover affects to this system, assume that a user has encrypted 128-bit plaintexts '1' and '2' under a secret 128-bit AES key k, producing the ciphertext that we denote by

$$c = (AES_k(1), AES_k(2)),$$

which is visible by the attacker. Thus, if we define

$$f(x) := (AES_x(1), AES_x(2)) - c,$$

this function can be evaluated quickly by a small circuit, and it will take only 2^{64} quantum evaluations for Grover's algorithm to find a root of f. Hence, if qubit operations are small and fast enough, Grover's algorithm will threaten many cryptographic systemsn such as this one, the 128-bit AES keys, and, in general, systems that aim for 2^{128} security. One can simply switch to 256-bit AES keys: the extra costs are difficult to notice.

However, not every cryptographic system is necessarily broken by the use of a quantum computer. Some 'information-theoretic' MACs such as GMAC and Poly1305 already protect against quantum computers without any modifications: their security analysis already assumes an attacker with unlimited computing power.

In the following table (Figure 3.3), which has been extracted from [23], we can see some examples of some well-known cryptographic systems and their conjectured security levels after the use on them of a quantum computer:

In this table we can see that, as mentioned above, the statement "quantum computers destroy cryptography" is, in general, false. Indeed, there are many important classes of cryptographic systems beyond RSA (and DSA and ECDSA, which are also broken by quantum computers), as we can enlist below, and which constitute the basis for some possible schemes in post-quantum cryptography:

- **Hash-based cryptography.** The most classic example of this class is Merkle's hash-tree public-key signature system (1979) [40], and it is built upon a one-message-signature idea of Lamport and Diffie.

- **Code-based cryptography.** The most classic example of this class is McEliece's hidden-Goppa-code public-key encryptation system (1978) [41].

Name	Function	Pre-quantum security level	Post-quantum security level
Symmetric cryptography			
AES-128[8]	Symmetric encryption	128	64 (Grover)
AES-256[8]	Symmetric encryption	256	128 (Grover)
Salsa20[58]	Symmetric encryption	256	128 (Grover)
GMAC[59]	MAC	128	128 (no impact)
Poly1305[60]	MAC	128	128 (no impact)
SHA-256[61]	Hash function	256	128 (Grover)
SHA3-256[62]	Hash function	256	128 (Grover)
Public-key cryptography			
RSA-3072[1]	Encryption	128	Broken (Shor)
RSA-3072[1]	Signature	128	Broken (Shor)
DH-3072[42]	Key exchange	128	Broken (Shor)
DSA-3072[63,64]	Signature	128	Broken (Shor)
256-bit ECDH[4–6]	Key exchange	128	Broken (Shor)
256-bit ECDSA[66,67]	Signature	128	Broken (Shor)

Figure 3.3: Examples of some cryptographic systems and their conjectured security levels.

- **Lattice-based cryptography.** The most interesting example (not the first one) of this class is Hoffstein-Pipher-Silverman "NTRU" public-key encryptation system (1998) [42].

- **Multivariate-quadratic-equations cryptography.** One of the most interesting examples of this class is Patarin's "HFE$^v-$" public-key encryptation system (1996) [43], which generalizes a proposal by Matsumoto and Imai.

- **Secret-key cryptography.** The leading example is the Daemen-Rijmen "Rijndael" cipher (1998) [44], subsequently renamed "AES", the Advanced Encryption Standard.

We think that these systems resist both classical and quantum computers. Indeed, nobody has figured out a way to apply Shor's algorithm to any of these systems. However, Grover's algorithm has some applications to these systems, but Grover's algorithm is not as shockingly fast as Shor's algorithm, and cryptographers can easily compensate its impact by choosing larger key sizes, as we have already mentioned in some parts of this text.

Of course, it does not mean that there is no possible attack on these systems. This is a familiar risk in cryptography. That is the reason why the community invests huge amounts of time and energy in cryptanalysis. It is indeed possible that cryptanalysts find a devastating attack, showing that a system is useless for cryptography, and it happens sometimes. For example, any usable choice of parameters for the Merkle–Hellman knapsack public-key encryption system is easily breakable. There are situations in which cryptanalysts find attacks that are not so devastating but that force larger key sizes (as with Grover's algorithm), or, on the contrary, in some cases cryptanalysts study systems for years without finding any improved attacks, and the cryptographic community starts to think that the best possible attack has been found, and no attackers in the real-world will be able to come up with anything better.

In the following subsections, we will study some details of some proposals that, as mentioned above, have solidly resisted every suggested attack.

3.2.1 Code-based encryption

In general, high-reliability computer equipment uses an 'error-correcting code' to store 64 bits of logical data in 72 bits of physical memory. For that, there is a 64×72 generator matrix G, with each entry in $\mathbb{F}_2$: in other words, a 64×72 matrix of bits. Hence, this matrix specifies each of the 72 physical bits as a sum of some of the 64 logical bits modulo 2. The code is denoted by $\mathbb{F}_2^{64}G$, a 64-dimensional subspace of the vector space $\mathbb{F}_2^{72}$, namely the subspace generated by the rows of G.

Since it is an error-correcting code, it is designed so that any single error in the 72 bits (any change of a bit to its opposite) can be reliably corrected, and any double error (changing any two bits) can be reliably detected (although not necessarily corrected). Error-correcting codes can be scaled up to correct more errors in longer blocks of data. They are used in a wide range of applications, including hard drives, satellite communication, and fault-tolerant quantum computation.

In 1978, early in the history of public-key cryptography, as we have mentioned previously, McEliece proposed to use a generator matrix as a public key, and encrypting a codeword (an element of the code) by adding a specified number of errors to it. This gave birth to code-based encryption systems.

A simple but slow attack strategy against McEliece's system, as we will see below, is *information-set decoding* (ISD). An information-set is a collection of codeword positions used to determine the rest of the codeword. Indeed, ISD guesses an information set, hoping that the ciphertext is error-free in those positions; then, it uses linear algebra to determine the entire codeword, and, finally, it also checks that the ciphertext has the specified number of errors, understanding that in this case the codeword must be correct.

In this class of systems, we assume that b is a power of 2, i.e., $b = 2^m$. We write $n = 4b \log b$ and consider $d = \lceil \log n \rceil$ and $t = \lfloor 0,5n/d \rfloor$. For example, if we assume that $b = 128$, then we have $n = 3584$, $d = 12$ and $t = 149$.

In this system, the receiver's public key is a $(d \cdot t) \times n$ matrix K with coefficients in $\mathbb{F}_2$, i.e., 0 or 1. In this setting, the messages that are suitable for encryption are n-bit strings of "weight t", i.e., n-bit strings which have exactly t bits set to 1. To encrypt a message m, the sender only has to multiply K by m, producing a $(d \cdot t)$-bit ciphertext $K \cdot m$.

The basic problem for an attacker to these systems is to "syndrome-decode K", i.e., to undo the multiplication by K, knowing that the input had weight t. Using techniques from linear algebra, it is easy to work backwards from $K \cdot m$ to some n-bit vector v such that $K \cdot v = K \cdot m$. However, the problem lies in the fact that there are a huge number of choices for v, and finding a weight-t choice seems to be extremely difficult. Indeed, the best known attacks on this problem take time exponential in b for most matrices K.

For the receivers to solve the same problem, they have to generate the public key K with a secret structure, something that is called *hidden Goppa code* structure, which will allow the receiver to decode in a reasonable amount of time. It is possible that the attacker might be able to detect the hidden Goppa code structure in the public key, although no such attack is known so far.

More specifically, the receivers begin with some different elements $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}_{2^d}$ and a secret monic, irreducible polynomial of degree t, $g \in \mathbb{F}_{2^d}[x]$. Then, the main work for the receiver is to syndrome-decode the following matrix of size $(d \cdot t) \times n$:

$$H = \begin{pmatrix} 1/g(\alpha_1) & \dots & 1/g(\alpha_n) \\ \alpha_1/g(\alpha_1) & \dots & \alpha_n/g(\alpha_n) \\ \vdots & \ddots & \vdots \\ \alpha_1^{t-1}/g(\alpha_1) & \dots & \alpha_n^{t-1}/g(\alpha_n) \end{pmatrix},$$

where we can view each element of $\mathbb{F}_{2d}$ as a column of d elements of $\mathbb{F}_2$ in a standard basis of $\mathbb{F}_{2^d}$. This matrix H is a parity-check matrix for an irreducible binary Goppa code, and can be syndrome-decoded by *Patterson's algorithm* or by faster algorithms.

The public key K of the receiver can be seen as a scrambled version of the matrix H. More specifically, the receiver's secret key also includes an invertible $(d \cdot t) \times (d \cdot t)$ matrix S and an $n \times n$ permutation matrix P.

- The public key K is the following product: $S \cdot H \cdot P$.
- Given a ciphertext $K \cdot m = S \cdot H \cdot P \cdot m$, the receiver multiplies by S^{-1} to obtain $H \cdot P \cdot m$.
- Afterwards, they decode H to obtain $P \cdot m$.
- Finally, they multiply by P^{-1} to obtain m.

This constitutes a variant, due to Niederreiter (1986), of McEliece's original code-based public-key encryption system. Both systems are really efficient at key generation, encryption, and decryption, but have been held back by their long public keys.

The main practical problem with these systems is the key size, which is approximately a megabyte (in systematic form) at a high security level. Many newer code-based systems put more structure into public keys to allow more compression, but some of those proposals have already been broken. The only post-quantum public-key-encryption systems that have received enough study for us to recommend are the original McEliece/Niederreiter systems.

3.2.2 Lattice-based encryption

The security of lattice-based cryptosystems depends strongly on the conjecture that lattice problems, a class of optimization problems on lattices, are intractable. In particular, we shall mention in this section the problem *SVP (Shortest vector problem)*.

In SVP, we consider a lattice L in a vector space V, a basis of such space, and a norm $\|\cdot\|$. The problem consists of the following:

Find the shortest non-zero vector in V, measured by $\|\cdot\|$, in the lattice L.

Moreover, in other words, if we define

$$\lambda(L) := \min_{v \in L \setminus \{0\}} \|v\|,$$

the algorithm should output a non-zero vector v such that $\|v\| = \lambda(L)$. There is also a γ-approximation version of SVP, which is denoted by SVP_γ, and outputs a non-zero lattice vector of length, at most, $\gamma \cdot \lambda(L)$, for a given $\gamma \geq 1$.

To solve the exact version of SVP with the euclidean norm, there are several possible different approaches, which can be split into two main classes:

- *Algorithms that require subexponential time $2^{\omega(n)}$ and poly(n) memory.*

  The main examples of this class are lattice enumeration and random sampling reduction.

- *Algorithms that require both exponential time and space $2^{\theta(n)}$.*

  The main examples of this class include lattice sieving, computing the Voronoi cell of the lattice and discrete Gaussian sampling.

An open problem in this field is whether there is an algorithm for solving the exact SVP that runs in exponential time $2^{O(n)}$ and requires that memory scales polynomially in the lattice dimension.

For the γ-approximation version of SVP, with $\gamma > 1$, also with the euclidean norm, the situation is a bit different, and the best known approaches are based on lattice based reduction. For large γ, of order $2^{\Omega(n)}$, a solution can be found in polynomial time in the lattice dimension by using the Lenstra-Lenstra-Lovász (LLL) algorithm. And for smaller γ, the Block Korkine-Zolotarev algorithm (BKZ) is commonly used.

Let us introduce now an specific lattice-based encryption system. Around 1990, Hoffstein, Pipher and Silverman introduced an encryption system called *NTRU*, which has much smaller keys than McEliece's system (mentioned in the previous subsection) and that remains unbroken today. This encryption system was considered lattice-based a posteriori.

This system works as follows: The public key is given by a p-coefficient polynomial $h = h_{p-1}x^{p-1} + \ldots + h_1 x + h_0 = h$, where each coeficient is taken from $\{0, 1, \ldots, q-1\}$. Moreover, we define a *ciphertext* as another polynomial b in the same range. To compute it, the sender chooses two secret polynomials, called c and d, with small coeficients, and computes first

$$x = hc + d (\text{mod } x^p - 1),$$

and then b is given by

$$b = x (\text{mod } q).$$

Now, if we define by L the set of pairs (u, v) of p-coefficient polynomials with integer coefficients verifying $0 = (hu - v (\text{mod } x^p - 1)) (\text{mod } q)$ (considering both modulos consecutively), we get that L is a lattice in a $2p$-dimensional space, and it contains the point $(c, b - d)$, which is close to $(0, b)$. Hence, one can translate the problem of finding the secret c and d, given the ciphertext b and public key h, to the problem of finding a lattice point close to a certain given point. It is indeed analogous to the decoding problem for codes, just by taking into account the change of notation (in our case, we understand by 'close' the fact of every coefficient being small, but in the case of codes they just count the number of non-null coefficients).

The system mentioned above, NTRU, works following these schemes. It secretly generates the public key in a way that makes decoding efficient, since the receiver begins with a short vector of the form $(n, 3m)$, and uses Euclid's algorithm to find the element h such as the lattice contains this vector. In other words, such that

$$a = hn - 3m (\text{mod } x^p - 1)$$

and

$$0 = a (\text{mod } q).$$

Then, a simple analysis shows that

$$(bn(\text{mod } x^p - 1))(\text{mod } q) = (3cm + dn(\text{mod } x^p - 1))(\text{mod } q),$$

and, thus,

$$3cm + dn(\text{mod } x^p - 1)$$

almost certainly has all coefficients strictly between $-q/2$ and $q/2$, which makes it easy to find the element (c, d) given both m and n.

However, in principle, there are many possible ways to attack NTRU and other lattice-based cryptosystems. One example of this is the recent use of the 'cyclotomic' structure of $x^p - 1$ to break some lattice-based cryptosystems, using an extension of Shor' algorithm. As far as we know, NTRU has not been directly affected, but this new class of attacks has not been adequately explored yet.

Another example that this procedure might not be the right one is the following: Recently, there have been some attacks in arbitrary lattices that work without exploiting any polynomial structure, and they have smaller exponents than the best such attacks known just a few years ago; these lattices are used in cryptosystems which are based on the so-called 'learning with errors'. There is still much more research to do in order to gain confidence in the security of lattice-based cryptography.

3.2.3 Lattice-based signatures

Turning hard lattice problems into signature systems is another possibility, although the first attempts to do that were already attacked. Surviving systems, though, suffered from large signature sizes. After this, from 2012, the most promising signature systems are based on Lyubashevsky's signature system.

This system can be presented using integer matrices, although its implementations typically use polynomial rings and fast Fourier transforms for compact representations and efficiency. It also uses several system parameters, which we denote by k, m, n, to determine the sizes of matrices, the parameter κ to limit the Hamming weight of certain vectors, and q a modulus.

Consider now A an $n \times m$ integer matrix modulo q (i.e., $A \in \mathbb{Z}_q^{n \times m}$). The private key is given by a matrix $S \in \mathbb{Z}^{m \times k}$ with small entries (much smaller than q), often restricted to $\{-1, 0, 1\}$. Hence, the public key is the $n \times k$ matrix given by

$$T = A \cdot S,$$

whose entries are computed modulo q. Moreover, if A is not shared from the beginning, it is also part of the public key.

These systems also use a hash function

$$H : \mathbb{Z}_q^n \times \{-1, 0, 1\}^* \to \{0, 1\}^k,$$

with the further assumption that the output vectors have no more than κ non-negative entries. This H can be built from a traditional hash function h by encoding inputs and outputs appropriately.

This system works in the following way: First, the signer picks y from an m-dimensional distribution (usually a discrete Gaussian distribution), and computes

$$x = A \cdot y (\text{mod } q)$$

and

$$c = H(x, \mu),$$

for μ the message.

Consider now $z = S \cdot c + y$; then, the signature is given by the pair (c, z). Furthermore, to avoid the possibility of leaking information about the private key S through the distribution of (c, z), this method forces an S-independent distribution, by means of some 'rejection sampling'.

This basically means that the process is restarted with probability depending on (c, z). Hence, the signature is accepted if c and z are sufficiently small, and given

$$x_2 = A \cdot z - T \cdot c (\text{mod } q),$$

the following holds:

$$H(x_2, \mu) = c.$$

Currently, the most important problems to solve in this setting are the following:

- Generate the distribution in a way to prevent the leaking of information on S.
- Analyze the security of the underlying algorithmic problem (finding short integer solutions to a system of equations modulo q).

3.2.4 Multivariate-quadratic-equation signatures

This class of signatures was first developped from a signature C^* that was introduced in 1988 by Matsumoto and Imai. Afterwards, in 1995, Patarin broke this system and one year later introduced a stronger system, HFE^{v-} which is still unbroken today.

The public key of such system is a sequence of polynomials $p_1, \ldots, p_m$ in the n-variable polynomial ring $\mathbb{F}_2[x_1, \ldots, x_n]$ over the field $\mathbb{F}_2$, with $m \leq n$. These polynomials are limited to quadratics and have no squared terms, i.e., they are of the form:

$$p_i = a_i + \sum_j b_{i,j} x_j + \sum_{j<k} c_{i,j,k} x_j x_k,$$

with the coefficients $a_i, b_{i,j}, c_{i,j,k} \in \mathbb{F}_2$, and which lack obvious public structure.

In this case, the signature of a message is a string of n-bits, $(s_1, \ldots, s_n) \in \mathbb{F}_2^n$ such that the string $(p_1(s_1, \ldots, s_n), \ldots, p_m(s_1, \ldots, s_n)) \in \mathbb{F}_2^m$ is equal to the standard hash of m-bits of the message. These short signatures are an attractive feature of this signature system.

The procedure to follow in these systems is the following:

1. First, the signer chooses the polynomials with a secret structure that allows him/herself to solve the simultaneous quadratic equations

$$h_1 = p_1(s_1, \ldots, s_n), \ldots h_m = p_m(s_1, \ldots, s_n).$$

2. The signer views an m-bit hash value, along with a randomly chosen $(n - v - m)$-bit string, as an element $H \in \mathbb{F}_q$. For that, starting from the hash value, they have to choose $n - m$ random bits from the ones mentioned above and construct H. For that, the signer considers a polynomial $P \in \mathbb{F}_q[x, y_1, \ldots, y_v]$ of the form

$$A + \sum_j B_j x^{2^j} + \sum_{j>k} C_{j,k} x^{2^j+2^k} + \sum_j D_j y_j + \sum_{j,k} E_{j,k} y_j x^{2^k} + \sum_{j>k} F_{j,k} y_j y_k,$$

which gives an equation connecting S and H:

$$P(S, r_1, \ldots, r_v) = H.$$

To write the equation in a quadratic polynomial form, the signer writes each bit of S^{2^j} as a linear combination of $s_1, \ldots, s_n$. To solve it, the signer just notices that it is a univariate equation in S for any particular choice of random bits $r_1, \ldots, r_v$.

There is an important number of works on other multivariate-quadratic signature systems and on algorithms to attack these systems. In particular, all known attacks take time exponential in approximately $(n - m + \lceil \log_2 d \rceil) \log_2 n$.

3.2.5 Hash-based signatures

One of the design goals for hash functions is the fact that finding a pre-image for a given output string is computationally hard. In 1975, Lamport realized that this could be used to build a one-time signature system.

This signature system requires a standard cryptographic hash function H that produces $2b$ bits of output. For example, for $b = 128$, one could choose H as the $SHA-256$ hash function.

In the last few years, some people have raised some concerns regarding the security of popular hash functions, and this has resulted, among other things, on the fact that, over the next few years, NIST will run a competition for a $SHA-256$ replacement. However, all known attacks against $SHA-256$ are really expensive.

In these systems, the signer's public key has $8b^2$ bits. For example, for $b = 128$, it has 16 kilobytes. The key consists of $4b$ strings, enumerated as $y_1[0], y_1[1], y_2[0], y_2[1], \ldots, y_{2b}[0], y_{2b}[1]$, each of which has $2b$ bits.

Moreover, a *signature* of a message m has $2b(2b+1)$ bits (for example, in the previous case of $b = 128$, it has 8 kilobytes). The signature consists of $2b$-bit strings $x_1, \ldots, x_{2b}$ such that the bits $(h_1, \ldots, h_{2b})$ of H satisfy $y_1[h_1] = H(x_1), y_2[h_2] = H(x_2), \ldots, y_{2b}[h_{2b}] = H(x_{2b})$.

In this setting, the procedure for the signer to follow to find $y = H(x)$ starts by generating a secret x and computing afterwards $y = H(x)$ for this secret x. More specifically, the signer's secret key has $8b^2$ bits, which are $4b$ uniformly random strings that are independent,

$$\{x_1[0], x_1[1], x_2[0], x_2[1], \ldots, x_{2b}[0], x_{2b}[1]\},$$

each of which as $2b$ bits (giving the total of $8b^2$ bits). Then, the public key

$$\{y_1[0], y_1[1], y_2[0], y_2[1], \ldots, y_{2b}[0], y_{2b}[1]\}$$

is computed by the signer as

$$\{H(x_1[0]), H(x_1[1]), H(x_2[0]), H(x_2[1]), \ldots, H(x_{2b}[0]), H(x_{2b}[1])\}.$$

Therefore, to sign a message m, the signer has to perform these tasks:

- Generates a uniform random string r.

- Compute the bits $(h_1, \ldots, h_{2b})$ of $H(r, m)$.
- Reveal $(r, x_1[h_1], \ldots, x_{2b}[h_{2b}])$ as a signature of m.
- Finally, discard the remaining x values and refuse to sign any more messages.

The procedure described above is the Lamport-Diffie one-time signature system. However, if the signer wants to sign more than one message, they just have to chain, i.e., include in the signed message a new public key that will be used in the next one. Hence, the verifier checks the first signed message, and since it includes the new public key, they can also check the signature of the next message (and so on).

Hash functions appear in all signature systems, and standard hash functions are affected only by Grover's attack, not by Shor's attack. This fact makes Merkle's very simple signatures that are excellent candidates for post-quantum signatures. Indeed, they have a clear security track record, and computing hash functions is very fast (let us recall that Merkle's hash-tree signature system, scale logarithmically with the number of messages signed, see Figure 3.4, extracted from [23]). Hash-based cryptography can convert any hard-to-invert function into a secure public-key signature system.

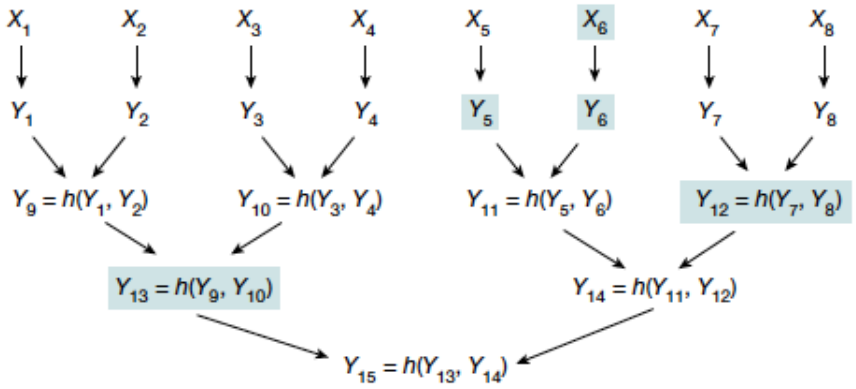

Figure 3.4: Merkle tree with public key Y_{15} to sign eight messages.

Finally, to finish this subsection, and all the previous ones, in Figure 3.5, extracted from [23], we can see a comparison between these different kinds of systems and signatures, with each ones' advantages and disadvantages.

3.2.6 Isogeny-based cryptography

This is the most recent class of primitives, and that is why it does not appear in Figure 3.5. The new isogeny-based primitives are interesting and have nice properties, such as small key sizes and forward security. They deserve more academic scrutiny to establish a consensus on their security properties, but are not supported by public challenge problems.

Their security is based on the difficulty of recovering an unknown isogeny between a pair of supersingular elliptic curves that are known to be isogenous. Hence, these agreements have security reductions from Diffie-Hellman style problems in isogeny graphs, reductions that are a special case of more general results on semi-group actions.

These schemes have some underlying isogeny problems which have not been proved to be difficult, but the complexity of recovering unknown isogenies between elliptic curves constitutes a field that has been relatively well studied. In particular, for supersingular

Approach	Advantages	Disadvantages
Code-based encryptation (using Goppa codes)	High confidence in security, very fast encryption and short ciphertexts	Large public keys
Lattice-based encryption (using NTRU or related)	Short ciphertexts and keys, very fast encryption	Require more security analysis
Lattice-based signatures	Short keys and signatures, and quite fast	Require more security analysis, side-channel attacks on discrete Gaussians
Multivariate-quadratic-equation signatures	Very short signatures	Require more security analysis
Hash-based signatures (stateful version)	High confidence, simple description	Management of state
Hash-based signatures (stateless version)	High confidence, simple description	Large signatures

Figure 3.5: Qualitative overview of the described post-quantum systems.

curves, unless the curves are both defined over a prime field, the best known algorithms are exponential.

The isogeny-based keys agreement consists of small public keys, and a reasonably efficient key generation process. However, it is not as fast as some of the other primitives. It could therefore be used in a forward secure protocol.

Chapter 4

Conclusions

4.1 General conclusions and attainment of the aims initially proposed

In this project, we have presented an introduction to post-quantum cryptography, starting from the most basic concepts of quantum information theory and quantum mechanics, to further introduce the formalism of quantum algorithms, and, later, we have introduced some quantum attacks to the current cryptographic systems, thus motivating the necessity of the development of this new field of post-quantum cryptography.

One of the main underlying aims of this work has been to show that, indeed, research on post-quantum cryptography is extremely necessary, mostly due to two reasons: The first one, the fact that the most popular public-key algorithms can be efficiently broken by a sufficiently strong hypothetical quantum computer; and the second one, the reality that the hypothetical quantum computer seems to be likely to appear in the next few years. These two facts have been motivated throughout the whole text, giving references of papers and textbooks to support it.

The main general objective of this work has been to provide a well-motivated and self-contained introduction to this field of post-quantum cryptography. For that, before presenting the field of post-quantum cryptography itself, we have introduced some quantum attacks to motivate its necessity, as well as the main basic concepts to understand the background in quantum algorithms. We think that we have succeeded in this task, since we obtained the desired results, as the manuscript is quite self-contained, in the sense that, besides from some basic mathematical concepts, most of the concepts needed to the text have been introduced throughout it, and we hope that we have convinced the reader of the importance of the matter.

We have also achieved to introduce a topic mostly associated to the computer science community in a mathematical formalism, and with a physical motivation and background. With this, we hope that the field of post-quantum cryptography, and the approach followed in this work, can be accesible to researchers from a wide range of different fields.

4.2 Follow-up of the planning and methodology

As a personal evaluation, I think that I have achieved the main objectives that were introduced at the beginning of the manuscript. First of all, I consider that I have improved my research capacity by looking into the bibliographic material needed to understand the topics developped on this project. I have also improved my way of expressing information

from a field that lives in the intersection of important several branches of sciences, as in this case, in the intersection of mathematics, physics and computer science.

This project has also allowed me to consolidate and increase my knowledge in quantum information theory, quantum mechanics, and cryptography, as well as some other fields of mathematics, physics and computer science. Furthermore, I have noticed the effect that comes out when different fields of mathematics are studied along with computer science to achieve their goals more efficiently.

Finally, in general, the temporary planning exposed at the beginning of the project has been followed, although the dates have suffered several modifications during the previous year, mostly due to unexpected problems or drawbacks. However, the structure of the planning and the methodology have turned out to be adecquate for the development of the project, and it has been possible to deliver the project fully finished and reviewed on time.

4.3 Future lines of work

The field introduced in this work has many lines of possible future work. Indeed, it is clear that deploying a cryptographic system has some physical costs associated, such as the time and energy consumed by cryptographic computations and by communication of keys, signatures, and so on. Currently, the deployment of cryptography for billions of users relies on the fact that cryptography fits the users' budget. With the problem associated to these costs, we go back in time, and some problems that were already solved with the use of public key schemes derived from RSA and discrete logarithms appear again.

In general, these are exciting times for post-quantum cryptography, since researchers have identified many different ways to provide critical functions such as public-key encryption and public-key signatures, as we have seen in the previous chapter. Some of these proposals have survived many years of scrutiny, but they have serious costs associated, especially in network traffic. Some other proposals are though more attractive for deployment, but their security is less clear, and it is likely that some of them will be broken soon.

Experts on the field expect an outstanding increase in research in post-quantum cryptography motivated by NIST's competition, taking the form of more designs, more optimizations and implementations, and also more attacks. An important part of progress will be to understand what systems are vulnerable to attacks. Much more work is needed to build post-quantum systems that are widely deployable while at the same time inspiring confidence. This constitutes the main open line that arises from this work.

Moreover, from the other side, another interesting line of work that is strongly related with this project is that of finding new quantum attacks to cryptographic schemes, both for the previous families of cryptographic schemes and for the new ones that have appeared recently. This is the most related line of future work associated to this project.

Bibliography

[1] C.H. Papadimitriou, Computational complexity, *Addison-Wesley* (1994).

[2] M.A. Nielsen and I.L. Chuang, Quantum Computation and Quantum Information, *Cambridge University Press*, New York (2000).

[3] M.M. Wilde, Quantum Information Theory, *Cambridge University Press* (2013).

[4] C. Palazuelos, Introduction to Quantum Information Theory, (2013).

[5] R. de Wolf, Quantum Computing: Lecture Notes, (2011).

[6] R. de Wolf, Quantum Computing and Communication Complexity, *PhD Thesis. University of Amsterdam* (2001).

[7] D. Deutsch and R. Jozsa, Rapid solution of problems by quantum computation, *Proceedings of the Royal Society of London* **A439** (1992), 553-558.

[8] D. Deutsch, Quantum theory, the Church-Turing principle, and the universal quantum Turing machine, *Proceedings of the Royal Society of London* **A400** (1985), 97-117.

[9] D. Deutsch, Quantum computational networks, *Proceedings of the Royal Society of London* **A425** (1989), 97-117.

[10] L.K. Grover, A fast quantum mechanical algorithm for database search, *Proceedings of 28th ACM STOC* **A425** (1996), 212-219.

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

[12] W.K. Wootters and W.H. Zurek, A single quantum cannot be copied, *Nature* **299** (1982), 802-803.

[13] C. Bennet, E. Bernstein, G. Brassard and U. Vazirani, Strengths and weaknesses of quantum computing, *SIAM Journal on Computing* **26(5)** (1997), 1510-1523.

[14] C. Bennet, G. Brassard, C. Crépeau, R. Jozsa, A. Perez and W. Wootters, Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels, *Physical Review Letters* **70** (1993), 1895-1899.

[15] C. Bennet and S. Wiesner, Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states, *Physical Review Letters* **69** (1992), 2881-2884.

[16] A. Einstein, B. Podolsky and N. Rosen, Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?, *Physical Review* **47** (1935), 777.

[17] C.E. Shannon, A mathematical theory of communication, *Bell System Technical Journal* **27** (1948), 379-423.

[18] A.C.-C. Yao, Quantum circuit complexity, *Proceedings of the 34th IEEE FOCS* (1993), 352-360.

[19] G. Leander, A. May, Grover Meets Simon - Quantumly Attacking the FX-construction, *Takagi, T., Peyrin, T. (eds.) Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II. Lecture Notes in Computer Science* **10625** (2017), 161-178.

[20] X. Bonnetain, M. Naya-Plasencia, Hidden Shift Quantum Cryptanalysis and Implications, *preprint* (2018).

[21] U. Vazirani, Notes of the Course CS294-2 Quantum Computation, *University of Berkeley* (2004).

[22] G. Brassard, P. Hoyer, M. Mosca, A. Tapp, Quantum amplitude amplification and estimation, *Contemporary Mathematics* **305** (2002), 53-74.

[23] D.J. Bernstein, T. Lange, Post-quantum cryptography, *Nature* **549** (2017), 188-194.

[24] D.J. Bernstein, J. Buchmann, E. Dahmen, Post-quantum cryptography, *Springer-Verlag Berlin Heidelberg* (2009).

[25] G. Alagic, A. Russell, Quantum-Secure Symmetric-Key Cryptography Based on Hidden Shifts, *Coron, J.S., Nielsen, J.B. (eds.) EUROCRYPT (3). LNCS* **10212** (2017), 65-93.

[26] D. Boneh, M. Zhandry, Secure signatures and chosen ciphertext security in a quantum computing world, *Advances in Cryptology-CRYPTO 2013. Springer* (2013), 361-379.

[27] D.J. Bernstein, The Poly1305-AES Message-Authentication Code, *FSE. LNCS* **3557** (2005), 32-49.

[28] S. Even, Y. Mansour, A construction of a cipher from a single pseudorandom permutation, *J. Cryptology* **10**(3) (1997), 151-162.

[29] M. Kaplan, G. Leurent, A. Leverrier, M. Naya-Plasencia, Breaking symmetric cryptosystems using quantum period finding, *Robshaw, M., Katz, J., eds: Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II. Volume 9815 of Lecture Notes in Computer Science, Springer* (2016), 207-237.

[30] J. Kilian, P. Rogaway, How to protect DES against exhaustive key search, *Koblitz, N., ed.: Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings. Volume 1109 of Lecture Notes in Computer Science* (1996), 252-267.

[31] J. Kilian, P. Rogaway, How to protect DES against exhaustive key search (an analysis of DESX), *J. Cryptology* **14**(1) (2001), 17-35.

[32] H. Kuwakado, M. Morii, Quantum distinguisher between the 3-round feistel cipher and the random permutation, *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings, IEEE* (2010), 2682-2685.

[33] H. Kuwakado, M. Morii, Security on the quantum-type even-mansour cipher, *Proceedings of the International Symposium on Information Theory and its Applications, ISITA 2012, Honolulu, HI, USA, October 28-31, 2012, IEEE* (2012), 312-316.

[34] G. Kuperberg, A Subexponential-Time Quantum Algorithm for the Dihedral Hidden Subgroup Problem, *SIAM J. Comput.* **35**(1) (2005), 170-188.

[35] NIST, Post quantum project, *http://csrc.nist.gov/groups/ST/ post-quantum-crypto/* (2017).

[36] https://www.technologyreview.com/s/609451/ibm-raises-the-bar-with-a-50-qubit-quantum-computer/

[37] https://www.sciencenews.org/article/google-moves-toward-quantum-supremacy-72-qubit-computer

[38] https://www.digitaltrends.com/computing/intel-quantum-computing-research-lab/

[39] Quantum-Safe Cryptography (QSC); Quantum-safe algorithmic framework, ETSI GR QSC 001 V1.1.1 (2016-07).

[40] R. Merkle, Secrecy, authentication and public key systems / A certified digital signature, *PPh.D. dissertation, Dept. of Electrical Engineering, Stanford University* (1979).

[41] R. McEliece, A Public-Key Cryptosystem Based On Algebraic Coding Theory, *DSN Progress Report* **44** (1978), 114-116.

[42] J. Hoffstein, J. Pipher, J.H. Silverman, NTRU: A ring-based public key cryptosystem, *Buhler J.P. (eds) Algorithmic Number Theory. ANTS 1998. Lecture Notes in Computer Science* **1423** (1998).

[43] J. Patarin, Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms, *Maurer, U.M. (ed.) EUROCRYPT* **1070** (1996), 33-48.

[44] J. Daemen, V. Rijmen, The Block Cipher Rijndae, *Quisquater JJ., Schneier B. (eds) Smart Card Research and Applications. CARDIS* **1820** (1998).

Index

Algorithm, 29
  Amplitude Amplification, 64
  Deutsch-Jozsa, 41
  Euclid, 52
  Grover, 60
  Shor quantum factoring, 50
  Simon, 47

Bell basis, 38

Circuit
  Boolean, 30
  Classical, 30
  Quantum, 31
Continued fractions, 56
CPA-setting, 66

Energy Eigenstates, 24
Entanglement, 26
EPR pair, 26
EPR-pair, 36
Even-Mansour construction, 66

FX construction, 67

Gate
  Bitflip, 31
  Classical, 30
  CNOT, 32
  Hadamard, 32
  Phase, 32
  Phaseflip, 31
  Quantum, 31
  Toffoli, 33
Ground State, 24

Hidden Goppa code, 74

Matrix
  Density, 23
  Pauli, 21
Measurement, 17
  Positive Operator Valued Measurement, 20
  Projective, 20

NTRU, 76

Observable, 20
Operator
  Unitary, 21

Post-quantum cryptography, 65
Problem
  Bernstein-Vazirani, 46
  Deutsch-Jozsa, 41
  Factorization, 51
  Order-finding, 52
  Simon, 47
  Unstructured Search, 60

Quantum Fourier Transform, 49
Quantum parallelism, 40
Query, 40

Schrödinger Equation, 24
Spectral Gap, 24
State
  Bell, 26
  Mixed, 27
  Pure, 26
State space, 23
State vector, 23
SVP, 75

Theorem
  Chinese Remainder Theorem, 53
  Fermat's Little Theorem, 52
Trace, 22
Turing machine, 31