Alma Mater Studiorum · Università di Bologna

Scuola di Scienze
Dipartimento di Fisica e Astronomia
Corso di Laurea in Fisica

QUANTUM PHASE ESTIMATION ALGORITHMS

Relatore:
Prof.ssa Elisa Ercolessi

Presentata da:
Tommaso Antonelli

Correlatore:
Dott. Federico Dell'Anna

Anno Accademico 2021/2022

Abstract

Al contrario dei computer classici, i computer quantistici lavorano tramite le leggi della meccanica quantistica, e pertanto i qubit, ovvero l'unità base di informazione quantistica, possiedono proprietà estremamente interessanti di sovrapposizione ed entanglement. Queste proprietà squisitamente quantistiche sono alla base di innumerevoli algoritmi, i quali sono in molti casi più performanti delle loro controparti classiche. Obiettivo di questo lavoro di tesi è introdurre dal punto di vista teorico la logica computazionale quantistica e di riassumere brevemente una classe di tali algoritmi quantistici, ossia gli algoritmi di Quantum Phase Estimation, il cui scopo è stimare con precisione arbitraria gli autovalori di un dato operatore unitario. Questi algoritmi giocano un ruolo cruciale in vari ambiti della teoria dell'informazione quantistica e pertanto verranno presentati anche i risultati dell'implementazione degli algoritmi discussi sia su un simulatore che su un vero computer quantistico.

Contents

Introduction

Quantum computation is the field that studies the information processing tasks that can be implemented via a device governed by the laws of quantum mechanics. As straightforward as this idea may sound, this field only emerged in recent years and still faces serious and intriguing challenges that make it a very active and vibrant area of research. In fact, it is widely believed that quantum computers will be in many tasks much more powerful than what classical computers will ever be, and for their features they will find applications in the most diverse fields, ranging from cryptography to simulation of many-body systems.

The story begins in the early 1980s with the works primarily of Paul Benioff [1], who first thought about a quantum mechanical model of a computer, and of Richard Feynman [2], who pointed out the potentiality of a quantum device of simulating efficiently a quantum system. This culminated in 1985 with the work of David Deutsch [3] that described the first theoretical model of a Universal Quantum Computer, the quantum analog of the classical Turing Machine. In the subsequent years, the major theoretical breakthroughs were represented by the works of Peter Shor (1994) [4] and Lov Grover (1996) [5]. Shor demonstrated that the problem of finding the prime factors of an integer and the problem of the discrete logarithm could be performed on a quantum device in polynomial time, while classically they would require exponential time. Grover instead found an algorithm that could outperform classical computers in the task of searching through an unstructured database. These algorithms, in virtue of their widespread applicability, brought considerable attention to this field.

As of today, the theoretical foundations of quantum computing are pretty much well understood. The prevailing model of quantum computation is the Quantum Circuit model, in which computation is seen as a series of quantum gates and measurements applied to a collection of qubits, which are the quantum analog of the classical bits of information. In this model, the qubits enjoy the quantum mechanical properties of superposition and entanglement, which are the keys to understanding the potential of quantum computation over classical computation. The power of quantum computation is at the heart of the studies of Quantum Complexity Theory, which strongly suggests that quantum computers are more powerful than classical ones, although this is not a proven statement. This hypothesis, that goes by the name of "quantum supremacy", is

supported by a plethora of very efficient quantum algorithms that exploit the aforementioned quantum mechanical properties of the qubits.

In parallel to the development of the theory of quantum computing, it began the engineering challenge to physically realise such a quantum computer. To this day there are many different candidates, distinguished by the physical systems used to implement the qubits. To mention a few, there are Superconductor Quantum Computing (on which companies as Google or IBM conduct today's research), Trapped Ion Quantum Computers, and Neutral Atoms in Optical Lattices. In order to be considered a practical quantum device, all these various implementations must satisfy some requirements, known as DiVincenzo criteria (1996) [6]: scalability with the number of qubits, possibility of initializing the qubits, implementability of a set of universal quantum gates, capability of measurement, and decoherence times longer than gate-operation times. In particular quantum decoherence, that is the loss of quantum information of a non-isolated system with the surroundings, represents a significant challenge in the realization of quantum computers, since no qubit can be thought as perfectly isolated if one needs to act on it via a quantum gate. The mitigation of decoherence errors is studied by Quantum Error Correction techniques.

To this day, there is the possibility for anyone to implement and run its own quantum algorithm. Mainly IBM grants public or premium cloud-based access to its quantum processors through IBM Quantum Experience service. On such quantum processors one can easily create, run and simulate quantum circuits by means of the open-source software development kit Qiskit.

On the other side, despite considerable efforts (IBM has even launched the first commercial quantum computer, IBM Q System One), the hardware limitations of these devices are still significant, and further research needs to be done in order to practically achieve quantum supremacy. As such, for instance, one can think of various algorithms for the same problem that yield the result with different accuracy, depending on the implementation features of such algorithms. One example is given by Quantum Phase Estimation algorithms.

Quantum Phase Estimation algorithms, which are precisely the topic of this thesis, are a class of quantum algorithms that have as objective the estimation with arbitrary precision of the eigenvalue of a unitary operator, which is a complex phase. As pointless as this may sound, phase estimation is mainly used as a subroutine in many other quantum algorithms of interest, such as Shor's factoring algorithm, or the HHL algorithm for solving linear systems of equations. In addition, some of these algorithms take advantage of the concept of the Quantum Fourier Transform, which is one of the key quantum algorithms that provides exponential speedup over the classical analog, and is used in many applications.

As such, the first chapter is mainly focused on introducing the key ideas to quantum computing: what is a qubit, what is a quantum gate, how to build quantum circuits and

some interesting examples that shed light on the advantages and differences of quantum computation over classical computation.

The second chapter is instead devoted to introduce the various Quantum Phase Estimation Algorithms of interest, which are Kitaev's algorithm, IQFT QPE and Inverse-AQFT QPE, and discuss their pros and cons. In this chapter we further introduce the aforementioned algorithm of Quantum Fourier Transform, which is relevant to two of these Quantum Phase Estimation algorithms.

Finally, the third chapter contains the Qiskit implementation of the algorithms discussed in the previous chapter, as well as the results obtained by running them on a simulator and on a real IBM quantum device, where one can effectively see the state of the art of these quantum processors.

Chapter 1

Quantum computation logic

In this chapter we will give an introductory overview on the main features of quantum computation that make it different from classical computation. We will describe the quantum mechanical space of qubits and the operations that can manipulate such qubits, which are effectively the ingredients that are needed to understand what quantum logic is and how powerful it can be. The main references of this chapter are [7] and [8].

1.1 Single qubit

Classical computation and information theory is built upon the concept of the *classical bit*, which is a system that can live in either one of two *classical states*: 0 or 1.

In a similar fashion, when dealing with quantum computation, we come across the concept of the *quantum bit* or *qubit*, which again similarly is a system that can have two *quantum states*: $|0\rangle$ or $|1\rangle$. The main difference between these two entities is that the quantum bit can live in a *linear superposition* of its two states, according to the rules of quantum mechanics. Namely, a qubit can be represented by a physical ket $|\psi\rangle$ that lives in a two-dimensional complex Hilbert space $\mathcal{H}_2$:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{1.1.1}$$

where $\alpha, \beta \in \mathbb{C}$ are the coefficients of the expansion of $|\psi\rangle$ in the $|0\rangle, |1\rangle$ orthonormal basis of $\mathcal{H}_2$.

Since the space $\mathcal{H}_2$ is two-dimensional over the complex field, one would expect the qubit $|\psi\rangle$ to be parametrized by 4 real parameters, but this is not the case. The requirement of physicality of the ket $|\psi\rangle$ imposes a probabilistic intepretation, which results in the constraint

$$\langle\psi|\psi\rangle = 1 \quad \Longleftrightarrow \quad |\alpha|^2 + |\beta|^2 = 1 \tag{1.1.2}$$

Upon this normalization constraint, a measurement of the qubit gives the result 0 with probability $|\alpha|^2$, and gives the result 1 with probability $|\beta|^2$.

Furthermore the ket $|\psi\rangle$ is invariant for what regards its physical interpretation under a change of its global phase:

$$|\psi\rangle \;\mapsto\; e^{i\phi}\,|\psi\rangle\,, \qquad \phi \in \mathbb{R} \tag{1.1.3}$$

which amounts to a degree of freedom that can be arbitrarily set to a convenient value without changing the observable properties of our system.

These two constraints reduce the number of real degrees of freedom from 4 to 2, which implies that the space of qubits is a two-dimensional real manifold. To see this more clearly, let's compute an explicit expression for the coefficients α and β. In polar form

$$\alpha = a\,e^{i\chi} \quad \beta = b\,e^{i\zeta}, \qquad a, b \in [0, +\infty[\quad \chi, \zeta \in [0, 2\pi[\tag{1.1.4}$$

From the constraint (1.1.2) we can conclude that

$$a^2 + b^2 = 1 \tag{1.1.5}$$

and that allows us to reparametrize a and b in the following form, without loss of generality:

$$a = \cos\frac{\theta}{2} \quad b = \sin\frac{\theta}{2}, \qquad \theta \in [0, \pi] \tag{1.1.6}$$

Furthermore, from the invariance with respect to the transformation (1.1.3), we can choose the global phase of $|\psi\rangle$ such that $\alpha = \cos(\theta/2) \in \mathbb{R}$. We thus have constructed a more explicit form for $|\psi\rangle$ which captures the relevant and physical degrees of freedom of a single qubit:

$$|\psi\rangle = \cos\frac{\theta}{2}\,|0\rangle + \sin\frac{\theta}{2}\,e^{i\varphi}\,|1\rangle\,, \qquad \theta \in [0, \pi] \quad \varphi \in [0, 2\pi[\tag{1.1.7}$$

From the periodicity properties of this expression, we can see that θ and φ clearly represent respectively the polar and azimuthal angles on a sphere, which is called the *Bloch sphere*, depicted in Fig. 1.1.

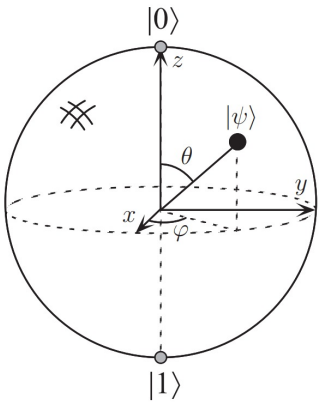

Figure 1.1: Bloch sphere for a single qubit, from [7].

We have therefore found the correct manifold structure of the space of physical states for a qubit. Manipulations on a single qubit can thus be seen as rotations on the Bloch sphere, as we will show in more detail in the following sections.

1.2 Multiple qubits

Since for any practical computation one qubit is not necessary, in this section we will revolve our attention to multiple qubit states.

Firstly, let's focus our attention on a 2-qubit system. The space of states of this composed system is the *tensor product* of the spaces of the two single qubits: $\mathcal{H}_2 \otimes \mathcal{H}_2$. A convenient choice of orthonormal basis of this space is represented by the four kets $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$, which encode the four possible combinations of states of the two qubits. The most general 2-qubit state can thus be represented by a linear superposition of the basis states, and it is given by the state ket $|\psi\rangle$:

$$|\psi\rangle = \alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle + \alpha_{10}\,|10\rangle + \alpha_{11}\,|11\rangle \tag{1.2.1}$$

which again, in order to recover a probabilistic interpretation, must satisfy the constraint:

$$\langle\psi|\psi\rangle = 1 \quad \Longleftrightarrow \quad \sum_{x\in\{0,1\}^2} |\alpha_x|^2 = 1 \tag{1.2.2}$$

Similarly to the single qubit, one obtains the measurement result $x \in \{00, 01, 10, 11\}$ with probability $|\alpha_x|^2$, but one can also measure only one qubit. For example, measuring the first qubit gives 0 with probability $|\alpha_{00}|^2 + |\alpha_{01}|^2$, while it gives 1 with probability $|\alpha_{10}|^2 + |\alpha_{11}|^2$.

Another fruitful orthonormal basis of this space is given by the so-called *Bell states* or *EPR pairs*:

$$\begin{cases} |\beta_{00}\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \\ |\beta_{01}\rangle = \frac{1}{\sqrt{2}}\left(|01\rangle + |10\rangle\right) \\ |\beta_{10}\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle\right) \\ |\beta_{11}\rangle = \frac{1}{\sqrt{2}}\left(|01\rangle - |10\rangle\right) \end{cases} \tag{1.2.3}$$

These four states form a set of *quantum entangled pairs*, which means that the two qubits are *correlated*. Take for example $|\beta_{00}\rangle$: if one measures the first qubit the result can be either 0 or 1 with equal probabilities 1/2, but then the second qubit is consequently forced to give the same result of the first one under measurement.

More formally, an entangled pair is described by a *non-decomposable tensor* in the space $\mathcal{H}_2 \otimes \mathcal{H}_2$, which represents a quantum state of the composite system that cannot be reduced to a product of single qubit states. Indeed, one can check that the Bell states $|\beta_{xy}\rangle$ are non-decomposable states.

If we then want to describe a n-qubit system, we can easily see that the space of states of this composite system is the tensor product $\mathcal{H}_2^{\otimes n}$, where a basis is given by the 2^n kets $|x\rangle$, $x \in \{0,1\}^n$. A general state ket is then given by the linear superposition

$$|\psi\rangle = \sum_{x\in\{0,1\}^n} \alpha_x |x\rangle \tag{1.2.4}$$

for which the usual normalization constraint is required. Any n-qubit state is therefore specified by 2^n complex amplitudes.

1.3 Quantum gates

Now that we have found the mathematical structure to describe correctly quantum bits of information, we are interested in manipulating these bits to effectively implement a quantum algorithm.

Since we are working in the realm of quantum mechanics, our manipulation of qubits should be done accordingly to the laws of this theory. Manipulating a quantum state in practice means taking a normalized ket $|\psi\rangle$ of an Hilbert space $\mathcal{H}$ and applying an operator U to obtain a second normalized ket $|\psi'\rangle$ in the same Hilbert space. This operator U can be seen as a time-evolution operator.

The rules of quantum mechanics are strict about U. Firstly, it must preserve the linear structure of the Hilbert space, which in physical terms implies that states evolve independently when they are in a linear superposition, and in mathematical terms implies that U is linear:

$$U(\alpha\,|\psi\rangle + \beta\,|\phi\rangle) = \alpha\; U\,|\psi\rangle + \beta\; U\,|\phi\rangle \tag{1.3.1}$$

Secondly, it must preserve the probabilistic interpretation of the state ket, which mathematically implies that

$$\langle\psi|\psi\rangle = \langle\psi'|\psi'\rangle = 1 \tag{1.3.2}$$

which can be easily seen to be an equivalent condition to U being a unitary operator:

$$U^\dagger U = I \quad \iff \quad U^\dagger = U^{-1} \tag{1.3.3}$$

It's a remarkable result that these are the only constraints that U must satisfy; any unitary operator is a valid quantum gate.

From a circuital point of view, we can see the unitary operator U as a "black box" which in general acts on a state $|\psi\rangle$ of n qubits (reversibly, since U^{-1} exists) and produces a new state $|\psi'\rangle$ of n qubits, which is summarized in Fig. 1.2.

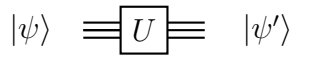

Figure 1.2: Circuital representation of a generic U gate.

Now let's focus on a single qubit system, described by a state ket in $\mathcal{H}_2$. An important gate in this system is the NOT gate X which acts on the basis $|0\rangle$, $|1\rangle$ as such:

$$X|0\rangle = |1\rangle \quad X|1\rangle = |0\rangle \tag{1.3.4}$$

and therefore acts on a general qubit state in the form of (1.1.1) as such:

$$X(\alpha|0\rangle + \beta|1\rangle) = \beta|0\rangle + \alpha|1\rangle \tag{1.3.5}$$

A more compact notation consists of writing the state (1.1.1) as a column vector in the $|0\rangle$, $|1\rangle$ basis:

$$\psi = \begin{bmatrix}\alpha\\ \beta\end{bmatrix} \tag{1.3.6}$$

and the quantum NOT gate as a matrix:

$$X = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} \tag{1.3.7}$$

The action of the quantum gate is therefore synthesized in the row-column product of X and ψ:

$$X\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \begin{bmatrix}\beta\\ \alpha\end{bmatrix} \tag{1.3.8}$$

Other important gates on a single qubit (always written in matrix form) are the Y and Z gates:

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{1.3.9}$$

and the *Hadamard* gate:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{1.3.10}$$

which turns $|0\rangle$ into the superposition $(|0\rangle + |1\rangle)/\sqrt{2}$, and turns $|1\rangle$ into $(|0\rangle - |1\rangle)/\sqrt{2}$.

Very important is also the family of the so-called *phase* gates

$$P(\varphi) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}, \qquad \varphi \in [0, 2\pi[\tag{1.3.11}$$

where the most notable ones are the Z, S and T gates:

$$Z = P(\pi) = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad S = P\left(\frac{\pi}{2}\right) = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \quad T = P\left(\frac{\pi}{4}\right) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \tag{1.3.12}$$

Together, the X, Y and Z gates are the 3 *Pauli matrices*:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{1.3.13}$$

It's a fact that a general unitary operator U on $\mathcal{H}_2$ can be expressed as such:

$$U = e^{i\alpha} R_{\hat{n}}(\theta) = e^{i\alpha} \left[\cos\left(\frac{\theta}{2}\right) I - i \sin\left(\frac{\theta}{2}\right) (n_x X + n_y Y + n_z Z) \right] \tag{1.3.14}$$

where $\alpha \in [0, 2\pi[$, $\theta \in [0, \pi]$ and (n_x, n_y, n_z) are the 3 cartesian components of the versor $\hat{n}$, so $n_x^2 + n_y^2 + n_z^2 = 1$.

The $R_{\hat{n}}(\theta)$ operator has a nice interpretation within the Bloch sphere picture: it encodes a 3D rotation of θ degrees around the versor $\hat{n}$ that affects the 3-vector representing the qubit. In this way, every quantum gate involving a single qubit can be seen, up to a phase, as a rotation on the Bloch sphere. For example, the Hadamard gate is parameterized by $\alpha = \pi/2$, $\theta = \pi$, $\hat{n} = (1, 0, 1)/\sqrt{2}$.

Otherwise, another useful parameterization for U is through Euler angles:

$$\begin{aligned} U &= e^{i\alpha} R_z(\beta) R_y(\gamma) R_z(\delta) = \\ &= e^{i\alpha} \begin{bmatrix} e^{-i\beta/2} & 0 \\ 0 & e^{i\beta/2} \end{bmatrix} \begin{bmatrix} \cos(\gamma/2) & -\sin(\gamma/2) \\ \sin(\gamma/2) & \cos(\gamma/2) \end{bmatrix} \begin{bmatrix} e^{-i\delta/2} & 0 \\ 0 & e^{i\delta/2} \end{bmatrix} \end{aligned} \tag{1.3.15}$$

and if we set $A = R_z(\beta)R_y(\gamma/2)$, $B = R_y(-\gamma/2)R_z(-(\delta+\beta)/2)$ and $C = R_z((\delta-\beta)/2)$, one can check that we have obtained three unitary matrices with the properties

$$U = e^{i\alpha}AXBXC, \qquad ABC = I \tag{1.3.16}$$

where X is the NOT gate. This decomposition will turn to be useful later on.

A general quantum gate for a single bit is expressed through its circuital representation as a box with the name of the gate on top. An example is shown in Fig. 1.3.

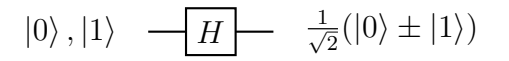

Figure 1.3: Circuital representation of the Hadamard gate.

We may now consider the case of quantum gates involving n qubits. As one can imagine, the variety of unitary operators U is much richer, since the Hilbert space $\mathcal{H}_2^{\otimes n}$ is much bigger than $\mathcal{H}_2$.

A prototype of multi-qubit quantum gate is the *controlled*-NOT gate or simply CNOT gate U_{CN}, which acts on 2 qubits: the *control* qubit and the *target* qubit. It acts on the basis of $\mathcal{H}_2^{\otimes 2}$ as such:

$$U_{CN}|00\rangle = |00\rangle \quad U_{CN}|01\rangle = |01\rangle \quad U_{CN}|10\rangle = |11\rangle \quad U_{CN}|11\rangle = |10\rangle \tag{1.3.17}$$

which can be summarized in the following notation:

$$|x, y\rangle \;\mapsto\; |x, y \oplus x\rangle, \qquad x, y \in \{0, 1\} \tag{1.3.18}$$

where $\oplus$ is the addition modulo 2, which is equivalent to a classical XOR gate. This gate basically flips the logical state of the second qubit (the target) if and only if the first qubit (the control) is activated.

From the matrix representation of U_{CN} ($2^2 \times 2^2$ matrix)

$$U_{CN} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{1.3.19}$$

one can easily see that this gate is indeed unitary, since $U_{CN}^\dagger U_{CN} = I$, hence it is a valid quantum gate.

Another useful way to visualize the CNOT gate is through its circuital representation, shown in Fig. 1.4, which emphasizes the action on the target qubit (bottom line) controlled by the control qubit (top line).

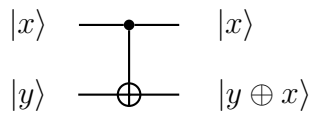

Figure 1.4: Circuital representation of CNOT.

Another useful gate in quantum computation is the so-called *Toffoli* gate which acts on 3 qubits in the following manner:

$$|x, y, z\rangle \;\mapsto\; |x, y, z \oplus xy\rangle\,, \qquad x, y, z \in \{0, 1\} \tag{1.3.20}$$

Basically the first two qubits act as control qubits, and when they are both set they flip the sign of the third one.

In matrix representation the Toffoli gate is a $2^3 \times 2^3$ matrix:

$$U_T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{1.3.21}$$

with circuital representation given in Fig. 1.5.

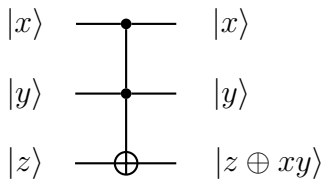

Figure 1.5: Circuital representation of Toffoli.

A first conceptual use of the Toffoli gate is proving that quantum computation can fully implement classical algorithms: the Toffoli gate can be set to operate as a NAND or a FANOUT gate (see Fig. 1.6), which are two universal gates for classical computation, provided that we dispose of enough qubits.

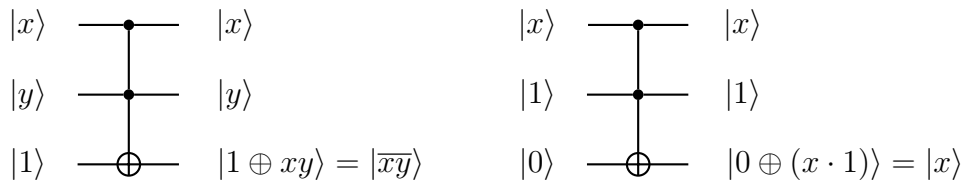

Figure 1.6: On the left, a Toffoli gate simulating a NAND on the target qubit. On the right, a Toffoli gate simulating a FANOUT on the target qubit.

However, the power of quantum computation is not limited to merely replicating classical computation, as one can imagine numerous unitary matrices that do not have a classical counterpart: for example the Hadamard gate can superpose the qubit states $|0\rangle$ and $|1\rangle$.

1.3.1 No-cloning theorem

Another remarkable difference between classical and quantum computation is that in quantum computation a generic qubit $|\psi\rangle$ cannot be copied. This fact goes by the name of *no-cloning theorem* and it is one of the main results of quantum information theory.

More precisely, the no-cloning theorem states that a given arbitrary state $|\psi\rangle$ of an Hilbert space $\mathcal{H}$ cannot be copied into a fixed state $|s\rangle$ of $\mathcal{H}$. In other terms, it does not exists a unitary gate U that realizes the evolution:

$$\begin{aligned} U \;:\; \mathcal{H} \otimes \mathcal{H} \;&\to\; \mathcal{H} \otimes \mathcal{H} \\ |\psi\rangle \otimes |s\rangle \;&\mapsto\; |\psi\rangle \otimes |\psi\rangle\,, \qquad \forall\, |\psi\rangle \end{aligned} \tag{1.3.22}$$

This can be inferred from the constraints of linearity and unitarity we put on U.

If we have two different states $|\psi\rangle$ and $|\phi\rangle$ that we somehow managed to duplicate with the U gate, we can see that due to the constraint of linearity, we cannot duplicate an arbitrary superposition $\alpha\,|\psi\rangle + \beta\,|\phi\rangle$, with $|\alpha|^2 + |\beta|^2 = 1$:

$$\begin{aligned} &(\alpha\,|\psi\rangle + \beta\,|\phi\rangle) \otimes |s\rangle = \\ &= \alpha\,|\psi\rangle \otimes |s\rangle + \beta\,|\phi\rangle \otimes |s\rangle \;\overset{U}{\mapsto}\; \alpha\,|\psi\rangle \otimes |\psi\rangle + \beta\,|\phi\rangle \otimes |\phi\rangle \end{aligned} \tag{1.3.23}$$

The final state in general is clearly different from the duplicate of $\alpha\,|\psi\rangle + \beta\,|\phi\rangle$:

$$\begin{aligned} &(\alpha\,|\psi\rangle + \beta\,|\phi\rangle) \otimes (\alpha\,|\psi\rangle + \beta\,|\phi\rangle) = \\ &= \alpha^2\,|\psi\rangle \otimes |\psi\rangle + \beta^2\,|\phi\rangle \otimes |\phi\rangle + \alpha\beta(|\psi\rangle \otimes |\phi\rangle + |\phi\rangle \otimes |\psi\rangle) \end{aligned} \tag{1.3.24}$$

In fact, they are only equal when

$$\begin{cases} \alpha^2 = \alpha \\ \beta^2 = \beta \\ \alpha\beta = 0 \\ |\alpha|^2 + |\beta|^2 = 1 \end{cases} \tag{1.3.25}$$

which implies that either $\alpha = 1, \beta = 0$ or $\alpha = 0, \beta = 1$, which means that U can only copy $|\psi\rangle$ and $|\phi\rangle$ when they are not superposed.

To characterize more precisely which states can be copied or not, we use the constraint of unitarity of U. Suppose again we have two states $|\psi\rangle$ and $|\phi\rangle$ that we are able to duplicate. Now we can see that the unitary gate U preserves the inner product between states:

$$\begin{aligned} \langle\phi|\psi\rangle^2 = \langle\phi|\psi\rangle \langle\phi|\psi\rangle &= (\langle\phi| \otimes \langle\phi|)(|\psi\rangle \otimes |\psi\rangle) = \\ &= (\langle\phi| \otimes \langle s|)U^\dagger U(|\psi\rangle \otimes |s\rangle) = \\ &= (\langle\phi| \otimes \langle s|)(|\psi\rangle \otimes |s\rangle) = \langle\phi|\psi\rangle \langle s|s\rangle = \\ &= \langle\phi|\psi\rangle \end{aligned} \tag{1.3.26}$$

which implies that either $\langle\phi|\psi\rangle = 1$ or $\langle\phi|\psi\rangle = 0$, which in turn means that either $|\psi\rangle$ and $|\phi\rangle$ are the same state, or they are mutually orthogonal.

So we can see that the unitarity condition on U forces the states that can be duplicated to be orthogonal to each other, and superpositions are not allowed to be duplicated. This reconciles with the FANOUT operation that we have seen in the previous section: the Toffoli gate can only duplicate the qubits $|0\rangle$ and $|1\rangle$, which are mutually orthogonal in $\mathcal{H}_2$, but it cannot duplicate an arbitrary superposition of these two states.

1.4 Quantum circuits

The problem is now finding a way to combine efficiently basic quantum gates in order to obtain the needed arbitrary unitary gate U.

It's a remarkable fact [7] that the CNOT gate and single qubit gates form a set of *universal quantum gates*, which means that every unitary gate U can be made up by a clever combination of these gates only. If we then wish to use just a discrete set of gates, we cannot obviously represent exactly every unitary operator U, since the set of unitary operators is continuous. Nevertheless, one can demonstrate that we can approximate a general U with arbitrary accuracy only using a discrete set of gates.

To see a basic example, consider the *swap* gate, which acts on 2 qubits as such:

$$|x, y\rangle \;\mapsto\; |y, x\rangle\,, \qquad x, y \in \{0, 1\} \tag{1.4.1}$$

This gate can be expressed as a combination of 3 CNOT gates, as shown in Fig. 1.7.

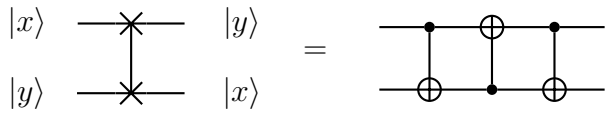

Figure 1.7: The swap gate (on the left) and its realization through CNOTs (on the right).

The equivalence between the two circuits in Fig. 1.7 can be easily verified:

$$\begin{aligned} |x, y\rangle &\mapsto |x, y \oplus x\rangle \\ &\mapsto |x \oplus (y \oplus x), y \oplus x\rangle = |y, y \oplus x\rangle \\ &\mapsto |y, (y \oplus x) \oplus y\rangle = |y, x\rangle \end{aligned} \tag{1.4.2}$$

Another nice example is the circuit given in Fig. 1.8 that entangles 2 qubits, effectively creating a Bell state.

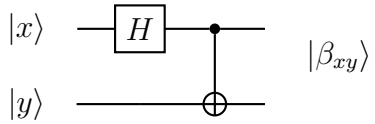

Figure 1.8: Circuit that implements the creation of Bell states.

In this circuit the Hadamard gate superposes the $|0\rangle$ and $|1\rangle$ states, and the CNOT gate acts in a linear manner, effectively entangling the target qubit to the control qubit.

Particularly useful is also the operation of the controlled-U gate on 2 qubits, as shown in Fig. 1.9.

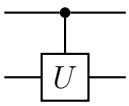

Figure 1.9: Circuital representation of the controlled-U gate.

This circuit basically implements an if: if the control qubit is set, then the operator U acts on the target qubit. A pratical way to implement is through the decomposition (1.3.16), as shown in Fig. 1.10.

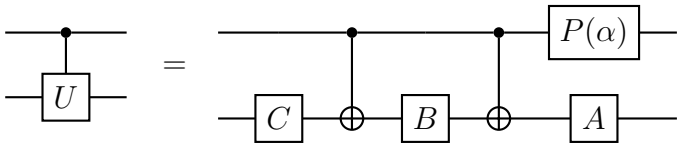

Figure 1.10: Circuital implementation of the controlled-U gate.

The claim of equivalence of the two circuits in Fig. 1.10 can be easily seen, as the phase gate $P(\alpha)$ on the control qubit is equivalent to a controlled-$e^{i\alpha}$ gate that affects the target qubit. So if the control qubit is set, then $U = e^{i\alpha}AXBXC$ is applied to the target qubit, otherwise $I = ABC$ is applied.

In conclusion, an arbitrary operation U on n qubits can be implemented only using CNOT and single qubit gates, but it turns out that this construction requires a number of gates that grows like $O(n^2 4^n)$, which is not very efficient. Thus, in our search of optimal quantum algorithms we clearly need a different and more clever approach.

1.4.1 Quantum Teleportation

A first interesting quantum circuit is the one that realizes *quantum teleportation*. The name may be quite misleading, since there is no real teleportation of matter, but only of quantum information.

In practice, this circuit exploits the potentiality of quantum entanglement to transfer the content of an unknown qubit $|\psi\rangle$ from point A to point B, using a pair of entangled qubits and two classical bits. The quantum teleportation protocol is summarized in Fig. 1.11.

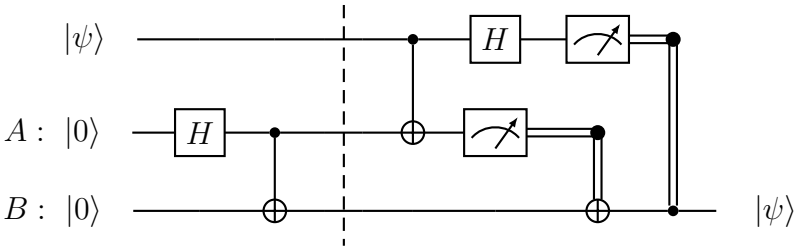

Figure 1.11: Quantum teleportation protocol.

The protocol goes as follows: a pair of qubits are entangled in the state $|\beta_{00}\rangle$ (as one can see in the first portion of Fig. 1.11), and then they can be placed wherever in the universe, the first one at point A and the second one at point B.

Subsequently, a mysterious qubit $|\psi\rangle$ arrives at point A, and the system is manipulated locally through a CNOT and a Hadamard gate. If we write $|\psi\rangle$ as in (1.1.1), one

can see that these gates act as such:

$$\begin{aligned}
|\psi\rangle \otimes |\beta_{00}\rangle &= (\alpha\,|0\rangle + \beta\,|1\rangle) \otimes \frac{1}{\sqrt{2}}\,(|00\rangle + |11\rangle) \\
&= \frac{1}{\sqrt{2}}\,[\alpha\,|0\rangle \otimes (|00\rangle + |11\rangle) + \beta\,|1\rangle \otimes (|00\rangle + |11\rangle)] \\
&\mapsto \frac{1}{\sqrt{2}}\,[\alpha\,|0\rangle \otimes (|00\rangle + |11\rangle) + \beta\,|1\rangle \otimes (|10\rangle + |01\rangle)] \\
&\mapsto \frac{1}{\sqrt{2}}\left[\alpha\frac{1}{\sqrt{2}}\,(|0\rangle + |1\rangle) \otimes (|00\rangle + |11\rangle) + \beta\frac{1}{\sqrt{2}}\,(|0\rangle - |1\rangle) \otimes (|10\rangle + |01\rangle)\right] \\
&= \frac{1}{2}\,[\,|00\rangle \otimes (\alpha\,|0\rangle + \beta\,|1\rangle) + |01\rangle \otimes (\alpha\,|1\rangle + \beta\,|0\rangle) \\
&\qquad + |10\rangle \otimes (\alpha\,|0\rangle - \beta\,|1\rangle) + |11\rangle \otimes (\alpha\,|1\rangle - \beta\,|0\rangle)\,] \qquad (1.4.3)
\end{aligned}$$

The protocol then instructs to measure the two qubits at point A, which can be seen from (1.4.3) to produce every $x \in \{0,1\}^2$ with probability $1/4$. Once these two qubits are measured, and therefore converted into two classical bits, they need to be sent at point B, where a controlled-X and a controlled-Z operations are applied on the qubit at that point (the control bits are the classical bits themselves).

We therefore have four possible states for our last qubit, depending on the state of the two classical bits:

$$00: \quad \alpha\,|0\rangle + \beta\,|1\rangle \xmapsto{C-X} \alpha\,|0\rangle + \beta\,|1\rangle \xmapsto{C-Z} \alpha\,|0\rangle + \beta\,|1\rangle \qquad (1.4.4)$$

$$01: \quad \alpha\,|1\rangle + \beta\,|0\rangle \xmapsto{C-X} \alpha\,|0\rangle + \beta\,|1\rangle \xmapsto{C-Z} \alpha\,|0\rangle + \beta\,|1\rangle \qquad (1.4.5)$$

$$10: \quad \alpha\,|0\rangle - \beta\,|1\rangle \xmapsto{C-X} \alpha\,|0\rangle - \beta\,|1\rangle \xmapsto{C-Z} \alpha\,|0\rangle + \beta\,|1\rangle \qquad (1.4.6)$$

$$11: \quad \alpha\,|1\rangle - \beta\,|0\rangle \xmapsto{C-X} \alpha\,|0\rangle - \beta\,|1\rangle \xmapsto{C-Z} \alpha\,|0\rangle + \beta\,|1\rangle \qquad (1.4.7)$$

After these operations we have therefore faithfully reconstructed our starting qubit $|\psi\rangle$ at point B. This procedure however does not imply the possibility of superluminal messages, since the classical bits still need to be transferred from A to B to complete the protocol.

1.4.2 Grover's Algorithm

Another remarkable circuit is the one that implements *Grover's algorithm*, which exploits the advantage of quantum superposition to outperform classical computers in the task of searching through a database of N objects.

The problem is defined as such: let $f : \{0, 1, \ldots N-1\} \to \{0,1\}$ be a function such that it admits one and only one value ω in the domain for which $f(\omega) = 1$, while for every other $x \neq \omega$ we have that $f(x) = 0$. The task is to find ω.

Classically, the only viable method is evaluating the function at every possible input value and looking at the result: one can therefore understand that it takes $O(N)$ iterations of the program to find ω.

By using Grover's algorithm instead, the correct output ω is given with good probability by just $O(\sqrt{N})$ evaluations of the function. The circuit that performs Grover's algorithm is given in Fig. 1.12.

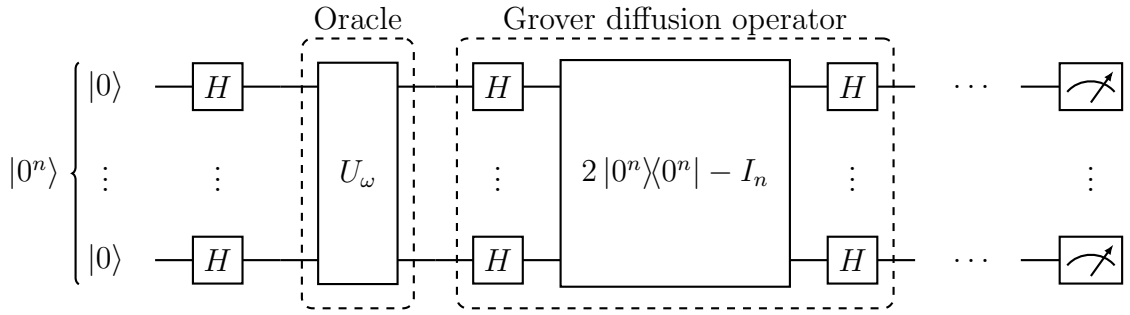

Figure 1.12: Circuital implementation of Grover's algorithm.

A few comments to Fig. 1.12 are in order. First of all, the N items database is represented through $n = \lceil \log_2 N \rceil$ qubits. Then n Hadamard gates in parallel are applied to the qubits, in order to obtain a state $|s\rangle$ which is the uniform superposition of all possible values of the domain:

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \tag{1.4.8}$$

This capability of quantum computers of superposing all possible input values, known as *quantum parallelism*, provides a great advantage over classical algorithms, since the function f will be evaluated simultaneously by linearity over the whole domain.

The function f is implemented into the *oracle* U_ω, which is a gate that acts as such:

$$\begin{cases} U_\omega\,|\omega\rangle = -\,|\omega\rangle \\ U_\omega\,|x\rangle = |x\rangle \qquad \text{if } x \neq \omega \end{cases} \iff \quad U_\omega\,|x\rangle = (-1)^{f(x)}\,|x\rangle$$
$$\iff \quad U_\omega = I_n - 2\,|\omega\rangle\langle\omega| \tag{1.4.9}$$

Then the circuit is composed of the so-called *Grover diffusion operator* U_s, which one can check from Fig. 1.12 to be:

$$U_s = H^{\otimes n}\,(2\,|0^n\rangle\langle 0^n| - I_n)\,H^{\otimes n} = 2\,|s\rangle\langle s| - I_n \tag{1.4.10}$$

This procedure of applying U_ω and U_s is repeated $r \approx \pi\sqrt{N}/4$ times to get ω as the result of the subsequent measure, with an error $O(1/N)$.

The correctness of these statements can be seen via a nice geometric proof. Consider the subspace of the Hilbert space spanned by the kets $|s\rangle$ and $|\omega\rangle$, as represented in Fig. 1.13. The system starts in $|s\rangle$, then the application of U_ω can be seen from (1.4.9) to reflect the state through the hyperplane orthogonal to $|\omega\rangle$. U_s can instead be interpreted from (1.4.10) as a reflection through $|s\rangle$.

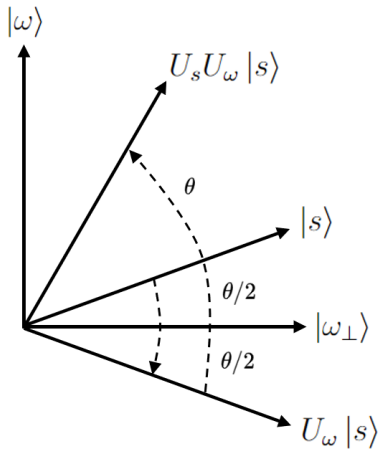

Figure 1.13: Geometric representation of Grover's algorithm.

Basically, at every iteration of $U_s U_\omega$, the system state vector is rotated by an angle θ in the plane spanned by $|s\rangle$ and $|\omega\rangle$, and after r iterations the probability of measuring $|\omega\rangle$ is:

$$P(\omega) = |\langle\omega| (U_s U_\omega)^r |s\rangle|^2 = \cos^2\left[\frac{\pi}{2} - \left(r + \frac{1}{2}\right)\theta\right] = \sin^2\left[(2r+1)\frac{\theta}{2}\right] \tag{1.4.11}$$

In turn, θ is given by:

$$\begin{aligned}\cos\theta &= \langle s| U_s U_\omega |s\rangle = \langle s| (2|s\rangle\langle s| - I)(I - 2|\omega\rangle\langle\omega|) |s\rangle = \langle s| (|s\rangle - 2|\omega\rangle \langle\omega|s\rangle) \\ &= 1 - 2|\langle\omega|s\rangle|^2 = 1 - 2\left|\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}\langle\omega|x\rangle\right|^2 = 1 - \frac{2}{N} \\ \implies \quad \sin\frac{\theta}{2} &= \frac{1}{\sqrt{N}}\end{aligned} \tag{1.4.12}$$

From (1.4.11), one can see that the maximum probability of getting $|\omega\rangle$ is achieved when $2r\,\theta/2 \approx \pi/2$, and from (1.4.12) this condition is met whenever

$$r \approx \frac{\pi}{4\,(\theta/2)} = \frac{\pi}{4\,\sin^{-1}(1/\sqrt{N})} \approx \frac{\pi\sqrt{N}}{4} \tag{1.4.13}$$

Plugging these formulas back into (1.4.11) one can also check that the error is of order $O(1/N)$.

Chapter 2

Quantum Phase Estimation

Quantum Phase Estimation (QPE) techniques denote a class of algorithms that aim to estimate with arbitrary precision the eigenvalues of a unitary operator U, which have proven to be extremely useful in many areas of quantum computation, for example in Shor's factoring algorithm. In this chapter we will discuss several ways to implement QPE. The exposition will follow the one from [9].

2.1 Basic setup

Every unitary operator U admits a complete set of orthonormal eigenvectors $|\psi\rangle$ that diagonalize it, with eigenvalues in the form of a complex phase:

$$U|\psi\rangle = e^{2\pi i\phi}|\psi\rangle, \qquad \phi \in [0,1[\tag{2.1.1}$$

The scope of QPE is precisely to extract ϕ with arbitrary precision.

A basic setup for QPE is summarized in Fig. 2.1, and it consists simply of two Hadamard gates, a controlled-U gate and a measurement device.

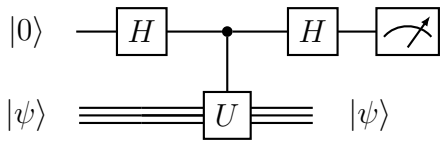

Figure 2.1: Circuital representation of a basic setup for QPE.

The circuit in Fig. 2.1 basically isolates the eigenvalue of U as a relative phase

between the states $|0\rangle$ and $|1\rangle$ of the first qubit, and in the first two steps it acts as such:

$$\begin{aligned} |0\rangle \otimes |\psi\rangle &\xmapsto{H\otimes I} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle \otimes |\psi\rangle + |1\rangle \otimes |\psi\rangle) \\ &\xmapsto{C-U} \frac{1}{\sqrt{2}}(|0\rangle \otimes |\psi\rangle + |1\rangle \otimes e^{2\pi i\phi} |\psi\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i\phi} |1\rangle) \otimes |\psi\rangle \end{aligned} \tag{2.1.2}$$

Now, focusing only on the first qubit, the intermediate state after the first two gates is simply

$$\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i\phi} |1\rangle) \tag{2.1.3}$$

In order to extract the information about ϕ one applies the other Hadamard gate on the first qubit, and obtains the following state:

$$\begin{aligned} \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i\phi} |1\rangle) &\xmapsto{H} \frac{1}{\sqrt{2}} \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} + e^{2\pi i\phi} \frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \\ &= \frac{1}{2} \left[(1 + e^{2\pi i\phi}) |0\rangle + (1 - e^{2\pi i\phi}) |1\rangle \right] \end{aligned} \tag{2.1.4}$$

Then the probabilities for measuring $|0\rangle$ and $|1\rangle$ are

$$P(0) = \frac{1 + \cos(2\pi\phi)}{2}, \quad P(1) = \frac{1 - \cos(2\pi\phi)}{2} \tag{2.1.5}$$

and repeating this process enough times, one can estimate a value for ϕ from the statistical distribution of the values for the first qubit.

This algorithm has two main flaws: firstly, it cannot distinguish between ϕ and $(1-\phi)$, and secondly the statistical fluctuations in the measurement process cannot give a very precise value of ϕ. In order to accomodate these issues, we will give a more refined QPE algorithm: *Kitaev's algorithm.*

2.2 Kitaev's algorithm

The setup for this algorithm is very similar to the one seen in the previous section, and is summarized in Fig. 2.2.

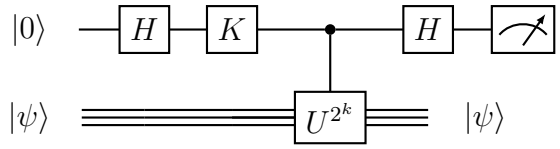

Figure 2.2: Circuital representation of Kitaev's algorithm setup.

In this setup the K gate can either represent an identity gate I or a S gate. Moreover, the controlled-U gate is replaced by a controlled-U^{2^k} gate, which is basically a controlled-U gate applied 2^k times, with $k = 0, 1, 2, ... n-1$.

As one can quickly see from the fact that $|\psi\rangle$ is an eigenvector of U, U^{2^k} also has $|\psi\rangle$ as eigenvector with the following eigenvalue:

$$U^{2^k} |\psi\rangle = e^{2\pi i\, 2^k \phi} |\psi\rangle \tag{2.2.1}$$

Therefore, as k ranges from 0 to $n-1$, this algorithm extracts the phases $\phi, 2\phi, ... 2^{n-1}\phi$ (modulo 1), which are useful to extract with increasing precision the first n binary digits in the fractional expansion of ϕ.

Similarly to before, when $K = I$ the final probabilities of getting $|0\rangle$ or $|1\rangle$ for the first qubit are

$$P(0) = \frac{1 + \cos\big(2\pi\, 2^k \phi\big)}{2}, \quad P(1) = \frac{1 - \cos\big(2\pi\, 2^k \phi\big)}{2} \tag{2.2.2}$$

If we instead used $K = S$, one can demonstrate in a similar fashion that the output probabilities of $|0\rangle$ or $|1\rangle$ on the first qubit are

$$P(0) = \frac{1 - \sin\big(2\pi\, 2^k \phi\big)}{2}, \quad P(1) = \frac{1 + \sin\big(2\pi\, 2^k \phi\big)}{2} \tag{2.2.3}$$

One can then extract the quantities $\cos\big(2\pi\, 2^k \phi\big)$ and $\sin\big(2\pi\, 2^k \phi\big)$ from a statistical analysis and use them to find $2^k\phi$ without ambiguities:

$$2^k \phi = \frac{1}{2\pi} \tan^{-1}\left(\frac{\sin\big(2\pi\, 2^k \phi\big)}{\cos(2\pi\, 2^k \phi)}\right) \tag{2.2.4}$$

As one can see, this algorithm requires a good amount of classical processing after the quantum elaboration results. In the next sections we will derive some results that move in the direction of creating algorithm without all this classical post-processing.

2.3 Quantum Fourier Transform

The *Quantum Fourier Transform* (QFT) is a unitary transformation of n qubits, from the *canonical basis* $|x\rangle$, with $x \in \{0,1\}^n$, to the so-called *Fourier basis*:

$$|x\rangle \overset{QFT}{\longmapsto} |\tilde{x}\rangle = \frac{1}{2^{n/2}} \sum_{k \in \{0,1\}^n} \omega_n^{xk} |k\rangle, \qquad \omega_n = e^{2\pi i/2^n} \tag{2.3.1}$$

This is the quantum version of the discrete Fourier transform. One can prove that it can be implement using $O(n^2)$ quantum gates, while the classical Fast Fourier Transform

algorithm requires $O(n\,2^n)$ steps [7], which amounts in an exponential speedup.

To see how to implement it, we first need to manipulate our expression for $|\tilde{x}\rangle$, expressing a n-digit binary number k in its binary notation:

$$k = k_1\,2^{n-1} + ... + k_n\,2^0 = \sum_{j=1}^{n} k_j\,2^{n-j} \tag{2.3.2}$$

$$|k\rangle = |k_1...k_n\rangle = \bigotimes_{j=1}^{n} |k_j\rangle \tag{2.3.3}$$

where we used the convention that k_1 is the most significant bit and k_n is the least significant bit ($k_j \in 0, 1$).

Now, for what regards $|\tilde{x}\rangle$ itself:

$$\begin{aligned}
|\tilde{x}\rangle &= \frac{1}{2^{n/2}} \sum_{k\in\{0,1\}^n} \omega_n^{xk}\,|k\rangle \\
&= \frac{1}{2^{n/2}} \sum_{k_1=0}^{1} ... \sum_{k_n=0}^{1} \omega_n^{x\sum_{j=1}^{n} k_j 2^{n-j}} \bigotimes_{l=1}^{n} |k_l\rangle \\
&= \frac{1}{2^{n/2}} \sum_{k_1=0}^{1} ... \sum_{k_n=0}^{1} \prod_{j=1}^{n} \omega_n^{xk_j 2^{n-j}} \bigotimes_{l=1}^{n} |k_l\rangle \\
&= \frac{1}{2^{n/2}} \sum_{k_1=0}^{1} ... \sum_{k_n=0}^{1} \bigotimes_{j=1}^{n} \left[\omega_n^{xk_j 2^{n-j}}\,|k_j\rangle\right] \\
&= \frac{1}{2^{n/2}} \bigotimes_{j=1}^{n} \sum_{k_j=0}^{1} \omega_n^{xk_j 2^{n-j}}\,|k_j\rangle \\
&= \frac{1}{2^{n/2}} \bigotimes_{j=1}^{n} \left[|0\rangle + \omega_n^{x\,2^{n-j}}\,|1\rangle\right]
\end{aligned} \tag{2.3.4}$$

Focusing now on the relative phase $\omega_n^{x\,2^{n-j}}$ we can express it as such:

$$\begin{aligned}
\omega_n^{x\,2^{n-j}} &= \exp\left(\frac{2\pi i}{2^n}\,x\,2^{n-j}\right) = \exp\left(2\pi i\,x\,2^{-j}\right) = \exp\left(2\pi i \sum_{l=1}^{n} x_l\,2^{n-l}\,2^{-j}\right) \\
&= \exp\left(2\pi i \sum_{l=1}^{n} x_l\,2^{n-j-l}\right) = \exp\left(2\pi i \sum_{l=j+1-n}^{j} x_{l+n-j}\,2^{-l}\right) \\
&= \prod_{l=j+1-n}^{j} \exp\left(2\pi i\,x_{l+n-j}\,2^{-l}\right) = \prod_{l=1}^{j} \exp\left(2\pi i\,x_{l+n-j}\,2^{-l}\right)
\end{aligned} \tag{2.3.5}$$

So finally, we can write $|\tilde{x}\rangle$ as:

$$|\tilde{x}\rangle = \frac{1}{2^{n/2}} \bigotimes_{j=1}^{n} \left[|0\rangle + \prod_{l=1}^{j} \exp\left(2\pi i \, x_{l+n-j} \, 2^{-l}\right) |1\rangle \right] \tag{2.3.6}$$

At first glance the equation (2.3.6) seems more complicated than the initial expression, but at a closer look one can see that it factorizes the action of each input qubit $|x_1\rangle, ... |x_n\rangle$ on each output qubit.

The circuital implementation of QFT is then represented in Fig. 2.3, which can be checked to implement precisely equation (2.3.6) with $O(n^2)$ gates as promised.

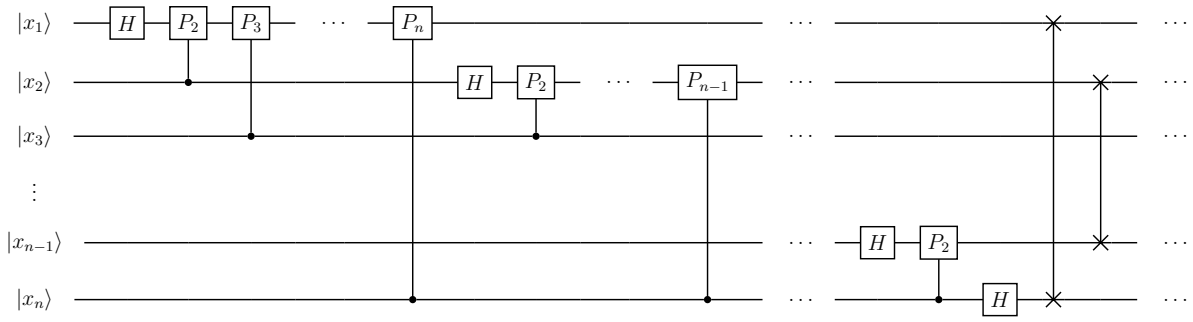

Figure 2.3: Circuital implementation of QFT.

In Fig. 2.3 the P_l gates represent a phase gate of the form:

$$P_l = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^l} \end{bmatrix} \tag{2.3.7}$$

and, at the end, all the qubits are swapped to match the adopted convention in equation (2.3.6). The whole circuit is summarized in Fig. 2.4.

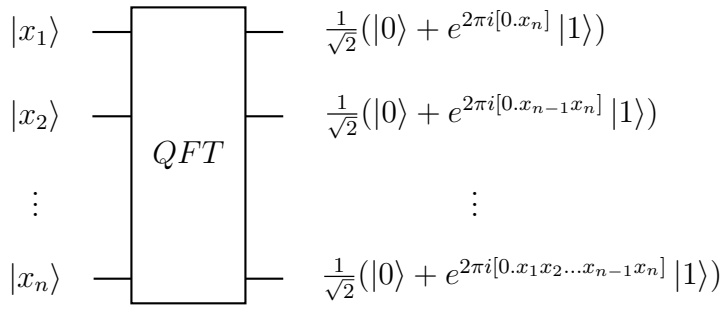

Figure 2.4: Circuital representation of QFT.

We are also interested in the inverse operation, which goes by the name of *Inverse Quantum Fourier Transform* (IQFT), which, by unitarity of the QFT, is the Hermitian conjugate of QFT:

$$|x\rangle \xmapsto{IQFT} \frac{1}{2^{n/2}} \sum_{k\in\{0,1\}^n} \omega_n^{-xk} |k\rangle \tag{2.3.8}$$

This transformation can be very easily implemented by using the same setup used for QFT, conjugating the phase gates P_l.

2.4 QPE via IQFT

We are now ready to see another algorithm for QPE which is based on IQFT. The setup is summarized in Fig. 2.5.

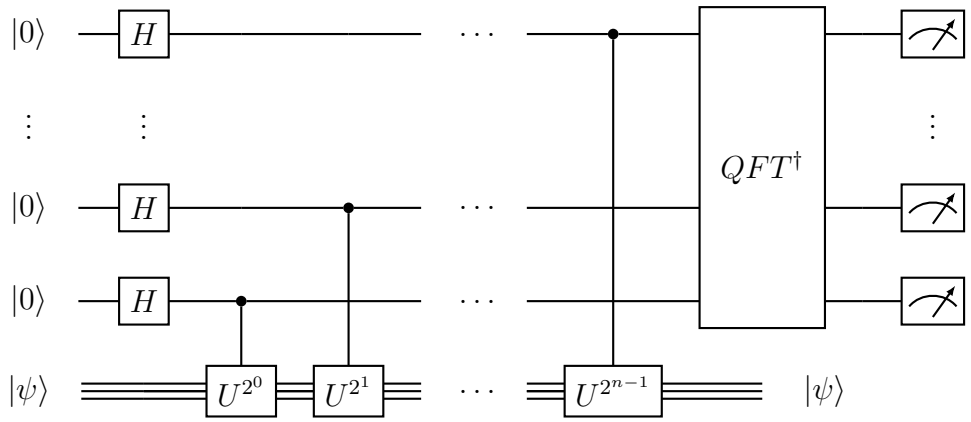

Figure 2.5: Circuital representation of QPE based on IQFT.

We can see that the first n qubits are manipulated by the Hadamard gates and the controlled-U^{2^k} gates as such:

$$\begin{aligned} |0\rangle^{\otimes n} &\xmapsto{H^{\otimes n}} \frac{1}{2^{n/2}}(|0\rangle + |1\rangle)^{\otimes n} \\ &\xmapsto{C-U} \frac{1}{2^{n/2}} \bigotimes_{j=1}^{n} (|0\rangle + e^{2\pi i\, 2^{n-j}\phi} |1\rangle) = \frac{1}{2^{n/2}} \sum_{k\in\{0,1\}^n} e^{2\pi i \phi k} |k\rangle \end{aligned} \tag{2.4.1}$$

The resulting state is very similar to a state of the Fourier basis, and in this sense the IQFT is optimal to recollect the information about the phase in the resulting state. In

fact, if we apply the IQFT to this state we obtain:

$$\frac{1}{2^{n/2}}\sum_{k\in\{0,1\}^n} e^{2\pi i\phi k}\,|k\rangle \xrightarrow{IQFT} \frac{1}{2^{n/2}}\sum_{k\in\{0,1\}^n} e^{2\pi i\phi k}\frac{1}{2^{n/2}}\sum_{x\in\{0,1\}^n} e^{-\frac{2\pi i}{2^n}kx}\,|x\rangle$$
$$= \frac{1}{2^n}\sum_{x,k\in\{0,1\}^n} e^{-\frac{2\pi i}{2^n}k(x-2^n\phi)}\,|x\rangle \tag{2.4.2}$$

Now, let a be the nearest integer to $2^n\phi$. Then we can express $2^n\phi$ as such:

$$2^n\phi = a + 2^n\delta, \qquad 0 \le |2^n\delta| \le \frac{1}{2} \tag{2.4.3}$$

Substituting this expression into (2.4.2) we obtain

$$\frac{1}{2^n}\sum_{x,k\in\{0,1\}^n} e^{-\frac{2\pi i}{2^n}k(x-2^n\phi)}\,|x\rangle = \frac{1}{2^n}\sum_{x,k\in\{0,1\}^n} e^{-\frac{2\pi i}{2^n}k(x-a)}e^{2\pi ik\delta}\,|x\rangle \tag{2.4.4}$$

If we now calculate the probability of obtaining $|a\rangle$ as the final state, we get:

$$P(a) = \left|\frac{1}{2^n}\sum_{x,k\in\{0,1\}^n} e^{-\frac{2\pi i}{2^n}k(x-a)}e^{2\pi ik\delta}\,\langle a|x\rangle\right|^2 = \left|\frac{1}{2^n}\sum_{x,k\in\{0,1\}^n} e^{-\frac{2\pi i}{2^n}k(x-a)}e^{2\pi ik\delta}\delta_{ax}\right|^2$$
$$= \left|\frac{1}{2^n}\sum_{k\in\{0,1\}^n} e^{2\pi ik\delta}\right|^2 = \begin{cases} 1 & \text{if } \delta = 0 \\ \dfrac{1}{2^{2n}}\left|\dfrac{1-e^{2\pi i\,2^n\delta}}{1-e^{2\pi i\delta}}\right|^2 & \text{if } \delta \neq 0 \end{cases}$$
$$= \begin{cases} 1 & \text{if } \delta = 0 \\ \dfrac{1}{2^{2n}}\dfrac{\sin^2(\pi\,2^n\delta)}{\sin^2(\pi\delta)} & \text{if } \delta \neq 0 \end{cases} \tag{2.4.5}$$

So, the final state is exactly $|a\rangle$ whenever $\delta = 0$. If $\delta \neq 0$ then our final state will not be $|a\rangle$ with certainty, but since $0 \le |2^n\delta| \le 1/2$, one can demonstrate that $P(a) \ge 4/\pi^2 \approx 40.5\%$, which is still the most probable state.

This algorithm, unlike the others we have seen, requires very little classical post-processing, since a, the best n-bit estimate of ϕ, is simply the most probable output of the measurement.

2.5 QPE via Inverse-AQFT

The main downside to the approach for QPE based on IQFT is the presence in the circuit of high precision phase gates, which are very difficult to realize in practice. The approach

for QPE presented in this section is therefore based on *Approximate Quantum Fourier Transform* (AQFT), which is basically implemented with the same circuit in Fig. 2.3, but keeping just the phase gates P_l up to a fixed degree m and removing all the phase gates of higher degree. The result is summarized in Fig. 2.6.

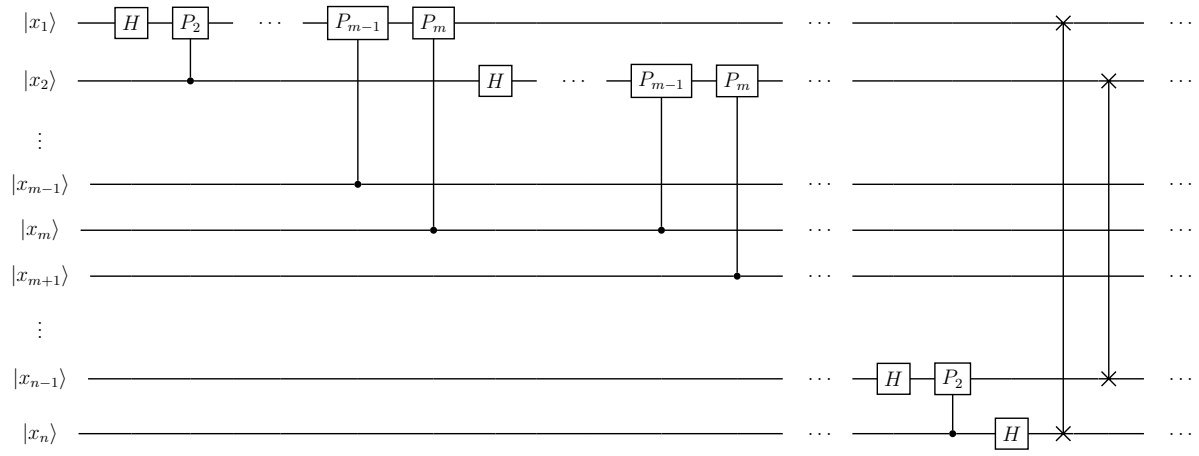

Figure 2.6: Circuital implementation of AQFT of order m.

The approach then is very similar to QPE based on IQFT: one simply replaces $QFT^\dagger$ with $AQFT^\dagger$ of the desired order in Fig. 2.5. What we are basically doing is trading off smaller success probability of getting the correct result at the end of the circuit with a smaller degree of the phase gates and a shorter circuit. The result is shown in Fig. 2.7.

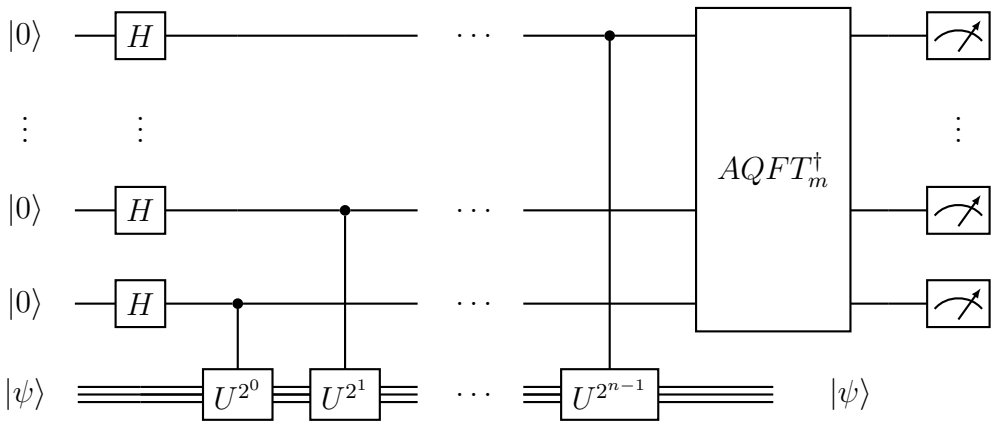

Figure 2.7: Circuital representation of QPE based on Inverse AQFT.

One can then demonstrate [10] that if we set $m = 3$ the probability of getting the right output for each single qubit is $P \geq \cos^2(\pi/8) \approx 85.4\%$, which guarantees good results by using only small degree phase gates.

Chapter 3

Implementation of QPE

In this chapter, we will discuss the implementation results for the various QPE algorithms presented in the previous chapter. The implementation has been done in Qiskit.

Qiskit [11] is an open-source software development kit that provides tools for creating quantum circuits and running them either on simulators on your local computer, or on real quantum devices on IBM Quantum Experience. Qiskit primarily uses Python as its programming language and can be easily accessed through Jupyter Notebook. In turn, IBM Quantum Experience [12] is IBM's cloud-based quantum computing service, which provides public and premium access to IBM's quantum processors in order to run quantum algorithms. IBM's quantum processors are physically made up of superconducting transmon qubits, which are a class of superconducting charge qubits designed to have reduced sensitivity to charge noise.

In this chapter the implementation has been done on both the Qiskit Aer Simulator and the IBM Quantum Experience 5-qubit quantum computing hardware platform "ibmq_manila".

In general, the approach taken consisted in fixing an arbitrary phase $\phi = 0.1010011$ and replacing the controlled-U^{2^k} gates with the single-qubit phase gates $P(2\pi\, 2^k\phi)$ on the control qubit, thus minimizing the number of necessary qubits. Then the algorithms try to extract this phase ϕ up to the 5^{th} digit (ϕ is intended to have more than 5 digits in order to simulate a more realistic eigenvalue of a generic unitary operator U).

One important remark is in order. The simulation results are obtained as projection of the simulated wave function on the computational basis, hence giving the probabilities of each measurement. In contrast, the quantum device gives only one instance of the computational basis after the measurement, so the circuit needs to be run several times to obtain statistical information. In each of these implementations the quantum routine has been run 1024 times.

3.1 Kitaev QPE

Kitaev's algorithm is characterized by requiring multiple single-qubit circuits, two for every order of approximation of ϕ.

Therefore, the relevant code that realizes the various quantum circuits can be implemented as follows:

```
from qiskit import QuantumCircuit

def kitaev(order, phase):
    qc_list = []

    for ii in range(order):
        qc1 = QuantumCircuit(1, 1)
        qc1.h(0)
        qc1.p(2*np.pi*(2**ii)*phase, 0)
        qc1.h(0)
        qc1.measure(0, 0)

        qc2 = QuantumCircuit(1, 1)
        qc2.h(0)
        qc2.s(0)
        qc2.p(2*np.pi*(2**ii)*phase, 0)
        qc2.h(0)
        qc2.measure(0, 0)

        qc_list.append(qc1)
        qc_list.append(qc2)

    return qc_list
```

Kitaev's algorithm also requires classical post-processing of the histograms of the results, which is summarized in the following code:

```
def process(histos):
    phases = []
    shots = histos[0]['0'] + histos[0]['1']

    for ii in range(len(histos)//2):
        cos = (histos[2*ii]['0'] - histos[2*ii]['1'])/shots
        sin = (histos[2*ii+1]['1'] - histos[2*ii+1]['0'])/shots
        phases.append((np.arctan2(sin, cos)/(2*np.pi))%1)

    return phases
```

The results obtained from implementing this algorithm are then summarized in Fig. 3.1, where the quantum device phases are obtained as the statistical mean of the runs of the program. One can see that the simulation results are in perfect agreement with the

theoretical predictions, while the quantum device results suffer from an inevitable noise which makes them less reliable.

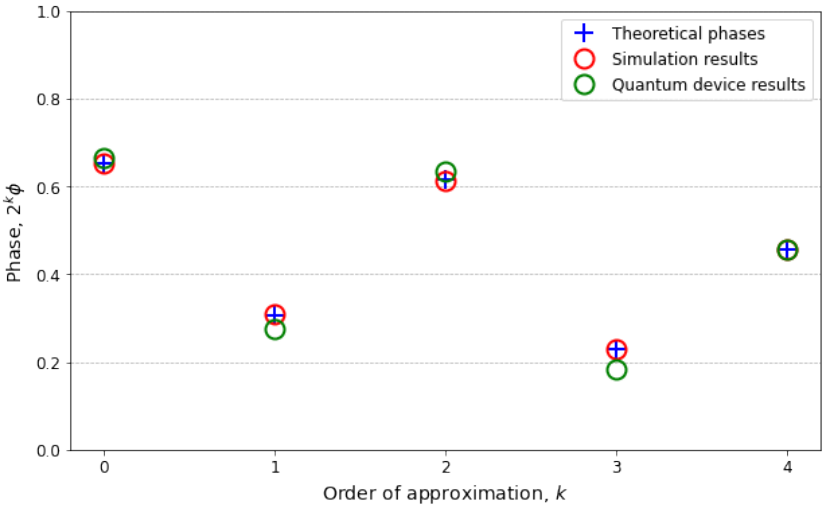

Figure 3.1: Plot of the first 5 orders of approximation of the phase ϕ, comparing the theoretical predictions to the simulation and the quantum device results. For $k = 4$ the simulation phase is hidden under the quantum device phase, as they coincide perfectly.

3.2 IQFT QPE

The implementation of the QPE algorithm based of IQFT has been done using the standard Qiskit QFT subcircuit, so the relevant code that realizes it is the following:

```
from qiskit import QuantumCircuit
from qiskit.circuit.library import QFT

def IQFT_QPE(order, phase):
    qc = QuantumCircuit(order, order)

    for ii in range(order):
        qc.h(ii)
        qc.p(2*np.pi*(2**ii)*phase, ii)
    qc.append(QFT(order, inverse=True), qc.qubits)
    qc.measure(qc.qubits, qc.clbits)

    return qc
```

This algorithm has been implemented varying the number of qubits (hence the order of approximation of ϕ) from 3 to 5, and the results are summarized in Fig. 3.2, Fig. 3.3 and Fig. 3.4.

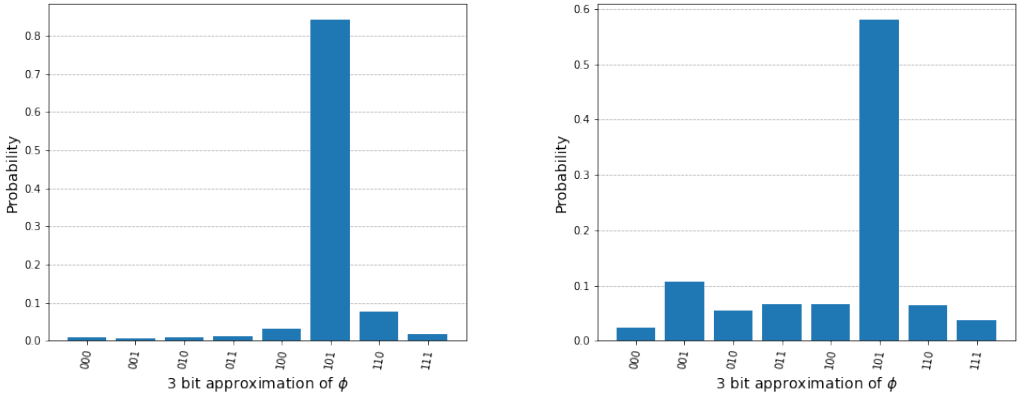

Figure 3.2: Histograms of the probabilities of each 3-bit outcome of the IQFT QPE. On the left the simulation results, and on the right the quantum device results.

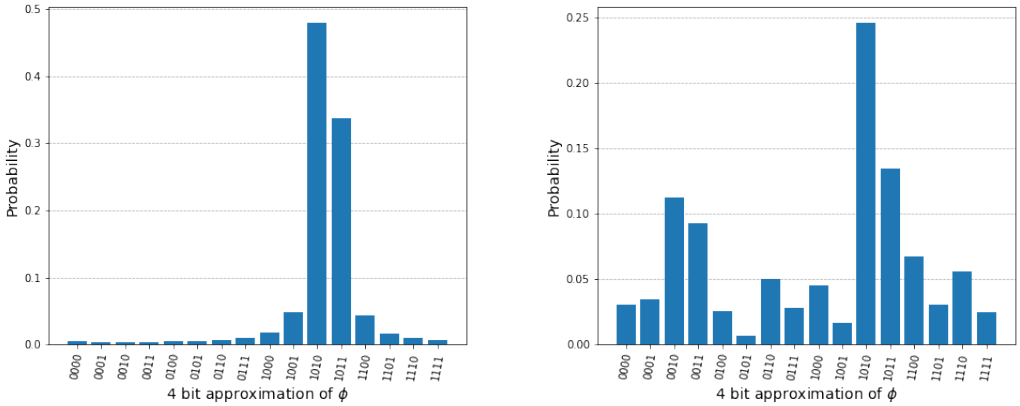

Figure 3.3: Histograms of the probabilities of each 4-bit outcome of the IQFT QPE. On the left the simulation results, and on the right the quantum device results.

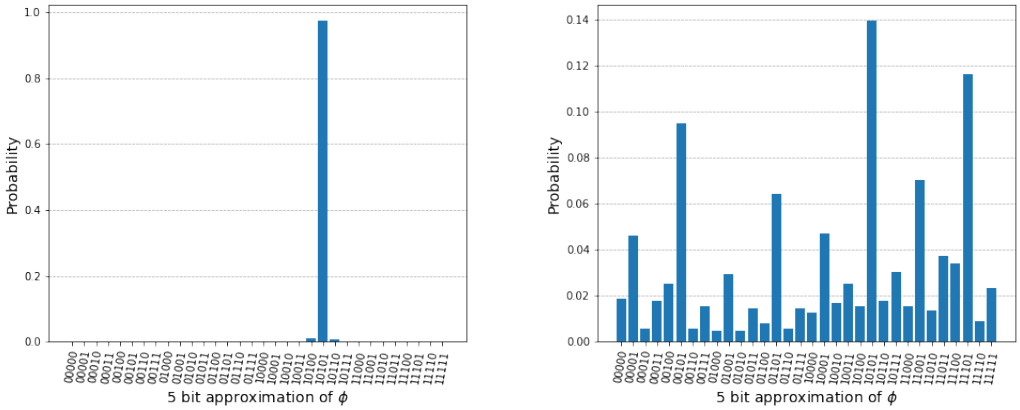

Figure 3.4: Histograms of the probabilities of each 5-bit outcome of the IQFT QPE. On the left the simulation results, and on the right the quantum device results.

As one can see, the simulated results coincide perfectly with our expectations: in each case the probabilities are centered around the best n-bit estimate of ϕ and rapidly decrease as the approximation of ϕ becomes less precise. If we had chosen a particular ϕ with fewer digits, the probabilities would have become more peaked around the true value of ϕ, and in the limit of ϕ having the same or fewer number of digits as the number of simulation qubits, we would have found the result ϕ with probability 1.

The real quantum device results are instead much more messy: one can still distinguish the most probable output from the distribution, but especially when we increase the number of qubits, the noise becomes more and more present. The probabilities hence are far more distributed and less peaked, thus there is a visible change in the y-scales of the histograms between the simulation and the real quantum device results.

3.3 Inverse-AQFT QPE

In order to implement this algorithm, the AQFT subcircuit was realized from scratch, and it was used in an analogous fashion to the IQFT case:

```
from qiskit import QuantumCircuit

def myAQFT(nqubits, degree, inverse):
    qc = QuantumCircuit(nqubits)

    for ii in reversed(range(nqubits)):
        qc.h(ii)
        for jj in reversed(range(max(ii-degree+1, 0), ii)):
            qc.cu1(((-1)**inverse)*np.pi/(2**(ii-jj)), jj, ii)

    return qc

def AQFT_QPE(order, degree, phase):
    qc = QuantumCircuit(order, order)

    for ii in range(order):
        qc.h(ii)
        qc.p(2*np.pi*(2**ii)*phase, ii)
    qc.append(myAQFT(order, degree, inverse=True), qc.qubits)
    qc.measure(qc.qubits, qc.clbits)

    return qc
```

This algorithm has been implemented with 4 and 5 qubits, with a degree of the AQFT of 3 (the case with 3 qubits is equivalent to the standard IQFT case), and the results are summarized in Fig. 3.5 and Fig. 3.6. Again, the quantum device histograms are obtained as the result of 1024 runs of the program.

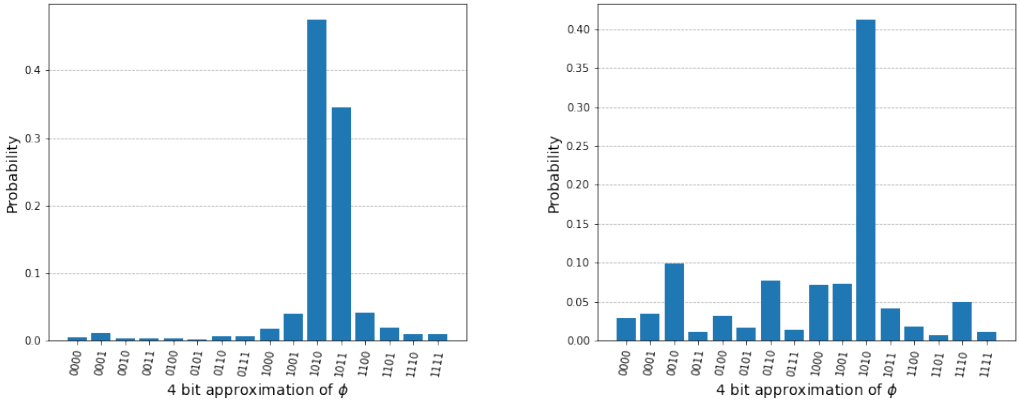

Figure 3.5: Histograms of the probabilities of each 4-bit outcome of the Inverse-AQFT QPE. On the left the simulation results, and on the right the quantum device results.

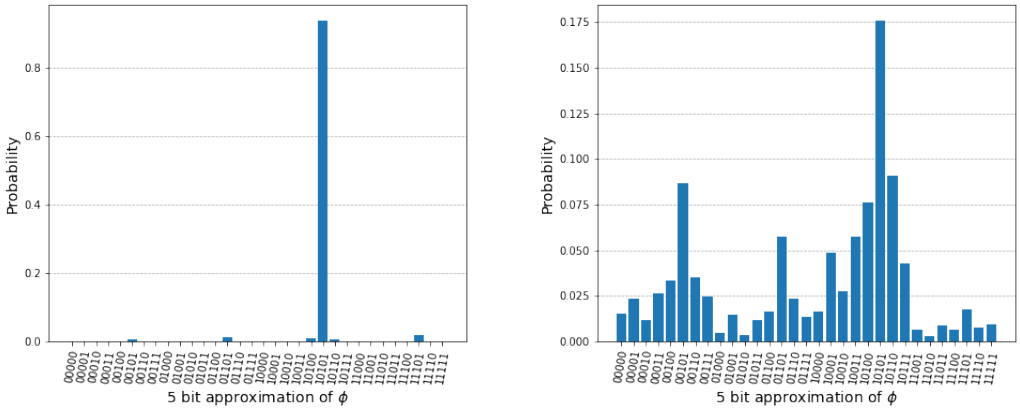

Figure 3.6: Histograms of the probabilities of each 5-bit outcome of the Inverse-AQFT QPE. On the left the simulation results, and on the right the quantum device results.

The same considerations we did in the IQFT case apply more or less also here. The only difference is that the simulation results present slight deviations from before, symptom of the fact that we used phase gates only up to degree 3. This has also the effect of reducing drastically the noise in the real quantum device, as compared to the results of the previous section.

Bibliography

[1] Paul Benioff. "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines". In: *Journal of Statistical Physics* 22.5 (May 1980), pp. 563–591. DOI: 10.1007/BF01011339.

[2] Richard P. Feynman. "Simulating Physics with Computers". In: *International Journal of Theoretical Physics* 21.6-7 (June 1982), pp. 467–488. DOI: 10.1007/BF02650179.

[3] D. Deutsch. "Quantum theory, the Church-Turing principle and the universal quantum computer". In: *Proceedings of the Royal Society of London Series A* 400.1818 (July 1985), pp. 97–117. DOI: 10.1098/rspa.1985.0070.

[4] P.W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700.

[5] Lov K. Grover. "A fast quantum mechanical algorithm for database search". In: *arXiv e-prints*, quant-ph/9605043 (May 1996), quant–ph/9605043. arXiv: quant-ph/9605043 [quant-ph].

[6] David P. Divincenzo. "The Physical Implementation of Quantum Computation". In: *Fortschritte der Physik* 48.9-11 (Jan. 2000), pp. 771–783. DOI: 10.1002/1521-3978(200009)48:9/11\textless{}771::AID-PROP771\textgreater{}3.0.CO;2-E. arXiv: quant-ph/0002077 [quant-ph].

[7] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010. DOI: 10.1017/CBO9780511976667.

[8] John Preskill. "Lecture notes for physics 229: Quantum information and computation". In: (1998).

[9] Hamed Mohammadbagherpoor et al. *An Improved Implementation Approach for Quantum Phase Estimation on Quantum Computers*. 2019. DOI: 10.48550/ARXIV.1910.11696.

[10] Hamed Ahmadi and Chen-Fu Chiang. "Quantum Phase Estimation with Arbitrary Constant-precision Phase Shift Operators". In: (2010). DOI: 10.48550/ARXIV.1012.4727.

[11] https://qiskit.org/.

[12] https://quantum-computing.ibm.com/.