VNIVERSITAT DE VALÈNCIA Facultat de Física

Trabajo de Fin de Grado en Física

Simulating physics with quantum computers

Septiembre 2020

Alumno: Alejandro Blázquez Salas

Tutor (1): Armando Pérez Cañellas
Tutor (2): Andreu Anglés Castillo

Contents

1 Introduction **2**

2 Quantum circuits **5**

2.1 Qubits 5

2.1.1 Multiple qubits 6

2.1.2 Entanglement 6

2.2 Quantum logic gates 7

2.2.1 Hadamard gate 7

2.2.2 NOT-gate 7

2.2.3 Controlled-U gate 7

2.2.4 CNOT-gate and Toffoli (CCNOT) gate 8

2.2.5 Measurement 8

2.3 Universal quantum gates 8

3 Quantum walks **9**

3.1 Discrete Time Quantum Walk (DTQW) on 1D 10

3.2 Physical imlementations 15

3.3 Quantum walk is universal for quantum computation 15

4 Dirac equation's simulation by DTQW **16**

5 DTQW's implementation on universal quantum computers **20**

5.1 IBM Quantum Experience and Qiskit 20

5.2 Quantum noise 21

5.3 Scalable networks for discrete quantum random walks 23

6 Conclusions **27**

Abstract

Quantum computation is a new paradigm for the design and execution of algorithms which are not based anymore on the boolean logic applied in current computation, but instead on the mathematical framework provided by quantum mechanics. Although ongoing research is still seeking for a real useful quantum computer, it is expected that this new model of computation may solve *efficiently* some of the current open problems in computational science and allow simulating physical systems driven by the laws of quantum mechanics. As a basis upon which constructing quantum computation, we will describe the quantum circuit model of computation. It is a universal language for describing sophisticated quantum computations. Moreover, we will study *quantum walks (QW)* as a generalization of *random walks (RW)* born in classical computation. The quantum walks represent an essential tool in algorithmic development and physical simulations. As this last aspect is our concern, we will show how a QW can simulate the behaviour of a fermionic field in one dimension so that its implementation on a real quantum computer allows simulating naturally this quantum system. Finally, we present IBM quantum experience and Qiskit, which allow us not only to write our own quantum circuits into a programming language but also run it on actual real quantum prototypes of IBM and analyze how the performance of the available computers is nowadays. Specifically, we will implement a QW on an IBM's computer, basing it on the study done by S. Fujiwara, H. Osaki, I. M. Buluta, and S. Hasegawa [1].

La computación cuántica se presenta como un nuevo paradigma en que la lógica booleana de los ordenadores actuales es reemplazada por las leyes de la mecánica cuántica, para la elaboración y ejecución de algoritmos. Su importancia radica en que se espera que este nuevo modelo de computación permita resolver *eficientemente* problemas abiertos actualmente en el campo de la ciencia computacional, así como la simulación de sistemas físicos, alcanzando la llamada supremacía cuántica. En este estudio se presenta una introducción al modelo de circuitos cuánticos, que representa un lenguaje universal para la descripción de computaciones cuánticas, y se profundiza sobre el concepto de los *quantum walks (QW)*. Estos son extensiones de los *random walks (RW)* introducidos en computación clásica, de gran importancia no sólo a nivel de elaboración de algoritmos, sino en la simulación de sistemas físicos. Con el propósito de demostrar la importancia de la computación cuántica en la simulación de sistemas físicos cuánticos, utilizaremos el modelo de QW para modelizar una simulación de un campo fermiónico a lo largo de una dimensión espacial, de forma que su implementación en un ordenador cuántico permitirá una simulación natural de este sistema. Finalmente, se presentarán IBM quantum experience y Qiskit, dos herramientas creadas por IBM que nos permitirán no sólo acceder a un lenguaje de programación basado en Pyton con el que escribir circuitos cuánticos, sino también tener acceso a los prototipos actuales de IBM y ejecutar allí nuestro propio circuito, que será una implementación de un QW basada en el estudio hecho por S. Fujiwara, H. Osaki, I. M. Buluta, and S. Hasegawa [1].

1 Introduction

Quantum mechanics (QM) represents the framework that arose from the necessity to explain observations that could not be reconciled with classical physics at the beginning of the 20th century so that it allowed for the discovery of new physical theories that acomplished this purpose. For example, quantum electrodynamics is a theory that describes accurately the interaction of atoms and light by using the rules that quantum mechanics provides. Likewise, *quantum computation is the study of the information processing tasks that can be accomplished using quantum mechanical systems* [2]. Hence, quantum computation is the use of quantum mechanical phenomena to perform any computation and, therefore, a quantum computer is a physical device which performs these quantum processes.

The early development of quantum computation was driven by a desire for a better understanding of the counterintuitive rules of quantum mehcanics. In the early 1980s, the study on whether it is possible to signal faster than light by using quantum effects, thus violating the rules of relativity led to the *no-clonig theorem*, which demonstrates that this is not possible since QM does not allow to construct a copy of a quantum state. Another historical example of the development of quantum computation arose from the interest of obtaining complete control over single quantum systems by methods like the *atom trap*, which allows to isolate the atom from the rest of the environment and prove or use different aspects of its quantum behavior. Nevertheless, the construction of quantum information processing systems for making large-scale operations a reality remains a great issue to work on.

Nonetheless, to give a historical summary of the development of quantum computation [2, 3], one should first start with a discipline, without which it would have not been possible: computer science, whose modern manifesation was announced by Turing (1936). He developed an abstract notion of what we would now call a programmable computer, which represents a theoretical model for computation known as the *Turing machine*. Although it is not a practical model for computation in real machines, it has had great importance in the theoretical development of computer science, due to its simplicity, and to the so-called *Church-Turing thesis* that asserts that *if an algorithm can be performed on any piece of hardware, then there is an equivalent algorithm for a Turing machine which performs the same task*. This sentence connects the physical concept of what class of algorithms can be performed in a real computer with the rigorous mathematical concept of a Turing machine. Moreover, a system that can simulate any Turing machine is said to be *Turing complete* or *computationally universal*, and then it is capable of performing any possible algorithm. Hence, the Turing machine model is in fact Turing complete [1].



Figure 1: Example of a Turing Machine, where the 3 possible symbols (0, 1 or blanck) and the head are shown.

A Turing machine consists of an infinitely-long tape divided into squares over which a head can be positioned. These squares act like memory units that can be in three states: blank, 0 or 1. The head can perform three basic operations: it can either read the symbol of the square which is below it, edit it or erase it. At the same time, the head is given the necessary instructions to move along the tape. This is all represented in Fig. (1), which schematizes a

[1] A universal Turing machine is a Turing machine that can simulate the behaviour of any other Turing machine.

Turing machine [4]. It is worth emphasising that despite its simplicity, a Turing machine can simulate any computer algorithm, no matter how complicated it is, just by using the described rules.

Not long after Turing's advance, John von Neumann was the one who developed a model that is computationally universal and can be applied to a real computer; in fact, almost every current computer is based on it.

On the other hand, hardware development came with the invention of the transistor in 1947 by John Bardeen, Walter Brattain and Will Shockley. The power of the hardware components has been improving at an increasing pace following Moore's law, which states that computer power will double for constant cost roughly once every two years. This law has held true through decades, although the current size of components are arriving at a point where quantum effects start to arise, therefore interfering in the functioning of electronic devices. This means that we will reach a point where it is no longer possible to improve the classical computation by hardware. However, quantum computation can make a difference, since it supposes a new paradigm of computation not tied to these limitations, as it is based on the idea of using quantum mechanics instead of classical physics to perform the calculations.

The very birth of quantum computation is usually found in the talk Richard Feynman gave at MIT in 1981 [5]. He pointed out the difficulties that are encountered in simulating quantum systems by using classical computers, since the number of variables the computer has to keep in memory increases exponentially with the size of the system being simulated. In order to solve this problem, he proposed to build a machine that works making use of quantum effects so it would efficiently simulate any quantum system. All this gave birth to the idea of a quantum computer. The Feynmann's idea of a universal quantum simulator that would allow imitating any quantum system went beyond this concept, and led to the development of the quantum computation that might overcome the computational power of classical computation. Since then, the harnessing of the quantum phenomena to improve computation started to catch attention, and quantum computation alongside with quantum information speeded up their development.

It is important to stress out that this quantum computer is also Turing complete so it can simulate any algorithm, including every algorithm previously developed for classical computers. Nonetheless, so is a classical computer, since it can simulate any algorithm run by a quantum computer. However, it turns out that this simulation cannot be achieved efficiently for genuine quantum algorithms. Thus, quantum computers may offer an essential speed advantage over classical computers, up to the point where some researches believe that no amount of progress in classical computation could overcome the gap between the power of a classical computer and the one of a quantum computer.

Therefore, the key word in order to point out the importance of quantum computing is *efficiency*. The general idea can be summed up by identifying an efficient algorithm as the one that runs in polynomial time in correlation to the size of the problem being solved. In contrast, an inefficient algorithm requires super-polynomial (typically exponential) time.

This notion allows us to introduce the *strong Church-Turing thesis: any algorithmic process can be simulated efficiently using a probabilistic Turing machine.*

A probabilistic Turing machine is a modification of the original concept that has access to a real random number generator, which allows to use non-deterministic algorithms that have been found to perform better and solve further problems.

This statement was another hint for quantum computation: if it has been possible to find a new model of computation which is more efficient than the deterministic Turing machine,

the next step is to examine whether it is possible to find even more efficient models to solve problems that are not efficiently soluble within Turing's model of computation, which led again to the idea of quantum computation. In 1984 David Deutsch [6] endeavoured to find an answer grounded on the physical laws, so that they provide a foundation for finding a more efficient computational model. Because the laws of nature are quantum mechanical, he considered computing devices based on this set of precepts, so quantum analogues of the machines defined earlier by Turing, led ultimately to the modern conception of a quantum computer and, therefore, described mathematically the ideas brought by Feynnman. Thereby, Deutsch enabled a challenge for the strong form of the Church - Turing thesis by opening up the question about whether a quantum computer can efficiently solve computational problems which have no efficient solution on a classical computer, even on a probabilistic Turing machine. In 1994 Shor did the first significant advancement. He demonstrated that the problem of finding the prime factors of an integer, and the so-called discrete logarithm problem, could be solved efficiently on a quantum computer. Both problems were and still are believed to have no efficient solution on classical computers. Hence, these results are significant hints that quantum computers are computationally more powerful than probabilistic Turing machines.

As Feynman suggested, one of the main applications of quantum computers will likely be to simulate quantum systems that a classical computer cannot adequately do. In fact, this seems, at the present moment, to be more plausible as a modest aim, rather than developing a real quantum computer which is powerful enough to achieve the universal computational power of classical computers. Nonetheless, there are more areas where quantum computation can imply a revolution such as cryptography. The main problem of developing applications for a quantum computer, apart from the fact of building it, is the difficulty of elaborating quantum algorithms. This difficulty appears mainly because human minds are programmed to deal with daily problems where the quantumness of reality vanishes, so our intuition comes from the classical world making it challenging to design useful quantum algorithms since the ideas one comes up with are rooted on classical precepts. To design practical quantum algorithms, we have to turn off this classical intuition and build a new one based on truly quantum effects. Besides, it is not enough for these algorithms to be useful, also they have to overcome the classical ones. By digging into this last problem, it should be considered a more general question, and ask what class of problems can be solved efficiently on a quantum computer that cannot be on a classical computer. The answers to these questions are not yet clear, so there is much work in this field to do.

In summary, quantum computation can make a remarkable difference on open computational problems and simulating physical systems. However, the notion that a Universal Quantum Computer is sufficient to simulate any physical system is nothing but a conjecture that may be possibly refuted. In the same way, as classical computers cannot efficiently simulate quantum phenomena, these quantum computers may also have trouble on simulating systems described by more fundamental theories as string theory. Nonetheless, this might also imply the existence of a more sturdy model of computation.

Throughout the present study, we first introduce the main ideas and the very basis of quantum computation in Sect. 2. We can go further and study a useful tool for computing and simulating physical systems, known as the quantum walk, in Sect. 3. It will not only allow us to explore the advantages of quantum computation but also to see how we can use them to do a real physical simulation of a Dirac equation in Sect. 4. Finally, in Sect. 5 we show how to program a real quantum computer by using the IBM Quantum Experience to run our own quantum program on the available prototypes of a quantum computer, and we explore the problems that one can encounter when building a real quantum computer.

2 Quantum circuits

Although we have seen that the Turing machine has supposed a theoretical model of great importance in the development of computational science and its quantum analogue for quantum computation, we can also talk about another approach, the *ciruit model* of computation. It is especially useful as a starting point for our study on quantum computers since Turing machines cannot be assumed as models for real computers as long as these are not endowed with an infinite amount of memory as it is assumed in a Turing machine. Hence, the circuit model represents an alternative model of computation that posses the same computational power and is more realistic.

The elements that compose a circuit are wires and gates. The wires are the structures that carry the information, while the gates are simple computational tasks that act on them. In classical computation, the wires are the elements of the register, the bits. A logic gate is a function $f : (0,1)^k \to (0,1)^l$ from some fixed number k of input bits to some fixed number l of output bits.

The counterpart for quantum computers are quantum circuits; they constitude a universal language for describing sophisticated quantum computations and work in the same way as described above for a classical computer, but instead of using bits for gathering the information, they use quantum bits (qubits) and quantum logic gates, which both will be deeply discussed below.

2.1 Qubits

On the one hand, classical information is stored in registers whose minimal piece of information is called a *bit*, which can be in two states, 0 or 1. On the other hand, in a quantum computer, the registers are made of qubits (quantum bits) that follow the rules of quantum mechanics, which means that they experience superposition, interference and entanglement. Superposition means that a qubit has no longer a defined unique value 0 or 1, but it exists in a state where both states manifest simultaneously. As any quantum system, its state is represented by a vector in a Hilbert space. To connect with classical notions, we will set as a basis of this space the following states $\{|1\rangle, |0\rangle\}$, which it is called the *computational basis* and represent the classical values for the bit. Hence the most general state for a register with only one qubit can be described as follows:

$$|\psi\rangle = e^{i\gamma}(\cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle), \tag{1}$$

which allows to represent each of these states as a point on a unit sphere in spherical coordenates θ and φ, since γ only represents an irrelevant global phase. This is the *Bloch Sphere* representation of the quantum state, as shown in Fig. (2). The vector, $\vec{\boldsymbol{M}}$, on the sphere is related to the quantum state following the next relation:

$$\vec{\boldsymbol{M}} = (M_x, M_y, M_z) \quad M_i = \langle\psi|\sigma_i|\psi\rangle, \tag{2}$$

where σ_i are the Pauli matrixes.

Although the above description of a qubit is purely mathematical, possible physical realizations have been already suggested and studied, i.e. the two polarizations of a photon, the alignment of nuclear spin in a uniform magnetic field or two electronic levels in an atom.

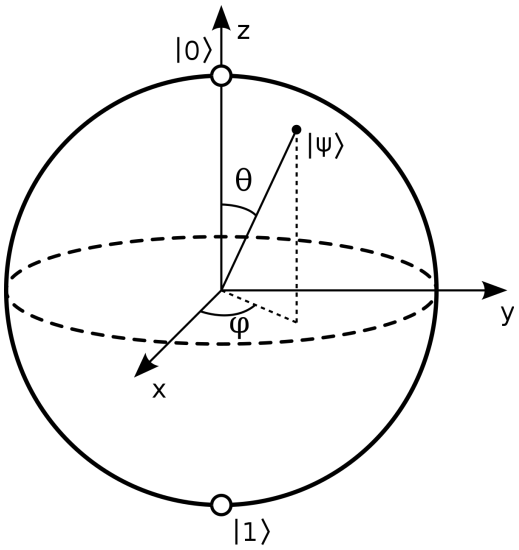


Figure 2: Representation of a qubit state on the Bloch sphere. The state $|\psi\rangle$ is entirely determined by the two angles, θ and φ.

At first sight, it may seem that a qubit can store an infinite amount of information since the surface of the Bloch sphere is characterized by two real-valued continuous parameters. However, to extract the information we have to measure the qubit usually with respect to the computational basis $\{|1\rangle, |0\rangle\}$. Hence, according to the rules of quantum mechanics one can get either the value 0 or 1, as in the classical case. Nonetheless, it differs from the classical case because the result of the measurement is not determined a priori; instead, one has a certain probability of obtaining one of the two values depending on the state of the qubit, following the quantum mechanical rules. This all means that one cannot extract more information from a qubit than from a classical bit.

Furthermore, measurement alters the state of a qubit, collapsing it from its superposition of $|0\rangle$ and $|1\rangle$ states to the corresponding state of the measurement result.

2.1.1 Multiple qubits

A quantum register composed of multiple qubits is described by a quantum state belonging to the Hilbert space constructed by the tensor product of each individual Hilbert space for each qubit, which implies a state space of 2^n dimensions, for a system of n qubits.

2.1.2 Entanglement

When describing a quantum register of multiple qubits, it is obviously possible to find it in a superposition state. For some states, it is not possible to be decomposed in a product of the states of its constituents, which also implies that the qubits are intrinsically correlated since one cannot define them separately; this is called entanglement. Even though the state of the system is well defined, neither of the qubits are in a well-defined state separately. This will imply that the measurements on one qubit will change the whole system state as a result of the wave function collapse. Such a measurement will affect the entangled system as a whole, affecting the possible measurements of the rest of the qubits of the register instantaneously. This is again in contrast to classical systems, whose states can always be broken down into the individual states of their components.

2.2 Quantum logic gates

In quantum computation and especially in the quantum circuit model we are working on, the information stored in the quantum register can be manipulated by quantum logic gates or simply quantum gates (QG), analogously to the classical ones. As this model is set into the framework of quantum mechanics, these quantum gates must be unitary linear operators acting on the Hilbert space of the register: $\mathbb{C}^{2^n} \to \mathbb{C}^{2^n}$, where n is the number of qubits in it. Hence, they transform an input state into another output state, transforming the initial information in order to accomplish the calculations. We inspect the most relevant gates in the following sections. The matrix notation for the qubits and quantum gates used are based upon the computational basis, then we have chosen $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

2.2.1 Hadamard gate

The Hadamard gate transforms the states of the computational basis into states over which measurement will have the same probability to become 0 or 1. That is to say:

$|0\rangle$ —[H]— $\frac{|0\rangle+|1\rangle}{\sqrt{2}} \equiv |+\rangle$

$|1\rangle$ —[H]— $\frac{|0\rangle-|1\rangle}{\sqrt{2}} \equiv |-\rangle$

The matrix representation is:

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

2.2.2 NOT-gate

This gate is the quantum equivalent to the NOT-gate in classical computation.

$|0\rangle$ —⊕— $|1\rangle$

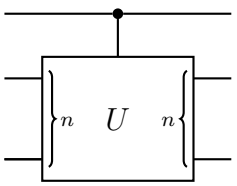
$|1\rangle$ —⊕— $|0\rangle$

The matrix representation is:

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

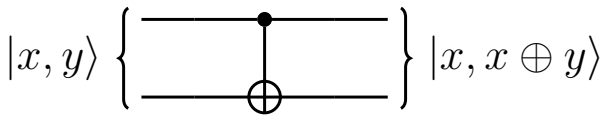
2.2.3 Controlled-U gate

The previous gates act on a single qubit, now a controlled-U gate is an operation on one *control* qubit and additional n qubits. The *control* determines whether or not the U operator acts over the rest of the n qubits. If the *control* is 1 (0) U is (not) applied. These gates are responsible for the creation of entanglement between qubits.



2.2.4 CNOT-gate and Toffoli (CCNOT) gate

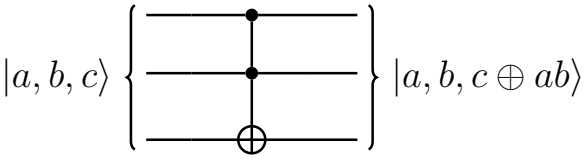
The controlled-NOT gate (CNOT) is an essential example of a controlled-U gate that acts on two qubits, performing the NOT operation on the second qubit when the *control* is $|1\rangle$.[2]



The matrix representation is:

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

The Toffoli gate is the generalization for two *control* qubits.

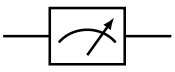


The matrix representation is:

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

2.2.5 Measurement

An special operation is the measurement over a qubit. This operation is not unitary, and converts an intial qubit $a|0\rangle + b|1\rangle$ into a classic bit whose value will be 0, with probability $|a|^2$ or 1 with probability $|b|^2$. The final state of the qubit will be $|0\rangle$ or $|1\rangle$, respectively. Due to the collapse of the wave function of the qubit's state when the measurement is applied. After the measurement the information of the qubits is also dumped into a classical register, where the result of the measurement is stored.



2.3 Universal quantum gates

A set of gates is said to be universal if any unitary operation may be approximated to arbitrary accuracy by a quantum circuit involving only those gates.

One of these sets is conformed by the set of all gates acting on a single qubit, and CNOT gates so that any arbitrary unitary operator (a quantum gate) can be expressed by using only them. These kind of gates are a core element of quantum computation. Their importance falls on finding families of universal quantum gates that can be appropriately implemented in real computers and perform the operations efficiently. Hence, creating a universal quantum computer amounts to build a physical device capable of implementing one set of these universal quantum gates.

[2]The $\oplus$ symbol represents a modular 2 sum.

3 Quantum walks

One of the reasons for the birth of quantum computation comes from the need of simulating efficiently quantum systems, since classical computers fail at it. The main difficulty classical computation finds on this task is because describing a quantum system generally grows exponentially with the number of complex numbers needed (as c^n, where n is the number of components of the system, and c a constant depending on the concrete system), rather than linearly as occurs in classical ones. By contrast, in a quantum computer one recovers the linearity when simulating quantum systems due to the quantum nature of qubits (it grows as kn, where k is another characteristic constant of the system).

Nevertheless, although the number of qubits needed is fewer for the simulation, kn, when measured these qubits collapse into a defined state, so the c^n bits of hidden information in the wavefunction encoded on the qubits is not entirely accessible. Thus, a crucial objective is the development of systematic means by which desired answers can be efficiently extracted from these simulations.

The study of these quantum simulations is not only of great importance in fields where quantum phenomena take place, but can also help the development of new algorithms for quantum computers since it has been already observed that some of the quantum algorithms already developed can be viewed as quantum simulations. However, we are not getting deep into the details since it exceeds the objectives of this study.

In this section, we will explore an example of quantum simulation. We will introduce the notion of a quantum walk (QW) as an extension of random walks (RW) developed in the framework of classical computation, and then it will be applied for simulating the dynamics of the Dirac equation in Sect. 4.

The notion of the quantum (random) walk (QW) was introduced by Aharonov in [7] in 1992 as a counterpart for the classical random walk. *A random walk is a mathematical object, we call the particle, that describes a path in which each step, within a mathematical space, is decided by a random process.* Hence the RW is defined in terms of probabilities for the particle to take any possible step allowed in the space. The generalization to the QW comes by describing it in terms of probability amplitudes instead, following the framework of quantum mechanics. In a more general definition, *a quantum walk is a process whose time evolution obeys the laws of quantum mechanics and is constrained by graph locality* [8]. As a first approach, one could think that a quantum walk might be constructed by a walker that follows the naive superposition of all possible random walks, instead of following just one particular sequence of steps. Hence at each step, the walker would be in a superposition of states along with the graph that would follow the same distribution one finds after many trials with its correspondent classical random walk. Nevertheless, this construction cannot be quantum since it is not reversible. The solution is to make the coin, which is the responsible to decide the next step of the walker in a random fashion, quantum too.

The process for the selection of the path the particle will follow on the next step is incorporated into the theory by correlating each possible step to another degree of freedom (usually called spin) apart from the one associated to the position on the given space. Instead of tossing a coin, we will give the spin a unitary twist. Hence this spin plays the role of a *quantum coin.* This correlation due to the superposition of the different possible paths can manifest a variety of novel features via quantum coherence beyond the classical random walk, as we will see in more detail later on.

3.1 Discrete Time Quantum Walk (DTQW) on 1D

For simplicity we will be exploring the simplest QW, which will be limited on a 1D discrete graph following the notation introduced in [9]. The vertices of this graph (also called nodes) are countable, so each of them is labeled by an integer p, and is two-regular, which means that each vertex is surrounded by two neighbours, $p-1$ and $p+1$. Depending on the boundary condition, it can be pictured as a line with an infinite number of vertices, so that the walker (the particle) never reaches the boundaries, or also as a circle with a periodic boundary condition. In order to introduce all the notations to describe such a system, we will consider the first case for simplicity, although the extension to the circle is trivial; one only has to take into account the periodicity on the graph and set the evolution rules in the boundaries. To have a better picture of the problem you can imagine each vertex as a position on a line and will associate the direction of growing p's to the movement of the particle to the right.

As the walker is considered to be described by quantum mechanics, its position state belongs to a Hilbert space $\mathcal{H}_{position}$ which is spanned by the basis $\{|p\rangle\}_{p\in\mathbb{Z}}$, where each vector represents the state where the particle is perfectly located on the p-vertex of the graph. In addition, we now need to introduce another degree of freedom to describe the 'coin space' (also called spin) described by another Hilbert space, $\mathcal{H}_{spin}$ which is the analogus for the probability of moving through any direction of the graph on the RW. For our QW in 1D, it will be required a minimum space of dimension two, so that we can stablish a base $\{|\uparrow\rangle, |\downarrow\rangle\}$ that we call up-down basis. Hence, the state of the walker belongs to the tensor product space $\mathcal{H} = \mathcal{H}_{spin} \otimes \mathcal{H}_{position}$, spanned by the basis $\{|p\rangle \otimes |s\rangle\}_{p\in\mathbb{Z}, s\in\{\uparrow,\downarrow\}}$. Furthermore, the time evolution of the walker will be set to be also discrete, so this evolution will also be labeled by another discrete time step $n \in \mathbb{Z}$. This model is known as the Discrete Time Quantum Walk (DTQW) on a 1D graph.

Thereby, the general state of the walker can be written as:

$$|\psi, n\rangle = \sum_{s\in\{\uparrow,\downarrow\}, p\in\mathbb{Z}} \psi^s_{n,p} |s\rangle \otimes |p\rangle = \sum_{p\in\mathbb{Z}} (\psi^{\uparrow}_{n,p} |\uparrow\rangle + \psi^{\downarrow}_{n,p} |\downarrow\rangle) \otimes |p\rangle , \tag{3}$$

the probability for the walker to be located at time n on the position p with spin s is $\left|\psi^s_{n,p}\right|^2$.

It just remains to determine the evolution of the walker for each step through the graph. It is governed by a one-step evolution operator W, also called walk operator, which is the composition of two operators S and C.

$$|\psi, n+1\rangle = W |\psi, n\rangle ; \qquad W = S(C \otimes \mathbb{I}_{position}), \tag{4}$$

where S is the shift operator that will act in such a way that a walker in the upper spin state $|\uparrow\rangle \otimes |p\rangle$ moves to the right and the lower one, $|\downarrow\rangle \otimes |p\rangle$, moves to the left (by an arbitrary convention):

$$S = |\uparrow\rangle \langle\uparrow| \otimes \sum_i |i+1\rangle \langle i| + |\downarrow\rangle \langle\downarrow| \otimes \sum_i |i-1\rangle \langle i| . \tag{5}$$

It is crucial to notice that this operator entangles spin and position; thus it cannot be factorized as an operator acting on the spin space and another one acting on the position space. C is an operator acting only on the spin space which represents a rotation of the walker spin and is called the coin operator since it is the operator that twists the spin in analogy to the coin

toss in the RW. Because this spin space is analogous to the real spin-1/2 space, we already know that the more general unitary operation over this space belongs to the set U(2), which is a Lie group generated by the spin-1/2 operator $\vec{\boldsymbol{S}} = \frac{\hbar}{2}\vec{\boldsymbol{\sigma}}$, represented by any 2 x 2 unitary matrix (which is required to preserve the total occupation probability of the walker), and can be expressed with the Euler angles, so that in the up-down basis is written as follows:

$$C = e^{-i\theta\frac{\vec{\boldsymbol{n}}\vec{\boldsymbol{S}}}{\hbar}} = U^{Euler}(\alpha,\theta,\varepsilon,\zeta) = e^{i\alpha}\begin{pmatrix} e^{i\varepsilon}\cos\theta & e^{i\zeta}\sin\theta \\ -e^{-i\zeta}\sin\theta & e^{-i\varepsilon}\cos\theta \end{pmatrix}, \tag{6}$$

where $\vec{\boldsymbol{n}}$ is an unitary vector on the real numbers, and $(\alpha,\theta,\varepsilon,\zeta)$ the Euler's angles. Consequently, the coin operator is the one which is free to be set, so according to its selection, we will have different behaviours for the walker.

The evolution rule for the walker after n steps is driven by the following equation:

$$|\psi, n\rangle = W^n\,|\psi, 0\rangle\,. \tag{7}$$

Moreover, the total probability for the walker to be measured at p (regardless of the spin) is $\left|\psi^{\uparrow}_{n,p}\right|^2 + \left|\psi^{\downarrow}_{n,p}\right|^2$, giving the probability distribution of the quantum walker.

The main reason for the development of a quantum walk is that its quantumness provides it with a singular behaviour that may represent an advantage, not only in describing quantum systems but in its application on several algorithmic problems so that its performance is faster than when the classical one is used. Hence, for a better understanding of these new features we will be comparing both models.

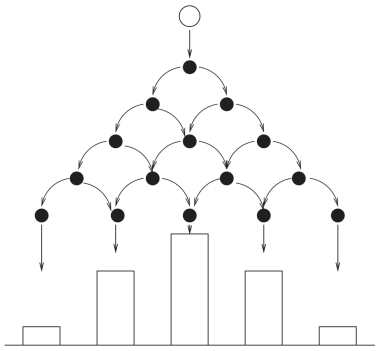


Figure 3: Illustration of the evolution of a classical random walk which is equally likely to fall in both directions at each iteration.

The description of the classical random walk, illustrated in Fig. (3), lays in a walker also placed on a discrete graph with vertices labelled by the integer p. Each of them has two neighbours, so that, at every step, the walker has a probability g to move to the right, and q to move to the left. The goal is to obtain the probability distribution for the walker to be placed on a certain position p after n steps, given that the walker can only take one step per iteration. Thereby we will be able to compare it to the probability distribution given by the probability amplitude of the QW.

In order to achieve this classical distribution, let us consider that, after n steps, the walker will have taken n_l steps to the left and n_r to the right, keeping true $n = n_r + n_l$, and that the position of the walker, considering that started at $p = 0$, would be $p = n_r - n_l$. We have to take into account the number of ways by which it is possible to get to the position

p and the probability of moving to any of both directions. Given n and p, both n_r, n_l are completely determined ny $n_r = \frac{p+n}{2}, n_l = -\frac{p-n}{2}$. Thus, counting the number of ways by which it is possible to get to that position in n steps, is analogous to counting the number of ways it is possible to put n_r indistinguishable objects in n distinguishable boxes that can only hold one of those objects. The objects would correspond to the right steps and the boxes to the iteration. This situation is described by the following binomial number $\binom{n}{n_r}$. If we also take into account the probability for the walker to have taken n_l (n_r) steps, given by q^{n_l} (g^{n_r}), the probability amplitude will be:

$$W(p,n) = \begin{cases} \binom{n}{\frac{p+n}{2}} g^{\frac{p+n}{2}} q^{-\frac{p-n}{2}}; & \text{with } p+n \text{ an even number} \\ \\ 0; & \text{with } p+n \text{ an odd number and} |p| > n, \end{cases} \tag{8}$$

the zero probability appears because the walker moves one step each iteration. The above can be rewritten as:

$$W(p,n) = \begin{cases} \frac{2^n n!}{(n+p)!!(n-p)!!} g^{\frac{n+p}{2}} q^{\frac{n-p}{2}}; & \text{with } p+n \text{ an even number} \\ \\ 0; & \text{with } p+n \text{ an odd number and } |p| > n. \end{cases} \tag{9}$$

We have seen that there are an infinite amount of diferent QW and RW, depending on the coin operator and the probabilities of the directions, respectively. In order to compare them, both models will be set to be unbiased. This means that they will treat both directions equally, in the following sense. For the RW both probabilities for the walk to go to the right or left are equal, $g = q = \frac{1}{2}$. The case of the QW is more sofisticated. We have to use balanced unitary operators as the coin, as it is the case for the Hadamard gate, which will be the one used in this study. This means that if a walker is on an state with a completly well-defined position and spin, [3] the measurement after one iteration of the QW (the coin has been applied once) follows the same classical distrubution as in the classical walker after one iteration. This election is not an arbitrary one, as we will see later on. It is chosen so that we can reproduce the classical limit from the quantum one.

To sum up, for making the comparison between classical and quantum models we have chosen the following conditions for the QW: i) The coin operator used is a Hadamard gate ii) the initial state of the walker is: $|\psi, 0\rangle = \frac{1}{\sqrt{2}}(|\uparrow\rangle + i\,|\downarrow\rangle) \otimes |0\rangle$. This condition will allow us to obtain a symmetric distribution since we have an equal 'amount' of up and down components. On the other hand, the RW will have an equal probability to move in both directions.

It is easy to see that the Hadamard gate is balanced:

$$|\uparrow\rangle \otimes |0\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle \xrightarrow{S} \frac{1}{\sqrt{2}}(|\uparrow\rangle \otimes |1\rangle + |\downarrow\rangle \otimes |-1\rangle) \tag{10}$$

so by measuring the position on the resulting state we get an equal probability of $\frac{1}{2}$ to be at $x = 1, -1$.

In Figs. (4-5), it will be shown both the classical and the quantum walker evolution after 30 iterations. Both models have been implemented in algorithms for classical computation to obtain these results; the implementation of the QW on a quantum computer will be done in Sect. 5.

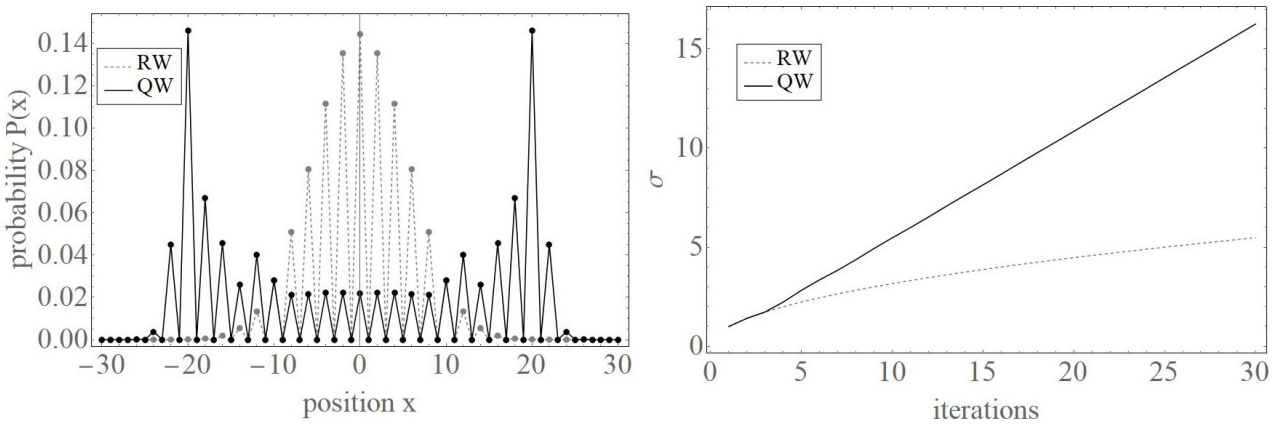


Figure 4: *Left:* Comparision of quantum and classical walks on a line, after 30 iterations. The coin used for the QW is the Hadamard gate, and the intial state is set to be $|\psi, 0\rangle = \frac{1}{\sqrt{2}}(|\uparrow\rangle + i\,|\downarrow\rangle) \otimes |0\rangle$. The RW is balanced ($g = q = \frac{1}{2}$). *Right:* Comparison of the standard deviation as each of the walkers evolve up to 30 iterations.

In the left plot of Fig. (4), one can see that the behaviour of the QW differs significantly from the classical's probability distribution. The RW approaches a Gaussian distribution with mean 0 and variance $\sigma^2 \sim n$, so that the spreading from the origin scales as $\sigma \sim n^{\frac{1}{2}}$ with the number n of steps, as can be seen in the right plot of Fig. (4). On the other hand, the QW's distribution does not follow a Gaussian distribution. In Fig. (4), it is shown that the spreading from the starting point scales as $\sigma \sim n$, which implies that the variance scales with $\sigma^2 \sim n^2$. The quantum walker spreads quadratically faster than the classical one.

As the RW follows a Gaussian distribution, the probability is peaked around the origin, and drops rapidly within several standard deviations σ away from the starting point. By contrast, the quantum walker spreads almost uniformly around the origin up to approximately half of σ. Then the probability grows up as one moves away from the center in a very intricated pattern. In Fig. (5), we can see the evolution of both classical and quantum walkers, so we can clearly distinguish the different patterns they follow throughout time. We can see that they only match after the first step, since we have imposed for the quantum walker to be balanced, as we have explained.

The next step is to analyze the fundamental features that the quantumness adds to the walker, thus changing its behaviour radically. The result for the RW shown in Fig. (4) is achieved by using Eq. (9), but we can obtain the same result from the QW. This happens if one measures both the position and the spin of the walker after each iteration. This makes the state of the walker collapse into a defined state, by removing any quantum superposition, so that in the next iteration it will have the same probability of moving to left or right as we have verified in Eq. (10). Hence, this reproduces the behaviour of the clasical walker by obtaining the Gaussian distribution. Instead, if no measurement is done between iterations, we get the QW's distribution since what it is happening is that the quantum correlations between the different positions are not being broken by the collapse of the wavefunction of the walker. At each iteration, they are interfering between them so that we get the final intricate pattern typical of the quantum world. This fact gives us anoter way to define a quantum walk as *quantum dynamics that reduces to a classical random walk when completely decohered* [10].

[3]It means that the state of the walker is formed by a unique vector of the position-spin bais as $|\uparrow\rangle \otimes |p\rangle$.

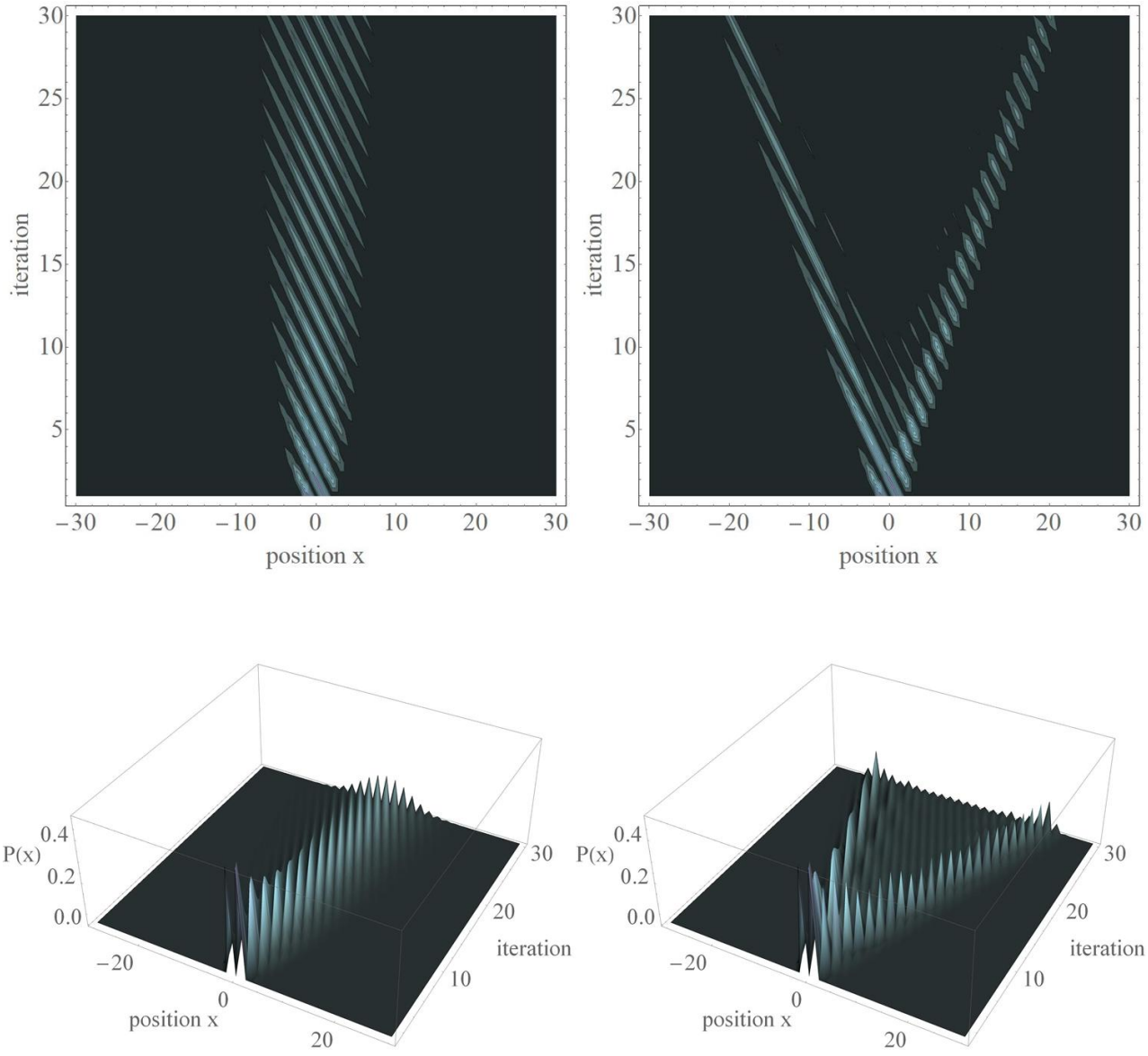


Figure 5: *Right column:* evolution of the quantum walker's distribution probability up to 30 iterations, using a Hadamard coin and a symmetric initial state $\frac{1}{2}(|\uparrow\rangle + i\,|\downarrow\rangle)\otimes|0\rangle$. *Left column:* evolution of the balanced classical walker's distribution probability up to 30 iterations.

This shows that introducing the RW into the world of quantum mechanics adds new features to this well-known and useful tool in theoretical computer science [4]. The algorithms that are based on the RW depend on the properties they exhibit (as the variance of the walker we have studied), achieved by changing the graph they move on or the different probabilities of the possible paths on it. Hence, the QW must be seen, not only as an instrument for physical simulation (as we are showing in the following section) but also as a new useful tool for this kind of algorithms with exotic new features for the walker (of which one takes advantage when implemented on a qunatum computer) that can be applied to solving many computational open problems. Of course, the result we have obtained for the quantum walker spreading

[4]For example, the RW provides a general paradigm for sampling and exploring an exponentially large set of combinatorial structures

quadratically faster on a line does not mean that we have found a new algorithm that would show the supremacy of the quantum version of a RW, but it was the first result that led Shenvi [11] to find that a quantum walk could search an unsorted database with a quadratic speed up.

Besides, as we have seen, the quantum circuit model is universal; this means that it can reproduce all classical algorithms. One might think that this is not possible since all quantum circuits are reversible, because of the unitarity of the operators representing gates, whereas a classical circuit is not. Bennett [12] showed that this is not the case, since any classical circuit can be replaced by another one, which is completely reversible. This is an essential result and may allow to improve any algorithm which uses a classical random walk by turning it into a quantum circuit (after making it reversible) and then changing the classical coin-flip by a quantum coin-flip that might exhibit these advantageous features, improving the performance of the algorithm.

3.2 Physical imlementations

Once one has understood the importance of quantum walks, the next stage is to figure out how they could be implemented efficiently on a real physical device. Of course, a universal quantum computer would be capable of it, but it is still far achivable by nowadays standards. Nevertheless, quantum walks properties may be implemented in certain physical systems that do not constitute a full quantum computer, but could be used to study them or to solve some algorithmic problems based on them.

Several options have been studied to implement a quantum random walk making use of the structure of the graph or the character of the coin. The key is to encode both the position space and the coin space in a suitable discrete degree of freedom that does not have to be neither real positions nor spins. As an example, we can take the ion-trap based implementation suggested by Milburn [13]. An ion is confined in a radio-frequency ion trap, where the positions of the walker are encoded into the discrete motional states of the ion in the trap. The spin is encoded in the internal electronic states of the ion. The last piece needed is to realise the shift and coin operator, which can be made by Raman beam pulses. The idea is that this can be applied several times to reproduce a concrete number of steps of the walker and then to measure the position of the walker. Nonetheless, due to quantum noise and decoherence, current technology only allows performing a few iterations, although these studies may suppose a protocol for future ion trap quantum computers.

3.3 Quantum walk is universal for quantum computation

In [14] Childs shows that a continuous-time quantum walk is universal for quantum computation, proving that it implements a universal gate set. Later on, the same proof was done for the discrete-time quantum walk in [15]. In both cases, they give an explicit construction that converts any circuit into a graph, so that the same algorithm is performed by the walker by traversing the graph derived from the circuit. This allows the mathematics of the quantum walks to be applied in theoretical computer science in a more general way that may imply potential advances, but this does not prove that building a physical quantum walk is an efficient way of creating a general purpose quantum computer.

4 Dirac equation's simulation by DTQW

In this section, we see how it is possible to reproduce the behaviour of a fermionic field in 1D, driven by the Dirac equation, by using a DTQW [16]. In order to achieve this commitment, we are making some changes on the evolution operator, so we first apply the shift operator and then the coin operator. At the same time, we redefine the shift operator so that the up (down) component moves now towards the left (right). The reason why a different convention was chosen in the previous section is that it is the most widespread one. However, in order to arrive at our current goal, this new convention is more convenient since it makes the equations less cumbersome. The key idea of the whole process is to choose the coin operator in a manner that, allows us to reproduce the Dirac's equation in the continuous limit for the quantum walk.

The new evolution operator, W, takes the following form:

$$W = (C \otimes \mathbb{I}_{position})S = US, \qquad S = |\uparrow\rangle\langle\uparrow| \otimes \sum_i |i-1\rangle\langle i| + |\downarrow\rangle\langle\downarrow| \otimes \sum_i |i+1\rangle\langle i|\,. \quad (11)$$

For convenience, a matrix reprsentation on the spin space on the up-down basis, shall be used, so the W is now a 2×2 matrix acting on a two-component wavefunction $\psi_{p,n} = (\psi^{\uparrow}_{p,n}, \psi^{\downarrow}_{p,n})^T$, and takes the form:

$$W = e^{i\alpha}\begin{pmatrix} e^{i\varepsilon}\cos\theta & e^{i\zeta}\sin\theta \\ -e^{-i\zeta}\sin\theta & e^{-i\varepsilon}\cos\theta \end{pmatrix}\begin{pmatrix} \sum_i |i-1\rangle\langle i| & 0 \\ 0 & \sum_i |i+1\rangle\langle i| \end{pmatrix} \quad (12)$$

hence we can write in an explicit practical one-step equation for the walker:

$$\begin{pmatrix} \psi^{\uparrow}_{p,n+1} \\ \psi^{\downarrow}_{p,n+1} \end{pmatrix} = e^{i\alpha}\begin{pmatrix} e^{i\varepsilon}\cos\theta & e^{i\zeta}\sin\theta \\ -e^{-i\zeta}\sin\theta & e^{-i\varepsilon}\cos\theta \end{pmatrix}\begin{pmatrix} \psi^{\uparrow}_{p+1,n} \\ \psi^{\downarrow}_{p-1,n} \end{pmatrix}. \quad (13)$$

In order to simulate the Dirac equation, we need to establish the coin operator, which rules the dynamics, so that the walker evolves as the fermionic Dirac's fields. First of all, we need to make the connection between discrete and continuous space and time. $\{|p\rangle\}_{p\in\mathbb{Z}}$ spans the discrete graph, whereas $\{|x\rangle\}_{-\infty}^{+\infty}$ spans a continuous one dimensional space, where the vectors of the basis $|x\rangle\,/X\,|x\rangle = x\,|x\rangle$ are the eigenvectors of the usual position operator. We can associate each vertex p to a certain position x by choosing a constant distance ϵ_x between two positions related to two contiguous verteces. If $p = 0$ is associated to $x = 0$, it can, then, be written $x_p = p\epsilon_p$, so $|x_p\rangle \equiv |p\rangle$. In the continuous case, a momentum operator, K, can be defined, which is the translation generator. Therefore, a translation of length l in the continuous space is done by the following operator e^{-iKl}. Hence, the translation between two adjacent vertex of the discrete graph is:

$$\sum_p |p\pm 1\rangle\langle p| = e^{\mp iK\epsilon_x}, \quad (14)$$

and the shift operator:

$$S = |\uparrow\rangle\langle\uparrow| \otimes e^{iK\epsilon_x} + |\downarrow\rangle\langle\downarrow| \otimes e^{-iK\epsilon_x}. \quad (15)$$

which is diagonal in the basis of defined momentum $\{|s\rangle \otimes |k\rangle\}_{s\in\{\uparrow,\downarrow\}}$, where $|k\rangle$ are the eigenvectors of the quasi-momentum operator, $K\,|k\rangle = k\,|k\rangle$. Hence the shift operator takes the following form under this representation:

$$S\left|\uparrow\downarrow\right\rangle\otimes\left|k\right\rangle=e^{\pm k\epsilon_x}\left|\uparrow\downarrow\right\rangle\otimes\left|k\right\rangle\Rightarrow S=\begin{pmatrix}e^{ik\epsilon_x} & 0\\ 0 & e^{-ik\epsilon_x}\end{pmatrix}. \tag{16}$$

We repeat the same process for time, by identifying $t_n=n\epsilon_t$, where t is the continuous-time and n the discrete one. Since the evolution operator, W, is unitary, it can be defined as the exponential of a Hamiltonian H as:

$$W=e^{-iH\epsilon_t}. \tag{17}$$

The next step is to find the eigenvalues of W, which are related to the energies of the system since this operator is related to the hamiltonian given by Eq. (17). Then, to find them, the W operator shall be diagonalized. Using Eq. (15) we can rewrite the evolution operator as:

$$W=e^{i\alpha}\begin{pmatrix}e^{i\epsilon}\cos\theta & e^{i\zeta}\sin\theta\\ -e^{-i\zeta}\sin\theta & e^{-i\epsilon}\cos\theta\end{pmatrix}\begin{pmatrix}e^{iK\epsilon_x} & 0\\ 0 & e^{-iK\epsilon_x}\end{pmatrix}=e^{i\alpha}\begin{pmatrix}e^{i\epsilon}\cos\theta e^{iK\epsilon_x} & e^{i\zeta}\sin\theta e^{-iK\epsilon_x}\\ -e^{-i\zeta}\sin\theta e^{iK\epsilon_x} & e^{-i\epsilon}\cos\theta e^{-iK\epsilon_x}\end{pmatrix}, \tag{18}$$

therefore it can be easily diagonalized in the momentum basis, using the result in Eq. (16). One can obtain the following eigenvalues for the W matrix:

$$\lambda_k^{\pm}(\theta,\varepsilon)=\cos(\theta)\cos(\epsilon_x k+\varepsilon)\mp i\sqrt{1-\cos(\theta)^2\cos(\epsilon_x k+\varepsilon)^2}, \tag{19}$$

hence the energy spectrum can be derived taking into account Eq. (17), i.e. $\lambda=e^{-iE_k\epsilon_t}$:

$$\epsilon_t E_k^{\pm}(\theta,\varepsilon)=\pm\arctan\frac{\sqrt{1-\cos(\theta)^2\cos(\epsilon_x k+\varepsilon)^2}}{\cos(\theta)\cos(\epsilon_x k+\varepsilon)}. \tag{20}$$

If the proper limit is taken in Eq. (20), we can recover the relativistic energy dispersion relation. By making the redifinition $\theta\Rightarrow(-\epsilon_m m)$ and taking $(\epsilon_m m)\to 0$, $(\epsilon_x k)\to 0$ and $\varepsilon=0$, we obtain, up to first order:

$$\epsilon_t E_k^{\pm}=\pm\sqrt{\epsilon_x^2k^2+\epsilon_m^2m^2}, \tag{21}$$

therefore, when $\epsilon_x=\epsilon_t=\epsilon_m\to 0$ we finally recover the relativistic formula for the energy, so that this is the limit where a relativistic equation is reproduced. In adition, this condition ensures that causality is preserved, since it imposes a maximum speed for the walker equal to the speed of light $c=\frac{\epsilon_x}{\epsilon_t}=1$ in natural units. The result

$$E_k^{\pm}=\pm\sqrt{k^2+m^2} \tag{22}$$

indicates that the election of the redefinition for the θ angle is adequate since m represents the mass of the walker[5]. Thus, the relativistic limit has imposed the value of θ, which is the only one that induces superpositions of spin up and down, and hence entangles the spatial and spin degree of freedom through the alternation of repeated spin-dependent shifts coin operators, and rules the dynamics of the system in the continuous limit. In order to determine the value

[5]The negative sign is chosen for making it possible to interprete m as the mass of the walker in the Dirac equation, as it will be shown at the end of the demonstrataion.

of the rest of the angles let us look at the coin matrix, which can also be written in terms of the Pauli matrices:

$$\psi_{n+1} = e^{i\alpha} e^{i\frac{\varepsilon+\zeta}{2}\sigma_3} e^{i\theta\sigma_2} e^{i\frac{\varepsilon-\zeta}{2}\sigma_3} S\psi_n. \tag{23}$$

When looking at Eq. (7) where W is applied, the α angle only introduces a global phase to the wavefunction so its value can be put to zero with no effect on the dynamics. Hence, one can rewrite the above equation as:

$$\psi'_{n+1} = e^{i\frac{\varepsilon}{2}\sigma_3} e^{i\theta\sigma_2} e^{i\frac{\varepsilon}{2}\sigma_3} S\psi'_n, \qquad \psi' = e^{-i\frac{\zeta}{2}\sigma_3}\psi, \tag{24}$$

which shows that ζ only makes a rotation on the Bloch sphere around the z-axes. Hence, the proportion between up and down components do not vary [6], and consequently this angle does not alter the dynamics of the system since the DTQW dynamics couples the spatial degree of freedom to the spin components on the up-down spin basis. Eq. (23) can also be rewritten as:

$$\psi''_{n+1} = e^{i\frac{\zeta}{2}\sigma_3} e^{i\theta\sigma_2} e^{-i\frac{\zeta}{2}\sigma_3} \begin{pmatrix} e^{i(k+\varepsilon)} & 0 \\ 0 & e^{-i(k+\varepsilon)} \end{pmatrix} \psi''_n, \qquad \psi'' = e^{-i\frac{\varepsilon}{2}\sigma_3}\psi, \tag{25}$$

where ε not only rotates the states on the z-axes (which would not affect spatial dynamics) but also makes a shift of the momentum by ε, as one can see from the eigenvalues (Eq. (19)). This can also be seen as a change on the momentum basis, so again, it does not affect the dynamics.

As we have demonstrated the values of these angles are irrelevant, so we set them as $\varepsilon = 0$ [7], $\zeta = \frac{\pi}{2}$. We can finally write the one-step evolution equation for the DTQW for describing Dirac equation as:

$$\begin{pmatrix} \psi^{\uparrow}_{p,n+1} \\ \psi^{\downarrow}_{p,n+1} \end{pmatrix} = \begin{pmatrix} \cos(\epsilon_m m) & -i\sin(\epsilon_m m) \\ -i\sin(\epsilon_m m) & \cos(\epsilon_m m) \end{pmatrix} \begin{pmatrix} \psi^{\uparrow}_{p+1,n} \\ \psi^{\downarrow}_{p-1,n} \end{pmatrix}. \tag{26}$$

This equation describes a DTQW which, in the limit $\epsilon \equiv \epsilon_m = \epsilon_x = \epsilon_t \to 0$, describes the dynamics of a fermionic Dirac's field in 1D, where ϵ_x and ϵ_t are the spatial and temporal distances taken at each iteration.

In order to verify our results, let us derive the Dirac equation from Eq. (26). When ϵ tends to zero, we can replace the discrete variables by the continuous ones so one can write:

$$\begin{pmatrix} \psi^{\uparrow}(x_p, t_n+\epsilon) \\ \psi^{\downarrow}(x_p, t_n+\epsilon) \end{pmatrix} = \begin{pmatrix} \cos(\epsilon m) & -i\sin(\epsilon m) \\ -i\sin(\epsilon m) & \cos(\epsilon m) \end{pmatrix} \begin{pmatrix} \psi^{\uparrow}(x_p+\epsilon, t_n) \\ \psi^{\downarrow}(x_p-\epsilon, t_n) \end{pmatrix}. \tag{27}$$

Assuming ψ to be differentiable, it can be developed into Taylor's series in ϵ at first order,

$$\begin{cases} \psi^{\uparrow}(x,t) + \frac{\partial\psi^{\uparrow}}{\partial t}|_{x,t}\epsilon = \psi^{\uparrow}(x,t) + \frac{\partial\psi^{\uparrow}}{\partial x}|_{x,t}\epsilon - i\epsilon m\psi^{\downarrow}(x,t) \\ \psi^{\downarrow}(x,t) + \frac{\partial\psi^{\downarrow}}{\partial t}|_{x,t}\epsilon = \psi^{\downarrow}(x,t) - \frac{\partial\psi^{\downarrow}}{\partial x}|_{x,t}\epsilon - i\epsilon m\psi^{\uparrow}(x,t). \end{cases} \tag{28}$$

[6] This means that the modulus of the coefficient of each component remains invariant under this operation

[7] This choice is the same one that was taken to arrive to Eq. (22) with no shift in the momentum. One could have also chosen $\varepsilon = (\epsilon_x k') \to 0$.

One finally arrives to

$$\begin{cases} \partial_o \psi^{\uparrow} - \partial_1 \psi^{\uparrow} = -im\psi^{\downarrow} \\ \partial_o \psi^{\downarrow} + \partial_1 \psi^{\downarrow} = -im\psi^{\uparrow} \end{cases} \tag{29}$$

where $\partial_0 = \frac{\partial}{\partial t}$ and $\partial_1 = \frac{\partial}{\partial x}$, and only first order terms have been kept. This equation can be recast as

$$(i\gamma^{\mu}\partial_{\mu} - m)\psi = 0 \qquad \psi = \begin{pmatrix} \psi^{\uparrow} \\ \psi^{\downarrow} \end{pmatrix} \tag{30}$$

where μ is summed over from 0 to 1, $\gamma^0 \equiv \sigma_1$ and $\gamma^1 \equiv i\sigma_2$. It can be checked that the γ^{μ} matrices satisfy the Clifford algebra. This corresponds to the flat-space Dirac equation in one dimension, which is in fact the seeked result.

We now make a comparison between the solution that we obtain when we implement the Dirac's DTQW, and a numerical solution of the Dirac equation by using a finite difference method. The solutions are shown in Fig. (6). The parameters used for the walker have to verify the previous conditions we have found so that it truly simulates the behaviour of the Dirac's field. This happens when the product of the mass of the particle, m, and the time-spatial step, ϵ, is small enough. In this case we have used the following values: $m = 1$ and $\epsilon = 0.05$, in natural units. The same values are used in the numerical solution. In order to obtain a symmetric distribution, the initial state for the walker's spin will be $\frac{1}{\sqrt{2}}(|\uparrow\rangle + |\downarrow\rangle)$ starting at a fixed position $x = 0$. This initial condition implies that we are simulating the evolution of a Dirac's field on a 1D graph whose initial state has a null value except for $x = 0$, what makes the numerical solution difficult to calculate, since it corresponds to a Dirac delta function that has to be approximated by a Gaussian distribution narrowly picked at $x = 0$. We have to take this into account since it may be another cause of error apart from the proper error associated with the numerical solution.

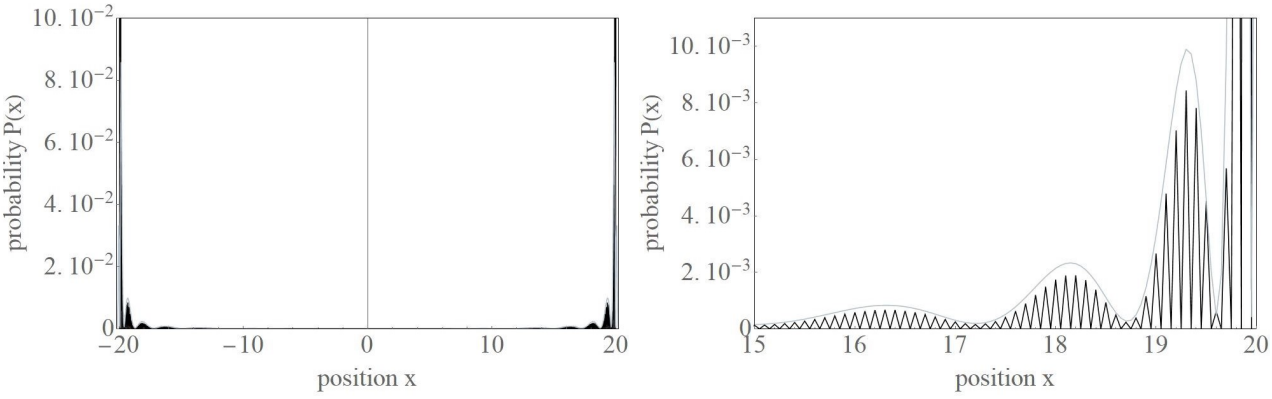


Figure 6: Comparison between the solution of the Dirac equation on 1D space obtained both using a DTQW (black) and a direct numerical solution using a finite difference method (gray). *Left:* complete range of the probability distribution after 20 iterations. *Right:* the range of interest of the probability distribution after 20 iterations is shown to better see the agreement between both solutions.

We obtain a discrepancy between both solutions of the order of 10^{-4}. Since the DTQW is an approximation in ϵm at first order, the truncation error is $\mathcal{O}\left((\epsilon m)^2\right) \sim 10^{-4}$. Hence both solutions are compatible, what proves, again, the power of quantum walks in simulating physical systems.

5 DTQW's implementation on universal quantum computers

5.1 IBM Quantum Experience and Qiskit

IBM Quantum Experience is a project launched to allow researches from all over the world to have access to IBM's prototypes of quantum computers. Hence Quantum Experience represents a platform where users can take advantage of the development of a quantum computer to run their algorithms, whilst IBM can test its computers and receive the feedback from the users, so they can improve faster in the career for building a completely functioning quantum computer. Moreover, IBM also released an open-source development kit, Qiskit, for the development of Software for quantum computers that can be tested on their computers. Qiskit uses Python and is based on the library OpenQASM for the representation of the quantum circuits. Therefore, we can write every circuit by using Qiskit and then run it on a real quantum computer.

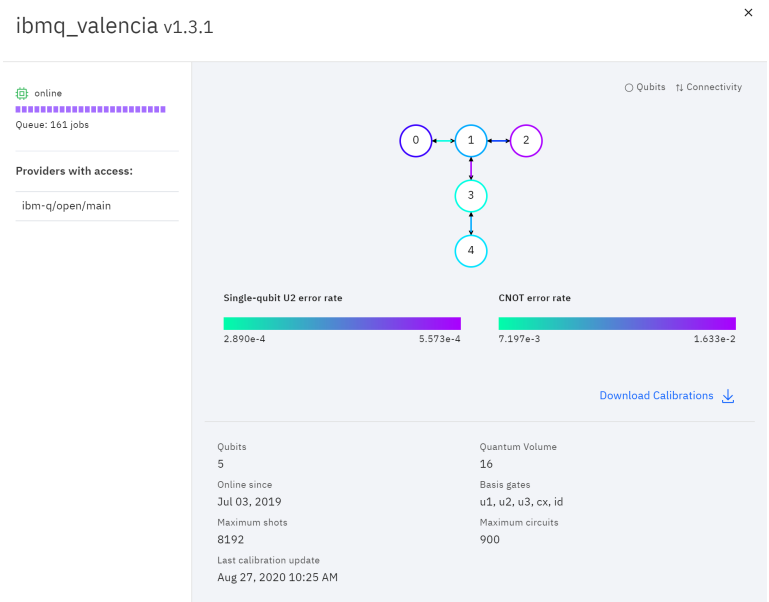


Figure 7: Specification sheet of IBM's quantum computer, *ibmq_valencia* $v1.3.1$ prototype. It shows the number of qubits and their layout, together with the quantum gates that its hardware implements, which are a set of universal gates, along with their error rate.

In Fig. (7), it is shown the specification sheet about one of the available computers. There, one can read the information about the number of qubits and the morphology of the device, which shows how the qubits are connected among them. This is a crucial issue to take into account for developing software for a concrete computer, since applying an operation over qubits which are directly connected decreases the errors. At the same time, it also shows the basis gates the computer implements and their error rates. Qiskit will directly translate the circuits we create to this basis and implement it on this particular device. The errors are obtained by a calibration comparing the results the computer produces when running circuits whose results are well-known.

In Sect. 5.3, we will be implementing a DTQW on a real quantum computer via IBM quantum experience. Hence, we will write a quantum circuit which describes the QW and then we will run it on one of the accessible IBM's quantum computers.

5.2 Quantum noise

Nowadays we have access to prototypes of quantum computers as we have seen. Nevertheless, we are still far from building a real useful quantum computer with enough computational power to achieve the universality predicted theoretically. This is mainly due to the quantum noise that engineers and scientist are still facing.

During this study, we have only dealt with closed quantum systems, systems that are entirely isolated from the rest of the universe, so they do not suffer any interaction with it. These systems are idealizations of the real world, since such closed systems do not exist. These interactions with the outside world are responsible for the quantum noise that prevent a real system from reproducing the results calculated when considering the ideal closed systems. Hence, when building useful quantum processing systems, we need to understand and control these interactions to mitigate their impact on the computations. Consequently, one of the most critical issues in research is to study the dynamics of open quantum systems.

As a first approach to the problem, we will start by introducing the noise in a classical system in order to get some intuition.

We can imagine a bit stored on a hard disk. This bit initially has a particular state 0 or 1. However, after a certain time, due to the interactions with the environment that may contain magnetic fields, the bit may be forced to flip to the opposite value with a probability p.

This situation implies that, due to the environment interactions, after a certain time, we do not precisely know the state of the bit, but can only know the probability of the bit to be in a particular state: p to stay in the same state, and $1-p$ to flip to the other one. In order to handle these interactions we need to build a model of the environment (in this case, a model for the magnetic fields), and another one that describes the interaction of the bit with it (in this case the Maxwell's equations). However, this uncertainty on the state of the bit is not due to any quantum property; it arises from the lack of knowledge of the interactions the bit is having, so we can only describe it probabilistically.

Similar scenarios happen in a quantum system, so the environment noise does not allow us to know the state of a quantum register perfectly but instead, we can only know the probabilities for the quantum register to be in a specific quantum state. This means that in addition to the probabilistic nature of the quantum system described by its quantum state, we have to add a 'classical probability' for the register to be in different quantum states.

To describe this situation, in a classical system one introduces a vector of probabilities. In the quantum case, we shall describe quantum states in terms of the density operator, ρ. The density operator is defined as:

$$\rho = \sum_n p_n \left|\Psi_n\right\rangle \left\langle\Psi_n\right|, \qquad \sum_n p_n = 1, \qquad 1 \geq p_n \geq 0, \tag{31}$$

where p_n are the probabilities for the system to be in the state $|\Psi_n\rangle$. For an isolated system, the evolution of the density operator is also driven by a unitary evolution. If pure states evolve under unitary transforms as $|\Psi\rangle \to U|\Psi\rangle$, then the evolution for a density operator is given by $\rho \to U\rho U^\dagger$.

As an example of quantum noise, we will discuss *phase damping* [3], which describes the loss of quantum information without loss of energy. Physically this model can be applied to describe how electronic states in an atom are perturbed when interacting with distant electrical charges.

Let us suppose a qubit in a pure state $|\Psi\rangle = a\,|0\rangle + b\,|1\rangle$, which is physically implemented by two states of an electron. Since we suppose that the interaction with the environment does not imply the loss of energy, if we denote the state 0 to be the one of lower energy of the electron, $|0\rangle = |E_-\rangle$, and the state 1 for the upper energy, $|1\rangle = |E_+\rangle$, we get that the mean value for the energy must be preserved: $\langle E\rangle = |a|^2 E_- + |b|^2 E_+$.

This can be modelled so that the interaction with the environment happens through rotations around the z-axes on the Bloch sphere (Fig. (2)). Hence, the components on the computational basis for the qubit only vary in a complex phase, so that the energy does not change through time. Plus, we cannot know the exact angle the rotation performs, since it shall depend on the environment state. Hence, this angle θ is well represented as a random variable which has a gaussian distribution with mean value 0 and average 2λ, so the most probable event is for the qubit to stay invariant.

The evolution of the density operator, after each interaction is:

$$\rho = \frac{1}{\sqrt{4\pi\lambda}} \int_{-\infty}^{+\infty} R_z(\theta)\,|\Psi\rangle\,\langle\Psi|\,R_z(\theta)^\dagger e^{-\frac{\theta^2}{4\lambda}} d\theta = \begin{pmatrix} |a|^2 & ab^* e^{-\lambda} \\ a^* b e^{-\lambda} & |b|^2 \end{pmatrix}, \tag{32}$$

if we consider that the number of interactions grows linearly with time, the evolution will be:

$$\rho(t) = \begin{pmatrix} |a|^2 & ab^* e^{-\lambda t} \\ a^* b e^{-\lambda t} & |b|^2 \end{pmatrix}. \tag{33}$$

In eq.(33) we see that the phase kicking causes the off-diagonal values to decay exponentially to zero along with time, so we finally obtain in the limit, $t \gg 1/\lambda$:

$$\rho(t \gg \lambda) = \begin{pmatrix} |a|^2 & 0 \\ 0 & |b|^2 \end{pmatrix} = |a|^2\,|0\rangle\,\langle 0| + |b|^2\,|1\rangle\,\langle 1|\,. \tag{34}$$

The initial superposition state disappears, and finally, we get a state that no longer exhibits superposition, but behaves as a classical bit with noise with a probability $p = |a|^2$ to have the value 0 and a probability $q = |b|^2$ to have the value 1. This means that phase damping makes the system lose their quantum properties. This is called decoherence.

Another significant quantum noise that has to be taken into account is the so-called *amplitude damping*. This phenomenon describes the energy dissipation effects due to the interaction with the environment that the quantum system may be coupled to, for instance the spontaneous decay of a two level system (qubit). This kind of quantum noise can also be modelled, but it will consider rotations off the z-axes, so the energy of the system changes. A similar equation to Eq. (33) can be derived, but the exponential decay falls on the diagonal elements. Nevertheless, it can be proven that these phenomena always occur at later times than the phase damping. Hence, the main problem physicists encounter when dealing with quantum noise is phase damping and the loss of *quantumness* associated with this kind of interaction with the environment. If this process is diminished, the dissipative quantum noise will also be reduced.

In the following section, Sect. 5.3, we will see how important these effects are in the development of quantum computers, because of their significant impact on the results of the computations.

5.3 Scalable networks for discrete quantum random walks

In Sect. 3.2, we have already seen that the physical implementation of QWs is being studied and several experimental schemes have been proposed. Nevertheless, these physical implementations based on the realization of a physical quantum walk encounter serious problems when it comes to the implementations on higher dimensions. The proposal of S. Fujiwara, H. Osaki, I. M. Buluta, and S. Hasegawa [1] is a universal method for implementing QWs in arbitrary physical systems and not only specific devices that are designed for this propose. Moreover, this method can be generalized to higher dimensions; notwithstandind, we are only going to study the quantum networks for DTQW on circles and lines.

In order to achieve this universality, we will express the algorithm of an arbitrary QW into a circuit which can be implemented in a general-purpose quantum computer. This method encodes all the information in a quantum register where the position is stored, and there is no random walk taking place in the real world. Likewise, the evolution of the walker is accomplished only by universal gates, so the independence of the physical implementation is assured. Consequently, these networks are purely computational. In addition, as we will show later, the implementation of quantum walks requires $log_2 n$ qubits, being n the number of vertices, whereas in many forms of implementation for the QW in laboratories the number of devices needed scales with n, so a quantum computer gives us an exponential advantage.

Hence, we will not only study it but also implement it by using Qiskit so that we can test it on a real quantum computer by using the IBM Quantum Experience. As it is the main objective of this study, we will focus only on the DTQW on 1D, although, as it has been mentioned before, it is possible to generalize the network to higher dimensions as it is described in [1]. Now we recover the convention of Sect. 3 where the coin operator is first applied (and then the shift operator). Now the spin space will be represented by the following basis, $\{|0\rangle, |1\rangle\}$. The zero (one) - state moves to the left (right).

The preparation of the initial state can also be done by universal gates acting on the quantum register. The results are obtained by measuring the quantum register excluding the ancilla qubits, which represent the coin space state.

To describe the quantum walk on a circle the only new property one has to take into account is the periodicity condition: we can imagine it as a segment with a determined number of nodes so that the first and the last one are connected to each other. The position of the walker will be encoded in a register of n qubits, so that the circle will have 2^n nodes. The shift operator takes the following form:

$$S = \sum_{s\in\{0,1\}} |s\rangle\langle s| \otimes Z^{(n)}_{2s-1}, \tag{35}$$

$$Z^{(n)}_{\pm 1} = \sum_{k=0}^{2^n-1} |k \pm 1 (mod 2^n)\rangle\langle k| \qquad \begin{cases} Z^{(n)}_{+1}|k\rangle = |k+1(mod\,2^n)\rangle \\ \\ Z^{(n)}_{-1}|k\rangle = |k-1(mod\,2^n)\rangle \end{cases}, \tag{36}$$

where $Z^{(n)}_{+1}$ relates a fixed position state to the next one, and $Z^{(n)}_{-1}$ to the previous one. The arithmetic $mod\,2^n$ ensures the shift operator verifies the boundary condition on the circle. For describing a DTQW on a line, a large enough register has to be used, so the walker never reaches the boundary condition.

The general scheme for the network of a DTQW on a 1D is described in Fig. (8). We have a wire representing the n qubits for the position and another one, which is an ancilla qubit,

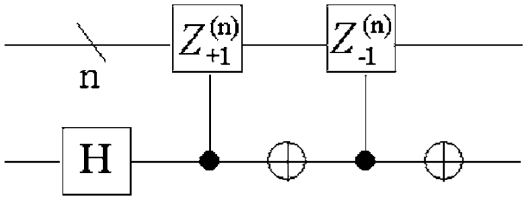


Figure 8: General circuit for a DTQW on a circle. The bottom is the ancilla qubit for the coin-spin and the other qubits (represented by a single line) store the infromation of the position. In this case, the coin operator is a Hadamard gate, and the following set of gates are the representation of the shift operator.

for the coin space. The coin operator is applied only on the ancilla qubit. The implementation of the shift operator is done by using controlled-U and NOT gates, previously described in Sect. 2. The first controlled-U gate makes the $Z_{+1}^{(n)}$ operator act over the states with value 1 on the coin space. The use of the NOT gates before and after the second controlled-U gate makes the $Z_{-1}^{(n)}$ operator act over the states with value 0 on the coin space. Therefore, the 1 components move to the right and the 0 components to the left.

The only problem left to be solved is how to perform the $Z_{\pm 1}^{(n)}$ operators as quantum networks. The solution, also obtained in this paper [1], is shown in Fig. (9),

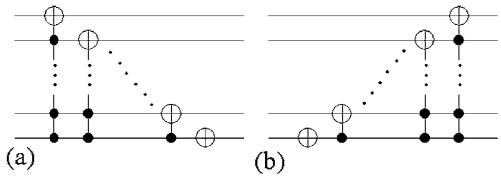


Figure 9: Decomposition of the shifts operators into NOT and m-CNOT gates: (a) $Z_{+1}^{(n)}$ and (b) $Z_{-1}^{(n)}$. To perform the controlled-shift operator with the coin qubit as the controlled qubit (Fig. (8)), each m-CNOT gate has to be replaced by another (m+1)-CNOT that adds the coin qubit as a new controlled qubit.

where both $Z_{+1}^{(n)}$ in (a) and $Z_{-1}^{(n)}$ in (b) are described, so one can see that only NOT and m-CNOT gates are used. The m-CNOT gate is a NOT-gate which acts over a single qubit only if the other m qubits are set to state $|1\rangle$.

The next step to implement the QW on real hardware is to write it in Qiskit, and learn how to program on a real quantum computer. The only tasks left are to find how to properly encode the positions in the quantum register, and to find how to implement the m-CNOT on Qiskit.

In the implementation we found, described in Fig. (10), we had to add, in addition to the $m+1$ previous qubits, $m-2$ ancilla qubits in order to perform the m-CNOT by using CNOT gates. The ancilla qubits are needed to store the information of the state of the m control qubits. The NOT operation over the target qubit is only applied if in the quantum state all control qubits are set to 1. The qubits labelled from 0 to $m-1$ are the control qubits, and the m qubit is the target. The other $m-2$ qubits are the ancilla ones. Up to the CNOT gate that acts on the target qubit, the previous CNOT changes the state of the ancilla qubits, initially set to 0. Then, the CNOT with target on the m qubit will only act if all the ancilla qubits were set to 1, which is only possible if the states of all control qubits is 1. The rest of the CNOTs ensures that all values of the ancilla qubits are restored to 0.

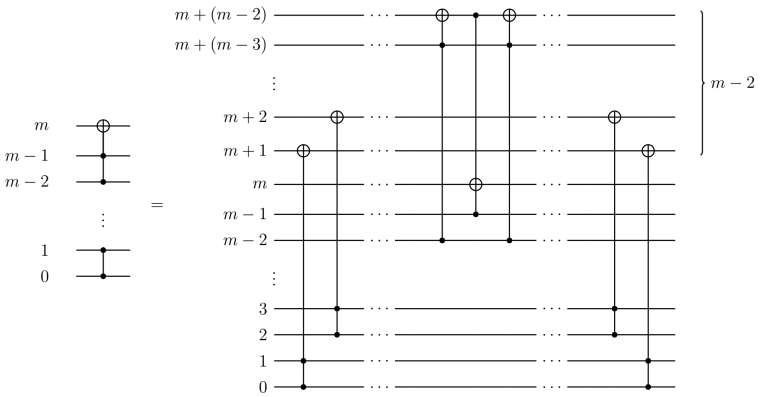


Figure 10: Squematical implementation of a m-controlled NOT gate by using CNOT gates and $m-2$ ancilla qubits.

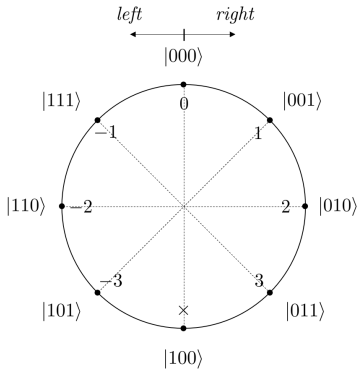


Figure 11: Circular graph with 3 qubits: 8 vertices.

In addition, a quantum register of n qubits is described in a Hilbert space of dimension 2^n. This means that the basis has also 2^n states, so a graph with 2^n nodes can be encoded. If we are considering a walker on a line, we have to avoid the boundary conditions of the circle. Therefore if the initial state of the walker is fixed on a single position, the number of allowed iterations is $2^{n-1}-1$ since the QW can walk one step in both directions at each iteration. In the formula, we have taken into account that one state of the basis cannot be used since in it, both directions would join. This situation is perfectly shown in Fig. (11) where we can see an example with three qubits, hence a maximum number of iterations of 3. However, this only occurs in a theoretical case; if the QW implementation is not perfect, even if the number of steps is limited, one can find non-zero probabilities in all possible states due to quantum noise, so this will have to be taken into account when dealing with the results. Let us suppose the walker starts in the position state $|000\rangle$, so we can associate the decimal value 0 to that binary number. When the walker spreads towards the right, it will get to the state $|001\rangle$, so we can label it by the corresponding decimal value, 1. Nevertheless, when moving to the left, the state is $|111\rangle$, whose decimal number is 7. This is because, in this model, we have considered a circle as the graph, with its corresponding modular arithmetic. To adequately describe the walker on a line we need to codify the position, so the movement to the left from the initial position corresponds to negative numbers. That is to say, in this example, the state $|111\rangle$ would be the position, -1. As the number of nodes is always even, we can use the usual relation between binary and decimal systems up to the position 2^{n-1} (in decimal) that is the node where right and left touch each other if the walker starts in the state $|000\rangle$. After that position we only have to use the following modification: $x'(decimal) = x(decimal) - 2^n$, where x is the usual decimal number corresponding to the binary one, and x' is the new number we need to represent the position on a line accurately.

Finally, in `https://github.com/alejandroTFG-UV/TFG-quantum-code.git`, one can find the code used for writing a program that executes a DTQW on a circle of 2^n nodes and N steps based on Fujiwara's implementation.

Now, we have a program to run a quantum walker on IBM's quantum computers using Qiskit. Once we have written the quantum circuit, Qiskit allows us to run it on a simulation of a quantum computer. Given that the computation we are running is not very demanding, the simulation will work correctly, although it may not be efficient. Hence, the simulation will give us the expected results, which can be used to compare the ones we will get from IBM's computer.

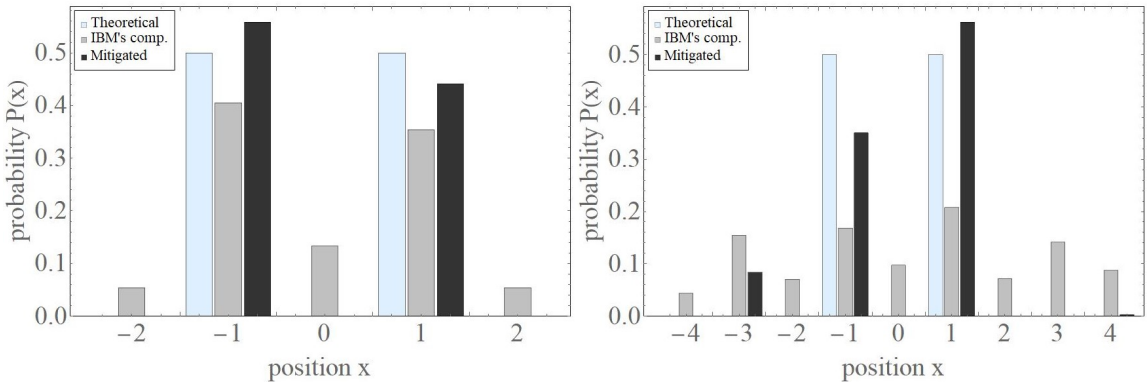


Figure 12: Results of the measuremt of the position of the DTQW after 1 steps with a different amount of qubits for coding the position: *Left:* 2 qubits. *Right:* 3 qubit. In both graphs there are shown the results given by the simulation, the IBM's computer (*ibmq_valencia* v1.3.1, specification sheet in Fig. (7)) and using the mitigation error processing.

In Fig. (12), we show the results after one step of the walker using 2 and 3 qubits for the position register. We see that even after applying only one iteration, the values given by the quantum computer do not match the theoretical ones, even appearing non-zero probabilities where the quantum walker is not expected to be found. As we have mentioned, this is due to quantum noise whose effects depend mainly on the calculations' time because of the life-time of the damping phase effect. Hence, one of the most important sources of error comes from increasing the number of iterations, which ended up on results that were any more reliable due to the quantum noise, so only one step could be run. At the same time, the results obtained when using fewer qubits are better, since the number of possible states is fewer. In the left graph in Fig. (12), although we find a significant discrepancy when comparing the results to the simulation, we still find that the correct values $(-1, 1)$ are the most likely. This tendency begins to disappear when increasing the qubits in the register. The more qubits, the more quantum noise the system may suffer, since the interaction with the environment is stronger and the number of gates needed is larger, which causes the simulation to run longer and be affected by the noise. This shows one of the difficulties on building quantum computers which are computationally powerful enough, since increasing their size makes it more difficult to control the quantum noise. Another source of error may be coming from the translation Qiskit does from the circuit's gates we write to the gate basis implemented on the quantum computer, since the number of gates is usually increased with respect to the original circuit, which can lead to more errors during the calculation than originally expected. These conclusions strengthen the need to learn how to control the quantum noise in order to achieve the quantum supremacy by building a real useful quantum computer.

As a first approach to the management of noise, we also applied the so-called measurement error mitigation, that corrects a simple form of noise that occurs during the final measurement, when the circuit extracts a bit string as an output. In a measurement of n qubits, there are 2^n possible outputs, so one will be selected in a perfect and noiseless manner. However, this ideal output will then be randomly perturbated before it is returned to the user.

Therefore, by analysing how noise affects the measurement over each of the 2^n possible outcomes, we can use these results to correct the measurement over a general state of the register. This can be done by creating a matrix that gathered the data obtained when studying the noise affecting each state of the basis. This matrix can be then thought as a change of basis matrix that transforms the states of the basis in the states that would ideally reproduce

the noisy results. The linearity of the matrix will then let us apply the same process to a more general state. Hence, this matrix gives us a good approximation of the results when measurement noise is present: $C_{noisy} = MC_{ideal}$. Consequently, we can obtain the ideal measurement by finding the inverse matrix: $C_{ideal} = M^{-1}C_{noisy}$. This process allows us to mitigate the noise after the measurement of the register. In Fig. (12), the mitigated result appears in black. This method improves the results significantly. For two qubits (left), the mitigated results are almost correct, and only the right outputs are measured. In the three qubits' case, there are other output results, but the correct ones still are the most likely ones.

fidelity	*ibmq_valencia* v1.3.1		mitigated	
	2 qubits	3 qubits	2 qubits	3 qubits
$1-d$	0.7593	0.3766	0.9414	0.8505

Table 1: Fidelity of the probability distributions of the quantum computer and the mitigated measures with respect to the theoretical values. $d = \frac{1}{2}\sum_x |p_x - q_x|$, is the total variation distance.

To give a mathematical notion of the deviation of both experimetal and mitigated distributions with the theoretical values, the *fidelity* is calculated in Table 1. The results show that, as we have analyzed, the mitigated distribution is closer to the theoretical distribution since its fidelity is closer to 1, and it improves when only 2 qubits are used.

6 Conclusions

Quantum computation is based on the same ideas that gave birth to classical computation. Their objective is the manipulation of the information in order to solve problems using algorithms. The innovation comes from the way that the information is encoded and treated through systems whose behaviour is ruled by quantum mechanics.

This advance not only postulates the possibility to simulate complex quantum systems efficiently, since they both are based upon the same precepts, but its fundamental properties such as superposition and entanglement open up the chance for developing algorithms that can solve efficiently problems which are irresoluble up to now. Quantum parallelism consists of the simultaneous computation of multiple output values by making use of the superposition of the quantum register's states. This feature may seem to be a useful tool capable of making fast calculations, but one also has to take into account that the measurement of the information stored in the quantum state destroys the superposition. However, the design of quantum algorithms that exploit this parallelism so that the measurement does not cancel the total gain can be developed, as Deutsch's algorithm of factorisation [6]. This is a great chance for quantum supremacy.

The *quantum walk* (QW) is a purely quantum mathematical model that constitutes the foundations of an extended variety of quantum algorithms and a Turing complete model for quantum computation, see Sec. 3.3; besides it is a widespread scheme for simulating quantum systems. In Sec. 3.1 we deeply explore a DTQW on 1D by comparing it with its classical counterpart and finding that a QW spreads quadratically faster than a RW on a line, Fig. (4), which led to more efficient algorithms such as the quadratic speed advantage on searching algorithms first found by Shenvi [11]. We also studied the QW's application to physics simulations and verified that it can reproduce the behaviour of a fermionic Dirac field of spin 1/2 in a one-dimensional space, as can be seen in Fig. (6). Nevertheless, this result just represents

the beginning of an extensive list of theoretical application on the development of simulations of physical systems such as the simulation of electric and gravitation fields [17], the simulation of the neutrinos' oscillations [18] or the simulations of high energy theories in extra dimensions [19].

Once stated the relevance of the QW, we can think of the importance of building quantum computers that allow them to be efficiently computed. Naturally, they can be implemented on classical computers. Nevertheless, this kind of implementation losses the quantum advantages and has to face the usual problems on simulating quantum systems by classical means. In particular, the number of complex amplitudes that describe a system increases exponentially with its size. For a DTQW on 1D with 10^6 steps, approximately 16 Mb of memory is needed in a classical computer whilst a quantum computer only needs 20 qubits. This shows how fast can a quantum computer overpass the classical capabilities in simulating quantum systems, so a quantum computer of only 40 qubits could achieve this supremacy.

Therefore, in Sect. 5, we explored the implementation of the DTQW-1D on a quantum computer. In Fig. (12), we can see the results of an IBM's prototypes, which are remarkably affected by quantum noise (see Table 1). However, a new scheme was recently developed that allows a QW to take more steps and walk in a higher-dimensional space with success [20]. Quantum noise is one of the current difficulties that the development of a quantum computer 'reliable enough to be computationally useful' is facing.

Finally, we remark that along with the research, we have not only studied and developed the theory underlying the quantum computation and the QW, but we have also seen the main limitations that the quantum computers are facing nowadays. Nevertheless, the effort to overpass them is worth it, under the pledge of the realisation of a new model of computation that will be able to solve a bunch of complex problems and simulating physical systems that could lead to a wide range of advances in many research fields, even outside physics.

La computación cuántica esta basada en las mismas ideas que originaron la computación clásica. Su objetivo es el tratamiento de la información para la resolución de un problema mediante un algoritmo. La innovación se basa en que esta información es codificada y manipulada bajo sistemas cuyo comportamiento está regido por las leyes de la mecánica cuántica.

Este avance no sólo postula la posibilidad de simular sistemas cuánticos complejos de forma más eficiente, ya que ambos se basan en los mismos preceptos, sino que sus propiedades fundamentales como la superposición y el entrelazamiento abren la puerta al desarrollo de algoritmos para la resolución eficiente de problemas hasta ahora irresolubles. El paralelismo cuántico o paralelismo exponencial, consiste en la computación simultánea de varios valores de salida, gracias a la superposición de estados de los registros cuánticos; las operaciones se ejecutan de forma simultánea sobre cada uno de los valores de la superposición cuántica. Esto permitiría realizar cálculos muy eficientemente, pero como es conocido, a la hora de acceder a la información, su estado se altera destruyendo dicha superposición tras la medida. No obstante, es posible diseñar algoritmos cuánticos que exploten el paralelismo sin que la medida afecte de manera total a la ganancia de las superposiciones cuánticas, como demostró Deustch con su algoritmo de factorización [6]. Esta es la gran baza de la supremacía de los ordenadores cuánticos.

Dentro de este nuevo modelo computacional surgió de forma natural el *quantum walk* (QW) como generalización del *random walk* (RW). El QW se presenta como un modelo matemático

puramente cuántico que supone un importante cimiento sobre el que se estructura la concepción de muchos algoritmos cuánticos y que puede constituir por sí mismo un modelo de computación universal (ver Sec. 3.3), así como un esquema de simulación de sistemas físicos. En un estudio detallado del DTQW en 1D se ha obtenido como primer resultado que este se dispersa cuadráticamente más rápido que su análogo clásico, como se ven en la Fig. (4). Aunque esto no implica por sí mismo un avance que suponga su superioridad sobre el modelo clásico, fue lo que llevó a Shenvi al desarrollo de algoritmos de búsqueda más eficientes [11]. A su vez su capacidad para la modelización de sistemas cuánticos ha sido también expuesta mediante la demostración y posterior comprobación de la simulación del DTQW en 1D de un campo fermiónico de spin 1/2 a lo largo de una dimensión , en la Fig. (6). Sin embargo esto sólo supone el inicio de una larga serie de aplicaciones teóricas en la simulación de sistemas físicos entre las que podemos encontrar la simulación de campos eléctricos y gravitatorios [17], la simulación de la oscilación de neutrinos [18] o la simulación de teorías de altas energías en dimensiones extra [19].

Vistas las importantes aplicaciones del QW, se ve necesario el desarrollo de ordenadores cuánticos que permitan computarlos de forma eficiente. Naturalmente, estos pueden ser implementados en ordenadores clásicos, pero sufriendo los problemas de la simulación de los sistemas cuánticos. El número de amplitudes complejas de la función de onda del sistema escala exponencialmente con su tamaño. Para un DTQW en 1D con 10^6 pasos, se requieren en torno a 16 Mb de memoria en un ordenador clásico mientras que solo 20 qubits son necesarios. Así es fácil ver como esto puede escalar hasta superar las capacidades de los ordenadores clásicos para un mayor número de pasos, y en general para sistemas más complejos, de forma que un ordenador de apenas 40 qubits podría superar la capacidad de los ordenadores clásicos.

De ese modo, en la Sec. 5 se ha estudiado la implementación del DTQW 1D en un ordenador cuántico. En Fig. (12) se muestran los resultados obtenidos en un prototipo de IBM, donde se ve como estos están altamente afectados por el ruido cuántico (ver Tabla 1). El ruido cuántico es una de las dificultades actuales a las que se enfrenta el desarrollo de un ordenador cuántico con la fidelidad suficiente para ser computacionalmente útil. Sin embargo, esto puede cambiar en poco tiempo; recientemente se han hecho avances en este campo de forma que se han podido realizar un mayor número de pasos en el QW [20].

A lo largo del trabajo se ha resaltado no sólo el marco teórico de la computación cuántica y del QW, sino que hemos visto las posibles limitaciones de los ordenadores cuánticos y como estas, sin embargo, merecen ser superadas por la promesa de un modelo de computación que abra las puertas a la resolución de nuevos y más complejos problemas, y simulaciones de sistemas físicos que podrían desembocar en grandes avances en un amplio abanico de campos de investigación, incluso fuera de la física.

References

[1] S. Fujiwara, H. Osaki, I.M. Buluta, and S. Hasegawa. *Scalable networks for discrete quantum random walks*. 2005.

[2] M. Nagy and Selim G. Akl. *Quantum computation and quantum information*. 2006.

[3] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. 2010.

[4] R. Mullins. *What is a Turing machine?* 2012.

[5] R. P. Feynman. *Simulating Physics with Computers*. 1981.

[6] D. Deutsch. *Quantum theory, the Church-Turing principle and the universal qunatum computer*. 1984.

[7] Y. Aharanov, L. Davodovich, and N. Zagury. *Quantum random walks*. 1993.

[8] R. Portugal. *Quantum Walks and Search Algorithms*. 2014.

[9] J. Kempe. *Quantum random walks: An introductory overview*. 2010.

[10] V. Kendon. *A random walk approach to quantum algorithms*. 2006.

[11] Shenvi, Kempe, and B. Whaley. *A quantum random walk search algorithm*. 2003.

[12] C. H. Bennet. *Local Reversibility of Computation*. 1973.

[13] G. J. Milburn, S. Schneider, and D. F. V. James. *Ion Trap Quantum Computing with Warm Ions*. 2000.

[14] A. M. Childs. *Universal computation by quantum walk*. 2009.

[15] N. B. Lovett, S. Cooper, M. Everit, M. Trevers, and V. Kendon. *Universal quantum computation using the discrete-time quantum walk*. 2010.

[16] P. Arnault. *Discrete-time qunatum walks and gauge theories*. 2017.

[17] G. Di Molfetta, M. Brachet, and F. Debbasch. *Quantum Walks in artificial electric and gravitational Fields*. 2018.

[18] G. Di Molfetta and A. Pérez. *Quantum walks as simulators of neutrino oscillations in a vacuum and matter*. 2016.

[19] I. Márquez-Martín, G. Di Molfetta, and A. Pérez. *Fermion confinement via quantum walks in (2+1)-dimensional and (3+1)-dimensional space-time*. 2017.

[20] F. Acasiete, F. P. Agostini, J. K. Moqadam, and R. Portugal. *Experimental Implementation of Quantum Walks on IBM Quantum Computers*. 2020.