UNIVERSITAT JAUME I

Computational Mathematics Degree

Final Degree Project

An introduction to Quantum algorithms

Author:
Vicente López Oliva

Academic tutor:
José Manuel Badia Contelles
Ximo Gual Arnau

Reading date: 08 of November of 2020
Academic year 2019/2020

Abstract

Nowadays, we have powerful computers capable of performing very complex operations in seconds. However, there are problems that cannot be addressed at reasonable execution times, such as NP-Complete problems. For these problems no polynomial solution is known, having all of them exponential cost, which makes them unfeasible for classical computers.

Quantum properties have impressed equally to computer scientists, physicist and all kinds of scientists since they were discovered, since they are very unintuitive. However, these properties, such as entanglement or superposition, lead us to have quantum computers and quantum algorithms that are able to solve in polynomial time some of the NP-Complete problems. This would be a breakthrough in fields with high computational demands such as machine learning, medicine, chemistry, etc.

In this work, we will focus on knowing the computing mathematical basis that will allow us to study the complex quantum world, which includes, among other things, complex vector spaces and complex algebra. We will study the properties offered by the quantum computing world, such as the superposition of states or entanglement. We will also see quantum gates that will allow us to build quantum circuits to be able to create algorithms that we can execute on our quantum computers. We will finish the work by studying some well knows algorithms which will allow us to see how to take advantage of the quantum properties to accelerate our computing capacity.

Keywords

Complex vector space, Hilbert space, complex algebra, quantum computing, quantum algorithms, quantum teleportation, Deutch's algorithm.

Contents

List of Figures

1

Introduction to Complex Space

In order to learn quantum computing and quantum algorithms, it is first necessary to become familiar with the complex space. In this chapter we introduce some basics about complex space, in order to give the necessary concepts to have a basis understanding of the basics of quantum mechanics. This introduction is based on [1]

1.1 Basic Definitions

The original motivation for defining the complex numbers was the fact that there are cases in which an algebraic equation has no solution, like in this example:

$$x^2 + 1 = 0 \tag{1.1}$$

Indeed, any x^2 of $\mathbb{R}$ would be positive or zero, so there is no possible solution for this equation. Let's supose that there is a number that can solve this equation, then, this number is in fact:

$$x = \sqrt{-1} \tag{1.2}$$

This number does not exists in the real space and it is not defined. To have a solution to equation (1), we need to have defined this number in a new space, so let's define it.

Definition 1.1.1: The solution of the equation (1.2) is known as *imaginary number* and it is denote by i.

Definition 1.1.2: A number which is formed by a real part and an imaginary part is known as *complex number* and it has the form $c = a + bi$ where a is the real part and b is the imaginary part with $a, b \in \mathbb{R}$. The set of all complex numbers is denoted by $\mathbb{C}$.

Complex numbers can be added and multiplied. Let $c_1, c_2 \in \mathbb{C}$ such that $c_1 = a_1 + b_1 i$ and $c_2 = a_2 + b_2 i$, so we can compute addition and multiplication of this two arbitrary complex numbers as

$$\begin{aligned} c_1 + c_2 &= a_1 + b_1 i + a_2 + b_2 i = (a_1 + a_2) + (b_1 + b_2)i \qquad (1.3) \\ c_1 \times c_2 &= (a_1 + b_1 i) \times (a_2 + b_2 i) \\ &= (a_1 a_2 - b_1 b_2) + (a_1 b_2 + a_2 b_1)i \qquad (1.4) \end{aligned}$$

From now on, let us denote $c_1 \times c_2$ as $c_1 c_2$. Definition 1.1.2 tell us that a complex number is formed by a real part plus an imaginary part, so we can imagine that we can identified any complex number as a pair $(a, b) \in \mathbb{R}^2$. With this notation of vectors, we can redifine (1.3) and (1.4) as follows. Let $c_1 = (a_1, b_1)$ and $c_2 = (a_2, b_2)$.

$$\begin{aligned} c_1 + c_2 = (a_1, b_1) + (a_2, b_2) &= (a_1 + a_2,\ b_1 + b_2) \qquad (1.5) \\ c_1 c_2 = (a_1, b_1) \times (a_2, b_2) &= (a_1 a_2 - b_1 b_2,\ a_1 b_2 + a_2 b_1) \qquad (1.6) \end{aligned}$$

It is trivial to see that both aperations, multiplication and addition, are commutative and associative, that means:

$$\begin{aligned} c_1 + c_2 &= c_2 + c_1 \qquad (1.7) \\ (c_1 + c_2) + c_3 &= c_1 + (c_2 + c_3) \qquad (1.8) \\ c_1 c_2 &= c_2 c_1 \qquad (1.9) \\ (c_1 c_2) c_3 &= c_1 (c_2 c_3) \qquad (1.10) \end{aligned}$$

Moreover, we can see that multiplication distributes over addition.

$$c_1 \times (c_2 + c_3) = c_1 \times c_2 + c_1 \times c_3 \qquad (1.11)$$

Let us verify this property. Let $c_1 = (a_1, b_1)$, $c_2 = (a_2, b_2)$ and $c_3 = (a_3, b_3)$ then

$$\begin{aligned} c_1(c_2 + c_3) &= (a_1, b_1) \times ((a_2, b_2) + (a_3, b_3)) \\ &= (a_1(a_2 + a_3) - b_1(b_2 + b_3), \\ &\quad a_1(b_2 + b_3) + b_1(a_2 + a_3)) \\ &= (a_1 a_2 + a_1 a_3 - b_1 b_2 - b_1 b_3, \\ &\quad a_1 b_2 + a_1 b_3 + b_1 a_2 + b_1 a_3). \qquad (1.12) \end{aligned}$$

Now we will calculate the right side of equation (1.11)

$$\begin{aligned} c_1c_2 &= (a_1a_2 - b_1b_2,\ a_1b_2 + a_2b_1) && (1.13)\\ c_1c_3 &= (a_1a_3 - b_1b_3,\ a_1b_3 + a_3b_1) && (1.14)\end{aligned}$$

if we sum them up, we have

$$\begin{aligned} c_1c_2 + c_1c_3 &= (a_1a_2 + a_1a_3 - b_1b_2 - b_1b_3, \\ &\quad a_1b_2 + a_1b_3 + b_1a_2 + b_1a_3). && (1.15)\end{aligned}$$

And that is exactly what we get in the left side of the same equation (equation 1.12). For both operations, we can define their identity. For the additive operation, we have that $(0, 0)$ is the aditive operation, because $\forall c \in \mathbb{C}$ such that $c = (a, b)$

$$c + (0, 0) = (a, b) + (0, 0) = (a + 0, b + 0) = (a, b) = c \tag{1.16}$$

So we have that

$$c + (0, 0) = (0, 0) + c = c \tag{1.17}$$

And for the multiplicative operation, we have that (1,0) is the identity, that is

$$\begin{aligned} c \times (1, 0) &= (a, b) \times (1, 0) \\ &= (a1 - b0, b1 + a0) \\ &= (a, b) = c && (1.18)\end{aligned}$$

So we have that

$$c \times (1, 0) = (1, 0) \times c = c \tag{1.19}$$

Now that we have defined the multiplication and addition operations, therefore we need their complementary. Let start with the complementary operation of the addition. Complementary operation of addition is substraction and it is defined as follow:

$$c_1 - c_2 = (a_1, b_1) - (a_2, b_2) = (a_1 - a_2, b_1 - b_2) \tag{1.20}$$

It is turn to think about the complementary operation of multiplication: division. Our intuition for division of two complex numbers c_1 and c_2 is another complex (x, y) such that

$$(x, y) = \frac{(a_1, b_1)}{(a_2, b_2)} \tag{1.21}$$

So, by definition of division we have

$$\begin{aligned}(a_1, b_1) &= (x, y) \times (a_2, b_2) && (1.22)\\ &= (a_2x - b_2y,\ xb_2 + a_2y) && (1.23)\end{aligned}$$

So we end up with

$$\begin{aligned}a_1 &= a_2x - b_2y && (1.24)\\ b_1 &= a_2y + b_2x && (1.25)\end{aligned}$$

Solving the equation for x and y give us

$$x = \frac{a_1a_2 + b_1b_2}{{a_2}^2 + {b_2}^2} \tag{1.26}$$

$$y = \frac{a_2b_1 - a_1b_2}{{a_2}^2 + {b_2}^2} \tag{1.27}$$

Or in more compact notation

$$\frac{(a_1, b_1)}{(a_2, b_2)} = (\frac{a_1a_2 + b_1b_2}{{a_2}^2 + {b_2}^2},\ \frac{a_2b_1 - a_1b_2}{{a_2}^2 + {b_2}^2}) \tag{1.28}$$

In the real world, we have an unitary operation called absolute value, given by

$$|a| = +\sqrt{a^2} \tag{1.29}$$

We can define a generalization of this operation for the complex space, called the modulus of a complex number, by letting

$$|c| = |a + bi| = +\sqrt{a^2 + b^2} \tag{1.30}$$

Supose that we have two complex numbers, $c_1, c_2 \in \mathbb{C}$, such that $c_1 = (a_1, b_1)$ and $c_2 = (a_2, b_2)$. Let's calculate the multiplication of the modulus:

$$\begin{aligned}|c_1|^2|c_2|^2 &= \sqrt{{a_1}^2 + {b_1}^2}^2 \sqrt{{a_2}^2 + {b_2}^2}^2 && (1.31)\\ &= ({a_1}^2 + {b_1}^2)({a_2}^2 + {b_2}^2) && (1.32)\\ &= {a_1}^2{a_2}^2 + {a_1}^2{b_2}^2 + {b_1}^2{a_2}^2 + {b_1}^2{b_2}^2 && (1.33)\\ &= {a_1}^2{a_2}^2 + {b_1}^2{b_2}^2 - 2a_1a_2b_1b_2 + && (1.34)\\ &\quad +{a_1}^2{b_2}^2 + {b_1}^2{a_2}^2 + 2a_1a_2b_1b_2 && \\ &= (a_1a_2 - b_1b_2)^2 + (a_1b_2 + a_2b_1)^2 && (1.35)\\ &= |c_1c_2|^2 && (1.36)\end{aligned}$$

Now, for the same complex numbers, we have another important property for addition

$$|c_1 + c_2| \leq |c_1| + |c_2| \tag{1.37}$$

To prove this property, let's square the left side of the equation

$$\begin{aligned} |c_1 + c_2|^2 &= |(a_1 + a_2, b_1 + b_2)|^2 && (1.38)\\ &= (a_1 + a_2)^2 + (b_1 + b_2)^2 && (1.39)\\ &= {a_1}^2 + {a_2}^2 + {b_1}^2 + {b_2}^2 + 2(a_1a_2 + b_1b_2) && (1.40) \end{aligned}$$

On the other hand, in the other side of the equation we have

$$\begin{aligned} |c_1|^2 + |c_2|^2 &= \left(\sqrt{{a_1}^2 + {b_1}^2}\right)^2 + \left(\sqrt{{a_2}^2 + {b_2}^2}\right)^2 && (1.41)\\ &= {a_1}^2 + {a_2}^2 + {b_1}^2 + {b_2}^2 && (1.42)\\ &= +2\sqrt{({a_1}^2 + {b_1}^2)({a_2}^2 + {b_2}^2)} && (1.43) \end{aligned}$$

Replacing this equalities in Eq 1.37

$$\begin{aligned} \cancel{{a_1}^2 + {a_2}^2 + {b_1}^2 + {b_2}^2} + 2(a_1a_2 + b_1b_2) &\leq \\ \cancel{{a_1}^2 + {a_2}^2 + {b_1}^2 + {b_2}^2} + 2\sqrt{({a_1}^2 + {b_1}^2)({a_2}^2 + {b_2}^2)} &\rightarrow && (1.44)\\ a_1a_2 + b_1b_2 \leq \sqrt{({a_1}^2 + {b_1}^2)({a_2}^2 + {b_2}^2)} && (1.45) \end{aligned}$$

Squaring both sides of the inequality again

$$\begin{aligned} (a_1a_2 + b_1b_2)^2 &\leq \sqrt{({a_1}^2 + {b_1}^2)({a_2}^2 + {b_2}^2)}^2\\ \cancel{{a_1}^2{a_2}^2 + {b_1}^2{b_2}^2} + 2a_1a_2b_1b_2 &\leq \cancel{{a_1}^2{a_2}^2 + {b_1}^2{b_2}^2} + {a_1}^2{b_2}^2 + {a_2}^2{b_1}^2\\ 2a_1a_2b_1b_2 &\leq {a_1}^2{b_2}^2 + {a_2}^2{b_1}^2\\ {a_1}^2{b_2}^2 + {a_2}^2{b_1}^2 - 2a_1a_2b_1b_2 &\geq 0\\ (a_1b_2 - a_2b_1)^2 &\geq 0 \end{aligned}$$

So, we have proved that the modulus have two important properties, that are (for $c_1, c_2 \in \mathbb{C}$):

1. $|c_1c_2| = |c_1||c_2|$
2. $|c_1 + c_2| \leq |c_1| + |c_2|$

In order to define all the interesting basic operators over $\mathbb{C}$, we need to introduce one last operator: conjugation. Conjugation operator is a function $c \mapsto \overline{c}$ such that, if $c = (a, b)$ then $\overline{c} = (a, -b)$.

Now, as for previous operators, let's check the properties with respect to addition and multiplicative operators. Let's have $c_1, c_2 \in \mathbb{C}$, such that $c_1 = (a_1, b_1)$ and $c_2 = (a_2, b_2)$. That means $\overline{c_1} = (a_1, -b_1)$ and $\overline{c_2} = (a_2, -b_2)$

$$\begin{aligned} \overline{c_1 + c_2} &= \overline{(a_1 + a_2, b_1 + b_2)} && (1.46)\\ &= (a_1 + a_2, -(b_1 + b_2) && (1.47)\\ &= (a_1 + a_2, -b_1 + (-b_2)) && (1.48)\\ &= \overline{c_1} + \overline{c_2} && (1.49) \end{aligned}$$

And, in other hand

$$\begin{aligned} \overline{c_1 c_2} &= \overline{(a_1 a_2 - b_1 b_2, a_1 b_2 + a_2 b_1)} && (1.50)\\ &= (a_1 a_2 - b_1 b_2, -(a_1 b_2 + a_2 b_1)) && (1.51)\\ &= (a_1 a_2 - b_1 b_2, -a_1 b_2 - a_2 b_1)) && (1.52)\\ &= (a_1 a_2 - (-b_1)(-b_2), -a_1 b_2 - a_2 b_1) && (1.53)\\ &= \overline{c_1}\ \overline{c_2} && (1.54) \end{aligned}$$

So we have that the conjugation operators respects addition and multiplication.

1.2 Geometric Interpretation

As we have seen in the previous section, complex numbers can be represented as a pair of real numbers. This suggest a natural means of representations: pairs of real values correspond to points on the plain (or vectors starting at the origin and ending in that point). As far as a complex number $c \in \mathbb{C}$ can be represented as a pair (a, b) with $a, b \in \mathbb{R}$, complex numbers can be represented as vectors in $\mathbb{R}^2$, in wich Y-axis correspond to the imaginary part of the complex number and X-axis correspond to the real part (as it shown in Figure 1.1).

Through this representation, we can give another view of the algebraic properties of the complex numbers. For example, if we think about a sum of the complex numbers as a vector of $\mathbb{R}^2$, we see that the vectors can be added using the so-called parallelogram rule (as it show in Figure 1.2).

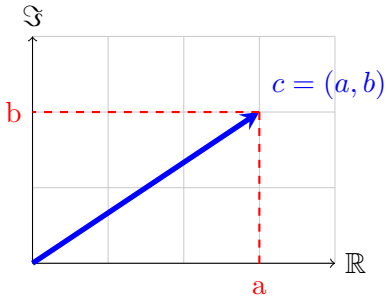

Figure 1.1: Representation in $\mathbb{R}^2$ of complex number.

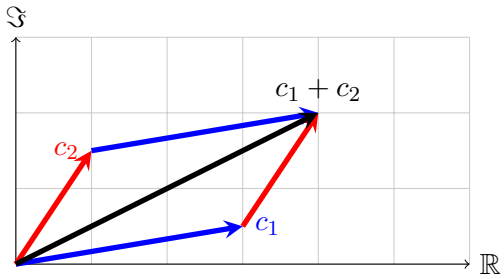

Figure 1.2: Adding two complex with parallelogram rule.

To give a geometrical meaning of multiplication, we need to develop another characterization. As for vectors in real world, we can define the polar representation, in which we have a radius r (for complex numbers, radius is the modulus) and an angulus θ, and they satisfy the next equalities (respect her vector form $c = (a, b)$):

$$r = |c| = +\sqrt{a^2 + b^2} \tag{1.55}$$

$$\theta = \arctan \frac{b}{a} \tag{1.56}$$

And with these two parameters, we can represent any $c \in \mathbb{C}$ as

$$c = re^{i\theta} \tag{1.57}$$

Polar representation yields us an interesting question: How many complex numbers share the same modulus? As you can see in Figure 1.3, there is an entire circle centered at origin formed by complex numbers with the same modulus.

Working with polar coordinates we obtain another definitions for the Multiplication operator. Given two complex numbers $c_1 = (r_1, \theta_1)$ and $c_2 = (r_2, \theta_2)$, multiplication is defined in polar coordinates as follow

$$c_1 c_2 = r_1 e^{i\theta_1} r_2 e^{i\theta_2} = r_1 r_2 e^{i(\theta_1 + \theta_2)} = (r_1 r_2,\ \theta_1 + \theta_2) \tag{1.58}$$

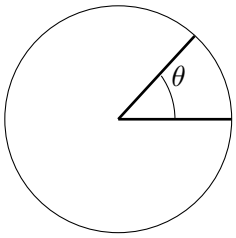

Figure 1.3: Complex numbers with same modulus.

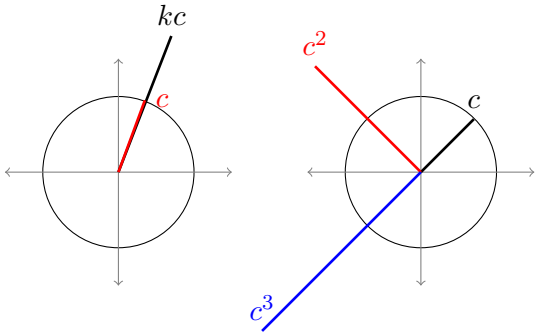

Figure 1.4: Geometric meaning of multiplication.

Multiplication have a geometric interpretation. The interpretation is different if we multiply by a real or complex number as it is shown in Figure 1.4. If we multiply a complex number c by a real number k, then the angulus of the vector does not change, only modulus changes (in order to k). But if we multiply our complex vector by another complex number c_0, then the modulus will be multiplied (so modulus of c will change by modulus of c_0 as in real values) also will change the angulus (in the figure, we use powers of one complex number for better understanding).

If we have defined multiplication, we can also define division. Division is not more than the inverse operation of multiplication. So, assume that we have two complex numbers $c_1 = (r_1, \theta_1)$ and $c_2 = (r_2, \theta_2)$ such that

$$\frac{c_1}{c_2} = c := (r, \theta) \tag{1.59}$$

As well as division is the inverse operation of multiplication, we have

$$(r_1, \theta_1) = c_1 = c_2 c = (r_2, \theta_2) \times (r, \theta) \tag{1.60}$$

$$= (r_2 r, \theta_2 + \theta) \tag{1.61}$$

A deep look into the equality, tell us that

$$r_1 = r_2 r \rightarrow r = \frac{r_1}{r_2} \tag{1.62}$$

$$\theta_1 = \theta_2 + \theta \rightarrow \theta = \theta_1 - \theta_2 \tag{1.63}$$

So, for two complex number c_1 and c_2, division in polar coordinates is defines as follow

$$\frac{c_1}{c_2} = (\frac{r_1}{r_2}, \theta_1 - \theta_2) \tag{1.64}$$

Power operation also can be deduced. It is not hard to see that, for a given complex number $c = (r, \theta)$ and a natural number n, the power operation has the form

$$c^n = (re^{\theta})^n = r^n e^{n\theta} = (r^n, n\theta) \tag{1.65}$$

Let us move to the inverse operation of powers: roots. As you already know, root can be defined as a fraction power, that means, for a given complex number $c = (r, \theta)$ and a natural number n

$$\sqrt[n]{c} = c^{\frac{1}{n}} = (re^{\theta})^{\frac{1}{n}} = r^{\frac{1}{n}} e^{\frac{1}{n}\theta} = \left(r^{\frac{1}{n}}, \frac{1}{n}\theta\right) \tag{1.66}$$

But remember, θ is only defined for multiples of 2π. Therefore, we must rewrite the equation (with $k \in \mathbb{N}$) as

$$\sqrt[n]{c} = \left(r^{\frac{1}{n}}, \frac{1}{n}(\theta + 2k\pi)\right) \tag{1.67}$$

That means that there are several roots of the same complex number. In other words, the solution is not unique. In fact, there are precisely n roots for a complex number, and we can yield every root varying the k between 0 and $n - 1$ (both includes).

The reason because we have n roots for a given complex number is the following: suppose that we have a complex number define as in Eq 1.67, so if we varying k, we have the solutions given in Table 1.1

$k = 0$	$\frac{1}{n}\theta$
$k = 1$	$\frac{1}{n}\theta + \frac{1}{n}2\pi$
$\vdots$	$\vdots$
$k = n - 1$	$\frac{1}{n}\theta + \frac{n-1}{n}2\pi$
$k = n$	$\frac{1}{n}\theta + \frac{n}{n}2\pi = \frac{1}{n}\theta + 2\pi$

Table 1.1: Solutions of complex roots

If we think about the solution shown in Table 1.1, the solution when $k = n$ is $\frac{1}{n}\theta + 2\pi$ but, as we said before, solution is between 0 and 2π so we need to set the solution in our domain, by letting the solution congruent with 2π, and that means

$$\left(\frac{1}{n}\theta + 2\pi\right) \ \% \ 2\pi = \frac{1}{n}\theta \tag{1.68}$$

And equation 1.68 is precisely the solution when $k = 0$, and that means that it is the same solution and there is no more distinc solutions.

When we work with polar coordinates, we have a formula that will prove very handy in many situations. That is the Euler formula and say that

$$e^{i\theta} = \cos\theta + i\sin\theta \tag{1.69}$$

One of the utilities of this formula is given by De Moivre's formula. This formula can be deduced thanks to the Euler formula, and says that

$$\left(e^{\theta i}\right)^n = \cos n\theta + i\sin n\theta \tag{1.70}$$

This formula can be proved by induction. Let's prove the formula for $n = 2$

$$\begin{aligned}
\left(e^{\theta i}\right)^2 &= (\cos\theta + i\sin\theta)^2 && (1.71)\\
&= cos\theta cos\theta - \sin\theta\sin\theta + i(cos\theta\sin\theta + \sin\theta cos\theta) && (1.72)\\
&= cos\theta + \theta + i\sin\theta + \theta && (1.73)\\
&= cos2\theta + i\sin 2\theta && (1.74)
\end{aligned}$$

Suppose that the formula works until term $n-1$, then for the term n, we have that

$$\begin{aligned}
\left(e^{\theta i}\right)^n &= \left(e^{\theta i}\right)^{n-1} e^{\theta i} && (1.75)\\
&= [\cos([n-1]\theta) + i\sin([n-1]\theta)][\cos\theta + i\sin\theta] && (1.76)\\
&= cos([n-1]\theta)cos\theta - \sin([n-1]\theta)\sin\theta && (1.77)\\
&\quad + i[cos\theta\sin([n-1]\theta) + \sin\theta cos([n-1]\theta)]\\
&= cos([n-1]\theta + \theta) + i\sin([n-1]\theta + \theta) && (1.78)\\
&= cosn\theta + i\sin n\theta && (1.79)
\end{aligned}$$

So the formula is proved.

The last thing we are going to introduce to characterize geometrically any function on complex numbers comes with the polynomials. An arbitrary polynomial function with complex coefficients ($c_i \in \mathbb{C}$) looks like

$$P(x) = c_0 + c_1 x + \cdots + c_n x^n \tag{1.80}$$

From polynomials, we can define the rational functions, which are generated by division of polynomial functions:

$$R(x) = \frac{P(x)}{Q(x)} = \frac{c_0 + c_1 x + \cdots + c_n x^n}{d_0 + d_1 m + \cdots + d_m x^m} \tag{1.81}$$

Where $P(x) = c_0 + c_1x + \cdots + c_nx^n$ and $Q(x) = d_0 + d_1m + \cdots + d_mx^m$ are polynomials with $Q(x) \neq 0$. This is a generalization of all types of transformations than we can have. The simplest case of these functions are Möbius transformations, that have the form:

$$R_{a,b,c,d}(x) = \frac{ax+b}{cx+d} \tag{1.82}$$

where $a, b, c, d \in \mathbb{C}$ and $ad - bc \neq 0$. This transformation is an important one. It also have important properties. One of them is that the composition of Möbius transformations is also a Möbius transformation. Let have two Möbius transformations $R_{a,b,c,d}(x)$ and $R_{a',b',c',d'}(x)$, then

$$\begin{aligned}
R_{a',b',c',d'}(R_{a,b,c,d}(x)) &= R_{a',b',c',d'}\left(\frac{ax+b}{cx+d}\right) = \frac{a'\frac{ax+b}{cx+d} + b'}{c'\frac{ax+b}{cx+d} + d'} && (1.83)\\
&= \frac{a'ax + ba' + (cx+d)b'}{c'ax + c'b + (cx+d)d'} && (1.84)\\
&= \frac{a'ax + ba' + cb'x + b'd}{c'ax + c'b + d'cx + dd'} && (1.85)\\
&= \frac{(aa' + b'c)x + (ba' + b'd)}{(c'a + d'c)x + (bc' + dd')} && (1.86)
\end{aligned}$$

This can be rewrited as

$$R_{a'',b'',c'',d''} = \frac{a''x + b''}{c''x + d''} \tag{1.87}$$

with $a'' = aa' + b'c$, $b'' = ba' + b'd$, $c'' = c'a + d'c$ and $d'' = bc' + dd'$. To be a Möbius transformation, it must verify that $a''d'' - b''c'' \neq 0$

$$\begin{aligned}
a''d'' - b''c'' &= (aa' + b'c)(bc' + dd') - (ba' + b'd)(c'a + d'c) && (1.88)\\
&= aa'bc' + aa'dd' + bb'cc' + b'cdd' && (1.89)\\
&\quad -aa'bc' - a'bcd' - ab'c'd - b'cdd'\\
&= aa'dd' + bb'cc' - a'bcd' - ab'c'd && (1.90)\\
&= ad(a'd' - b'c') - bc(a'd' - b'c') && (1.91)\\
&= (ad - bc)(a'd' - b'c') && (1.92)
\end{aligned}$$

But $ad-bc \neq 0$ and $a'd'-b'c' \neq 0$ from hypothesis, so $(ad-bc)(a'd'-b'c') \neq 0$ and $R_{a',b',c',d'}(R_{a,b,c,d}(x))$ is a Möbius transformation.

Another important property of Möbius transformations is that they have an inverse operation which it is also a Möbius transformation. Identity Möbius transformation is $R_{1,0,0,1}$. Let calculate the inverse of $R_{a,b,c,d}(x)$, that means

$$R_{1,0,0,1} = R_{a',b',c',d'}(R_{a,b,c,d}(x)) \tag{1.93}$$

where $R_{a',b',c',d'}$ is the inverse of $R_{a,b,c,d}$. To prove the last property of Möbius transformation, we also prove that

$$\begin{aligned} 1 &= aa^{'} + b^{'}c && (1.94)\\ 0 &= a^{'}b + b^{'}d && (1.95)\\ 0 &= ac^{'} + cd^{'} && (1.96)\\ 1 &= bc^{'} + dd^{'} && (1.97) \end{aligned}$$

From equations 1.95 and 1.96 we can deduce that

$$\begin{aligned} b^{'} &= \frac{a^{'}b}{d} && (1.98)\\ c^{'} &= \frac{-cd^{'}}{a} && (1.99) \end{aligned}$$

Using equation 1.98 in 1.94 and using equation 1.99 in 1.97 we have that

$$\begin{aligned} a^{'} &= \frac{d}{ad-bc} && (1.100)\\ d^{'} &= \frac{a}{ad-bc} && (1.101) \end{aligned}$$

Now, if we use 1.100 in 1.98 and using equation 1.101 in 1.99 we have that

$$\begin{aligned} b^{'} &= \frac{-b}{ad-bc} && (1.102)\\ c^{'} &= \frac{-c}{ad-bc} && (1.103) \end{aligned}$$

So we have calculated the coefficients of the inverse of $R_{a,b,c,d}$, but we need to be sure that it is, in fact, a Möbius transformation.

$$\begin{aligned} a^{'}d^{'} - b^{'}c^{'} &= \frac{d}{ad-bc}\frac{a}{ad-bc} - \frac{-b}{ad-bc}\frac{-c}{ad-bc} && (1.104)\\ &= \frac{ad-bc}{(ad-bc)^2} = \frac{1}{ad-bc} && (1.105) \end{aligned}$$

And from hypothesis, $ad - bc \neq 0$ so $\frac{1}{ad-bc} \neq 0$ and for so the inverse of $R_{a,b,c,d}$ is also a Möbius transformation.

1.3 Definition of Complex Vector Space

In order to define the complex vector space, due to the fact that complex vector space is an extension of real vector space, let start defining the real vector space.

Definition 1.3.1: A *real vector space* is a nonempty set $\mathbb{W}$ along with an addition operation, negation operation and a scalar multiplication which satisfy the next properties ($\forall v, w, y \in \mathbb{W}$ and $\forall x, x_1, x_2 \in \mathbb{R}$):

$$\begin{aligned} v + w &= w + v && (1.106)\\ (v + w) + y &= v + (w + y) && (1.107)\\ v + 0 = v &= 0 + v && (1.108)\\ v + (-v) = 0 &= -v + v && (1.109)\\ 1v &= v && (1.110)\\ x_1(x_2 v) &= (x_1 x_2)v && (1.111)\\ x(v + w) &= xv + xw && (1.112)\\ (x_1 + x_2)v &= x_1 v + x_2 v && (1.113) \end{aligned}$$

Definition 1.3.2: A *complex vector space* is a nonempty set $\mathbb{V}$ along with an addition operation, negation operation and a scalar multiplication which satisfy the analogous properties of the real vector space but with the coefficients in the complex space (that means that the complex space satisfyies the last 8 equations but $\forall v, w, y \in \mathbb{V}$ and $\forall x, x_1, x_2 \in \mathbb{C}$).

As in the real vector space, a usuall example of a complex vector space is the set of matrices. The set of all m-by-n matrices with complex coefficients is denoted by $\mathbb{C}^{m\times n}$. Given two matrices $X, Y \in \mathbb{C}^{m\times n}$, $i = 0, \cdots, m-1$; $j = 0, \cdots, n-1$ and for $k \in \mathbb{C}$, the operations of this set are defined as follows

$$\begin{aligned} (X + Y)[i, j] &= X[i, j] + Y[i, j] && (1.114)\\ (-X)[i, j] &= -(X[i, j]) && (1.115)\\ (kX)[i, j] &= kX[i, j] && (1.116) \end{aligned}$$

where $X[i, j]$ denotes the complex entry in the i-th rowand j-th column. Let us prove that this set is, in fact, a complex vector space. The first 5 properties are trivial, so let prove the last three properties. Let $k, k_1, k_2 \in \mathbb{C}$ and $X, Y \in \mathbb{C}^{m\times n}$. Let start proving that the scalar multiplication respects complex multiplication, that means

$$\begin{aligned} k_1(k_2 X) &\to (k_1(k_2 X))[i, j] && (1.117)\\ &= k_1((k_2 X)[i, j]) && (1.118)\\ &= k_1(k_2 X[i, j]) && (1.119)\\ &= (k_1 k_2)X[i, j] && (1.120)\\ &\to (k_1 k_2)X && (1.121) \end{aligned}$$

So that means that $k_1(k_2X) = (k_1k_2)X$ and the property is proved. Now let move into the next property: scalar multiplication distributes over addition

$$\begin{aligned} k(X+Y) &\rightarrow k(X+Y)[i,j] && (1.122)\\ &= k((X+Y)[i,j]) && (1.123)\\ &= k(X[i,j]+Y[i,j]) && (1.124)\\ &= kX[i,j]+kY[i,j] && (1.125)\\ &\rightarrow kX+kY && (1.126) \end{aligned}$$

It is time to prove the last of the properties: scalar multilication distributes over complex addition,

$$\begin{aligned} (k_1+k_2)X &\rightarrow ((k_1+k_2)X)[i,j] && (1.127)\\ &= (k_1+k_2)X[i,j] && (1.128)\\ &= k_1X[i,j]+k_2X[i,j] && (1.129)\\ &\rightarrow k_1X+k_2X && (1.130) \end{aligned}$$

So, because of $\mathbb{C}^{m\times n}$ verifies all the properties to become a complex vector space for an arbitrary two matrices and 3 complex scalars, $\mathbb{C}^{m\times n}$ is a complex vector space.

So we have proved that $\mathbb{C}^{m\times n}$ with three operations: inverse, addition and scalar multiplication is a complex vector space. However, these are not the only ways to operate two elements of this vector space, in fact, there exists more complicated and useful operators for this set. An example of this new operators is the transpose operator. For $X \in \mathbb{C}^{m\times n}$, the transpose is denoted by X^T and is defined as follows

$$X^T[i,j] = X[j,i] \tag{1.131}$$

This operation is interesting because of the next three properties. For $X, Y \in \mathbb{C}^{m\times n}$ and $k \in \mathbb{C}$, we have the following properties

$$\begin{aligned} \left(X^T\right)^T &= X && (1.132)\\ (X+Y)^T &= X^T+Y^T && (1.133)\\ (kX)^T &= kX^T && (1.134) \end{aligned}$$

Complex numbers have an special operator to switch the sign of the imaginary part: the conjugate. So with this operation in mind, we can define the conjugate of a complex matrix as the conjugate of every element of the matrice.

$$\overline{X}[i,j] = \overline{X[i,j]} \tag{1.135}$$

And this operator has analogous properties to the last one, that means

$$\overline{\overline{X}} = X \tag{1.136}$$
$$\overline{X+Y} = \overline{X}+\overline{Y} \tag{1.137}$$
$$\overline{kX} = \overline{k}\cdot\overline{X} \tag{1.138}$$

We can also combine the two operations defined and as a result we have the adjoint operator. The adjoint operator (or dagger operation) is denoted by $A^\dagger$ and is defined as

$$X^\dagger = \overline{X^T} = \overline{X}^T \tag{1.139}$$

And its properties can be deduced from the properties of the conjugation and transpose operations

$$(X^\dagger)^\dagger = (\overline{X}^T)^\dagger = \overline{(\overline{X}^T)^T} = \overline{\overline{X}} = X \tag{1.140}$$
$$(X+Y)^\dagger = (\overline{X+Y})^T = (\overline{X}+\overline{Y})^T = \overline{X}^T+\overline{Y}^T = X^\dagger+Y^\dagger \tag{1.141}$$
$$(kX)^\dagger = (\overline{kX})^T = (\overline{kX})^T = \overline{kX}^T = \overline{k}X^\dagger \tag{1.142}$$

The last basic operator of $\mathbb{C}^{m\times n}$ is the multiplication. The multiplication is given between two matrices, $X \in \mathbb{C}^{m\times n}$ and $Y \in \mathbb{C}^{n\times p}$, which their multiplication is the matrix $X \times Y \in \mathbb{C}^{m\times p}$ and is defined as follows (with $i = 0, \cdots, m-1$ and $j = 0, \cdots, p-1$):

$$(X \star Y)[i,j] = \sum_{h=0}^{n-1} X[i,h]Y[h,j] \tag{1.143}$$

Multiplication of matrices is a very important operation in the quantum theory, so we need to know the properties of the operator, that are the following. Let $X, Y, Z \in \mathbb{C}^{n\times n}$

1. Associative: $(X \star Y) \star Z = X \star (Y \star Z)$
2. Let $I_n \in \mathbb{C}^{n\times n}$ the identity matrice, $I_n \star A = A = A \star I_n$
3. Distributes over addition

$$X \star (Y+Z) = (X \star Y) + (X \star Z) \tag{1.144}$$
$$(X+Y) \star Z = (X \star Z) + (Y \star Z) \tag{1.145}$$

4. Respects scalar multiplication. Let $c \in \mathbb{C}$

$$c(X \star Y) = (cX) \star Y = X \star (cY) \tag{1.146}$$

In addition, we have three more properties with respect the transpose, conjugate and adjoint operators

$$\begin{aligned} (X \star Y)^T &= Y^T \star X^T && (1.147)\\ \overline{X \star Y} &= \overline{X} \star \overline{Y} && (1.148)\\ (X \star Y)^\dagger &= Y^\dagger \star X^\dagger && (1.149) \end{aligned}$$

The last property is easy to prove from the other two

$$(X \star Y)^\dagger = (\overline{X \star Y})^T = (\overline{X} \star \overline{Y})^T = \overline{Y}^T \star \overline{X}^T = Y^\dagger \star X^\dagger \qquad (1.150)$$

1.4 Other Operators in Complex Vector Spaces

We have defined a set of basic operators over vectors and matrices, but there are other important operators that we can define over this spaces. For example, over a complex vector space $\mathbb{C}^n$, we can define a function

$$\langle -, - \rangle : \mathbb{C}^n \times \mathbb{C}^n \to \mathbb{C} \qquad (1.151)$$

This function is called inner product if it satisfies the following properties: Let $V_1, V_2, V_3 \in \mathbb{C}^n$ and $c \in \mathbb{C}$,

1. Nondegenerate:

$$\begin{aligned} \langle V_1, V_1 \rangle &\geq 0 && (1.152)\\ \langle V_1, V_1 \rangle = 0 &\leftrightarrow V_1 = 0 && (1.153) \end{aligned}$$

2. Respects addition

$$\begin{aligned} \langle V_1 + V_2, V_3 \rangle &= \langle V_1, V_3 \rangle + \langle V_2, V_3 \rangle && (1.154)\\ \langle V_1, V_2 + V_3 \rangle &= \langle V_1, V_2 \rangle + \langle V_1, V_3 \rangle && (1.155) \end{aligned}$$

3. Respects scalar multiplication

$$\begin{aligned} \langle cV_1, V_2 \rangle &= c\langle V_1, V_2 \rangle && (1.156)\\ \langle V_1, cV_2 \rangle &= \bar{c}\langle V_1, V_2 \rangle && (1.157) \end{aligned}$$

4. Skew symmetric

$$\langle V_1, V_2 \rangle = \overline{\langle V_2, V_1 \rangle} \qquad (1.158)$$

Let $V = (v_1, \cdots, v_n), W = (w_1, \cdots, w_n) \in \mathbb{C}^n$. In this space, we can define the inner product as

$$\langle V, W \rangle = W^\dagger \star V = \sum_{i=1}^{n} \overline{w_i} v_i \tag{1.159}$$

To be an inner product, it must satisfies the defined properties, so let's check them. Let $V = (v_1, \cdots, v_n), W = (w_1, \cdots, w_n), U = (u_1, \cdots, u_n) \in \mathbb{C}^n$ and $c \in \mathbb{C}$

1. Nondegenerate:

$$\langle V, V \rangle = \sum_{i=1}^{n} \overline{v_i} v_i = \sum_{i=1}^{n} v_i \overline{v_i} = \sum_{i=1}^{n} |v_i|^2 \tag{1.160}$$

but $|v_i|^2 \geq 0$ So

$$\sum_{i=1}^{n} |v_i|^2 = 0 \rightarrow \forall v_i \in V : v_i = 0 \rightarrow V = 0 \tag{1.161}$$

2. Respects addition

$$\begin{aligned}
\langle V + W, U \rangle &= \langle (v_1 + w_1, \cdots, v_n + w_n), U \rangle && (1.162)\\
&= \sum_{i=1}^{n} \overline{u_i}(v_i + w_i) && (1.163)\\
&= \sum_{i=1}^{n} \overline{u_i} v_i + \overline{u_i} w_i && (1.164)\\
&= \sum_{i=1}^{n} \overline{u_i} v_i + \sum_{i=1}^{n} \overline{u_i} w_i && (1.165)\\
&= \langle V, U \rangle + \langle W, U \rangle && (1.166)
\end{aligned}$$

Alternativelly, we have

$$\begin{aligned}
\langle V, W + U \rangle &= \langle V, (w_1 + u_1, \cdots, w_n + u_n) \rangle && (1.167)\\
&= \sum_{i=1}^{n} \overline{(w_i + u_i)} v_i && (1.168)\\
&= \sum_{i=1}^{n} \overline{w_i} v_i + \overline{u_i} v_i && (1.169)\\
&= \sum_{i=1}^{n} \overline{w_i} v_i + \sum_{i=1}^{n} \overline{u_i} v_i && (1.170)\\
&= \langle V, W \rangle + \langle V, U \rangle && (1.171)
\end{aligned}$$

3. Respects scalar multiplication

$$\langle cV, W\rangle = \sum_{i=1}^{n} \overline{w_i} c v_i = c \sum_{i=1}^{n} \overline{w_i} v_i = c\langle V, W\rangle \tag{1.172}$$

Alternativelly, we have

$$\langle V, cW\rangle = \sum_{i=1}^{n} \overline{c w_i} v_i = \sum_{i=1}^{n} \overline{c}\ \overline{w_i} v_i = \overline{c} \sum_{i=1}^{n} \overline{w_i} v_i = \overline{c}\langle V, W\rangle \tag{1.173}$$

4. Skew symmetric

$$\overline{\langle V, W\rangle} = \overline{\sum_{i=1}^{n} \overline{w_i} v_i} = \sum_{i=1}^{n} \overline{\overline{w_i} v_i} = \sum_{i=1}^{n} w_i \overline{v_i} = \sum_{i=1}^{n} \overline{v_i} w_i = \langle W, V\rangle \tag{1.174}$$

So it is proved that the operator defined in Equation 1.161 is an inner product. With them, we canmake the following definition

Definition 1.4.1: Let $\langle -, -\rangle$ be an inner product and $\mathbb{V}$ be a complex vector space. Then we denote as a complex inner product space to the tuple $(\mathbb{V}, \langle -, -\rangle)$.

Let's define new operators inside the complex inner product space

Definition 1.4.2: Let $(\mathbb{V}, \langle -, -\rangle)$ be a complex inner product space, then we can define a norm or length which is a function $|.| : \mathbb{V} \to \mathbb{R}$ defined as $|V| := \sqrt{\langle V, V\rangle}$ with $V \in \mathbb{V}$

Norm was defined using the inner product so, is it possible that the norm has the same properties as the inner product? let answer this question. Let $V, W \in \mathbb{V}$ and $c \in \mathbb{C}$, then

$$\text{Supose that } |V|^2 = 0 \to \langle V, V\rangle = 0 \to V = 0 \tag{1.175}$$

So norm is nondegenerate.

$$\begin{aligned}
|V + W| &= \sqrt{\langle V + W, V + W\rangle} && (1.176)\\
&= \sqrt{\langle V, V + W\rangle + \langle W, V + W\rangle} && (1.177)\\
&= \sqrt{\langle V, V\rangle + \langle V, W\rangle + \langle W, V\rangle + \langle W, W\rangle} && (1.178)\\
&\leq \sqrt{\langle V, V\rangle + \langle W, W\rangle} = \sqrt{|V|^2 + |W|^2} && (1.179)\\
&\leq |V| + |W| && (1.180)
\end{aligned}$$

So norm respects the triangle inequality.

$$|cV| = \sqrt{\langle cV, cV\rangle} = \sqrt{\overline{c}c\langle V, V\rangle} = \sqrt{\overline{c}c}\sqrt{\langle V, V\rangle} \quad (1.181)$$

$$= \sqrt{\langle c, c\rangle}\sqrt{\langle V, V\rangle} = |c||V| \quad (1.182)$$

So norm respects scalar multiplication. In resume, the norm has the following properties

1. Nondegenerate:

$$|V| \geq 0 \text{ and } |V| = 0 \rightarrow V = 0 \quad (1.183)$$

2. Satisfies the triengle inequality

$$|V + W| \leq |V| + |W| \quad (1.184)$$

3. Respects scalar multiplication

$$|cV| = |c||V| \quad (1.185)$$

Given a norm, we can proceed and define a distance function

Definition 1.4.3: For all complex inner product space $(\mathbb{V}, \langle -, -\rangle)$, we can define a distance function has $d : \mathbb{V} \times \mathbb{V} \rightarrow \mathbb{R}$ such that

$$\forall V_1, V_2 \in \mathbb{V}, \quad d(V_1, V_2) := |V_1 - V_2| = \sqrt{\langle V_1 - V_2, V_1 - V_2\rangle} \quad (1.186)$$

This distance function have the same properties that the norm function, that means, $\forall V_1, V_2, V_3 \in \mathbb{V}$, distance satisfies the next properties

1. Nondegenerate: $d(V_1, V_2) = 0$ if and only if $V_1 = V_2$
   Proof:

$$d(V_1, V_2) = 0 \rightarrow |V_1 - V_2| = 0 \rightarrow V_1 - V_2 = 0 \rightarrow V_1 = V_2 \quad (1.187)$$

2. Satisfies the triengle inequality: $d(V_1, V_3) \leq d(V_1, V_2) + d(V_2, V_3)$
   Proof:

$$d(V_1, V_3) = |V_1 - V_3| = |V_1 - V_2 + V_2 - V_3| \quad (1.188)$$

$$\leq |V_1 - V_2| + |V_2 - V_3| \quad (1.189)$$

$$= d(V_1, V_2) + d(V_2, V_3) \quad (1.190)$$

3. Distance is symmetric: $d(V_1, V_2) = d(V_2, V_1)$
   Proof:

$$\begin{aligned} d(V_1, V_2) &= \langle V_1 - V_2, V_1 - V_2 \rangle && (1.191)\\ &= \langle V_1, V_1 - V_2 \rangle - \langle V_2, V_1 - V_2 \rangle && (1.192)\\ &= -\langle V_1, V_2 - V_1 \rangle + \langle V_2, V_2 - V_1 \rangle && (1.193)\\ &= \langle V_2 - V_1, V_2 - V_1 \rangle = d(V_2, V_1) && (1.194) \end{aligned}$$

Cartesian product is not the only operator to combine vector spaces. Tensor product is an important operator because, if we have two spaces, A and B, that describes two quantum systems, then their tensor product describes both quantum systems as one.

Definition 1.4.4: Given two vector spaces V and W, the tensor product is defined as $V \otimes W := \{v \otimes w \setminus v \in V,\ w \in W\}$. The elements of $V \otimes W$ looks like $c_0(v_0 \otimes w_0) + \cdots + c_{p-1}(v_{p-1} \otimes w_{p-1})$ with $v_i \in V$, $w_i \in W$, $c_i \in \mathbb{C}$, where $\otimes$ is just a symbol.

Another usefull notation for tensor product is

$$V \otimes W = \sum_{i=0}^{p-1} c_i (v_i \otimes w_i) \tag{1.195}$$

With this alternative notation, we can define easily the addition of tensor products and the scalar multiplication as follows

$$(V \otimes W) + (X \otimes Y) = \sum_{i=0}^{p-1} c_i (v_i \otimes w_i) + \sum_{j=0}^{q-1} k_j (x_j \otimes y_j) \tag{1.196}$$

$$k(V \otimes W) = k \sum_{i=0}^{p-1} c_i (v_i \otimes w_i) = \sum_{i=0}^{p-1} (k c_i)(v_i \otimes w_i) \tag{1.197}$$

The tensor product is given as follows: suppose that we have matrices A and B with A in the form

$$A = \begin{bmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{bmatrix} \tag{1.198}$$

$$B = \begin{bmatrix} b_{1,1} & \cdots & b_{1,t} \\ \vdots & \ddots & \vdots \\ b_{p,1} & \cdots & b_{p,t} \end{bmatrix} \tag{1.199}$$

Then, we can write the tensor product of A and B as

$$A \otimes B = \begin{bmatrix} a_{1,1} \cdot B & \cdots & a_{1,n} \cdot B \\ \vdots & \ddots & \vdots \\ a_{m,1} \cdot B & \cdots & a_{m,n} \cdot B \end{bmatrix} = \begin{bmatrix} a_{1,1}b_{1,1} & \cdots & a_{1,1}b_{1,t} & a_{1,2}b_{1,1} & \cdots & a_{1,n}b_{1,t} \\ \vdots & & \vdots & \vdots & & \vdots \\ a_{1,1}b_{p,1} & \cdots & a_{1,1}b_{p,t} & a_{1,2}b_{p,1} & \cdots & a_{1,n}b_{p,t} \\ a_{2,1}b_{1,1} & \cdots & a_{2,1}b_{1,t} & a_{2,2}b_{1,1} & \cdots & a_{2,n}b_{1,t} \\ \vdots & & \vdots & \vdots & & \vdots \\ a_{m,1}b_{1,1} & \cdots & a_{m,1}b_{p,t} & a_{m,2}b_{p,1} & \cdots & a_{m,n}b_{p,t} \end{bmatrix} \tag{1.200}$$

Tensor product must respect addition in both spaces and scalar multiplication, that means (for $v_1, v_2 \in V$, $w_1, w_2 \in W$ and $c \in \mathbb{C}$).

$$(v_1 + v_2) \otimes w_1 = (v_1 \otimes w_1) + (v_2 \otimes w_1) \tag{1.201}$$
$$v_1 \otimes (w_1 + w_2) = (v_1 \otimes w_1) + (v_1 \otimes w_2) \tag{1.202}$$
$$c(v_1 \otimes w_1) = (cv_1) \otimes w_1 = v_1 \otimes (cw_1) \tag{1.203}$$

Let see an example of tensor product. Let

$$A = \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix} \tag{1.204}$$
$$B = \begin{bmatrix} 0 & 1 & 2 \\ 2 & 1 & 0 \end{bmatrix} \tag{1.205}$$

Then

$$A \otimes B = \begin{bmatrix} 1\cdot 0 & 1\cdot 1 & 1\cdot 2 & 2\cdot 0 & 2\cdot 1 & 2\cdot 2 \\ 1\cdot 2 & 1\cdot 1 & 1\cdot 0 & 2\cdot 2 & 2\cdot 1 & 2\cdot 0 \\ 0\cdot 0 & 0\cdot 1 & 0\cdot 2 & -1\cdot 0 & -1\cdot 1 & -1\cdot 2 \\ 0\cdot 2 & 0\cdot 1 & 0\cdot 0 & -1\cdot 2 & -1\cdot 1 & -1\cdot 0 \end{bmatrix} \tag{1.206}$$

$$= \begin{bmatrix} 0 & 1 & 2 & 0 & 1 & 4 \\ 2 & 1 & 0 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & -1 & -2 \\ 0 & 0 & 0 & -2 & -1 & 0 \end{bmatrix} \tag{1.207}$$

$$B \otimes A = \begin{bmatrix} 0\cdot \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix} & 1\cdot \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix} & 2\cdot \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix} \\ 2\cdot \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix} & 1\cdot \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix} & 0\cdot \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix} \end{bmatrix} \tag{1.208}$$

$$= \begin{bmatrix} 0 & 0 & 1 & 2 & 2 & 4 \\ 0 & 0 & 0 & -1 & 0 & -2 \\ 2 & 4 & 1 & 2 & 0 & 0 \\ 2 & -2 & 0 & -1 & 0 & 0 \end{bmatrix} \tag{1.209}$$

Properties of the tensor product:

1. Associative. Given arbitrary matrices A, B and C, then

$$A \otimes (B \otimes C) = (A \otimes B) \otimes C \tag{1.210}$$

Proof:
Let matrices A, B and C with the form

$$A = \begin{bmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{bmatrix} \tag{1.211}$$

$$B = \begin{bmatrix} b_{1,1} & \cdots & b_{1,p} \\ \vdots & \ddots & \vdots \\ b_{q,1} & \cdots & b_{q,p} \end{bmatrix} \tag{1.212}$$

So, we have that

$$
\begin{aligned}
(A\otimes B)\otimes C &= \begin{bmatrix} a_{1,1}b_{1,1} & \cdots & a_{1,1}b_{1,p} & \cdots & a_{1,n}b_{1,1} & \cdots & a_{1,n}b_{1,p} \\ \vdots & \ddots & \vdots & & \vdots & \ddots & \vdots \\ a_{1,1}b_{q,1} & \cdots & a_{1,1}b_{q,p} & \cdots & a_{1,n}b_{q,1} & \cdots & a_{1,n}b_{q,p} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a_{m,1}b_{1,1} & \cdots & a_{m,1}b_{1,p} & \cdots & a_{m,n}b_{1,1} & \cdots & a_{m,n}b_{1,p} \\ \vdots & \ddots & \vdots & & \vdots & \ddots & \vdots \\ a_{m,1}b_{q,1} & \cdots & a_{m,1}b_{q,p} & \cdots & a_{m,n}b_{q,1} & \cdots & a_{m,n}b_{q,p} \end{bmatrix} \otimes C \\
&= \begin{bmatrix} a_{1,1}b_{1,1}\cdot C & \cdots & a_{1,1}b_{1,p}\cdot C & \cdots & a_{1,n}b_{1,1}\cdot C & \cdots & a_{1,n}b_{1,p}\cdot C \\ \vdots & \ddots & \vdots & & \vdots & \ddots & \vdots \\ a_{1,1}b_{q,1}\cdot C & \cdots & a_{1,1}b_{q,p}\cdot C & \cdots & a_{1,n}b_{q,1}\cdot C & \cdots & a_{1,n}b_{q,p}\cdot C \\ \vdots & & \vdots & & \vdots & & \vdots \\ a_{m,1}b_{1,1}\cdot C & \cdots & a_{m,1}b_{1,p}\cdot C & \cdots & a_{m,n}b_{1,1}\cdot C & \cdots & a_{m,n}b_{1,p}\cdot C \\ \vdots & \ddots & \vdots & & \vdots & \ddots & \vdots \\ a_{m,1}b_{q,1}\cdot C & \cdots & a_{m,1}b_{q,p}\cdot C & \cdots & a_{m,n}b_{q,1}\cdot C & \cdots & a_{m,n}b_{q,p}\cdot C \end{bmatrix} \\
&= A\otimes \begin{bmatrix} b_{1,1}\cdot C & \cdots & b_{1,p}\cdot C \\ \vdots & \ddots & \vdots \\ b_{q,1}\cdot C & \cdots & b_{q,p}\cdot C \end{bmatrix} = A\otimes(B\otimes C) \qquad (1.213)
\end{aligned}
$$

2. Adjoint of a tensor product of matrices

$$(A\otimes B)^\dagger = A^\dagger \otimes B^\dagger \qquad (1.214)$$

Proof:
Let matrices A and B with the form

$$A = \begin{bmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{bmatrix} \qquad (1.215)$$

$$B = \begin{bmatrix} b_{1,1} & \cdots & b_{1,p} \\ \vdots & \ddots & \vdots \\ b_{q,1} & \cdots & b_{q,p} \end{bmatrix} \qquad (1.216)$$

Then, we have that

$$A^\dagger = \begin{bmatrix} \overline{a_{1,1}} & \cdots & \overline{a_{m,1}} \\ \vdots & \ddots & \vdots \\ \overline{a_{1,n}} & \cdots & \overline{a_{m,n}} \end{bmatrix} \qquad (1.217)$$

$$B^\dagger = \begin{bmatrix} \overline{b_{1,1}} & \cdots & \overline{b_{q,1}} \\ \vdots & \ddots & \vdots \\ \overline{b_{1,p}} & \cdots & \overline{b_{q,p}} \end{bmatrix} \qquad (1.218)$$

So

$$A^\dagger \otimes B^\dagger = \begin{bmatrix} \overline{a_{1,1}} \cdot B^\dagger & \cdots & \overline{a_{m,1}} \cdot B^\dagger \\ \vdots & \ddots & \vdots \\ \overline{a_{1,n}} \cdot B^\dagger & \cdots & \overline{a_{m,n}} \cdot B^\dagger \end{bmatrix} \tag{1.219}$$

$$= \begin{bmatrix} \overline{a_{1,1}} \cdot B^\dagger & \cdots & \overline{a_{m,1}} \cdot B^\dagger \\ \vdots & \ddots & \vdots \\ \overline{a_{1,n}} \cdot B^\dagger & \cdots & \overline{a_{m,n}} \cdot B^\dagger \end{bmatrix}^{\dagger\dagger} \tag{1.220}$$

$$= \begin{bmatrix} \overline{\overline{a_{1,1}} \cdot B^{\dagger T}} & \cdots & \overline{\overline{a_{1,n}} \cdot B^{\dagger T}} \\ \vdots & \ddots & \vdots \\ \overline{\overline{a_{m,1}} \cdot B^{\dagger T}} & \cdots & \overline{\overline{a_{m,n}} \cdot B^{\dagger T}} \end{bmatrix}^{\dagger} \tag{1.221}$$

$$= \begin{bmatrix} a_{1,1} \cdot \overline{B^{\dagger T}} & \cdots & a_{1,n} \cdot \overline{B^{\dagger T}} \\ \vdots & \ddots & \vdots \\ a_{m,1} \cdot \overline{B^{\dagger T}} & \cdots & a_{m,n}\overline{B^{\dagger T}} \end{bmatrix}^{\dagger} \tag{1.222}$$

$$= \begin{bmatrix} a_{1,1} \cdot B^{\dagger\dagger} & \cdots & a_{1,n} \cdot B^{\dagger\dagger} \\ \vdots & \ddots & \vdots \\ a_{m,1} \cdot B^{\dagger\dagger} & \cdots & a_{m,n}B^{\dagger\dagger} \end{bmatrix}^{\dagger} \tag{1.223}$$

$$= \begin{bmatrix} a_{1,1} \cdot B & \cdots & a_{1,n} \cdot B \\ \vdots & \ddots & \vdots \\ a_{m,1} \cdot B & \cdots & a_{m,n}B \end{bmatrix}^{\dagger} = (A \otimes B)^\dagger \tag{1.224}$$

3. Tensor product of inner products. Let $A, A^{'}, B$ and $B^{'}$ matrices of the appropiate size, then

$$(A \times A^{'}) \otimes (B \times B^{'}) = (A \otimes B) \times (A^{'} \otimes B^{'}) \tag{1.225}$$

Proof:
Let

$$A = \begin{bmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{bmatrix} \tag{1.226}$$

$$A^{'} = \begin{bmatrix} a^{'}_{1,1} & \cdots & a^{'}_{1,n} \\ \vdots & \ddots & \vdots \\ a^{'}_{m,1} & \cdots & a^{'}_{m,n} \end{bmatrix} \tag{1.227}$$

So, we have that

$$A \times A^{'} = \begin{bmatrix} \sum_{i=0}^{n} a_{1,i}a_{1,i}^{'} & \cdots & \sum_{i=0}^{n} a_{1,i}a_{m,i}^{'} \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{n} a_{m,i}a_{1,i}^{'} & \cdots & \sum_{i=0}^{n} a_{m,i}a_{m,i}^{'} \end{bmatrix} \tag{1.228}$$

$$(A \times A^{'}) \otimes (B \times B^{'}) = \begin{bmatrix} \sum_{i=0}^{n} a_{1,i}a_{1,i}^{'} \cdot (B \times B^{'}) & \cdots & \sum_{i=0}^{n} a_{1,i}a_{m,i}^{'} \cdot (B \times B^{'}) \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{n} a_{m,i}a_{1,i}^{'} \cdot (B \times B^{'}) & \cdots & \sum_{i=0}^{n} a_{m,i}a_{m,i}^{'} \cdot (B \times B^{'}) \end{bmatrix}$$

So, if we look at the other part of que equation, we see that

$$(A \otimes B) \times (A^{'} \otimes B^{'}) = \begin{bmatrix} a_{1,1} \cdot B & \cdots & a_{1,n} \cdot B \\ \vdots & \ddots & \vdots \\ a_{m,1} \cdot B & \cdots & a_{m,n} \cdot B \end{bmatrix} \times \begin{bmatrix} a_{1,1}^{'} \cdot B^{'} & \cdots & a_{1,n}^{'} \cdot B^{'} \\ \vdots & \ddots & \vdots \\ a_{m,1}^{'} \cdot B^{'} & \cdots & a_{m,n}^{'} \cdot B^{'} \end{bmatrix} \tag{1.229}$$

$$= \begin{bmatrix} \sum_{i=0}^{n}(a_{1,i} \cdot B) \times (a_{1,i}^{'} \cdot B^{'}) & \cdots & \sum_{i=0}^{n}(a_{1,i} \cdot B) \times (a_{m,i}^{'} \cdot B^{'}) \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{n}(a_{m,i} \cdot B) \times (a_{1,i}^{'} \cdot B^{'}) & \cdots & \sum_{i=0}^{n}(a_{m,i} \cdot B) \times (a_{m,i}^{'} \cdot B^{'}) \end{bmatrix}$$

But, as long as $a_{i,j}, a_{i,j}^{'} \in \mathbb{C}\ \forall a_{i,j} \in A$ and $\forall a_{i,j}^{'} \in A^{'}$, then

$$(A \otimes B) \times (A^{'} \otimes B^{'}) = \begin{bmatrix} \sum_{i=0}^{n}(a_{1,i} \cdot B) \times (a_{1,i}^{'} \cdot B^{'}) & \cdots & \sum_{i=0}^{n}(a_{1,i} \cdot B) \times (a_{m,i}^{'} \cdot B^{'}) \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{n}(a_{m,i} \cdot B) \times (a_{1,i}^{'} \cdot B^{'}) & \cdots & \sum_{i=0}^{n}(a_{m,i} \cdot B) \times (a_{m,i}^{'} \cdot B^{'}) \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{i=0}^{n} a_{1,i}a_{1,i}^{'} \cdot (B \times B^{'}) & \cdots & \sum_{i=0}^{n} a_{1,i}a_{m,i}^{'} \cdot (B \times B^{'}) \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{n} a_{m,i}a_{1,i}^{'} \cdot (B \times B^{'}) & \cdots & \sum_{i=0}^{n} a_{m,i}a_{m,i}^{'} \cdot (B \times B^{'}) \end{bmatrix} \tag{1.230}$$

$$= (A \times A^{'}) \otimes (B \times B^{'}) \tag{1.231}$$

As an example of complex vector space we have polynomials of degree n or less. Polynomials have the form

$$P(x) = c_0 + c_1 x + \cdots + c_n x^n \tag{1.232}$$

With $c_i \in \mathbb{C}\ \forall c_i$. Another useful notation for polinomials is

$$P(x) = \sum_{i=0}^{n} c_i x^i \tag{1.233}$$

For completeness, let us go through the operations. Let $k \in \mathbb{C}$ and let $P(x), Q(x) \in Poly_n$ with $P(x) = \sum_{i=0}^{n} c_i x^i$ and $Q(x) = \sum_{i=0}^{n} d_i x^i$

1. Addition

$$\begin{aligned} P(x)+Q(x) &= \sum_{i=0}^{n} c_i x^i + \sum_{i=0}^{n} d_i x^i && (1.234)\\ &= \sum_{i=0}^{n} (c_i + d_i) x^i && (1.235) \end{aligned}$$

2. Negation

$$-P(x) = \sum_{i=0}^{n} -c_i x^i \tag{1.236}$$

3. Scalar multiplication

$$kP(x) = \sum_{i=0}^{n} k c_i x^i = k \sum_{i=0}^{n} c_i x^i \tag{1.237}$$

With these operations, $Poly_n$ forms a complex vector space. In other words, it satisfies the following properties. Let $k_1, k_2 \in C$ and let $P(x), Q(x), R(x) \in Poly_n$ with $P(x) = \sum_{i=0}^{n} c_i x^i$, $Q(x) = \sum_{i=0}^{n} d_i x^i$ and $R(x) = \sum_{i=0}^{n} e_i x^i$

1. Commutativity

$$\begin{aligned} P(x)+Q(x) &= \sum_{i=0}^{n} (c_i + d_i) x^i = \sum_{i=0}^{n} (d_i + c_i) x^i \\ &= Q(x) + P(x) && (1.238) \end{aligned}$$

2. Associativity

$$\begin{aligned} (P(x)+Q(x))+R(x) &= \sum_{i=0}^{n} (c_i + d_i) x^i + R(x) && (1.239)\\ &= \sum_{i=0}^{n} ([c_i + d_i] + e_i) x^i && (1.240)\\ &= \sum_{i=0}^{n} (c_i + [d_i + e_i]) x^i && (1.241)\\ &= P(x) + \sum_{i=0}^{n} (d_i + e_i) x^i && (1.242)\\ &= P(x) + (Q(x) + R(x)) && (1.243) \end{aligned}$$

3. Zero is identity. Let $0 \in \mathbb{C}$

$$P(x) + 0 = \sum_{i=0}^{n} c_i x^i + \sum_{i=0}^{n} 0 x^i = \sum_{i=0}^{n} c_i x^i = P(x) \tag{1.244}$$

4. Inverse

$$P(x) + (-P(x)) = \sum_{i=0}^{n} c_i x^i + \sum_{i=0}^{n} -c_i x^i = 0 \tag{1.245}$$

5. Multiplicative identity. Let $1 \in \mathbb{C}$

$$1P(x) = \sum_{i=0}^{n} 1 c_i x^i = \sum_{i=0}^{n} c_i x^i = P(x) \tag{1.246}$$

6. Respect complex multiplication

$$k_1(k_2 P(x)) = k_1(\sum_{i=0}^{n} k_2 c_i x^i) = \sum_{i=0}^{n} k_1 k_2 c_i x^i \tag{1.247}$$

$$= k_2(\sum_{i=0}^{n} k_1 c_i x^i) = k_2(k_1 P(x)) \tag{1.248}$$

1.5 Hilbert Spaces and Hermitian Matrices

Hilbert spaces are widely used due to their comfortable properties. To get their definition, first we need to define two concepts

Definition 1.5.2: Let $(\mathbb{V}, \langle -, - \rangle)$ an inner product space with the derived norm and distance function. Therefore, a Cauchy sequence is a sequence of vectors $V_0, V_1, \cdots \in \mathbb{V}$ such that $\forall \epsilon > 0$, there exists an $N_0 \in \mathbb{N}$ such that

$$\forall m, n \geq N_0, \quad d(V_m, V_n) \leq \epsilon \tag{1.249}$$

Definition 1.5.3: Let a complex vector space $\mathbb{V}$, then $\mathbb{V}$ is complete if for any Cauchy sequence of vectors $V_0, V_1, \cdots \in \mathbb{V}$, exists $L \in \mathbb{V}$ such that

$$\lim_{n \to \infty} |V_n - L| = 0 \tag{1.250}$$

The intuition behind this is that a vector space with an inner product is complete if any sequence accumulating somewhere converges to a point. With these definitions, we are in position of defining the Hilbert space.

Definition 1.5.4: A Hilbert space is a complex inner product space that is complete.

Eigenvectors and eigenvalues are the base of the Quantum Mechanics and for so it is important to keep in mind its definition

Definition 1.5.5: Let a matrix $A \in \mathbb{C}^{n\times n}$. If there exists $c \in \mathbb{C}$ and $v \neq 0 \in \mathbb{C}^n$ such that $Av = cv$ then c is called a eigenvalue of A and v an eigenvector of A associated with c.

Proposition: Every eigenvector determines a complex vector subspace of the vector space and it is known as eigenspace associated with the given eigenvector

Proof Let the matrix A with eigenvalue e and with eigenvector associated v. To be a complex vector subspace, we must prove that the space is closed under addition and scalar multiplication.

1. Closed under addition
   Let $e^{'}$ and $e^{''}$ eigenvalue for the eigenvector v, then

$$\begin{aligned} A(e^{''}v + e^{'}v) &= Ae^{''}v + Ae^{'}v = e^{''}Av + e^{'}Av \qquad (1.251)\\ &= e^{''}(ev) + e^{'}(ev) = e(e^{''} + e^{'})v \qquad (1.252)\end{aligned}$$

2. Closed under scalar multiplication
   $\forall c \in \mathbb{C}$ we have that

$$A(cv) = cAv = ceV = e(cv) \qquad (1.253)$$

We shall need certain types of important square matrices and their properties. Let us generalize this notion from to the complex numbers

Definition 1.5.6: Let a matrix $A \in \mathbb{C}^{n\times n}$. If $A^{\dagger} = A$, we say that A is hermitian. In other words, A is hermitian if $A[i,j] = \overline{A[j,i]}$ for $i, j = 1, \cdots, n$

Definition 1.5.7: Let a hermitian matrix $A \in \mathbb{C}^{n\times n}$. Then, the operator that it represents is called self-adjoint.

Let's develop the definition 1.5.6

$$A^{\dagger} = A \to \overline{A^T} = A \to \overline{\overline{A^T}} = \overline{A} \to A^T = \overline{A} \qquad (1.254)$$

It follows that A is Hermitian if and only if $A^T = \overline{A}$

Let see some properties of the hermitian matrices.

Proposition: Let $A \in \mathbb{C}^{n\times n}$ be a hermitian matrix. $\forall v, v^{'} \in \mathbb{C}^n$ we have that

$$\langle Av, v^{'}\rangle = \langle v, Av^{'}\rangle \tag{1.255}$$

Proof

$$\langle Av, v^{'}\rangle = (Av)^{\dagger}v^{'} = v^{\dagger}A^{\dagger}v^{'} = v^{\dagger}Av^{'} = v^{\dagger}(Av^{'}) = \langle v, Av^{'}\rangle \tag{1.256}$$

Proposition: Let $A \in \mathbb{C}^{n\times n}$ be a hermitian matrix. Then all its eigenvalues are real.

Proof: Let $A \in \mathbb{C}^{n\times n}$ be a hermitian matrix with an eigenvalue $c \in \mathbb{C}$ and an eigenvector $v \in \mathbb{C}^n$, then from the properties of the inner product we have

$$c\langle v, v\rangle = \langle cv, v\rangle \tag{1.257}$$

But from the properties of the definition of eigenvalue we have

$$\langle cv, v\rangle = \langle Av, v\rangle \tag{1.258}$$

And from the last propostion

$$\langle Av, v\rangle = \langle v, Av\rangle \tag{1.259}$$

Finally, using again the definition of eigenvalue and the properties of inner product

$$\langle v, Av\rangle = \langle v, cv\rangle = \overline{c}\langle v, v\rangle \tag{1.260}$$

So we end up with the following equality

$$c\langle v, v\rangle = \overline{c}\langle v, v\rangle \rightarrow c = \overline{c} \tag{1.261}$$

So c must be real

Proposition: For a given hermitian matrix, distinc eigenvectors that have distinct eigenvalues are orthogonal for a given hermitian matrix.

Proof: Let $A \in \mathbb{C}^{n\times n}$ be a hermitian matrix with an eigenvalues $c_1, c_2 \in \mathbb{C}$ and an eigenvectors $v_1, v_2 \in \mathbb{C}^n$ such that

$$Av_1 = c_1v_1 \quad Av_2 = c_2v_2 \quad c_1 \neq c_2 \tag{1.262}$$

then, using the properties from the inner product and the definition of the eigenvectors we have the following equalities

$$c_1\langle v_1, v_2\rangle = \langle c_1v_1, v_2\rangle = \langle Av_1, v_2\rangle \tag{1.263}$$

since A is hermitian, we have that

$$\langle Av_1, v_2\rangle = \langle v_1, Av_2\rangle \tag{1.264}$$

again we have this equality from the properties of inner product and definition of eigenvectors

$$\langle v_1, Av_2\rangle = \langle v_1, c_2v_2\rangle = \overline{c_2}\langle v_1, v_2\rangle \tag{1.265}$$

finally, from the last proposition we know that c_2 is real, so

$$\overline{c_2}\langle v_1, v_2\rangle = c_2\langle v_1, v_2\rangle \tag{1.266}$$

so we end up with

$$c_1\langle v_1, v_2\rangle = c_2\langle v_1, v_2\rangle \rightarrow c_1\langle v_1, v_2\rangle - c_2\langle v_1, v_2\rangle = 0 \rightarrow (c_1 - c_2)\langle v_1, v_2\rangle = 0 \tag{1.267}$$

but from hypotesis, $c_1 \neq c_2$, so $\langle v_1, v_2\rangle = 0$ and they are orthogonal.

Definition 1.5.8: Let matrix $A \in \mathbb{C}^{n\times n}$ such that

$$AA^\dagger = A^\dagger A = Id_n \tag{1.268}$$

then A is called unitary matrix

Unitary matrices have also important properties that we need to know. Let's see some of them

Proposition: Let $U, V \in \mathbb{C}^{n\times n}$ be a unitary matrices, then UV is a unitary matrix.

Proof: U and V are unitary so $UU^\dagger = Id_n = VV^\dagger$ and for so, the next equalities follow

$$Id_n = UU^\dagger = UId_nU^\dagger = UVV^\dagger U^\dagger = (UV)(VU)^\dagger = (UV)(UV)^\dagger \tag{1.269}$$

and that is the definition of being unitary matrix, so UV is a unitary matrix.

Proposition: Let $U \in \mathbb{C}^{n\times n}$ be a unitary matrix. Then $\forall V, W \in \mathbb{C}^n$ we have $\langle UV, UW\rangle = \langle V, W\rangle$.

Proof:

$$\langle UV, UW\rangle = (UV)^\dagger(UW) = V^\dagger(U^\dagger U)W = V^\dagger Id_nW = V^\dagger W = \langle V, W\rangle \tag{1.270}$$

Proposition: Let $U \in \mathbb{C}^{n\times n}$ be a unitary matrix. Then $\forall V, W \in \mathbb{C}^n$ we have $d(UV, UW) = d(V, W)$.

Proof: Let consider $d(UV, UW)^2$. From definition, we have that

$$d(UV, UW)^2 = |UV - UW|^2 = \langle UV - UW, UV - UW\rangle \tag{1.271}$$

and from the properties of inner product, we have

$$\begin{aligned}
\langle UV - UW, UV - UW\rangle &= \langle UV, UV - UW\rangle - \langle UW, UV - UW\rangle \\
&= (\langle UV, UV\rangle - \langle UV, UW\rangle) - (\langle UW, UV\rangle - \langle UW, UW\rangle) \\
&= \langle V, V\rangle - \langle V, W\rangle - \langle W, V\rangle + \langle W, W\rangle \\
&= \langle V, V - W\rangle - \langle W, V - W\rangle \\
&= \langle V - W, V - W\rangle = |V - W|^2 = d(V, W)^2
\end{aligned} \tag{1.272}$$

And for so, we end up with

$$d(UV, UW)^2 = \langle UV - UW, UV - UW\rangle = d(V, W)^2 \rightarrow d(UV, UW) = d(V, W) \tag{1.273}$$

2

Introduction to Quantum Theory

Before studing Quantum Algorithms and how to program them, we must study the basic principles in wich Quantum Mechanics is based in. Our goal is to understand the properties that we are going to use and how to exploit them to gain more power with respect to the classical systems. Hence, first we study the differences between the classical systems and Quantum Systems by using graphs. Next, we will go deeper in the undestarding of Quantum Mechanics so we can explain the Quantum Architecture in which our program languages are based in order to program a Quantum Computer. Finally, we will see some well known tools to programs and execute quantum algorithms. This chapter is based in [2] and [3].

2.1 Classical Systems vs Quantum Systems

Consider a system described by a graph in which we place some identical flints on the vertices. The state of the system is represented by the number of flints located in each vertex. To describe completely our system, we need to know how the system evolves over time. In other words, we need to know its dynamics.

Our graph is directed, indicating where the flints are moving. We are considerating a non-probabilistical system, so we do not permit an arbitrary graph. The graphs than can represent these systems are those which have exactly one outgoing edge. If the edge points to the same flint, it is not

moving. Next we show an example matrix representing a valid graph

$$G = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \tag{2.1}$$

Where $G[i, j] = 1$ indicates that exists a directed edge from vertex j to vertext 0 and $G[i, j] = 0$ that it does not exist. The matrix corresponds to the graph shown in Figure 2.1

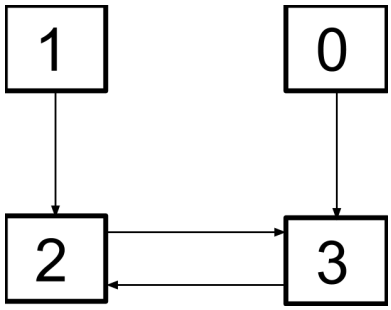

Figure 2.1: Classical Non-Probabilistic System Graph.

Suposse that we have a vector X that describes the state of our system at $t = 0$. Then, GX describes the state of our system one time step later $(t = 1)$. Let's see an example. Let $X = [2, 5, 3, 1]^T$

$$GX = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 5 \\ 3 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 6 \\ 5 \end{bmatrix} \tag{2.2}$$

In general, if we have a vector X that describes the state of our system at $t = 0$ and being G, the matrix that represents the dynamics of our system, then $G^n X$ describes the state of our system at $t = n$.

In quantum mechanics, there exists an inherent indeterminacy in our knowledge about the system. Our dynamics are governed by a probabilistic behaviour. Let's modify our previous definition to make room for this new paradigm. Now there can have more than 1 one arrow from each vertex, but the sum of all these arrows must be 1, because the global probability of

moving to some place must be one. Let see an example

$$G = \begin{bmatrix} 0 & 0.3 & 0.3 \\ 1 & 0 & 0.3 \\ 0 & 0.7 & 0.4 \end{bmatrix} \tag{2.3}$$

Where $G[i, j]$ indicates the probability of going from vertex j to vertex i. G corresponds to the graph show in Figure 2.2

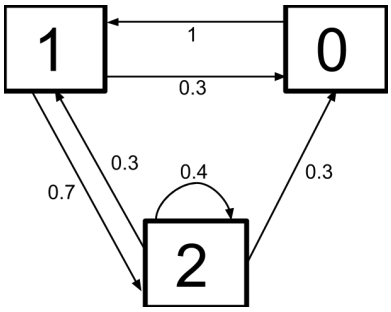

Figure 2.2: Probabilistic System Graph.

Definition 2.1.1: Let matrix G represent a graph, then G is called adjacency matrix (for the graph).

As we have seen before, the sum of probabilities of arrows leaving a vertex must be one -and all arrows must have positive probabilities-. So the sum of the elements of each column of the adjacency matrix must be one.

Definition 2.1.2: Let matrix G be an adjacency matrix such that the sum of each row is one. Then G is a doubly stochastic matrix.

The importance of doubly stochastics matrices is given by the following properties:

Proposition: Let $M \in \mathbb{R}^{n\times n}$ be a doubly stochastic matrix. Let $X = (x_1, \cdots, x_n)^T \in \mathbb{R}^n$ and let the result $Y = (y_1, \cdots, y_n) = MX$. Then

$$\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \tag{2.4}$$

Proof: Let M be a doubly stochastic matrix such that

$$M = \begin{bmatrix} m_{1,1} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n} \end{bmatrix} \tag{2.5}$$

Let's calculate MX

$$MX = \begin{bmatrix} m_{1,1} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} m_{1,i} x_i \\ \vdots \\ \sum_{i=1}^{n} m_{n,i} x_n \end{bmatrix} \tag{2.6}$$

But $Y = MX$ so

$$\sum_{i=1}^{n} y_i = \sum_{j=1}^{n} \sum_{i=1}^{n} m_{j,i} x_i = \sum_{i=1}^{n} \sum_{j=1}^{n} m_{j,i} x_i = \sum_{i=1}^{n} x_i \sum_{j=1}^{n} m_{j,i} \tag{2.7}$$

As long as M is doubly stochastic, then

$$\sum_{i=1}^{n} m_{i,j} \quad \sum_{i=1}^{n} m_{j,i} \quad \forall j = 1, \cdots, n \tag{2.8}$$

and so

$$\sum_{i=1}^{n} y_i = \sum_{i=1}^{n} x_i \sum_{j=1}^{n} m_{j,i} = \sum_{i=1}^{n} x_i \tag{2.9}$$

Proposition: Let $M \in \mathbb{R}^{n \times n}$ be a doubly stochastic matrix. Let $X = (x_1, \cdots, x_n) \in \mathbb{R}^n$ and let the result $Y = (y_1, \cdots, y_n)^T = XM$. Then

$$\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \tag{2.10}$$

Proof: Let doubly stochastic matrix M such that

$$M = \begin{bmatrix} m_{1,1} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n} \end{bmatrix} \tag{2.11}$$

Let's calculate XM

$$XM = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \begin{bmatrix} m_{1,1} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} m_{i,1} x_i \\ \vdots \\ \sum_{i=1}^{n} m_{i,n} x_n \end{bmatrix}^T \tag{2.12}$$

But $Y = MX$ so

$$\sum_{i=1}^{n} y_i = \sum_{j=1}^{n} \sum_{i=1}^{n} m_{i,j} x_i = \sum_{i=1}^{n} \sum_{j=1}^{n} m_{i,j} x_i = \sum_{i=1}^{n} x_i \sum_{j=1}^{n} m_{i,j} \tag{2.13}$$

As long as M is doubly stochastic, then

$$\sum_{i=1}^{n} m_{i,j} \quad \sum_{i=1}^{n} m_{j,i} \quad \forall j = 1, \cdots, n \tag{2.14}$$

Therefore

$$\sum_{i=1}^{n} y_i = \sum_{i=1}^{n} x_i \sum_{j=1}^{n} m_{i,j} = \sum_{i=1}^{n} x_i \tag{2.15}$$

Proposition: Let $M, N \in \mathbb{R}^{n\times n}$ be doubly stochastic matrices. Then MN is a doubly stochastic matrix.

Proof: Let the doubly stochastic matrices

$$M = \begin{bmatrix} m_{1,1} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n} \end{bmatrix} \qquad N = \begin{bmatrix} n_{1,1} & \cdots & n_{1,n} \\ \vdots & \ddots & \vdots \\ n_{n,1} & \cdots & n_{n,n} \end{bmatrix} \tag{2.16}$$

then, we can calculate the product MN

$$MN = \begin{bmatrix} \sum_{i=1}^{n} m_{1,i}n_{1,i} & \cdots & \sum_{i=1}^{n} m_{1,i}n_{n,i} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{n} m_{n,i}n_{1,i} & \cdots & \sum_{i=1}^{n} m_{n,i}n_{n,i} \end{bmatrix} \tag{2.17}$$

Let's calculate the sum of the s-row

$$\sum_{j=1}^{n}\sum_{i=1}^{n} m_{s,i}n_{j,i} \;=\; \sum_{i=1}^{n}\sum_{j=1}^{n} m_{s,i}n_{j,i} = \sum_{i=1}^{n} m_{s,i} \sum_{j=1}^{n} n_{j,i} \tag{2.18}$$

But N and M are doubly stochastic, so

$$\sum_{j=1}^{n}\sum_{i=1}^{n} m_{s,i}n_{j,i} \;=\; \sum_{i=1}^{n} m_{s,i} \sum_{j=1}^{n} n_{j,i} = \sum_{i=1}^{n} m_{s,i} = 1 \tag{2.19}$$

Now let's calculate the sum of the s-column

$$\sum_{j=1}^{n}\sum_{i=1}^{n} m_{j,i}n_{s,i} \;=\; \sum_{i=1}^{n}\sum_{j=1}^{n} m_{j,i}n_{s,i} = \sum_{i=1}^{n} n_{s,i} \sum_{j=1}^{n} m_{j,i} \tag{2.20}$$

But N and M are doubly stochastic, so

$$\sum_{j=1}^{n}\sum_{i=1}^{n} m_{j,i}n_{s,i} \;=\; \sum_{i=1}^{n} n_{s,i} \sum_{j=1}^{n} m_{j,i} = \sum_{i=1}^{n} n_{s,i} = 1 \tag{2.21}$$

Therefore, MN is a doubly stochastic matrix.

We have seen that, given a doubly stochastix matrix M, if we want to know the state of the system at time $t = 2$, we need the matrix $M^2 = MM$. But we can multiply M to another doubly stochastic matrix. Let G, M be

two doubly stochastic matrices of the same size. Then we can define their multiplication

$$GM[i,j] = \sum_{k=1}^{n} G[i,k]M[k,j] \tag{2.22}$$

The meaning of this multiplication is the following: $GM[i,j]$ is the sum of the probabilites of going from vertex j to some arbitrary vertex k, with the probabilities of G, and then going from this vertex k to the vertex i, with the probabilities of M. If, for example, G describes the dynamics of the system to go from $t = 0$ to $t = 1$ and M to go from timestep $t = 1$ to $t = 2$, then GM describes the dynamics of the system to go from timestep $t = 0$ to $t = 2$.

We are now ready to go into the world of quantum. Quantum mechanics works in a similar way as the probabilistic system described before. The difference is that in the probabilistic systems, the probabilities are given by real numbers between 0 and 1 and in quantum mechanics, these probabilities are given by complex numbers $c \in \mathbb{C}$ such that $|c|^2$ is between 0 and 1.

There is one fundamental difference between using real numbers and using complex numbers. We know that given two positive real numbers $r_1, r_2 \in \mathbb{R}$ then $p_1 \leq p_1 + p_2$ and $p_2 \leq p_1 + p_2$. But when we work with two complex numbers $c_1, c_2 \in \mathbb{C}$ such that $0 \leq |c_1|^2, |c_2|^2 \leq 1$, then there's no need for $|c_1 + c_2|^2$ to be bigger than $|c_1|^2$ or $|c_2|^2$ because complex numbers can cancel each other. This phenomenon is referrered to as **interference** and it's one of the most important concepts in quantum theory.

Instead of asking that the adjacency matrix be a doubly stochastic matrix, we ask instead that the adjancency matrix U be unitary ($UU^\dagger = Id$). An example of a well know unitary matrix is

$$U = \begin{bmatrix} cos(\theta) & -sen(\theta) & 0 \\ sen(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.23}$$

Why we want unitary matrices instead of the adjacency matrices described before? The answer of these questions is in the next two propositions:

Proposition: Given any unitary matrix U such that

$$U = \begin{bmatrix} u_{1,1} & \cdots & u_{1,n} \\ \vdots & \ddots & \vdots \\ u_{n,1} & \cdots & u_{n,n} \end{bmatrix} \tag{2.24}$$

Then, using the modulus squared of each of the entries

$$\begin{bmatrix} |u_{1,1}|^2 & \cdots & |u_{1,n}|^2 \\ \vdots & \ddots & \vdots \\ |u_{n,1}|^2 & \cdots & |u_{n,n}|^2 \end{bmatrix} \tag{2.25}$$

we get a doubly stochastic matrix.

Proof: Let U be a unitary matrix such that $u_{l,j} = a_{l,j} + b_{l,j} \in \mathbb{C}\ \forall l, j = 1, \cdots, n$ with $a_{l,j}, b_{l,j} \in \mathbb{R}$. As U is unitary, then

$$UU^\dagger = Id \tag{2.26}$$

and for so, we have that

$$\sum_{l=1}^{n} u_{l,j}\overline{u}_{l,j} = 1 \rightarrow \sum_{l=1}^{n}(a_{l,j} + b_{l,j}i)(a_{l,j} + b_{l,j}i) = 1 \tag{2.27}$$

$$\rightarrow \sum_{l=1}^{n}(a_{l,j}^2 + b_{l,j}^2) = 1 \rightarrow \sum_{l=1}^{n}|u_{l,j}|^2 = 1 \tag{2.28}$$

$$\sum_{j=1}^{n} u_{l,j}\overline{u}_{l,j} = 1 \rightarrow \sum_{j=1}^{n}(a_{l,j} + b_{l,j}i)(a_{l,j} + b_{l,j}i) = 1 \tag{2.29}$$

$$\rightarrow \sum_{j=1}^{n}(a_{l,j}^2 + b_{l,j}^2) = 1 \rightarrow \sum_{j=1}^{n}|u_{l,j}|^2 = 1 \tag{2.30}$$

So the matrix described in Eq 2.25 is doubly stochastic.

Proposition: Given any unitary matrix $U \in \mathbb{C}^{n\times n}$ and given any column vector $c \in \mathbb{C}^n$, then Uc preserves the sum of the modulus squared of c.

Proof: first, we need to prove the next property. Let arbitrary $(a + bi)$, $(c + di) \in \mathbb{C}$

$$\begin{aligned} |(a+bi)(c+di)|^2 &= |(ac-bd)+(ad+bc)i|^2 \\ &= (ac-bd)^2 + (ad+bc)^2 \\ &= a^2c^2 + b^2d^2 - \cancel{2abcd} + a^2d^2 + b^2c^2 + \cancel{2abcd} \\ &= a^2c^2 + b^2d^2 + a^2d^2 + b^2c^2 \\ &= (a^2+b^2)(c^2+d^2) \\ &= |a+bi|^2|c+di|^2 \end{aligned} \tag{2.31}$$

Let $c = (c_1, \cdots, c_n)^T \in \mathbb{C}^n$ such that $c_i = d_i + e_i$ and let U be a unitary matrix such that

$$U = \begin{bmatrix} u_{1,1} & \cdots & u_{1,n} \\ \vdots & \ddots & \vdots \\ u_{n,1} & \cdots & u_{n,n} \end{bmatrix} = \begin{bmatrix} a_{1,1} + b_{1,1}i & \cdots & a_{1,n} + b_{1,n}i \\ \vdots & \ddots & \vdots \\ a_{n,1} + b_{n,1}i & \cdots & a_{n,n} + b_{n,n}i \end{bmatrix} \tag{2.32}$$

From the previous proposition we know that

$$\sum_{i=1}^{n} |a_{i,j} + b_{i,j}i|^2 = 1 \quad \sum_{i=1}^{n} |a_{j,i} + b_{j,i}i|^2 = 1 \quad \forall j = 1, \cdots, n \tag{2.33}$$

Now we are going to calculate the product Uc

$$Uc = \begin{bmatrix} \sum_{j=1}^{n}(a_{1,j} + b_{1,j}i)(d_1 + e_1 i) \\ \vdots \\ \sum_{j=1}^{n}(a_{n,j} + b_{n,j}i)(d_n + e_n i) \end{bmatrix} \tag{2.34}$$

If we calculate the sum of the modulus squared of the multiplication we get

$$\begin{aligned} & \sum_{k=1}^{n}\sum_{j=1}^{n} |(a_{k,j} + b_{k,j}i)(d_k + e_k i)|^2 && (2.35) \\ = & \sum_{k=1}^{n}\sum_{j=1}^{n} |a_{k,j} + b_{k,j}i|^2 |d_k + e_k i|^2 && (2.36) \\ = & \sum_{k=1}^{n} |d_k + e_k i|^2 \sum_{j=1}^{n} |a_{k,j} + b_{k,j}i|^2 && (2.37) \\ = & \sum_{k=1}^{n} |d_k + e_k i|^2 && (2.38) \end{aligned}$$

One of the main principle in quantum mechanics is the **superposition**. This principle is the most important quality to overcome the information procesing capacity of classical systems. Paul Dirac described this concept in [2] as follow: *The general principle of superposition of quantum mechanics applies to the states [that are theoretically possible without mutual interference or contradiction] ... of any one dynamical system. It requires us to assume that between these states there exist peculiar relationships such that whenever the system is definitely in one state we can consider it as being partly in each of two or more other states. The original state must be regarded as the result of a kind of superposition of the two or more new states, in a way that cannot be conceived on classical ideas. Any state may be considered as the result of a superposition of two or more other states, and indeed in an infinite number of ways. Conversely, any two or more states may be superposed to give a new state...*

The non-classical nature of the superposition process is brought out clearly if we consider the superposition of two states, A and B, such that there exists an observation which, when made on the system in state A, is certain to lead to one particular result, a say, and when made on the system in state B is certain to lead to some different result, b say. What will be the result of the observation when made on the system in the superposed state? The answer is that the result will be sometimes a and sometimes b, according to a probability law depending on the relative weights of A and B in the superposition process. It will never be different from both a and b [i.e., either a or b]. The intermediate character of the state formed by superposition thus expresses

itself through the probability of a particular result for an observation being intermediate between the corresponding probabilities for the original states, not through the result itself being intermediate between the corresponding results for the original states.

The mind blowing consequence of superposition is that one particle is "in some degree" in several places at the same time. But what happens if we measure if the particle is in one of these places? Then, the superposition collapses and the particle is only in one of them. That means that we can work with particles that behave as if they are simultaneously in multiple places, but if we observe them, they are only in one of the places, with a given probability.

2.2 Basic Quantum Theory

In the Quantum world a different notation is used to represent the vectors. The commonly used notation is the Dirac or Bra notation. The Diract ket is used to represent column vectors. For example, let $x = (c_1, \cdots, c_n)^T \in \mathbb{C}^n$, then this vector in the Diract notation is represented by $|x\rangle$. Alternatively, for the row vectors we have the bra notation. Let the vector $y = (d_1, \cdots, d_n) \in \mathbb{C}^n$, then this vector in the bra notation is represented by $\langle y|$.

One of the basic concepts that we need in Quantum Computing -and in Quantum Mechanicals in general- is the quantum states, that describes the state of our system at a given time. To undestand the concept, supose that we have one particle that can be found in one of the positions $x_1, \cdots, x_n$. Then the state of the system is the vector $(c_1, \cdots, c_n)^T \in \mathbb{C}^n$ that represents the probabilities of the particle being found in each position. The complex weights $c_1, \cdots, c_n$ are know as complex amplitudes. We define the state $|\psi\rangle$ of the particle as

$$|\psi\rangle = c_1|x_1\rangle + \cdots + c_n|x_n\rangle \tag{2.39}$$

Where the ket $|x_i\rangle$ represents the state of the particle being in this position. If we see $|\psi\rangle$ in this way, we can say that $|\psi\rangle$ is a superposition of the basic states $|x_1\rangle, \cdots, |x_n\rangle$. In this way, the probability for being in state $|x_i\rangle$ is

$$P(|x_i\rangle) = \frac{|c_i|^2}{\sum_{j=1}^{n} |c_j|^2} \tag{2.40}$$

Let $c \in \mathbb{C}$ be an arbitrary complex number and consider $c|\psi\rangle$, then if we

calculate again the probability of being at state $|x_i\rangle$ we have that

$$P_{c|\psi\rangle}(|x_i\rangle) = \frac{|cc_i|^2}{\sum_{j=1}^{n}|cc_j|^2} = \frac{|c|^2|c_i|^2}{\sum_{j=1}^{n}|c|^2|c_j|^2} \tag{2.41}$$

$$= \frac{\cancel{|c|^2}|c_i|^2}{\cancel{|c|^2}\sum_{j=1}^{n}|c_j|^2} = \frac{|c_i|^2}{\sum_{j=1}^{n}|c_j|^2} = P_{|\psi\rangle}(|x_i\rangle) \tag{2.42}$$

Therefore, the probabilities of being at each state of $|\psi\rangle$ does not change if we multiply them by any complex number. This means that, given any $|\psi\rangle$, we can always find another $|\psi^{'}\rangle$ that represents the same state as $|\psi\rangle$ but with modulus 1. This ket is known as the normalized vector of $|\psi\rangle$. We can calculate $|\psi^{'}\rangle$ as follow

$$|\psi^{'}\rangle = \frac{|\psi\rangle}{||\psi\rangle|} \tag{2.43}$$

To introduce the next important concept, we need to explain a property of subatomic particles called spin. Spin is an intrinsic form of angular momentum and it is one of two types of angular momentum in quantum mechanics. In some ways, spin is like a vector quantity: it has a defined magnitude, and it has a "direction". There are only two basic spin states for each direction in space. From the point of view of the vertical axis, these states are spin up $|\uparrow\rangle$ and spin down $|\downarrow\rangle$. Then a generic superposed state $|\psi\rangle$ of this basic states is

$$|\psi\rangle = c_1|\uparrow\rangle + c_2|\downarrow\rangle \tag{2.44}$$

Where $c_1, c_2 \in \mathbb{C}$ allows us to compute probabilities of finding the particle with spin up ($|c_1|^2$) or finding them with spin down ($|c_2|^2$).

Inner product is important to determine how likely the state of the given system will change to another state after measuring. The complex number that determines this concept is known as the transition amplitude. Supose that we have two arbitrary states $|\psi\rangle = (c_1, \cdots, c_n)^T$ and $|\psi^{'}\rangle = (c_1^{'}, \cdots, c_n^{'})^T$ then we need to calculate the inner product between them. In order to do so in ket-bra notation, let's put $|\psi^{'}\rangle$ in bra notation

$$\langle\psi^{'}| = |\psi^{'}\rangle^{\dagger} = (\overline{c_1^{'}}, \cdots, \overline{c_n^{'}}) \tag{2.45}$$

And then, we multiply them using an inner product to find the transition amplitude

$$\langle\psi^{'}|\psi\rangle = (\overline{c_1^{'}}, \cdots, \overline{c_n^{'}}) \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = \overline{c_1^{'}}c_1 + \cdots + \overline{c_n^{'}}c_n \tag{2.46}$$

The next concept allows us to have information about the properties of the system: observables. We call observable to the physical quantities than can be observed in each state. We can think of observables as questions about the system that we want to answer. We have two important postulates that we need to know.

Postulate 1:
To each physical observable there corresponds a hermitian operator.

Postulate 2:
The eigenvalues of a hermitian operator ω associated with a physical observable are the only possible values the observable can take as a result of measuring it on any given state. Furthermore, the eigenvectors of ω form a basis for the state space.

We may be concerned about manipulating an observable to obtain another one, so we are going to study how we can transform an observable -that is a hermitian operator- so that it remains being an observable.

Proposition: Let an hermitian matrix H and a real scalar r. Then rH is hermitian.

Proof: Let $H \in \mathbb{R}^{n\times n}$ be a hermitian matrix. As it is hermitian, then

$$H[i,j] = \overline{H[j,i]} \qquad \forall i,j = 1,\cdot,n \tag{2.47}$$

We know that $r \in \mathbb{R}$ so $\overline{r} = r$. If rH is hermitian then it must be fulfilled for all $i,j = 1,\cdot,n$ that

$$rH[i,j] = \overline{rH[j,i]} = \overline{r}\overline{H[j,i]} = r\overline{H[j,i]} \tag{2.48}$$

$$\rightarrow \not{r}H[i,j] = \not{r}\overline{H[j,i]} \rightarrow H[i,j] = \overline{H[j,i]} \tag{2.49}$$

And that is true by hypothesis.

So we can multiply an arbitrary hermitian matrix by any real number and still be hermitian but, what about complex numbers? When we try an example, we see that this property is not true for complex numbers. Let's see an example. Let

$$H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad c = i \tag{2.50}$$

then

$$cH = \begin{bmatrix} i & 0 \\ 0 & i \end{bmatrix} \qquad (cH)^\dagger = \begin{bmatrix} -i & 0 \\ 0 & -i \end{bmatrix} \tag{2.51}$$

And so, cH is not an hermitian matrix. Another useful transformation is the addition. If we have an observable Ω_1 and another observable Ω_2 then we can have their sum $\Omega_1 + \Omega_2$ as the sum of their hermitian matrices. But in order to have it, we will need the following proposition.

Proposition: Let two hermitian matrices $H, H^{'} \in \mathbb{C}^{c\times n}$, then $H + H^{'}$ is hermitian.

Proof:
To prove that, we need to prove that $H[i,j] + H^{'}[i,j] = \overline{H[i,j] + H'[i,j]}$

$$\overline{H[i,j] + H'[i,j]} = \overline{H[i,j]} + \overline{H'[i,j]} \tag{2.52}$$

but both H and $H^{'}$ are hermitian, so

$$\overline{H[i,j] + H'[i,j]} = \overline{H[i,j]} + \overline{H'[i,j]} = H[i,j] + H^{'}[i,j] \tag{2.53}$$

The sum of hermitian matrices is hermitian but, what about multiplication? Well, multiplication generally does not produce an hermitian matrix. Let $\Omega_1, \Omega_2 \in \mathbb{C}^{n\times n}$ two hermitian matrices and let ψ, ϕ two arbitrary states. Then we have

$$\langle \Omega_1\Omega_2\phi, \psi\rangle = \langle \Omega_2\phi, \Omega_1\psi\rangle = \langle \phi, \Omega_1\Omega_2\psi\rangle \tag{2.54}$$

where the equalities comes from the fact that Ω_1 and Ω_2 are hermitian. Then, for the multiplication being hermitian we need that

$$\langle \Omega_1\Omega_2\phi, \psi\rangle = \langle \phi, \Omega_2\Omega_1\psi\rangle \tag{2.55}$$

which implies that

$$\Omega_1\Omega_2 = \Omega_2\Omega_1 \tag{2.56}$$

and we know that for general matrices this is not always true. In the case that it is, then the multiplication is hermitian. If they are not hermitian, we can define a usefull operator called commutator as

$$[\Omega_1, \Omega_2] = \Omega_1\Omega_2 - \Omega_2\Omega_1 \tag{2.57}$$

which is a hermitian matrix. To prove this, Let's $H, \Omega \in \mathbb{C}^{n\times n}$ be a hermitian

matrices. Now, for $i, j = 1, \cdots, n$ consider

$$\begin{aligned}
[H, \Omega][i, j] &= (H\Omega)[i, j] - (\Omega H)[i, j] && (2.58)\\
&= \sum_{h=0}^{n}(H[i, h]\Omega[h, j]) - \sum_{h=0}^{n}(\Omega[i, h]H[h, j]) && (2.59)\\
&= \sum_{h=0}^{n}(H[i, h]^{\dagger}\Omega[h, j]^{\dagger}) - \sum_{h=0}^{n}(\Omega[i, h]^{\dagger}H[h, j]^{\dagger}) && (2.60)\\
&= \sum_{h=0}^{n}(\Omega[i, h]H[h, j])^{\dagger} - \sum_{h=0}^{n}(H[i, h]\Omega[h, j])^{\dagger} && (2.61)\\
&= \left(\sum_{h=0}^{n}(\Omega[i, h]H[h, j]) - \sum_{h=0}^{n}(H[i, h]\Omega[h, j])\right)^{\dagger} && (2.62)\\
&= [H, \Omega][i, j]^{\dagger} && (2.63)
\end{aligned}$$

To introduce the next postulate, we need to remember that a hermitian operator Ω is an operator that, for two given states $|\psi\rangle, |\phi\rangle$, respects the inner product, that is

$$\langle\Omega|\psi\rangle, |\phi\rangle\rangle = \langle|\psi\rangle, \Omega|\phi\rangle\rangle \tag{2.64}$$

So, if it is the same state, then it is a real unique value and we will denote as

$$|\Omega\rangle_{\psi} = \langle\Omega|\psi\rangle, |\phi\rangle\rangle = \langle|\psi\rangle, \Omega|\phi\rangle\rangle \tag{2.65}$$

Postulate 3:
$|\Omega\rangle_{\psi}$ is the expected value of observing Ω repeteadly on the same state $|\psi\rangle$.

Let's explain the postulate. Let Ω be a hermitian operator and $\lambda_1, \cdots, \lambda_n$ the list of its eigenvalues. When we observed this states, we are going to obtain one of the eigenvalues. Supose that we make m observations. Then, we observed p_i times every λ_i with $0 \geq p_i \geq m$. Now perform the calculation

$$\lambda_1\frac{p_1}{m} + \cdots + \lambda_n\frac{p_n}{m} \tag{2.66}$$

If m is large enough, the previous value will be very close to $|\Omega\rangle_{\psi}$.

As we will see, variance is an important concept in quantum mechanics. In order to define it, we need first to introduce the hermitian operator

$$\Delta_{\psi}(\Omega) = \Omega - \langle\Omega\rangle_{\psi}Id \tag{2.67}$$

where Id is the identity matrix, Ω is an hermitian operator and $|\psi\rangle$ is a normalized vector. So, we can now define the variance of Ω at $|\psi\rangle$ as

$$Var_{\psi}(\Omega) = \langle(\Delta_{\psi}(\Omega) \star \Delta_{\psi}(\Omega))\rangle_{\psi} \tag{2.68}$$

Now we are ready to get into one of the most important principles of quantum mechanics.

Heisenberg's Uncertain Principle: The product of the variances of two arbitrary hermitian operators on a given state is always greater than or equal to one-fourth the square of the expected value of their commutator. In formulas:

$$Var_\psi(\Omega_1) \times Var_\psi(\Omega_2) \geq \frac{1}{4} \left| \langle [\Omega_1, \Omega_2] \rangle_\psi \right|^2 \tag{2.69}$$

So, using the commutator, Heisenberg's Uncertain Principle tells us how good a simultaneous measurement of two observables can be. If we look deeper to the formula of the principle, we can notice than if the commutator is 0, then there's no limit to how good the measurament can be.

Measuring is the act of observing a given physical system. If we think about the metaphor in which observables represents questions posed to the system, then the act of measuring is to ask one of these specific questions. In classical systems, we assume that measuraments will not change the system and will always give a predictable state but these two assumptions prove wrong when we consider quantum systems. In a quantum system, measuring will perturb a system and will modify it. Futhermore, the state that will yield as a result of this measurament can not be well defined beforehand, we can only calculate the probability of being in specific states. Then, we do not have a way to determine how frecuently we are going to see a specific eigenvalue λ. Moreover, we do not have a way to know what happens if we actually observe this value λ. We need the next postulate to handle concrete measurements.

Postulate 4:
Let Ω be an observable and $|\psi\rangle$ be a state. If the result of measuring Ω is the eigenvalue λ, then the state after measurament will always be an eigenvector corresponding to λ.

In order to understand the postulate, let's see an example. Let a quantum system described by

$$\Omega = \begin{bmatrix} 0 & -1 \\ -1 & 2 \end{bmatrix} \tag{2.70}$$

Then, thanks to the previous postulate, we can define all the posible states that this system can be. Let $|\psi\rangle$ be an arbitrary state of system described by Ω. Then that means that ψ is an eigenvector of Ω and so there exists an eigenvalue $c \in \mathbb{C}$ such that

$$\Omega|\psi\rangle = \lambda|\psi\rangle \tag{2.71}$$

If we suppose that $|\psi\rangle$ has the form $|\psi\rangle = (a, b)^T$ where $a, b \in \mathbb{C}$, then we can express the equality as

$$\Omega|\psi\rangle = \begin{bmatrix} 0 & -1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} -b \\ -a + 2b \end{bmatrix} = \lambda|\psi\rangle \tag{2.72}$$

So, we have both equalities

$$-b = \lambda a \rightarrow \lambda = \frac{-b}{a} \tag{2.73}$$

$$\lambda b = -a + 2b \tag{2.74}$$

replacing the first equality into the second we have that

$$\frac{-b}{a} b = -a + 2b \rightarrow -a^2 + b^2 + 2ab = 0 \rightarrow b = a \pm a\sqrt{2} \tag{2.75}$$

replacing this result into the first one, we have that

$$\lambda = \frac{-\cancel{a} \mp \cancel{a}(\sqrt{2}}{\cancel{a}} = \mp\sqrt{2} - 1 \tag{2.76}$$

So far, we have studied quantum systems that do not evolve over time. But in the real world, the system evolve over time, and for so we need to study the dynamics of a system. The next postulate tell us how to represent such dynamics.

Postulate 5:
The evolution of a quantum system (that is not a measurament) is given by a unitary operator or transformation.

So the dynamics of a system is given by another unitary operator. Let us see an example of how the dynamics works. Let be $|\phi\rangle = [1, 0, 0, 0]^T$ the initial state vector and let the dynamics be given by

$$\Omega = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{i}{\sqrt{2}} & 0 & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & 0 & \frac{i}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 \end{bmatrix} \tag{2.77}$$

Then, the state after 3 timesteps is

$$\Omega^3|\phi\rangle = \frac{1}{4}[4i - 2, 0, 0, 2i]^T \tag{2.78}$$

2.3 Quantum Architecture

At this point we already know how to work with one particle in a quantum system. We know how to determine its possible states, the unitary operations that we can apply over it an how to determine its dynamics. But we are interested in combining multiple particles to deal with a more complex quantum system. In this section we will see how to assemble quantum systems. The following postulate describes how to combine two independent systems.

Postulate 6:
Assume we have two independent quantum systems Q and $Q^{'}$, represented respectively by the vectors spaces $\mathbb{V}$ and $\mathbb{V}^{'}$. The quantum system obtained by merging Q and $Q^{'}$ will have the tensor product $\mathbb{V} \otimes \mathbb{V}^{'}$ as a state space.

Let us see an example to understand how to combine systems. Let $Q^{'}$ and Q in $\mathbb{C}^2$ be two independent systems with their respective basis $\{|x_0\rangle, |x_1\rangle\}$ and $\{|y_0\rangle, |y_1\rangle\}$. Then, the combined system has the basis

$$\{|x_0\rangle \otimes |y_0\rangle, |x_0\rangle \otimes |y_1\rangle, |x_1\rangle \otimes |y_0\rangle, |x_1\rangle \otimes |y_1\rangle\} \tag{2.79}$$

Therefore, the basic states of the combined system are just the tensor product of basic states of each system. We can think that all states from the assembled system can be rewritten as a tensor products of basic states, each one from one system. But we find that this is not always true. When we can not make this separation it means that the states are **entangled**.

Let us work with the simplest nontrivial system of two particles in which each particle has two posible states. Then, we can express any state of the first particle as

$$c_0|x_0\rangle + c_1|x_1\rangle \tag{2.80}$$

with $c_0, c_1 \in \mathbb{C}$. Similary, we can express any state of the second particle as

$$c_0^{'}|y_0\rangle + c_1^{'}|y_1\rangle \tag{2.81}$$

With $c_0^{'}, c_1^{'} \in \mathbb{C}$. Then, from Postulate 6 we can write any state $|\phi\rangle$ of the combinated system as

$$\begin{aligned} |\phi\rangle &= (c_0|x_0\rangle + c_1|x_1\rangle) \otimes (c_0^{'}|y_0\rangle + c_1^{'}|y_1\rangle) && (2.82)\\ &= c_0c_0^{'}|x_0\rangle \otimes |y_0\rangle + c_0c_1^{'}|x_0\rangle \otimes |y_1\rangle && (2.83)\\ &\quad + c_1c_0^{'}|x_1\rangle \otimes |y_0\rangle + c_1c_1^{'}|x_1\rangle \otimes |y_1\rangle && (2.84) \end{aligned}$$

Supose that we have the state

$$|\psi\rangle = |x_0\rangle \otimes |y_1\rangle + |x_1\rangle \otimes |y_1\rangle \tag{2.85}$$

To determine from which basic states is $|\psi\rangle$ built, we match this state with the general expression, getting the following system of equations

$$\begin{aligned} c_0 c_0^{'} &= 0 && (2.86)\\ c_0 c_1^{'} &= 1 && (2.87)\\ c_1 c_0^{'} &= 0 && (2.88)\\ c_1 c_1^{'} &= 1 && (2.89) \end{aligned}$$

Solving it, we get that the state $|\psi\rangle$ is a combination of the states

$$\begin{aligned} |\psi_0\rangle &= |x_0\rangle + |x_1\rangle && (2.90)\\ |\psi_1\rangle &= |y_1\rangle && (2.91) \end{aligned}$$

As long as we can split $|\psi\rangle$ in a combination of basics states, we call it a **separable state**. On the contrary, supose that we have the state

$$|\psi^{'}\rangle = |x_0\rangle \otimes |y_0\rangle + |x_0\rangle \otimes |y_1\rangle + |x_1\rangle \otimes |y_0\rangle + |x_1\rangle \otimes |y_1\rangle \tag{2.92}$$

This state yields us the next system of equations

$$\begin{aligned} c_0 c_0^{'} &= 1 && (2.93)\\ c_0 c_1^{'} &= 1 && (2.94)\\ c_1 c_0^{'} &= 1 && (2.95)\\ c_1 c_1^{'} &= 1 && (2.96) \end{aligned}$$

which has no solution, so we can not split $|\psi^{'}\rangle$ into a combination of basics states, so we denote $|\psi^{'}\rangle$ as a entangled state.

As we mentioned before, the simplest quantum system that we can have is a two dimensional system, and so it will become the basic unit of information in quantum computing -such as the bits for classical computing- with the next definition:

Definition 2.3.1: A quantum bit or qubit is a unit of information describing a two-dimensional quantum system.

So, qubits have two elemental states. We denote them as $|0\rangle$ and $|1\rangle$. Also, the states of a qubit have to be normalized, so if we want to represent the state $[1+i, 1-i]$, then we need to calculate its norm ($\sqrt{4}$) and divide the whole vector by it, obtaining the state

$$|\phi\rangle = \frac{1+i}{\sqrt{4}}|0\rangle + \frac{1-i}{\sqrt{4}}|1\rangle \tag{2.97}$$

Postulate 6 tell us how to combine quantum systems, including qubits by means of the tensor product. Let us see an easy example. Supose we have the qubits $|\phi_0\rangle, |\phi_1\rangle$ such that $|\phi_0\rangle = |1\rangle$ and $|\phi_1\rangle = |0\rangle$, so the result of combining these two systems is

$$|\phi_0\rangle \otimes |\phi_1\rangle = |1\rangle \otimes |0\rangle \tag{2.98}$$

We can also represent this state in vector form as $[0, 0, 1, 0]^T$ where the first element corresponds to the coefficient of the state $|00\rangle$, the second element of $|01\rangle$ and so on. Now, supose that we combine $|\phi\rangle$ and $|\phi_0\rangle$, so as a result we have

$$|\phi\rangle \otimes |\phi_0\rangle = \frac{1+i}{\sqrt{4}}|01\rangle + \frac{1-i}{\sqrt{4}}|11\rangle \tag{2.99}$$

and so, its vector form is

$$\frac{1}{\sqrt{4}} \begin{bmatrix} 0 \\ 1+i \\ 0 \\ 1-i \end{bmatrix} \tag{2.100}$$

2.4 Quantum Gates

So far, we have defined the basic structure of information of the quantum computing, but we need a way to manipulate that information. In classical systems, we use classical gates such as NAND or NOR, so in quantum mechanics we have quantum gates. In order to study these gates, we first need a formal definition of a quantum gate.

Definition 2.4.1: A quantum gate is an operator that acts on qubits. Such an operator will be represented by a unitary matrix.

The previous definition implies that quantum gates are reversible. A gate is reversible if and only if we can deduce the input with the output of the gate, in other words, a gate is reversible if and only if exists another gate such that applying both gates in succesion leaves the input unchanged. Many

classical gates are not reversible. For example, the inputs of an AND gate can not be deducted from its output.

Now we are going to study the basic gates that are used in quantum computing. The simplest gate is the identity, as long as $Id \star Id = Id$, so it is reversible. One of the most important quantum gate is the controlled-NOT gate. We can see the behaviour of the gate in Figure 2.3 in which we can see $|x\rangle$ and $|y\rangle$ being the two qubits input. $|x\rangle$ is the control qubit and $|y\rangle$ is the target qubit, so that this gate applies and NOT operation to the target qubit if and only if the control qubit is in state $|1\rangle$.

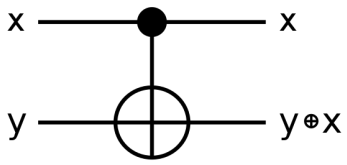

Figure 2.3: CNOT Gate Diagram.

The controlled-NOT gate is given by the next unitary matrix

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.101}$$

We can easily see that the CNOT gate is reversible:

$$CNOT \star CNOT = Id \tag{2.102}$$

The other most important gate is the Hadamard gate. It is important because this gate allows us to build a superposition of the two basic states given one of them, that means

$$|0\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{2.103}$$

$$|1\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{2.104}$$

The Hadamard gate is represented by

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.105}$$

And we can also easily see that Hadamard matrix is unitary and hermitian, as long as

$$H \star H^{\dagger} = H \star H = Id \tag{2.106}$$

The next gates that we are going to see are called the Pauli gates, whose associated matrices are:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{2.107}$$

Each of these matrices is also unitary and hermitian, as we show below

$$X \star X = Id \tag{2.108}$$
$$Y \star Y = Id \tag{2.109}$$
$$Z \star Z = Id \tag{2.110}$$

Another important matrices that we will use to perform rotations of the quantum state are the next ones

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \qquad T = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{bmatrix} \tag{2.111}$$

These matrices are unitary and they are not hermitian

$$S \star \overline{S} = Id \tag{2.112}$$
$$T \star \overline{T} = Id \tag{2.113}$$

These last 5 quantum gates have interesting properties that relate them to each other.

1.
$$X^2 = Y^2 = Z^2 = Id \tag{2.114}$$

Proof:
Trivial due to the demostration of being reversible.

2.
$$H = \frac{1}{\sqrt{2}}(X + Z) \tag{2.115}$$

Proof:

$$\frac{1}{\sqrt{2}}(X + Z) = \frac{1}{\sqrt{2}} \left(\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \right) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = H \tag{2.116}$$

3.

$$X = H \star Z \star H \tag{2.117}$$

Proof:

$$\begin{aligned} H \star Z \star H &= \left(\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\right) \star \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} \\ &\quad \star \left(\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\right) \\ &= \frac{1}{2}\left(\begin{bmatrix}1 & -1\\ 1 & 1\end{bmatrix} \star \begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\right) \\ &= \frac{1}{2}\begin{bmatrix}0 & 2\\ 2 & 0\end{bmatrix} = X \end{aligned} \tag{2.118}$$

4.

$$Z = H \star X \star H \tag{2.119}$$

Proof:

$$\begin{aligned} H \star X \star H &= \left(\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\right) \star \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} \\ &\quad \star \left(\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\right) \\ &= \frac{1}{2}\left(\begin{bmatrix}1 & 1\\ -1 & 1\end{bmatrix} \star \begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\right) \\ &= \frac{1}{2}\begin{bmatrix}2 & 0\\ 0 & -2\end{bmatrix} = Z \end{aligned} \tag{2.120}$$

5.

$$-Y = H \star Y \star H \tag{2.121}$$

Proof:

$$\begin{aligned} H \star Y \star H &= \left(\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\right) \star \begin{bmatrix}0 & -i\\ i & 0\end{bmatrix} \\ &\quad \star \left(\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\right) \\ &= \frac{1}{2}\left(\begin{bmatrix}i & -i\\ -i & -i\end{bmatrix} \star \begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\right) \\ &= \frac{1}{2}\begin{bmatrix}0 & 2i\\ -2i & 0\end{bmatrix} = -Y \end{aligned} \tag{2.122}$$

6.

$$S = T^2 \tag{2.123}$$

Proof:

$$T^2 = \begin{bmatrix}1 & 0\\ 0 & e^{\frac{i\pi}{4}}\end{bmatrix}^2 = \begin{bmatrix}1 & 0\\ 0 & e^{\frac{i\pi}{2}}\end{bmatrix}^2 = \begin{bmatrix}1 & 0\\ 0 & i\end{bmatrix} = S \tag{2.124}$$

As long as

$$e^{\frac{i\pi}{2}} = cos\left(\frac{\pi}{2}\right) + isen\left(\frac{\pi}{2}\right) = i \tag{2.125}$$

7.

$$-Y = X \star Y \star X \tag{2.126}$$

Proof:

$$\begin{aligned} X \star Y \star X &= \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} \star \begin{bmatrix}0 & -i\\ i & 0\end{bmatrix} \star \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} \\ &= \begin{bmatrix}i & 0\\ 0 & -i\end{bmatrix} \star \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} \\ &= \begin{bmatrix}0 & i\\ -i & 0\end{bmatrix} = -Y \end{aligned} \tag{2.127}$$

Summarizing, the properties are

1.

$$X^2 = Y^2 = Z^2 = Id \tag{2.128}$$

2.

$$H = \frac{1}{\sqrt{2}}(X + Z) \tag{2.129}$$

3.

$$X = H \star Z \star H \tag{2.130}$$

4.

$$Z = H \star X \star H \tag{2.131}$$

5.

$$-Y = H \star Y \star H \tag{2.132}$$

6.

$$S = T^2 \tag{2.133}$$

7.

$$-Y = X \star Y \star X \tag{2.134}$$

All the quantum gates that we have shown before are gates designed to use with two qubits, but we can design quantum gates for more than two qubits. For example, the Toffoli gate is an important one that it is applied over 3 qubits. It is a doubly controlled NOT gate. In Figure 2.4 we see that the gate applies a NOT to the third qubit $|z\rangle$ if and only if te first two qubits $|x\rangle$ and $|y\rangle$ are $|1\rangle$.

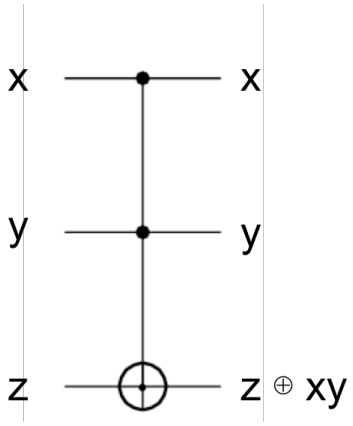

Figure 2.4: Toffoli Gate Diagram.

Using the Toffoli gates we can build reversible versions of the classical gates, such as AND, OR or NAND. Toffoli gate is represented by the next unitary

and hermitian matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.135}$$

2.5 Simulating Quantum Computer

So far, we have introduced the basic concepts needed to develop quantum algorithms. But as long as this is an introduction to quantum computing, we are not only interested in the mathematical foundations, but also in how to implement quantum algorithms. Quantum computers are too expensive to buy one, so we can not test the algorithms on our own quantum computer. Instead, we will use two different tools that allow us to implement our algorithms and run them in a quantum simulator or a real quantum computer.

The first tool that we are going to use is Quirk[1]. Quirk is an open source quantum simulator developed by the community to study quantum computing. It provides a graphic interface in which we can design our quantum circuits by dragging and dropping the gates, make meassuraments and obtain graphicall and statistical information about the resulting quantum states. It is very helpful to start in the quantum computing world due to its simplicity and expressiveness.

In Figure 2.5 we can see the initial interface of Quirk. In the upper part we have the basic gates that we studied before, in the lower part more complex gates (see [3] for more information) and in the center of the interface we can build, analyze and test our quantum circuit. Each line represent a qubit and its transformations over time, and the circles at the end of the lines represent the states of each qubit using a Bloch sphere representation. If we want to apply a gate to a qubit i.e. a H -Hadamard- gate to the first qubit, we just drag and drop the gate over the line as shown in Figure 2.6. As you can see, an H gate is now in the first line, and that modifies the final state

[1] https://algassert.com/quirk

of our qubit. H gate, as we studied before, builds a superposition of the two basic states of a qubit. Now, the final state of the first qubit has changed and two of the final circles have an equal blue background. That means that we can measure the qubit in two different states with 50% probability. If we put the cursor over one of these sphere, we can see the probability of being in this specific state. If we want to remove a gate applied to one qubit, we just drag and drop the gate outside the circuit and it will be removed. More information about the usage of the Quirk simulator can be found in its web page [2].

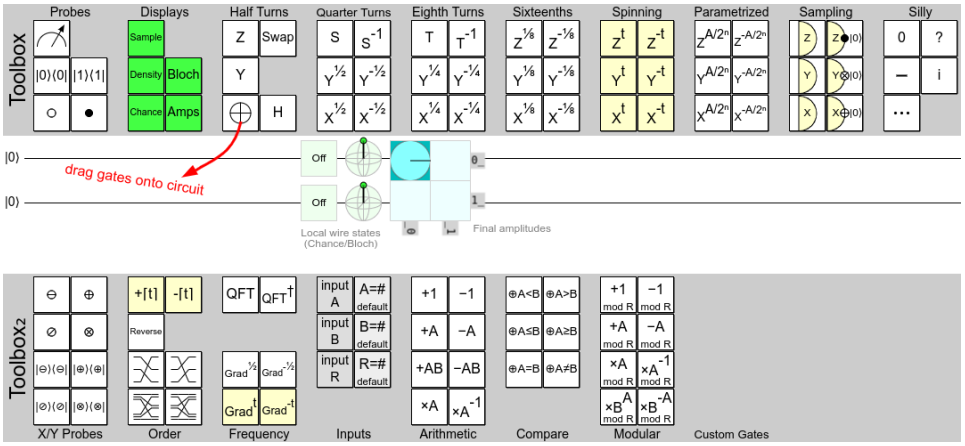

Figure 2.5: Quirk Interface.

Figure 2.6: Quirk Basic Circuit.

[2] https://github.com/Strilanc/Quirk/wiki/How-to-use-Quirk

The other tool that we are going to use is projectQ [3]. This tool offers us a python API to implement quantum algorithms. It offers us the capability to declare qubits and use quantum gates as functions. To use this API, the first thing we need to do in our program is declaring an engine in which to simulate or execute the quantum algorithm. We can use the function *allocate_qubit* to declare one qubit. The funcion returns the qubit, and we can then apply gates to it. To apply the gates (i.e. the H gate), we need to pipe the qubit into the gate ($H \mid qubit$). When we put the gates into our algorithm, they do not modify the qubit. To truly apply the gates to one qubit, we need to use the function *flush* at the end of our quantum circuit. To measure a given qubit, we need to apply the *Measurament* function as a gate. To understand how to use ProjectQ, we include a code example in Figure 2.7 in which we apply the H gate to a qubit and then measure it.

```
from projectq import MainEngine # import the main compiler engine
from projectq.ops import H, Measure # import the operations we want
    to perform (Hadamard and measurement)

eng = MainEngine() # create a default compiler (the back-end is a
    simulator)
qubit = eng.allocate_qubit() # allocate 1 qubit

H | qubit # apply a Hadamard gate
Measure | qubit # measure the qubit

eng.flush() # flush all gates (and execute measurements)
print("Measured {}".format(int(qubit))) # output measurement result
```

Figure 2.7: Basic Program in ProjectQ.

[3] http://projectq.ch/

3

Algorithms

We have seen that the quantum computing world offers a very unusual paradigm that it is not always intuitive. In fact, Richard Faynman said "If you think you understand quantum mechanics, you don't understand quantum mechanics". So, why do we want to program anything using this complex paradigm? The answer is that quantum computing can be much faster than classical computing in some cases and it allows us to simulate many quantum phenomenons.

In this chapter we review some basic quantum algorithms, including their implementations, and the benefits to use the quantum versions instead of the classical ones.

3.1 Quantum Teleportation

The first algorithm that we are going to see is the Quantum Teleportation. This algorithm leverages the property that, if two qubits are entangled, then no matter how far apart they are, when we change one of them, then the other changes simultaneously. With this property in mind, we are going to send data using this entangled qubits with no time delay! Indeed, we are going to see that this is not really true due to the fact that we need to send classical data to perform the quantum teleportation.

Supose that we have two persons, Alice and Bob that want to share information, in fact, Alice wants to send a qubit to Bob. First, Alice and Bob entangle two qubits, $|y\rangle$ and $|z\rangle$ and each one takes one of them. Then,

when Alice wants to send the state of a third qubit $|x\rangle$ to Bob, she starts by entangling the qubit that she wants to send to Bob to her component $|y\rangle$ of the initial entangled pair. Next, Alice measures both qubits $|x\rangle$ and $|y\rangle$. With this measurament, the state of the qubit $|x\rangle$ that we want to send is teleported to Bob's qubit $|z\rangle$. Therefore, Bob gets instanteniusly the state of $|x\rangle$ in his component $|z\rangle$ of the initial entangled pair.

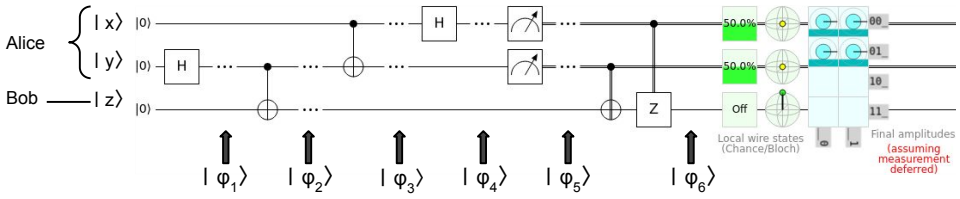

Figure 3.1: Quantum Teleportation in Quirk.

In Figure 3.1 we show an implementation of quantum teleportation using quirk. In it, we start with all the qubits inicialized to $|0\rangle$, and we can also see the final states of the qubits. In $|\varphi_4\rangle$ the first two qubits can be measured in any combination of the basic states with equal probability. After Bob applies the appropiate gates based on the classical information recived from Alice, it gets the state of the qubit that Alice wanted to send him. Let's detail the main steps of the algorithm.

Let $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ be the qubit that Alice wants to send to Bob. In order to do that, Alice and Bob need to share two entangled qubits. The next definition describes this two entangled qubits

Definition 3.1.1: A Bell pair is a two qubits quantum system entangled in one of the four possible Bell states. The Bell pair obtained from entangling the state $|00\rangle$ is $|\phi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$

To build this Bell pair, we use two qubits in state $|0\rangle$ and, by applying a H gate to the first one, we obtain the state

$$|\varphi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{3.1}$$

$$= \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle) \tag{3.2}$$

If now we apply a CNOT gate, that changes the state $|10\rangle$ to $|11\rangle$, we obtain

$$|B\rangle = CNOT \star |\varphi_1\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix} \tag{3.3}$$

Now we have the desired Bell pair -and we will denote it as $|B\rangle$. If we combine the qubit to teleport to this Bell pair we obtain the following three-qubit quantum state:

$$|\varphi_2\rangle = |\psi\rangle \otimes |B\rangle = (\alpha|0\rangle + \beta|1\rangle) \otimes \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{3.4}$$

$$= \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \otimes \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \left[\frac{\alpha}{\sqrt{2}}, 0, 0, \frac{\alpha}{\sqrt{2}}, \frac{\beta}{\sqrt{2}}, 0, 0, \frac{\beta}{\sqrt{2}} \right]^T \tag{3.5}$$

$$= \frac{1}{\sqrt{2}}(\alpha|000\rangle + \alpha|011\rangle + \beta|100\rangle + \beta|111\rangle) \tag{3.6}$$

Remember that Alice has the first two qubits and Bob has the last one. Now, after Alice applies a CNOT gate to her qubits, our global state is

$$|\varphi_3\rangle = (CNOT \otimes Id) \cdot |\phi_2\rangle = \left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \cdot |\phi_2\rangle \tag{3.7}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \frac{\alpha}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{\alpha}{\sqrt{2}} \\ \frac{\beta}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{\beta}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} \frac{\alpha}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{\alpha}{\sqrt{2}} \\ 0 \\ \frac{\beta}{\sqrt{2}} \\ \frac{\beta}{\sqrt{2}} \\ 0 \end{bmatrix} \tag{3.8}$$

$$= \frac{1}{\sqrt{2}}(\alpha|000\rangle + \alpha|011\rangle + \beta|110\rangle + \beta|101\rangle) \tag{3.9}$$

An equivalent expresion is

$$|\varphi_3\rangle = \frac{1}{\sqrt{2}}(\alpha|0\rangle \otimes (|00\rangle + |11\rangle) + (\beta|1\rangle \otimes (|10\rangle + |01\rangle) \tag{3.10}$$

Next, Alice applies a H gate to $|\psi\rangle$, so the state of the whole quantum system

becomes

$$|\varphi_4\rangle = (H \otimes Id \otimes Id)\left(\frac{1}{\sqrt{2}}[\alpha|0\rangle \otimes (|00\rangle + |11\rangle)] + [\beta|1\rangle \otimes (|10\rangle + |01\rangle)]\right)$$

$$= \frac{1}{\sqrt{2}}\left(\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \otimes \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \otimes \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\right) \cdot \begin{bmatrix}\frac{\alpha}{\sqrt{2}}\\ 0\\ 0\\ \frac{\alpha}{\sqrt{2}}\\ 0\\ \frac{\beta}{\sqrt{2}}\\ \frac{\beta}{\sqrt{2}}\\ 0\end{bmatrix} \tag{3.11}$$

$$= \frac{1}{2}\begin{bmatrix}1 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1\\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1\end{bmatrix} \cdot \begin{bmatrix}\alpha\\ 0\\ 0\\ \alpha\\ 0\\ \beta\\ \beta\\ 0\end{bmatrix} = \frac{1}{2}\begin{bmatrix}\alpha\\ \beta\\ \beta\\ \alpha\\ \alpha\\ -\beta\\ -\beta\\ \alpha\end{bmatrix} \tag{3.12}$$

$$= \frac{1}{2}\left[\alpha(|000\rangle + |011\rangle + |100\rangle + |111\rangle) + \beta(|010\rangle + |001\rangle - |110\rangle - |101\rangle)\right]$$

$$= \frac{1}{2}[|00\rangle(\beta|1\rangle + \alpha|0\rangle) + |01\rangle(\alpha|1\rangle + \beta|0\rangle)$$
$$+|10\rangle(-\beta|1\rangle + \alpha|0\rangle) + |11\rangle(\alpha|1\rangle - \beta|0\rangle)] \tag{3.13}$$

Now is time for Alice to measure her two qubits. If Alice gets, for example, $|00\rangle$ as a result, then, due to the act of measuring, the two qubits collapse to this state, and the whole quantum system is in the state

$$|\varphi_5\rangle = |00\rangle(\beta|1\rangle + \alpha|0\rangle) \tag{3.14}$$

That implies that Bob's qubit is in the state

$$|z\rangle = \beta|1\rangle + \alpha|0\rangle \tag{3.15}$$

and that is exactly the state that Alice wanted to send to Bob. Notice that only in the case that Alice obtains $|00\rangle$ in her measurament we have sent this state. If Alice obtains, for example, $|01\rangle$, then the whole state is

$$|\varphi_5\rangle = |01\rangle(\alpha|1\rangle + \beta|0\rangle) \tag{3.16}$$

So Bob's qubit is not exactly in the same state that Alice wanted to send to him. In this case, Bob needs to apply some transformations to get the

state $|\psi\rangle$. If the first qubit that Alice measures is on the state $|1\rangle$, then we apply a Z gate to Bobs qubit. If the second qubit is on state $|1\rangle$, we apply a X gate, and if both are in state $|1\rangle$, then we apply both gates, ending in the state

$$|\varphi_6\rangle = |\Phi\rangle(\beta|1\rangle + \alpha|0\rangle) \tag{3.17}$$

where $|\Phi\rangle$ denotes one of the states $|00\rangle$, $|01\rangle$, $|10\rangle$ or $|11\rangle$. Therefore, if Bob's has to apply the correct gates, Alice has to send him the result of measuring her two qubits throught a classical channel. Note that in $|\varphi_6\rangle$ the initial state qubit of $|\psi\rangle$ has been destroyed.

Another implementation of the algorithm using ProjectQ can be found in Figure 3.2, in which the qubit that Alice wants to send is in a superposition of states given by the Hadamard gate.

```
from projectq.ops import H, CNOT, X, Z, Measure
from projectq import MainEngine

# Setting up the simulator
eng = MainEngine()

# Creating the qubits to operate
mystery_qubit = eng.allocate_qubit()
alice_qubit = eng.allocate_qubit()
bob_qubit = eng.allocate_qubit()

# Arbitatry transformations to mystery qubit for test purposes
H | mystery_qubit

# Entagle qubits to form phi +
H | alice_qubit
CNOT | (alice_qubit, bob_qubit)

# Interact with the entangled qubit
CNOT | (mystery_qubit, alice_qubit)
H | mystery_qubit

# Measurig Alice qubit
Measure | mystery_qubit
Measure | alice_qubit
b1 = int(mystery_qubit)
b2 = int(alice_qubit)

# Recovering the original qubit
if b1 == 1:
  Z | bob_qubit
if b2 == 1:
  X | bob_qubit

# Doing the operations
eng.flush()

# Mesuring the final qubit
Measure | bob_qubit
print("Measured {}".format(int(bob_qubit))) # output measurement
    result
```

Figure 3.2: Quantum Teleportation in ProjectQ.

3.2 Deutsch's Algorithm

This algorithm is a toy example with two qubits that is used as the base to develop the Deutsch-Jozsa algorithm. The problem that is solved by this algorithm is to determine if a function is constant or balanced in functions with one bit input and one bit output. If we have a function $f : \{0, 1\} \rightarrow \{0, 1\}$, we call it constant if $f(0) = f(1)$, and balanced if $f(0) \neq f(1)$.

To solve this problem, we need to face another important problem. f is a function defined in the classical way, that means, if we represent f as a matrix, there is no need for this matrix to be unitary, so the gate represented for this matrix will not always be reversible. Remember that quantum gates must be reversible, so f can not be always a quantum circuit. When we apply this algorithm, we need to find an **oracle** -that we will call U_f- that is the quantum circuit that represents the function f but with an unitary matrix. We will not explain how to construct this quantum gates, to get more information about them, you should see the bibliography.

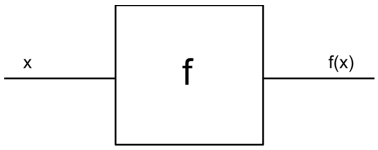

Figure 3.3: Classical Version of f

To undestand better the behaviour of the U_f circuit, let's explain how to build the reversible version of a classical one-qubit function. In figure 3.3 we see the classical circuit in with x is the input and, when we apply the circuit, outputs $f(x)$. The quantum version is different, we need two input qubits, $|x\rangle$ and $|y\rangle$, and apply the oracle U_f to them. When we do so, then the first qubit $|x\rangle$ does not change but the qubit $|y\rangle$ becomes $|y \oplus f(x)\rangle$ as we see in Figure 3.4. So, for example, if the qubit $|y\rangle$ is $|0\rangle$ as in Figure 3.5, then the result after applying the oracle is $|f(x)\rangle$, in other words, we have as a result a qubit with the input $|x\rangle$ and another with it's evaluation over f, $|f(x)\rangle$.

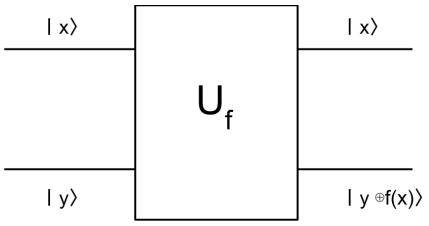

Figure 3.4: Quantum Computing of a Function f

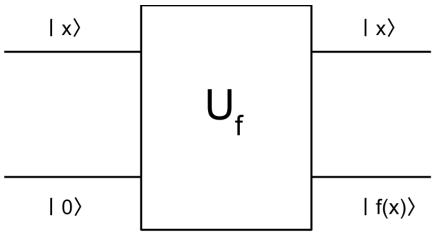

Figure 3.5: Application of the Circuit to Compute f

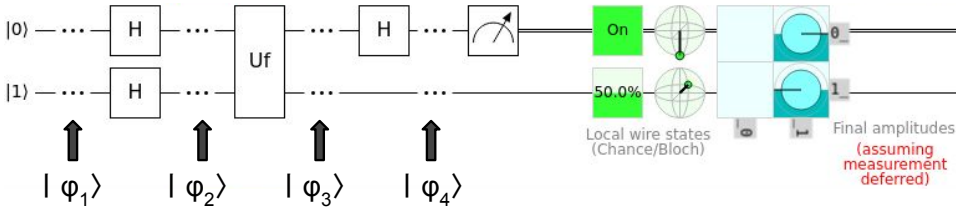

Figure 3.6: Deutch's Algorithm in Quirk.

Once we have the U_f gate, to figure if f is balanced or constant, we will use as input a uniform superposition of all basic states $\{|0\rangle, |1\rangle\}$ to evaluate the function in all posible states at the same time. We will use Figure 3.6 to follow the behaviour of the algorithm. We start with two qubits at the state $|\varphi_1\rangle = |01\rangle$ and then we apply a Hadamard gate to the both of them, getting the state

$$|\varphi_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle) \quad (3.18)$$

Note that by applying a Hadamard gate to the first qubit in state $|0\rangle$, we have obtained the desired uniform superposition of states $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. Applying the U_f gate, we get the state

$$|\varphi_3\rangle = \left[\frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \quad (3.19)$$

Note that, in some sense, we have stored the result of evaluation f in the amplitudes of the basic states of the first qubit. This is an example of **phase kickback**, which is where the eigenvalue added by a gate to a qubit is 'kicked back' into a different qubit via a controlled operation. In this state, if f is

constant, then $f(0) = f(1)$ and for so we get the state

$$|\varphi_4\rangle = (\pm 1)\left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \tag{3.20}$$

and if it is balanced, then we get the state

$$|\varphi_4\rangle = (\pm 1)\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \tag{3.21}$$

Applying a Hadamard gate to the first qubit, if f is constant we get the state

$$|\varphi_3\rangle = (\pm 1)|0\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \tag{3.22}$$

And if f is balanced, then we get

$$|\varphi_3\rangle = (\pm 1)|1\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \tag{3.23}$$

So, finally, to know if f is constant or balanced, we measure the first qubit. If we measure 0, then f is constant. Otherwise f is balance. In Figure 3.7 we show the implementation of Deutch's algorithm using ProjectQ. Note that, in the classical version of this problem, to determine if f is balanced or constant, we need to make two evaluations of f but in the quantum version we only need one thanks to the superposition. That means that the quantum version needs half of the evaluations of the classical version, so it is twice as fast as the classical version.

```
from projectq.ops import H, X, CNOT, Measure
from projectq import MainEngine

# Setting up the simulator
eng = MainEngine()

# Creating the qubits to operate
q1 = eng.allocate_qubit()
q2 = eng.allocate_qubit()

# Putting the q2 to state 1
X | q2

# Entangle states
H | q1
H | q2

# Uf gate
CNOT | (q1, q2)

# Calculating the result
H | q1

# Doing the operations
eng.flush()

# Getting the result
Measure | q1
Measure | q2
r = int(q1)

# Printing result
if r == 1:
    print("Uf is a non-constant function")
else:
    print("Uf is a constant function")
```

Figure 3.7: Deutsch's Algorithm in ProjectQ.

3.3 Deutsch-Jozsa Algorithm

This algorithm is a generalization of the previous one. In this case, we have a function $f : \{0,1\}^n \to \{0,1\}$. We say that f is balanced if exactly half of the outputs are 0 and the other half are 1 and we say that f is constant if all outputs are 0 or 1. For this problem, we assume that f can only be constant or balanced.

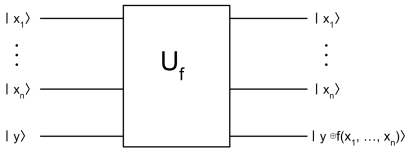

Figure 3.8: Quantum Computing of a n-qubit Function

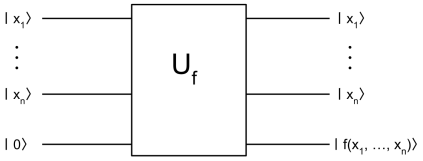

Figure 3.9: Application of the Circuit to Compute f

In order to build the quantum circuit for this algorithm, we need a generalization for the oracle U_f used in Deutch's algorithm because this oracle is only usefull for function with one input qubit. The generalization is the obviously one that can see in Figure 3.8, in which we have n input qubits and 1 output qubit. As with the other oracle, we see in Figure 3.9 that if the output qubit $|y\rangle$ of the oracle is at state $|0\rangle$, then the output is directly the evaluation of the funcion f.

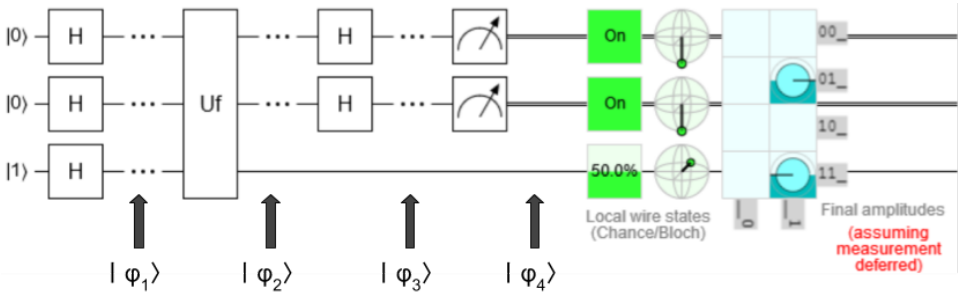

Figure 3.10: Deutch-Jozsa Algorithm in Quirk.

To build the circuit, we need $n+1$ initial qubits as we see in the Quirk implementation in Figure 3.10. We need to apply a Hadamard gate to all the qubits. We denote as $H^{\otimes n}$ to apply the Hadamard gate to n qubits. To make the notation easily, we will denote the first n qubits as $|\varphi\rangle$, so we have

$$|\varphi\rangle = H^{\otimes n} \star |00\cdots 0\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle \tag{3.24}$$

that is a uniform superposition of all basic states. With this notation, after the first step, we get the state

$$|\varphi_1\rangle = |\varphi\rangle \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \tag{3.25}$$

Apliying the U_f oracle, we get the state

$$|\varphi_2\rangle = |\psi\rangle \otimes \frac{|f(\psi)\oplus 0\rangle - |f(\psi)\oplus 1\rangle}{\sqrt{2}} = |\psi\rangle \otimes \frac{|f(\psi)\rangle - |f(\psi)\oplus 1\rangle}{\sqrt{2}} \tag{3.26}$$

if we denote $\overline{f(x)}$ as the opposite result of $f(x)$ (remember that f has only two possible outputs, 0 and 1) then we can rewrite our state as

$$\begin{aligned} |\varphi_2\rangle &= |\psi\rangle \otimes \frac{|f(\psi)\rangle - |\overline{f(\psi)}\rangle}{\sqrt{2}} && (3.27)\\ &= |\varphi\rangle \otimes \Big((-1)^{f(\varphi)}(|0\rangle - |1\rangle)\Big) && (3.28)\\ &= (-1)^{f(\psi)}|\psi\rangle \otimes \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] && (3.29) \end{aligned}$$

Replacing back the notation contraction using the equation 3.24, we can express our state as

$$|\varphi_2\rangle = \left[\frac{\sum_{x\in\{0,1\}^n}(-1)^{f(x)}|x\rangle}{\sqrt{2^n}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \tag{3.30}$$

Note how we are exploting the **quantum parallelism** obtained from evaluating a function in a superposition of states. We are only applying the function circuit once and computing the result of the function for all 2^n posible input states $x \in \{0,1\}^n$. Applying the Hadamard gates to a given state $|x\rangle$ yields us

$$H^{\otimes n} \star |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y\in\{0,1\}^n} (-1)^{\langle x,y\rangle}|y\rangle \tag{3.31}$$

where

$$\langle x, y\rangle = x_0y_0 \oplus \cdots \oplus x_{n-1}y_{n-1} \tag{3.32}$$

is the sum mod 2 of the bitwise product of x and y. So, if we apply again the n Hadamard gates to state $|\varphi_2\rangle$ we get

$$\begin{aligned}
|\varphi_3\rangle &= \left[\frac{\sum_{x\in\{0,1\}^n}(-1)^{f(x)}\frac{1}{\sqrt{2^n}}\sum_{y\in\{0,1\}^n}(-1)^{\langle x,y\rangle}|y\rangle}{\sqrt{2^n}}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \\
&= \left[\frac{\sum_{x\in\{0,1\}^n}\sum_{y\in\{0,1\}^n}(-1)^{f(x)}(-1)^{\langle x,y\rangle}|y\rangle}{2^n}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \\
&= \left[\frac{\sum_{x\in\{0,1\}^n}\sum_{y\in\{0,1\}^n}(-1)^{f(x)\oplus\langle x,y\rangle}|y\rangle}{2^n}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \qquad (3.33)
\end{aligned}$$

Now it's time to figure out the state $|\varphi_4\rangle$, but is very complex analyzing all the posible output states with respect to the input states, so let's change the main objective and try to figure out when we get get the first n qubits of $|\varphi_4\rangle$ equal to $|0\cdots0\rangle$. The probability is given with $|y\rangle = 0$ in state $|\varphi_3\rangle$. In this case, $\langle x, y\rangle = 0$ for all x, and so we have reduced $|\varphi_3\rangle$ to

$$\left[\frac{\sum_{x\in\{0,1\}^n}(-1)^{f(x)}|0\cdots0\rangle}{2^n}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \qquad (3.34)$$

Therefore, the state $|\varphi_4\rangle$ only depends on the evaluations of f. If f is constant at 1, then the top n qubits become

$$\frac{\sum_{x\in\{0,1\}^n}(-1)|0\cdots0\rangle}{2^n} = \frac{-2^n|0\cdots0\rangle}{\sqrt{2}} = -1|0\cdots0\rangle \qquad (3.35)$$

and if f is constant at 0, then they become

$$\frac{\sum_{x\in\{0,1\}^n}1|0\cdots0\rangle}{2^n} = \frac{2^n|0\cdots0\rangle}{\sqrt{2}} = |0\cdots0\rangle \qquad (3.36)$$

that means that the probability of measuring $|0\cdots0\rangle$ if f is constant is 1. On the contrary if f is balance, half of the amplitudes (-1) will cancel with the half $(+1)$, and we will get

$$\frac{\sum_{x\in\{0,1\}^n}(-1)^{f(x)}|0\cdots0\rangle}{2^n} = \frac{0|0\cdots0\rangle}{\sqrt{2}} = 0|0\cdots0\rangle \qquad (3.37)$$

Therefore, if we measure $|0\cdots0\rangle$, then f is constant, as long as there is no probability to measure this state if f is balanced. In this algorithm we

have seen how to use the **interference** to cancel probabilities of certain states. The more impresive thing about this algorithm is that in classical computing we need $2^n - 1$ evaluation of the function f to check if is balanced or constant, but in this algorithm we only need 1 evaluation of the function f, which means an impresive exponential speedup.

We can find another implementation of this algorithm in Figure 3.11 using ProjectQ with a balanced function.

```
from projectq.ops import H, X, Measure, CNOT
from projectq import MainEngine

# Constants
n = 4 #length of the problem

# Setting up the simulator
eng = MainEngine()

# Creating the qubits to operate
qubits = [eng.allocate_qubit() for _ in range(n+1)]

# Putting the last qubit to state 1
X | qubits[-1]

# Entangle states
for q in qubits:
    H | q

# Uf gate
i = 0
while i < len(qubits) - 1:
    CNOT | (qubits[i], qubits[i+1])
    i += 1

# Calculating the result
for i in range(n):
    H | qubits[i]

# Doing the operations
eng.flush()

# Getting the result
r = 0
for q in qubits:
    Measure | q
    r += int(q)
r -= int(qubits[-1])

# Printing result
if r == 0:
    print("Uf is a constant function")
else:
    print("Uf is a balance function")
```

Figure 3.11: Deutsch-Jozsa Algorithm in ProjectQ.

4

Conclusion and Future Work

Quantum computing can offer exponential speedups with respect to classical computing when applied to some problmes. This fact makes us rethink some classical problems without an efficient solution -NP and NP-Complete-, since in quantum computing these problems could have one quantum algorithm that solves them in a polynomial time. Many fields can be affected and greatly benefited by this speedup, fields such as artificial intelligence or medicine, since the algorithms that they execute are computationally very expensive. Quantum computing could break down these barriers imposed by classical computers' lack of computing power to solve these problems.

Nowadays, both quantum algorithms and quantum computers are still in a very early stage, so we can not use them with better results than a classical computer in any problem or situation. But this situation it may not last long and we may soon be able to apply quantum computing to solve practical problems. When that time comes, we must be ready to use the new computational power that these systems can offer us. For this reason, it is interesting to continue researching and developing new quantum algorithms for these kind of systems, which can range from new approaches to classical algorithms to adapting classical algorithms to the quantum paradigm.

At this point, we know much of the concepts necessary to develop more quantum algorithms applied to a specific field. In [4] we can see examples of quantum algorithms in the field of machine learning. As future work, such algorithms can be studied and implemented, such as quantum random walks or quantum minimal spaning tree.

Of the main quantum algorithms that are necessary to know, one of them is missing to study. As future work, the quantum Fourier transform will be studied, essential for the development of other algorithms such as HHL.

Bibliography

[1] Noson S. Yanofsky and Mirco A: Mannucci. Quantum Computing for Computer Scientist. Ca, bridge. 2008.

[2] P.A.M. Dirac (1947). The Principles of Quantum Mechanics (2nd ed.). Clarendon Press. p. 12.

[3] Jack D. Hidary: Quantum Computing: An Applied Approach. Mountain View, CA, USA. 2019.

[4] Abhijith J., Adetokunbo Adedoyin: Quantum Algorithm Implementations for Beginners. Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA.