National Polytechnic School

Control systems and Automation Department

Seminar

Quantum systems optimisation
HHL algorithm

Written by :

BOUDJOGHRA Mohamed El Amine

DAIMELLAH Sid Ali Sofiane

Contents

Introduction

Optimization is being considered as extremely crucial in control theory and controller design, where the main objective is arriving at the best controller that ascertain optimal energy consumption, which is mathematically represented as a cost function, in the aim of enhancing the conventional methods of optimization, implemented using classical computers based on conventional calculations using bits manipulations (such as gradient descent). Quantum computing offers numerous algorithms based on quantum-bits referred to as Qubits which are constructed in a way to have quantum characteristics such as being in a superposition of two different states $|0\rangle$ and $|1\rangle$ simultaneously. This feature allows an exponential speed up where we can see that a quantum optimization algorithm built using 300 Qubits, requires 2^{300} classical bits to be constructed which is impossible even using the latest supercomputers where the maximal storage size can never reach this number of bits. The details of these quantum characteristics in addition to the reason why they provide such a huge flexibility will be discussed in further sections.

Classical computers can run classical algorithms but can only simulate quantum algorithms that will perform 2^n slower because of the non-quantum nature of the regular computers. On the other hand, quantum computers can run both quantum and classical algorithms, since it is possible to choose whether to employ the quantum characteristics like quantum superposition and quantum entanglement -leveraged by quantum algorithms- or not. However, these characteristics, which are considered as discriminating features of quantum computing, provide the capability to perform specific computations exponentially faster. Thus, giving the ability to solve what so-called intractable problems that would take years to solve on a regular computer, in few minutes. As an example of algorithms that leverage those advantages, we find Shor's algorithm, used for prime numbers factorization, or Grover's algorithm used for searching an unstructured database. Furthermore, there are algorithms that aim to solve optimisation problems like the variational quantum eigensolver algorithm, where it focuses on gradually reaching an optimal state that corresponds to the eigenvector assigned to the minimum eigenvalue. This algorithm is run using both classical and quantum computers to compensate for the quantum computers' physical limitations like small coherence time which is the time for the information carried on the Qubits to perish, and the limited amount of Qubits carried by current quantum computers, which forces the CPU to use the stack of Qubits only for specific quantum tasks and perform the rest of the classical part on regular bits.

Quantum mechanics can be very confusing, and it can be extremely challenging to make sense out of its core concepts at first sight, since quantum objects belong to the microscopic world, which obey to a different set of rules in comparison to the rules that we are subject to in our daily life. Therefore, if classical computing may be very complicated, quantum computing is completely unintuitive. It is important for new learners to be aware of that in order to be able to be more open-minded to the idea of accepting the new concepts as they come and develop a different way of thinking that will be in line with the quantum world. Otherwise, the learning curve would be too massive to overcome, and the learning process would be a nightmare, As the learner will see very weird concepts like negative probabilities for example. Hence, even though quantum objects may seem random and chaotic at first, it is crucial to keep in mind that they simply follow a different set of rules, and once the learner has gone through those rules, the learning process will happen naturally.

Quantum mechanics was and still is the science of unusual phenomena that can help computer science develop more reliable computational methods easily, constructed to obey the quantum mechanical laws such as the previously mentioned entanglement phenomenon, these laws help computer science flourish its logic from an utterly different perspective to solve exceedingly complicated problems related to quantum particles. Nonetheless, this type of computation was heavily doubted in its early days due to the difficulty encountered while trying to manipulate the quantum particles like electrons and nucleus due to their instability when they are applied to any external form of energy created from an attempt of measuring.

Quantum computing began in 1980 when a physicist called Paul Benioff proposed a mathematical model of computation termed Turing Machine, which was the main subject of his book entitled "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines". This machine was founded to simulate any classical algorithm based on symbolic representation also in this book Benioff presented the theoretical basis of quantum computing and suggested that such a computer can be built. Afterwards, Yuri Manin laid out the core idea of quantum computing in his book "Computable and Non-computable".Later on Feynman explained in his lecture entitled "Simulating physics with computers" that a classical system can never adequately represent a quantum one as a reason of information loss and the huge differences between the two. Once the doors to this new field were opened by Benioff, Manin and Feynman,Other researchers became interested in investigating the possible opportunities offered by this freshly Born area. As a reason of that, several theoretical algorithms came to light. Thereafter, a physicist at Oxford university called David Deutsch had presented in his 1985 paper the first comprehensive framework for quantum computing. Besides, in his paper, he describes in detail what a quantum algorithm would look like in addition to explaining the possibility of technologically constructing a quantum computer in the future. He also aimed at developing an algorithm that would run faster on a quantum computer along with Richard Jozsa. This algorithm showed the tremendous power of quantum computers over classical ones. Eight years after, Bernstein and Vazirani developed an ameliorated version of Deutsch an Jozsa algorithm that functions even when small errors occur. This algorithm is being considered as powerful because it differentiates the quantum and classical computing when errors are present and that was not possible using Deutsch-Jozsa algorithm. In 1995, Peter Shor developed the first revolutionary quantum algorithm for factorizing integer numbers. Subsequently, this algorithm became a remarkable threat to communication protocols that rely on factoring integers for encryption. As an example, RSA-encrypted communications. Shor's breakthrough led more researchers to work on quantum algorithms since it then became clear that quantum computers, if built, would be extremely powerful. In fact, Shor's algorithm is one of the first to have been demonstrated on early quantum computers.

Not to mention the major breakthrough october, 23 2019 when Google AI in partnership with the U.S.National Aeronautics and Space Administration claimed to have performed a computation on a quantum computer that was considered to be infeasible on any classical computer.

Part I
Superposition, Entanglement and Reversibility

1 Quantum computer

A quantum computer is a calculator that applies the laws of quantum mechanics, quantum information theory and computer science. when the previous areas join together they form quantum computing science.

In computer science the classical laws are applied where a bit can either take the HIGH value or the LOW value where both represented as an electrical voltage, when the quantum mechanical laws take place, a bit can take the values $|0\rangle$, $|1\rangle$ or a superposition of both states $\alpha|0\rangle + \beta|1\rangle$. This quantum bit is referred to as Qubit and its logical state is the digital version of the analogical wave function noted $|\psi(t)\rangle$.

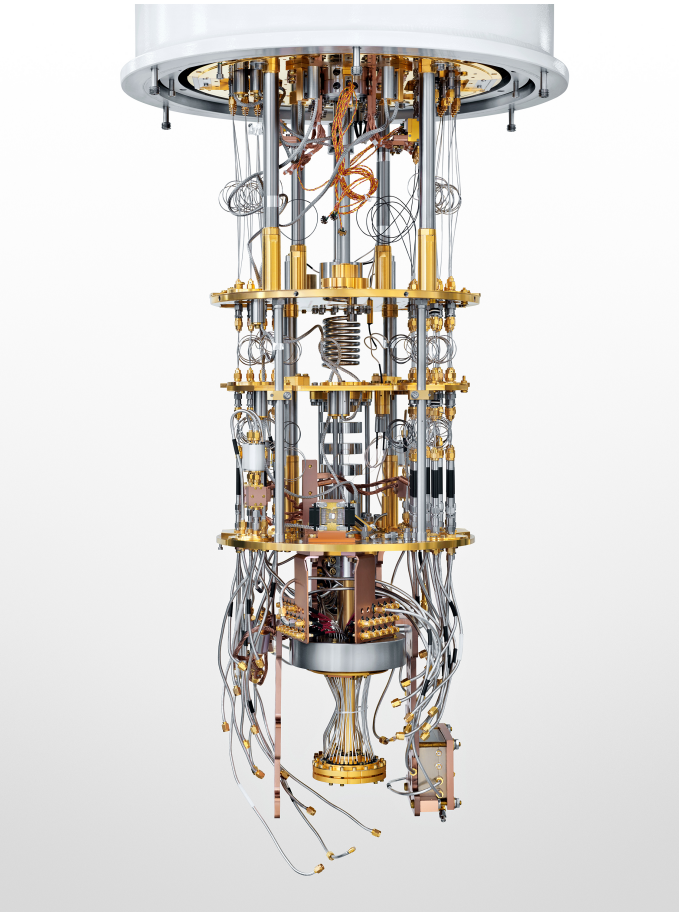


Figure 1: Quantum Computer

1.1 Components of Quantum Computers

1.1.1 Shell

When the computer is operational, five covers envelop the machine. These cans nest inside each other and act as thermal shields, keeping everything super cold which will allow the Qubits to persist.

Figure 2: Shell of a Quantum Computer

1.1.2 Nerves

Nerves are responsible of delivering signals to and from the chip, in order to allow quantum computations and measurement operations.

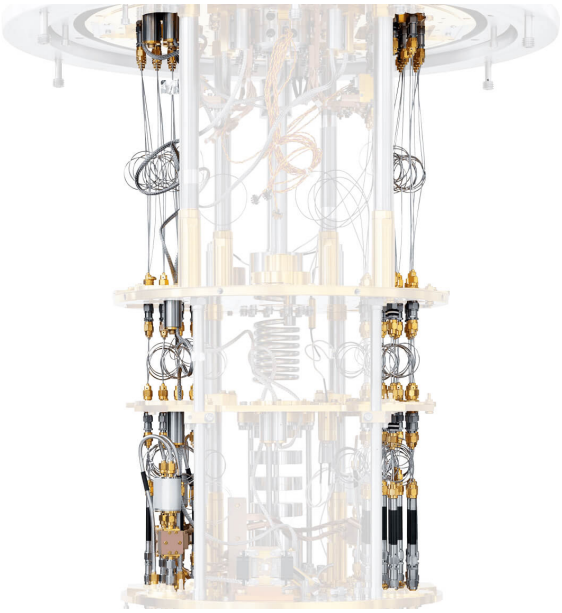


Figure 3: Nerves of a Quantum Computer

1.1.3 Skeleton

Those gold plates isolate cooling zones. Thus, allowing the chip to excessively drop the temperature to one-hundredth of a kelvin.

Figure 4: Skeleton of a Quantum Computer

1.1.4 Heart

In the center of a Quantum Computer, we find its heart, also referred to as the *mixing chamber*. Inside, it contains different forms of liquid helium separate and evaporate, to diffuse heat.

Figure 5: Heart of a Quantum Computer

1.1.5 Brain

The QPU (quantum processing unit) is made of gold-plated copper disk with a silicon chip inside that contains the machine's brain.

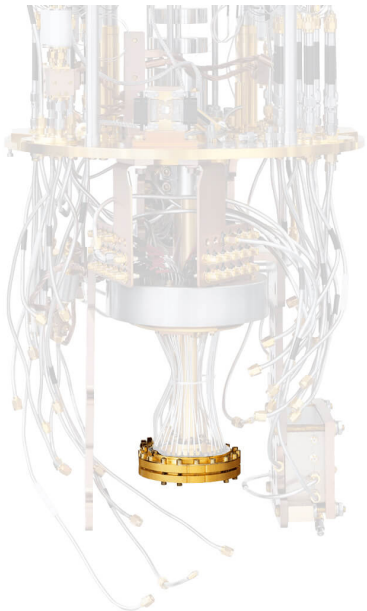


Figure 6: Brain of a Quantum Computer

1.2 The properties of a wave function in quantum mechanics

We use the following representation to describe the Qubit's state:

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

All of these properties are the ones characterising the solution of schrodinger equation which is called the wave function and labeled $|\psi(t)\rangle$, this wave function is defined as being a variable quantity that describes the wave characteristics of a quantum particle.

$$i\hbar\frac{\partial}{\partial t}\,|\psi(t)\rangle = \hat{H}\,|\psi(t)\rangle$$

In this section we will be discussing the numerous properties of a wave function in quantum mechanics

1.2.1 Linearity

The Schrodinger equation is a linear differential equation, which means that if two wave functions $|\psi_1\rangle$ and $|\psi_2\rangle$ are a solution, the linear combination of both is also a solution $|\psi_3\rangle = |\psi_1\rangle + |\psi_2\rangle$

$$\begin{cases} i\hbar\dfrac{\partial}{\partial t}\,|\psi_1(t)\rangle = \hat{H}\,|\psi_1(t)\rangle \\ \\ i\hbar\dfrac{\partial}{\partial t}\,|\psi_2(t)\rangle = \hat{H}\,|\psi_2(t)\rangle \end{cases} \quad \Longrightarrow \quad i\hbar\frac{\partial}{\partial t}\,|\psi_3(t)\rangle = \hat{H}\,|\psi_3(t)\rangle$$

1.2.2 Unitarity

For a constant Hamiltonian $\hat{H}$, the Schrodinger equation has the next solution:

$$|\psi(t)\rangle = e^{-i\frac{\hat{H}}{\hbar}t}\,|\psi(0)\rangle$$

The operator $\hat{U}(t) = e^{-i\frac{\hat{H}}{\hbar}t}$ is known as the time evolution operator and it is unitary it preserves the inner product between two vectors in a hilbert space.

2 The Superposition principle

This principle can be observed by looking at the linearity property where we can see that "the linear combination of two or more state vectors is another state vector in the hilbert space".

2.1 Photonic superposition

As an example we can take the photon, which is considered as a quantum particle with no fixed polarization.

Let's say that the photon's initial polarization state is written as a combination of two other states a horizontal state we label $|\uparrow\rangle$ with a probability of 50% and a vertical state we label $|\rightarrow\rangle$ with the same probability which means that it is directed 45 degrees to the top right corner, we note the polarization state of the photon with the greek symbol representing a wave function $|\psi\rangle$, we subsequently find:

$$|\psi\rangle = \frac{1}{\sqrt{2}}|\uparrow\rangle + \frac{1}{\sqrt{2}}|\rightarrow\rangle$$

after we use a horizontal polarizer filter to redirect the photon's polarization state to the horizontal direction, which means that the polarization state becomes:

$$|\psi\rangle = |\rightarrow\rangle$$

now we know for sure that the photon is directed to the horizontal direction which means that the amplitude of the state $|\rightarrow\rangle$ is 1 which represents a probability of 100% of finding the photon polarized to the horizontal direction.

Now we select another polarization state using another filter with an axis oriented 45 degrees to the top left corner, due to the same reason we find that the new polarization state becomes oriented to the filters polarization axis:

$$|\psi\rangle = |\nwarrow\rangle$$

we observe that the horizontal polarization can also be expressed as a superposition of two other polarization states in the same hilbert space:

$$|\rightarrow\rangle = \frac{1}{\sqrt{2}}|\nearrow\rangle - \frac{1}{\sqrt{2}}|\nwarrow\rangle$$

which refers to the next, if the photon is polarized to the horizontal direction it means that there is 50% chance of finding it in the $|\nwarrow\rangle$ polarization state and with the same probability of finding it in the $|\nearrow\rangle$ polarization state.

2.2 Probabilities and amplitudes

What it is being considered interesting here is the fact that the amplitudes of this linear combination of these numerous states can be negative which is shown in the previous superposition state, in quantum mechanics these amplitudes are complex and a given state is expressed with the mathematical formula :

$$\begin{cases} |\psi\rangle = \alpha|\rightarrow\rangle + \beta|\uparrow\rangle \\ \\ \sqrt{\alpha^2 + \beta^2} = 1(\text{the born rule}) \end{cases}$$

α^2: is the probability of the state $|\psi\rangle$ to collapse into the $|\rightarrow\rangle$ state.
β^2: is the probability of the state $|\psi\rangle$ to collapse into the $|\uparrow\rangle$ state.
α and β are both the complex numbers which represent the amplitudes of the state basis $|\rightarrow\rangle$ and $|\uparrow\rangle$ respectively, we use complex numbers instead of real positive numbers due to the properties of quantum particles where we can find that a particle like electron can have specific spin state and an orientation in the space which indicates that using one real number is inadequately not

possible, the details about the ways of representing the state of a Qubit with respecting a specific basis will be discussed subsequently in the section "Qubits representation".

3 Entanglement

3.1 Definition

If two particles or Qubits are entangled, it means that the measurement of the state of one of them is correlated with the state of the other. This correlation is much stronger than the classical correlation. In other words, they are considered inseparable. In quantum computing, entanglement can be acquired using two quantum gates which will be discussed in detail in the coming sections, The first gate is the Hadamard gate and the second one is the CNOT gate.

3.2 Utility

This peculiar characteristic of quantum particles is leveraged to perform a variety of computations and communications such as information encoding where the information is not encoded in both of them separately but rather in the correlation of the two.

4 Reversibility

4.1 Definition

Reversibility is related to the energy dissipated in the process of a basic unit of computation. Due to Rolf Launderer, the world now is certain that the amount of energy dissipated in a computing process is bounded under a specific limit called Launder's Bound. This bound is defined with the mathematical formula:

$$\Delta J = nkTln(2)$$

It explains the maximal amount of energy that can be spent while performing a specific computation using n bits in an environment of T temperature expressed in Kelvin.

4.2 Reversibility in quantum computing

Unfortunately, logical irreversibility where the state of an output does not uniquely define the state the input is accompanied by dissipative effects. This follows the second thermodynamics law where we observe a continuous increase in the entropy of a system which has an irreversible effect. We can find so many irreversible gates in classical computing like the OR gate where we can not tell the state of inputs In1 and In2 if the Output is HIGH since three possible combinations can result this state.

In1	In2	Out
0	0	0
0	1	1
1	0	1
1	1	1

In quantum computing, we limit ourselves to reversible logical operations only. If those operations are irreversible, it means that the amount of energy dissipated increases in addition to the information loss in case of decoherence. On the other hand, if the quantum operations are reversible it means that coherence can be maintained and information can be restored at any moment if we know the state of either the output or the input. When we say reversible, we are assuming a theoretical noiseless quantum computer. In a noisy quantum computer that decoheres, we cannot, of course, reverse the operation.

All quantum gates are reversible and they will be treated separately in the coming chapters, to clarify the idea we will be using the next example which is called the NOT quantum gate where its main function is to invert the state of the input and we can easily visualize that the operation is reversible where the input state can be obtained by a multiplication of the output state vector by the inverse of the NOT matrix.

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} . \begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} . |1\rangle$$

$$|1\rangle = \begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} . \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} . |0\rangle$$

Part II
Physical Qubits

Now we have covered the essential workings of a quantum computer. Next, we will delve into the depth of its physical internal architecture. There are numerous types of architectures each with its pros and cons, to develop a deep understanding of the paradigms related to quantum optimization. We will focus on its crucial physical units like the physical Qubit.

The quantum architectures we will be discussing subsequently require a classical system to function properly where the readout of the data from the quantum system is represented in a classical form then it is fed to the classical system for further processing and then the final instruction is used by the classical system to control the quantum part and apply the necessary quantum operations.

The next figure shows the architecture of a superconducting quantum computer controlled by an external classical computer. The classical computer labeled HOST in the figure, is used for HIGH-LOW level conversion of code instructions, then feeding them to the following superconducting quantum computer.

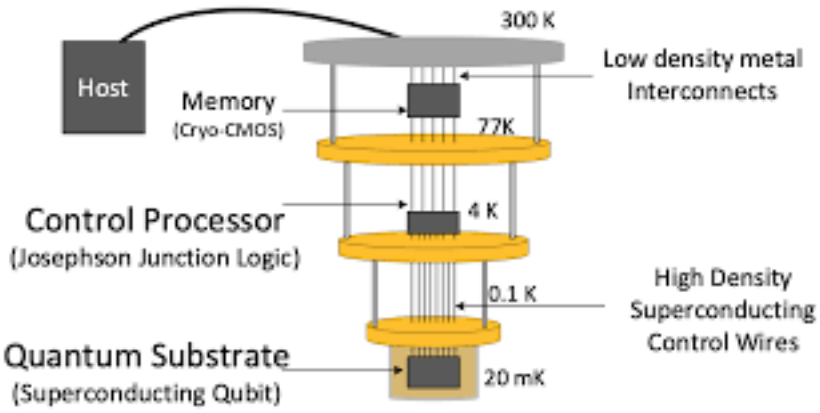


Figure 7: The architecture of a superconducting QC

The same logic is applied while executing Shor's algorithm, where many tasks are performed by the classical computer and the implementation of the period-finding algorithm is sent as a subroutine to the quantum computer to be executed. Afterwards, the results will be sent back to the classical computer. Finally, a quantum computer can be thought of as a subroutine in a much larger computation only for executing the hard parts of the program.

5 Assessing a quantum computer(DiVincenzo criteria):

The process of building a quantum computer must respect a set of rules. Otherwise, the built quantum computer may be futile.

5.1 Universality

The most important question to ask at the beginning of the quantum computer test is whether the machine is universal or Turing-complete. This requirement is important due to the high sensitivity of quantum particles such as the obligation of realizing all quantum bits as individually addressable for the reason of the huge correlation between them where measuring each one is important. And sometimes, measuring a Qubit can affect the information carried on another one if they are entangled.

5.2 Fidelity

Is it presented to be the ability of a Qubit to maintain the useful information, we refer to it as staying in a coherent state as long as possible.

5.3 Scalability

It is important to examine the ability of a quantum computer in reaching a scale of 10^6 Qubits and beyond or a fault-tolerant platform.

5.4 Qubits

Maintaining coherence in larger numbers of Qubits might be a technical challenge as a result of cross-talk between neighbor Qubits while trying to operate on one of them. Since quantum particles are extremely sensitive to external energy and sometimes they can be found in an entangled state.

5.5 Circuit depth

This refers to how many operations we can implement on a quantum computer before coherence breaks down(we remind that coherence means the maintain of information). Constructing a quantum computer with a large number of Qubits would be great but if the coherence time is small, it is of no use to apply a variety of operations, because the exact results would never be generated.

5.6 Logical connectivity

Limited logical connectivity requires Logical SWAP gates to be inserted into our algorithms to effectively simulate greater connectivity. Also, we know that more operators means larger error rates.

6 Types of Qubits

Unlike the nature of bits where only two state representation is required, a voltage of 0V to indicate the 0 state and 5V to indicate the 1 state will suffice. However, in quantum computing a Qubit can take an infinity of states which means an utterly new physical representation is mandatory,to accomplish this representation, researchers have been working on it tirelessly to reduce the errors occurred due to the high sensitivity of quantum particles. New approaches are coming out continuously. Here, we will present some of them.

Qubits can be realized with different ways each type has its pros and cons we will find out how to construct these Qubits in each way separately.

6.1 Neutral atoms

Neutral atom systems present an intriguing approach to quantum computing. While trapped ion research has been going on for some time, several labs have remarkably ramped-up their ability to control ensembles of trapped ions.

Engineers can set up a trap made of four laser beams to trap an ensemble of ions. This technique can help to cool the group of atoms down considerably to mili-kelvins. After cooling it down, the addressing process becomes possible where an array of ions is chosen and the selection will be with a simple translation of the four laser beams. Furthermore, a superposition can be obtained by positioning the laser between two atoms as it is shown in Figure 3.

Neutral atom systems meet the DiVincenzo criteria where all atoms are separable and addressable. Furthermore, we can apply a measurement on them.

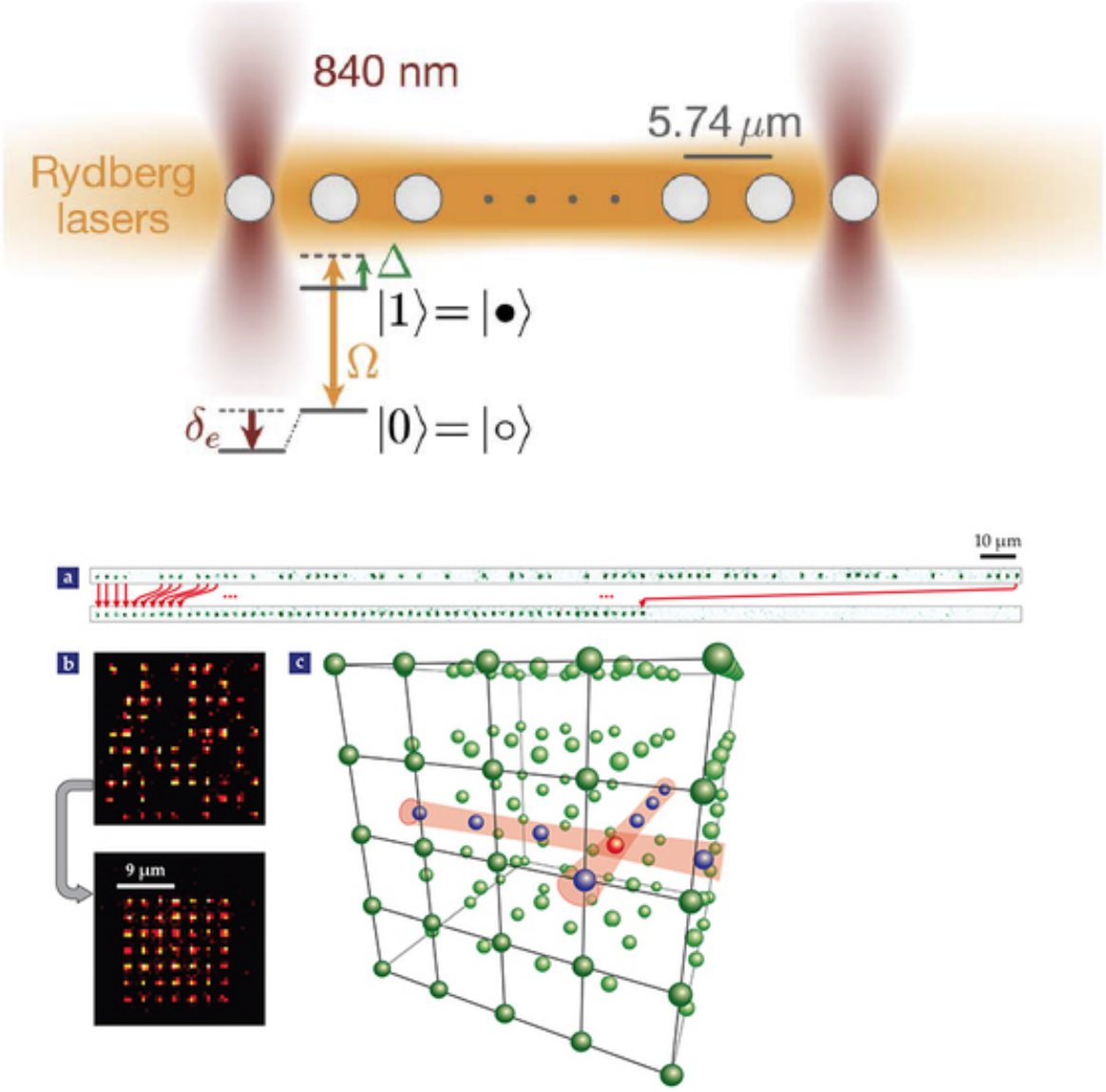


Figure 8: Addressing a neutral atom as Qubit

6.2 Nuclear Magnetic Resonance

This method is used to measure the spin of an electron, since it is considered as a tiny magnet, in case of applying a constant strong magnetic field the spins of all electrons within a specific matter in the field will be oriented to this magnet's direction. We subsequently perturb the electrons using an electro-magnetic field with a specific frequency. As a result, we observe a generated electromagnetic field with a frequency proportional to the geometrical configuration of the electron.

As an example, we take the case of the chemical element C_2H_5OH. After measuring the generated magnetic field and applying the Fourier transform to it, we visualize the next frequency diagram representing the amplitude of every group of electrons in the element where amplitude refers to spin value of the three groups. The amplitude changes as a reason to the difference in the number

of electrons where the magnitude value increases when the number of interacting particles does the same as a consequence of the magnetic fields superposition property.

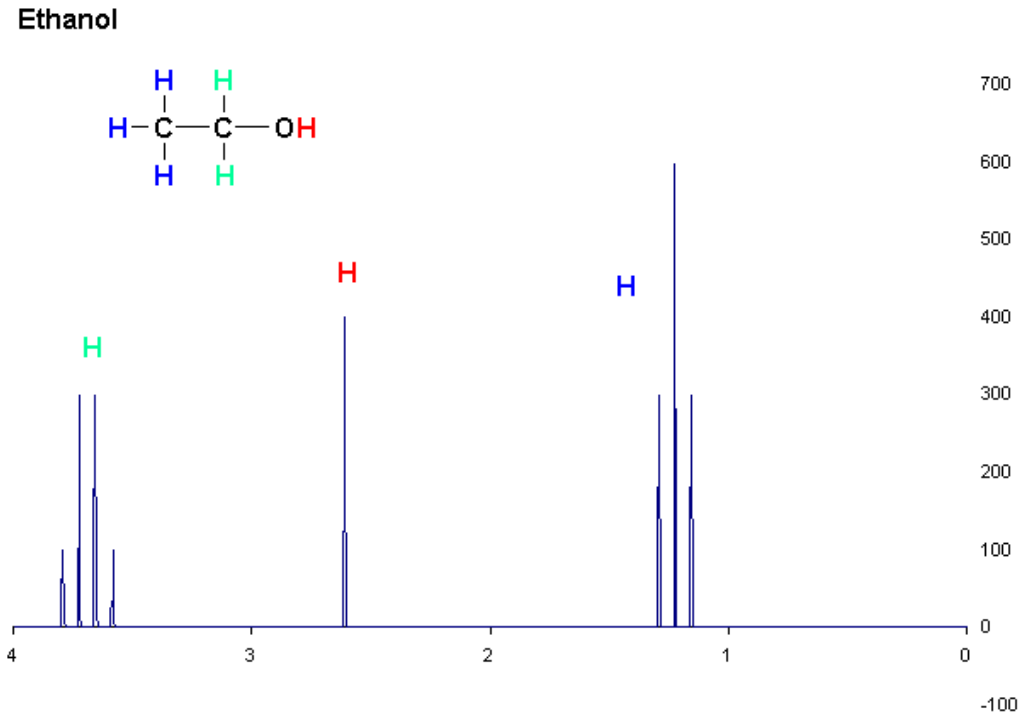


Figure 9: NMR diagram of ethanol element

To summarize, this method helps us reading the spin value of an electron in a particular matter, which means that we can use them as a readable Qubits. This type of Qubits was used to factor the number 15 into prime numbers using a quantum computer based on that type.

6.3 Nuclear Vacancy center-in-diamond

This is an approach to quantum computing. When two carbon atoms are missing from the carbon lattice in the diamond, one of them is replaced with a nitrogen ion. This defect creates para-magnetic effect that can act as a Qubit besides the leverage of manipulating it using microwaves and optically reading the state of the nitrogen spin out.

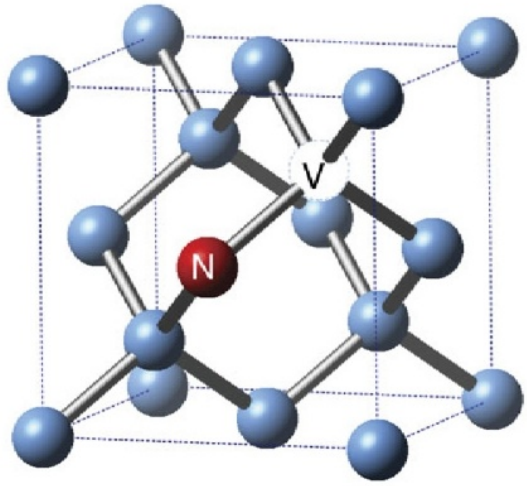


Figure 10: NV center-in-diamond Qubit

6.4 Spin Qubits

This is another interesting approach to quantum computation based on silicon doping. In case our technological knowledge arrives at constructing spin Qubits, which can be manipulated using CMOS transistors. This means, the ability of applying a decade of Know-How in the integrated chip industry on this freshly born field.

Even though this might be interesting to realize, it is being considered difficult to construct stable Qubits based on CMOS platform. Numerous attempts have been made in accomplishing that but they were just small prototypes with a difficulty of stabilizing them for a scale up.

In the following figure we present a Simplified three-dimensional schematic of a silicon-on-insulator nano-wire field-effect transistor with two gates, gate 1 and gate 2. Using a bias tee, gate 1 is connected to a low-pass-filtered line, used to apply a static gate voltage V_{g1}, and to a 20GHz-bandwidth line, used to apply the high-frequency modulation necessary for Qubit initialization, manipulation and read-out. .

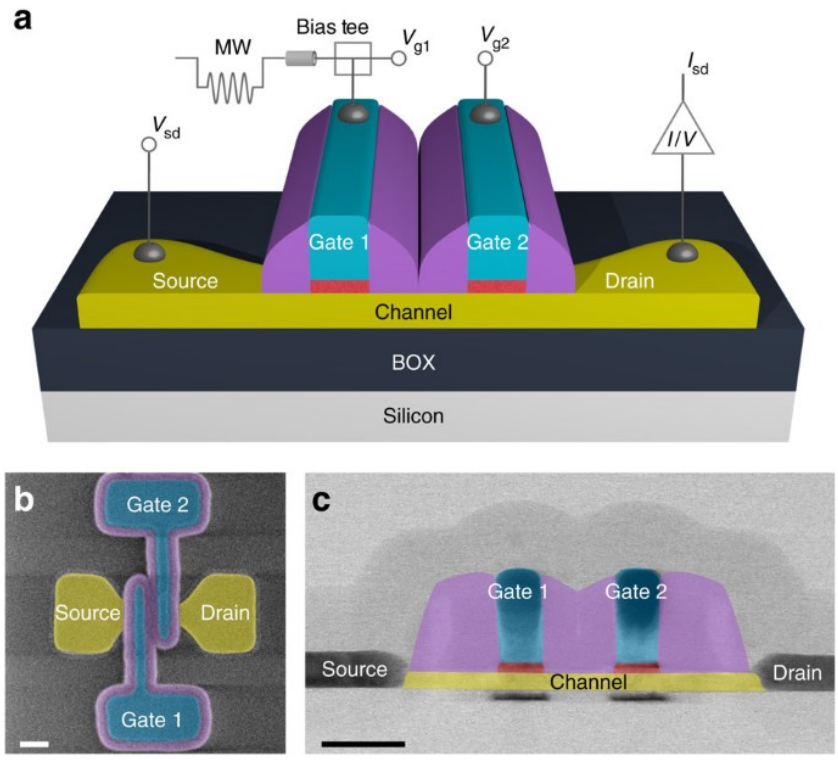


Figure 11: CMOS silicon spin Qubit

6.5 Superconducting Qubits

This approach to quantum computing uses a Qubit created with a cooper pair with a Josephson junction. Microwave leads are attached to the Qubit to control it. More importantly, the user can apply a range of operators by sending microwave pulses. However, the whole appartus must be cooled down to 10mK to operate.

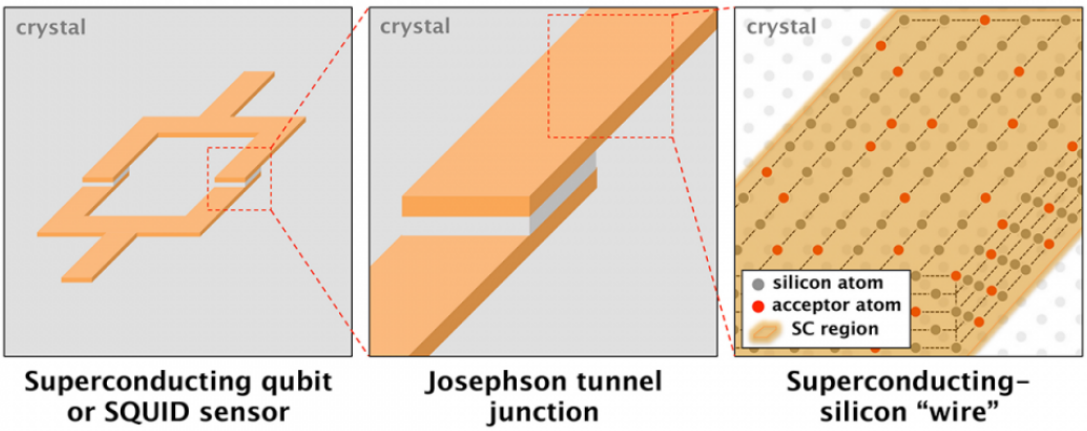


Figure 12: Superconducting Qubit

Number of groups are working on superconducting Qubit quantum computers including: Google, IBM, Rigetti, QCI and others

Part III
Qubits, operators and measurements

Quantum physics is a term widely used but much less understood. It is a mathematical model first used to describe the behavior of small things in a laboratory, which exposed gaps in the preceding theory of *classical physics*. Quantum theory explains this behavior and gives us a more complete picture of our universe. We have realized we can use this previously unexplained behavior to perform certain computations that we previously did not believe possible. We call this quantum computing. Quantum computing is the perfect way to dip your toes into quantum physics. It distills the core concepts from quantum physics into their simplest forms, stripping away the complications of the physical world. This kind of computations is characterised by a new kind of unitary elements : Qubits.

Whether we are using Qubits or regular bits, we need to manipulate them in order to turn the inputs we have into the outputs we need to be able to perform a wanted task. To do so, we need to make those inputs go through a series of operations which we call *gates* in ordrer to come out with the appropriate outputs that we can extract using some unique ways of *measurement*, in a specific *representation*. In quantum mechanics, we represent Qubits' states as vectors, operators (or gates) as matrices, and we use dirac notation instead of traditional linear algebra symbols to represent vectors and other abstractions as we will see later in this chapter.

7 Qubits and Hilbert space

7.1 What is a Qubit ?

The *Qubit* word can be decomposed into two parts: *qu*, which refers to quantum, and *bit*, also known as *binary digit*. Bits are designed to be the world's simplest alphabet. Using only two characters (0 and 1) we can represent any piece of information. However, even though most classical and quantum computers are known to be built upon bits and Qubits respectively, there exist other types of architectures like *qutrits*, which are three-level systems, and *qudits* which are general systems that could have any number of levels. Note that the complexity goes up when we try to work with systems which have several levels. It is the simplicity of the binary basis that made regular bits and Qubits so widely used when building computers.

Qubits are known to be the extension of regular bits to quantum mechanics. They are two-level systems. Same as bits, they can take either 0 or 1 as a state, with the difference that Qubits can also be in a continuous range of values representing a superposition of states. A physical Qubit is a two-level quantum mechanical system. As we saw in the earlier chapter, there are many ways to construct a physical Qubit. We can *represent* a Qubit as a two-dimensional complex Hilbert space, C^2. The state of the Qubit at any given time can be represented by a vector in this complex Hilbert space.

7.2 The Hilbert space

The Hilbert space is a vector space equipped with the inner product, an operation that allows lengths and angles to be defined. This will allows us to determine the relative position of two vectors representing Qubit states. We denote the inner product of two vectors $|x\rangle$, $|y\rangle$ as $\langle x\,|y\rangle$. Knowing the two vectors have the same origin, their inner product will equal 0 if they are orthogonal, and 1 if $|x\rangle$=$|y\rangle$.

Note : the $|x\rangle$ notation is known as the ket notation, which is one of the dirac notations that we will encounter as we go along this work.

To represent two or more Qubits we can tensor product Hilbert spaces together to represent the combined states of the Qubits. As we shall see, we have methods to represent separable states, where the Qubits are independent of one another, and entangled states such as a Bell state, where we cannot separate the two Qubit states.

8 Circuit diagrams

As mentionned earlier, computer calculations in general are based on manipulating a set of inputs through different kind of operators in order to have the wanted values as outputs. For simple programs with few bits, it is often useful to represent this process in a diagram known as *circuit diagram*. These have inputs on the left, outputs on the right, and operations represented by arcane symbols in between. A well known example of what a standard, bit-based circuit would look like is the half adder.

8.1 Classical circuit diagrams

Here is what a classical circuit based on traditional XOR and AND looks like :

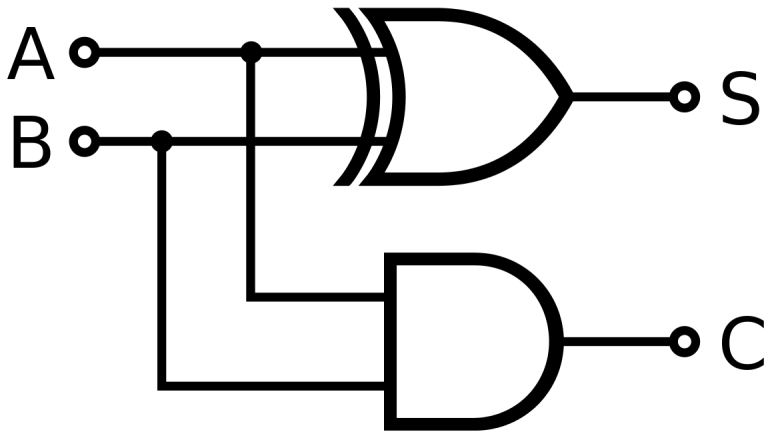


Figure 13: Traditional circuit diagram of a half adder

8.2 Quantum circuit diagrams

For quantum computers, we use the same basic idea but have different conventions for how to represent inputs, outputs, and the symbols used for operations. Here is an equivalent quantum circuit of the half adder :

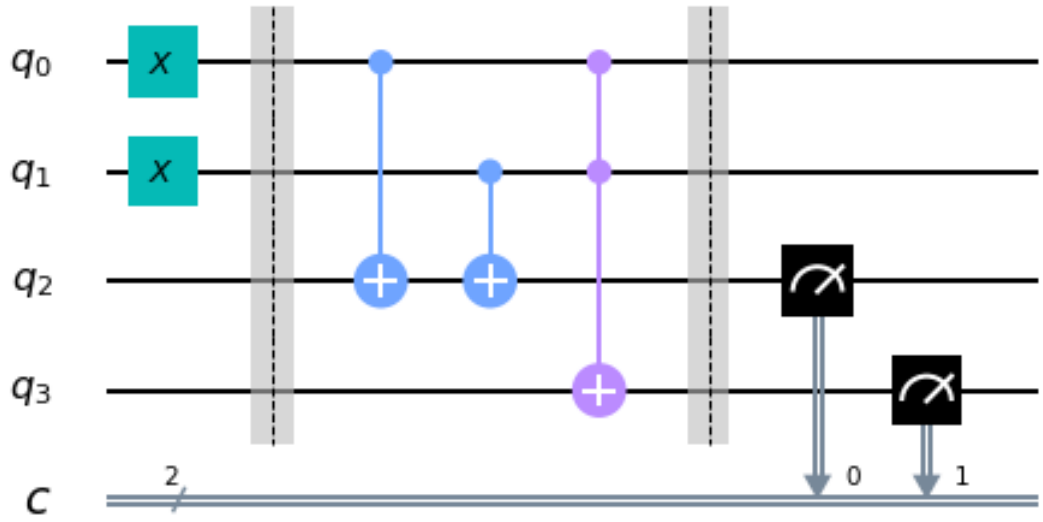


Figure 14: Quantum circuit diagram of a half adder

We will get to understand the building blocks of quantum diagrams by going through the core concepts these blocks are representing.

9 Qubits state representation

In order to be able to manipulate Qubits to perform computations and come out with some wanted result, it is necessary to find a way to describe the state of a Qubit at any point in time. To achieve that, there are some state representations used in quantum mechanics that we will see in this section.

9.1 Statevectors

Statevectors are used to represent the state of quantum physics systems. They are a collection of complexe numbers describing the probability of our system being in a certain state. In classical systems, this is not a very smart representation since the system can only be at one state at a time, that is for most cases pretty straightforward to determine. So, it is pointless to keep track of huge vectors when we only need one number. However, in quantum systems, *statevectors* happen to be a very good way of keeping track of quantum systems like Qubits.

Classical bits always have a completely well-defined state: they are either 0 or 1 at every point during a computation. In quantum computing, we see this as a restriction that needs to be lifted in

order to leverage the power of quantum rules. To do so, we perform *measurement* operations only when we want to extract the output of a quantum computation. Because, when a measurement is performed, Qubits must commit to one of the two options : 0 or 1. At any other time, its state will be something more complex than can be captured by a simple binary value, it will be a linear combination of the $|0\rangle$ state and the $|1\rangle$ state.(note that the $|0\rangle$ is the ket notation which is part of the bra-ket notation introduced by Dirac).

The $|0\rangle$ and $|1\rangle$ can form an orthonormal basis for all other possible states of a Qubit, as they are two exclusive Qubit states which do not overlap. Either the Qubit output 0 with a 100% confidence, or it definitely outputs 1. We can represent those states using the *statevector* representation as follows :

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Still, the main point of such a representation is the ability to represent more complex states, for example :

$$|q\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} \end{bmatrix}$$

By observing the statevectors $|0\rangle$, $|1\rangle$ and $|q\rangle$, we can clearly notice that $|q\rangle$ is a linear combination of the other two (this is an expected result since $|0\rangle$ and $|1\rangle$ form an orthonormal basis for all possible states). With that, we can write :

$$|q\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \frac{i}{\sqrt{2}} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Therefore :

$$|q\rangle = \frac{1}{\sqrt{2}} |0\rangle + \frac{i}{\sqrt{2}} |1\rangle$$

The vector $|q_0\rangle$ is called the Qubit's *statevector*, it tells us everything we could possibly know about this Qubit. For now, we are only able to draw a few simple conclusions about this particular example of a *statevector*: it is neither entirely $|0\rangle$ nor entirely $|1\rangle$. Instead, it is described by a linear combination of the two. In quantum mechanics, we typically describe linear combinations such as this using the word *superposition*.

To make sense out of this *statevector* representation, we need to calculate some probabilities. For example, the $|1\rangle$ vector has a probability of 100% of outputing the value 1, whereas the $|q\rangle$ *statevector* has his probabilities equally split when outputing one of the two binary values.

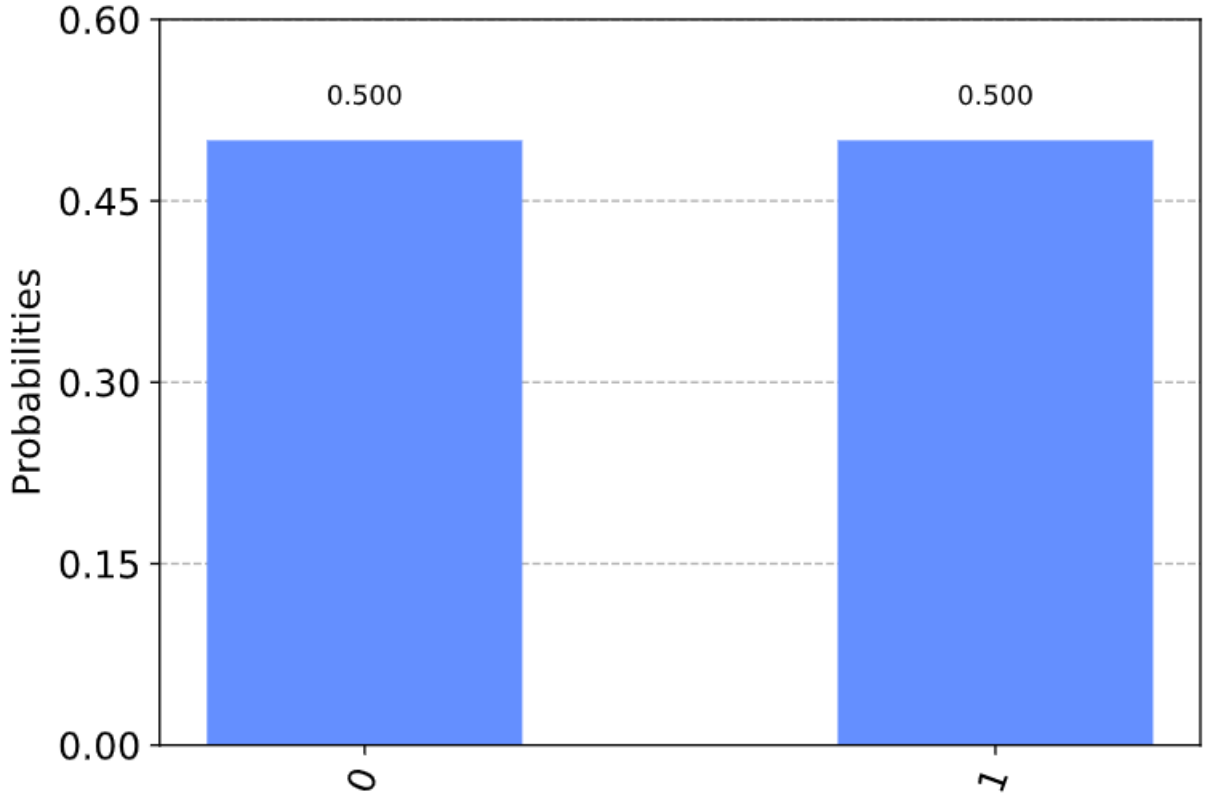


Figure 15: Output probabilities of the Qubit q

To understand how these probabilities are calculated, we need to introduce a new concept : *measurement*.

9.2 Measurement

Even though in classical physics, the act of measurement is an exceedingly straightforward process, where we assume that measurements do not affect the state of the system. This is not necessarily the case in quantum mechanics, where the act of measurement has a profound effect on the observation, as we will see in this section.

9.2.1 The important rule of measurement

To find the probability of measuring a state of the Qubit $|q\rangle$ in the state $|x\rangle$ we apply this simple rule :

$$p(|x\rangle) = |\langle x\,|q\rangle\,|^2$$

the $\langle x|$ representation is called the bra notation, together with the ket notation seen earlier, they form the Dirac bra-ket notation. A vector represented with the bra notation is a row vector, whereas we represent column vectors with the ket notation. Note that every ket $|a\rangle$ vector has his corresponding bra $\langle a|$ vector, and we convert between them using the conjugate transpose.

In the equation above, $\langle x|$ could be possible state of the Qubit. So, back to our example, calculating the probability of having $|0\rangle$ as an output from the Qubit whose *statevector* is $\langle q|$ would be as follows :

$$|q\rangle = \tfrac{1}{\sqrt{2}}|0\rangle \ \tfrac{i}{\sqrt{2}}|1\rangle$$

$$\langle 0|q\rangle = \tfrac{1}{\sqrt{2}}\langle 0|0\rangle + \tfrac{i}{\sqrt{2}}\langle 0|1\rangle$$

$$\langle 0|q\rangle = \tfrac{1}{\sqrt{2}} \cdot 1 + \tfrac{i}{\sqrt{2}} \cdot 0$$

$$\langle 0|q\rangle = \tfrac{1}{\sqrt{2}}$$

$$|\langle 0|q\rangle|^2 = \tfrac{1}{2}$$

Since this rule describes how we get information out of quantum bits, by calculating probabilities, which will be the bridge between what happens in the quantum world and the classical one. It is important to notice some implications of this rule in order to develop a certain intuition with the quantum world.

9.2.2 Implications of the rule

9.2.2.1 Normalisation

As seen earlier, a statevector is composed of elements that are related to probabilities. We call these elements *amplitudes*. From the rule exposed above, it is simple to notice that the probability of measuring one of the two states ($|0\rangle$ or $|1\rangle$) is its amplitude squared. Moreover, we know that the probabilities of all possible output values should sum up to one. Therefore, the magnitude of the state vector needs to be *normalized* to 1 :

$$\langle \Phi|\Phi\rangle = 1$$

Thus if :

$$|\Phi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Then :

$$\sqrt{\alpha^2 + \beta^2} = 1$$

This explains the $\sqrt{2}$ that we saw earlier in the $|q\rangle$ statevector. In fact, if we took the statevector $|q\rangle$ as follows :

$$|q\rangle = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Then, the magnitude of $|q\rangle$ would be :

$$\langle q\,|q\rangle = \sqrt{1+1} = \sqrt{2}$$

So, in order to *normalize* the statevector $|q\rangle$, we need to divide it by its magnitude $\sqrt{2}$. Which gives us :

$$|q\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

We will discuss about the imaginary operator i of the earlier $|q\rangle$ state vector in the upcoming implications.

9.2.2.2 An important note

The measurement rule gives the probability $p(|x\rangle)$ that a state $|q\rangle$ is measured as $|x\rangle$. Notice that $|x\rangle$ does not necessarily need to be $|0\rangle$ or $|1\rangle$, it can be any possible statevector. Also, there exists an infinite number of orthogonal pair of states where we can define a measurement that would cause the Qubit to choose between these two states.

9.2.2.3 Relative phase and Global phase

Let's first form a new orthonormal basis by introducing a new orthogonal pair states that are in a superposition of states :

$$\begin{cases} |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{cases}$$

Notice that these two states differ with a minus sign on the $|1\rangle$ state. Since the statevector's amplitudes are complex numbers, the minus sign is interpreted as a rotation of π in the complex plan. For complex numbers, we call this rotation a *phase*. More specifically, in statevectors representation, the difference of phase of its different amplitudes is referred to as *relative phase*.

Relative phases are considered as one of fundamental features of quantum calculations, as they allow *constructive interference* and *destructive interference*. Let's clarify these concepts by an example. Let's evaluate the sum of the above states :

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

Here, we see that the difference in the relative phase of the two states causes the $|1\rangle$ to cancel out. We say that the amplitudes of the state $|1\rangle$ interfere *destructively*. On the other hand, the amplitudes of the $|0\rangle$ state interfere *constructively* as they sum to 1 because of their same relative phase.

Notice when we calculate the probabilities of having either 0 or 1 as a final output, we will find that the two states $|+\rangle$ and $|-\rangle$ have both 50% chance of outputting either 0 or 1. Therefore, relative phase has no effect when measuring a certain state, still, it has a huge impact when doing computations as it allows *destructive interference* which is used to completely delete the probability of unwanted answers for the considered problem.
Now, let's subtract the two states :

$$\frac{1}{\sqrt{2}}(|+\rangle - |-\rangle) = -|1\rangle$$

Here, we say that the amplitudes of the $|0\rangle$ state interfere *destructively* and the $|1\rangle$ interfere *constructively* although in this example, we end up with an extra minus sign which is again interpreted as a rotation of π in the complex plan. This phase is referred to as a global phase.
When we apply the measurement rule to the -$|1\rangle$ statevector, we find :

$$|\langle x|(-|1\rangle)|^2 = |-\langle x|1\rangle|^2 = |\langle x|1\rangle|^2$$

We can see that the minus disappears once we take the magnitude of the complex number. This effect is completely independent of the measured state $|x\rangle$. No matter what measurement we are considering, the probabilities for the state -$|1\rangle$ and $|1\rangle$ are identical. Since measurements are the only way to extract information from a quantum system, these two states are equivalent in all ways that are physically relevant.
More generally, we refer to any overall factor γ on a state for which $|\gamma| = 1$ as a *global phase*. States that differ only by a global phase are physically indistinguishable.

$$|\langle x|(\gamma|1\rangle)|^2 = |\gamma\langle x|1\rangle|^2 = |\langle x|1\rangle|^2$$

9.2.2.4 The observer effect

As we saw in the statevector representation, the amplitudes of the different possible states that form the statevector contain information about the probabilities of us finding the Qubit in a specific state. However, once we measure a Qubit, we know with certainty what the state of the Qubit is. For example, if we measure a Qubit in the state :

$$|q\rangle = \alpha|0\rangle + \beta|1\rangle$$

And find it in the state $|0\rangle$, if we process another measurement, we will be a 100% sure of finding the Qubit in the state $|0\rangle$. This means the act of measuring *changes* or *orientates* the state of our Qubits.

$$|q\rangle = \begin{bmatrix}\alpha\\ \beta\end{bmatrix} \xrightarrow{Measure|0\rangle} |q\rangle = |0\rangle = \begin{bmatrix}1\\ 0\end{bmatrix}$$

We refer to this phenomenon as *collapsing* the state of the Qubits, and this is where the complexity of measuring the state of quantum systems lays. Therefore, it is important to determine when to allow measurement operations. For example, if we were to constantly keep track of our Qubits' states, they would always simply be in either $|0\rangle$ or $|1\rangle$. Thus, becoming no different from classical bits, and our computation would be easily replaced by a classical one, since quantum computations rely upon the Qubits exploring more complex states than just $|0\rangle$ or $|1\rangle$. Which is why we perform all the measurements at the end of a certain computation.

9.3 The Bloch Sphere

The Bloch Sphere is used to represent the state of a single Qubit. Any state vector begins at the origin and terminates on the surface of the unit Bloch Sphere. We can move the end of a state vector along the surface of the Bloch Sphere by applying unitary operators (which we will introduce in the next section) to the state vector. The idea behind this representation, as we will see, is to get rid of the complex nature of the states' amplitudes, and only deal with real numbers by representing the relative phase between the amplitudes within a statevector with a rotation in the Bloch sphere, and neglecting the global phase since it has no effect on the output.

9.3.1 Qubit states in the Bloch Sphere

As we saw in the previous section, a general state description of the Qubit is given by :

$$|q\rangle = \hat{\alpha}\,|0\rangle + \hat{\beta}\,|1\rangle$$
$$\hat{\alpha}, \hat{\beta} \in \mathbb{C}$$

Instead of describing the state of a Qubit with complex numbers, we can do that using only real numbers, by applying some modifications to the Qubit's state description above. For that, let's right the two complex numbers above in the exponential form :

$$\hat{\alpha} = \alpha e^{i\theta_1}$$
$$\hat{\beta} = \beta e^{i\theta_2}$$

And replace them in the state description above :

$$|q\rangle = \alpha e^{i\theta_1}\,|0\rangle + \beta e^{i\theta_2}\,|1\rangle$$
$$|q\rangle = e^{i\theta_1}(\alpha\,|0\rangle + \beta e^{i(\theta_2-\theta_1)}\,|1\rangle)$$

θ_1 being considered as global phase, it has no effect on the state, so we can get rid of it. Also, $(\theta_2 - \theta_1)$ is the relative phase between the amplitudes of the state, we will label it as φ. So, our state description will be of the form :

$$|q\rangle = \alpha\,|0\rangle + \beta e^{i\varphi}\,|1\rangle$$

With α, β, φ all real numbers. Finally, since the Qubit state must be normalised, we can use the trigonometric identity :

$$\sqrt{\alpha^2 + \beta^2} = \sqrt{cos^2\frac{\theta}{2} + sin^2\frac{\theta}{2}} = 1$$

Therefore, with :

$$\alpha = cos\frac{\theta}{2}$$

$$\beta = sin\frac{\theta}{2}$$

We can right :

$$|q\rangle = cos\frac{\theta}{2}|0\rangle + sin\frac{\theta}{2}e^{i\varphi}|1\rangle$$

With $\theta, \varphi \in \mathbb{R}$.
This is the form of the state description that we will use in order to represent our states in the Bloch Sphere.

9.3.2 Visually representing a Qubit state

Our goal is to plot the state representation of the Qubit q, in a 3D coordinate system (O,X,Y,Z) where the position of the end of the vector is determined by φ and θ. Moreover, since the magnitude of the Qubit state is 1, we can plot any single Qubit state on the surface of a sphere, known as the Bloch Sphere.

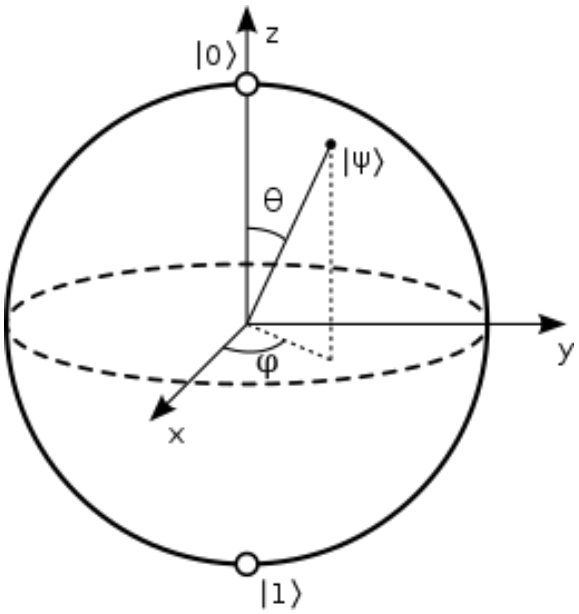


Figure 16: The Bloch Sphere

Here, φ is a rotation over the Z-axis, which represents the relative phase of the Qubit state, and θ is a rotation over the Y-axis when $\varphi = 0$, which determines the amplitudes of the statevector. For example, if we wanted to plot a Qubit in the state $|+\rangle$ we would replace θ by $\frac{\pi}{2}$ and φ by 0.

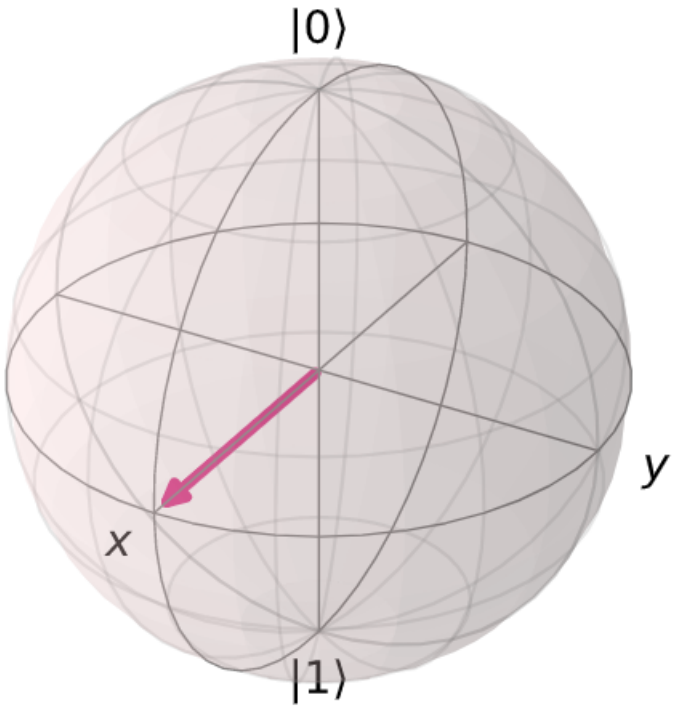


Figure 17: $|+\rangle$ state representation in the Bloch Sphere

10 Quantum operators

In the previous sections, we looked at all possible states that a Qubit could be in, and we introduced some representations of Qubits' states. We saw that all states that a Qubit can be in are limited to the form :

$$|q\rangle = cos\frac{\theta}{2}|0\rangle + sin\frac{\theta}{2}e^{i\varphi}|1\rangle$$

Where θ and φ are real numbers, and the end of the statevector can be anywhere in the surface of the Bloch Sphere as θ and φ vary. However, in order to be able to change the position of the Qubit's statevector anywhere in the Bloch sphere, we need to introduce some tools that will allow us to do so. We refer to these operators as quantum *gates*.

Unlike classical gates, which are only described by symbols and truth tables, quantum gates are concrete matrices, which will be applied to the statevector in order to come up with a new statevector representing the new state of the Qubit. There exist *unitary* gates, or *single* gates, *binary* gates and *ternary* gates, which can also be referred to as *multiple* Qubit gates.

10.1 Single Qubit gates

Single Qubit gates or unitary gates, are quantum operators designed to be applied to single Qubit statevectors, they are represented by 2D matrices.

10.1.1 The Pauli gates

We will first introduce the Pauli gates. These are three of the most famous operators as they have some very specific features. Along with the identity matrix, they form what so-called the *Pauli Group*.

10.1.1.1 The X-gate

First, we have the X-gate, given by the Pauli-X matrix :

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = |0\rangle\langle 1| + |1\rangle\langle 0|$$

This gate turns the state $|0\rangle$ into the state $|1\rangle$ and vice-versa. More generally, it swaps the amplitudes of the state $|1\rangle$ and $|0\rangle$. To see the effect of this gate, let's apply this gate to the $|0\rangle$ statevector :

$$X|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$$

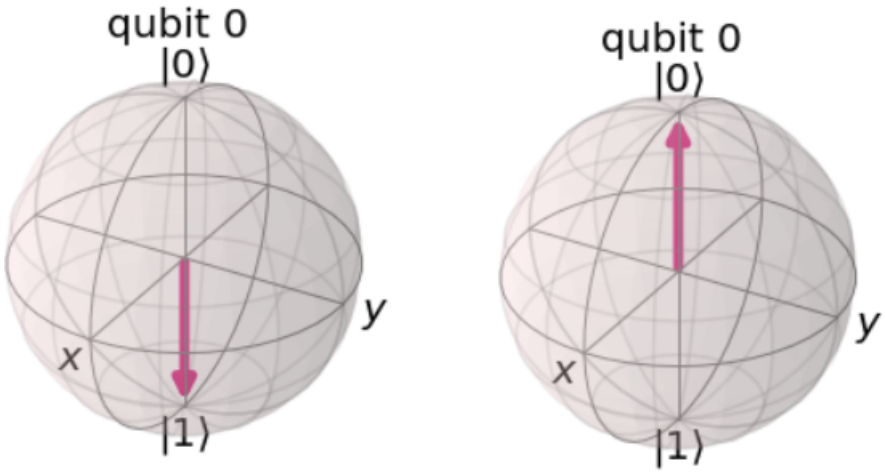


Figure 18: Representation of the $|0\rangle$ state and the $|1\rangle$ state in the Bloch Sphere

Notice that the X-gate provokes a rotation by π around the x-axis in the Bloch Sphere. It is also known as the NOT-gate because of the similarities it presents with the classical NOT-gate.
The representation of the X-gate in the quantum circuit diagram has the following form :

Figure 19: X-gate representation

10.1.1.2 The Y-gate

We also have the Y-gate, given by the Pauli-Y matrix :

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = -i\,|0\rangle\langle 1| + i\,|1\rangle\langle 0|$$

Similarly to the X-gate, the Y-gate performs a rotation of π radians around the y-axis in the Bloch Sphere. For example, when we apply the Y-gate to the $|0\rangle$ state, we expect to have the $|1\rangle$ as a result as a rotation over the y-axis in the Bloch Sphere would suggest.

$$Y\,|0\rangle = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ i \end{bmatrix} = i\,|1\rangle$$

As we saw earlier, the overall factor i is considered as a global phase which has no physical effect on the Qubit state. Therefore, we can get rid of it and end up with the $|1\rangle$ state. The representation of the Y-gate in the quantum circuit diagram has the following form :

Figure 20: Y-gate representation

10.1.1.3 The Z-gate

Finally, we have the Z-gate, given by the Pauli-Z matrix :

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = |0\rangle\langle 0| + |1\rangle\langle 1|$$

Unsurprisingly, like the X and Y-gates, the Z-gate performs a rotation of the statevector by π over the z-axis in the Bloch Sphere. Interestingly, the Z-gates has no effect on the $|0\rangle$ and the $|1\rangle$ states. We will give an interpretation of this result later in this section, but for now, let's just focus on the fact that the states $|0\rangle$ and $|1\rangle$ are left as they are when going through a Z-gate :

$$Z\,|0\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

The representation of the Z-gate in the quantum circuit diagram has the following form :

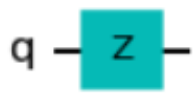


Figure 21: Z-gate representation

10.1.1.4 Some important notes

We saw earlier that the Z-gate appeared to have no effect on the Qubit when it is the either of the two state $|0\rangle$ or $|1\rangle$. We call these statevectors the *eigenstates* of the Z-gate. In fact, the computational basis formed with theses two states is often referred to as the Z-basis. Similarly, we could also notice that the two states $|+\rangle$ and $|-\rangle$ are the *eigenstates* of the X-gates. Thus, forming the so-called X-basis. Additionally, the *eigenstates* of the Y-gate are symbolized by $|\circlearrowleft\rangle$ and $|\circlearrowright\rangle$. The last two state are given by :

$$|\circlearrowleft\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle + i\,|1\rangle)$$
$$|\circlearrowright\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle - i\,|1\rangle)$$

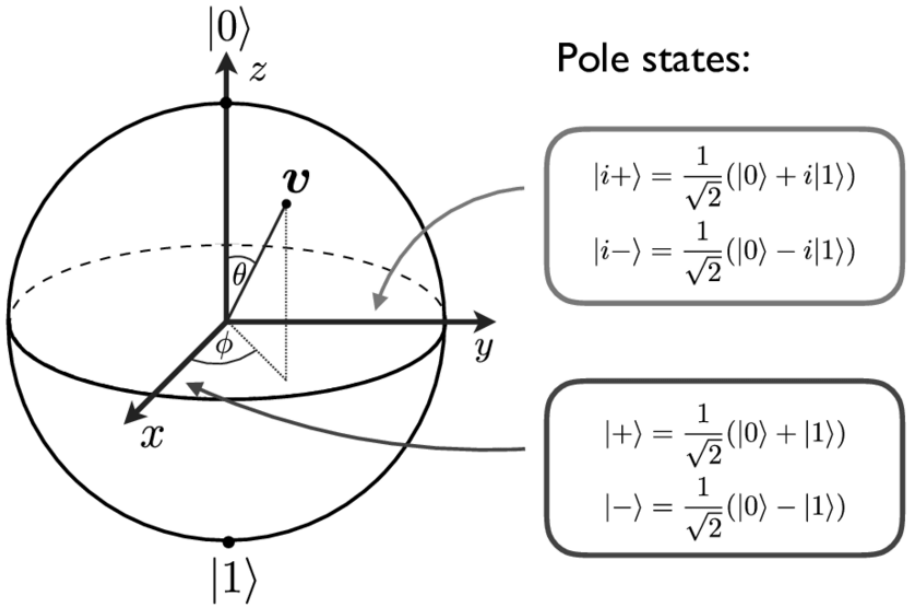


Figure 22: Representation of the eigenstates of the three Pauli gates

10.1.2 The Hadamard gate

The Hadamard gate of the H-gate is a crucial quantum gate as it allows to move from a definite Qubit state ($|0\rangle$ or $|1\rangle$) to a superposition of the two. It is defined by the matrix :

$$H = \tfrac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Let's see this gate in action :

$$H\,|0\rangle = \tfrac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \tfrac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} = |+\rangle$$

$$H\,|1\rangle = \tfrac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \tfrac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix} = |-\rangle$$

We see that applying an H-gate to the two states positioned in the poles of the Bloch Sphere gave us two statevectors that are in a superposition of the two first states. The representation of the Hadamard-gate in the quantum circuit diagram has the following form :

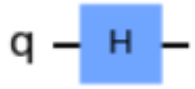


Figure 23: H-gate representation

10.1.3 The S, T and P-gate

10.1.3.1 The S-gate

The S-gate makes a quarter turn to the statevector around the z-axis. In other words, it performs a rotation of $\frac{\pi}{2}$. Therefore, we also refer to this gate as the $\sqrt{Z}$-gate. It is described by the matrix bellow :

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

Note that unlike the gates introduced previously, the S-gate is not its own inverse. Hence, the inverse of the S-gate is also a separate gate noted $S^\dagger$:

$$S = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{2}} \end{bmatrix}, S^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-i\frac{\pi}{2}} \end{bmatrix}$$

The representation of the S-gate in the quantum circuit diagram has the following form :

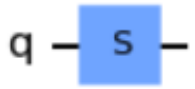


Figure 24: S-gate representation

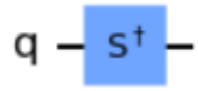


Figure 25: $S^\dagger$-gate representation

10.1.3.2 The T-gate

The T-gate in defined by a rotation of $\frac{\pi}{4}$ around the z-axis of the Bloch Sphere. It is also known as the $\sqrt[4]{Z}$-gate. The T-gate and its inverse are given by :

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{bmatrix}, T^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-i\frac{\pi}{4}} \end{bmatrix}$$

The representation of the T-gate in the quantum circuit diagram has the following form :

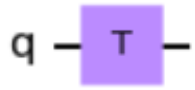


Figure 26: T-gate representation

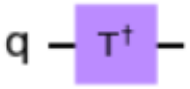


Figure 27: $T^\dagger$-gate representation

10.1.3.3 The P-gate

The P-gate, or the Phase-gate is a more general operator as it performs a rotation of ψ to the state over the z-axis of the Bloch Sphere, where ψ could take any real value. Thus, it is obvious to notice that the Z,S and T gates are special cases of the P-gate.

$$P(\psi) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\psi} \end{bmatrix}, P(\psi)^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-i\psi} \end{bmatrix}$$

10.1.3.4 The U-gate

We saw earlier that the Z,T and S gates are special cases of the P-gate. Moreover, the U-gate is the most general of all single quantum gates. As we can represent any of the previous gates with a U-gates given the right parameters. the U-gate has the following form :

$$U(\theta, \psi, \lambda) = \begin{bmatrix} cos\frac{\theta}{2} & -e^{i\lambda}sin\frac{\theta}{2} \\ e^{i\psi}sin\frac{\theta}{2} & e^{i(\psi+\lambda)}cos\frac{\theta}{2} \end{bmatrix}$$

For example, we could construct a Hadamard-gate as follows :

$$U(\tfrac{\pi}{2}, 0, \pi) = \tfrac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = H$$

10.2 Multiple Qubit gates and entanglement

In this section, we will discuss *binary* and *ternay* gates which give us the ability to handle the state of multiple Qubits and essentially allow us to leverage the core features of quantum computing. Yet, to be able to use this kind of gates, we will need to find a representation that will contain the states of multiple Qubits at the same time.

10.2.1 Representing Multi-Qubit states

In two-Qubit systems, we consider the following computational basis :

$$|00\rangle = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}, \qquad |01\rangle = \begin{bmatrix}0\\1\\0\\0\end{bmatrix}, \qquad |10\rangle = \begin{bmatrix}0\\0\\1\\0\end{bmatrix}, \qquad |11\rangle = \begin{bmatrix}0\\0\\0\\1\end{bmatrix}$$

Therefore, we represent two-Qubit systems' states such as :

$$|a\rangle = a_{00}\,|00\rangle + a_{01}\,|01\rangle + a_{10}\,|10\rangle + a_{11}\,|11\rangle = \begin{bmatrix}a_{00}\\a_{01}\\a_{10}\\a_{11}\end{bmatrix}$$

Where the rules of measurement remain the same :

$$p(|00\rangle) = |\left\langle 00\,|a\right\rangle|^2 = |a_{00}|^2$$

And the implications of the rule hold, such as the normalisation of the statevector :

$$|a_{00}|^2 + |a_{01}|^2 + |a_{10}|^2 + |a_{11}|^2 = 1$$

The collective state of two or more Qubits is deducted from the tensor product of the single statevectors :

$$|a\rangle = \begin{bmatrix}a_0\\a_1\end{bmatrix}, \qquad |b\rangle = \begin{bmatrix}b_0\\b_1\end{bmatrix}$$

$$|ab\rangle = |a\rangle \otimes |b\rangle = \begin{bmatrix}a_0 \times \begin{bmatrix}b_0\\b_1\end{bmatrix}\\ \\ a_1 \times \begin{bmatrix}b_0\\b_1\end{bmatrix}\end{bmatrix} = \begin{bmatrix}a_0b_0\\a_0b_1\\a_1b_0\\a_1b_1\end{bmatrix}$$

Also, if we wanted to apply a single Qubit gate to a multi-state vector, we need to need to rely on the tensor product. For example, if we wanted to apply an H-gate to the first Qubit of a two-level quantum system, and a NOT gate to the second one, we would just have to construct a new gate out of the tensor product of the two single gates in question.

$$X \otimes H = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} \otimes H = \begin{bmatrix}0 & 1 \times H\\1 \times H & 0\end{bmatrix}$$

$$X \otimes H = \begin{bmatrix}0 & H\\H & 0\end{bmatrix}$$

10.2.2 Binary-Qubit gates

10.2.2.1 The CNOT-gate

We have previously come across the NOT-gate also known as the X-gate where we saw that this operator swaps the amplitudes of the two definite states $|0\rangle$ and $|1\rangle$. The CNOT-gate is a binary operator, where it applies a NOT operation to the second Qubit if and only if the first Qubit (referred to as the control Qubit) is at state $|1\rangle$. The gate is drawn as follows in the quantum circuit diagram, with the q_0 as the control and the q_1 as the target:

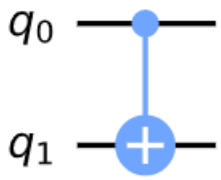


Figure 28: Representation of the CNOT-gate

And the CNOT-matrix is given by :

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

As a result, we end up with the following truth table :

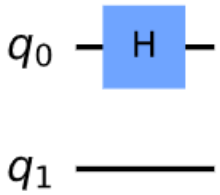
Input (t,c)	Output (t,c)
00	00
01	11
10	10
11	01

Figure 29: CNOT truth table

10.2.2.2 Entangled states

So far, we have seen how the CNOT-gate affects our system for definite states. Now, let's see

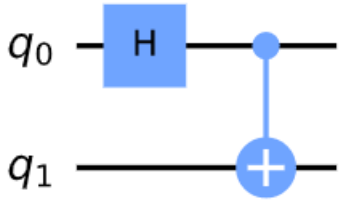
how this performs on Qubits that are in a superposition of states. To do so, we will apply a Hadamard-gate to have the $|+\rangle$ statevector.



This will produce the state :

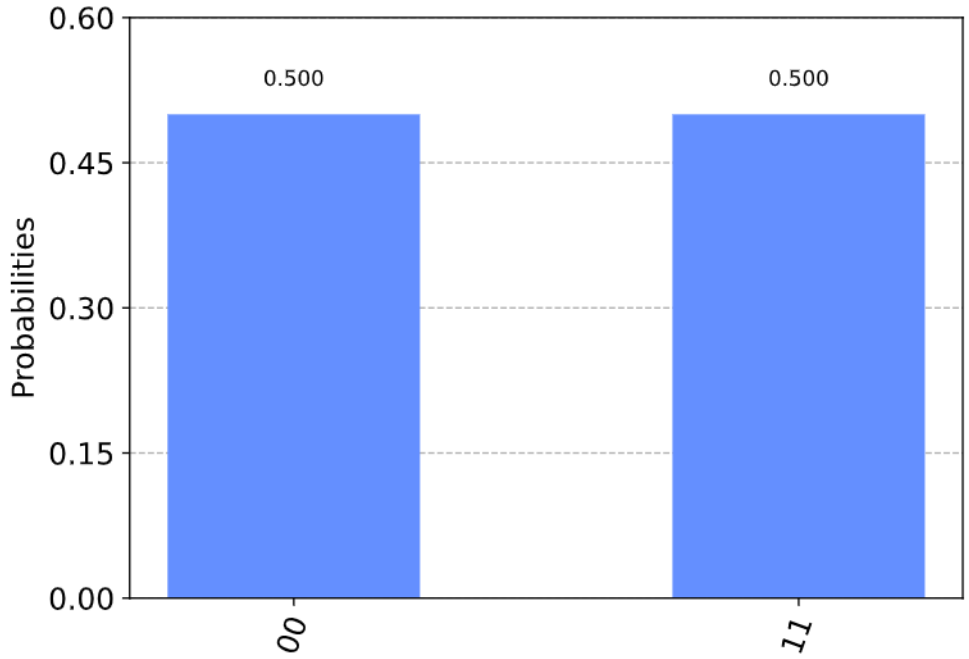
$$|0\rangle \otimes |+\rangle = |0+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |01\rangle)$$

Now, let's see what happens when we apply a CNOT-gate like we did earlier to the classical states:



$$\text{CNOT}|0+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

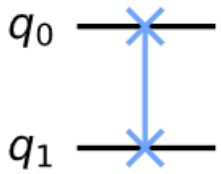
This is known as the *bell* state. We can see that the probabilities are equally split between the two states $|00\rangle$ and $|11\rangle$. Most importantly, this state has 0% chance of being measured either of the states $|01\rangle$ and $|10\rangle$.



This is a very interesting result. As the state of the first Qubit is strongly correlated with the state of the second one. Therefore, if we measure the first Qubit and find it in the state $|1\rangle$, then we will be certain that the collective state of the two-level quantum system is $|11\rangle$. Even if the two Qubits are light years away from each other. We describe these two Qubits as being *Entangled* (we introduced this notion in an earlier chapter).

10.2.2.3 The SWAP-gate

Sometimes we need to move information around in a quantum computer. Instead of having to physically moving them, a better solution would by to apply a SWAP-gate. This will move the state between two Qubits.
The representation of SWAP-gate in a circuit is given by :



And the matrix form of this gate is as follows :

$$\text{SWAP}= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

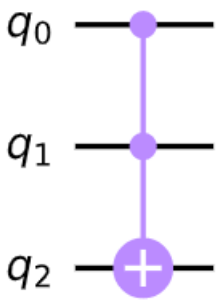
As we can see, the SWAP-gate turn the $|01\rangle$ state into a $|10\rangle$ state and vice versa. Thus, moving states between the first and the second Qubit.

10.2.3 Ternary-Qubit gates

10.2.3.1 The Toffoli gate

The Toffoli gate, also referred to as the CCNOT-gate, performs the same effect as the CNOT gate except that it requires two control Qubits instead of one. It performs an NOT gate is both the two control Qubits have a state of $|1\rangle$. The target Qubit is then equal to either an AND or a NAND of the two control Qubits.

The Toffoli-gate is drawn in the quantum circuit as follows :



And its matrix is of the form :

$$\text{Toffoli}= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

10.2.3.2 The CSWAP-gate

The CSWAP, also known as the Fredkin-gate, performs a swap on the target Qubits (the last two Qubits) when the control Qubit (the first one) is in state $|1\rangle$. The matrix representing this operation is :

$$\text{CSWAP}= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

For example, the Fredkin gate applied to $|110\rangle$ gives :

$$\text{CSWAP}|110\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = |101\rangle$$

10.3 Universality of quantum operators

In classical computing, we can construct any possible gate exclusively with NAND gates. Therefore, we refer to the NAND gate as the universal gate for classical computing. The question is, is there any gate or a set of gates that are universal for quantum computing? In fact, there is no gate that can lead to universality on its own. However, we can find some sets of unitary, binary and ternary operators that yield to universality. Here are two examples :

- The Toffoli gate paired with a basis-changing unary gate (such as H)
- Another set of gates that leads to universality is composed of {CNOT,T,H}

Part IV
Complexity theory

Quantum computing is a thoroughly new approach to computation, it helped so many researches discover new methods of solving problems. However, this new field may seem new, investigation is always taking place to develop more optimized solutions to help the technological world progress even more. As researches go, problems that were considered intractable in the past became solvable now due to the unusual behaviour of quantum particles which make the basis units of a quantum computer.

In spite of the effectiveness of a quantum system to run algorithms, it is however paramount to take into consideration a well made process for determining the best way to solve a given problem class. That's what we call the *Complexity theory*, which is the study of problem classes we may encounter. Then we lay our interest on discovering the computational resources needed for solving a specific problem encompassed by the *Complexity analysis* study.

11 Time Complexity

In computer science, the time complexity of an algorithm quantifies the amount of time taken by this algorithm to run as a function of the length of the string representing the input.

The time complexity of an algorithm is expressed using the notation O which excludes the terms with a lower degree taking into consideration only the term with the highest one.As an example, if the time taken by an algorithm to run is n^5+6n+2 the time complexity will be $O(n^5)$.

11.1 Constant time $O(1)$

When a particular algorithm takes the same amount of time to run in every execution we quantify its time complexity using constant complexity time $O(1)$, as an example to the algorithms with this time complexity we take.

1. array: accessing any element
2. fixed-size stack: push and pop methods
3. fixed-size queue: enqueue and dequeue methods

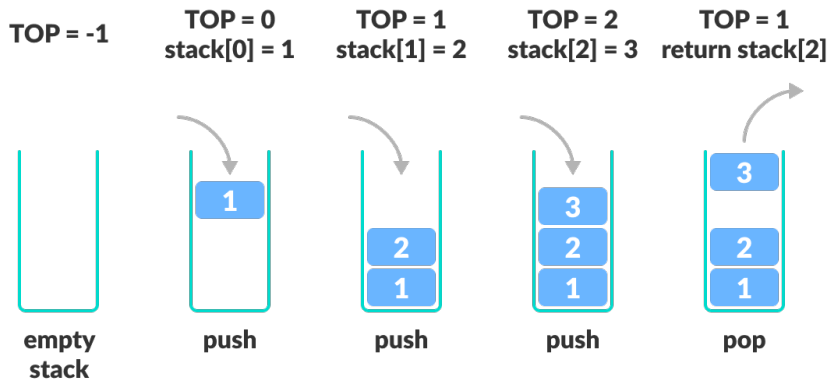


Figure 30: Constant time complexity example "Push and pop method"

11.2 Linear time $O(n)$

An algorithm is said to have a linear time complexity if the time taken to run is directly proportional to the size of the input. Which means that the time rises linearly when the input size increases.

As an example to these algorithms, we take the following ones:

1. Array: Linear Search, Traversing, Find minimum etc
2. ArrayList: contains method
3. Queue: contains method

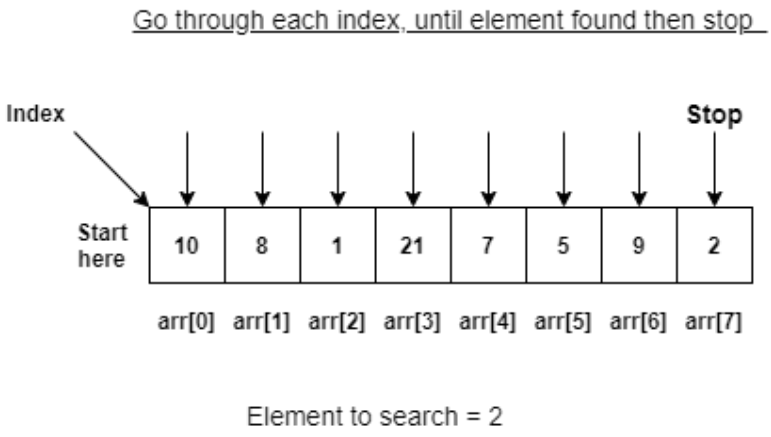


Figure 31: linear time complexity example "Linear search"

11.3 Logarithmic time $O(log(n))$

An algorithm is said to be having a logarithmic time complexity in case of having a growing time proportional to the logarithm of the input size. The binary search makes a perfect example to that type time complexity.

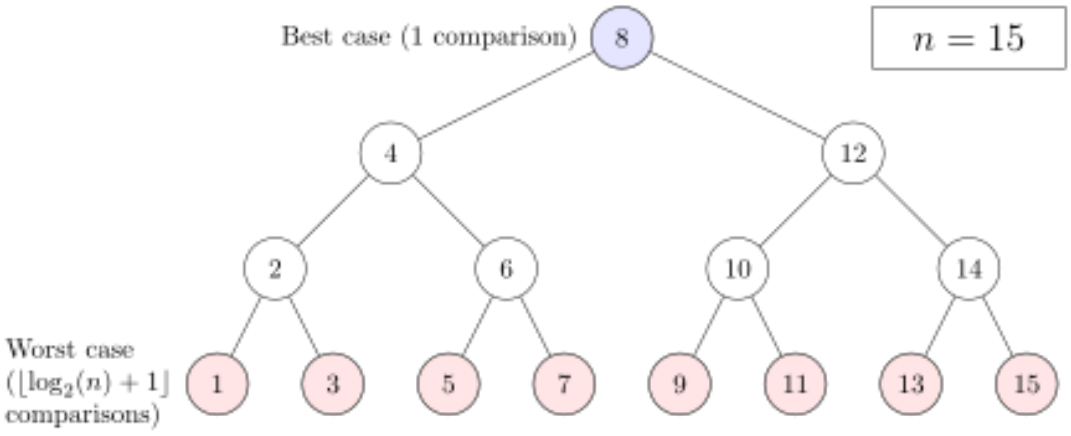


Figure 32: logarithmic time complexity example "Binary search"

11.4 Quadratic time $O(n^2)$

We take the next sorting algorithms as an example:

1. Bubble Sort
2. Selection Sort
3. Insertion Sort

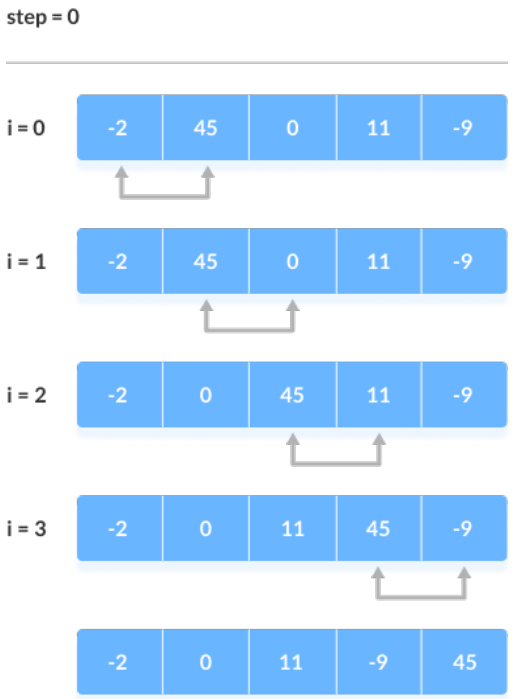


Figure 33: quadratic time complexity example "Bubble sort"

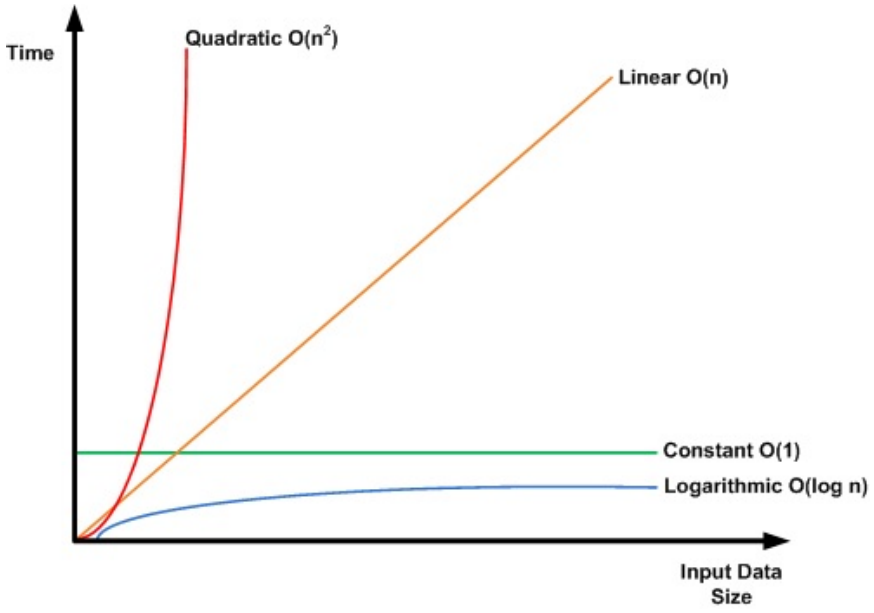


Figure 34: the evolution of time over input data size

12 Complexity classes

Sets of related computational problems , defined in terms of computational difficulty of solving problems with respecting some resources linked to time and memory constraints.

In the coming subsections we will be presenting a number of common complexity classes from classical computing.

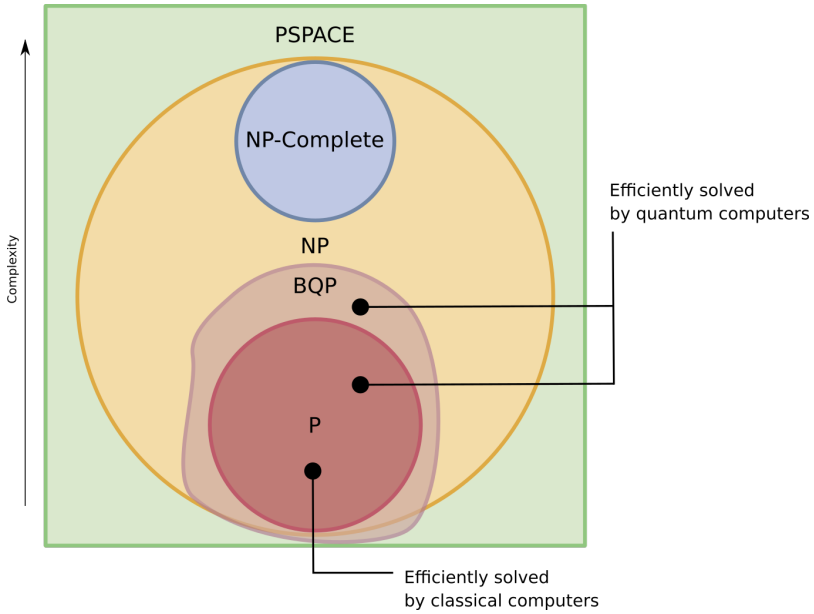


Figure 35: A schematic representation of complexity classes

12.1 P class

Shortened for polynomial class, encompasses all problems that can be solved using a Turing machine within polynomial time including problems of $O(1)$, $O(n)$,$O(log(n))$ and $O(n^2)$.

This class is included within all of the remaining classes since all of its problems are solvable and the take a reasonable amount of time to be resolved on a classical computer. plentiful of algorithms are included in that class counting the previously mentioned ones in the preceding examples.

12.2 NP

Shortened for non-deterministic polynomial time. This class envelops all problems that can be solved in a polynomial time using a non-deterministic Turing machine. The difference between the NP and P class is that the solution of the NP problem is obtained by firstly making a guess of the problem's solution by the non-deterministic Turing machine in a non-deterministic way. Then, this guess will be examined in a logical way to check if either it really is a solution or not. If not, the non-deterministic machine would make another guess. This way of solving problems may seem random but it sometimes aid in remarkably decreasing the amount of time taken to solve them.

12.3 NP-complete

A problem is said to be NP-complete if, first, this problem is in NP class and ,second, all NP problems are polynomial-time reducible to that problem.

12.4 PSPACE

This class focuses on storage resources rather than time. It envelops all problems that are solvable by an algorithm whose total data size storage is upper bounded by a polynomial in the instance size. This class contains all of the other classes for the reason that if an algorithm is bounded by a polynomial size. It means that it will take a finite amount of time to solve the problem.

12.5 BQP

Shortened for error-Bounded Quantum Polynomial time, a decision problem belongs to BQP class if it runs in a polynomial time and yields the correct results with a high probability. We focus on this class in quantum computing. Furthermore, we observe in figure 28 that certain problems which belong to this class are solvable only using quantum computers where some specific characteristics are required. As an example, we give the entanglement phenomenon that helps us reducing the required amount of storage size considerably while performing the same task on a classical computer would take a size exponentially larger that the one taken by the quantum computer which sometimes can be heavy to perform and sometimes is impossible.

12.6 EQP

Every solvable problem by an algorithm whose running time is bounded by a polynomial complexity time the same as the BQP class with only one difference which is the probability for resulting a correct output must be 100% for problems belongs to that class.
However P class is shown as being included in the EQP class, it is proved that not all P problems are tractable using EQP algorithms but only those with a particular configuration that can be utilized to create an algorithm to solve it, such as Shor's algorithm that leverages the periodicity of the function which subsequently enables us to solve an equivalent problem for factoring large numbers.

Part V
Optimization algorithms

13 Solving Linear Systems of Equations using HHL algorithm

13.1 Introduction to linear systems in quantum computing

A linear system is widely present in a variety of fields specifically engineering, it helps the user presenting a particular problem in a linear formulation to describe the relationship between unknown physical parameters. Furthermore, this formulation can be mathematically presented as follows

$$A.\vec{x} = \vec{b}$$

where $A \in \mathbf{C}^{N\times N}$ is a matrix defines the system, $\vec{x} \in \mathbf{C}^N$ is the unknown parameter vector and finally $\vec{b} \in \mathbf{C}^N$ is a known vector.

In automation and control systems, linear systems are heavily used such as calculating an optimized LQR regulator where both a precise and rapid optimization procedure is recommended to help us quickly accomplishing the most advantageous solution for a better performance of a physical system represented in a linear format, or in identification purposes using the least squares method where sometimes a large number of Input/Output data is being handled which will subsequently result a huge dimension of the A matrix. Hence, more reliable optimization approaches are needed, To achieve that we are going to take a look at a Quantum algorithm for solving a linear system with a time complexity is equal to $O\left(\frac{log(N)k^2s^2}{\epsilon}\right)$ Where k denotes the condition number of the system, ϵ the desired accuracy at the end of the execution, N is the input size and s describes the number of non-zeros elements in a column or a row of A matrix. Nonetheless, the condition that the linear system is s-sparse must be respected before applying the algorithm. On the other hand, the classical equivalent algorithm to HHL algorithm has a time complexity equal to $O(N^3)$, we can observe that by implementing the quantum one we can reach an exponential speed-up.

Moreover, another crucial remark is to be kept in mind is that the classical algorithm returns the full solution, On the other hand the quantum one returns the expectation value of the solution represented by $\langle\vec{x}|\, M\, |\vec{x}\rangle$ here M is a Hamiltonian matrix.

13.2 The HHL algorithm

13.2.1 Mathematical development

The first step towards solving a linear system using a quantum approach is to encode the problem to a quantum language, To do so we need to rescale the parameter vectors $\vec{x}$ and $\vec{b}$ to a quantum format denoted by $|x\rangle$ and $|b\rangle$ where the amplitude b_j represents the value j^{th} element of the $\vec{b}$ vector (the same applies to $\vec{x}$).

After reshaping the problem we will focus on quantum mechanically solving an equivalent problem defined by the following mathematical formulation

$$A\,|x\rangle = |b\rangle$$

Here the matrix A is a Hamiltonian which means that it can be decomposed to a sum of eigenvectors scaled by their respective eigenvalues as follows

$$A = \sum_{j=0}^{N-1} \lambda_j\,|u_j\rangle\,\langle u_j| \qquad\qquad \lambda_j \in \mathbf{R}$$

here λ_j is the eigenvalue corresponds to the u_j eigenvector.The inverted matrix A^{-1} can be described in the consecutive way

$$A^{-1} = \sum_{j=0}^{N-1} \lambda_j^{-1}\,|u_j\rangle\,\langle u_j|$$

Similarly, the righthand term can be written in the A eigenbasis

$$|b\rangle = \sum_{j=0}^{N-1} b_j\,|u_j\rangle \qquad\qquad b_j \in \mathbf{C}$$

Finally the goal is to exit the algorithm with the readout register in the state

$$|x\rangle = A^{-1}\,|b\rangle = \sum_{j=0}^{N-1} \lambda_j^{-1} b_j\,|u_j\rangle$$

13.2.2 Description of the HHL algorithm

Three registers are used and being set to $|0\rangle$ at the beginning of the algorithm. a register denoted n_l used for storing the binary representation of the matrix's A eigenvalues . the next one denoted n_b used for storing the vector solution , from now on $N = 2^{n_b}$.

The next schematic representation to expose the applied instructions in way that the algorithm result the desired output, more detailed explanation of the non-exact case is given in Section 13.2.4.

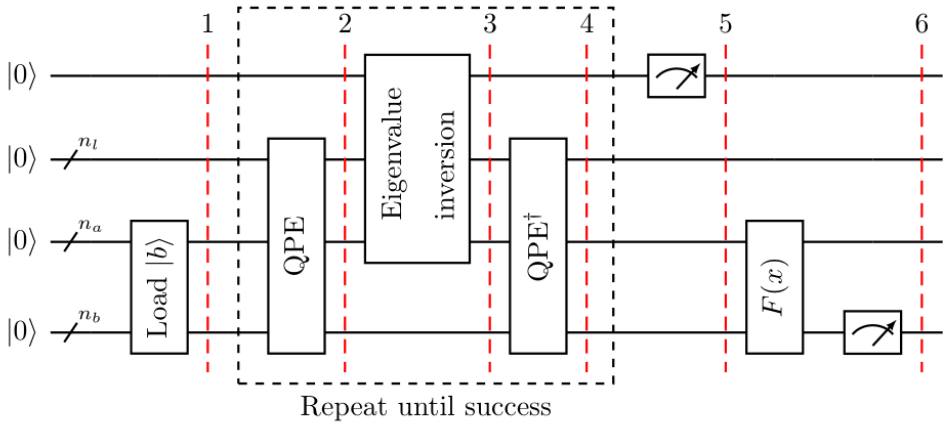


Figure 36: HHL algorithm circuit

- Load the data $|b\rangle \in \mathbb{C}^N$. That is, perform the transformation

$$|0\rangle_{n_b} \mapsto |b\rangle_{n_b}$$

- Apply Quantum Phase Estimation (QPE) with

$$U = e^{iAt} := \sum_{j=0}^{N-1} e^{i\lambda_j t}|u_j\rangle\langle u_j|$$

  The quantum state of the register expressed in the eigenbasis of A is now

$$\sum_{j=0}^{N-1} b_j|\lambda_j\rangle_{n_l}|u_j\rangle_{n_b}$$

  Where $|\lambda_j\rangle_{n_l}$ is the n_l bit representation of λ_j

- Add an auxiliary Qubit and apply a rotation conditioned on $|\lambda_j\rangle$

$$\sum_{j=0}^{N-1} b_j|\lambda_j\rangle_{n_l}|u_j\rangle_{n_b}\left(\sqrt{1-\frac{C^2}{\lambda_j^2}}|0\rangle + \frac{C}{\lambda_j}|1\rangle\right)$$

  where C is a normalisation constant, and, as expressed in the current form above, should be less than the smallest eigenvalue λ_{min} in magnitude, i.e., $|C| < \lambda_{min}$.

- Apply $QPE^\dagger$. Ignoring possible errors from QPE, this results in

$$\sum_{j=0}^{N-1} b_j|0\rangle_{n_l}|u_j\rangle_{n_b}\left(\sqrt{1-\frac{C^2}{\lambda_j^2}}|0\rangle + \frac{C}{\lambda_j}|1\rangle\right)$$

- Measure the auxiliary Qubit in the computational basis. If the outcome is 1 , the register is in the post-measurement state

$$\left(\sqrt{\frac{1}{\sum_{j=0}^{N-1}|b_j|^2/|\lambda_j|^2}}\right)\sum_{j=0}^{N-1}\frac{b_j}{\lambda_j}|0\rangle_{n_l}|u_j\rangle_{n_b}$$

  which up to a normalisation factor corresponds to the solution.

- Apply an observable M to calculate

$$F(x) := \langle x|M|x\rangle$$

13.2.3 Quantum Phase Estimation (QPE) within HHL

The meaning behind constructing such an algorithm is for approximating the eigenvalue of a given eigenvector, given a unitary $U \in \mathbb{C}^{2^m\times 2^m}$ with an eigen vector $|\psi\rangle_j \in \mathbb{C}^{2^m}$ with an eigen value $e^{2\pi\theta_j}$ the role of the QPE is to find an approximated value $\tilde{\theta}_j$ to the real value θ_j by giving $|0\rangle\,|\psi\rangle_j$ at the input along with the unitary gate U

$$\mathrm{QPE}(U, |0\rangle|\psi\rangle_j) = |\tilde{\theta}\rangle_j|\psi\rangle_j$$

In order to exploit that algorithm in the HHL algorithm we use a unitary operator described as $U = e^{iAt}$ after exiting the subroutine we will be left with the eigenvalues of the matrix A since

$$e^{iAt} = \sum_{j=0}^{N-1} e^{i\lambda_j t}|u_j\rangle\langle u_j|$$

and then

$$\mathrm{QPE}(e^{iAt}, \sum_{j=0}^{N-1} b_j|0\rangle_{n_l}|u_j\rangle_{n_b}) = \sum_{j=0}^{N-1} b_j|\lambda_j\rangle_{n_l}|u_j\rangle_{n_b}$$

13.2.4 Example 4-Qubit HHL

We take the next example to clarify more the advantages of HHL algorithm.

$$A = \begin{pmatrix} 1 & -1/3 \\ -1/3 & 1 \end{pmatrix} \quad \text{and} \quad |b\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

in that example we had used $n_b = 1$ to describe $|b\rangle$, afterwards $n_l = 2$ to store the binary representation of the eigenvalues. Finally, one auxilaty Qubit to read the success state of the algorithm.

To illustrate the functioning process of the algorithm we had chosen the value of t by calculating the eigenvalue of the matrix A, thus to get the exact binary representation of the rescaled eigenvalues in the n_l register.

However, keep in mind that for the HHL algorithm implementation one does not need previous knowledge of the eigenvalues. Having said that, a short calculation will give

$$\lambda_1 = 2/3 \quad \text{and} \quad \lambda_2 = 4/3$$

Recall from the previous section that the QPE will output an n_l -bit (2-bit in this case) binary approximation to $\frac{\lambda_j t}{2\pi}$. Therefore, if we set

$$t = 2\pi \cdot \frac{3}{8}$$

the QPE will give a 2-bit binary approximation to

$$\frac{\lambda_1 t}{2\pi} = 1/4 \quad \text{and} \quad \frac{\lambda_2 t}{2\pi} = 1/2$$

which is, respectively,

$$|01\rangle_{n_l} \quad \text{and} \quad |10\rangle_{n_l}$$

The eigenvectors are, respectively,

$$|u_1\rangle = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad \text{and} \quad |u_2\rangle = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

In fact, a general Hermitian matrix A of dimension N can have up to N different eigenvalues.Therefore, calculating them would take $\mathbb{O}(N)$ time and the quantum advantage would be lost.

We can then write $|b\rangle$ in the eigenbasis of A as

$$|b\rangle_{n_b} = \sum_{j=1}^{2} \frac{1}{\sqrt{2}} |u_j\rangle_{n_b}$$

Now we are ready to go through the different steps of the HHL algorithm.

1. State preparation in this example is trivial since $|b\rangle = |0\rangle$
2. Applying QPE will yield
$$\frac{1}{\sqrt{2}}|01\rangle|u_1\rangle + \frac{1}{\sqrt{2}}|10\rangle|u_2\rangle$$
3. Conditioned rotation with $C = \frac{1}{8}$ that is less than the smallest (rescaled) eigenvalue of $\frac{1}{4}$. Note, the constant C here needs to be chosen such that it is less than the smallest (rescaled) eigenvalue of $\frac{1}{4}$ but as large as possible so that when the auxiliary Qubit is measured, the probability of it being in the state $|1\rangle$ is large.
$$\frac{1}{\sqrt{2}}|01\rangle|u_1\rangle \left(\sqrt{1 - \frac{(1/8)^2}{(1/4)^2}}|0\rangle + \frac{1/8}{1/4}|1\rangle\right) + \frac{1}{\sqrt{2}}|10\rangle|u_2\rangle \left(\sqrt{1 - \frac{(1/8)^2}{(1/2)^2}}|0\rangle + \frac{1/8}{1/2}|1\rangle\right)$$
$$= \frac{1}{\sqrt{2}}|01\rangle|u_1\rangle \left(\sqrt{1 - \frac{1}{4}}|0\rangle + \frac{1}{2}|1\rangle\right) + \frac{1}{\sqrt{2}}|10\rangle|u_2\rangle \left(\sqrt{1 - \frac{1}{16}}|0\rangle + \frac{1}{4}|1\rangle\right)$$
4. After applying $QPE^\dagger$ the quantum computer is in the state
$$\frac{1}{\sqrt{2}}|00\rangle|u_1\rangle \left(\sqrt{1 - \frac{1}{4}}|0\rangle + \frac{1}{2}|1\rangle\right) + \frac{1}{\sqrt{2}}|00\rangle|u_2\rangle \left(\sqrt{1 - \frac{1}{16}}|0\rangle + \frac{1}{4}|1\rangle\right)$$

5. On outcome 1 when measuring the auxiliary Qubit, the state is

$$\frac{\frac{1}{\sqrt{2}}|00\rangle|u_1\rangle\frac{1}{2}|1\rangle + \frac{1}{\sqrt{2}}|00\rangle|u_2\rangle\frac{1}{4}|1\rangle}{\sqrt{5/32}}$$

A quick calculation shows that

$$\frac{\frac{1}{2\sqrt{2}}|u_1\rangle + \frac{1}{4\sqrt{2}}|u_2\rangle}{\sqrt{5/32}} = \frac{|x\rangle}{||x||}$$

6. Without using extra gates, we can compute the norm of $|x\rangle$: it is the probability of measuring 1 in the auxiliary Qubit from the previous step.

$$P(|1\rangle) = \left(\frac{1}{2\sqrt{2}}\right)^2 + \left(\frac{1}{4\sqrt{2}}\right)^2 = \frac{5}{32} = ||x||^2$$

13.3 Qiskit implementation

13.3.1 Running HHL on a simulator

We execute the following code on qiskit simulator

main.py ● | Python - Get Started | Extension: Python

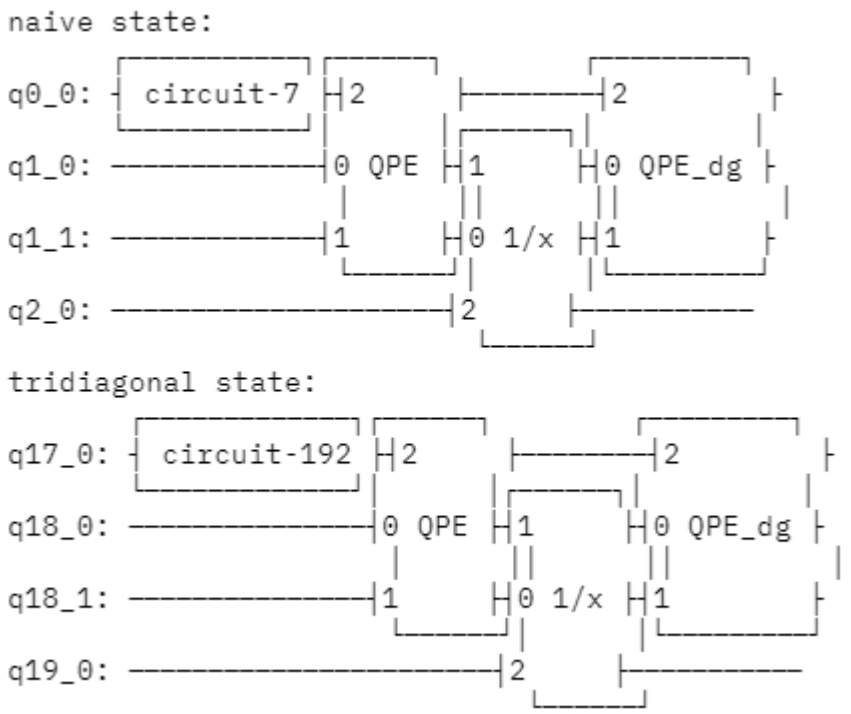
C: > Users > amine > OneDrive > Bureau > main.py > ...

```
1   import numpy as np
2   from qiskit.algorithms.linear_solvers.numpy_linear_solver import NumPyLinearSolver
3   from qiskit.algorithms.linear_solvers.hhl import HHL
4   matrix = np.array([[1, -1/3], [-1/3, 1]])
5   vector = np.array([1, 0])
6   naive_hhl_solution = HHL().solve(matrix, vector)
7   classical_solution = NumPyLinearSolver().solve(matrix, vector / np.linalg.norm(vector))
8   print('classical state:', classical_solution.state)
9   print('naive state:')
10  print(naive_hhl_solution.state)
11  print('tridiagonal state:')
12  print(tridi_solution.state)
13  print('classical Euclidean norm:', classical_solution.euclidean_norm)
14  print('naive Euclidean norm:', naive_hhl_solution.euclidean_norm)
15  print('tridiagonal Euclidean norm:', tridi_solution.euclidean_norm)
16
```

Figure 37: HHL algorithm simulation

We obtain the following results

classical state: [1.125 0.375]



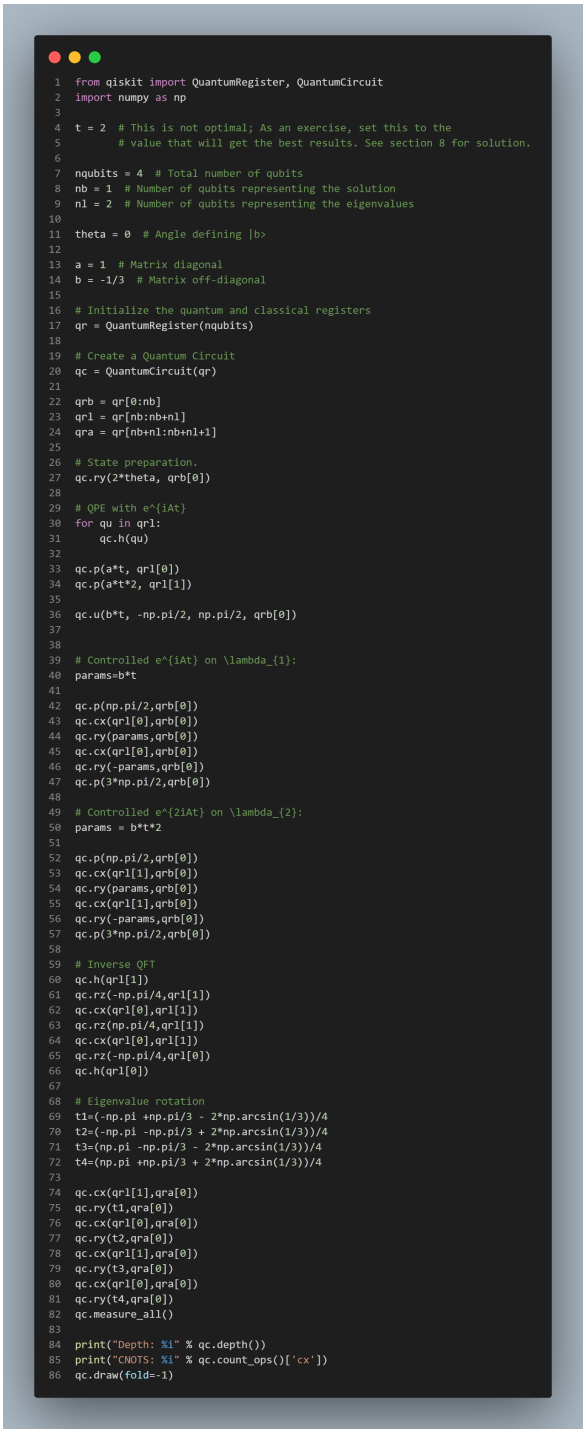
classical Euclidean norm: 1.1858541225631423

naive Euclidean norm: 1.1858541225631405

tridiagonal Euclidean norm: 1.185854122563142

We observe that the quantum simulation arrives at the exact same norm in an exponentially lower time.

13.3.2 Running the algorithm on a physical quantum computer



```python
from qiskit import QuantumRegister, QuantumCircuit
import numpy as np

t = 2  # This is not optimal; As an exercise, set this to the
       # value that will get the best results. See section 8 for solution.

nqubits = 4  # Total number of qubits
nb = 1  # Number of qubits representing the solution
nl = 2  # Number of qubits representing the eigenvalues

theta = 0  # Angle defining |b>

a = 1  # Matrix diagonal
b = -1/3  # Matrix off-diagonal

# Initialize the quantum and classical registers
qr = QuantumRegister(nqubits)

# Create a Quantum Circuit
qc = QuantumCircuit(qr)

qrb = qr[0:nb]
qrl = qr[nb:nb+nl]
qra = qr[nb+nl:nb+nl+1]

# State preparation.
qc.ry(2*theta, qrb[0])

# QPE with e^{iAt}
for qu in qrl:
    qc.h(qu)

qc.p(a*t, qrl[0])
qc.p(a*t*2, qrl[1])

qc.u(b*t, -np.pi/2, np.pi/2, qrb[0])


# Controlled e^{iAt} on \lambda_{1}:
params=b*t

qc.p(np.pi/2,qrb[0])
qc.cx(qrl[0],qrb[0])
qc.ry(params,qrb[0])
qc.cx(qrl[0],qrb[0])
qc.ry(-params,qrb[0])
qc.p(3*np.pi/2,qrb[0])

# Controlled e^{2iAt} on \lambda_{2}:
params = b*t*2

qc.p(np.pi/2,qrb[0])
qc.cx(qrl[1],qrb[0])
qc.ry(params,qrb[0])
qc.cx(qrl[1],qrb[0])
qc.ry(-params,qrb[0])
qc.p(3*np.pi/2,qrb[0])

# Inverse QFT
qc.h(qrl[1])
qc.rz(-np.pi/4,qrl[1])
qc.cx(qrl[0],qrl[1])
qc.rz(np.pi/4,qrl[1])
qc.cx(qrl[0],qrl[1])
qc.rz(-np.pi/4,qrl[0])
qc.h(qrl[0])

# Eigenvalue rotation
t1=(-np.pi +np.pi/3 - 2*np.arcsin(1/3))/4
t2=(-np.pi -np.pi/3 + 2*np.arcsin(1/3))/4
t3=(np.pi -np.pi/3 - 2*np.arcsin(1/3))/4
t4=(np.pi +np.pi/3 + 2*np.arcsin(1/3))/4

qc.cx(qrl[1],qra[0])
qc.ry(t1,qra[0])
qc.cx(qrl[0],qra[0])
qc.ry(t2,qra[0])
qc.cx(qrl[1],qra[0])
qc.ry(t3,qra[0])
qc.cx(qrl[0],qra[0])
qc.ry(t4,qra[0])
qc.measure_all()

print("Depth: %i" % qc.depth())
print("CNOTS: %i" % qc.count_ops()['cx'])
qc.draw(fold=-1)
```

Figure 38: Implementation on a physical computer

In order to implement the algorithm on a real quantum computer we have to use a language that can be described in a quantum assembly that a much lower language that can be understood by the quantum computer, we start by defining the three required registers for this example, afterwards we commence applying the quantum gates as follows.

this code allows us to find the eigenvalues thus we can use the same formula implemented in the simulation to find the solution.

Part VI
Conclusion

This work covered many aspects of quantum computing. First, we saw the core features of quantum mechanics that made the strength of quantum computing: Superposition, entanglement and reversibility are the main concepts that discriminate quantum computing from the classical one. As they allow a new way of thinking about computations. Furthermore, we introduced so to say the most important features that allow quantum computing: Qubits, quantum operators and the process of measurement with all its implications. Then, we treated some concepts of complexity theory, which allowed us to evaluate the quantum algorithms' performance over the classical ones. Finally, we used the concepts introduced previously to demonstrate one of the commonly used of the optimisation algorithms called *HHL algorithm*.

Even though quantum computing might seem extremely advantageous over the classical one, it is, however, subject to several physical limitations that prevented us from fully leveraging the power of quantum computing. Therefore, while waiting for consequent technological advances, the theoretical progress will always remain theoretical.

References

[1] Jack D. Hidary *Quantum Computing: An Applied Approach*. Springer Nature Switzerland AG 2019

[2] Eleanor Rieffel, Wolfgang Polak *Quantum Computing, a Gentle Introduction*. Massachusetts Institute of Technology, 2011.

[3] Qiskit: *Learn Quantum Computation using Qiskit*, 01/07/2021.
https://qiskit.org/

[4] S. Aaronson *The Limits of Quantum Computers*. Scientific American, March 2008.

[5] S. Aaronson *Quantum Complexity Theory*. Fall 2010. Massachusetts Institute of Technology: MIT.

[6] Yangyang Li, Senior Member, IEEE, Mengzhuo Tian, Guangyuan Liu, Cheng Peng and Licheng Jiao *Quantum Optimization and Quantum Learning: A Survey*. IEEE,2019.

[7] Forest: *Think quantum*, 06/07/2021.
https://www.rigetti.com/

[8] Wikipedia: *NP_(complexity)*, 6/6/2021.
https://en.wikipedia.org/wiki/NP_(complexity)

[9] Wikipedia: *Quantum optimization algorithms*, 6/6/2021.
https://en.wikipedia.org/wiki/Quantum_optimization_algorithms

[10] Bruker Corporation: *What's Nuclear Magnetic Resonance (NMR)? How Does It Work? What's It Used For? A Brief Introduction.*, 4/6/2021.
https://www.youtube.com/watch?v=Sn3dNMv-67k

[11] T. Bækkegaard, L. B. Kristensen, N. J. S. Loft, C. K. Andersen, D. Petrosyan : *Realization of efficient quantum gates with a superconducting Qubit-qutrit circuit*, 4/6/2021.
https://www.nature.com/articles/s41598-019-49657-1