People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University Of Larbi Tebessi Tebessa
Faculty Of Exact Sciences And Science Nature and life
Department Of Mathematics And computer science

MASTER'S THESIS
Quantum Computing

Prepared by :
Chihani Aymen

Supervised by :
Mr. Mekhaznia.T

Presented on : June 27, 2020

Before the jury:

Pr.MR. Bendjena. H
Mr. Tag. S

2019-2020

Abstract

Over time and increasing of power Computers are getting smaller and smaller. There is more crunching capability in today phones compared with room sized computer developed 5 decades ago. although of these continuous improvement, many complex problems still unsolved even with the most powerful computers. One critical problem is computer's memory and switching unit (transistor) that have achieved a level where they will be just an atom. This rise the necessity of powerful possibilities of quantum computing that have millions times faster processors than the used ones in classical computers. quantum computing still more complexed than classical computing and based on quantum physics where classical physics laws don't apply. This work gives a brief overview about quantum basics, detailed analysis and comparison between classical computers and quantum computers, benefits, disadvantages and future prospects.

Dedication.

To my mother and father to my

family and my friends

To my teachers

To my colleagues and colleagues

To candles that burn to light up for others to

everyone who taught me characters

I dedicate this humble research to the Lord

Almighty to find acceptance and success

Thanks.

Thanks, and gratitude We must take our last steps in university life from a stand back to years

We spent it in the university with our distinguished professors who gave us a lot

... there is a great effort in building the future generation to re-emit the nation

And before we go forward, my name is the signs of thanksgiving, gratitude,

appreciation, and love to those

... They carried the most sacred message in life

... to those who paved our way to knowledge and knowledge

... to all our illustrious professors Thanks and appreciation to Dr.Mekhaznia I don't forget the Examiner Pr.MR. Bendjena. H and Mr.Tag.S.

Which we say to them with the words of the Messenger of Allah peace be upon all of you

"The whale in the sea and the bird in the sky to guard the teacher of the people"

We thank all those who helped to complete this research, gave us help and provided us with a helping hand

Be a scientist ... If you cannot be educated ... If you cannot love scientists ... if not You cannot hate them

Contents

List of Figures

List of Tables

General Introduction

Since the middle of the 20th century and, for the past 60 years, Quantum technologies contributed to small scale developments and improvements. However, this field has grown significantly and its technologies are becoming increasingly relevant across different sectors, with great potential impact on community. We are about entering new era, the age of quantum technology.

Quantum technologies will effect most of the emerging technologies we know today, empowering many of them and at the same time will threat others security. They will disrupt current technology used in biology, genetics, medicine, education, economy and finance, energy, agriculture, transportation, and meteorology, among others, achieving a high social impact as well. So the global forces and the well known technology firms are investing massively to understand, develop and implement these new technologies.
This report is divided into four chapters that explain different aspects of quantum technologies, including the way they work, their inevitable impact, and the actions technology firms are taking to incorporate them into their programs and infrastructure.

In this work we will explain some quantum basics from Qubits, superposition,logic gates, Quantum Dechoerence, Quantum Reversibility, Quantum entanglement and how this quantum com-

puters works and their forecasts also we will discuss the differents between classical and quantum computers, programming language useually used to solve problems and the impact of quantum computing on some industries and present some details of today realistic quantum computers for public use that belongs to IBM and D-Wave companies, then we will pass to the soft side of quantum computing some known algorithms and problems Grover search algorithm, Deutsch-Jozsa, Bernstein-Vazirani, shor's factoring ...etc with the implemantation of grover's and shor's algorithms on real quantum computers (IBM Q EXPERIENCE) and discuss the results of outputs, then we will finish with analysis of quantum programming environment such quantum complexity, programming tools and compared it to the tools used in classical programming.

Chapter 1

Introduction to Quantum Computing

1.1 Introduction

Nowadays, it is more important to think about changing modern computing because processor manufacturing is reaching its limits. The specialists in this field said that the limit will be reached with the arrival of the processors etched in 5 nm by the horizon 2021, the engraving fineness being the minimum width of a channel of a transistor. Other solutions must be found, and quantum computing is one of them.

In this chapter, the basic principle of this new technology will be detailed, as well as its usefulness in computing through quantum processors. We will also discuss what a Qubit is, superposition... Finally, we will study the possible applications of quantum computation that will improve our daily lives.

1.2 Computers and Quantum Mechanics

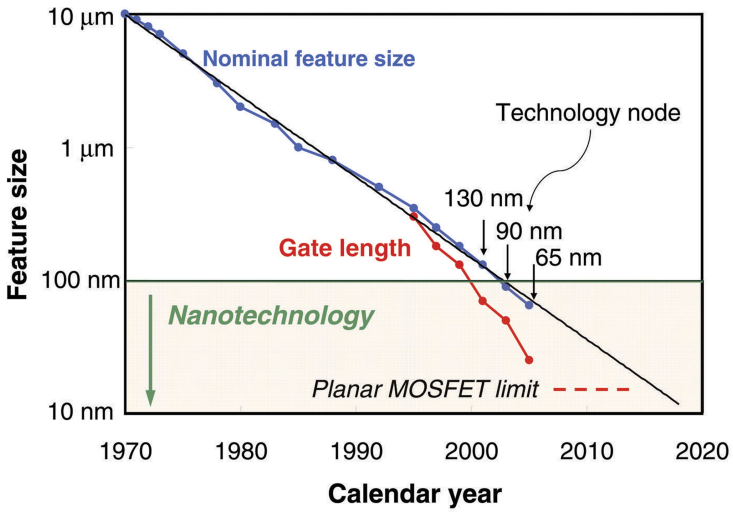

Figure 1.1: Logic technology node and transistor gate length versus calendar year.(Published at the International Electron Devices Meeting 2006.)

To understand quantum computing, we must first understand what quantum mechanics are, Quantum mechanics is the set of laws showing that all the particles of the universe are matter and therefore are subject to force fields. [1] According to these laws, particles such as photons or electrons are also, in a way, waves and are therefore subject to other physical principles than those described in classical physics. For example, a particle can be in two places at once. All quantum physics is based on probability amplitudes, that is, all the stated principles are governed by a probability of success. These amplitudes represent phenomena of interference, of diffraction, because these particles are also considered as waves and thus respond to the specific laws of these. Following these principles, energy is then quantified in many physical processes, including the electromagnetic field and atoms. [2]

1.2.1 Moore's Law

The micro devices and digital electronics development related to the advance of semi conductors manufacturing that have paved the way to smaller, powerful and faster components. [3] This decreasing in size precisely has allowed the chip to be loaded with more components, this will add more functions the chip so it will be more powerful. Simultaneously, the decreasing in size is necessary to make faster devices. As early as 1965, Gordon Moore noticed that the number of components that could be implanted on a chip had increased exponentially over many years, while the feature size had shrunk at a similar rate (Figure1.1) [4]
. the feature size of electronic devices is now in the range of 20 nm and decreasing at a rate of some 10% per year.

1.2.2 Quantum Decoherence

The decoherence is making a quantum system into an clearly classical state. the difference between quantum system and a classical system is the notion of a superposition. In classical physics, we say that a particle is at a position (x,y,z), but in quantum, the formalism allows us to state that a particle is in a superposition of positions (say $(x1, y1, z1)$ and $(x2, y2, z2)$). [5] However, as the quantum mechanics laws postulates, when we actually measure the position of this particle, we will find it at either of the two positions, that is, we will have "collapsed the wave function" into one or the other state. We call this "measurement problem". [5] David Albert puts it better when he writes, The dynamics and the postulate of collapse are flatly in contradiction with one another ... the postulate of collapse seems to be right about what happens when we make measurements, and the dynamics seems to be bizarrely wrong about what happens when we make measurements, and yet

the dynamics seems to be right about what happens whenever we aren't making measurements.
Let us be more explicit about what he means here.
It was von Neumann who first articulated the so called "dynamical dualism" that haunted the original formulations of quantum mechanics, though Bohr touched on the issue earlier in proposing the "quantum leap" into states which, like the wave collapse, is proposed as a dynamically discontinuous process. Primarily, the evolution of a quantum system is described by the Shrödinger equation [5] :

$$i\hbar\frac{\delta}{\delta t}\left|\psi\right\rangle = H\left|\psi\right\rangle \tag{1.1}$$

where i is the imaginary unit, $\hbar$ is the reduced Planck constant, ψ is the state vector of the quantum system, t is time, and H is the Hamiltonian operator.

applying to a system under isolation. There is also the evolution that occurs due to measurement, and this is the infamous collapse:

$$\left|\psi\right\rangle = \Sigma cn\left|n\right\rangle \longrightarrow \left|ni\right\rangle \tag{1.2}$$

Technically, Equation 2 is called by von Neumann the "first intervention" and Equation 1 is the "second intervention." The first intervention is what we will identify as the wave collapse. It describes a superposition of states suddenly 'collapsing' into one eigenstate, the measurement. [5]

1.3 Quantum Calculations

1.3.1 Difference between Quantum and Classical Computers

Consider some physical system that can be in N different, mutually exclusive classical states, Because we will typically start counting from 0, we call these states $|0\rangle, |1\rangle, ... |N-1\rangle$ Roughly, by a "classical" state we mean a state in which the system can be found if we observe it.

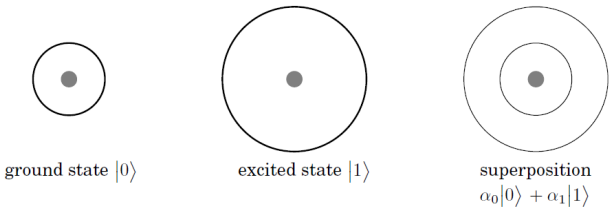

Figure 1.2: An electron States

A pure quantum state $|\phi\rangle$ is a superposition of classical states, written $|\phi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle +, ..., +\alpha_{N-1} |N-1\rangle$.

The memory of a classical computer is composed from strings of 0s and 1s, and it can execute operations on only one set of numbers simultaneously. The memory of a quantum computer is a quantum state that can be a superposition of different numbers. Quantum computer can do an arbitrary reversible classical computation on all the numbers simultaneously, making a computation on many different numbers at the same time and then interfering all the results to get a single answer, This makes the quantum computer much powerful than a classical one. [6]

1.4 Quantum Computing Prerequisites

1.4.1 Qubit

A qubit is a quantum system in which the Boolean states 0 **and** 1 **are represented by a prescribed pair of normalised and mutu-ally orthogonal quantum states labeled as** $|0\rangle, |1\rangle$**.The** $|0\rangle$ **is called "ground state;" the** $|1\rangle$ **is the "excited state". [9] As the most general electronic state is a superposition of the two basic states [9], we then have**

$$|\Psi\rangle = a\,|0\rangle + b\,|1\rangle \tag{1.3}$$

that is, a normalized vector, with $a, b \in C$. [9, 10, 11]

The two states form a computational basis and any other (pure) state of the qubit can be written as a superposition $\alpha\,|0\rangle + \beta\,|1\rangle$ **for** α **and** β **such as** $\mid \alpha \mid^2 + \mid \beta \mid^2 = 1$**. [9] Habitually, a qubit is a microscopic system, such as an atom, a nuclear spin, or apolarised photon. [9]**

1.4.2 Quantum Register

A quantum register is the quantum mechanical analogue of a classical processor register. A mathematical description of a quantum register is achieved by using tensor products of qubit bra or ket vectors. For example, a quantum register of size 4 can store individual numbers such as 13:

$$|1\rangle \otimes |1\rangle \otimes |0\rangle \otimes |1\rangle \equiv |1101\rangle \equiv |13\rangle \tag{1.4}$$

where $\otimes$ **denotes the tensor product.It can also store the two of them simultaneously.**

1.4.3 Quantum Logic Gates

For this, and for other manipulations on qubits, uni-tary operations have to be performed. Naturally, the definitions of quantum logic gate and quantum net-work follow. [12]

A quantum logic gate is a device which performs a fixed unitary operation on selected qubits in a fixed period of time. [9]

A quantum network is a device consisting of quantum logic gates whose computational steps are synchronised in time. A quantum gate acts on superpositions of different basis states of qubits, whereas classically this option is nonexistent. [11] Basic gates used in quantum computation are namely the Hadamard gate, the NOT gate, the C-NOT (Controlled-NOT, also known as as the XOR or the measurement gate [9]) gate, the controlled phase-shift gate, the Toffoli gate and the Fredkin gate. [9, 10, 11]

1.4.4 Quantum Process

Quantum computers do calculations based on the probability of an object's state before it is measured instead of just 1s or 0s which means they have the potential to process exponentially more data compared to classical computers that carry out logical operations using the definite position of a physical state, These are usually binary, meaning its operations are based on one of two positions [13].
However in quantum computing operations use the quantum state of an object to produce the qubit. These states are the undefined properties of an object before they've been detected, such as the spin of an electron or the polarisation of a photon.
Rather than having a clear position, unmeasured quantum states

occur in a mixed superposition, not unlike a coin spinning through the air before it lands in your hand, these superpositions can be entangled with those of other objects, meaning their final outcomes will be mathematically related even if we don't know yet what they are, the complex mathematics behind these unsettled states of entangled 'spinning coins' can be plugged into special algorithms to make short work of problems that would take a classical computer a long time to work out if they could ever calculate them at all, Such algorithms would be useful in solving complex mathematical problems, producing hard-to-break security codes, or predicting multiple particle interactions in chemical reactions. [13]

1.5 Comparison between Classical and Quantum computing

Quantum Computing is the use of all the possibilities of quantum mechanics laws to solve computational problems.
Conventional, or Classical computers only use only a small part of these possibilities. basically, they compute in the same way as our brains.[14] We would achieve great results if only we had a large enough quantum computer, we would be able to perform simulations of quantum mechanical processes in physics, chemistry and biology, which will never make it using classical computers. Let's compare some aspects of classical and quantum computers. [14] See Table 1

Specif	Comparison
Information	*In Classical Computing* a Information is stored in bits, which take the discrete values 0 and 1. b storing one number takes 64 bits, then storing N numbers takes N times 64 bits. *In Quantum Computing* a Information is stored in quantum bits, or qbits. A qbit can be in states labelled $\|0\rangle$ and $\|1\rangle$, but it can also be in a superposition of these states,$a\|0\rangle + b\|1\rangle$, where a and b are complex numbers. b If we think of the state of a qbit as a vector, then superposition of states is just vector addition.For every extra qbit you get, you can store twice as many numbers
Calculation	*In Classical Computing* a Calculations are done essentially in the same way as by hand. *In Quantum Computing* a perform calculation by unitary transformations on the state of the qbits. Using the principle of superposition, this creates possibilities impossible for hand calculations. This is the base of efficient algorithms like factoring algorithm, searching and simulation algorithms of quantum mechanical systems.

Table 1.1: Comparison Between Classical and Quantum Computing [14]

1.6 Quantum Reversibility

Consider the Boolean AND gate. We can't conclude the inputs of an AND gate from the outputs, because of its truth table, the And gate is not reversible.
A gate of AND type produces waste heat when working (i.e. giving outputs to its inputs). The "lost" information about the inputs are contained in this waste heat, this situation must not appear in quantum computers, the radiation of the heat depends on the state of the inputs to the quantum gate. Thus, in effect, the radiation of the heat would be a measurement on the inputs and decoherence would ensue. The universes would be so far apart as to be unable to interfere with each and the result, which depends upon the interference of these universes, would be invalid. Thus, quantum gates have to be reversible.
Reversible gates must, by their very definition, have an equal number of inputs and outputs n [15]. More formally, an n-bit reversible gate is a bijective map-ping f from the set $0, 1^n$ of n-bit data to itself. Reversible gates are also useful as they would be the only potential way to improve the energy efficiency of computers beyond the fundamental von Neumann-Landauer limit of [12]

$$kTln2 \tag{1.5}$$

energy dissipated per irreversible bit operation, where k is Boltzmann's constant of $1.38 \cdot 10^{23} J/K$, and T is the temperature of the environment into which unwanted entropy will be expelled. [16]

in Conclusion because Quantum computers work by applying quantum gates to quantum states. The evolution of quantum states is restricted by the unitarity property of quantum mechanics; that

is, every operation on a (normalized) quantum state must keep the sum of probabilities of all possible outcomes at exactly 1.Any quantum gate must thus be implemented as a unitary operator,and is therefore reversible.

1.7 Quantum Entanglement

We say that a pure state of two qubits is entangled if it cannot be written as a product of the individual states of the two qubits (thus, with a tensor product), such as $|v1\rangle \otimes |v2\rangle$ For example, the EPR (Einstein-Podolski-Rosen) state is not decomposable into a direct product of any form, and is therefore entangled:

$$|\Psi EPR\rangle = \frac{(|01\rangle + |10\rangle)}{\sqrt{2}} \tag{1.6}$$

as two qubits in this state display a degree of correla- tion impossible in classical physics and hence violate the Bell inequality which is satisfied by all local (i.e. classical) states. At the opposite, for two qubits $(n = 2)$, the state

$$\alpha\,|00\rangle + \beta\,|01\rangle = |0\rangle \otimes (\alpha\,|0\rangle + \beta\,|1\rangle) \tag{1.7}$$

is separable: $|\Psi 1\rangle = |0\rangle$ and $|\Psi 2\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$.
The exploitation of a number of entangled qubits can lead to a considerable computational speed-up in a quantum computer over its classical counterpart.This leads us to the interest of using quantum computers.[11]

1.7.1 Forecasts

The interest of these machines also makes it possible, through its unique computing capabilities, to make predictive calculations in

the field of finance, thanks to an easy calculation of the stochastic nature of the stock markets. We could also easily calculate climate models, and thus make weather forecasts safer.

1.7.2 Calculations

Finally, in basic mathematics, which is at the origin of quantum mechanics, the applications are many, but the most obvious, by their combinatorial nature, are : [17]

1. Products of prime factors and prime numbers
2. The discrete logarithm
3. Searching a database

1.8 Quantum Programming

Using quantum programming, one can allow the expression of quantum algorithms using high-level constructs. Its aim is to provide a tool for researchers to understand better how quantum computation works and how to formally reason about quantum algorithms. [12]

1.8.1 Imperative Quantum Programming Languages

Imperative languages are described by specifying how the execution of a given program modifies a global state. The imperative approach has the following two advantages: First, it models more closely how actual hardware works and thus makes it easier to actually implement these languages. Second, programmers without a background in formal languages tend to find the imperative approachmore natural; in fact, programming is mostly done in imperative programming languages like C++ and Java. [18]

1.8.1.1 Quantum Pseudocode

Quantum pseudocode proposed by E. Knill is the first formalised language for description of quantum algorithms was introduced and, moreover, it was tightly connected with model of quantum machine called Quantum Random Access Machine (QRAM) [19].

1.8.1.2 Quantum Computer Language

After it, a Quantum Computer Language (QCL) was proposed. It is one of the first implemented quantum programming languages. Its syntax similar to the syntax of the C programming language and classical data types are similar to data types in C. Quantum data type in QCL is based on the qureg (quantum register). It can be interpreted as an array of qubits (quantum bits). An example of such a code will be like this. [20]

```
qureg x1[2]; // 2-qubit quantum register x1          (1.8)

qureg x2[2]; // 2-qubit quantum register x2          (1.9)

H(x1);// Hadamard operation on x1                    (1.10)

H(x2[1]);// Hadamard operation on the first qubit of the register x2
                                                     (1.11)
```

As the qcl interpreter uses qlib simulation library, it is possible to observe the internal state of the quantum machine during execution of the quantum program:

```
qcl> dump: STATE: 4 / 32 qubits allocated,
```

28 / 32 qubits free 0.35355 |0>+ 0.35355 |1> + 0.35355 |2> + 0.35355 |3> + 0.35355 |8> + 0.35355 |9> + 0.35355 |10> + 0.35355 |11>

Fortunately, the dump operation is different from measurement, since it does not influence the state of the quantum machine and can be realised only using a simulator. Mainly, the QCL standard library pro- vides standard quantum operators used in quantum algorithms such as: [21]

1. controlled-not with many target qubits,
2. Hadamard operation on many qubits,
3. parse and controlled phase.

The most important feature of QCL appears to be the support for user-defined operators and functions. Like in modern programming languages, it is possible to define new operations which can be used to manipulate quantum data. [21]

1.8.1.3 Q Language

Q Language was implemented as an extension of C++. It offers classes for basic quantum operations like QFourier, QHadamard, QNot, and QSwap, New operators can be defined using C++ class mechanism. Q code example. [22]

Qreg x1(); // 1-qubit quantum register with initial value 0 (1.12)

Qreg x2(2,0); // 2-qubit quantum register with initial value 0 (1.13)

Computation process is executed using provided simulator. Noisy environment can be simulated using parameters of the simulator.[22]

1.8.1.4 qGCL

Quantum Guarded Command Language (qGCL) was defined by P. Zuliani in his PhD thesis. It is based on Guarded Command Language created by Edsger Dijkstra. It can be described as a language of quantum programmes specification. [23]

1.8.2 Functional Quantum Programming Languages

During the last few years many quantum program- ming languages based on the functional program- ming paradigm were proposed. Functional program- ming languages are well-suited for reasoning about programs.

1.8.2.1 QFC and QPL

QFC and QPL are two closely related quantum programming languages defined by Peter Selinger. They differ only in their syntax: QFC uses a flow chart syntax, whereas QPL uses a textual syntax. These languages have classical control flow, but can operate on quantum or classical data. Selinger gives a denotational semantics for these languages in a category of superoperators. [12]

1.8.2.2 QML

QML is a Haskell like quantum programming language by Altenkirch and Grattage. Un like Selinger's QPL, this language takes duplication, rather than discarding, of quantum information as a primitive operation. Duplication in this context is understood to be the operation that maps $|\phi\rangle$ to $|\phi\rangle \otimes |\phi\rangle$. [24]

1.8.3 Quantum Lambda Calculi

Quantum lambda calculi are extensions of the lambda calculus, introduced by Alonzo Church and Stephen Cole Kleene in the 1930s. The purpose of *quantum lambda calculi* is to extend quantum programming languages with a theory of higher order functions. [24]
The first attempt to define a quantum lambda calculus was made by Philip Maymin in 1996, His lambda-q calculus is powerful enough to express any quantum computation. This language can efficiently solve NP-complete problems, and therefore appears to be strictly stronger than the standard quantum computational models (such as the quantum Turing machine or the quantum circuit model). [24]
In 2003, Andre van Tonder defined an extension of the lambda calculus suitable for proving correctness of quantum programs. He also provided an implementation in the Scheme programming language.
In 2004, Selinger and Valiron defined a strongly typed lambda calculus for quantum computation with a type system based on linear logic. [24]

1.9 Quantum Interest

1.9.1 Cryptography

Quantum computation will affect Cryptography and cryptanalysis, because cryptography is controled by algorithms that are not easy to compute using standard computer, the complexity of calculations increases directly according to the size of the key (a 4096-RSA key bits is now very hard to calculate and even more to decode without the encoding key). But with quantum computation, we

get rid of the binarity computations, composed of 0 and 1, and bases of almost infinite size could be used. [25]
The benefit of a qubit in cryptography is that we can't reproduce the state of a qubit, so only machines with the decryption key created according to the state of the qubit at the moment of creation of the key would read the encrypted data. [25]

1.9.1.1 IronBridge

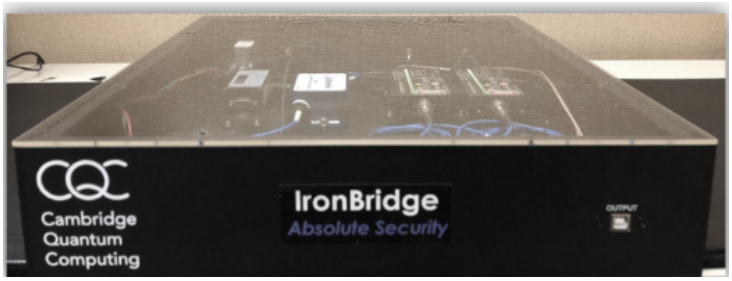

Figure 1.3: Server-rack device Entangled photonics 16 Mbp. (Published at Supercomputing asia 2019)

Cambridge Quantum Computing has recently developed IronBridge™
The world's first commercially ready certifiable quantum cryptographic device.
Cambridge Quantum Computing uses photonics and quantum entanglement to generate truly random numbers. They also constantly verify that quantum randomness is being maintained. Failure to quantum randomness tests indicates that there has been interference and tampering.
This practical solution allows governments and businesses around the world to attain unparalleled levels of quantum-enhanced encryption and security relating to much of the technology that underpins daily digital interactions including IoT, big data, cloud infrastructure, networks and communications, By slotting neatly

into existing network configurations, IronBridge provides a solution that works today whilst simultaneously protecting against the threats of tomorrow. [26]

1.9.2 Artificial Intelligence

The applications in the field of artificial intelligence are huge, because the fact that we are able to get rid of the calculation of the possibilities on a binary basis makes it possible to perform complex combinatorial computations, which is the calculation principle making it possible to solve all the possibilities of an equation (For Exmple HHL Alogrithme). An artificial intelligence must be able to perform this task extremely quickly, which is impossible with conventional processors without using a super computer, for exemple artificial intelligence such as machine learning muche more powerful when data sets are very large such as in searching images or video. [27]

1.9.2.1 HHL: Solving Linear Systems of Equations

HHL (named after its discoverers: Harrow, Hassidim and Lloyd) is an algorithm for solving a system of linear equations: given a matrix A and a vector b, find a vector x such that $Ax = b$. While this may seem like an esoteric problem unrelated to machine learning, they are in fact intimately related. For example, the HHL algorithm can be employed to give a speedup in perceptron training, that is exponential in the size of the training vectors. It is also the underlying mechanism behind data-fitting procedures such as linear regression and, as such,becomes a workhorse for generic classification problems. [28, 29]

1.10 Realistic Quantum Computers

1.10.1 IBM Q

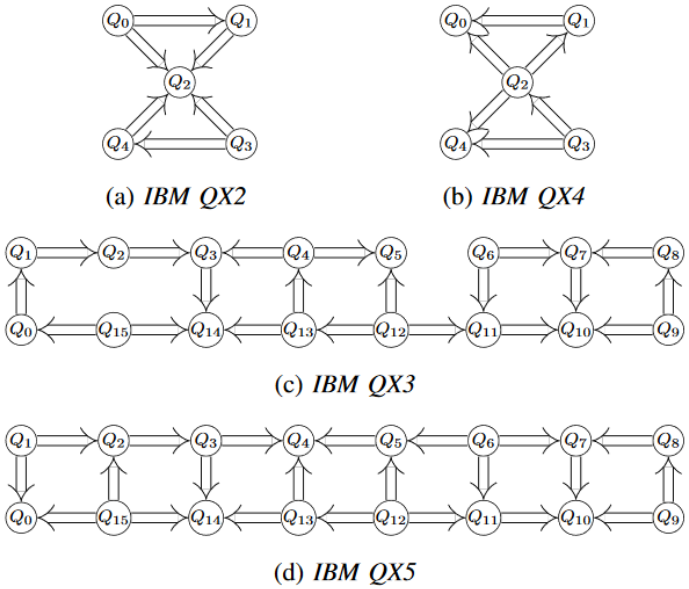

Figure 1.4: coupling map of different IBM quantum devices [32]

IBM's project IBM Q, which launched in March 2017 with the goal to provide access to a quantum computer to the broad audience, can be seen as evidence of this progress.Initially, they started with the 5 qubit quantum processor IBM QX2, on which anyone could run experiments through cloud access. In June 2017, IBM added a 16 qubit quantum processor named IBM QX3 to their cloud [30] and, thus,more than tripled the number of available qubits within afew months. Since then, IBM has been working intensely on improving their quantum computers – leading to 5-qubit and 16-qubit quantum computers (named IBM QX4 and IBM QX5,respectively) which were added to the cloud in September 2017. [31]

The rapid progress in the number of available qubits is still going on. While IBM has already manufactured a 20-qubit quantum computer which is available for their partners and members of the IBM Q network show quantum supremac. [31]

1.10.1.1 Ibmqx5

The ibmqx5 is a 16-qubit quantum computer manufactured by IBM available to the public as a part of the IBM Q cloud service. It is accessible through a programming interface called QISKit.

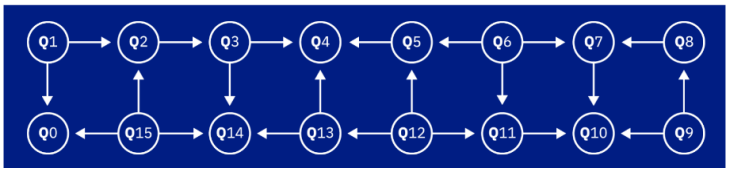

Figure 1.5: Coupling map of ibmqx5
[34]

The computer uses a superconductivity transmon qubit implementation which has consequences for algorithm design. The possible inter-actions between qubits are controled by the superconducting bus connections between them [33] and can be described by the ibmqx5 coupling map. In the coupling map in Figure 1.5 an arrow from qubit A to qubit B represents that a cNOT gate can be created with A as the control bit and B as target.

1.10.1.2 QISKit

QISKit is a Python-based programming interface for programming quantum computers. The interface provides access to the quantum computers available though IBM Q and also provides access to a local simulator for making simulation runs. The simulator can be

configured to have a coupling map equal to that of ibmqx5.The QISKit interface provides a built in mapper for mapping a qubit in thecode to a hardware qubit. Because of this the qubits can be arbitrarily named in the QISKit code. The mapper works provided that the requested implementation can be made to fit the coupling map. [33]

1.10.1.3 The IBM Q Network

IBM Q Network is a worldwide organization of hubs, members, and partners enabled by IBM Q systems with the shared mission to :

a Collaborate with the most advanced academic and research organizations to advance quantum computing technology.

b Engage industry leaders to combine IBM's quantum computing expertise with industry specific expertise to accelerate development of the first commercial use cases

c Expand and train the ecosystem of users, developers, and application specialists that will be essential to the adoption and scaling of quantum computing.

1.10.2 The D-Wave 2000Q

The D-Wave 2000Q™ quantum computer leverages quantum dynamics to accelerate and enable new methods for solving discrete optimization, sampling, and machine learning problems. D-Wave systems use a process called quantum annealing to search for solutions to a problem. Quantum annealing is fundamentally different from classical computing.

The Computation is performed by initializing the quantum processing unit (QPU) into a ground state of a known problem and annealing the system toward the problem to be solved such that it remains in a low energy state throughout the process. At the end of the computation, each qubit ends up as either a 0 or 1. This final state is the optimal or near-optimal solution to the problem to be solved. [35]

1.10.2.1 D-Wave 2000Q Specification

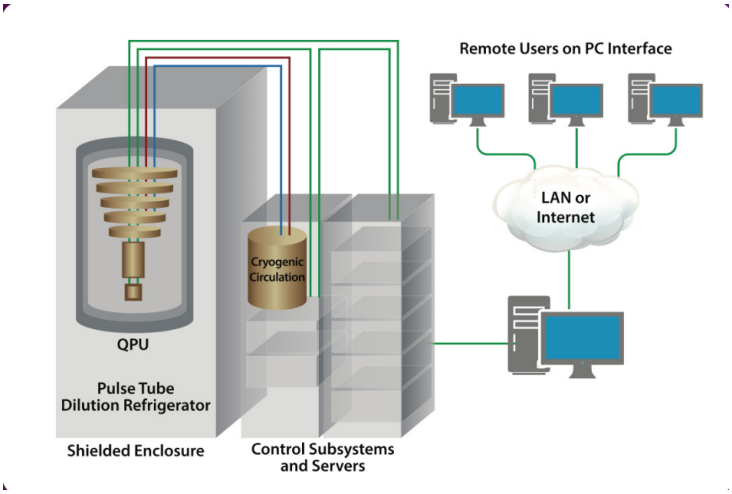

Figure 1.6: D-Wave 200Q Architecture
[35]

The D-Wave 2000Q system has a footprint of approximately 10' x 7' x 10'(L x W x H). Its physical enclosure houses sophisticated cryogenic refrigeration, shielding, and I/O systems to support a single thumbnail-sized QPU. Most of the physical volume of the system is required to accommodate the refrigeration system and to provide easy service access.For quantum effects to play a role in computation, the QPU requires an extreme, isolated environment.

The refrigerator and layers of shielding create an internal high vacuum environment with a temperature close to absolute zero that is isolated from external magnetic fields, vibration, and RF signals of any form. Adjoining cabinets contain the control subsystems and the front-end servers that provide connectivity to the system.[36]

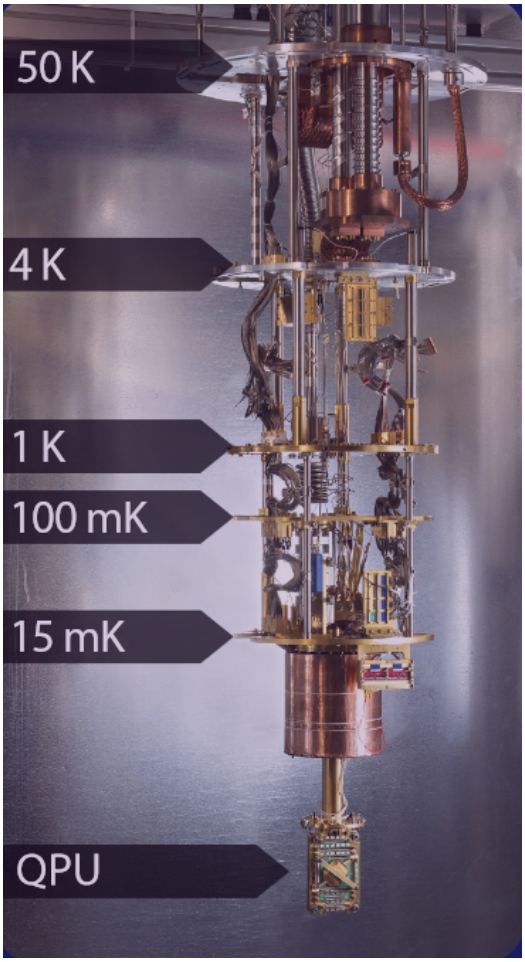

Figure 1.7: Starting at room temperature at the top, the temperature decreases at each level until it is close to absolute zero where the QPU itself is located.

[35]

The D-Wave 2000Q system operates near absolute zero. This extremely low temperature, along with the shielded environment that isolates the QPU from its surroundings, enables the QPU to behave quantum mechanically, D-Wave systems operate at less than 15 millikelvin, approximately 180 times colder than interstellar space.

D-Wave's "dry" dilution refrigerator uses liquid helium refrigerant in a closed-loop system, avoiding the need for on-site replenishment.While dilution refrigerators are not uncommon in research environments, D-Wave has advanced the technology to ensure long run-life and reliability in a commercial product setting. Despite the extreme environment inside the system, a standard data center can normally accommodate the D-Wave 2000Q quantum computer. [35]

The extreme isolated environment required for the QPU places unusual demands on the design, materials, and manufacturing processes required for the various subsystems. [35] The I/O subsystem that passes information to the QPU and back while filtering out all unwanted noise requires a variety of normal and superconducting materials to provide the required performance.The magnetic shielding subsystem provides the low-field environment required for the QPU, using high-permeability and superconducting materials to achieve fields below 1 nanotesla. This is 50,000 times less than the Earth's magnetic field. [35]

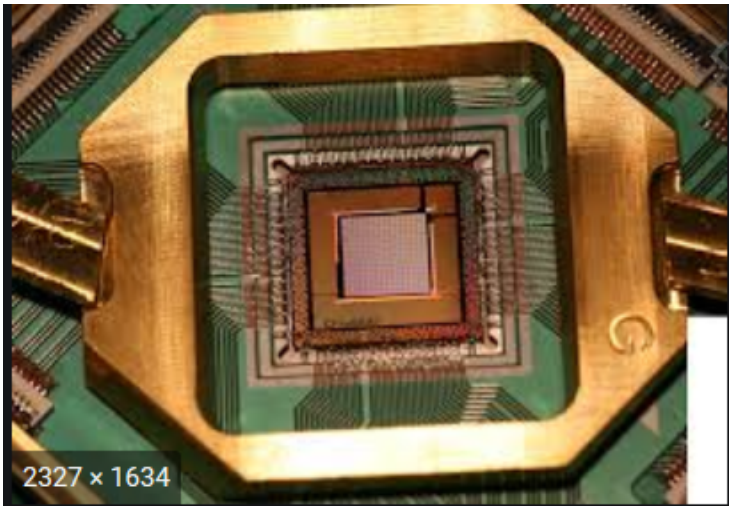

Figure 1.8: D-Wave2000Q QPU
[35]

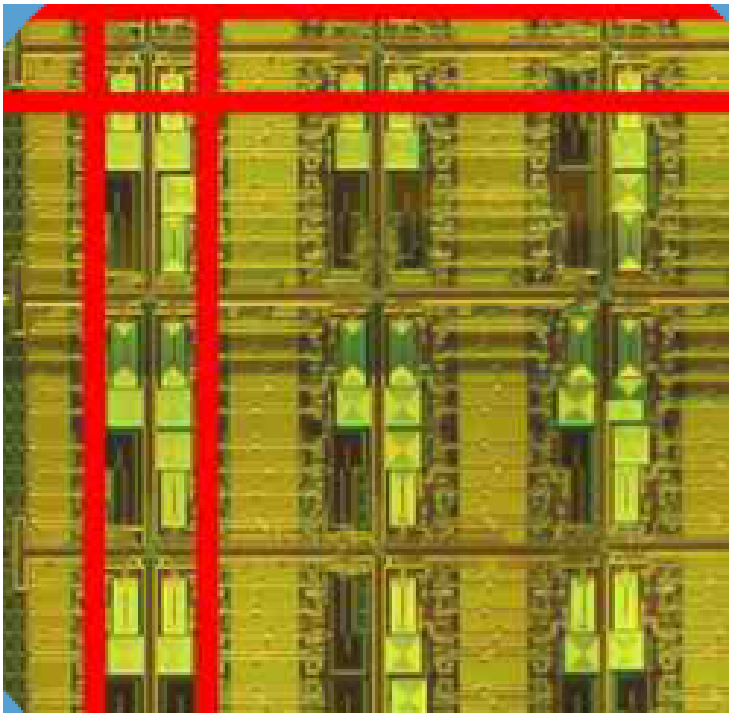

Figure 1.9: Qubits inside QPU
[35]

The D-Wave QPU is built from a lattice of tiny loops of the metal niobium, each of which is one qubit(shown on the Figure 1.9). Below temperatures of 9.2 kelvin, niobium becomes a superconductor and exhibits quantum mechanical effects.When in a quantum state, current flows in both directions simultaneously, which means that the qubit is in superposition—that is, in both a 0 and a 1 state at the same time. At the end of the problem-solving process, this superposition collapses into one of the two classical states, 0 or 1. [35]

Going from a single qubit to a multi-qubit QPU requires that the qubits be inter connected to exchange information. Qubits are connected via couplers, which are also superconducting loops. The

interconnection of qubits and couplers, together with control circuitry to manage the magnetic fields, creates an integrated fabric of programmable quantum devices [37]. When the QPU arrives at a solution to a problem, all qubits settle into their final states and the values they hold are returned to the user as a bit string.The D-Wave 2000Q system has up to 2048 qubits and 5600 couplers. To reach this scale, it uses 128,000 Josephson junctions, which makes the D-Wave 2000Q QPU by far the most complex superconducting integrated circuit ever built.

Unlike the CPUs of classical computers, D-Wave's superconducting QPU dissipates negligible amounts of heat during computation. While traditional supercomputers generate massive amounts of heat and consume massive amounts of power, the D-Wave system consumes less than 25 kW of power, most of which goes towards operating the cooling and front-end servers. This low power consumption has remained constant since the introduction of the first D-Wave system despite the dramatic increase in system performance with each successive product generation. The required water cooling is on par with what a kitchen tap can provide. The required air conditioning is one-tenth of what would be expected in a data center for a system with a similar footprint.As more powerful D-Wave systems are released in the future, power consumption will remain constant, resulting in huge increases in performance per watt and per dollar. Tens of kilowatts means tens of thousands of dollars in operating costs per year in contrast to millions of dollars per year for even a modest high-performance computer system that consumes megawatts of power. If realized today, exascale supercomputers would consume power on the order of that produced by the Hoover dam. [35]

1.10.2.2 The D-Wave 2000Q Application

Table 1.2: D-Wave 2000Q Application Areas

Machine Learning & Computer Science	Security & Mission Planning
• Detecting statistical anomalies • Finding compressed models • Recognizing images and patterns • Training neural networks • Verifying and validating software • Classifying unstructured data • Diagnosing circuit faults	• Detecting computer viruses • Detecting network intrusion • Scheduling resources & optimal paths • Determining set membership • Analyzing graph properties • Factoring integers
Financial Modeling	Healthcare & Medicine
• Detecting market instabilities • Developing trading strategies • Optimizing trading trajectories • Optimizing asset pricing and hedging • Optimizing portfolios	• Detecting fraud • Generating targeted cancer drug therapies • Optimizing radiotherapy treatments • Creating protein models

D-Wave quantum computers are ideally suited to solving many hard problems in optimization, machine learning, sampling and cyber security. With 2000 qubits and new control features, the D-Wave 2000Q quantum computer can solve larger problems than was previously possible, and with better performance. A growing community of developers are using the unique capabilities of D-Wave systems to solve challenging problems in a diverse set of application areas. [35] (See Table 1.2).

1.10.2.3 Software and Programming

Just Like classical computing community need a software ecosystem to build a society of application developers and users, the quantum computing world does as well. D-Wave, new quantum software companies, D-Wave customers, and users are starting

to develop system software, higher level tools [38], and applications that leverage the power of the D-Wave system, the D-Wave 2000Q system provides a standard Internet API (based on RESTful services), with client libraries available for C/C++, Python, and MATLAB, This interface allows users to access the system either as a cloud resource over a network, or integrated into their high-performance computing environments and data centers [39]. Access is also available through D-Wave's hosted cloud service. Using D-Wave's development tools and client libraries, developers can create algorithms and applications within their existing environments using industry-standard tools. While users can submit problems to the system in a number of different ways, ultimately a problem represents a set of values that correspond to the weights of the qubits and the strength of the couplers, the system takes these values along with other user-specified parameters and sends a single quantum machine instruction (QMI) to the QPU [37], Because quantum computers are probabilistic rather than deterministic, multiple values can be returned, providing not only the best solution found, but also other very good alternatives from which to choose. Users can specify the number of solutions they want the system to return. [40]

1.10.3 Quantum Computers Price and forecasts

Today, a single qubit will cost you $10,000$ without mentioning research and development costs, Temporal Defense Systems a cyber security company bought D-wave 2000Q with 15 million dollar and aim to use it for cyber security problems. For public it will be possible to access the 2000Q online through a subscription service, there is also a quantum computing platform named The "IBM Quantum Experience" available via the IBM Cloud service for public to test

algorithms and experiments.
Initial applications will leverage algorithms that can tolerate or mitigate errors found in approximate quantum computers. [41]

1.11 Conclusion

The quantum computing revolution just started and it will replace today techniques and solve previously complexed problems, creating real solutions for different secteurs.
Quantum computers is the ideal solution for complexed systems that needs to be simulated, predicting the financial markets, improving forecasts, to modelling the behaviour of individual electrons, using quantum computing to understand quantum physics, Cryptography will be another key application, in a world with quantum computers, the systems that currently safe guard business transactions on the Internet (based on the RSA) will no longer be secure.

Chapter 2

Quantum Software

2.1 Introduction

In this chapter we explain how a quantum computer can apply computational steps to its register of qubits. Two models exist for this: the quantum Turing machine [42] and the quantum circuit model [43]. These models are equivalent, in the sense that they can simulate each other in polynomialtime, assuming the circuits are appropriately "uniform." We only explain the circuit model here,which is more popular among researchers,Then we pass to the different elementary quantum gates and explain its operations and the most known quantum algorithmes Like Grover search algorithm, Shor's Factoring algorithm, Deutsch-Jozsa Algorithm with their mathematical implemantation.

2.2 Quantum circuits

In classical complexity theory, a Boolean circuit is a finite directed a cyclic graph with AND, OR,and NOT gates. It has n input nodes, which contain the n input bits $(n0)$. The internal nodes are AND, OR, and NOT gates, and there are one or more designated output

nodes. The initial input bits are fed into AND, OR, and NOT gates according to the circuit, and eventually the output nodes assume some value. We say that a circuit computes some Boolean function $f : \{0\,,1\}^n \to \{0\,,1\}^m$ if the output nodes get the right value $f(x)$ for every input $x \in \{0,1\}^n$. A circuit family is a set $C = \{C_n\}$ of circuits, one for each input sizen. Each circuit has one output bit. Such a family recognizes or decides a language $L \subseteq \{0,1\}^* = U_{n\geq 0}\{0,1\}^n$ if, for every n and every input $x \in \{0,1\}^n$.the circuit C_n outputs 1 $x \in L$ and outputs 0 otherwise.Such a circuit family is uniformly polynomial if there is a deterministic Turing machine that outputs C_n given n as input, using space logarithmic in n.Note that the size (number of gates) of the circuits C_n can then grow at most polynomially with n. It is known that uniformly polynomial circuit families are equal in power to polynomial-time deterministic Turing machines: a languageL can be decided by a uniformly polynomial circuit family if $L \in P$ where P is the class of languages decidable by polynomial-time Turing machines.

In quantum computers can carry out a Fourier transform exponentially faster than classical computers. But what do these computers actually look like? What is a quantum circuit made up of, and exactly how does it compute Fourier transforms so quickly? [6].

2.3 Elementary quantum gates

An elementary quantum operation is analogous to an elementary gate like the AND or NOT gate in a classical circuit. It operates upon either a single qubit or two qubits. One of the most important examples is the Hadamard gate, denoted by H, which operates on a single qubit.On input $|0\rangle$ **[44]**

it outputs

$$H(|0\rangle) = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \tag{2.1}$$

And for input $|1\rangle$

$$H(|1\rangle) = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle \tag{2.2}$$

Notice that in either case, measuring the resulting qubit yields 0 with probability $1/2$ and 1 with probability $1/2$. But what happens if the input to the Hadamard gate is an arbitrary superposition $\alpha_0|0\rangle + \alpha_1|1\rangle$? The answer, dictated by the linearity of quantum physics, is the superposition [44]

$$\alpha_0 H(|0\rangle) + \alpha_1 H(|1\rangle) = \frac{\alpha_0 + \alpha_1}{\sqrt{2}}|0\rangle + \frac{\alpha_0 - \alpha_1}{\sqrt{2}}|1\rangle \tag{2.3}$$

And so,if we apply the Hadamard gate to the output of a Hadamard gate, it restores the qubit to its original state!

Another basic gate is the controlled-NOT, or CNOT. It operates upon two qubits, with the first acting as a control qubit and the second as the target qubit. The CNOT gate flips the second bit if and only if the first qubit is a 1. Thus $\mathbf{CNOT}(|00\rangle) = |00\rangle$ and $\mathbf{CNOT}|10\rangle = |11\rangle$:

(2.4)

Yet another basic gate, the controlled phase gate, is described below in the subsection describing the quantum circuit for the QFT.

Now let us consider the following question: Suppose we have a quantum state on n qubits, $|\alpha\rangle = \sum_{x\epsilon\{0,1\}^n} \alpha_x |x\rangle$.How many of these 2^n amplitudes change if we apply the Hadamard gate to only the first qubit? The answer is all of them! Now the new superposition becomes $|\beta\rangle = \sum_{x\epsilon\{0,1\}^n} \beta_x |x\rangle$, where ,where $\beta_{0y} = \frac{\alpha_{0y}+\alpha_{1y}}{\sqrt{2}}$ and $\beta_{1y} = \frac{\alpha_{0y}-\alpha_{1y}}{\sqrt{2}}$. Looking at the results more closely, the quantum operation on the first qubit deals with each $n-1$ bit suffix y separately. Thus the pair of amplitudes α_{0y} and α_{1y} are transformed into $(\frac{\alpha_{0y}+\alpha_{1y}}{\sqrt{2}})$ and $(\frac{\alpha_{0y}-\alpha_{1y}}{\sqrt{2}})$. This is exactly the feature that will give us an exponential speedup in the quantum Fourier transform.

2.4 Quantum Algorithms

2.4.1 Deutsch-Jozsa

For $N = 2^n$, we are given $x \in \{0,1\}^N$ such that either (1) all x_i have the same value ("constant"), or (2) $N/2$ of the x_i are 0 and $N/2$ are 1 ("balanced").The goal is to find out whether x is constant or balanced.

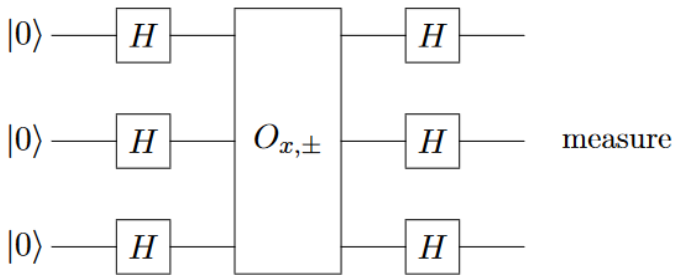

Figure 2.1: The Deutsch-Jozsa algorithm for n= 3
[45]

The algorithm of Deutsch and Jozsa is as follows. We start in the $n-qubit$ zero state $|0^n\rangle$, apply a Hadamard transform to each qubit,

apply a query (in its $\pm-$ form), apply another Hadamard to each qubit, and then measure the final state. As a unitary transformation, the algorithm would be $H^{\otimes n}O_{x,\pm}H^{\otimes n}$ We have drawn the corresponding quantum circuit in Figure 4 (where time again progresses from left to right). Note that the number of wires going into the query is n, not N; the basis states on this sequence of wires specify an n-bit address. [45]

$$H^{\otimes n}\left|i\right\rangle = \frac{1}{\sqrt{2^n}}\sum_{j\in\{0,1\}^n}(-1)^{i.j}\left|j\right\rangle \tag{2.5}$$

Let us follow the state through these operations. Initially we have the state $|0_n\rangle$. By Equa-tion (23), after the first Hadamard transforms we have obtained the uniform superpo-sition of all i :

$$\frac{1}{\sqrt{2^n}}\sum_{i\in\{0,1\}^n}\left|i\right\rangle \tag{2.6}$$

The $O_{x,\pm-}$ query turns this into :

$$\frac{1}{2^n}\sum_{i\in\{0,1\}^n}(-1)^{x_i}\left|i\right\rangle \tag{2.7}$$

Applying the second batch of Hadamards gives (again by Equation (23)) the final superposition

$$\frac{1}{2^n}\sum_{i\in\{0,1\}^n}(-1)^{xi}\sum_{j\in\{0,1\}^n}(-1)^{i.j}\left|j\right\rangle \tag{2.8}$$

where $i.j = \sum_{k=1}^{n} i_k j_k$ denotes the inner product of then n-bit strings $i, j \in \{0,1\}^n$. For example:

$$\frac{1}{2^n} \sum_{i \in \{0,1\}^n} (-1)^{x_i} = \begin{cases} 1 & if \quad x_i = 0 \quad for \quad all \quad i, \\ -1 & if \quad x_i = 1 \quad for \quad all \quad i, \\ 0 & if \quad x \quad is \quad balanced \end{cases} \tag{2.9}$$

2.4.2 Bernstein-Vazirani

2.4.2.1 Bernstein-Vazirani Problem

For $N = 2^n$, we are given $x \in \{0,1\}^N$ with the property that there is some unknown $a \in \{0,1\}^n$ such that $x_i = (i \cdot a) \mod 2$. The goal to find a.

The Bernstein-Vazirani algorithm is exactly the same as the Deutsch-Jozsa algorithm, but now the final observation miraculously yields a. Since $(-1)^{x_i} = (-1)^{(i \cdot a) mod 2} = (-1)^{(i \cdot a)}$,we can write the state obtained after the query as:

$$\frac{1}{\sqrt{2^n}} \sum_{i \in \{0,1\}^n} (-1)^{x_i} |i\rangle = \frac{1}{\sqrt{2^n}} \sum_{i \in \{0,1\}^n} (-1)^{i \cdot a} |i\rangle \tag{2.10}$$

Since Hadamard is its own inverse, applying a Hadamard to each qubit of the above state will turn it into the classical state jai and hence solves the Bernstein-Vazirani problem with 1 query and $O(n)$ other operations. In contrast, any classical algorithm (even a randomized one with small error probability) needs to ask n queries for information-theoretic reasons, the final answer consists of n bits and one classical query gives at most 1 bit of information.

Bernstein and Vazirani also defined a recursive version of this problem, which can be solved exactly by a quantum algorithm in poly(n) steps, but for which every classical randomized algorithm needs $n^{\Omega(\log n)}$ steps.

2.4.3 Shor's Factoring Algorithm

2.4.3.1 Factoring

Probably the most important quantum algorithm so far is Shor's factoring algorithm [46]. It can find a factor of a composite number N in roughly (logN)2 steps, which is polynomial in the length log N of the input. On the other hand, there is no known classical (deterministic or randomized) algorithm that can factor N in polynomial time. The best known classical randomized algorithms run in time roughly

$$2^{(logN)\alpha}, \tag{2.11}$$

where $\alpha = 1/3$ for a heuristic upper bound [47] and $\alpha = 1/2$ for a rigorous upper bound [48]. In fact, much of modern cryptography is based on the conjecture that no fast classical factoring algorithm exists [49]. All this cryptography (for example RSA) would be broken if Shor's algorithm could be physically realized. In terms of complexity classes, factoring (rather, the decision problem equivalent to it) is provably in BQP but is not known to be in BPP. If indeed factoring is not in BPP, then the quantum computer would be the first counter example to the "Strong" Church-Turing thesis, which states that all "reasonable" models of computation are polynomially equivalent (see [50] and [51],p.31,36).

2.4.3.2 Reduction From Factoring to Period Finding

The crucial observation of Shor was that there is an efficient quantum algorithm for the problem of period-finding and that factoring can be reduced to this, in the sense that an efficient algorithm for period-finnding implies an efficient algorithm for factoring.

We first explain the reduction. Suppose we want to find factors of the composite number $N > 1$. We may assume N is odd and not a prime power, since those cases can easily be filtered out by a classical algorithm. Now randomly choose some integer x $\in \{2, ..., N-1\}$ which is coprime to N. If x is not coprime to N, then the greatest common divisor of x and N is a nontrivial factor of N, so then we are already done. From now on consider x and N are coprime, so x is an element of the multiplicative group $\mathbb{Z}_N^*$ Consider the sequence

$$1 = x^0(\mod N), x^1(\mod N), x^2(\mod N), ...$$

This sequence will cycle after a while: there is a least $0 < r \leq N$ such that $x^r = 1\,(\mod N)$ This r is called the period of the sequence (a.k.a. the order of the element x in the group $\mathbb{Z}_N^*$).Assuming N is odd and not a prime power (those cases are easy to factor anyway), it can be shown that with probability $\geq \frac{1}{2}$ the period r is even and $x^{\left(\frac{r}{2}\right)} + 1$ and $x^{\left(\frac{r}{2}\right)} - 1$ are not multiples of N. In that case we have:

$$x^r \equiv 1 \mod N \Leftrightarrow$$

$$\left(x^{\frac{r}{2}}\right)^2 \equiv 1 \mod N \Leftrightarrow$$

$$\left(x^{\frac{r}{2}} + 1\right)\left(x^{\frac{r}{2}} - 1\right) \equiv 0 \mod N \Leftrightarrow$$

$$\left(x^{\frac{r}{2}} + 1\right)\left(x^{\frac{r}{2}} - 1\right) \equiv kN for some K.$$

Note that $k > 0$ because both $x^{\frac{r}{2}} + 1 > 0$ and $x^{\frac{r}{2}} - 1 > 0 (x+1)$.Hence $x^{\frac{r}{2}} + 1$ or $x^{\frac{r}{2}} - 1$ will share a factor with N. Because $x^{\frac{r}{2}} + 1$ and $x^{\frac{r}{2}} - 1$

are not multiples of N this factor will be $< N$, and in fact both these numbers will share a non-trivial factor with N. Accordingly, if we have r then we can compute the greatest common divisors $gcd(x^{\frac{r}{2}}+1, N)$ and $gcd(x^{\frac{r}{2}}-1, N)$, and both of these two numbers will be non-trivial factors of N. If we are unlucky we might have chosen an x that does not give a factor (which we can detect efficiently), but trying a few different random x gives a high probability of finding a factor. [52]

Thus the problem of factoring reduces to finding the period r of the function given by modular exponentiation $f(a) = x^a \mod N$ In general, the period-finding problem can be stated as follows:

The period-finding problem:

We are given some function $f : \mathbb{N} \to \{0, ..., N-1\}$ with the property that there is some unknown $r \in \{0, ..., N-1\}$ such thar $f(a) = f(b)$ if $a = b \mod r$ The goal is to find r.

We will show below how we can solve this problem efficiently, using $O(\log \log N)$ evaluations of f and $O(\log \log N)$ quantum Fourier transforms. An evaluation of f can be viewed as analogous to the application of a query in the previous algorithms. Even a some what more general kind of period-finding can be solved by Shor's algorithm with very few f-evaluations, whereas any classical bounded-error algorithm would need to evaluate the function $\Omega\left(N^{\frac{1}{3}}/\sqrt{\log N}\right)$ times in order to find the period [53].

2.4.3.3 Shor's Period Finding Algorithm

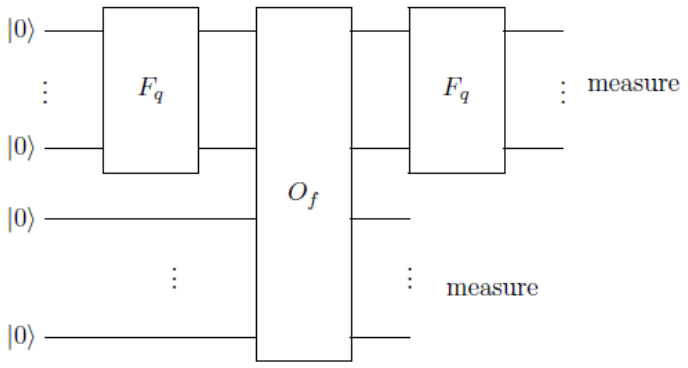

Figure 2.2: Shor's period-finding algorithm
[53]

Now we will show how Shor's algorithm finds the period r of the function f, given a "black-box" that maps $|a\rangle|0^n\rangle \mapsto |a\rangle|f(a)\rangle$.We can always efficiently pick some $q = 2^l$ such that $N^2 < q \leq 2N^2$,Then we can implement the Fourier transform F_q using $O((\log N)^2)$ gates. Let O_f denote the unitary that maps $|a\rangle|0^n\rangle \mapsto |a\rangle|f(a)\rangle$ where the first register consists of l qubits, and the second of $n = [\log N]$ qubits.

Shor's period-finding algorithm is illustrated in Figure 2.2 Start with $|0^l\rangle|0^n\rangle$. Apply the QFT (or just ' Hadamard gates) to the first register to build the uniform superposition.

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

The second register still consists of zeroes. Now use the "black--box" to compute $f(a)$ in quantum parallel:

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

Observing the second register gives some value $f(s)$, with $s<r$. Let m be the number of elements of $\{0,...,q-1\}$ that map to the observed value $f(s)$. Because $f(a)=f(s)$ iff $a=s \mod r$, the a of the form $a=jr+s\,(0\leq j<m)$ are exactly the a for which $f(a)=f(s)$. Thus the first register collapses to a superposition of $|s\rangle\,,|r+s\rangle\,,|r2+s\rangle\,,|r3+s\rangle\,,...$; this superposition runs until the last number of the form $jr+s$ that is $<q$, let's define ne m to be the number of elements in this superposition, i.e., the number of integers j such that $jr+s\in\{0,...,q-1\}$ (depending on s,this m will be $\lceil\frac{q}{r}\rceil$ or $\lfloor\frac{q}{r}\rfloor$). The second register collapses to the classical state $|f(s)\rangle$. We can now ignore the second register, and have in the first:

$$\frac{1}{\sqrt{m}}\sum_{j=0}^{m-1}|jr+s\rangle\,.$$

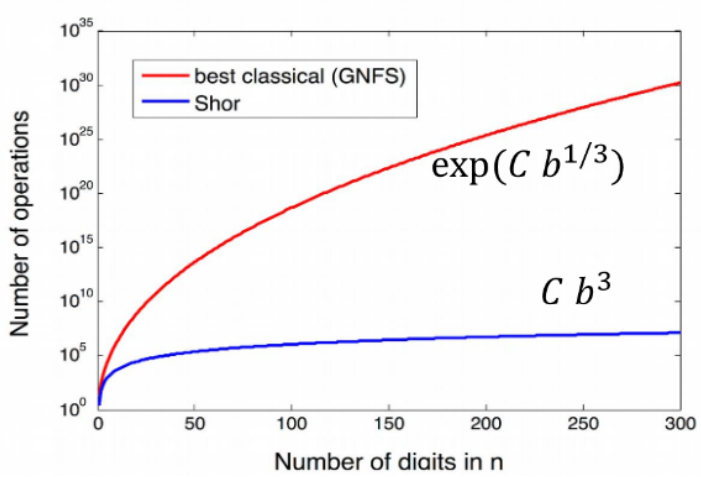

Figure 2.3: Shor's resolution run time estimation [54]

A task taking 300 years(2^{33} seconds) on classical computer might take a minute on a quantum computer.

2.4.4 Grover's Search Algorithm

The second-most important quantum algorithm after Shor's, is Grover's quantum search problem from 1996. While it doesn't provide an exponential speed-up, it is much more widely applicable than Shor's algorithm.

2.4.4.1 The Problem

For $N = 2^n$, we are given an arbitrary $x \in \{0,1\}^N$. The goal is to finnd an i such that $x_i = 1$ (and to output 'no solutions' if there are no such i).

This problem may be viewed as a simplification of the problem of searching an N-slot unordered database. Classifically, a randomized algorithm would need $O(N)$ queries to solve the search prob-

lem.Grover's algorithm solves it in $O\left(\sqrt{N}\right)$ queries, and $O\left(\sqrt{N}\log N\right)$ other gates.

2.4.4.2 Grover's Algorithm

Let $O_{\{x,i\pm\}|i\rangle=(-1)^{xi}|i\rangle}$ denote the $\pm$ type oracle for the input x, and R be the unitary transformation that puts a-1 in front all basis states $|i\rangle$ where $i \neq 0^n$,and that does nothing to the other basis states $|0^n\rangle$.The Grover iterate $\zeta = H^{\otimes n}RH^{\otimes n}O_{x,\pm}$.Note that 1 Grover iterate makes 1 query.
Grover's algorithm starts in the n-bit state $|0^n\rangle$,applies a Hadamard transformation to each qubit to get the uniform superposition $|U\rangle = \frac{1}{\sqrt{N}}\sum_i |i\rangle$of all N indices, applies ζ to this state k times, and then measures the final state. Intuitively, what happens is that in each iteration some amplitude is moved from the indices of the 0-bits to the indices of the 1-bits. The algorithm stops when almost all of the amplitude is on the 1-bits, in which case a measurement of the final state will probably give the index of a 1-bit. Figure 10 illustrates this. To analyze this, define the following "good" and "bad" states:

$$|G\rangle = \frac{1}{\sqrt{t}}\sum_{i:x_i=1} and\, |B\rangle = \frac{1}{\sqrt{N-t}}\sum_{i:x_i=1} \tag{2.12}$$

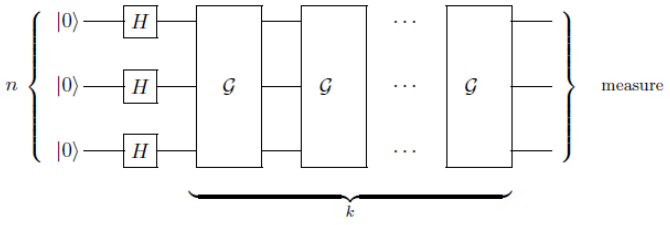

Figure 2.4: Grover's algorithm, with k Grover iterates
[6]

2.4.5 Hidden Subgroup Problem

2.4.5.1 Definition And Some Instances Of The HSP

The Hidden Subgroup Problem is the following:
Given a known group G and a function $f : G \to S$ where S is some finite set, suppose f has the property that there exists a subgroup $H \leq G$ such that f is constant within each coset, and distinct on different cosets: $f(g) = f(g')$ iff $gH = g'H$,the Goal is find H.

We assume f can be computed efficiently, meaning in time polynomial in $log\,|G|$ (the latter is the number of bits needed to describe an input $g \in G$ for f). Since H may be large, "finding H" typically means finding a generating set for H.
This looks like a rather abstract algebraic problem, but many important problems can be written as an instance of the HSP. We will start with some examples where G is Abelian.

Simon's problem. This is a very natural instance of HSP. Here G is the additive group $\mathbb{Z}_2^n = \{0,1\}^n$ of size 2^n, $H = \{0, s\}$ for a "hidden" $s \in \{0,1\}^n$,and f satisfies $f(x) = f(y)$ iff$x - y$ 2 $\in H$.
Clearly,finding the generator of H (i.e., finding s) solves Simon's problem. [6]
Discrete logarithm. Another problem often used in classical public-key cryptography is the discrete logarithm problem: given a generator γ of a cyclic multiplicative group C of size N(so $C = \{\gamma^a | a \in \{0, ..., N-1\}\}$),and $A \in C$, can we find the unique a $\in \{0, 1, ..., N-1\}$ such that $\gamma^a = A$?
This a is called the discrete logarithm of A (w.r.t. generator γ). [55]
It is generally believed that classical computers need time roughly exponential in log N to compute a from A (and one can actually prove this in a model where we can only implement group opera-

tions via some "blackbox"). This assumption underlies for instance the security of Diffie-Hellman key exchange(where $C = \mathbb{Z}_p^*$ for some large prime p),as well as elliptic curve cryptography.
Discrete log is an instance of the HSP as follows:
we take $G = \mathbb{Z}_N \times \mathbb{Z}_N$ and define function $f : G \to C$ by $f(x, y) = \gamma^x A^{-y}$, which is effciently computable by repeated squaring. For group elements $g_1 = (x1; y1); g2 = (x2; y2) \in G$ we have:
$f(g1) = f(g2) \Leftrightarrow \gamma^{x1-ay1} = \gamma^{x2-ay2} \Leftrightarrow (x1 - x2) = a\,(y1 - y2) \mod N \Leftrightarrow g1 - g2 \in < (a, 1) >$.

Let H be the subgroup of G generated by the element $(a; 1)$, then we have an instance of the HSP.
Finding the generator of the hidden subgroup H gives us a, solving the discrete logarithm problem. [6]

2.5 Conclusion

Many of other interesting quantum algorithms, for example quantum simulated annealing or quantum Bayesian networks needs some understanding of the underlying math. The study of quantum algorithms will soon be common place. Until then, this work at least demonstrates that simple quantum algorithms are not beyond the understanding of the average undergraduate computer science student, providing a gentle introduction to the basics of quantum computation to the undergraduate population.

Chapter 3

Quantum Algorithms Implementation

3.1 Introduction

Quantum computers will be available to the public so building a community of quantum programmers is important. In this chapter we will see some quantum algorithms and the implementation on IBM's quantum computer and simulator, and in each case, we discuss the results of the implementation with respect to differences between the simulator and the actual hardware runs, we used the IBM Q Experience because it's easy to compose quantum algorithms circuits by using it's graphical interface circuit composer(drag and drop) and available for public with powerful machines to test on.

3.2 Quantum Computing Known Simulators

Quantum simulators are controllable quantum systems used to simulate other quantum systems, and can solve problems impossible for classical computers, we will see some known existen simulators

such QX, Q-Kit and IBM Q.

3.2.1 QX Simulator

The QX Simulator is a universal quantum computer simulator developped at QuTech by Nader Khammassi. The QX allows quantum algorithm designers to simulate the execution of their quantum circuits on a quantum computer. [56] The simulator defines the Quantum Code(a low level quantum assembly language), which allows the users to describe their circuits in a simple textual source code file. The source code file is then used as the input of the simulator which executes its content.
The Quantum Code language allows the users to [57] :

1. Defining quantum register with a given qubits number.
2. Building the circuit through a sequence of quantum gates.
3. Simulating the calssical-quantum interface through binary-controlled gates.
4. Splitting the main circuit into several smaller sub-circuit.
5. ...

3.2.2 Q-Kit

Figure 3.1: Quantum-Kit Graphical User-Interface.Panels for (1) graphical quantum circuit builder, (2) quantum circuit command script, and (3) simulation execution status.(b) Visualization of quantum state after each quantum gate operation in the circuit.Single qubit states shown on a Bloch Sphere and multi-qubit gates on bar charts for probability distributions and amplitudes. [58]

Q-Kit is a graphical quantum circuit composer, no efforts needed to construct quantum circuits, analyzing them and visualizing the quantum states after every quantum operation. While there are several existing software freely available for these quantum simulations, they are either focused on learning or on large scale simulation capabilities. The former kind are equipped witha user-friendly interface-specifically set-up for learning on fewer qubits, while the latter require a knowledge of quantum programming languages to truly exploit the potential of these softwareon super computing clusters. [58] (Figure 3.1)

3.2.2.1 Q-Kit Feauters

Q-Kit has many features including the capability to copy data from measured qubits to classical bits and using classical controls for quantum operations. This feature of using classical bits and controls in a quantum circuit, specially in certain algorithms, provide exponential advantages in simulations, both in speed and memory requirements, as discussed in the next section. Q-Kit can be run both in performance-enhanced or memory-enhanced mode and is demonstrated to be extremely efficiental though it currently runs only on a single core. A parallelized Q-Kit is expected to enable much more powerful computations, with far fewer resources. [58]

3.2.3 IBM's Q Experience

IBM is one of the major companies that are investing massively to understand and develop quantum technologies, IBM currently has three quantum computers (one 5-qubit and two 16-qubit devices) available for public use over the cloud. They also have a publicly usable quantum simulator (32-qubit), which allows users to simulate quantum algorithms without any error.

3.2.3.1 IBM QX: Web Interface

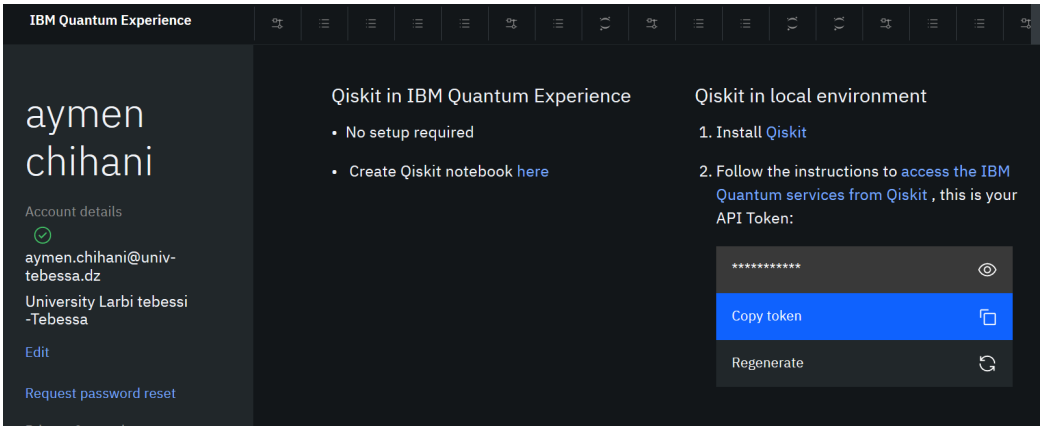

Figure 3.2: IBM QX Web interface

3.2.3.2 Graphical composer

1. Compose quantum circuits using drag and drop interface.
2. Save circuits online or as QASM text, and import later.
3. Run circuits on real hardware and simulator.

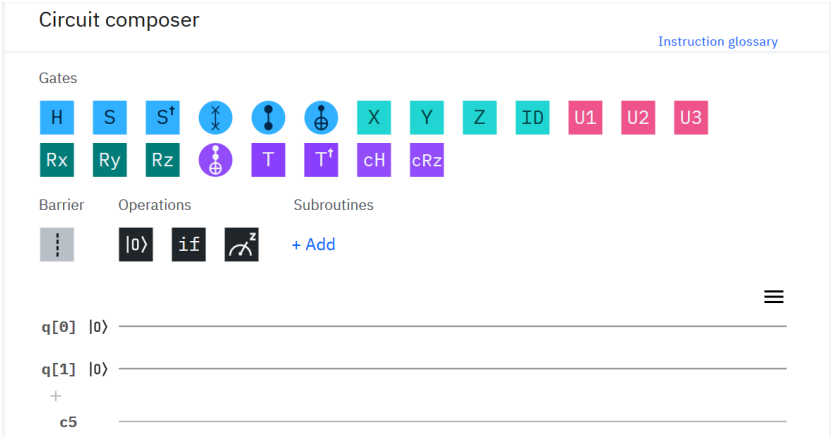

Figure 3.3: IBM's Q Experience Circuite composer(Drag and Drop)

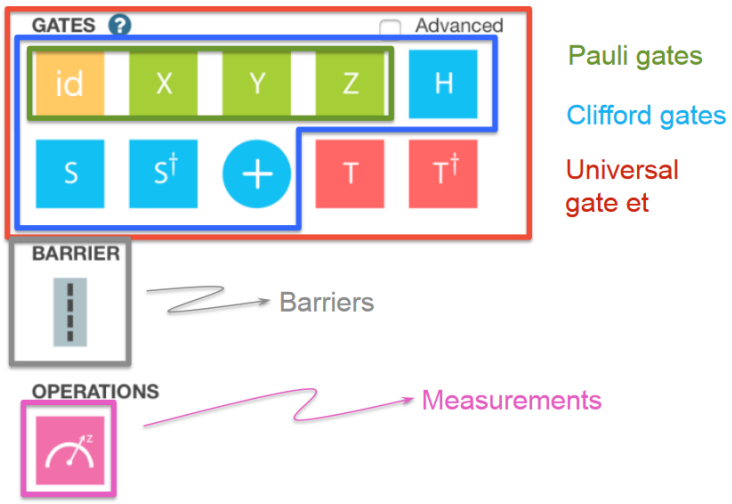

Figure 3.4: Basic Operation

The symbols on the circuit considred as quantum gates. The H stand for Hadamard, the + stand for a CNOT , X would be a Pauli X, and the S is called a Phase gate, the S gate is an extension of the Hadamard gate to allow complex superposition of each qubit's axes which IBM called a Phase gate.

3.3 Grover's Algorithm On IBM's Q Experience:

I used Circuite composer provided by IBM's Q Experience I was able to run a 2-bit Grover's algorithm on their quantum computer in addition to simulating the result. Below are the results of the experiment, it was ran on the IBM Quantum Machine ibmqx2(IBM 5 qubit Real Processer) 1024 times each;

3.3.1 Simulation

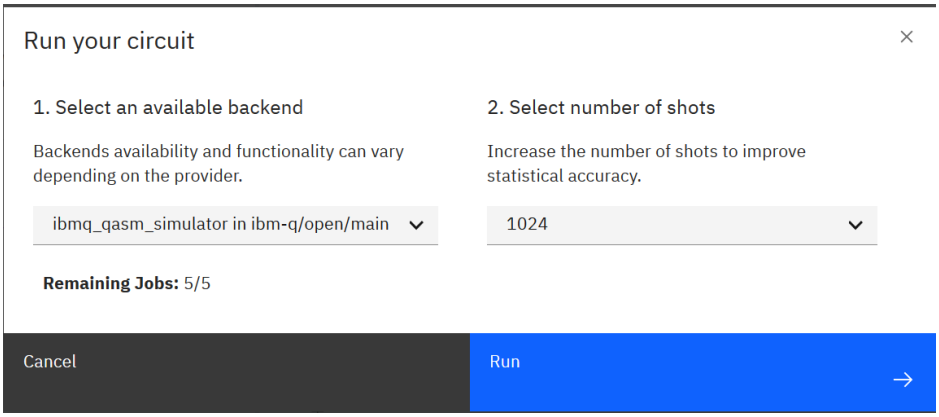

Figure 3.5: IBM Simulator

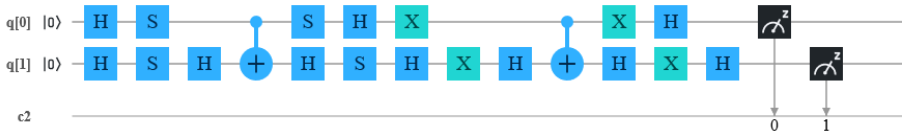

Figure 3.6: 2-bit Grover's Circuit

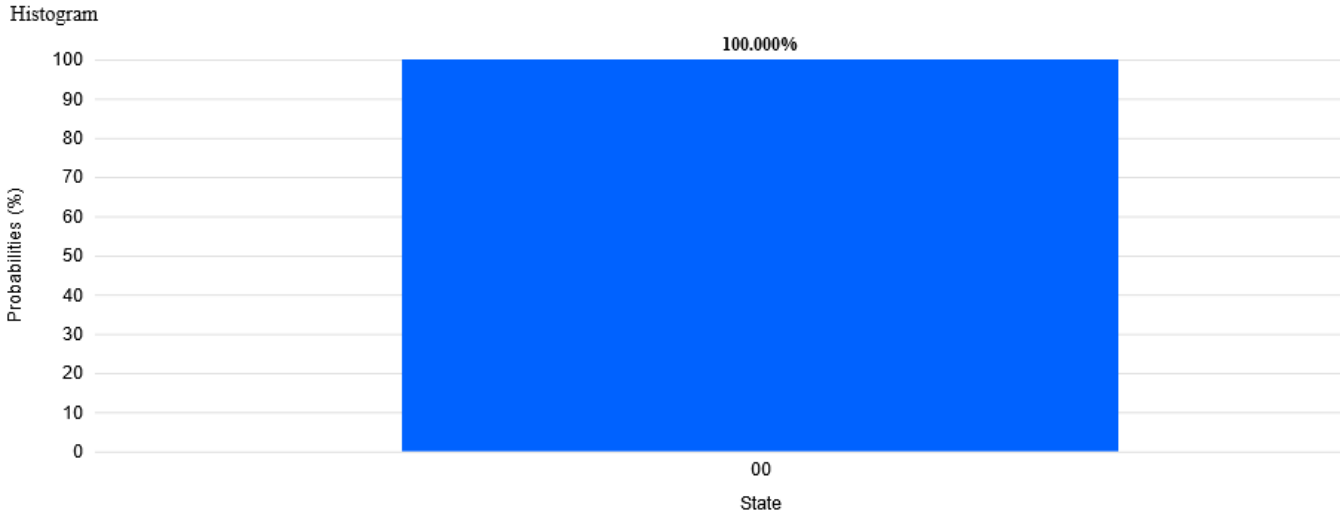

Figure 3.7: Histogram Of IBM Q EXPERIENCE X=00 SIMULATION

The simulator results of IBM Q for value 00 was 100% **probability**

3.3.2 Experiment on 5-Qubits Machine(ibmqx2)

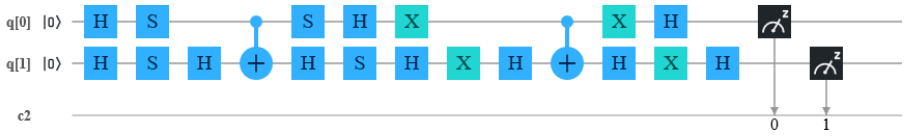

Figure 3.8: 2-bit Grover's Original Circuit On Real Device ibmqx2

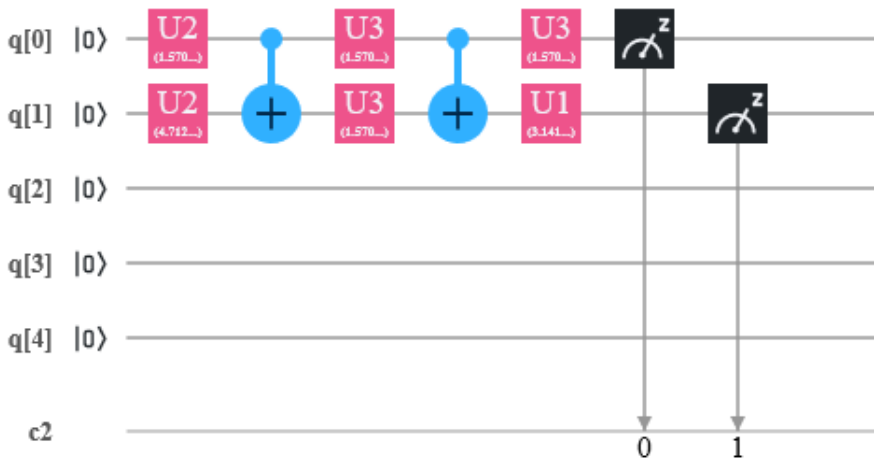

Figure 3.9: 2-bit Grover's Transpiled Circuit

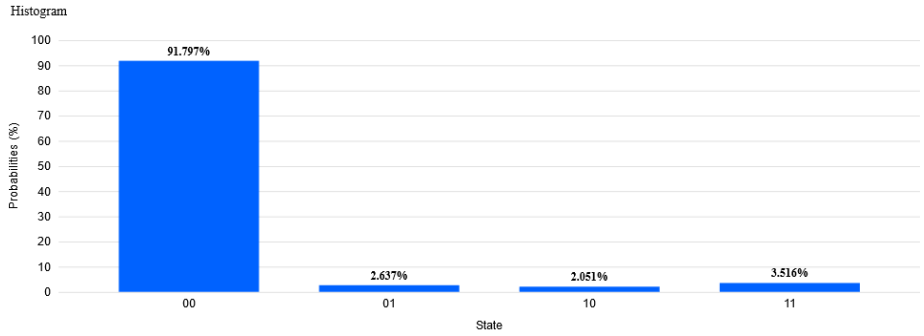

Figure 3.10: IBM Q EXPERIENCE X=00 5-QUBIT MACHINE RUN(ibmqx2)

3.3.3 Results

The results i found from the actual run on the quantum machine and the simulators was as i expected. Error correction for quantum computers is almost absent and not advanced as conventional computers. Because of hardware and environmental factors the results from this test seems to be highly prone to errors, similar

to early classical computers where error correction had not been implemented. The simulator results of IBM Q for value was 100% **probability.**

3.4 Shor's algorithm for integer factorization

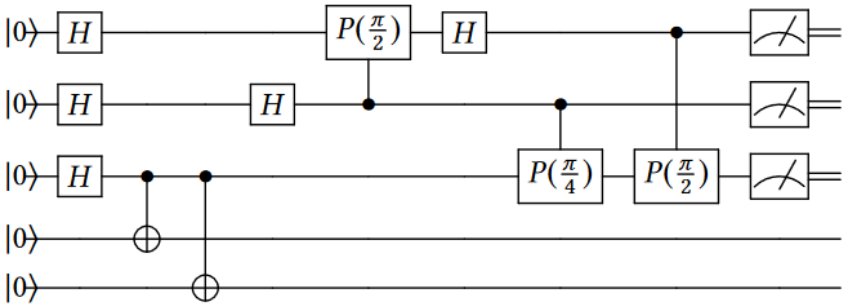

Figure 3.11: Circuit for Shor's algorithm for N=15 and x=11 [59]

We implemented the algorithm on ibmqx2, a 5-qubit quantum processor from the IBM Quantum Experience, in order to factor number 15 **with** $x = 11$**, i used a optimized/compiled version from [59] that uses 5 qubit and 11 gates (Fig 3.10).**

3.4.1 Sho'r Experiment

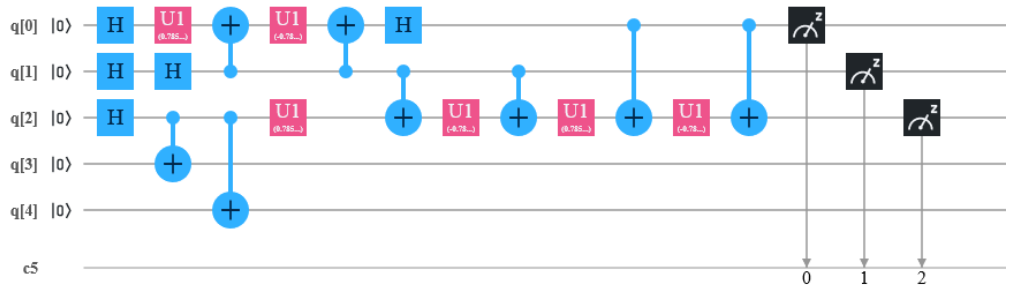

Figure 3.12: Shor's transpiled circuit

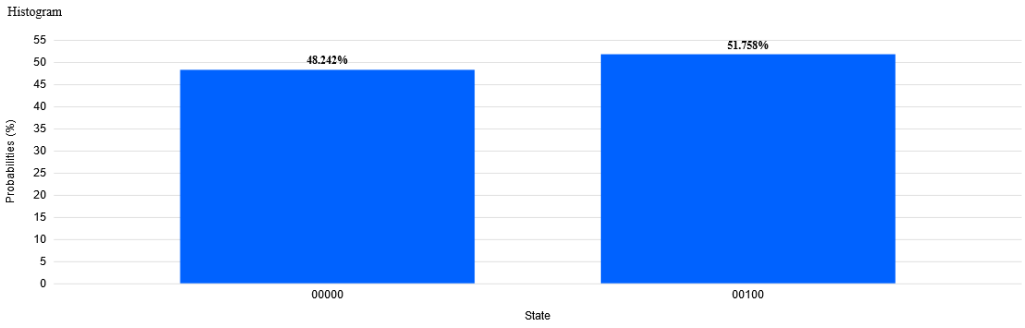

Figure 3.13: Shor's simulation histogram

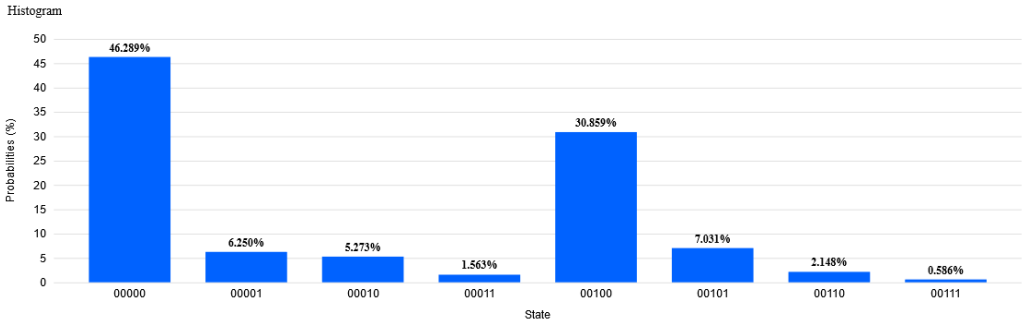

Figure 3.14: Shor's circuit on ibmqx2 histogram

3.4.2 Results

The output from the simulation finds the periods 0 and 4 with the highest probabilities (Fig 3.12), the same in ibmqx2 but contains much more noise(Fig 3.13).
The periods found by the simulator are $p = 0$, we can ignore it because it's a trivial period, and $p = 4$, which is a good one. Since M=8, we can conclude that r divides $M/p = 8/4 = 2$ (M is the length vector of x input), hence $r = 2$(r is the period), Then 15 divides $(x^r - 1) = (11^2 - 1) = (11 - 1)(11 + 1) = 10 \cdot 12$.
By computing $gcd(15, 10) = 5$ and $gcd(15, 12) = 3$, we find the factors of 15.

3.5 Conclusion

We found amazing simulation results for Grover's algorithm experimentation on IBM Q, I got a 100% probability of finding the correct value when the quantum system is stable. However, when the quantum system is not stable, we found that the results would decrease the probability of Grover's algorithm.
For shor's algorithme experimentation on simulation we found the periods 0 and 4 with the heighest probabilities, in ibmqx4 we find the same heighest periods but with other trivial periods due to machine errors and noise.
quantum computation is the computing revolution, as classical computer is reaching its limit, we reviewed the basics of quantum algorithms and went into the analysis of Grover's algorithm and shor's algorithm, The results show that Quantum algorithms and classical algorithms are quite similar to each other but complete different at the same time. soon quantum algorithms may possibly pass the current standards. With more companies like IBM fund-

ing research in to the field of quantum computing and tools such as IBM Q experience, quantum computing is not far from reality and the time of classical algorithms may be coming to an end.

Chapter 4

Analysis and Forecasts

4.1 Introduction

The quantum programming environment have many requirements. First, it needs a high level quantum programming language to execute quantum operations. quantum unitary transforms, complex numbers, gate libraries support. In addition, the environment and the programming language should be based on familiar concepts and constructs to make writing, debuging, and runing a quantum program much easier than using a totally different environment. The quantum programming environment also needs to allow easy separation of classical and quantum computations. Because a quantum computer has noise and limited coherence time, this separation necessary to limit computation time, quantum programming language compiler must translate a source code into an robust and useful quantum circuit or physical implementation. [60]

4.2 A Comparison Of Classical and Quantum Programming Tools

4.2.1 Quantum Computer Compiler

A generic compiler for a classical language on a classical machine consists of a sequence of phases that transform the source program from one representation into another.This partitioning of the compilation process has led to the development of efficient algorithms and tools for each phase. Because the front end processes for QCCs are similar to those of classical compilers, researchers can use the algorithms and tools to build lexical, syntactic, and semantic analyzers for QCCs. How ever, the intermediate representations, theo ptimization phase, and the code generation phase of QCCs differ greatly from classical compilers and require novel approaches, such as a way to insert error correction operations into the target language program. [60]

4.2.2 Error Correction

In experimental realizations of quantum computers, dealing with errors in the operations is so important, and we must find a way to protect the computation from such errors. [61]
Error correction in classical computers is essentially based on two facts:

a Computing with classical bits itself provides a simple way of error correction in the form of a lock in place mechanism. If the two bits are for instance realized by two different voltages (like it is the case in our computers as well as in our brains) then the difference can simply be chosen large enough such that typical fluctuations are small compared to the threshold

separating the two bits.

b The information can be copied and then stored or processed in a redundant way. If, for instance, an error occurs to one of three copies of a bit, we can recover the original information by applying a majority vote. Of course, there are much more refined versions of this method.

Unfortunately, in quantum computers we can not use either of these ideas :

a There is no lock-in-place mechanism

b Cant't copy the state of the qubits because of the no-cloning theorem.

Measuring the system state to find out what error has actually happened before correcting it does not help, as any such attempt would necessarily disturb the state in an irreversible manner. So it was at the very beginning of quantum information science not clear whether or not under physically reasonable assumptions fault tolerant quantum computing would be possible. [61]

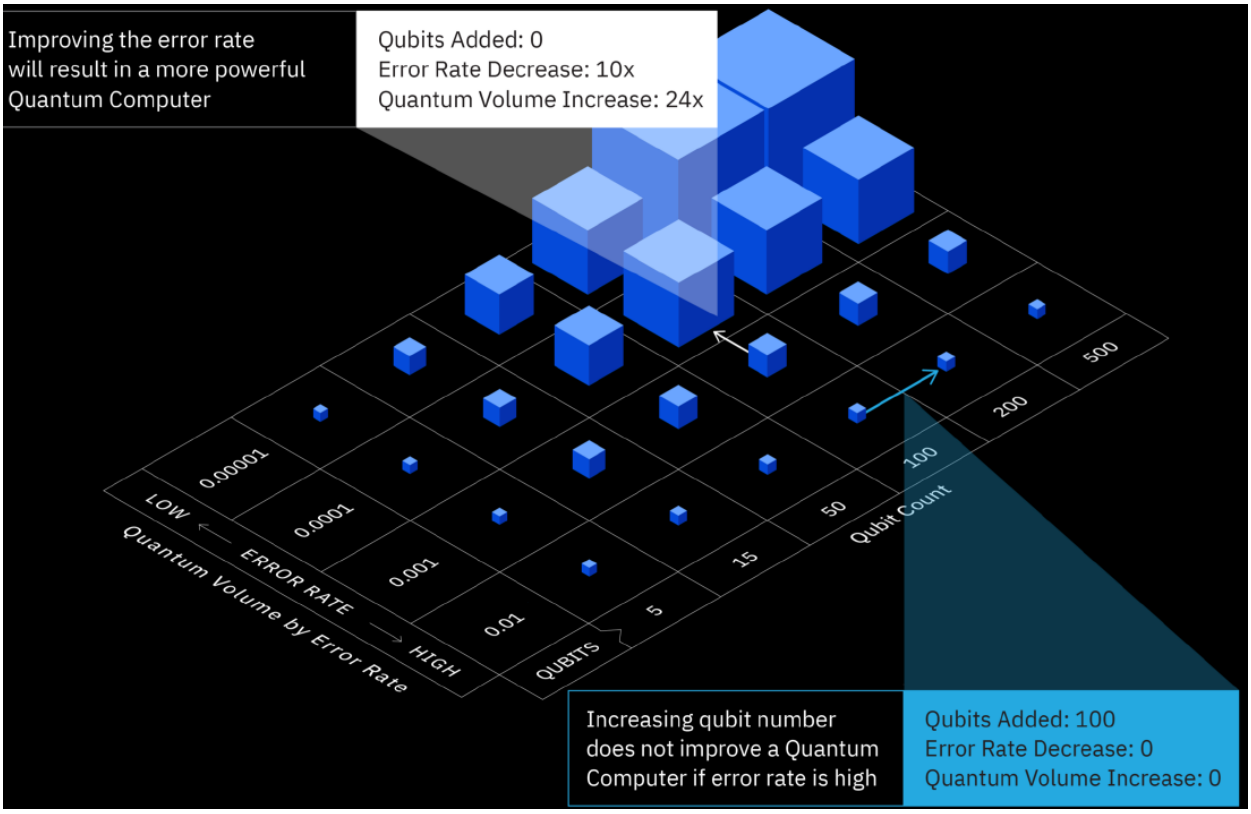

Figure 4.1: The effect of increasing qubits compared to error rate.
source IBM research
[62]

Increasing qubit number will not necessary Increase the computation power unless we decreased error rate as shown in (Figure 4.1), we see that how much we added qubits the quantum computer power volume is fixed with a high error rate but when we decrease error rate the results is different in eache case.

4.2.3 Software Engineering

In classic software engineering, a test case is a specification of the inputs, execution conditions, testing procedure, and expected re-

sults that define a single test to be executed to achieve a particular software testing objective. One of the desired characteristics of test cases is their *repeatability*: this is, the verdict of a test must be always the same. The "verdict" is the result of the test execution, usually *Pass* (if the test has not found any error in the system under test, the SUT) or *Fail* (if the system has shown a behavior different than the expected one). So, a test case has three parts: (1) specification of the initial situation, (2) execution of operations on the SUT and (3) an oracle, which compares the actual and expected outputs, so determining the verdict.
executing a test case on a quantum computer actually requires executing the same test case a number of times then compare the most repeated output to the expected output, as well as this adaptation is required, quantum software testing requires both the adaptation of other techniques, as well as the creation of new ones. [63]

4.3 Quantum Computers Complexity

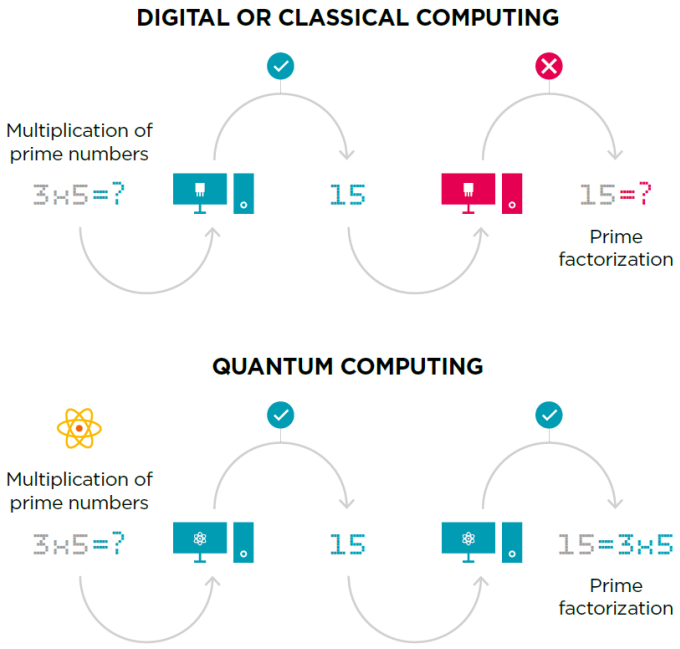

Figure 4.2: the prime factorization problem, the basis of many current encryption protocols. All the computational capacity in the entire world could not factor extremely large numbers within a reasonable amount of time. [64]

Quantum computers can solve certain types of problems faster than any classical computer, the boundary between easy and hard is different for quantum computers than classical computers. that means the time for solving the problem grows polynomially with the length of the input data like the problem of multiplying two numbers, where hard problems are those which the required time grows exponentially Prominent like the problem of factoring a number into primes. (Figure 4.2)

With any algorithm, we want to limit used resources(running time

and number of queries), the Grover's Algorithm in a perfect environment for a quantum computer would run in $O\sqrt{n}$ time and take $O\sqrt{n}$ operations, which is a quadratic speed up compared to the fastest search over an unordered data set in a classical algorithm which is $O(n)$.
Grover's Algorithm relies on the concept of quantum superposition of states to improve upon the computing/runtime of classical computer. [61]

4.4 Analysis

Search algorithms like Grover's database search and integer factoring algorithms Like Shor's algorithm are very promising. they distinguished with an exponential speed up compared to the best known classical algorithms. [61]
If we consider a quantum computer unitarily acting on a pure input state, then an exponential speed-up compared to conventional computers can only be achieved if the entanglement present in intermediate states of the computation increases with size of the input, it apears that computations based on such quantum evolutions can in general not be simulated efficiently on classical computers.

4.5 Conclusion

Quantum computing opens doors to solve some real problems which are simply impossible using conventional computers, this makes a threat and at the same time enormous opportunities. First, it threatens data authentication, having the greatest impact on areas where cryptography plays a fundamental role, like cyber security and blockchain.
quantum computing will be a significant threat for them as well.

Second, the computational capacity it offers will be essential in speeding up the development of many emerging technologies, especially those related to artificial intelligence. It will also be remarkably beneficial in the fields of medicine, biology, and genetics where, for example, quantum technologies will allow for the simulation of the effects of clinical drugs, exponentially reducing time and resources. More efficient research can be done in the quest to find cures for cancer and other diseases, including Alzheimer's, Parkinson's, and multiple sclerosis, among many others. In the field of finance, it will be possible to create much more precise mathematical models and to process data in real time in a more efficient manner for decision making. In energy and sustainable agriculture, it will be possible to explore new techniques for ammonia production at a lower energy and economic cost, replacing the current process that consumes 2% of the world's energy and drives up the cost of food.

General Conclusion

Today's computing machines uses classical physics laws to understand and represent the logical processing of information.
These classical descriptions of Newtonian physics provide an intuitive explanation of the physical universe, but they can't predict all observable phenomena. This awareness is the reason of the most important revolution in physics: the discovery of quantum mechanics laws.
In recent years we have seen amazing computing innovations. However, the architecture behind them has not changed (based on binary calculation 1's and 0's), With quantum computing, we can think of amazing possibilities in terms of power, speed and even more than that: a new way of solving problems considered impossible for classical computers.
Quantum computing is at its earliest stages of development and still has a long way to go compared to its much matured classical counterpart.
Throughout this work, we have discussed the concept, benefits, potential applications of quantum computers and some known quantum algorithms and their implemantation and we can say that quantum computing is the next industrial revolution, however there is technical barriers remain before a practical quantum computer can be fully achieved.
Different error sources due to the interactions between qubits and

the environment which add noise to qubits in turn affecting the robustness of gate operation. Current state of the art gate based quantum computer contains 20 qubits with gate error rate of about 5%. It is estimated that for a quantum computer to perform some task impossible for classical computer we need over 50 qubits with less than 0.1% error rate.

The challenges to create a large, error corrected quantum computer is important. unprecedented control of quantum coherence and improving existing tools and techniques or even by developing new ones is required for Successful quantum computation.

Bibliography

[1] Florent NOURRISSON NITSCH. The fundamentals of quantum computing, 2019.

[2] Sophie Laplante et Frédéric Magniez Julia Kempe. *L'ordinateur quantique, La Recherche, no 398,.* 2006.

[3] Dieter Suter Joachim Stolze. *Quantum Computing: A Short Course from Theory to Experiment.* 2004.

[4] Gordon E. Moore. *Cramming more components onto integrated circuits. Electronics,.* 1965.

[5] Timothy Jones. Is quantum decoherence the von neumann wave collapse?, 2007.

[6] Ronald de Wolf. *Quantum Computing.* 2012.

[7] Edgard Elbaz. *Quantique Ellipses/edition marketing S.A.* 1995.

[8] Mark Saffman. *Dirac Notation and rules of Quantum Mechanics.* 2006.

[9] Hayden Patrick Ekert, Artur and Hitoshi Inamori. *Coherent atomic matter waves.* 2001.

[10] John Preskill. *Quantum Information And Computation.* 1998.

[11] Vlatko Vedral and Martin B. Plenio. *Basics Of Quantum Computation.* 2002.

[12] Merciadri Luca. *Quantum Computers: A Brief Overview.* 2009.

[13] Quantum how do quantum computers work? `https://www.sciencealert.com/quantum-computers`. Accessed: 2020-4-14.

[14] Quantum vs classical computation. `http://www.thphys.nuim.ie/staff/joost/TQM/QvC.html`. Accessed: 2020-4-18.

[15] Jonathan Marshall. *,Simulating Quantum Circuits.* 2009.

[16] Neil Gershenfeld and Isaac L. Chuang. Quantum computing with molecules. *Scientific American*, 1998.

[17] Florent NOURRISSON NITSCH. The fundamentals of quantum computing. 2018.

[18] Dominique Unruh. Quantum programming languages, 2006.

[19] E. Knill. *Conventions for quantum pseudocode, Los Alamos National Laboratory technical report, LAUR-96-2724,.* 1996.

[20] B. Ömer. A procedural formalism for quantum computing„ 1998.

[21] Jarosław Adam Miszczak. Introduction to models of quantum computation and quantum programming languages. 2010.

[22] S. Bettelli. *Toward an architecture for quantum programming.* PhD thesis, 2003.

[23] P. Zuliani. *Quantum programming with mixed states, Proceedings of the 3rd International Workshop on Quantum Programming Languages, pages 169-179.* PhD thesis, Princeton University, 2005.

[24] Jonathan Grattage. *QML: A Functional Quantum Programming Language.*

[25] Seth Lloyd. *Confidentialité et Internet quantique, Pour la science, vol. 391, mai 2010, p. 60–65.* 2010.

[26] IronBridge commercially-ready certifiable quantum cryptographic device. `https://cambridgequantum.com/cqc-unveils-the-worlds-first-commercially-ready-certifiable-quant` Accessed: 2020-5-10.

[27] National Academies of Sciences Engineering and Medicine. *Quantum Computing: Progress and Prospects.* 2019.

[28] M. Day* S. Frick J. Hinchliff M. Johnson S. Morley-Short S. Pallister A. B. Price S. Stanisic J. C. Adcock, E. Allen. *Advances in quantum machine learning.* 2015.

[29] Seth Lloyd. *Confidentialité et Internet quantique », Pour la science, vol. 391, p. 60–65.* 2010.

[30] [ibmqx backend information. `https://github.com/qiskit/ibmqx-backend-information`. Accessed: 2020-4-14.

[31] Robert Wille Alwin Zulehner, Alexandru Paler. An efficient methodology for mappingquantum circuits to the ibm qx architectures. 2018.

[32] Ibmqx backend information. https://github.com/QISKit/ibmqx-backend-information. Accessed: 2020-4-14.

[33] VERA BLOMKVIST KARLSSON PHILIP STRÖMBERG. 4-qubit grover's algorithmimplemented for the ibmqx5architecture. 2018.

[34] VERA BLOMKVIST KARLSSON PHILIP STRÖMBERG. 4-qubit grover's algorithmimplemented for the ibmqx5architecture, 2018.

[35] D-wave 2000q quantum computer. https://www.dwavesys.com. Accessed: 2020-4-14.

[36] D-wave 2000q specification. https://www.linkedin.com/pulse/quantum-computers-more-secrets-michael-mather. Accessed: 2020-4-14.

[37] D-wave 2000q working. https://aws.amazon.com/braket/hardware-providers. Accessed: 2020-4-14.

[38] D-wave 2000q software. https://www.dwavesys.com/software. Accessed: 2020-4-14.

[39] How do you write a simple program for a d-wave device? https://quantumcomputing.stackexchange.com/questions/1451/how-do-you-write-a-simple-program-for-a-d-wave-device. Accessed: 2020-4-14.

[40] D-wave 2000q system. https://dwavefederal.com/system. Accessed: 2020-4-14.

[41] Ingolf Wittmann. An-60-min-introduction-into-ibm-q-strategy-and-offering-v1-1, 2018.

[42] D. Deutsch. *Quantum theory, the Church-Turing principle, and the universal quantum Turing machine. InProceedings of the Royal Society of London, volume A400, pages 97–117.* 1985.

[43] D. Deutsch. *Quantum computational networks. InProceedings of the Royal Society of London,volume A425.* 1989.

[44] U.V. Vazirani S. Dasgupta, C.H.Papadimitriou. *algorithms dasgupta c h papadimitriou and u v vazirani solution manual.* 2019.

[45] D. Deutsch and R. Jozsa. *Rapid solution of problems by quantum computation. InProceedings of the Royal Society of London, volume A439, pages 553–558.* 1992.

[46] U. Schoning. *A probabilistic algorithm for k-SAT and constraint satisfaction problems. In Proceedings of 40th IEEE FOCS, pages 410.* 1999.

[47] A. K. Lenstra and Jr H. W. Lenstra. *The Development of the Number Field Sieve, volume 1554 of Lecture Notes in Mathematics. Springer.* 1993.

[48] Jr. H. W. Lenstra and C. Pomerance. *A rigorous time bound for factoring integers. Journal of the American Mathematical Society, 5:483.* 1992.

[49] R. L. Rivest. *Cryptography. In van Leeuwen pages 717-755.*

[50] P. van Emde Boas. *Machine models and simulations.*

[51] C. H. Papadimitriou. *Computational Complexity. Addison-Wesley*. 1994.

[52] D.P. DiVincenzo and IBM. The physical implementation of quantum computation. 2000.

[53] R. Cleve. The query complexity of order-nding. in proceedings of 15th ieee conference on computational complexity, pages 54:59. 2000.

[54] Ingolf Wittmann. Ibm q. In *AN 60 minute introduction into IBM Q Strategy and offering*, 2018.

[55] Ronald de Wolf. *Quantum Computing Lecture Notes, Extra Chapter*. 2012.

[56] The qx simulator. `http://quantum-studio.net/`. Accessed: 2020-4-14.

[57] Quantum programming tools. `https://quantumcomputingreport.com/tools/`. Accessed: 2020-4-14.

[58] Hesameddin Ilatikhameneh Archana Tankasala. Quantumkit:simulatingshor's factorization of 24-bit number on desktop.

[59] G. Breyta C. S. Yannoni M. H. Sherwood L. M. K. Vandersypen, M. Steffen and I. L. Chuang. Experimental realizationof shor's quantum factoring algorithm using nuclear magnetic resonance 414:883–887,. *Nature*, December 2001.

[60] Andrew W. Cross Isaac Chuangn Igor L. Markov Krysta M. Svore, Alfred V. Aho. A layered softwarearchitecture for quantumcomputing design tools, 2006.

[61] Albert Y. Zomaya. *Handbook of Nature-Inspired and Innovative Computing: Integrating Classical models with the emerging technologies*. 2006.

[62] ibm. Ibm q. https://www.research.ibm.com/ibm-q/, 2017.

[63] Macario Polo Usaola. Quantum software testing. 2020.

[64] Marcos Allende López. *QUANTUM TECHNOLOGIES Digital transformation, social impact, and crosssector disruption.* 2019.