Departamento de computación
Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN ENXEÑERÍA INFORMÁTICA
MENCIÓN EN COMPUTACIÓN

Quantum modeling of uncertainty in classical rule-based systems

Estudante: Martín Sande Costa
Director/a/es/as: Vicente Moret-Bonillo

A Coruña, September 5, 2019.

A Xoel

Acknowledgements

This work has received financial support from the Xunta de Galicia (Centro singular de investigación de Galicia accreditation 2016-2019) and the European Union (European Regional Development Fund - ERDF) is gratefully acknowledged.

Agradecementos

Este traballo recibiu apoio financeiro da Xunta de Galicia (acreditación do Centro singular de investigación de Galicia 2016-2019) e agradece á Unión Europea (Fondo Europeo de Desenvolvemento Rexional - FEDER).

Gracias a todos os meus amigos e familia. Sen eles tería acabado este proxecto meses antes.

Gracias tamén a Vicente pola súa inestimable axuda.

Manifesto do autor

Eu, D. MARTÍN SANDE COSTA, presento o meu Traballo Fin de Grao, MODELAXE CUÁNTICA DA INCERTEZA EN SISTEMAS CLÁSICOS BASEADOS EN REGRAS seguindo o procedemento axeitado ó Reglamento, e DECLARO que:

- Onde teño consultado o traballo publicado doutras persoas, isto sempre se indica claramente.
- Sempre se dá a fonte onde citei o traballo doutros. Con excepción de citas, este Traballo de Fin de Grao é da miña propia elaboración.
- Recoñecín todas as fontes principais de axuda.
- Onde o Traballo Fin de Grao está baseado no traballo feito por outros, deixei claro o que fixeron os demais e o que aportei.

Abstract

Uncertainty is one of the cardinal obstacles when working with artificial intelligence — i.e., the managed information may be incomplete, incorrect or imprecise. It is particularly one of rule-based systems' (RBS) most essential and convoluted issues.
Albeit statistics has been historically the leading formalism to represent ambiguity, there exist alternative numerical methods to quantify it (e.g., fuzzy sets or belief functions [1]).
The associated uncertainty of a hypothesis can be considered a consequence of the propagation of imprecision through the different inferential logics of a system. Considering this, is it possible to model ambiguity?
A solution to this problem is proposed by [2]. The main goal of this work is to assemble and evaluate thoroughly a QRBS [3] that works analogously to its conventional predecessor. This new system treats uncertainty as an innate aftereffect of the inherent probabilistic nature of QM. It is a work of AI that uses QC techniques to solve the problem of uncertainty in RBSs.

Resumo

A incerteza é un dos obstáculos cardinais cando se traballa con intelixencia artificial - é dicir, a información xestionada pode ser incompleta, incorrecta ou imprecisa. É particularmente un dos temas máis esenciais e relacionados cos sistemas baseados en regras.
Aínda que a estatística foi historicamente o principal formalismo para representar a ambigüidade, existen métodos numéricos alternativos para cuantificalo (por exemplo, conxuntos difusos ou funcións de crenza [1]).
A incerteza asociada a unha hipótese pódese considerar consecuencia da propagación da imprecisión a través das distintas lóxicas inferenciais dun sistema. Tendo en conta isto, é posible modelar a ambigüidade?
[2] propón unha solución a este problema. O obxectivo principal deste traballo é reunir e avaliar a fondo un sistema cuántico baseado en regras [3] que funciona de xeito análogo ao seu predecesor convencional. Este novo sistema trata a incerteza como un efecto positivo do carácter probabilístico inherente da mecánica cuántica. Trátase dun traballo de AI que usa técnicas de computación cuántica para resolver o problema de incerteza nos sistemas baseados en regras.

Palabras chave:

Computación cuántica

Sistemas baseados en regras

Intelixencia artificial

Ambigüidade

Sistemas cuánticos baseados en regras

Keywords:

Quantum computing

Rule-based systems

Artificial intelligence

Uncertainty

Quantum rule-based systems

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Introduction sections of this study is a presentation aiming at bringing in itself to the readers and presenting the subsequent sections of the work. Firstly, it provides information about the general topic of the work which paves the way for the disclosure of the objectives as well as the methodology followed to achieve them. Finally, the last paragraphs analyze the encountered problems that lead to this study's necessity in the light of the current literature.

1.1 Objectives

The first two objectives of the project will be the basis for the following phases and will be to develop a classic rule-based system and its quantum counterpart. The rest of iterations will be built on this first approach by adding new features or using it as an analogy to come up with new solutions. In this way, the third and fourth phases will solve the first two problems in a categorical and quantum manner respectively. Once we have these two resolutions we can assign an uncertainty to each input variable and solve the problem with a quantum method with this uncertainty. In short, the specific objectives of the project can be summarized in:

- Develop a classic rule-based system and its quantum counterpart.
- Solve the classic problem categorically.
- Solve the proposed problem in a quantum way.
- Assign uncertainty to input variables.
- Solve the problem in a quantum way with this uncertainty.

1.2 Motivation

There are three essential questions that justify quantum approaches in computer sciences. This section dives into this issues and their relation with the study.

1.2.1 Current microprocessor size & Moore's Law consequences

The primary driving force of economic growth is the growth of productivity and the rate of financial realities is controlled by the rate of technological progress. A 2011 study [6] in the journal *Science* showed that every new year allowed humans to carry out roughly 60% more computation than possibly could have been executed by all existing general-purpose computers in the year before. This computational power increase is mainly due to two big factors: Moore's Law and Pollock's Rule.

Moore's law [4] is the observation that the number of transistors in a dense integrated circuit doubles about every two years. Gordon Moore described this effect (figure 1.1) on a paper in 1965 as an observation and projection of a historical trend and not a physical or natural law. It could be said that its effect still holds in our days although the rate held steady from 1975 until around 2012, when the rate increased up to a double every 18 months. Also, this transistor shrinkage provides more space to add specialized processing units to deal with features such as graphics, video, and cryptography.

On the other hand, Pollack's Rule states[7] that performance increases due to microarchitecture techniques approximate the square root of the complexity (number of transistors or the area) of a processor, formally:

$$\sqrt{\frac{Q_c}{Q_p}} = P$$

where Q_c, Q_p are current and previous number of transistors

and P is performance boost

Following Moore's statement that each new technology generation doubles number of transistors, and applying Pollack's rule implies that microarchitecture advances improve the performance by $\sqrt{2} \approx 41\%$. Therefore, the overall performance increase per generation is roughly two-fold, while the power consumption stays the same. This performance increase lies in exploiting dynamic execution and on-chip caching and prefetching at the expense of using more transistors and increasing the processor complexity.
To manage CPU power dissipation, processor makers favor multi-core chip designs. Many multi-threaded development paradigms will not see a linear increase in speed vs number of processors. This is particularly due to lock contention and communication time between

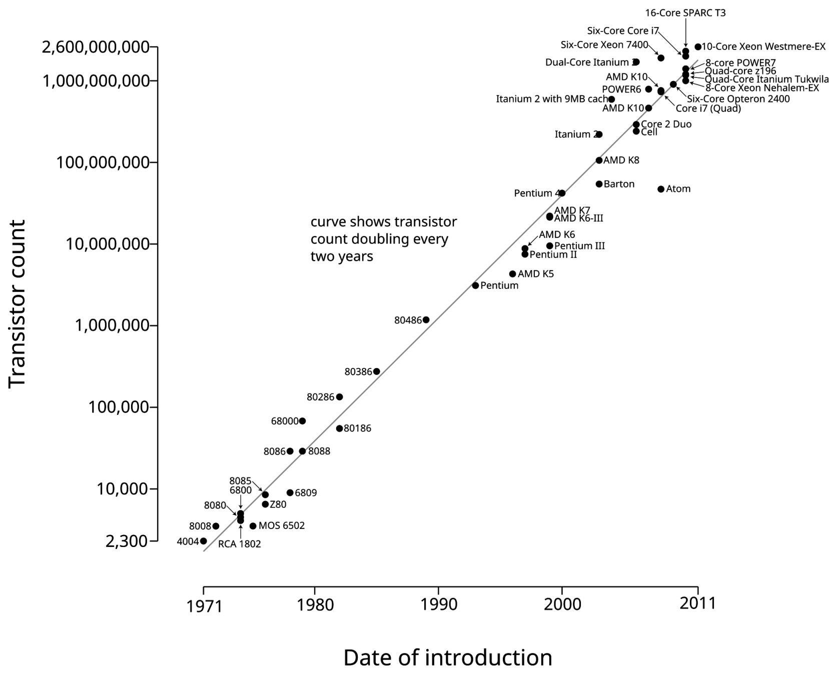


Figure 1.1: Microprocessor transistor counts & Moore's Law [4]

cores while accessing shared or dependent resources. The effect becomes more noticeable as the number of processors increases. To add on these problems, transistors smaller than $7nm$ will experience a quantum phenomenon known as quantum tunnelling. To understand the phenomenon, particles attempting to travel between potential barriers can be compared to a tiger trying to jump a zoo barrier; quantum mechanics and classical mechanics differ in their treatment of this scenario. Classical mechanics predicts that this tiger that does not have enough energy to classically surmount the barrier will not be able to reach the other side. However, in quantum mechanics the tiger can — with a given probability — tunnel to the other side, thus crossing the barrier. This phenomena can affect MOSFE transistors that are fabricated by doping a semiconductor, typically silicon ; i.e., intentionally introducing impurities for the purpose of modulating its electrical properties. A semiconductor doped to high levels acts more like a conductor than a semiconductor allowing electron flow as seen in 1.2. Due to this phenomenom, electrons may be able to tunnel pass through the oxide-insulator layer of a transistor.

Other alternatives - TFET

Other approaches have been taken to solve the issue. The state of the art in all the alternative approaches is clearly the TFE transistor [8]. It is an experimental type of less than $7nm$ transistor with a very similar structure to a MOSFET (with a different fundamental switching

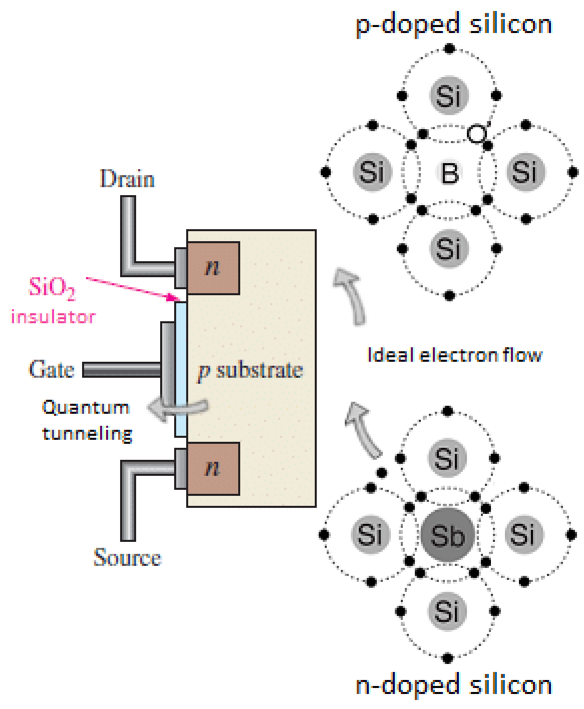


Figure 1.2: Quantum tunneling on transistors through SiO_2 insulating layer

mechanism) making this device a promising candidate for low power electronics. Due to the costs involved in development, less than $7nm$ transistors are predicted to take longer to reach market than the two years estimated by Moore's law.

1.2.2 Reversibility - Energy efficiency and utility

For the entire history of computing, our calculating machines have operated in a way that causes the intentional loss of some information (it's destructively overwritten) in the process of performing computations. Rolf Landauer's principle [9] argues that the logically irreversible character of conventional computational operations has direct implications for the thermodynamic behavior of a device that is carrying out those operations. Landauer's principle theorizes about the lower limit of energy consumption of computation. Informally, the principle enunciates that if a observer loses information about a physical system, also loses the ability to extract work from that system.
His reasoning can be better understood considering a game of billiards[10]. The collision physics would be the same whether you ran them backward or forward, and you could work out the future configuration of the balls from their past configuration or vice versa (with the same difficulty). Suppose now that the balls, cushions, and slate were not frictionless. Then, sure, two different initial configurations might end up in the same configuration. The fric-

tional loss of information would then generate heat.
Every single active logic gate in conventional designs destructively overwrites its previous output on every clock cycle. Physically it grounds one part of a circuit that holds a charge, in effect converting the charge and the information it represents into heat [11]. These high temperatures limit modern chips in several ways:

- By the classical statistical thermal Maxwell–Boltzmann tail of carriers [12].
- The risk of overheating, which in itself creates the need to cool the system down doubling the power consumption as well.

Thus, reversible computing does not only reduce computer chips' power consumption, it also boosts its speed. More so, computers are estimated to consume as much as 10 percent of electricity in the United States [13], and chips are rapidly reaching the upper limits of their heat tolerance.
Generally, entropy increases when information is lost or erased. The OR gate starts with two input bits and ends with only one, it must necessarily lose information given that there is no way to carry all the information through. If, instead, we require all of our gates to have the same number of input and output bits, it's possible then to undo the computation. Although this criteria is necessary it does not make it sufficient to make a reversible computation. Bijectivity must also be met; i.e., the output set has to have a one-to-one correspondence with the input set.
The deferred measurement principle is a mathematical theorem without loss of generality which states that delaying measurements until the end of a quantum computation doesn't affect the probability distribution of its outcome. As a result of this principle, every change except measurement must be reversible. This means that it must be possible for us to recreate the initial state using only the output state, without additional information. In the next chapter, we will delve into the minutiae of how this is accomplished.

1.3 Structure

This work is divided into 5 chapters and 2 annexes. This structure is related to the different phases of the project. Its content is detailed below:

Chapter 1: The purpose and objectives of the project as well as the reasons that motivated it.

Chapter 2: Provides a brief context on quantum mechanics and quantum computing.

Chapter 3: Treats the theoretical concepts, along with their mathematical basis, which are important for modeling the uncertainty. Includes the tools used to build the simulator, its structure and operation.

Chapter 4: Shows the obtained results and how close they come to bringing a efficient solution.

Chapter 5: The work done is valued and compared with the state of the art.

Annex A: Contains the associated costs.

Annex B: Gantt chart of the task planning.

Chapter 2

Theoretical framework

In classical computing the bit is the basic unit of information with a singular value, 0 or 1, also interpreted as true or false, or any two values mutually exclusive. Quantum computing uses a new basic information unit known as qubit. Qubits are fundamental to quantum computing and are somewhat analogous to bits in a classical computer. Qubits can be in a state 1,0 or a lineal combination of any of the two, i.e., the system has a probability to collapse to each state. However, when qubits are measured the result is always either a 0 or a 1; the probabilities of the two outcomes depends on the quantum state they were in. Quantum computing power lies in this superposition principle, allowing a parallelistic effect in operations. There are currently two main approaches to physically implementing a quantum computer: analog and digital. Digital quantum computers use quantum logic gates to do computation. To ensue a better understanding of the underlying basis of the work done, this chapter consists of concepts together with their definitions, reference to relevant scholarly literature and existing theory that is used for this particular study and that relate to the broader areas of knowledge being considered.

The main conceptions treated in the following sections encompass the building blocks upon which quantum computing is built. Specifically this study focuses on:

- Classical computing and its unconventional variants leading to reversible computing.
- The quantum mechanics principles that support quantum computing roots.
- A simple introduction to quantum computing concepts crucial for this work.

BITS	**QUBITS**
Operated with logical gates (OR,XOR,AND...)	Operated with unitary gates
Can be cloned	Cannot be cloned (no-clone theorem) [14]
Can be stored/have a long life	Cannot be stored nor have a long life (yet)

Aforementioned conceptions are important because they are the lens through which the research problems are evaluated. Theoretical and conceptual frameworks also provide evidence of academic standards and procedure. They also offer an explanation of why the study is pertinent and how this study expects to fill the gap in the literature.

2.1 The Principles of quantum mechanics

2.1.1 Bra-ket notation

Bra-ket notation, first introduced in 1939 by Paul Dirac [15], is a standard notation for describing quantum states and will be used throughout this text.
In quantum mechanics the state of a physical system is identified as a complex Hilbertian space, $\mathcal{H}$. Each vecto r is called the ket, and is typically represented as $|\psi\rangle$. In this instance, the vector space dovetails to $\mathbb{C}^2$ because we have two basis states — i.e., true and false. Then:

$$|\psi\rangle = \begin{bmatrix} c_0 \\ c_1 \end{bmatrix}$$
$$\text{where } c_i \in \mathbb{C}^2$$

For each ket $|\psi\rangle$ there exists a dual bra which is the Hermitian adjoint [1] of the ket with the same label, typically represented as a row vector, and written:

$$\langle\phi| = \overline{|\psi\rangle}^T = \begin{bmatrix} \overline{c_0} & \overline{c_1} \end{bmatrix}$$
$$\text{where } c_i \in \mathbb{C}^2$$

In quantum mechanics the scalar product or action is written as the expression:

$$\langle\phi|\psi\rangle \in \mathbb{C} = \begin{bmatrix} \overline{c_0} & \overline{c_1} \end{bmatrix} \times \begin{bmatrix} c_0 \\ c_1 \end{bmatrix}$$

Typically interpreted as the probability amplitude for the state ψ to collapse into the state ϕ.

2.1.2 Heisenberg's uncertainty principle

The uncertainty principle first introduced in 1927 [16] states that the more precisely the position of some particle is determined [2], the less precisely its momentum can be known, and

[1]Considering the linear operator $A* : H_1 \rightarrow H_2$ between Hilbert spaces. The adjoint operator is the linear operator $A : H_2 \rightarrow H_1$

[2]It must be emphasized that measurement does not mean only a process in which a physicist-observer takes part, but rather any interaction between classical and quantum objects regardless of any observer.

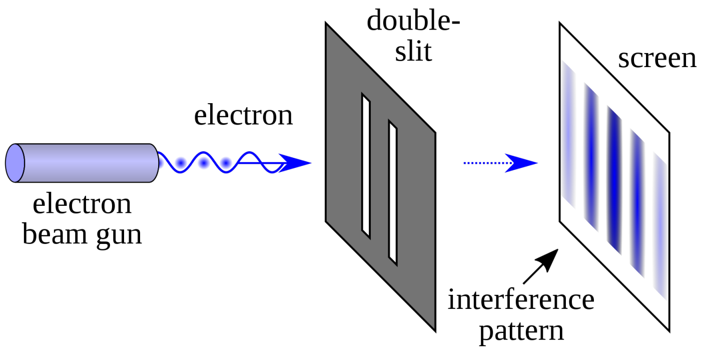


Figure 2.1: Two slit experiment

vice versa. The principle is inherent in the properties of all wave-like systems arises in quantum mechanics simply due to the matter-wave nature of all quantum objects. Heinsenberg demonstrated that it was not possible to contrive a method to locate the position of a subatomic particle unless we admitted some absolute uncertainty with regard to its exact velocity, since it is impossible to simultaneously and accurately measure both position and velocity

Double-slit experiment

Since the uncertainty principle is such a basic result in quantum mechanics, typical experiments in quantum mechanics routinely observe aspects of it. An especially unusual version of the effect is best demonstrated by the double-slit experiment[17]. Assuming light consisted strictly of particles, and assuming they were fired in a straight line through two slits and allowed to strike a screen on the other side, there should be a pattern corresponding to the size and shape of the apertures. Which means that there is certainty from which slit the particle came through. However, as can be seen in figure 2.1, when this experiment is actually performed a pattern with a series of alternating light and dark bands is observed. As it is a microscopic particle and its wave nature is significant, this leads to uncertainty in determining the position and momentum and a diffraction pattern is observed on the screen. This behaviour was later extended to electrons, atoms and molecules.

2.1.3 Quantum superposition

Quantum superposition states that, much like waves in classical physics, any two (or more) quantum states can be added together and the result will be another valid quantum state; and conversely, that every quantum state can be represented as a sum of two or more other distinct states. If given particle ψ has n possible basis states, then mathematically we have:

$$|x_0\rangle = \begin{bmatrix}1 & 0 & \dots & 0\end{bmatrix}^T$$
$$|x_1\rangle = \begin{bmatrix}0 & 1 & \dots & 0\end{bmatrix}^T$$
$$|x_{n-1}\rangle = \begin{bmatrix}0 & 0 & \dots & 1\end{bmatrix}^T$$

Where x_i represents each of the basis states of the system

As described precendently, ψ may be in a state $|\psi\rangle$ that is a linear combination of several basis states at the same time. Formally:

$$|\psi\rangle = \alpha_0 |x_0\rangle + \alpha_1 |x_1\rangle + \cdots + \alpha_n |x_n\rangle$$

Where the amplitudes $\alpha_i \in \mathbb{C}$ verify the law of total probability:

$$\sum |\alpha_i|^2 = 1$$

and $|\alpha_i|^2$ represent the probability of the system to collapse to state x_i

To better understand this phenomenon, suppose a beam of light shinnying through an imperfect sheet of glass that only transmits 95% of the photons. This makes perfect sense if light is a wave; the wave simply splits and a smaller wave is reflected back. Schrodinger's probability waves permits the existence of two or more waves. One wave would correspond to a photon passing through the glass and another wave would correspond to the photon bouncing back. But it is also possible for both waves to have superposed waves, which leads to the possibility of the photon being both transmitted and reflected, and therefore being on both sides of the glass simultaneously.

2.2 Introducing quantum computing

2.2.1 The Qubit

A quantum logical qubit state, as used in quantum information processing, is a quantum superposition of the basis states $|0\rangle$ and $|1\rangle$. Here $|0\rangle$ is the Dirac notation for the quantum state that will always give the result 0 when converted to classical logic by a measurement. Likewise $|1\rangle$ is the state that will always convert to 1. Mathematically:

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

However, whereas the state of a bit can only be either 0 or 1, the general state of a qubit according to quantum mechanics can be a coherent superposition of both. As stated in section 2.1.3, this superposition state could be described as:

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

where $|\alpha|^2$ and $|\beta|^2$ define the probability

for the system to collapse to state 0 and 1 respectively.

2.2.2 Measurement

Moreover, inasmuch as measuring a classical bit does not disturb its state, a measurement of a qubit destroys its coherence and irrevocably disturbs the superposition state. Then, how can the state of a qubit be measured? i.e., how can we assess the probability for the system to collapse to one state or the other? The procedure used to perform measurements on a qubit is simple to a certain extent. An analogy with a coin helps better understand the nature of the qubit. This coin has two possible static states, head or tails that could be represented: $|H\rangle\,, |T\rangle$. Now image this coin is falling through the air. In a certain moment it could be interpreted that it is in a superposition state of both basis states: $|H+T\rangle = \alpha\,|H\rangle + \beta\,|T\rangle$. We can not measure exactly the probability of the coin to fall on each side, and when it falls, the superposition state collapses to only one of the basis states: head or tails, i.e, the coherence is irrevocably disturbed. Now, if the event can be replayed from the first standpoint an empirical measurement of the number of times it lays on tails can be obtained. Then, using the classical definition of probability: P(Tails) = 1 - P(Heads) = $\frac{tails}{total}$. The average of the results obtained from a large number of trials should be close to the expected value, and will tend to become closer as more trials are performed [18]. This situation is very much alike to the qubit in that the initial configuration can be easily reconstructed and replayed a substantial number of times. Quantum algorithms are often probabilistic, in that they provide the correct solution only with a certain known probability, this is due to the fact that multiple measurements made on qubits in identical states will not always give the same result.

2.2.3 Bloch's sphere

The Bloch sphere is a unit 2-sphere seen in 2.2.

- A qubit is represented as a vector from the origin of the coordinate system to its surface.
- Each qubit is defined by the angles that said vector forms with the axes (latitude and longitude).

This vector can aim to the north and south poles of the Bloch sphere that are typically chosen to correspond to the standard basis vectors $|0\rangle$ and $|1\rangle$, respectively. It can also aim to any

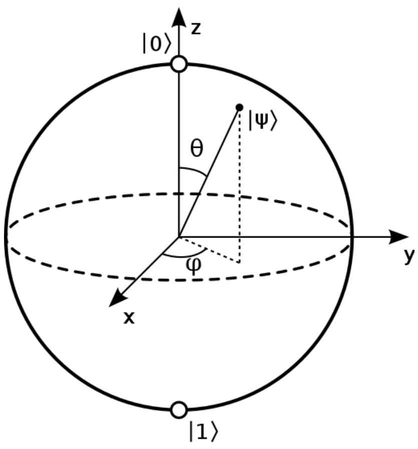


Figure 2.2: Bloch sphere [5]

other infinity of points distributed throughout the surface of the sphere, each one of them representing a particle in superposition. The probability of this superposition to collapse to any of the base states (north or south poles) corresponds to the latitude or longitude of the vector, i.e., how close its tip is to the endpoints of the geometric body.

2.2.4 Systems of qubits

The information contained in a qubit is relatively very small, so to be able to represent greater amounts of information, n-qubits systems are used. The laws of quantum mechanics describe the state of a system that consists of a set of n qubits as the tensor product of the n single qubits.

Tensor product

Suppose than $n = 2$ represents the spin of a system of two electrons. The spin of an electron can be in two states. When combined, four states for the system are generated:

$$\{|0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle\} = \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$$

This is an orthonormal basis with states that are unitary vectors. Hence, a n-qubit can be found in any state of the form:

$$\psi = \alpha_0 |0 \dots 0\rangle + \alpha_1 |0 \dots 1\rangle + \alpha_2 |1 \dots 0\rangle \dots \alpha_n |1 \dots 1\rangle$$
$$\alpha_0, \alpha_1, \alpha_2 \dots \alpha_n \in \mathbb{C}$$
$$|\alpha_0|^2 + |\alpha_1|^2 + |\alpha_2|^2 \dots |\alpha_n|^2 = 1$$

2.2.5 Operations with qubits

The dynamic evolution of an n-qubit is determined by unitary operators U in a Hilbert space. U is denoted by:

$$U\psi_1 = \psi_2$$

Furthermore, the following property applies:

$$U^\dagger U = I$$

The application of U transforms states into states, preserving the norm. In general, the evolution of m computation steps is given by:

$$U^m |\psi(0)\rangle \rightarrow |\psi(m)\rangle$$

These evolution operators that operate on an n-qubit are a unitary matrix of dimension 2^n that represents reversible quantum logic gates. For example, we can study the behaviour of the negation gate matrix as follows:

By definition,

N	$\lvert 0\rangle$	$\lvert 1\rangle$
$\lvert 0\rangle$	0	1
$\lvert 1\rangle$	1	0

then,

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

Reversible computing architecture

A reversible architecture after performing a given computation gives both the computation and the original inputs. With a almost-only energetic cost of re initializing the system in order to prepare it for another computation. Moreover, the cost of re initializing does not depend on the complexity of the computation but only on the number of bits in the answer. 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. If the converse were to happen to be true, then some information would have to be destroyed. Reversibility is a basic property that has to be considered to understand quantum computation. It can easily be verified that the preceding not gate is reversible.

Hadamard gate

The behaviour of the Hadamard gate, H, transforms a one-qubit into a superposition of the elements of the basis. The description and transformations implemented by the Hadamard gate are as follows.

By definition,

$$\begin{array}{|c|cc|} \hline H & |0\rangle & |1\rangle \\ \hline |0\rangle & \alpha & \beta \\ |1\rangle & \gamma & \delta \\ \hline \end{array} \mapsto \begin{cases} H\,|0\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \\ H\,|1\rangle = \gamma\,|0\rangle + \delta\,|1\rangle \end{cases}$$

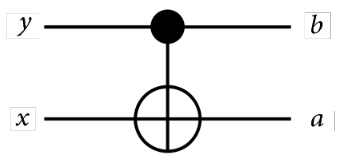
such that

$$|\alpha|^2 = |\beta|^2 = \frac{1}{2} \to |\alpha| = |\beta| = \frac{1}{\sqrt{2}}$$
$$|\gamma|^2 = |\delta|^2 = \frac{1}{2} \to |\gamma| = |\delta| = \frac{1}{\sqrt{2}}$$

To comply with the reversibility property, H gate must at least follow one of the ensuing:

$$\alpha \neq \gamma \vee \beta \neq \delta$$

This is due to the fact that both output values from applying H to $|0\rangle$ and $|1\rangle$ must be different. Given that probabilities are calculated using the absolute value of the complex amplitudes α,



Input		Output	
x	*y*	*a*	*b*
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	1

Figure 2.3: CNOT gate

β, γ and δ, then H can be built reversibly robust:

$$\gamma = -\frac{1}{\sqrt{2}} \text{ verifies all previous conditions and allows for reversibility}$$

$$H\left|0\right\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}}(\left|0\right\rangle + \left|1\right\rangle)$$

$$H\left|1\right\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}}(\left|0\right\rangle - \left|1\right\rangle)$$

Controlled not

Similar to the Not gate, but not identical, CNOT is a gate that operates on two inputs to generate two outputs. Changing the second if and only if the first is $\left|1\right\rangle$, illustrated in 2.3. ***b*** can also be interpreted as the output of a binary exclusive disjunction (XOR gate $\oplus$) with inputs x and y:

a = x $\oplus$y

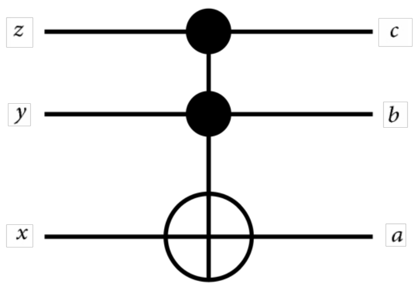
Controlled CNOT

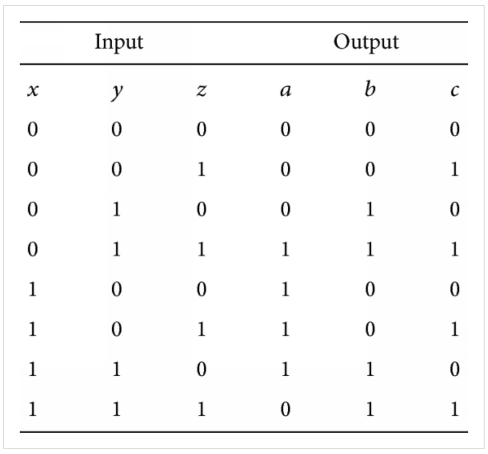
Similar to the CNOT gate, CCNOT (also known as Toffoli gate) is a gate that operates on three inputs to generate three outputs. Changing the third if and only if the two firsts are $\left|1\right\rangle$, illustrated in 2.4. ***b*** can also be interpreted as the output of two binary exclusive disjunction (XOR gate $\oplus$) with inputs x, y and z:

a = x $\oplus$y $\oplus$z

2.2.6 Quantum computer physical implementation

To be able to physically operate on a qubit, its coherent state must be preserved a sufficient amount of time for the operations to take place. One of the greatest challenges is controlling





Input			Output		
x	y	z	a	b	c
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	1	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	0	1	1

Figure 2.4: CCNOT gate

or removing quantum decoherence. This usually means isolating the system from its environment as interactions with the external world cause the system to decoher. In order to work with qubits for extended periods of time, they must be kept very cold as well as free of electromagnetic interference or vibrations. Any heat in the system can introduce error, which is why quantum computers are designed to create and operate at temperatures near absolute zero. However, other sources of decoherence also exist, such as the quantum gates[3], the lattice vibrations[4] and background thermonuclear spin of the physical system used to implement the qubits.

Decoherence is irreversible, as it is effectively non-unitary, and is usually something that should be highly controlled, if not avoided. Today's physical quantum computers are very noisy and quantum error correction is a burgeoning field of research. Existing hardware is so noisy and dependent on the uncertainty principle for their operations that include extremely low-noise technology such as that required in gravitational wave interferometers.

2.2.7 Summary

The most general normalized one-qubit that can be built is the linear superposition of two elements of the basis:

$$|x\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle ; \alpha_0, \alpha_1 \in \mathbb{C}; |\alpha_0|^2 + |\alpha_1|^2 = 1$$

As mentioned, a qubit can be in two possible states but can also be in intermediate states, that is to say, in states that are linear combination of $|0\rangle$ and $|1\rangle$. Thus, for example, the spin of an

[3]Each quantum gate applied to a qubit has a probability to fail due to technological blemishes. Usually this precision is measured using the controlled-not gate accuracy.

[4]Oscillations of atoms in a solid about the equilibrium position

	Physical implementation	**Gates**	**Mathematical Expression**
Classic computing	Voltages, fluidics[19]...	Logic gates	Boolean function
Quantum computing	Electron spin, trapped ions...	Quantum logic gates	Unitary matrices

Table 2.1: Quantum vs classical computation

electron can be in state:

$$\psi = \frac{1}{2}\left|0\right\rangle + \frac{\sqrt{3}}{2}\left|1\right\rangle$$

We conclude, therefore, that the probability of this superposition to collapse to any of the base states (north or south poles of the spherical coordinates on the Bloch sphere) corresponds to the latitude or longitude of the vector, i.e., how close its tip is to the endpoints of the geometric body. A quick comparison with classic computing can be seen in table 2.1.

Chapter 3

Modeling uncertainty

The main goal of this section is to assemble a QRBS [3] and put into use the proposed quantum method[2] to model the uncertainty that may appear in it. In the resolution of a problem the facts may be affected by imprecision and, nevertheless, the rules of the knowledge base must be used to obtain valid inferences. For example: the fact $\mathbb{A}$, which looks like A but it is not exactly A. We also have the fact $\mathbb{B}$, which looks like B but it is not exactly B. The question implies to be able of making inferences with $\mathbb{A}$, $\mathbb{B}$ and A $\wedge$B $\rightarrow$ C.

3.1 The Model

3.1.1 Classical Rule-Based Systems

In conventional RBSs, any categorical rule can be represented by the logical operators $and(\wedge)$, $or(\vee)$ and $not(\neg)$. Figure 3.1 exhibits the truth tables and associated probabilities of these conventional logical operators. The R1 program was a production rule-based system to assist in the ordering of computer systems by automatically selecting the computer system components based on the customer's requirements[20]. Consider the following rules of a similar but simpler system about diagnosing car problems[21]:

- R1: gas_in_tank$^{(A)}$ $\vee$ gas_in_carb$^{(B)}$ $\implies$ gas_in_engine$^{(C)}$
- R2: gas_in_engine$^{(C)}$ $\wedge$ turns_over$^{(D)}$ $\implies$ problem(spark_plugs)$^{(E)}$
- R3: not(turns_over)$^{(\neg D)}$ $\wedge$ lights_on$^{(F)}$ $\implies$ problem(starter)$^{(G)}$

The three previously defined rules can be represented classically by means of the inferential circuit of figure 3.1.

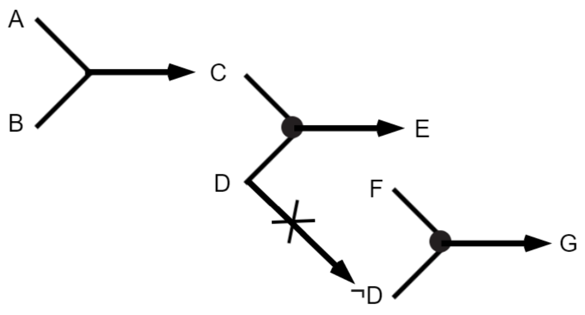


Figure 3.1: Simple inference circuit on diagnosing car problems

X	Y	NOT X	X AND Y	X OR Y
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1
		Probability = $\frac{2}{4}$ = 50%	Probability = $\frac{1}{4}$ = 25%	Probability = $\frac{3}{4}$ = 75%

Table 3.1: Conventional logical operators and associated probabilities

3.1.2 Quantum Rule-Based Systems

The combination of quantum gates allows to design and implement quantum logical operators. To explore the previous definitions in quantum terms, it is needed to formulate the $and(\wedge)$ and $or(\vee)$ quantum gates.

Quantum AND gate

In this regard, figure 3.2 shows the architecture of a quantum *and* gate, and table 3.2 the corresponding results obtained after 1024 executions over 64 iterations[1] in IBM Q[22] Tenerife quantum chip.

Quantum OR gate

Similarly, figure 3.3 shows the architecture of a quantum *or* gate and table 3.2 the corresponding results obtained after 1024 executions over 64 iterations in IBM Q Tenerife quantum chip.

[1]Experiments were conducted at 2019-08-23 11:53:36 am, with a CNOT gate error of 0.77×10^{-3} using projectQ and Python.

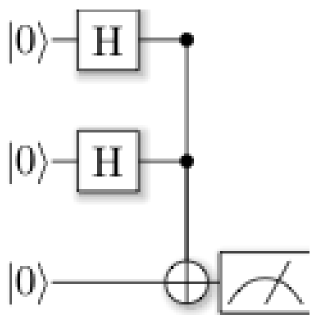


Figure 3.2: Quantum *and* gate

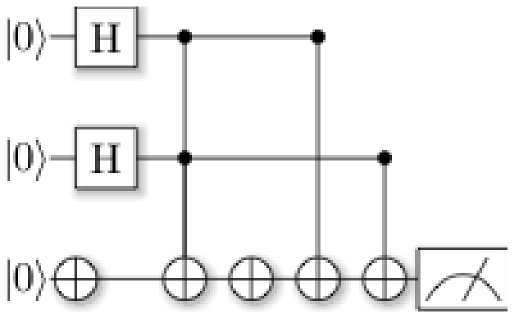


Figure 3.3: Quantum *or* gate

	Max % measured	**Min % measured**	**Standard deviation (over 64 iterations)**	**Mean (over 64 iterations)**	**Estimaded percentage**	**Precision**
AND	25.5%	24.87%	1.29%	25.20%	25%	99.20%
OR	75.1%	74.81%	1.31%	75.33%	75%	99.60%

Table 3.2: Results obtained after 1024 executions over 64 iterations in IBM Q Tenerife quantum chip.

3.1.3 Representing Uncertainty in Quantum Rule-Based Systems

For practical reasons the confidence in a given fact will be called “Credibility” from this stage forward. In such way that Credibility = 100 implies that the fact is true, and Credibility = 0 implies that the fact is false. The concept of credibility can be antonymously related with the concept of “Degree of Disbelief” associated to a given fact. The relation between these being Credibility = 100 − Disbelief. Going back to the Bloch sphere from section 2.2.3. The angle θ, which represents the displacement of the qubit vector along the Z axis, from the north pole to the south pole of the sphere, may be in three posible states:

- When Θ = 0 radians $\implies |0\rangle$
- When Θ = π radians $\implies |1\rangle$
- When $0 < \Theta < \pi \implies$ Coherent superposition

For the reasons just explained, the quantification of Z displacements in a Bloch sphere could be used to quantify our credibility associated to a given fact. To define a general procedure capable of representing any degree of uncertainty (or certainty) it would be convenient to create a single quantum gate respecting all the restrictions imposed by quantum mechanics. In this context there are already several universal gates, but none of them explicitly works with imprecise information in the domain of artificial intelligence. In this regard, and taking into account what has been described so far, [2] proposal continues as follows: Let γ be the degree of subjective disbelief that we can associate with a fact, defined in the closed interval [v_{min},v_{max}]. Then, γ, as any other variable (univariate distribution) can be rescaled into a parameter δ defined in the closed interval [0, 100], by the following formula:

$$\frac{100}{v_{min} - v_{max}} \times (v - v_{min})$$

The degree of subjective disbelief δ can be converted into an α azimuth that satisfies the restrictions of Z displacements and is defined in the closed interval [0, π].

- When $\delta = 0 \implies$ Our credibility in the fact is total $\implies \alpha = 0$
- When $\delta = 100 \implies$ Our credibility in the negation of the fact is total $\implies \alpha = \pi$
- When $0 < \delta < 100 \implies$ Subjective disbelief in the fact $\implies 0 < \alpha < \pi$

Then, as previous established, δ can be also be rescaled into a parameter α defined in the closed interval [0, π], by the following equation. Where α is compatible with the restrictions imposed by the Bloch sphere:

$$\alpha = \pi \times \frac{\delta}{100}$$

δ **(Subjective Disbelief)**	α **(Radians)**	Θ **(Radians)**
0	0	$\frac{\pi}{2}$
25	$\frac{\pi}{4}$	$\frac{3\pi}{8}$
50	$\frac{\pi}{2}$	$\frac{\pi}{4}$
75	$\frac{3\pi}{4}$	$\frac{\pi}{8}$
100	π	0

Table 3.3: Correspondence between the values of the parameters δ, α and Θ

Table 3.3 illustrates the values of α as a function of the values of δ —defined in the interval [0, 100], and the corresponding redefined values of the angle of rotation, or displacement, in Z as:

$$\Theta = \frac{\pi - \alpha}{2}$$

We will now define, based on the angle Θ, the following quantum gate matrix:

$$M(\Theta) = \begin{bmatrix} sin\Theta & cos\Theta \\ cos\Theta & -sin\Theta \end{bmatrix}$$

This matrix verifies that:

$$M(\Theta) \times M(\Theta)^{\dagger} = \begin{bmatrix} sin\Theta & cos\Theta \\ cos\Theta & -sin\Theta \end{bmatrix} \times \begin{bmatrix} sin\Theta & cos\Theta \\ cos\Theta & -sin\Theta \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Then, for any rule-based system, the degree of disbelief associated to a given fact can be managed with the following quantum model:

$$|\Psi(\Theta)\rangle = |\Psi(\frac{\pi - \alpha}{2})\rangle = |\Psi(\frac{\pi}{2})(1 - \frac{\delta}{100})\rangle = sin(\Theta)\,|0\rangle + cos(\Theta)\,|1\rangle$$
$$\text{Where } \alpha \in [0, \pi], \delta \in [0, 100]$$

Coming sections will focus on verification and thorough evaluation of this model implementing several RBS.

3.2 Implementation

3.2.1 Methodology

The evaluations needed to be developed must cover a wide angle of variables and restrictions:

- It is known in beforehand the expected results of the model, i.e., the behaviour of this system must match that of the predictions postulated by probability theory.

- The model must be proven using a real quantum chip. For this purpose IBM QE cloud quantum computing will be utilized.

- IBM QE requires a particular framework to communicate with its backend. Section 3.2.2 will delve into further explanations.

- Due to quantum computing aspects, it is impossible to know the input of the system for each output. Which makes testing and debugging much harder.

- Dealing with a non deterministic system also makes of this a harder piece of software to deal with. The approach chosen and the reasons are shown in section 3.3.2.

- Executions may take long queues and accomplishment times.

The particular nature of the software needed to be developed make of this project a highly distinctive one among the classic-type software development projects. These particular traits require a singular type of development. A SCRUM-like development will be taken as reference doing less emphasis on teamwork and more insistence on increments, functionality and overlapping of phases. Other agile development techniques will be used, such as: evolutionary development, control version and continuous improvements encouraging rapid and flexible response to change. In particular the fact that the result of the computation is previously estimated make of this a perfect situation to use test-driven development. The process will rely on the repetition of a very short development cycle: requirements are turned into very specific test cases, then the software is improved so that the tests pass. Figure 3.4 shows a schematic of the of the planned workflow. The task was divided into small iterations focusing on: building working pieces of software that comply with the requisites needed, adding functionalities and verifying their correct performance.
After each segment a small meeting was made in order to reorient the objectives and evaluating these functionalities.

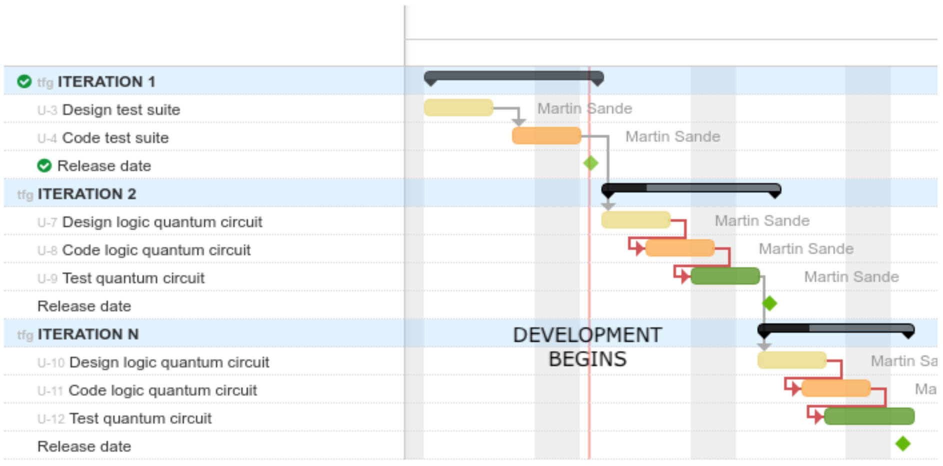


Figure 3.4: Methodology time line for the model evaluation

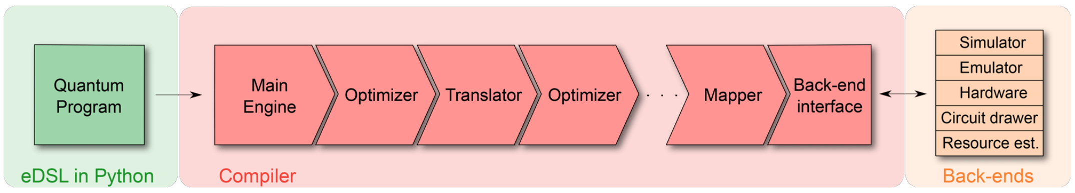


Figure 3.5: ProjectQ compilation framework

3.2.2 Technologies

ProjectQ and Python 3

ProjectQ[23] is an open-source software framework for python and quantum computing started at ETH Zurich. It features a compiler framework capable of targeting various types of hardware, a high-performance simulator with emulation capabilities, and compiler plug-ins for circuit drawing and resource estimation. The framework also allows testing of quantum algorithms through simulation and enables running them on actual quantum hardware using a back-end connecting to the IBM Quantum Experience cloud service.

Figure 3.5 shows the different compilation framework blocks:

- Main compiler engine, which executes a user-defined sequence of compilation steps by sending the circuit through a chain of so-called compiler engines.
- Each compiler engine manipulates the circuit to, e.g., reduce the number of gates or quantum bits required to run the quantum program. This is crucial due to quantum decoherence times in modern simulators.

- Engines further down the stack become more back-end-specific and take care of the mapping of the logical circuit to the layout of the back-end.

The framework also allows for quantum circuitry drawing in several formats: *PDF* and LaTeX using the TikZ package.

GitHub

GitHub provides hosting for software development version control using Git. As design goes on, it is common that there exist multiple versions of the same software and to be working simultaneously on updates. Therefore, for the purposes of locating and fixing bugs, it is vitally important to be able to retrieve and run different versions of the software to determine in which version(s) the problem occurs. GitHub offers all of the distributed version control and source code management functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project. Git supports rapid branching and merging, and includes specific tools for visualizing and navigating a non-linear development history. This ensures that the majority of management of version control steps are hidden behind the scenes.

Travis CI

When embarking on a change, the longer development continues on a branch without merging back to the mainline, the greater the risk of failures. Continuous integration involves integrating early and often, doing a complete build and passing all tests. Integration tests are usually run automatically on a CI server when it detects a new commit. Travis CI is a hosted continuous integration service used to build and test software projects hosted at GitHub. Travis CI supports integration with external tools such as coverage analyzers or static analyzers.

MutPy

MutPy is a mutation testing tool for Python source code that supports standard unittest modules. It applies mutation on AST[2] level and can boost mutation testing processes with high order mutations and code coverage analysis.

[2]The *ast* module helps Python applications to process trees of the Python abstract syntax grammar

Pylama

Pylama is a Python tool that wraps several code audit tools such as: *Pylint*[3], *Radon*[4] and *Pyflakes*.

3.3 Evaluation

There are 3 main focus while testing the model:

- Efficiently implementing *OR* and *AND* gate. This step is crucial in order to achieve a correct behaviour of the main rule-based systems.
- Creating pseudo-random rule-based systems.
- Computing classic (expected) values as well as quantum (experimental) values and comparing them.

Figure 3.2 and 3.3 show the quantum circuits for the quantum *OR* and *AND* gates. Considering that each *CNOT* gate used takes some time to execute, it is very important to use the minimum number of them to minimize execution time, thus preventing system decoherence before the computations ends. An issue to keep in mind is error propagation, which is also proportional to the number of *CNOT* gates used.

3.3.1 Simulator's structure and functions

The developed system creates pseudo-random rule-based systems and assigns each variable a random subjective degree of disbelief. Then, computes the a priori statistical probabilities (called "prior" from this stage forward). The rule-based systems are converted to their quantum counterpart and the experimental results are obtained in order to be later compared with the *prior* results. Take for example a simple arrangement seen in figure 3.6. Prior probabilities

[3]Source-code, bug and quality checker for Python. They follows the style recommended by PEP 8, the Python style guide [24].

[4]Python tool that computes cyclomatic complexity and Halstead metrics from the source code.

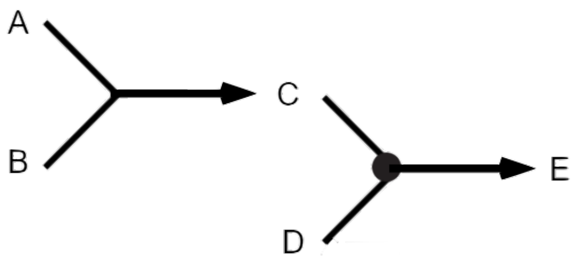


Figure 3.6: Sample system to evaluate

can be calculated formally as follows:

$$P(E) \vdash P(C) \wedge P(D)$$

Since C and D are independent, rule of product postulates:

$$P(E) \vdash P(C) \times P(D)$$

The only value left to achieve then is $P(C)$ that can be obtained:

$$P(C) \vdash P(A) \vee P(B)$$

Rule of sum states,

$$P(A) \vee P(B) \vdash P(A) + P(B) - P(A) \times P(B)$$

Reducing:

$$P(E) \vdash (P(A) + P(B) - P(A) \times P(B)) \times P(D)$$

Assuming each variable prior probability is its specific degree of disbelief, an actual value can be achieved.

If δ = 50, then:

$$P(E) = (0.5 + 0.5 - 0.5 \times 0.5) \times 0.5 = 0.375$$

The system then translates the rules to the correspondent quantum counterpart, seen in figure 3.7 Then, computations are done using *ProjectQ* framework to communicate with IBM QE cloud quantum computing backend. The overall structure of the simulator can be seen in figure 3.8. Measuring the fifth qubit gives us a experimental approximation of 37.69% with a derive of less than 0.19%[5].

[5]1024 executions on IBM QE Tenerife

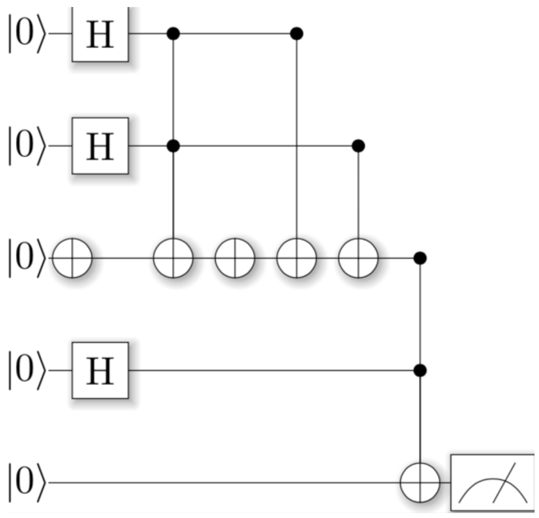


Figure 3.7: Quantum homologous of 3.6

3.3.2 Property-based testing

The main problem while testing quantum circuits is verifying the outputs from the system correspond to the correct inputs. To correctly solve this problem, the question to answer should be: What is the best way to test a classic OR gate? The first approach would be to check inputs and outputs.

But how can this be achieved if the input from the system can not be known? i.e. the input qubit is in a superposition state with a given probability to collapse to both $|0\rangle$ and $|1\rangle$. This is the situation that arises when trying to determine if quantum states or operations do what they are supposed to do, based only on classical input-output behavior. Posing two main issues:

The concrete state of the qubit is not known until the end of computations, when measurements are made.

The outputs are non-deterministic.

Property-based testing is a key concept to help solve the matter. The approach would be the same as to trying to proof the correctness of an arithmetic adding function. Is it a good approach to check random numbers and verifying that their sum adds up to the expected value? It sure must be necessary but would not be sufficient. Other properties should be met as well:

- Commutative: $x + y = y + x$
- Additive Identity: $x + 0 = x$

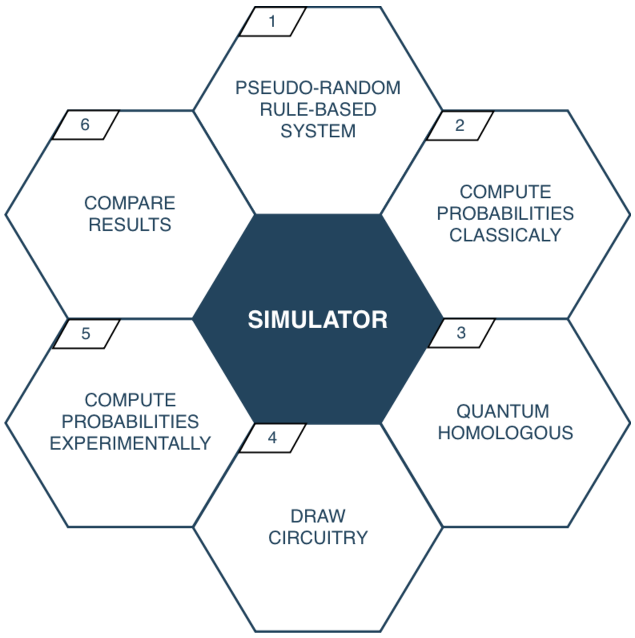

1
PSEUDO-RANDOM RULE-BASED SYSTEM
2
COMPUTE PROBABILITIES CLASSICALY
3
QUANTUM HOMOLOGOUS
4
DRAW CIRCUITRY
5
COMPUTE PROBABILITIES EXPERIMENTALLY
6
COMPARE RESULTS
SIMULATOR

Figure 3.8: Simulator workflow structure

- Some others...

The same happens when verifying the correctness of quantum circuitry. The system's experimental probabilities must match the expected values, but this is not sufficient. Other qualities must be satisfied as well:

- The calculation must end with a measurement, collapsing the system of qubits into one of the basic states, where each qubit is zero or one, decomposing into a classical state.
- The same observable has to show the same result every time after measurements are done.

Finally, surveying known bounds on testing various natural properties, such as whether two states are equal, whether a state is separable, whether two operations commute, etc.[25]

3.3.3 White-box testing

Mutation testing

Given that a huge requisite of this software is to test and thoroughly evaluate the given model[2], the creation of tests still poses the question whether the tests are correct and sufficiently cover the proposed issues. This technological problem is itself an instance of a deeper problem named *Quis custodiet ipsos custodes*[6].

Mutation testing is used to help in developing effective tests or locate weaknesses in the test data used for the program or in sections of the code that are seldom or never accessed during execution. Mutation testing involves modifying a program in small ways, such that unwanted problems must arise. These mutants are based on well-defined mutation operators that either mimic typical programming errors (such as using the wrong operator or variable name) or force the creation of valuable tests (such as dividing each expression by zero). If a mutant is added without being detected by the test suite, this indicates either that the code that had been mutated was dead[7] or that the test suite was unable to locate the faults represented by the mutant. A large number of mutants are usually introduced, leading to the compilation and execution of an extremely large number of copies of the program. However, MutPy(3.2.2) allows testing for individual portions of the application.

Code audit

Many higher-level languages, such as Python, have fewer potentially vulnerable functions. However, a comprehensive analysis of source code may be beneficial. Code audit is an integral part of the defensive programming paradigm. There are several tools that help with this task, especially with the intent of discovering bugs or violations of programming conventions. Such automated tools are applied as part of a policy-based approach only used to save time, but the project does not rely on them for an in-depth audit.

[6]Who will guard the guards?

[7]Referring to pieces of code that are never executed

Chapter 4

Results

Finally two new experiments were carried out. First off, an example inferential circuit with different values of the δ parameters associated with the different facts of the rules. The results obtained, illustrated in table 4.1, show that the approximation followed and the proposed model produce coherent results.

On the other hand, different circuits and δ were tested to obtain more general metrics and statistics.

4.1 Expected vs Experimental Results

4.1.1 Results ordered by different facts of the rules

Figure 4.2 represents the following rule-based system[2]:

- A AND $B \implies X$
- X OR $C \implies Y$
- D OR $E \implies Z$
- result $=$ Z AND Y

From this, a homologous quantum circuit was designed and assigned a given degree of disbelief associated with the fact of the rules. The experiment was conducted on IBM *ibmq_16_melbourne v1.0.0* (last calibration 2019-08-15 12:19:43 am). The backend system is represented in figure 4.1 and results are shown on table 4.1.

4.1.2 General experiment

As a next step of the experimentation, the developed model was applied to random rules with random δ. The experiment was conducted on IBM *ibmq_16_melbourne v1.0.0* (last calibration

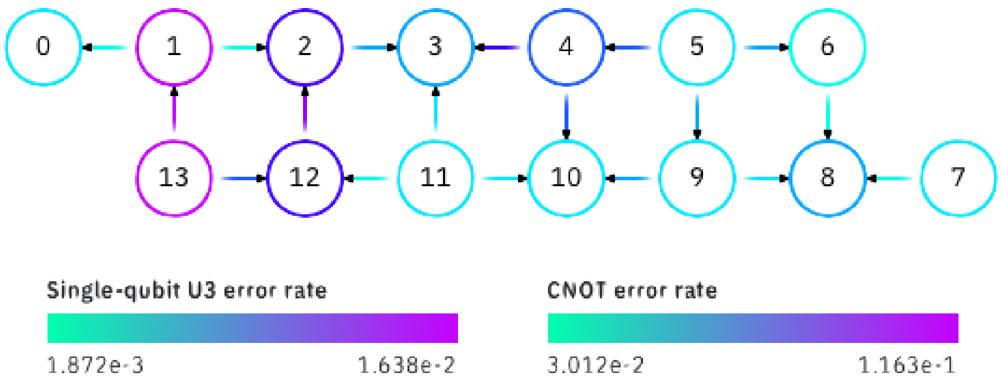


Figure 4.1: Melbourne system qubit distribution and error rates.

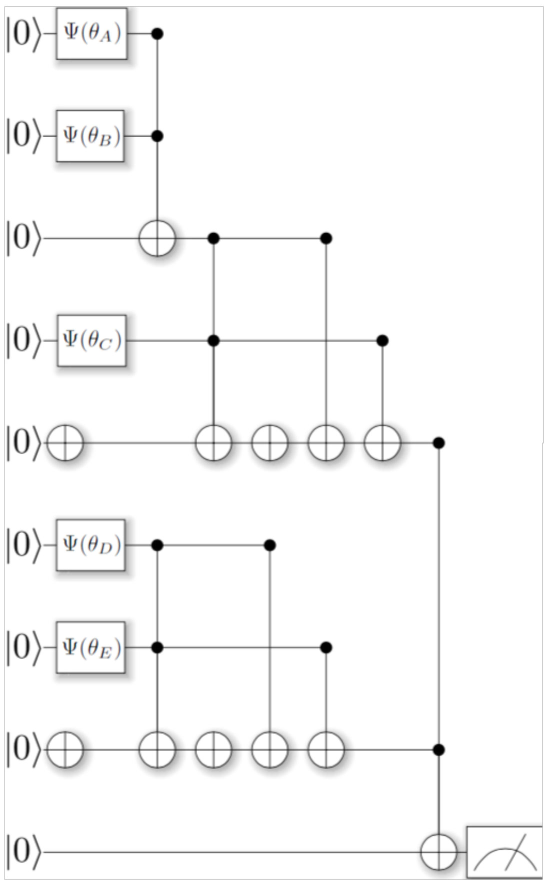


Figure 4.2: Quantum circuit and variables assigned different disbelief degrees.

δ(A)	δ(B)	δ(C)	δ(D)	δ(E)	EXPECTED	EXPERIMENTAL	PRECISION
0	0	0	0	0	0	0	100%
25	0	0	0	0	0	0	100%
50	25	0	0	0	0	0	100%
75	25	25	0	0	0	0	100%
100	50	25	25	0	0.075	0.078	96.15%
0	50	25	25	25	0.05	0.0419	83.80%
25	75	50	25	25	0.15	0.1587	94,52%
50	75	50	25	25	0.2	0.1975	98,75%
75	100	50	50	25	0.25	0.2490	99,60%
100	100	75	50	25	0.275	0.2778	98,99%
0	0	75	50	50	0.65	0.6519	99,71%
25	0	75	50	50	0.625	0.6332	98,70%
50	25	100	75	50	0.9225	0.9229	99,96%
75	25	100	75	50	0.9275	0.9283	99,91%
100	50	100	75	50	0.875	0.9324	93,84%
0	50	0	75	75	0	0	100%
25	75	0	100	75	0.125	0.1181	94,48%
50	75	0	100	75	0.375	0.4105	91,35%
75	100	25	100	75	0.8125	0.8841	91,90%
100	100	25	100	75	1.0	1.0	100,00%
0	0	25	0	100	0.15	0.1446	96,40%
25	0	50	0	100	0.5	0.5031	99,38%
50	25	50	0	100	0.5625	0.5451	96,91%
75	25	50	0	100	0.59375	0.5629	94,80%
100	50	75	25	100	0.875	0.932	93,88%
							Average = 96,92%

Table 4.1: Experimental results obtained with different δ values associated to the facts of the example 4.2

PRECISION	**PRECISION DEVIATION**	**MINIMUM PRECISION**	**MAXIMUM PRECISION**
96.45%	2.01%	98.43%[1]	86.71%

Table 4.2: Results obtained after 1024 executions on 128 different rule-based systems

2019-08-15 12:19:43 am). The backend system is represented in figure 4.1. The experimental results are compared with expected results in table 4.2.

Chapter 5

Conclusions

Disregarding the effectiveness of the model proposed in this thesis[2], the uncertainty associated with the information on real cases has induced the development of a myriad of models to try to solve the problem. The logical orientations of the different approaches on uncertainty vary throughout history. Also, faced with the same problem, different models produce different results, such as: definite approaches[26], probabilistic approaches[27], quasi-statistical approaches[1] or fuzzy methods [28]. Nevertheless, and regardless from the approaches taken to model it, uncertainty is still an extensive problem in artificial intelligence and, specifically, in Rule-Based Systems. The potential benefit of emerging theories could be taken into consideration to help solve the problem.

Among these applications, those that stand out, are Quantum Computing and its akin intrinsically probabilistic paradigms. This thesis proposes the *Q-AND* and *Q-OR* quantum operators, equivalent to the classic *AND* and *OR* operators. *Q-AND* and *Q-OR* allow the construction of Quantum Rule-Based Systems that benefit artificial intelligence establishing synergies with quantum computing. This work attempts to model uncertainty using Z displacements in the Bloch sphere[5] and δ values to represent subjectivity. A simulator was then built, bringing close focus on software development. The proposed approach is also thoroughly evaluated and verified using test-driven development with certain modifications to comply with quantum requirements. The results obtained after this exhaustive validation process allow us to conclude that Quantum Computation is an highly valid and efficient method to solve uncertainty problems in Artificial Intelligence.

Appendixes

.1 Materials

- MacBook Air 1,6 GHz Intel Core i5

- macOS Mojave 10.14.4

- Python 3.7.4

- Integration tests on: Python 3.7.1 on Xenial Linux

.2 Planning and cost

.2.1 Human resources

Table 1 shows human resource cost.

HUMAN RESOURCES	**Computer scientist**[1]	**Project Manager**[2]
cost €/year	23.296,60 €	37.274,92 €
cost €/hour	13.20€	21.13 €
total hours	257 days[3] $\times$ 1 hour $\approx$ 257 hours	10 days[4] $\times$ 1 hour $\approx$ 10 hours
total cost	257 hours $\times$ 13.20€ $\approx$ 3400€	10 hours $\times$ 21.13 € $\approx$ 220€ €

Table 1: Human resource cost for the project

.2.2 Material cost

All software utilized lies on free user license and computations were also done free of charge thanks to IBM QE.

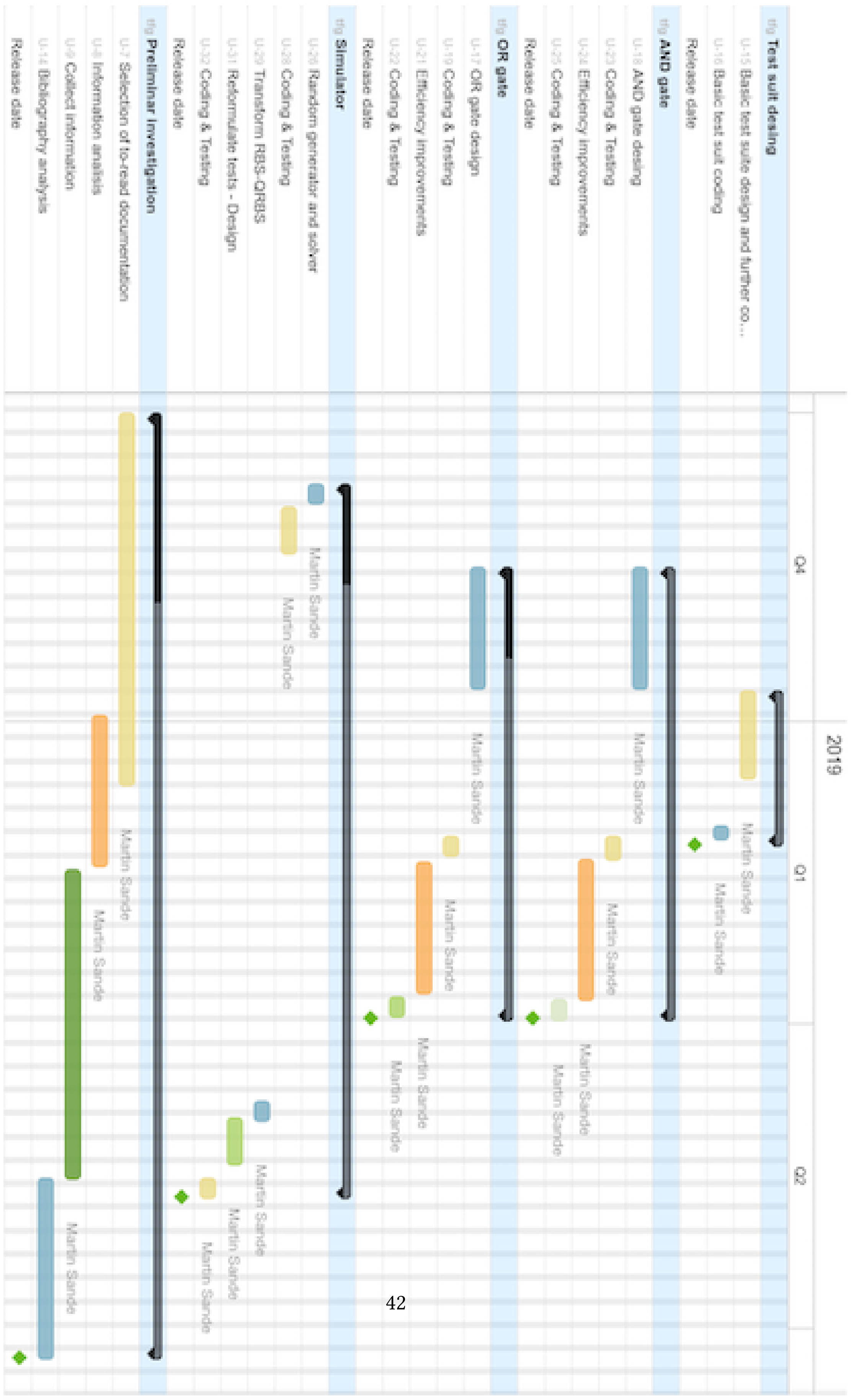


Figure 1: Planning: november 2018-july 2019

Appendix A

Glossary of acronyms

RBS	*Rule-based system.*
QRBS	*Quantum rule-based system.*
QC	*Quantum computing.*
AI	*Artificial intelligence.*
QM	*Quantum mechanics.*
CPU	*Central processing unit.*
MOSFET	*Metal-oxide-semiconductor Field-effect transistor.*
TFET	*Tunnel field-effect transistor.*
CNOT	*Controlled not.*
CCNOT	*Controlled controlled not.*
TDD	*Test-driven development.*

Appendix B

Glossary of terms

Qubit Quantum bit.

Microarchitecture Describes how an instruction set architecture is implemented in a particular processor.

Dynamic execution This paradigm, used in most high-performance CPUs a processor executes instructions in an order governed by the availability of input data and execution units, rather than by their original order in a program.

CPU Cache Small and faster memory, closer to a processor core used to reduce the average cost (time or energy) to access data from the main memory.

Pre-fetching Technique used by computer processors to boost execution performance by fetching instructions or data from their original storage in slower memory to a faster local memory before it is actually needed.

Bit Portmanteau of binary digit.

Bibliography

[1] A. P. Dempster, “Upper and lower probabilities induced by a multivalued mapping,” *The Annals of Mathematical Statistics*, vol. 38, no. 2, pp. 325–339, 1967.

[2] V. Moret-Bonillo, I. Fernández-Varela, and D. Álvarez Estévez, “Uncertainty in quantum rule-based systems,” *Under revision*, 2018.

[3] V. Moret-Bonillo, “Emerging technologies in artificial intelligence: Quantum rule- based systems,” *Progress in Artificial Intelligence*, vol. 7, no. 2, pp. 155–166, 2018.

[4] G. E. Moore, “Cramming more components onto integrated circuits,” *Electronics*, vol. 38, no. 8, April 1965.

[5] F. Bloch, “Nuclear Induction,” *Physical Review*, vol. 70, pp. 460–474, Oct. 1946.

[6] M. Hilbert and P. López, “The world’s technological capacity to store, communicate, and compute information,” *Science*, vol. 332, pp. 60–(), April 2011.

[7] F. J. Pollack, “New microarchitecture challenges in the coming generations of cmos process technologies,” *MICRO 32 Proceedings of the 32nd annual ACM/IEEE international symposium on Microarchitecture*, p. 2, 16-18 November 1999.

[8] F.-L. Yang, D.-H. Lee, H.-Y. Chen, C.-Y. Chang, S.-D. Liu, C.-C. Huang, and et al., “5nm-gate nanowire finfet,” *Digest of Technical Papers, 2004 Symposium on VLSI Technology*, 2004.

[9] R. Landauer, “Irreversibility and heat generation in the computing process,” *IBM Journal of Research and Development*, vol. 5, pp. 183–191, 1961.

[10] V. Moret-Bonillo, *Adventures in Computer Science: From Classical Bits to Quantum Bits.* Springer International Publishing, 2017.

[11] M. P. Frank, “Quantum computing’s practical cousin,” *Simons Conference Lecture*, May 28-31, 2003.

[12] G. DeMicheli, Y. Leblebici, M. Gijs, and J. Vörös, *Nanosystems Design and Technology.* Springer International Publishing, 2009.

[13] L.-B. Desroches, H. Fuchs, J. B. Greenblatt, S. Pratt, H. Willem, E. Claybaugh, B. Beraki, M. Nagaraju, S. K. Price, and S. J. Young, "Computer usage and national energy consumption: Results from a field-metering study," *LBNL Report number: LBNL-6876E*, 2015.

[14] J. L. Park, "The concept of transition in quantum mechanics," *Foundations of Physics*, vol. 1, pp. 23–33, 1970.

[15] P. A. M. Dirac, "A new notation for quantum mechanics," *Proceedings of the Cambridge Philosophical Society*, vol. 35, p. 416, 1939.

[16] W. Heisenberg, "Über den anschaulichen inhalt der quantentheoretischen kinematik und mechanik," *Zeitschrift für Physik*, vol. 43, pp. 172–198, 1927.

[17] S. Eibenberger, S. Gerlich, M. Arndt, M. Mayor, and J. Tüxen, "Matter-wave interference of particles selected from a molecular library with masses exceeding 10 000 amu," *Physical Chemistry Chemical Physics (Incorporating Faraday Transactions)*, vol. 15, p. 14696, 2013.

[18] J. Bernoulli, "Ars conjectandi: Usum applicationem praecedentis doctrinae in civilibus," 1713.

[19] J. McKetta, "Fluid Flow," *Encyclopedia of Chemical Processing and Design*, vol. 23, p. 28, Nov. 1985.

[20] J. McDermott, "R1: An Expert in the Computer Systems Domain," *Proceedings of the First AAAI Conference on Artificial Intelligence*, p. 269–271, 1980.

[21] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. V. de Velde, and B. Wielinga, *Knowledge Engineering and Management: The CommonKADS Methodology.* MIT Press, 2000.

[22] "Ibm makes quantum computing available on ibm cloud to accelerate innovation," Retrieved May 4, 2016.

[23] D. S. Steiger, T. Häner, and M. Troyer, "ProjectQ: An Open Source Software Framework for Quantum Computing," *Quantum*, vol. 2, p. 49, 2018.

[24] "Pep 8 style guide for python code," https://www.python.org/dev/peps/pep-0008/, retrieved 2019-01-09.

[25] A. Montanaro and R. de Wolf, "A Survey of Quantum Property Testing," *Theory of Computing Graduate Surveys*, vol. 7, p. 67, 2016.

[26] L. RS and L. LB., "symbolic logic, probability, and value theory aid our understanding of how physicians reason," *Science*, pp. 9–21, 1959.

[27] J. Pearl, "Bayesian networks: A model of self-activated memory for evidential reasoning," *Proceedings of the 7th Conference of the Cognitive Science Society*, 1985.

[28] L.A.Zadeh, "Fuzzy sets*," *Information and Control*, vol. 8, pp. 338–353, 1965.