UNIVERSITÀ DEGLI STUDI DI NAPOLI
"FEDERICO II"

Scuola Politecnica e delle Scienze di Base

Area Didattica di Scienze Matematiche Fisiche e Naturali

Dipartimento di Fisica "Ettore Pancini"

Laurea Triennale in Fisica

Quantum computing: tools, errors and algorithms

Relatori:
Massimo Taronna

Candidato:
Maria Palomba
Matr. $N85000844$

Anno Accademico 2019/2020

A nonna Emmanuella

Contents

Introduction

Information is physical. The development of Quantum Theory was the turning point of the 20th century and it was born as the explanation of the counter intuitive behavior of subatomic particles and the phenomenon of entanglement. Later on, it came up that quantum theory is not only suitable for atoms and molecules, but also for bits and logic operation in a computer. Here follows a revolution. So now we wonder: how can Quantum Theory improve theory of information? Information is something we can encode and store in the state of a physical system, more specifically in a quantum state. The field of quantum information science is based on three central ideas, which are *quantum entanglement*, *quantum computing* and *quantum error correction*.

In the 1960's, thanks to the work of John Bell, quantum entanglement arose as something potentially useful that we could use to study a system as a whole made of non-locally correlated parts. Quantum entanglement is a physical phenomenon occurring when two or more particles interact in such a interconnected way that the quantum state of a single particle cannot be described if not considering the state of the others, whatever the distance separating the particles is.

Furthermore, the power of a quantum computer lies in a property called *superposition* for which a qubit can represent the states $|0\rangle$, $|1\rangle$ or a linear combination of both.

The impact of quantum theory on computation was firstly demonstrated in principle by Peter Shor in 1994. He showed that a quantum computer

can factor a large number efficiently. This problem belongs to a class of problems whose solution is very difficult to find, although easy to verify once found. For example, taken two large prime numbers p and q, their product $n = pq$ is easily calculable, but this is not valid at all if we know n and want to find p and q: the time needed is superpolynomial in $\log(n)$. Shor found that a quantum computer can factor in polynomial time, hence it would be able to factor a 400 digit number in less than three years! [17, 22, 10]

Purpose

Before getting started, I would like to look at two counter intuitive principles of quantum physics in order to set up the point of view for the following. First of all, a physical system in a definite state can still be random. Secondly, two systems that are far away from each other can behave randomly individually, but still be highly connected.
Therefore, the purpose is to follow this logical path:

1. providing an introduction to the subject of quantum computing;
2. studying errors and consequently Quantum Error Correction (QEC) giving some examples;
3. analyzing quantum circuits and some important algorithms; finally describing the IBM Quantum Experience by reporting simulations and critically discussing the results.

Chapter 1

Quantum computing fundamentals

1.1 What's a *qubit*?

Unlike binary bits of classical information, the corresponding carrier of quantum information is the quantum bit or *qubit*. The qubit is a vector in a two-dimensional complex vector space with inner product. The elements of an orthonormal basis in this space are designated $|0\rangle$ and $|1\rangle$ and a normalized vector can be represented as follows:

$$|\Psi\rangle = a\,|0\rangle + b\,|1\rangle,$$

where a and b are complex numbers satisfying $|a|^2 + |b|^2 = 1$.
A qubit can be in state $|0\rangle$, $|1\rangle$ or in a linear combination of both states. A quantum state of N qubits can be expressed as a vector in a Hilbert space of dimension 2^N, thus by a linear superposition of its orthonormal basis states.

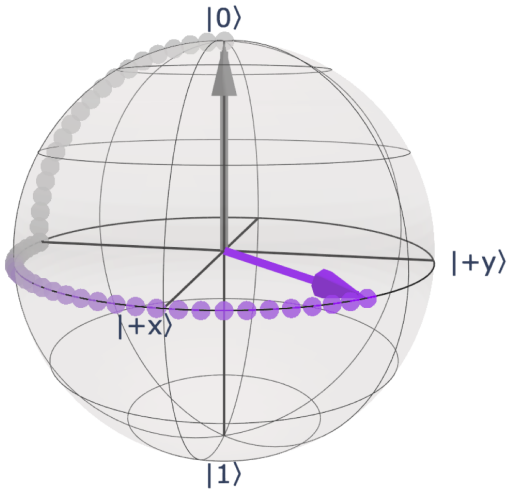

Figure 1.1: *Evolution of a qubit state vector after applying a series of gates*

Let's choose an orthonormal basis for this Hilbert space in which each qubit has a definite value, either $|0\rangle$ or $|1\rangle$, so that we can expand a general normalized vector as $\sum_0^{2^N-1} a_x|x\rangle$. a_x's are complex numbers satisfying $\sum_x |a_x|^2 = 1$. Now, we can shortly describe a *quantum computation*. It consists of assembling N qubits, preparing them in a standard initial state and then applying a unitary transformation U. After that, we can measure all the qubits by projecting them onto the $\{|0\rangle, |1\rangle\}$ basis. Besides, quantum algorithms are probabilistic and so a probability distribution of possible outcomes is generated. Except where $a = 0$ and $b = 0$, the measurement disturbs the state. However, after the measurement, the qubits have been prepared in a known state different from their previous state. Probabilities can add up in unexpected ways because of a phenomenon called *quantum interference*. To make an example, we may interpret a qubit as a spin $-\frac{1}{2}$ object, and its quantum state is characterized by a unit

vector (spin's direction) in three dimensions. A unitary transformation rotates the spin and the measurement of an observable can give two results: the spin is either up or down along a specified axis. Anyways, not every two-level system transforms as a spinor under spacial rotations. A classical computer can simulate a quantum computer until the number of qubits at stake does not increase significantly. But, as John Bell concluded, there is no local probabilistic algorithm that can reproduce the conclusions of quantum mechanics. Moreover, qubits can exhibit quantum entanglement. From a more practical perspective, qubits are particles that have some probability of being in each of two states $|0\rangle$ and $|1\rangle$ simultaneously. And when qubits interact, they become inevitably interdependent, plus the more operations are carried out, the more interconnected the qubits become. So, both the power and the weakness of a quantum computer lies in this exponentially growing number of possibilities. Qubits are very susceptible to errors, which we will see being of different types, so that even the weakest stimulus can cause them to under-go *bit-flips* or *phase flips*. So, in order to make a quantum computer work, ways to prevent the system from being corrupted by errors have to be developed. Correcting errors does not imply measurements leading to a collapse of the wave function. Qubits can be built by using any two level quantum mechanical system. Some examples are given: polarization encoding, electronic spin, electron number, superconducting flux qubit.

1.2 The density operator

Taken a Hilbert space H, the axioms of quantum mechanics establish that:

1. a state is a ray in H;
2. an observable is a self-adjoint operaror on H;
3. a measurement is an orthogonal projection;
4. time evolution is unitary.

But, if we consider just a part of a larger system, these axioms are violated. Instead, a quantum state is described by a density operator ρ rather than by a ray. Let's take into consideration a two-qubit system in which we observe only one of the qubits. We can use $\{|0\rangle_A, |1\rangle_A\}$ and $\{|0\rangle_B, |1\rangle_B\}$ as orthonormal bases for qubits A and B respectively. A quantum state of the two-qubits system in which qubits A and B are correlated is:

$$|\psi\rangle_{AB} = a|0\rangle_A \otimes |0\rangle_B + b|1\rangle_A \otimes |1\rangle_B.$$

Let's assume we measure A. $|a|^2$ is the probability to get the result $|0\rangle_A$ and the measurement prepares the state $|0\rangle_A \otimes |0\rangle_B$. The probability to obtain $|1\rangle_A$ is $|b|^2$ and here the prepared state is $|1\rangle_A \otimes |1\rangle_B$. Then, if we measure qubit B we have a 100% probability to find $|0\rangle_B$ if we have found $|0\rangle_A$ from the previous measurement; the same goes for $|1\rangle_B$ if we have already found $|1\rangle_A$. This means that the outcomes of the measurements are correlated in the state $|\psi\rangle_{AB}$.

At this point, let's consider an observable acting on qubit A only, that can be expressed as:

$$M_A \otimes \mathbf{1}_B.$$

The expectation value of the observable can be written in the form:

$$\langle \mathbf{M_A} \rangle = tr(\mathbf{M_A} \rho_\mathbf{A}),$$

where *tr()* is the trace and ρ_A is called the *density operator* or *density matrix* for qubit A. This operator is self-adjoint, positive and it has unit trace. So, for any observable acting on a certain qubit, it is likely to interpret ρ_A as an *ensemble* of possible quantum states, each one occurring with a specified probability. Furthermore, because of its properties, it follows that ρ_A can be diagonalized, the eigenvalues are real, nonnegative and they sum to one.
Density operator represents a state. If the state of the subsystem is a ray, we call the state *pure*; otherwise the state is *mixed*. If the state is a pure state $|\psi\rangle_A$, the density matrix $\rho_A = |\psi\rangle_{AA}\langle\psi|$ is the projection onto the one-dimensional space spanned by $|\psi\rangle_A$. So, a pure density matrix has the property $\rho^2 = \rho$.
In general, a density matrix expressed in the basis in which it is diagonal, has the following form:

$$\rho_A = \textstyle\sum_a p_a |\psi_a\rangle\langle\psi_a|,$$

where $0 < p_a \leq 1$ and $\sum_a p_a = 1$.
Yet, if the state is not pure, more terms appear in the sum and $\rho^2 \neq \rho$. In this case, ρ is an *incoherent* superposition of the states.
When a quantum system A interacts with another system B, they become *entangled*, i.e. correlated. Therefore, the entanglement breaks the coherence of a superposition of states and this means that some of the phases in the superposition are inaccessible if we consider one state at a time.

Density matrix of a qubit A qubit is a quantum system with a two-dimensional Hilbert space. The most general density matrix of a qubit takes the form:

$$\rho(\vec{P}) = \tfrac{1}{2}\left(\mathbf{1} + \vec{P}\cdot\vec{\sigma}\right),$$

where $\vec{P}$ is a 3-component vector of length $|\vec{P}| \leq 1$.

Ensembles The density operators on a Hilbert space form a convex set and the external points of the set are the pure states. A mixed state of a certain system A can be prepared as an *ensemble* of pure states. If we have a mixed state ρ_A of system A, any ensemble of pure states can be prepared by making a measurement in another system B entangled with A.

1.3 Quantum entanglement

The entanglement is a property of quantum systems comprised of more than one subsystem. It establishes an important difference between quantum information and classical information. Entangled states are interdependent and do not have a classical analog. Plus, an entangled state cannot be written as product of subsystem states. Consider two qubits A and B and suppose we trace over B in order to find the density operator ϱ_a of qubit A. Thus, we get a multiple of the identity operator $\varrho_a = tr\left(|\phi^+\rangle_{ABAB}\langle\phi^+|\right) = \frac{1}{2}\mathbb{I}_A$ and the same for ρ_B. This means that the state of the two qubits is maximally entangled. If we measure locally A or B we won't obtain any information about the state. A bipartite pure state can be expressed through the Schmidt

decomposition:

$$|\psi\rangle_{AB} = \sum_i \sqrt{p_i}|i\rangle_A|i'\rangle_B, \tag{1.1}$$

and the expansion is written in terms of orthonormal basis H_A and H_B. For instance, $|\psi\rangle_{AB}$ and $|\varphi\rangle_{AB} \in H_A \otimes H_B$ cannot simultaneously be expanded in the Schmidt decomposition. Now that we have defined the Schmidt decomposition, we can associate the Schmidt number (the number of nonzero eigenvalues in ρ_A) to any bipartite pure state, namely the number of terms in the Schmidt decomposition above. The reason why it is important to define Schmidt number is that it provides us with a parameter to establish whether a bipartite pure state is entangled or not. Indeed, $|\psi\rangle_{AB}$ is entangled if its Schmidt number is greater than one. A separable, or unentangled, pure state is a direct product of pure states in H_A and H_B expressed by the formula:

$$|\psi\rangle_{AB} = |\varphi\rangle_A \bigotimes |\chi\rangle_B.$$

A state that cannot be expressed this way is then entangled and the density operators are mixed.

1.4 Which role entropy plays?

Imagine to have a source that prepares messages of n letters, each of which is taken among an ensemble of quantum states. The tools at our disposal to transmit the signal are a set of quantum states ρ_x having a known a priori probability p_x. The probability of any outcome, stated that the observer has no information about which state was settled, is given by $\rho = \sum_x p_x \rho_x$. For any density matrix we can define the *Von Neumann entropy* [15], the concept of entropy in quantum statistical mechanics. It is given by:

$$S(p) = -tr(\rho \log \rho).$$

Von Neumann entropy plays a triple role. It quantifies quantum information content per letter of the ensemble, defining the minimum number of qubits per letter needed for the information to be encoded efficiently. Then it also quantifies classical information content and, last but not least, it quantifies the entanglement of a bipartite pure state.

Mathematical properties of Von Neumann entropy In order to better understand the significance of entropy in quantum statistical mechanics, let's bear in mind some important properties of Von Neumann entropy.

- **Purity**: a pure state $\rho = |\phi\rangle\langle\phi|$ has $S(\rho) = 0$
- **Invariance**: the entropy is unchanged by unitary change of basis, because of $S(\rho)$ depending only on ρ
- **Maximum**: randomness in choosing quantum state maximizes entropy. $S(\rho) \leqslant logR$
- **Concavity** The more we ignore the way the state has been prepared, the more entropy grows.This property is so called because of the convexity of logarithm function. For $\lambda_1, \lambda_2, ..., \lambda n \geq 0$ and $\lambda_1 + \lambda_2 + ... + \lambda_n = 1$ we have:

$$S(\lambda_1\rho_1 + \lambda_2\rho_2 + ... + \lambda_n\rho_n) \geq \lambda_1 S(\rho_1) + ... + \lambda_n S(\rho_n)$$

- **Entropy of measurement** it is possible to minimize the randomness of the measurement by measuring an observable which commutes with the density matrix
- **Entropy of preparation** Mixing nonorthogonal pure states, it won't be possible to distinguish the former state anymore.

- **Subadditivity** Entropy is additive if the systems are not correlated. Otherwise the total entropy is smaller than the sum of the entropies of the single parts;

- **Strong subadditivity** AB and BC are two overlapping subsystems; the entropy of their union plus the entropy of the intersection never exceeds the sum of the entropies of the subsystems AB and BC

- **Araki-Lieb inequality** Consider a bipartite system AB. This formula can be applied: $S(\rho_{AB}) \geq |S(\rho_A) - S(\rho_B)|$. This, again, means that we can only deduce information from the whole and will make completely random and unpredictable any outcome of measurement over separate subsystems. And that is how information is encoded in nonlocal quantum correlations.

Chapter 2

Quantum Error Correction

The qubits are delicate systems susceptible to errors.
A complex Hamiltonian describing the evolution of a system leads to errors and modifies information. This information can be recovered through a time-reversed dynamics which can also correct a damage due to a local measurement that causes a disturbance of the quantity gauged. Quantum Error Correction (QEC) is a subject emerging from the intersection of quantum mechanics and classical theory of error correcting codes [15, 23, 6]. QEC has to do with communication and information storage in the presence of noise [17, 6].
Error correction is important in different areas, but most of all in quantum computers because, the more efficient an algorithm is, the more it makes large-scale use of quantum interference, which is very inclined to inaccuracies. If there were no error correcting codes, large scale quantum computation would have been forbidden. The first quantum error correcting codes were discovered independently by Shor and Steane. Shor [4, 20, 19, 17] demonstrated that 9 qubits can be used to protect a single qubit against general errors. Steane described

a general code construction whose simplest example does the same job using 7 qubits. Knill and Laflamme, Bennett and others [23, 13, 11] described requirements for quantum error correcting codes and measures of fidelity of corrected states. Bennett *et. al.* [2, 1], and independently Laflamme *et. al.* [12], discovered the perfect 5-qubit code. The three central ideas on which QEC is based are:

- digitization of noise;
- manipulation of error operators and syndromes;
- quantum error correcting code (QECC) construction.

A quantum error correction code protects quantum information by encoding it non-locally, namely by spreading it among many qubits.

2.1 Environment and errors

Nonlocal correlations among different parts of a physical system are fragile and this is a problem to deal with if we think that in reality a quantum system is constantly correlated to its environment. So, we must take into consideration also the nonlocal correlations existing between a quantum apparatus and its environment. To be clearer, the environment becomes part of the system upholding the information itself. The contact between quantum computer and its environment is called *decoherence* and it causes errors. Decoherence is described by two phenomena: *energy relaxation* and *dephasing* [7].
Energy relaxation happens when a qubit decays from the high energy state to the low energy state, meanwhile dephasing refers to the phase relation between $|0\rangle$ and $|1\rangle$ states in a superposition degenerating. Errors are equivalent to quantum information spoiled. Thus, correcting

these errors makes a quantum computer be actually reliable. Before getting to the heart of the matter, let's see the difficulties we may encounter.

- **Bit-flips and phase errors**
  Bit-flip errors are like:
  $|0\rangle \rightarrow |1\rangle$
  $|1\rangle \rightarrow |0\rangle$
  While phase errors make the state $\frac{1}{\sqrt{2}}[|0\rangle + |1\rangle]$ flip to the orthogonal state $\frac{1}{\sqrt{2}}[|0\rangle - |1\rangle]$.
  No classical code corrects this type of error.

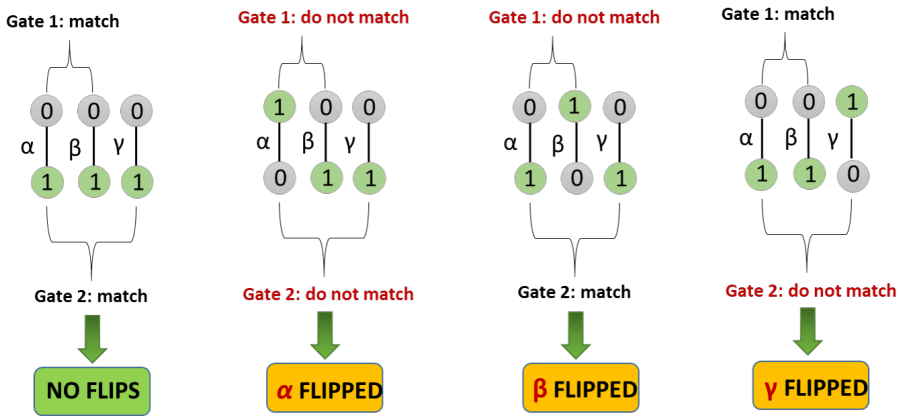

Figure 2.1: *Bit-flips*

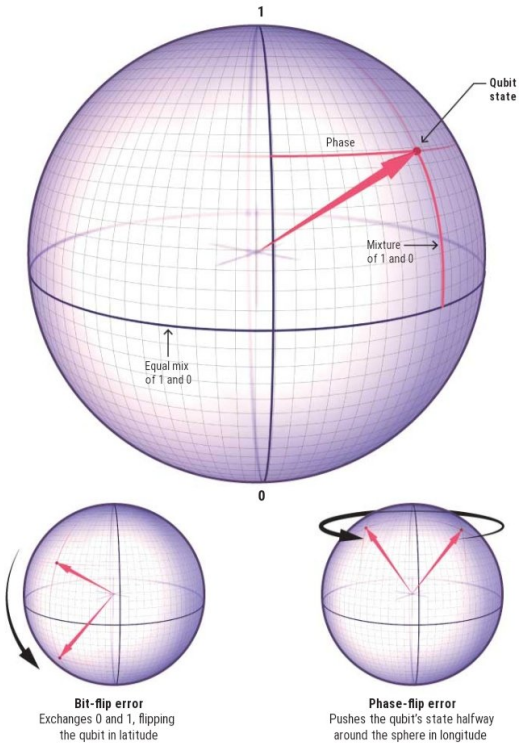

Figure 2.2: *Qubit state, bit-flip error and phase-flip error*

- **Small errors**
  Besides decoherence, other errors of order ε may occur. The quantum gates that the computer executes are unitary transformations that operate on a few qubits at a time. If we call U_0 the theoretical transformation, the actual one will be: $U = U_0 (1 + O(\varepsilon))$. Even though these errors are small, after $\frac{1}{\varepsilon}$ gates are executed, they accumulate and end up being one big error.

- **No disturbance, no measurement**
  As anticipated, measurement causes disturbance and makes quantum information, encoded among the qubits, fall.

- **No cloning**
  In classical error correction coding, the strategy is to make extra copies of the information to avoid it to be ruined definitely. But in QEC this strategy fails because quantum information can't be copied with the same fidelity.

2.2 The 3-qubit code

The most simple quantum error correcting code is the 3-qubit code [17, 23, 18, 25]. Suppose we have a noisy communication channel (no real channel is noise-free) through which a source A wants to transmit quantum information to a receiver B. But how does the noise act on the single qubit? The noise acts on each single qubit independently and its effect is chosen randomly between leaving the state of the qubit unchanged and applying a Pauli σ_x operator.

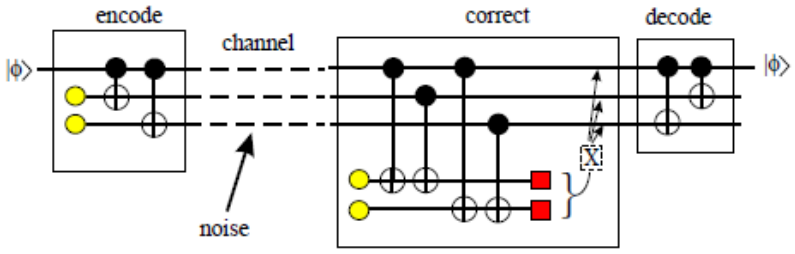

Figure 2.3: *3-qubit error correcting code scheme*

The simplest quantum error correction method is shown in the previous figure. By convention, the source is called Alice and the receiver is Bob. The state of any qubit that Alice wants to transmit is $a|0\rangle + b|1\rangle$. Alice prepares two further qubits in the state $|0\rangle$, thus the initial state will be $a|000\rangle + b|000\rangle$. Alice now operates a C-NOT gate

from the first qubit to the second, producing $a|000\rangle + b|110\rangle$. Then, another C-NOT gate is operated from the first qubit to the third giving the result $a|000\rangle + b|111\rangle$. At this point, Alice sends all three qubits down the channel. Bob receives the three qubits, but the noise of the channel has acted on them. The qubits can be in one of eight possible states, each one with some probability.

state	probability
$a\,\|000\rangle + b\,\|111\rangle$	$(1-p)^3$
$a\,\|100\rangle + b\,\|011\rangle$	$p(1-p)^2$
$a\,\|010\rangle + b\,\|101\rangle$	$p(1-p)^2$
$a\,\|001\rangle + b\,\|110\rangle$	$p(1-p)^2$
$a\,\|110\rangle + b\,\|001\rangle$	$p^2(1-p)$
$a\,\|101\rangle + b\,\|010\rangle$	$p^2(1-p)$
$a\,\|011\rangle + b\,\|100\rangle$	$p^2(1-p)$
$a\,\|111\rangle + b\,\|000\rangle$	p^3

At this stage, Bob introduces two more qubits, referred to as *ancilla qubits*, prepared in the state $|00\rangle$. Their function is to gather information about the noise. The receiver operates C-NOTs from the first and second received qubits to the first ancilla and then from the first and third received qubits to the second ancilla. The total state of all five qubits is given by eight possible states, each one with some probability.

state	probability
$(a\,\|000\rangle + b\,\|111\rangle)\,\|00\rangle$	$(1-p)^3$
$(a\,\|100\rangle + b\,\|011\rangle)\,\|11\rangle$	$p(1-p)^2$
$(a\,\|010\rangle + b\,\|101\rangle)\,\|10\rangle$	$p(1-p)^2$
$(a\,\|001\rangle + b\,\|110\rangle)\,\|01\rangle$	$p(1-p)^2$
$(a\,\|110\rangle + b\,\|001\rangle)\,\|01\rangle$	$p^2(1-p)$
$(a\,\|101\rangle + b\,\|010\rangle)\,\|10\rangle$	$p^2(1-p)$
$(a\,\|011\rangle + b\,\|100\rangle)\,\|11\rangle$	$p^2(1-p)$
$(a\,\|111\rangle + b\,\|000\rangle)\,\|00\rangle$	p^3

Finally, Bob measures the two ancilla qubits in the basis $\{|0\rangle, |1\rangle\}$ and he gets two classical bits of information. The latter is called the

error syndrome which is helpful in diagnosing the error in the received qubits and consequently correct it . The follow-up action depends on the measured syndrome. If this one is 00 no action will be made. If the syndrome is $01, 10, 11$, then Bob will apply σ_x to third, second and first qubit respectively. For instance, if measurements give 10, the state of the received qubits must be either $a|010\rangle + b|101\rangle$ with probability $p\,(1-p)^2$ or $a|101\rangle + b|010\rangle$ with probability $p^2\,(1-p)$. Here Bob applies a σ_x to the second qubit in order to correct the state. Thus, Bob obtains either $a|000\rangle + b|111\rangle$ which is more likely, or $a|111\rangle + b|000\rangle$. Finally, to extract the qubit sent by Alice, the receiver applies a C-NOT gate from the first qubit to the second and third obtaining either $(a|0\rangle + b|1\rangle)|00\rangle$ or $(a|1\rangle + b|0\rangle)|00\rangle$. The method has a probability greater than $1-p$ to succeed, even if the receiver doesn't know if he has the very same qubit that has been sent or this qubit operated on by σ_x. The failure probability coincide with the probability that at least two qubits are damaged by the noise and it is given by $3p^2 - 2p^3$. To sum up, Alice transmits a single qubit by expressing its state as a unified state of three qubits. As Bob receives the three qubits, he first applies error correction, then extrapolates a single qubit state. Incidentally, using more qubits, but still following the same ideas just shown, it is possible to achieve a by far better deletion of the noise along the channel. The error probability is reduced by a factor $\sim \frac{1}{3}p$ using three times as many qubits.

2.3 The 9-qubit code

The 9-qubit error correcting code [10, 25, 15, 6] is based on the repetition of the 3-qubit code and is the first full quantum error correcting code. It was firstly introduced by Shor and his simple idea was to use nine

physical qubits to encode one logical qubit.This idea is shown in the following picture.

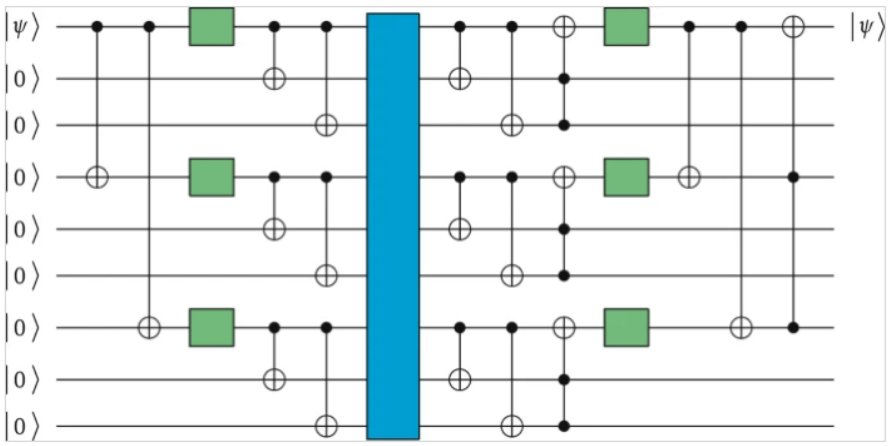

Figure 2.4: *9-qubit error correcting code scheme*

It is worth pointing out briefly that a *physical qubit* is two-state quantum system device, practically implemented as a component of a quantum processor. A *logical qubit* is a programming unit that logically performs the quantum algorithm and consists of one or more physical qubits. Since, in practice, the 9-qubit code would require a large number of physical qubits, later on other codes were invented, some of which used five physical qubits instead. Thereafter, the codes started to become more sophisticated introducing other qubits whose function is to protect the logical qubit.

The 9-qubit code is a single error correcting code and can recover any of the nine qubits from either one bit-flip, one phase error or both at the same time. Bit-flips and phase errors require different correction circuits to be detected and canceled.

The correction circuit for bit-flips is identical to the 3-qubit code, applied to each block of three qubits. Whilst the approach to correct phase errors consists of an analysis of the sign differences between the

three blocks. The 9-qubit code is degenerate, this means that different types of error on any qubit in a block of three have the same effect.

2.4 What about noise?

Digitization of noise [22, 23, 6] is based on the fact that any interaction between a set of qubits and another system can be expressed in the following form:

$$|\phi\rangle|\psi_0\rangle_e \rightarrow \sum_i \left(E_i|\phi\rangle\right)|\psi_i\rangle_e,$$

where E_i, the error operator, is a tensor product of Pauli operators acting on the qubits, $|\phi\rangle$ is the initial state of the qubits and $|\psi_i\rangle_e$ are states of the environment. Thus, general noise and decoherence are expressed in terms of Pauli operators $\sigma_x, \sigma_y, \sigma_z$ acting on the qubits, written as $X \equiv \sigma_x, Z \equiv \sigma_z, Y \equiv -i\sigma_y = XZ$. It emerges that correction of X and Z errors is sufficient to correct the most general possible noise expressed in the last formula. Noise usually affects all qubits all the time. The probability that an error occurs is equivalent to the probability that the syndrome extraction projects the state on another state that differs from the noise-free state by error operator. If we re-express the interaction between a system of qubits and its environment as $H_I = \sum_i E_i \bigotimes H_i^e$, we can calculate this probability. We can divide H_I up into a sum of terms having error operators of different weight. When only weight 1 terms are present, it means that the environment acts on the qubits independently not producing correlated errors directly. QEC recovers the terms in the density matrix with errors of weight $\leq t = (d-1)/2$. So, the fidelity of the corrected state, in the uncorrelated noise model, can be one minus the

probability for the noise to generate an error of weight $t+1$, i.e. one minus $P(t+1)$. Thus, QEC works very well when t is large. Another case in which QEC is successful is when a set of correlated errors, named *burst errors*, influence the system-environment coupling, but there is a stabilizer including all these correlated errors. This is rather error avoiding, because no correction is made on the logical state. A combined approach is nice in practice. Hence, first step is discovering correlated contributions of the noise in the system and outline a first layer of encoding. Second step is putting on a second layer optimized for minimum-distance coding. The recursive process of encoding one bit in several is called *code concatenation*.

2.5 Error operators, stabilizer, syndrome extraction

The set $\{I, X, Y, Z\}$ includes the identity $\mathbb{I}$ and the three Pauli operators. Error operators are, in this section, products of Pauli operators and may play the role of an error or a parity check. For n qubits in the system, the length of the error operators will be n. Whilst, the weight of an error operator is the number of terms not equal to $\mathbb{I}$. Let's consider a set of commuting error operators, thus having simultaneous eigenstates, and refer to it as $H = \{M\}$. Let $C = \{|u\rangle\}$ be an orthonormal set of simultaneous eigenstates with eigenvalue +1. So we have $M|u\rangle = |u\rangle \ \forall u \in C, \ \forall M \in H$. The set C is a quantum error correcting code and H is called its *stabilizer*. The states $|u\rangle$ are code vectors. Using stabilizer formalism to depict quantum error correction codes is very convenient and consists of describing quantum states in terms of operators rather than states. A state $|\psi\rangle$ is said stabilized by some operators K if it is a +1 eigenstate of K and,

furthermore, each stabilizer squares to the identity. In the case H is a group, C has 2^k members that form a 2^k dimensional subspace of the 2^n dimensional Hilbert space of the system. Thus, C encodes k qubits. An *encoded state* or *logical state* is a general state of this subspace and is given by $|\phi\rangle_L = \sum_{u \in C} a_u |u\rangle$. Each quantum error correcting code (QECC) only corrects a set of *correctable errors*. Error operators in the stabilizer are all correctable and if they are the only terms in the noise we are analyzing, then the QECC is said a noise-free subspace. The set of correctable errors can be called S and it can be any set of errors $\{E_i\}$ such that whatever product of two elements $E_1 E_2$ is still in H, or at least commutes with an element of H. Eventually, in order to extract the syndrome, all the observable in the stabilizer are to be measured, actually measuring any set of $n-k$ linearly independent M in H. In this way a projection of a noisy state onto an eigenstate of each M is operated with eigenvalue ± 1. The syndrome is the string of $n-k$ eigenvalues. E_1 and E_2 have different syndromes so that they are distinguishable. Finally the error can be derived from the syndrome and then recovered applying the error to the system again. Practically speaking, the extraction can be done adding an $n-k$ ancilla qubit to the system and storing it in the eigenvalues using a sequence of CNOTs and Hadamard rotations. How to disentangle the system from its environment and restore the initial state? Well, supposing that the errors have all different syndromes and making a projective measurement of the ancilla, the result will simply be the specific syndrome s_i. E_i is, then, easily found and re-applied to the system, because there is only one error with syndrome extracted. Thereby, the ancilla is a control qubit on which the noise is reversed in order to free the system up. But, in reality is not always true that the errors have different syndromes. Hence, it is necessary correcting only one error among a certain set of errors having same syndrome. Only one exception arises: $E_1 E_2 \in H$, namely two errors with same syndrome, so both correctable.

2.6 How to construct a code?

The stabilizer is completely described by any $n-k$ linearly independent error operators and its elements all commute. The whole stabilizer is defined by:

$$H\,(H_x|H_z)$$

where H_x and H_z are $(n-k)\times 2n$ binary matrices. Here we recall some important definitions. The number of non-zero components of a binary vector is called the *weight* and the number of coordinates by which two vectors differ is called the *Hamming distance*. Taken any two vectors u and v, their distance is given by $weight\,(u+v)$. Let's introduce the efficient so-called CSS (Calderbank, Shor, Steane) codes. Named G the set of error operators generated by the generator matrix G (the matrix whose k rows are any k vectors building the space) and containing H, we report an observation. If all the members of G have at least a weight d (minimum distance of the code), all error operators weighting less than d anticommute with a member of H and are recognisable. As a result, the code construction is equivalent to finding binary matrices H satisfying $H_xG_z^T + H_zG_x^T = 0$ and for which the generator matrices G have large weights. This last condition coincides with $H_xG_z^T + H_zG_x^T = 0$. So, now we can construct a code combining classical binary error correcting codes:

$$H = \begin{pmatrix} H_2 & 0 \\ 0 & H_1 \end{pmatrix}$$

$$G = \begin{pmatrix} G_1 & 0 \\ 0 & G_2 \end{pmatrix}$$

where H_i is the check matrix. Hence, $H_iG_i^T = 0$ and the large weight condition is satisfied. Secondly, to satisfy commutativity we state that $H_1H_2^T = 0$.

At this point of the construction, given the codes parameters $[n, k_1, d_1]$ and $[n, k_2, d_2]$, the quantum code size is $k = k_1 + k_2 - n$.

2.7 Is spacetime a quantum error correcting code?

What has been said so far could surprisingly be a reading key to better know space-time, woven on a network of quantum particles. Two ideas from mid-1990's are maybe closely related: holographic principle and quantum error correction [14]. Although quantum error correction codes are still being improved, evidence of a close connection between quantum error correction and the fabric of space, time and gravity has been found. According to general relativity, gravity is an effect of space-time curvature, but there may be something more. Gravity can have a quantum origin, from which then depends the structure of space-time. This evidence emerges from the work on a toy universe called a *Anti-de Sitter space* working like an hologram. The idea is: geometry emerges from entanglement. Physical variables of a quantum code reside on the boundary, while logical operators reside in the bulk. Thus, local operators in the bulk are linked to highly nonlocal operators located on the boundary.

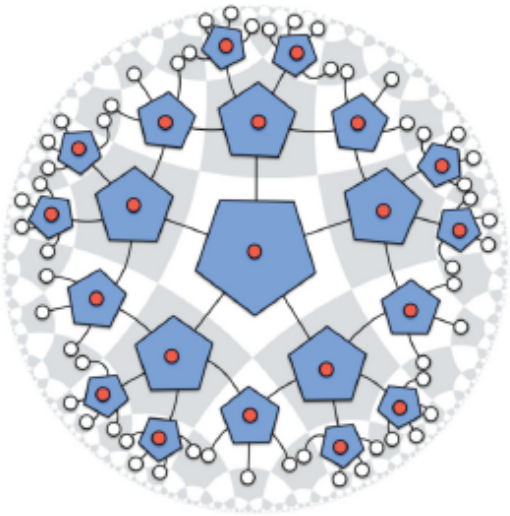

Figure 2.5: *Toy Model for bulk/boundary correspondence*

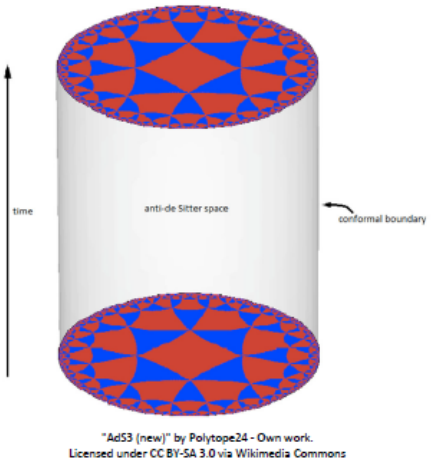

Figure 2.6: *AdS/CFT correspondence*

The holographic quantum error correcting codes provide toy models to know space-time deeper. So, exactly like an hologram, the structure

of space-time inside the universe comes up to be a projection of the entangled quantum particles located on the boundary of the universe. Calculations have been made suggesting that the holographic nature of space-time works like a quantum error-correcting code. Hence, at least is holographic universes, space-time is a QECC. The point now would be finding the code that space-time implements. Notice that Anti-de Sitter (AdS) space is characterized by a negative vacuum energy. As a consequence, the spacial dimension gradually get smaller and smaller, until disappearing, as we go from the center of the AdS space to its outer boundary. It has been noticed that any point inside the AdS space could be built as a quantum error-correcting code. For example simple code consists of three *qutrits*, particles existing in any of three states, placed at equidistant points around a circle. The qutrits are entangled and encode one logical qutrit. Stated that this qutrit is a single space-time point, the code protects this point against the erasure of any of the qubits.

Lately, another holographic code has been found called *the HaPPY code* (Harlow, Preskill, Pastawski, Yoshida) that is more explanatory than a single point can be. With this code the space is a connection of five-sided building blocks, where each block is a space-time point. To approximately visualize this, the role of the blocks can be compared to that of a single element in an Escher tiling [24, 14], as in the case of the fish in the artwork shown in the following figure.

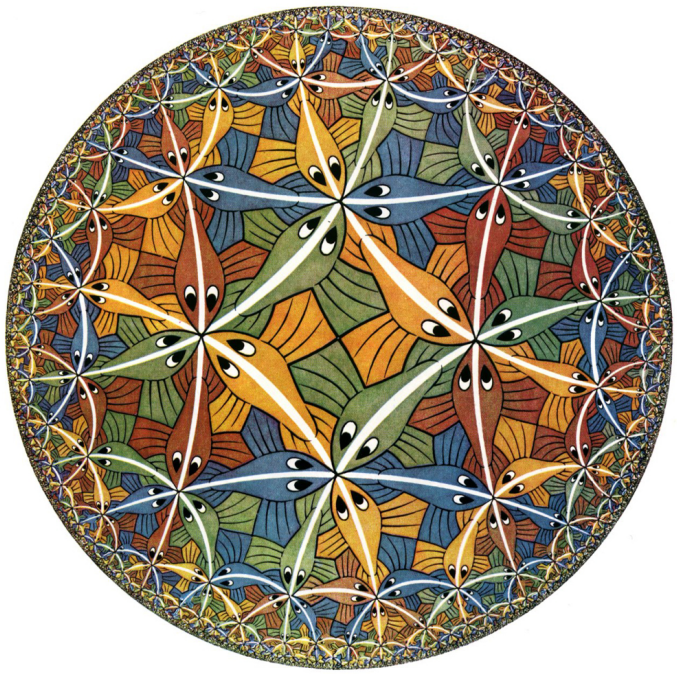

Figure 2.7: *The hyperbolic geometry, Circle Limit III, Escher 1959*

The interior of space-time is called *entanglement wedge* and is crucial because, everything belonging to this region can be reconstructed starting from the boundary adjacent qubits. Nevertheless, our universe is a de Sitter (dS) space, this means that it has a positive vacuum energy and no boundary; these features make it far more complex then the AdS space, which anyway has many properties in common with a dS space. Both spaces respect Einstein's theory, they only curve in different directions. A common feature of these two universes are black holes and this is why understanding which code space-time implements could be useful to learn more about black hole's interior.

Chapter 3

Algorithms and Applications

3.1 A qubit in the making

As introduced in the first chapter, the qubits are the building blocks of a Quantum Computer, determining its superior computing power. Practically speaking, there are various objects that can be used as qubits, for example an electron, a single photon, trapped ions, neutral atoms, quantum dots. For instance, the quantum computers the user interacts with in IBM Quantum Experience use a physical qubit called *superconducting transmon qubit* made of superconducting materials such as Aluminum and Niobium and put on a silicon substrate. This artificial qubit is built by isolating two levels of energy, the difference between which sets the characteristic frequency f of the qubit as:

$$\Delta E = hf \tag{3.1}$$

generally around 5 GHz.

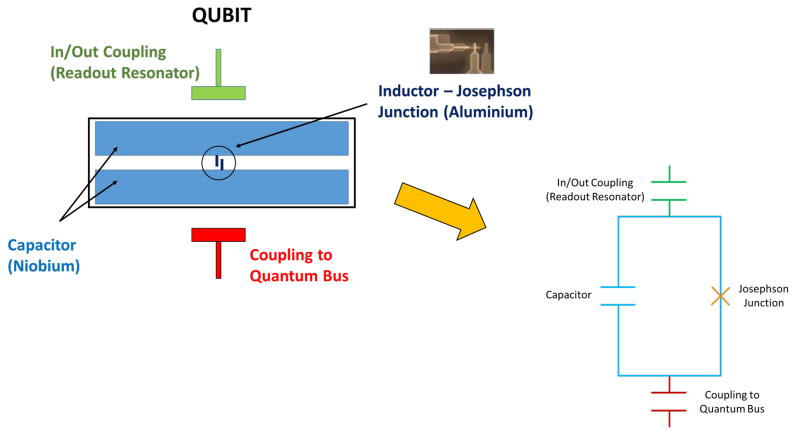

Figure 3.1: *Qubit scheme [21] - two pieces of superconductor are very weakly coupled. In practice, two superconducting electrodes are separated by a thin barrier which is an insulating layer. This Josephson Junction combined with a capacitor made out of superconducting material, act like a qubit.*

All the superconducting circuits share a Josephson Junction which is a circuit element that behaves as a non-linear and non-dissipative inductor.

3.2 How does a Quantum Computer work?

Let's take an electron as a qubit.The spin property of the electron makes it tend to align with a magnetic field in which the electron is placed. Now, the electron is in the lowest energy state, spin down ($|0\rangle$), and it will take some energy to bring it to the highest energy state, spin up ($|1\rangle$). Although this case is basically similar to the classical bit, what's new is the fact that if we measure the spin it will be either up or down, but before the measurement the electron is in a *quantum*

superposition. A quantum computer can process exponentially many logical states at once. The coefficient of the basis states is the relative probability of finding the electron in one state or the other. Thus, for two qubits, there are four basis states, which classical bits can be in, given by $|00\rangle, |10\rangle, |01\rangle, |11\rangle$, but infinitely many states formed by superpositions or combinations of the basis states. Each operation is performed by a quantum gate that changes the state of the qubits. Just for example, applying a quantum gate to $|00\rangle$, we can find the superposition $\sqrt{\frac{1}{2}} \cdot |01\rangle + \sqrt{\frac{1}{2}} \cdot |10\rangle$ and this means that there is a 50% probability of being in $|01\rangle$ state and the same for $|10\rangle$. In general, the superposition emerging depends on the gate we use. One qubit is represented as 2-dimensional vectors, two qubits as 4-dimensional vectors and so on. In particular, these 4-dimensional vectors point to a spot on the unit sphere in 4-dimensional space. If we have N qubits, then there are 2^N basis states and the quantum state is a vector on a sphere in a 2^N- dimensional space. So, changing the state of a system is equivalent to moving the state vector around the sphere. Each quantum gate is a different unitary matrix. Quantum computing is potentially powerful because it only takes few steps, hence little time, to find a state, while a classical computer would have taken thousands. This is due to the fact that classical computer can only be in a basis state at a time , while quantum computers process the algorithm overall, namely many classical states in parallel. But a superposition is not measurable because once we measure the qubits they fall into a basis state and the information about the preexistence of the state is lost. So, the final result has to be something measurable and once the wave function has collapsed, what we observe is a single basis state. Note that quantum computers are not a replacement of classical computers because they are not faster at everything.

3.3 IBM Quantum Experience

IBM Quantum Experience is an online platform that offers access via the cloud to IBM's quantum computers. This platform constantly evolves and is frequently improved and updated. The processors are located in a dilution refrigerator. The latter is needed because for quantum effects to be evident, quantum computer have to be cooled down to very low temperature such that $k_B T \ll hf$. The one shown in the following picture is know as *chandelier* and it's a supercharged refrigerator that gets colder with each layer down.

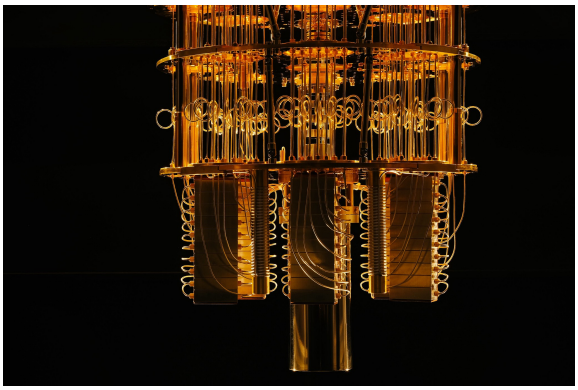

Figure 3.2: *The chandelier - IBM dilution refrigerator*

Its action is necessary for, as we have seen earlier, qubits need to be efficiently isolated from noise (heat, vibration or stray atoms). Thanks to IBM Quantum Experience it is possible from a classical terminal to develop quantum programs and use both quantum circuits and classical computations. The core of the experience is an intuitive drag and drop interface to built, inspect and visualize in different ways quantum circuits. Then, one can run the circuits on real IBM quantum systems or simulators. Furthermore, it is also possible to write a code using IBM Quantum Lab which is a cloud-enabled Jupyter notebook

environment.The user can also write programs in the OpenQASM - language. Among the tools available to the user there is the Quantum Composer which is a graphic user interface (GUI) designed by IBM. Both quantum and classical operations are available to manipulate qubits, build circuits and run them on a simulator or real quantum hardware. Specifically a quantum gate is a reversible (unitary) operation applied to one or more qubits in order to change their states.

Figure 3.3: *Quantum Gates available on IBM Quantum Composer. The red one is the Hadamard gate and it is a π rotation about the X+Z axis and has the effect of changing computation basis from $|0\rangle$,$|1\rangle$ to $|+\rangle$,$|-\rangle$ and vice-versa. The dark blue are classical gates. Light blue ones are phase gates.Grey ones are non-unitary operations.Purple ones are quantum gates.*

There are different circuits to entangle qubits; for example, Bell test on two qubits demonstrates that measurements of an entangled state cannot be explained by any local theory and so that there are non-classical correlations. Furthermore, other ways to entangle three qubits are GHZ state and W state examples. In the following sections some experiments run on IBM Quantum Experience will be reported with results and comments. Visualizations show different views of how a state of qubits is affected by the operations and are taken from a

single-shot statevector simulator. The latter generates random result based on a seed which is the initial value written in the algorithm. Each time the circuit is closed and then re-opened the seed has a new value, thus the result of the run will be different. The simulator seed can be changed at will to observe the consequent changes.

3.4 Grover's algorithm

Grover's algorithm [5, 8, 9] is a quantum mechanical algorithm for fast database search and it can speed up an unstructured search problem quadratically. What does it mean? Well, imagine we have a list of N items including an item with a special feature, *the winner*, that we want to spot. In a classical computation it would take on average $N/2$ checks to find the winner, while on a quantum computer we can do it using $\sqrt{N}$ operations. This algorithm solves oracles that add a negative phase to the solution states. The oracle is a diagonal matrix, where the entry that corresponds to the marked item has a negative phase. It is used to find the input value x_0 of an oracle function $f(x)$ with $f(x_0) = 1$ and $f(x) = 0$ for all the other values of x. Thus, for N possible input values, the algorithm only takes $O\left(\sqrt{N}\right)$ estimates of the oracle. The algorithm can be divided into five steps:

1. **Initialization**
   The qubits are set in $|0\rangle$ state and arranged to be in superposition using Hadamard gate. This gate is used to put a qubit into a superposition of $|1\rangle$ and $|0\rangle$ such that when one measures the qubit it will be $|1\rangle$ or $|0\rangle$ with the same probability. The states all have amplitude $\frac{1}{\sqrt{N}}$;

2. **Oracle**
   The oracle function marks the searched item using phase gates and, in general, recognizes solutions based on a given problem. It performs a phase flip which inverts the amplitude of the state turning it into $-\frac{1}{4}$;

3. **Amplification circuit**
   Also called *diffuser* or *amplitude purification*, it has the function of increasing the amplitude of the winner and reducing that of the other items;

4. **Measurement**
   The qubits are measured by a gate that projects the qubit's state onto the basis $|0\rangle$ and $|1\rangle$. This stage is necessary to extract a result computation.

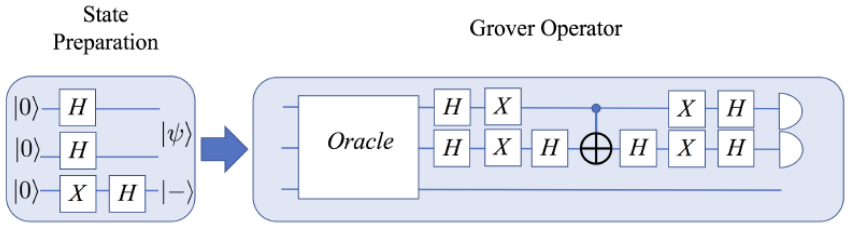

Figure 3.4: *Grover's Algorithm structure*

The oracle function can be also encoded in an operator, namely a series of quantum gates, and negates the probability amplitude of the input $|x\rangle$ if and only if $f(x) = 1$. This means that the oracle circuit flips the sign of an ancilla qubit if and only if the input is a solution to the problem. Furthermore, the internal structure of the oracle is not that important since it only has the function of spotting the solutions [3, 10]. The ensemble of the oracle and the diffuser is also referred to as *Grover operator*.

Notations

- The Hadamard gate H can be defined as the matrix $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ or as the states $|0\rangle \rightarrow \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)$ and $|1\rangle \rightarrow \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)$

Figure 3.5: *H gate*

- X is the classical NOT gate, or Pauli X matrix, defined as $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Figure 3.6: *NOT gate*

- Z is the phase shift gate, or Pauli Z matrix defined as $X = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

Figure 3.7: *Z gate*

- CNOT is the controlled NOT gate which applies a NOT on the second bit (the target) if the first bit (the control) is 1. This gate

is represented by the matrix $CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$

Figure 3.8: *CNOT gate*

A series of generic examples of Grover's algorithm are reported above with circuits, results and comments. Note that in the results histogram, the vertical axis labels the measurement probabilities, while the horizontal axis labels the computational basis states. The probabilities are related to the amplitudes α_i of the computational basis states $|i\rangle$ according to:

$$P_i = |\alpha_i|^2 . \tag{3.2}$$

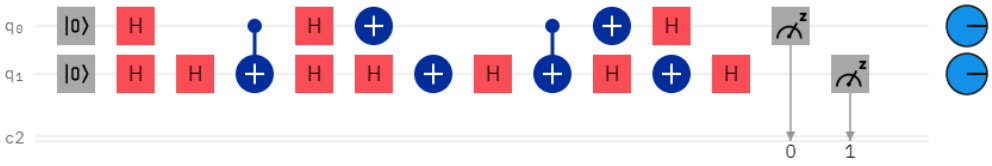

Figure 3.9: *2-qubit Grover's Algorithm, example 1*

Figure 3.10: *Initialization graphs: measurement probabilities, q-sphere, state vector*

This algorithm has been run many times and, as shown in the following sample histograms, results are always consistent with expectation: at each run there is always a very high probability of getting the marked state.

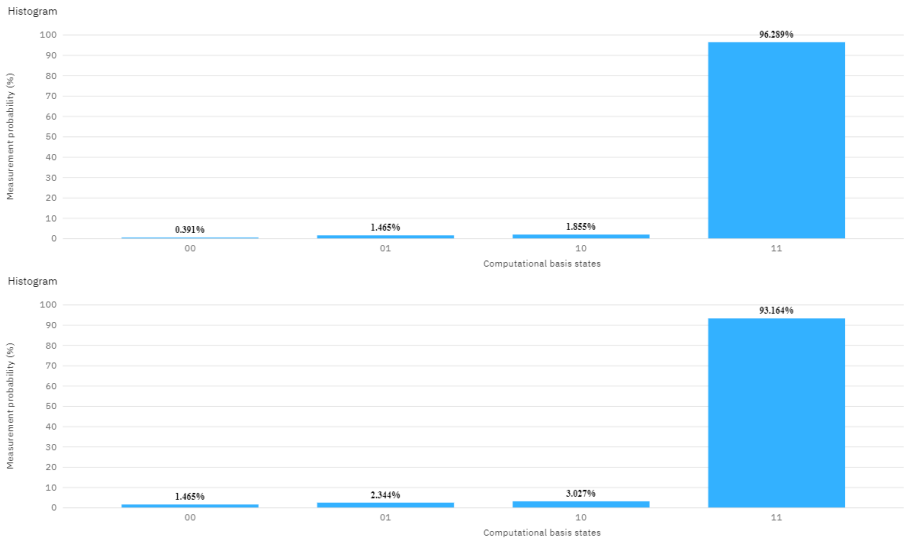

Figure 3.11: *Sample results histograms of 2-qubit Grover's Algorithm example 1*

Hereafter, another example of 2-qubits Grover's Algorithm that has been tested on IBM Quantum Experience. It still confirms what said about the previous one.

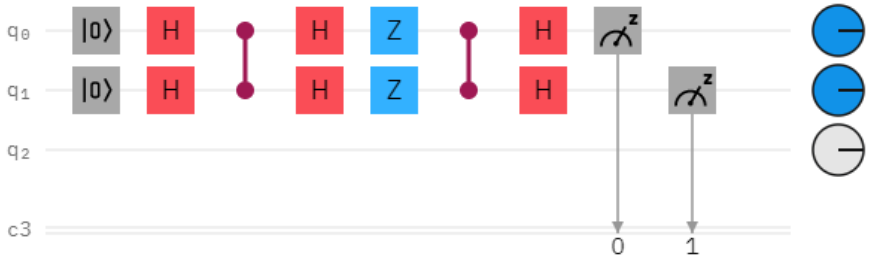

Figure 3.12: *2-qubit Grover's Algorithm, example 2*

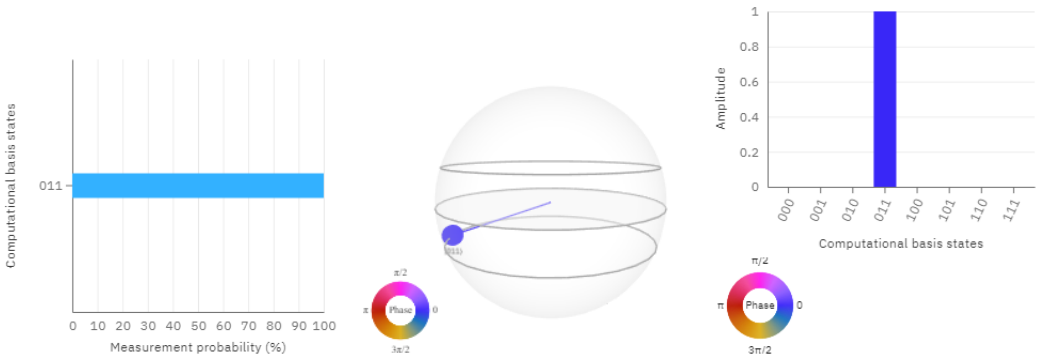

Figure 3.13: *Initialization graphs: measurement probabilities, q-sphere, state vector*

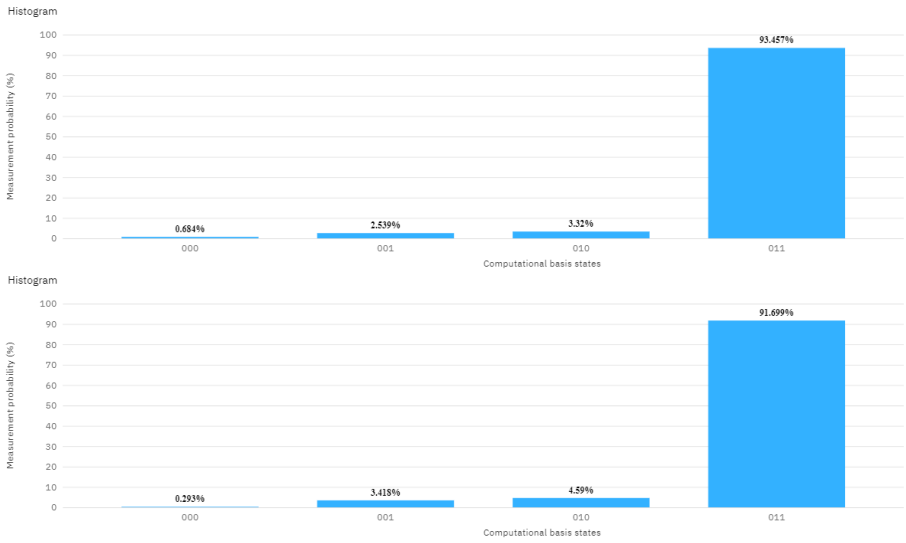

Figure 3.14: *Sample results histograms of 2-qubit Grover's Algorithm example 2*

It is possible to implement all kinds of oracle functions managing phase gates depending on what one is looking for. In the cases above, the circuits aim to find the winner state $|11\rangle$. This algorithm has been run many times and, as shown in the sample results histograms, in both examples, results are consistent with expectations: at each run on a real device the probability of getting the state $|11\rangle$ is considerably high and satisfactory compared to other states. Thus, in the majority of the iterations that state hidden in the oracle is found by the algorithm. The little probabilities on the other states are due to errors, but for the present purpose they are negligible. If we run the same circuit using simulators, we will get 100% chance of measuring the state $|11\rangle$. Below is reported one among the various 3-qubit Grover's algorithms tested on IBM Quantum Experience. Still the stages are the same described above. In this specific case the oracle operator negates the amplitude of the input if and only if the input is in the state $|110\rangle$. It jumps out that as the number of qubits increases, the errors get bigger. So, even

though the results clearly indicate the specific state requested in the oracle, the probability on that state is significantly reduced compared with the previous results and this fact highlights the need for more iterations.
In the following 3-qubit Grover's algorithm circuits, the states marked by the oracle are chosen to be $|011\rangle$ and $|110\rangle$, but other combinations have been tested as well. Notice that the barriers have been put in such a way as to make the execution order of the circuit unequivocal. Additionally, the CCZ gate has been recreated in two different but analogous ways because it was not possible to simply apply Z plus the single controls on the other two qubits in the circuit composer.

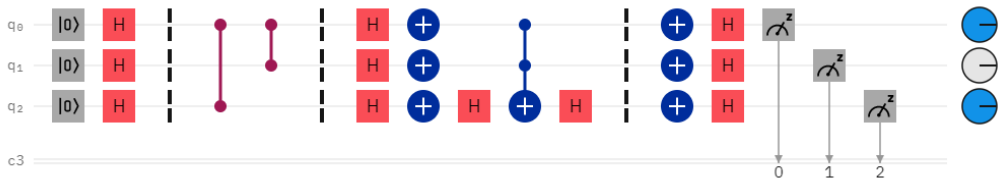

Figure 3.15: *3-qubit Grover's Algorithm example 1*

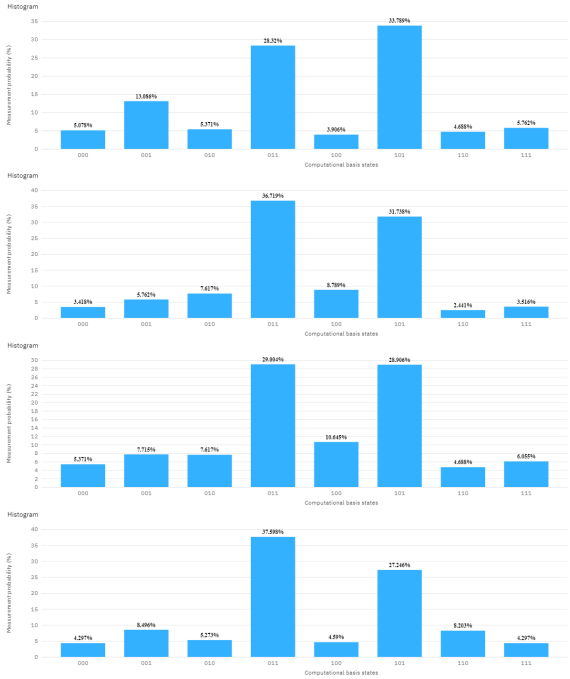

Figure 3.16: *Sample results histograms of the 3-qubit Grover's Algorithm example 1*

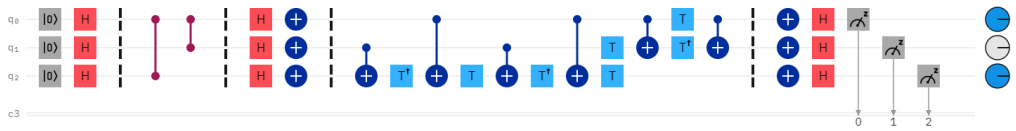

Figure 3.17: *3-qubit Grover's Algorithm example 2*

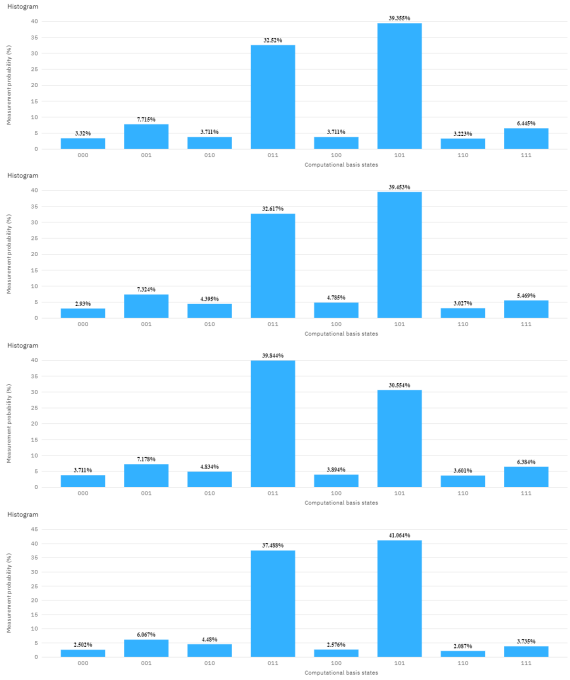

Figure 3.18: *Sample results histograms of the 3-qubit Grover's Algorithm example 2*

As expected, there is a higher chance of measuring the marked states $|110\rangle$ and $|011\rangle$. As seen before, the other results are linked to errors in the quantum computation.

3.5 The 3-qubit code

A good introduction to QEC is the 3-qubit code, as seen in section 2. This is not a complete quantum code because it cannot correct bit-flip and phase flip simultaneously, but a repetition of it creates the 9-qubit quantum code Shor used to demonstrate the consistence of quantum error correction. The 3-qubit code encodes a single logical qubit into

three physical qubits and it can correct for a single bit-flip error. The two basis states are defined as $|0\rangle = |000\rangle$ and $|1\rangle = |111\rangle$. So, first of all we prepare the $|0\rangle$ state for the 3-qubit code. An arbitrary single qubit state q_0 is coupled to two ancilla qubits q_1 and q_2, using CX gate, to prepare a logical state. The implementation of the 3-qubit code on the circuit composer looks like:

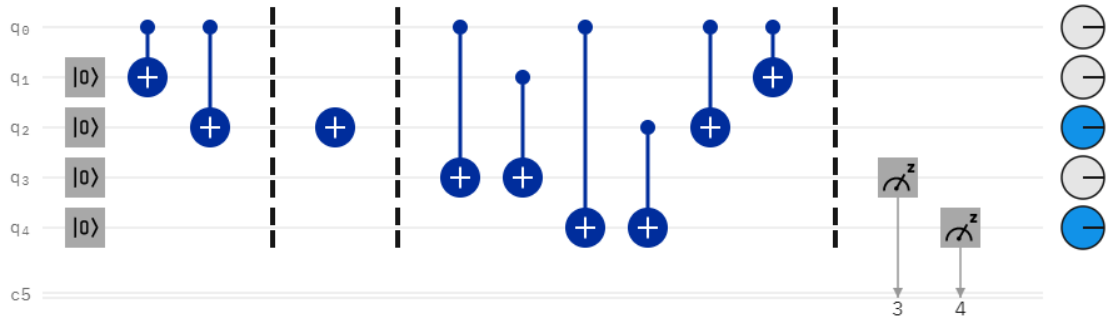

Figure 3.19: *3-qubit code with a bit flip on q_2 and ancilla qubits measurement*

The circuit above shows the stages of the 3-qubit code: encoding, eventual bit-flip, error detection and correction, measurement of the ancilla qubits. This measurement makes continuous errors discrete.

Why is this 3-qubit code correcting only one bit-flip error? The point is that the distance between the states is binary. In fact three bit-flips are necessary to make up the problem $|0\rangle \leftrightarrow |1\rangle$. The number of error that can be corrected is given by the formula $n = [(d-1)/2]$ where d is the distance between two states. Here $d = 3$, hence $n = 1$. At this point, we need to correct the error making no measurements on the logical states. To do this, we add two more initialized ancilla qubits q_3 and q_4 to the previous code and we use them to highlight syndrome information without being specific about the single qubits. Let's imagine that either a single bit-flip occurs on one of the three

qubits or none and that an error can only happen between encoding and correction steps. Then we apply a sequence of CX gates to check the parity between the qubits. After that, the q_3 and q_4 ancilla qubits are measured and from the result we can understand if and where an error has taken place. In this experience, run times varied from a few minutes to a couple of hours at most, depending on the queue and the changes being made to the IBM processors at the time of the run. Since only a single error can occur, the system can be restored because the resulting state is still closer to the right logical state. The code fails when more than one error occurs, so our wrong interpretation would lead to a wrong correction.

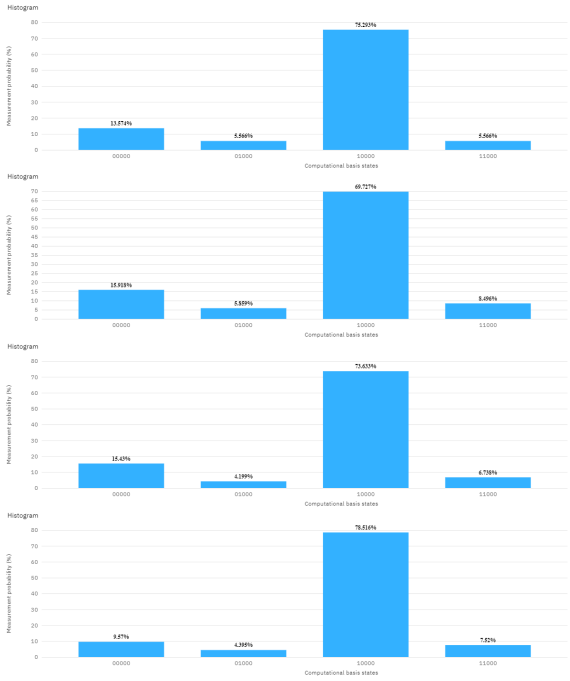

Figure 3.20: *Sample results histograms of the 3-qubit code*

A 3-qubit code is sufficient to protect against a single bit-flip. But, as quantum bits are subject to both bit-flips and phase flips, the 3-qubit

clusters are then repeated three times in order to protect also against phase errors [16]. The idea is that if a phase error occurs on one of the nine qubits, it is possible to detect the affected cluster measuring the two 6-qubit observables $X_1X_2X_3X_4X_5X_6$ and $X_4X_5X_6X_7X_8X_9$. The effect of the phase error on a particular cluster is to change its value relative to the other three clusters. Once the cluster has been identified, the error is corrected by applying a Z gate to one of the qubits in that specific cluster. The 3-qubit code acts first measuring a product of X's and then measuring the ancilla qubit in the $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$ basis. A single error occurring on one of the nine qubits causes a reversible damage, but two bit-flips in a single cluster lead to a misunderstanding in the diagnosis, thus will cause irreversible damage. Moreover, the effect of two bit-flips in a single cluster is a phase error in the logical qubit. Still the information is damaged if phase error occurs in two different clusters. That is why a phase error in the third cluster is introduced in order to recover from the errors globally. Different codes to correct both bit-flips and phase flips errors have been studied. Just for instance, CSS (Calderbank, Shor, Steane) codes have the special property of dividing the recovery procedure into two different stages one to correct bit-flips and the other to correct phase flips.

The 3-qubit code is based on the idea of correcting without decoding by just locating the flipped qubit and flip it back.

Furthermore, if one qubit flips the phase it is possible to find it and then correct it by applying a Z gate. But, when both bit-flip and phase flip occur at the same time, the 3-qubit code is no longer satisfactory. We would need the 9-qubit Shor's code as protection from such errors.

Conclusions

The aim pursued in this study was to introduce quantum computing and particularly the subject of quantum error correction in order to investigate why its development is so relevant. It has been shown that in the real implementation of a quantum computer, errors must always be taken into account as a natural part of the information transmission process and, generally, when executing an algorithm.
Consequently, the search for methods of error correction is the basis for a practical implementation of the theory of quantum computing to be effective. Finally, in a more application-oriented section, a report of the IBM Quantum Experience was made, showing some of the algorithms implementations made, with graphs and run results. Through simple examples and by means of real IBM quantum processors it was possible to practically interface with the object of study and obtain proof of the efficacy of some algorithms or, on the contrary, of their unreliability under certain conditions. In conclusion, henceforth, further insights into quantum computing, with particular attention to strategies for quantum error correction, will have to be done, in order to fully exploit the theoretical potential of a quantum computer. At the same time, one can focus even more on the possible applications of QEC and, also, investigate its analogies with other fields of knowledge.

Bibliography

[1] Charles H Bennett and David P DiVincenzo. Quantum information and computation. *nature*, 404(6775):247–255, 2000.

[2] Charles H Bennett, David P DiVincenzo, John A Smolin, and William K Wootters. Mixed-state entanglement and quantum error correction. *Physical Review A*, 54(5):3824, 1996.

[3] Eva Borbely. Grover search algorithm. *arXiv preprint arXiv:0705.4171*, 2007.

[4] A Robert Calderbank and Peter W Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54(2):1098, 1996.

[5] Patrick J Coles, Stephan Eidenbenz, Scott Pakin, Adetokunbo Adedoyin, John Ambrosiano, Petr Anisimov, William Casper, Gopinath Chennupati, Carleton Coffrin, Hristo Djidjev, et al. Quantum algorithm implementations for beginners. *arXiv*, pages arXiv–1804, 2018.

[6] Simon J Devitt, William J Munro, and Kae Nemoto. Quantum error correction for beginners. *Reports on Progress in Physics*, 76(7):076001, 2013.

[7] Jay M Gambetta, Jerry M Chow, and Matthias Steffen. Building

logical qubits in a superconducting quantum computing system. *npj Quantum Information*, 3(1):1–7, 2017.

[8] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[9] Lov K Grover. Quantum mechanics helps in searching for a needle in a haystack. *Physical review letters*, 79(2):325, 1997.

[10] Phillip Kaye, Raymond Laflamme, Michele Mosca, et al. *An introduction to quantum computing*. Oxford University Press on Demand, 2007.

[11] Emanuel Knill and Raymond Laflamme. Theory of quantum error-correcting codes. *Physical Review A*, 55(2):900, 1997.

[12] Emanuel Knill, Raymond Laflamme, Rudy Martinez, and Camille Negrevergne. Benchmarking quantum computers: The five-qubit error correcting code. *Physical Review Letters*, 86(25):5811, 2001.

[13] Raymond Laflamme, Cesar Miquel, Juan Pablo Paz, and Wojciech Hubert Zurek. Perfect quantum error correcting code. *Physical Review Letters*, 77(1):198, 1996.

[14] Fernando Pastawski, Beni Yoshida, Daniel Harlow, and John Preskill. Holographic quantum error-correcting codes: Toy models for the bulk/boundary correspondence. *Journal of High Energy Physics*, 2015(6):149, 2015.

[15] J Preskill. Chapter 5 of his lecture notes: Lecture notes on quantum information theory.

[16] J Preskill. Lecture notes for physics 229: Chapter 7: Quantum error correction. *California Institute of Technology*, 1999.

[17] J Preskill. Course information for physics 219/computer science 219 quantum computation, 2000.

[18] Matthew D Reed, Leonardo DiCarlo, Simon E Nigg, Luyan Sun, Luigi Frunzio, Steven M Girvin, and Robert J Schoelkopf. Realization of three-qubit quantum error correction with superconducting circuits. *Nature*, 482(7385):382–385, 2012.

[19] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee, 1994.

[20] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[21] IBM Research Editorial Staff. What's in a qubit? *IBM Reasearch Blog*, 2017.

[22] Andrew Steane. Quantum computing. *Reports on Progress in Physics*, 61(2):117, 1998.

[23] Andrew M Steane. A tutorial on quantum error correction. In *PROCEEDINGS-INTERNATIONAL SCHOOL OF PHYSICS ENRICO FERMI*, volume 162, page 1. IOS Press; Ohmsha; 1999, 2007.

[24] Natalie Wolchover. How space and time could be a quantum error-correcting code. *Quanta magazine*, 2019.

[25] Bin Yan and Nikolai A Sinitsyn. Recovery of damaged information and the out-of-time-ordered correlators. *Physical Review Letters*, 125(4):040605, 2020.