University of Trento
Department of Physics

Thesis
for the degree in
Bachelor of Science

Quantum Error Correction

Supervisor:
Prof. Philipp Hans Juergen Hauke,
University of Trento

Candidate:
Alexander Ferraro

Academic Year 2020/2021

Contents

Chapter 1

Introduction on quantum computing

Introduction on quantum computing In this chapter, I will present some basic concepts of quantum computing, and they will be useful for the following chapters, where I will instead discuss the process for correcting quantum errors. In the first section, I will discuss the fundamental unit of quantum information: the Qubit. The second section deals with how we can manipulate qubits through quantum logic gates. The third section instead discusses the density matrix, which is a useful tool for representing mixed states. In the end, the fourth section is a description of what noise and decoherence of a state are. Last but not least, the concept of fidelity is presented, a measure that indicates how similar the two states are.

1.1 Qubits

Classical computers and classical information rely on the concept of bit. The bit is the fundamental unit of classical information, i.e. the smallest portion into which any encoded information can be broken down; it is, therefore, the unit of measurement of encoded classical information.

Just as the bit is the smallest amount of information in classical computation, quantum computation is based on an analogous concept: the quantum bit or qubit.

A classical bit's state can be either 0 or 1, it can be turned off or turned on. The difference between bits and qubits is that a qubit can be in a state other than $|0\rangle$ or $|1\rangle$ (we use the bracket notation to identify the computational basis). Indeed, if we are dealing with qubits it is also possible to form linear combinations of states, often called

superpositions:

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

The numbers α and β are complex numbers and are called amplitudes. However, when we measure a qubit we get either the result 0, with probability $|\alpha|^2$, or the result 1, with probability $|\beta|^2$. Naturally, $|\alpha|^2+|\beta|^2= 1$, since the probabilities of all the possible outcomes must sum to one.

In other words, a qubit can be represented by a vector in the two-dimensional complex Hilbert space $\mathbb{C}^2$, where $|0\rangle$ and $|1\rangle$ are known as computational basis states and form an orthonormal basis for this vector space. Up to a global (and unphysical) phase factor, any single qubit can be expressed as follow:

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

where θ, φ are real numbers. The numbers θ and φ define a point on the unit three-dimensional sphere, as shown in 1.1.

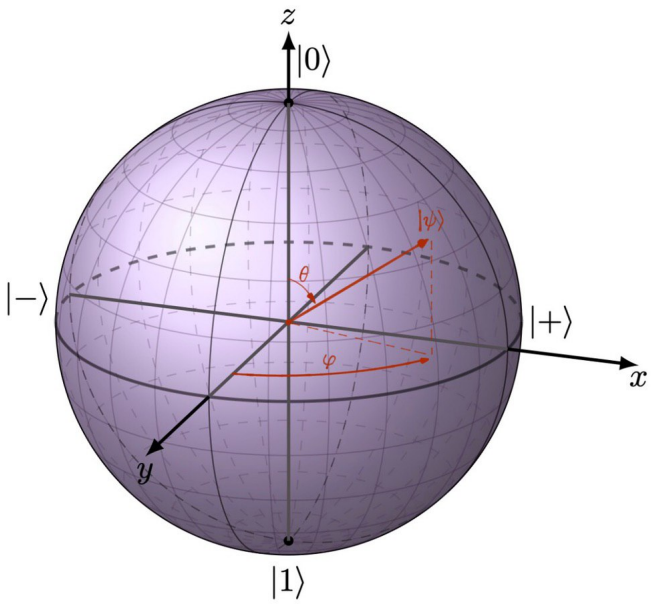


Figure 1.1: Bloch sphere

This sphere is called the Bloch sphere and provides a useful means of visualizing the state of a single qubit, and often serves as an excellent test for ideas about quantum computation and quantum information. Many of the operations can be done on a single qubit, through logic gates, and they can be described within the Bloch sphere picture.

However, it must be kept in mind that this intuition is limited because there is no simple generalization of the Bloch sphere known for multiple qubits.

A system of multiple qubits is represented by tensor-product Hilbert spaces; for example, a system composed of two qubits has a state $|\psi\rangle$ belonging to the tensor product of the Hilbert spaces of the two individual qubits, and we say that $|\psi\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2$. However, in this and in even larger spaces, there is a fundamental and physically crucial difference between the two kinds of states. There are the so called factorizable states, which can be expressed in the form of tensor product:

$$|\psi\rangle = (\alpha_1|0\rangle_1 + \beta_1|1\rangle_1) \otimes [\ldots] \otimes (\alpha_n|0\rangle_n + \beta_n|1\rangle_n)$$

and there are also the entangled states which are "un-factorizable", i.e the quantum state cannot be factorised as a product of states of its local constituents. A perfect example can be provided by considering a two qubits system; a factorizable state is:

$$|\psi_f\rangle = (\alpha_1\alpha_2|00\rangle + \alpha_1\beta_2|01\rangle + \beta_1\alpha_2|10\rangle + \beta_1\beta_2|11\rangle) = (\alpha_1|0\rangle + \beta_1|1\rangle) \otimes (\alpha_2|0\rangle + \beta_2|1\rangle)$$

while an example of entangled state could be:

$$|\psi_e\rangle = \alpha|00\rangle + \beta|11\rangle.$$

Qubit states can be manipulated and transformed in ways that lead to measurement outcomes that depend distinctly on the different properties of the state. However, in the end, we cannot measure a qubit in its state of superposition but we can manipulate the amplitudes to get the result which we are looking for. A quantum computer can create vast multidimensional spaces in which to represent very large problems, solve the problem using the properties of quantum mechanics, some logic gates, and operations. One of the techniques consists in adjusting the amplitude contributions, in fact, if some amplitudes are positive and others are negative, then the contributions can interfere destructively and cancel each other out. Likewise, they can interfere constructively and increase the likelihood of a given outcome. The goal in devising an algorithm for a quantum computer is to choreograph a pattern of constructive and destructive interference so that for each wrong answer the contributions to its amplitude cancel each other out, whereas for the right answer the contributions reinforce each other. If you can arrange that, one will see the right answer with a large probability when one measures. The tricky part is to do this without knowing the answer in advance, and faster than you could do it with a classical computer.

Another important question is: how much information can be carried by a qubit? Paradoxically, a qubit can exist in a continuum of states between the two bases, so there are an infinite number of linear combinations of the orthonormal basis apparently allowing, at least in principle, the representation of all human knowledge in a single qubit.

However, this is an erroneous conclusion by virtue of the behavior of the qubit during measurement. It must be kept in mind, in fact, that the outcome of the measurement of the state of a qubit can only be $|0\rangle$ or $|1\rangle$, with respective probabilities $|\alpha|^2$ and $|\beta|^2$. We can interpret this as the condition that the qubit's state is normalized to length 1. Moreover, the measurement of the qubit inexorably changes its state, collapsing the superposition state in one of the two specific states represented by the vectors of the computational basis as prescribed by the postulate of measure in quantum mechanics.

Thus, from the measurement of a qubit, it is possible to obtain the same amount of information that can be represented by a classical bit. This result has been proved rigorously by Holevo's Theorem.

There is currently no preferred qubit technology, a variety of physical systems are being explored for use as qubits: the two different polarizations of a photon, as the alignment of a nuclear spin in a uniform magnetic field... In contrast, bits in a classical computer are typically realized as the robust on/off states of transistor switches which are differentiated by billions of electrons. A shortcoming shared by all of these approaches to realize a quantum computer is that it is difficult to sufficiently isolate the qubits from the effects of external noise, meaning errors during quantum computation are inevitable, and quantum error correction codes have been developed in order to protect a qubit from the external noise. After this introduction in **??** I will describe how these codes work.

1.2 Quantum gates

Classically one can manipulate a system of bits and make some operations on it, using logic gates in order to solve a certain problem. Analogous to the way a classical computer is built from an electrical circuit containing wires and logic gates, a quantum computer is built from a quantum circuit containing wires and elementary quantum gates to carry around and manipulate the quantum information. A quantum gate can be represented as an operator acting on the states of one or more qubits, depending on the nature of the gate. A qubit can be represented as a two-dimensional ket. Thus, the operator algebra acting on those kets is 4-dimensional.

A quantum gate has to preserve the normalization condition of a state $|\psi\rangle$. Given

an operator $\hat{U}$ that transforms $|\psi\rangle \to |\psi'\rangle$, it follows:

$$\langle \psi' \mid \psi' \rangle = \langle \psi | \, \hat{U}^\dagger \hat{U} \, | \psi \rangle = 1$$

Hence,

$$\hat{U}^\dagger \hat{U} = \widehat{\mathbb{I}}$$

Operators satisfying this condition are called unitary operators. This imply that all quantum gates are also reversible, in fact, equation prove that $\hat{U^{-1}}$ exists and it is equal to $\hat{U}^\dagger$, thus:

$$\hat{U}^\dagger \, |\psi'\rangle = \hat{U}^\dagger \hat{U} |\psi\rangle = \hat{\mathbb{I}} |\psi\rangle = |\psi\rangle$$

If we consider a one qubit system, a suitable basis for these unitary operators acting on the system can be given by the identity matrix and the pauli matrices:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \sigma_X = X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \sigma_Y = Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad \sigma_Z = Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

All these matrices are unitary and also hermitian, which implies that $\sigma_i = \sigma_i^{-1} \quad \forall i = X, Y, Z$, where σ represent one of the 3 pauli matrices. Moreover, the Pauli matrices satisfy the following relations:

$$\begin{aligned} \sigma_i \sigma_j &= \delta_{ij} \mathbb{I} + i \epsilon_{ijk} \sigma_k \\ [\sigma_i, \sigma_j] &= 2i \epsilon_{ijk} \sigma_k \\ \{\sigma_i, \sigma_j\} &= 2\delta_{ij} \mathbb{I} \end{aligned}$$

where ϵ_{ijk} is the Levi-Civita symbol, $[\cdot, \cdot]$ is the commutator and $\{\cdot, \cdot\}$ is the anticommutator. These 4 matrices provide a basis of the operators acting on a single qubit. It follows that every single qubit operator can be expressed as a linear combination of I, X, Y, Z:

$$U = \alpha I + \beta X + \gamma Y + \delta Z \quad \text{with } \alpha, \beta, \gamma, \delta \text{ complex numbers.}$$

and a multi-qubit operator can be expressed as a tensor product of single qubit operators:

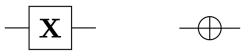
$$U = U_1 \otimes U_2 \otimes ... \otimes U_n.$$

Quantum gates are reversible by definition. Thus, the numbers of qubits in input must be the same as the qubits in output, otherwise, the gates are not reversible. An immediate

consequence of this property is, for instance, that the classical AND gate, largely used in classical computing, could not have an exact quantum analog. Among all the quantum gates having as an input a single qubit, the most remarkable for our purpose are the quantum NOT, the Z gate, the Hadamard gate, and the general phase shift gate. Instead, from all the gates that act on two qubits, the most important one in our context is the controlled-NOT gate. In what follows we briefly introduce these gates.

The quantum NOT gate

In quantum circuits, it is represented by:



Using the analogy with the classical NOT gate, which acts on the computational basis changing the state 0 in the state 1 and vice versa, one can suppose that the quantum not has the same action on the computational quantum basis, taking $|0\rangle \rightarrow |1\rangle$ and $|1\rangle \rightarrow |0\rangle$. It is interesting to see what happen to a state which is a superposition of the quantum basis:

$$NOT\,|\psi\rangle = NOT(\alpha\,|0\rangle + \beta\,|1\rangle) = (\alpha\,|1\rangle + \beta\,|0\rangle)$$

Thus, we can write down the "truth table", or better the action of the gate on the qubit:

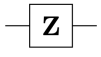
Input	Output
$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$

Table 1.1: Truth table quantum NOT gate

It is possible to notice that an operator which acts on a qubit like that is the X Pauli matrix, hence this gate it is also called X-gate.

The Z-gate

In quantum circuits, this gate it is represented by



. This gate is represented by the Z pauli matrix. The associated "truth table" is:

Input	Output
$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	-$\|1\rangle$

Table 1.2: Caption

This is a phase shifting in the space spanned by $|0\rangle$ and $|1\rangle$. However it can be understood as a bit-flip gate in other orthonormal basis states. For example, if we consider as computational basis $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ and $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$, with respect to this new base elements, the action of the Z-gate is:

Input	Output
$\|-\rangle$	$\|+\rangle$
$\|+\rangle$	$\|-\rangle$

Table 1.3: "Truth table" of Z-gate in the new basis

Another interesting observation that can be seen is the action of the X-gate on a generic state $|\psi\rangle = \alpha\,|+\rangle + \beta\,|-\rangle$ in this new basis. In this new base the X matrix is:

$$X_{|+\rangle,|-\rangle} = M^{-1}XM = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Hence, the action of X in the new basis is the same as that of operator Z in the old basis.

The general phase-shift gate

The Z gate is a particular case of the phase rotation gate, which corresponds to a phase shift of π. In quantum circuits, this gate it is represented by :

The phase-shift gate maps $|0\rangle \to |0\rangle$ and $|1\rangle \to e^{i\phi}\,|1\rangle$, where ϕ is the phase shift. It is possible to notice that if $\phi = \pi$. Thus, $e^{i\pi} = -1$ obtaining the Z-pauli matrix. The phase-shift ϕ can assume all the value from 0 to 2π.

The probability of measuring a $|0\rangle$ or $|1\rangle$ is unchanged after applying this gate, however it modifies the phase of the quantum state.

The associated truth table is:

Input	Output
$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$e^{i\phi}\|1\rangle$

Table 1.4: Truth table of R-gate

The phase shift gate is represented by the matrix:

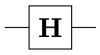
$$R_\phi = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}.$$

But, sometimes in literature a different convention is adopted. In the following thesis I will specify always which one I use. To get an example of the other convention is usually represented with the θ angle.

$$R_{\theta/2} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}.$$

The Hadamard gate

In quantum circuits, this gate is represented by:



The hadamard gate acts on a single qubit creating a superposition if a basis state is given. It maps $|0\rangle \to \frac{|0\rangle+|1\rangle}{\sqrt{2}} = |+\rangle$ and $|1\rangle \to \frac{|0\rangle-|1\rangle}{\sqrt{2}} = |-\rangle$. In particular, it corresponds exactly to the change of basis matrix from the $\{|0\rangle, |1\rangle\}$ to the $\{|+\rangle, |-\rangle\}$ bases and, thanks to the fact that it is self-reversible, also vice-versa. Its truth table is:

Input	Output
$\|0\rangle$	$\|+\rangle$
$\|1\rangle$	$\|-\rangle$

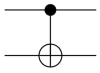
Table 1.5: Caption

And its matrix representation in the computational basis $|0\rangle, |1\rangle$ is:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

The controlled-NOT gate

The CNOT gate acts on 2 qubits inverting the second qubit $\oplus$ (the target qubit) if and only if the first qubit $\bullet$ (the control qubit) is $|1\rangle$. In quantum circuit, it is represented by:



By being binary its truth table traces 4 different possible input scenarios: and so

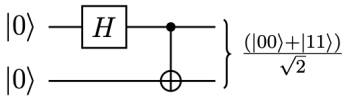
Input		Output	
Control	Target	Control	Target
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$

Table 1.6: Action of the CNOT gate

does the matrix representation which, this time, is based on a 4×4 matrix:

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

This gate is widely used to creates entangled states. As an example, consider this circuit:



where the final state is also known as $|\Phi^+\rangle$ Bell state. Starting from the second chapter, we are going to use these gates to build quantum circuits for quantum error correction codes. Instead, in the last chapter I will focus more on how to make these gates more reliable.

1.3 Density operator

The formalism for pure quantum states does not immediately allow for a formalisation of ignorance or missing knowledge about the state of a quantum system. Suppose that the mechanism that generates replicas of the same system does not always produce the same pure state but produces several of them. We do not exactly know in what state our system is, but instead we are given with some possible states $|\psi_\lambda\rangle$ with probability p_λ. The set (or ensemble) of replicas corresponds to a mixture of pure states. This mixture is called mixed state or mixed ensemble. In quantum mechanics, the density matrix operator is an useful tool to describe an ensemble of quantum states. It is a generalization of the more usual state vectors approach, which can only describe pure states. A pure state is by definition:

$$|\psi\rangle = \sum_j c_j |\psi_j\rangle \tag{1.1}$$

where the coefficients c_j are complex numbers $|\psi_j\rangle$ are the eigenvectors of an observable[1] that constitute a complete orthonormal basis in the Hilbert space for the states describing the system.

Instead, we call $\{p_\lambda, |\psi_\lambda\rangle\}$ an ensemble of pure state, where p_λ are real numbers that indicate the classical probability to pick up a pure state $|\psi_\lambda\rangle$ from the ensemble. Recall the fact that the probabilities p_λ are real numbers without a phase between them. While the coefficients of a decomposition are related by a phase.

Describing the state of a quantum system with such probabilistic mixtures opens up the possibility of having non-reversible evolution.

For instance, in the case of a qubit, we might start in a certain state $|\psi\rangle = a|\psi\rangle + b|\psi\rangle$ with unit probability, but if we measure the state the knowledge is lost or transferred to another register or ignored such that the final probabilistic mixture describing the qubit's state is $\{(p_0 = \frac{1}{2})||0\rangle, (p_1 = \frac{1}{2})||1\rangle\}$. That is, we have a uniform probability distribution over the computational basis states. Certainly, this evolution is not reversible as the final probabilistic mixture alone does not allow for a reconstruction of the initial state, because the same measurements results can be given by a mixture of states ($\{(p_0 = \frac{1}{2})||0\rangle, (p_1 = \frac{1}{2})||1\rangle\}$) or by a single state in a superposition ($|\psi\rangle = a|\psi\rangle + b|\psi\rangle$) [1].

In the light of the above, the density operator or density matrix of the system is

[1] for instance, in quantum computing the most common observable is the spin along the z axes S_z, which indicates the computational basis $|0\rangle$ and $|1\rangle$

defined:

$$\rho = \sum_{\lambda} p_\lambda \left|\psi_\lambda\right\rangle \left\langle\psi_\lambda\right| \tag{1.2}$$

with the constrain that $\sum_\lambda p_\lambda = 1$. If the system is composed of only one pure state, the density matrix is:

$$\rho = \left|\psi\right\rangle \left\langle\psi\right| \tag{1.3}$$

A way to estimate how pure a system is, one can compute the trace of ρ^2 since this is always less than 1. In fact:

$$Tr(\rho^2) = Tr(\rho^2_{diag}) = \sum_{\lambda} p_\lambda^2 \leq 1$$

If the probabilities p_λ are all zero except the one for some λ' for which $p_{\lambda'} = 1$, then the ensemble corresponds to a set of replicas of a system in the same pure state $|\psi_\lambda\rangle$. To evaluate the expectation value of a generic observable Q over this ensemble, we have to add up the expectation values we obtain for each pure state, weighing them with the probabilities p_λ:

$$\langle Q\rangle_{mix} = \sum_{\lambda} p_\lambda \left\langle\psi_\lambda|Q|\psi_\lambda\right\rangle = \sum_{\lambda} p_\lambda \sum_{i,j} c_i^{*(\lambda)} c_j^{(\lambda)} \left\langle a_i|Q|a_j\right\rangle$$

where the coefficients $c_j^{(p)} = \langle a_j \mid \psi_\lambda\rangle$ are those of the state decomposition $|\psi_\lambda\rangle$ on the basis of the eigenvectors of A (observable that characterises the system, for a qubit can be the spin along the z direction, which gives $|0\rangle$ and $|1\rangle$ as basis vector).

$$\begin{aligned}
\langle Q\rangle_{mix} &= \sum_{\lambda} p_\lambda \sum_{i,j} \left\langle a_i|Q|a_j\right\rangle \left\langle\psi_\lambda \mid a_i\right\rangle \left\langle a_j \mid \psi_\lambda\right\rangle \\
&= \sum_{i,j} \left\langle a_i|Q|a_j\right\rangle \left(\sum_{\lambda} p_\lambda \left\langle a_j \mid \psi_\lambda\right\rangle \left\langle\psi_\lambda \mid a_i\right\rangle\right) \\
&= tr(Q\rho)
\end{aligned}$$

Another advantage of working with the density matrix notation is that, when dealing with composite systems, for example system and environment, it provides a practical way to extract the state of each subsystem, even if they are entangled. This is done in the form of what is known as the reduced density matrix. Consider a quantum system composed of subsystems A and B, and fully described by the density matrix ρ_{AB}. The

reduced density matrix of subsystem A is then given by:

$$\rho_A = \mathrm{Tr}_B\left(\rho_{AB}\right)$$

Here, Tr_B is an operation known as the partial trace, which is defined as:

$$\mathrm{Tr}_B\left(|\psi_u\rangle\langle\psi_v| \otimes |\varphi_u\rangle\langle\varphi_v|\right) \equiv |\psi_u\rangle\langle\psi_v|\,\mathrm{Tr}\left(|\varphi_u\rangle\langle\varphi_v|\right)$$

$|\psi_u\rangle$ and $|\psi_v\rangle$ are arbitrary states in the subspace of A, and $|\varphi_u\rangle$ and $|\varphi_v\rangle$ arbitrary states in the subspace of B. Tr is the standard trace operation, which for two arbitrary states $\mathrm{Tr}\left(|\varphi_u\rangle\langle\varphi_v|\right) = \langle\varphi_v \mid \varphi_u\rangle$. As an example, let us reconsider the pure entangled state:

$$\left|\Phi^+_{AB}\right\rangle = \frac{1}{\sqrt{2}}\left(|0_A 0_B\rangle + |1_A 1_B\rangle\right)$$

This system is then composed of single-qubit subsystem A with basis vectors $\{|\psi_1\rangle, |\psi_2\rangle\} = \{|0_A\rangle, |1_A\rangle\}$, and single-qubit subsystem B with basis vectors $\{|\varphi_1\rangle, |\varphi_2\rangle\} = \{|0_B\rangle, |1_B\rangle\}$. We know that this system is not separable; however, by using the reduced density matrix, we can find a full description for subsystems A and B as follows.

The density matrix of our state $\left|\Phi^+_{AB}\right\rangle$ can be expressed in terms of outer products of the basis vectors:

$$\rho_{AB} = |\psi_{AB}\rangle\langle\psi_{AB}| = \frac{1}{2}\left[|0_A 0_B\rangle\langle 0_A 0_B| + |0_A 0_B\rangle\langle 1_A 1_B| + |1_A 1_B\rangle\langle 0_A 0_B| + |1_A 1_B\rangle\langle 1_A 1_B|\right]$$

. Then, for example, the reduced density matrix for the subsystem B is:

$$\begin{aligned}
\rho_B &= \mathrm{Tr}_A\left(\rho_{AB}\right) \\
&= \frac{1}{2}\left[\mathrm{Tr}_A\left(|0_A 0_B\rangle\langle 0_A 0_B|\right) + \mathrm{Tr}_A\left(|0_A 0_B\rangle\langle 1_A 1_B|\right) + \mathrm{Tr}_A\left(|1_A 1_B\rangle\langle 0_A 0_B|\right) + \mathrm{Tr}_A\left(|1_A 1_B\rangle\langle 1_A 1_B|\right)\right] \\
&= \frac{1}{2}[\mathrm{Tr}\left(|0_A\rangle\langle 0_A|\right)|0_B\rangle\langle 0_B| + \mathrm{Tr}\left(|0_A\rangle\langle 1_A|\right)|0_B\rangle\langle 1_B| + \mathrm{Tr}\left(|1_A\rangle\langle 0_A|\right)|1_B\rangle\langle 0_B| + \\
&+ \mathrm{Tr}\left(|1_A\rangle\langle 1_A|\right)|1_B\rangle\langle 1_B| \\
&= \frac{1}{2}\left[\langle 0_A \mid 0_A\rangle|0_B\rangle\langle 0_B| + \langle 1_A \mid 0_A\rangle|0_B\rangle\langle 1_B| + \langle 0_A \mid 1_A\rangle|1_B\rangle\langle 0_B| + \langle 1_A \mid 1_A\rangle|1_B\rangle\langle 1_B|\right] \\
&= \frac{1}{2}\left[|0_B\rangle\langle 0_B| + |1_B\rangle\langle 1_B|\right] = \frac{I}{2}
\end{aligned}$$

It is worth mentioning that so far we have described the concept of partial trace for a bipartite (two-party) system, but this can be generalized for multi-party systems.

Moreover, Using the density operator formalism we can successfully study the evolu-

tion of a closed quantum system, we will describe it better considering different type of noise and evolution in the next sections. The time evolution is usually described by the unitary operator $U(t, t_0)$. If the system was initially in the state $|\psi_\lambda\rangle$ with probability ψ_λ then after the evolution has occurred the system will be in the state $U|\psi_\lambda\rangle$ with the same classical probability . Thus, the evolution of the density operator is described by the equation:

$$\rho = \sum_\lambda p_\lambda U |\psi_\lambda\rangle \langle\psi_\lambda| U^\dagger = U\rho U^\dagger \tag{1.4}$$

The density operator approach really excels for two applications: the description of quantum systems whose state is not known, and the description of subsystems of a composite.

1.4 Quantum errors: Noise and Decoherence

The Schrödinger equation

$$i\hbar\frac{d|\psi\rangle}{dt} = H|\psi\rangle$$

describes the evolution of quantum systems in isolation, where $|\psi\rangle$ is the state vector. These closed systems have a well-defined Hamiltonian operator H, which gives complete information about how these systems evolve. The resulting evolution is unitary: the evolution of the state is given by a linear map $|\psi(0)\rangle \to |\psi(t)\rangle = U|\psi(0)\rangle$ where $U^\dagger U = UU^\dagger = I$. Note that these Hamiltonians may "come from outside" the system; for instance, we can turn external fields on and off, shine lasers, etc. What makes a quantum system closed is that it doesn't act back on the external world. The external fields, lasers, etc., can all be treated as classical potentials.

The unfortunate reality is that this idealization is fiction. All real quantum systems interact with the outside world, at least weakly; and the existence of interactions, which allow us to manipulate a system, also allows the system to interact with the external environment. This uncontrolled environmental interaction is called decoherence.

One can model the dynamics of a register of qubits with its surroundings. We imagine the system immersed into its environment (often called bath) and the whole (quantum register plus environment) as a closed system described in a general way by the following Hamiltonian:

$$H = H_S \otimes I_B + I_S \otimes H_B + H_{\mathcal{I}}$$

where $H_S\,(H_B)$ (the system (bath) Hamiltonian) acts on the system (bath) Hilbert space $\mathcal{H}_S\,(\mathcal{H}_B)\,, I_S\,(I_B)$ is the identity operator on the system (bath) Hilbert space, and H_I, which acts on both the system and bath Hilbert spaces $\mathcal{H}_S \otimes \mathcal{H}_B$, is the interaction Hamiltonian containing all the nontrivial couplings between system and bath. In general, $H_{\mathcal{I}}$ can be written as a sum of operators which act separately on the system (S_α) and on the bath (B_α) [2]:

$$H_{\mathcal{I}} = \sum_\alpha S_\alpha \otimes B_\alpha$$

In the absence of an interaction Hamiltonian ($H_{\mathcal{I}} = 0$), the evolution of the system and the bath are separately unitary: $\mathbf{U}(t) = \exp(-i\frac{H}{\hbar}t) = \exp\left(-i\frac{H_S}{\hbar}t\right) \otimes \exp\left(-i\frac{H_B}{\hbar}t\right)$. Information that has been encoded (mapped) into states of the system Hilbert space remains encoded in the system Hilbert space if $H_{\mathcal{I}} = 0$. However, in the case when the interaction Hamiltonian contains nontrivial couplings between the system and the environment ($H_{\mathcal{I}} \neq 0$), decoherence can happen.

Two things can cause decoherence. First, random influences from the outside can perturb the system's evolution, as if some random Hamiltonian was turned on, in addition to the usual Hamiltonian. Second, the interaction between the system and environment can cause information about the system to leak into the environment. This information leakage leaves the system correlated with the environment. In fact, a certain quantity of information that was previously carried by the system, then is spread out also in the environment. The No-hiding theorem shows how this property works. The effect on the system is as if unwanted measurements have been performed (without, in general, our knowing the measurement results). In fact, these two processes generally both occur, and their practical effects often look similar. Indeed, in quantum mechanics, there is no sharp distinction between them. If decoherence persists long enough, it is possible for all information about the original state of the system to be lost. In the shorter term, decoherence can destroy quantum effects such as interference and entanglement (on which quantum information processing depends).

In general, we can study the evolution of an isolated quantum systems subjected to decoherence and other sources of noise using quantum states, but as already said in the previous section, the state vector approach does not immediately allow for a formalisation of ignorance or missing knowledge. A more general description can be done using the density matrix formalism.

Maps that represent the evolution of the density operator must preserve certain properties: they are completely positive, trace-preserving and map density matrix into density matrix. Maps that satisfy these properties are called CPTP maps and can be written as:

$$\rho' = \mathcal{E}(\rho) \rightarrow \sum_{\mu} K_{\mu} \rho K_{\mu}^{\dagger} \quad \text{with} \sum_{\mu} K_{\mu}^{\dagger} K_{\mu} = I$$

where the K_μ are $N \times N$ matrices (N being the dimension of the Hilbert space) and are called Kraus operators. In general, the Kraus decomposition of a CPTP map is not unique, but one can approximately think of the map as the state $|\psi\rangle$ being multiplied by one of the operators K_μ chosen at random with probability $p_\mu = \langle\psi| K_\mu^\dagger K_\mu |\psi\rangle$. Since one does not know which operator has acted on the state, one uses a mixture of all of them.

Similarly, if an unknown influence is applied to the quantum system from the outside, we can model that as a set of unitaries $\{A_\mu\}$ that occur with respective probabilities $\{p_\mu\}$. Here, again, one would describe the state of the system as a mixture of all possible

evolved states:

$$\rho \to \rho' = \sum_{\mu} p_\mu A_\mu \rho A_\mu^\dagger, \quad \sum_{\mu} p_\mu = 1$$

In this case, again we have a CPTP map, and we can define the Kraus operators to be $K_\mu \equiv \sqrt{p_\mu} A_\mu$. CPTP maps give a unified description of all possible sources of Markovian noise[2], and in quantum information science one does not usually make a sharp distinction between different noise sources.

1.5 Fidelity

In communication problems, we would like to know how much information is preserved by some processes. We would like to compare the initial message and the final message after the effect of noise. A quantity that can represent the similarity between two states is fidelity. Hence, it is possible to use fidelity as an appropriate measure of the quality of a recovered code.

The fidelity can be written as the overlap between the final state ρ_f of a system ρ and the original state $|\psi\rangle$. If the combined operator consisting of an interaction with the environment (E_a) followed by a recovery operation ($\mathcal{R}_a$) is given by $\mathcal{A} = \{A_a = \mathcal{R}_a E_a, \ldots\}$[3], then:

$$\mathcal{F}\left(\rho_f, |\psi_i\rangle\right) = \langle\psi_i| \rho_f |\psi_i\rangle = \sum_a \langle\psi_i| A_a |\psi_i\rangle \langle\psi_i| A_a^\dagger |\psi_i\rangle .$$

It gives the probability that the final state would pass a test checking whether it agrees with the initial state [3]. As we are thinking of encoding arbitrary states, we do not know in advance the state that will be used. We, therefore, use the minimum fidelity (that is the worst-case fidelity)

$$\mathcal{F}_{\min} = \min_{|\psi\rangle} \langle\psi| \rho_f |\psi\rangle . \tag{1.5}$$

The best quantum code maximizes $\mathcal{F}_{\min}$.

A quantum communication channel can be used to transmit the state $|\psi\rangle$ from one

[2]Markovian dynamics means a limit or approximation in which the recent details do not matter, that is, the environment is "memoryless" and does not contain information about the history of the system. This is advantageous because it allows us to write an equation for $\rho(t)$ that depends only on $\rho(t)$, and not on, say, $\rho(t - t')$. This is called "time local" equation and is much easier to handle.

[3]The subscription a indicates a generic error E_a as kraus operator, followed by its recovery procedure $\mathcal{R}_a$. If we want to study the case where there is no recovery for that error we put $\mathcal{R}$ equal to the identity operator, getting only the action of error in the fidelity

location to another. No channel is ever perfect, so its action is described by a quantum operation $\mathcal{E}$ on $\rho = |\psi\rangle\langle\psi|$.

Let's take an example of a noisy channel: the depolarising channel denoted as $\mathcal{D}$. This channel leaves the qubit untouched with probability $1-p$ and with probability $\frac{p}{3}$ one of the 3 errors $\{X, Y, Z\}$ can act on the qubit:

$$\rho' = \mathcal{D}(\rho) = (1-p)\rho + \frac{p}{3}(X\rho X + Y\rho Y + Z\rho Z) \tag{1.6}$$

We can rewrite the expression as:

$$\begin{aligned}\rho' &= \frac{p}{3}(X\rho X + Y\rho Y + Z\rho Z) + (1-p)\rho \\ &= \frac{p}{3}(\rho + X\rho X + Y\rho Y + Z\rho Z) + \left(1 - \frac{4p}{3}\right)\rho.\end{aligned}$$

We can further simplify the expression more, using the following relation[4]:

$$\frac{1}{4}(\rho + X\rho X + Y\rho Y + Z\rho Z) = \frac{I}{2}.$$

Then, we can use this result and put it in the previous equations:

$$\begin{aligned}\rho' &= \frac{4p}{3}\frac{I}{2} + \left(1 - \frac{4p}{3}\right)\rho \\ &= \lambda\frac{I}{2} + (1-\lambda)\rho\end{aligned}.$$

Here, $\lambda = \frac{4p}{3} \in \left[0, \frac{4}{3}\right]$ is the depolarization parameter. Equivalently, the depolarizing channel can also be described as replacing the state with a completely mixed state with a chance of $\lambda = \frac{4p}{3}$. The effect of this error channel on the Bloch sphere (figure 1.2) can therefore be thought as an isotopical shrinking of the sphere.

From this we can calculate the fidelity between the initial and final mixed state:

$$\begin{aligned}\mathcal{F}(|\psi\rangle, \rho') &= \min_{|\psi\rangle} \langle\psi|\rho'|\psi\rangle \\ &= \langle\psi|\left(\lambda\frac{I}{2} + (1-\lambda)|\psi\rangle\langle\psi|\right)|\psi\rangle \\ &= \frac{\lambda}{2} + (1-\lambda) = 1 - \frac{2}{3}p.\end{aligned}$$

[4]It is straightforward to calculate that $\mathcal{E}(I) = I$ and $\mathcal{E}(X) = \mathcal{E}(Y) = \mathcal{E}(Z) = 0$. However, I, X, Y, Z form a basis of 2×2 matrices, so any ρ can be written as $\rho = aI + bX + cY + dZ$ and then $\mathcal{E}(\rho) = aI$. If ρ is a density matrix then $a = \frac{1}{2}$ and $\mathcal{E}(\rho) = I/2$. This means that $\mathcal{E}$ maps a pure state into a mixed state, for example $\mathcal{E}(|0\rangle\langle 0|) = \frac{1}{2}(|0\rangle\langle 0| + |1\rangle\langle 1|)$

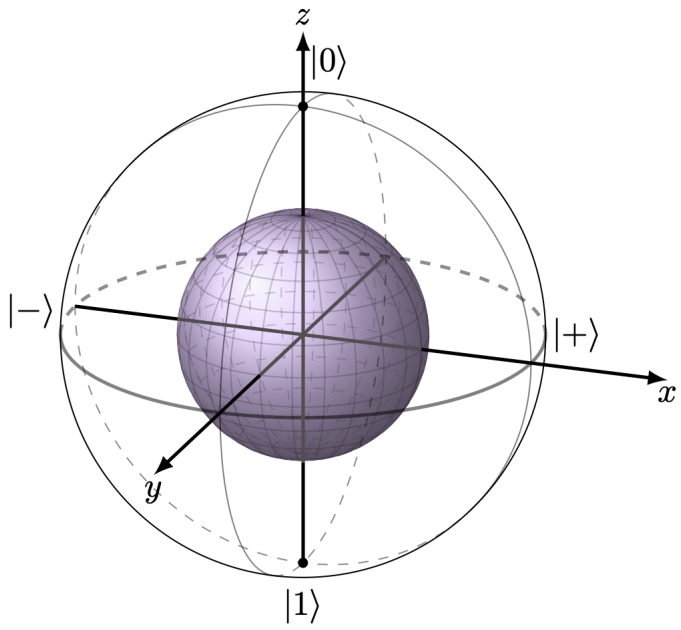


Figure 1.2: The depolarizing channel has the same probability for each type of error to occur. Consequently, the effect on the Bloch sphere is isotropic: the sphere is rescaled by a factor of $(1-p)$*. Since the depolarizing channel also has the interpretation of replacing a qubit with a completely mixed state, its outcome will always be an ensemble that is closer to the Bloch sphere's center than the original density operator.*

This result agrees with our intuition: the higher the probability p of depolarizing, the lower the fidelity is. For a very small p the fidelity is closer to one, and the state $\mathcal{E}(\rho)$ is practically indistinguishable from the initial state. More examples can be done using different channels can be studied in the same way.

Chapter 2

Quantum error correction codes

This chapter is the focus of my thesis, and aims to introduce the reader into the field of quantum error correction. In these sections several different procedures and some of the most important error correction codes will be described.

2.1 No-Cloning theorem

In classical communication we can copy a bit without any problem. Copying the information is one of the basic strategies for the classical error correction code: we increase the reliability of the code and then if a bit has been flipped, through a majority vote we can restore the information.

Unfortunately, in quantum mechanics, we are not allowed to copy an unknown quantum state. This is due to the No-Cloning theorem [4], which states that it is not possible to copy an arbitrary pure quantum state into another. The impossibility to copy an arbitrary quantum state means that there is no unitary transformation that can copy an arbitrary state from system A to the system B:

Theorem 1. *There is no unitary operator U on $\mathcal{H} \otimes \mathcal{H}$ such that for all normalised states $|\psi\rangle_A$ and $|s\rangle_B$ in $\mathcal{H}$*

$$U\left(|\psi\rangle_A|s\rangle_B\right) = e^{i\alpha(\psi,s)}|\psi\rangle_A|\psi\rangle_B,$$

for some real number α depending on ψ and s.

The no-cloning theorem states that it is not possible to perfectly clone the state but as shown in [5] it is possible to get an optimal copy of a state with certain fidelity. In addition, a different proof can be given showing that the No-cloning theorem works with

mixed states as well; in this case, the theorem is often known as the No-broadcasting theorem. The no-cloning theorem prevents the use of certain classical error correction techniques on quantum states. For example, backup copies of a state in the middle of quantum computation cannot be created and used for correcting subsequent errors. Hence, we cannot use the classical "copy and paste" tactic in quantum error correction codes to protect our information, but this does not mean that we have to get rid of the classical error correction theory; indeed it is very helpful to build reliable code with minimum effort, for instance, the CSS codes. Instead, in quantum error correction, another strategy is adopted: the information is not copied but is spread out on multiple qubits.

One of the simplest and emblematic quantum error-correcting codes that shows this property is the 3-qubit code, although it can correct only a bit flip (σ_X) error, it is still a good starting point for more complicated algorithms.

2.2 3-qubit error correction code

Just as every communication problem in literature typically starts with Alice and Bob I will do the same here.

Suppose that Alice wishes to transmit a qubit through a noisy channel to Bob, which leaves the qubit untouched with probability $1-p$, and flips the qubit with probability p introducing a bit-flip X error.

Before analysing the algorithm we need to make some assumptions about the noise[6]:

- it acts on each qubit independently,
- it is identically distributed on each qubit,
- the quantum gates in the encoding, recovery and decoding procedure are not subjected to errors,
- the error probability p has to be less than $\frac{1}{2}$ to be able to correct the error.

The quantum error correction method is summarised in figure 2.1.

The original quantum information that we want to protect can be written without loss of generality as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. As previously said, one might be tempted to clone the information like in the classical binary repetition code, but the no-cloning theorem prevents us from applying an operation that maps $|\psi\rangle \to |\psi\rangle|\psi\rangle|\psi\rangle$, showing that the classical idea cannot be directly taken over. Instead, we can encode our qubit

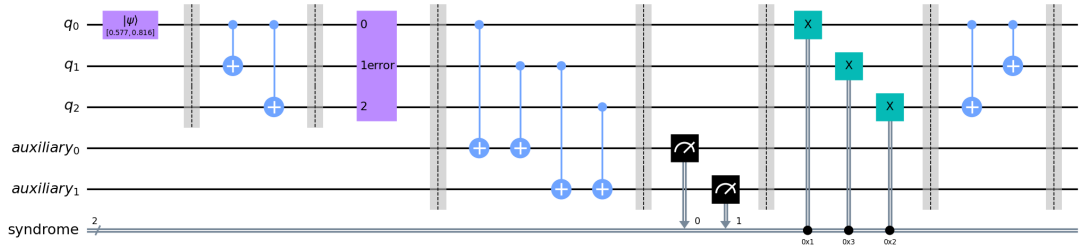


Figure 2.1: Circuit for the 3 bit-flip error correction code, built with CNOT gates, which allows for the correction of one X error on the encoded state. The thin horizontal lines are qubits, while the thick line indicates classical bits. All the different stages of the 3 bit-flip code are separated from grey barriers. From left to right: 1. Initialise the state, 2.Encoding procedure, 3.Error, 4. Syndrome measurement, 5. Store results in a classical register 6. Error correction, 7.Decoding.

with other two qubits initially in the state $|0\rangle$. First, we initialise the device in the state $|\psi\rangle \to (\alpha\,|0\rangle + \beta\,|1\rangle) \otimes |0\rangle \otimes |0\rangle$. Then we apply two CNOT gates as shown in figure 2.1, and encode the qubits in the following way:

$$\alpha\,|000\rangle + \beta\,|100\rangle \to \alpha\,|000\rangle + \beta\,|111\rangle := |\psi\rangle_L\,,$$

where we use the subscript L to denote that the final encoded state lives in the logical two-dimensional subspace spanned by $\{|0\rangle_L = |000\rangle\,, |1\rangle_L = |111\rangle\}$:

$$|\psi\rangle_L \in \mathcal{C} = \text{span}\{|000\rangle, |111\rangle\},$$

where $\mathcal{C}$ is called the codespace and its elements are the codewords.

Then the logical state $|\psi\rangle_L$ is sent through the channel, and an error may occur.

Notice that three individual bit flips are required to take $|0\rangle_L \leftrightarrow |1\rangle_L$. Hence if we assume $|\psi\rangle = |0\rangle_L$, a single bit-flip on any qubit leaves the final state closer to $|0\rangle_L$ than $|1\rangle_L$. One can quantify this observation using the code distance d, which defines the number of errors t that can be corrected, as $t = \lfloor \frac{d-1}{2} \rfloor$. In short, d represent the distance between two codewords, in this case, $d = 3$ because only a weight 3-error, such as $\tilde{X} = X^{\otimes 3}$, which maps $|0\rangle_L \to |1\rangle_L$. Using the previous notation a code with distance 3 can correct 1 error, but we will give a more in-depth explanation for the code distance and the number of correctable errors in section 2.5.

After the noisy channel, the outcome state is one of the following:

We can focus on the case where only one physical qubit has been flipped, and ignore all the errors that flipped 2 or more qubits; in fact, the code will not be able to correct all of them; the probability for them to occur is of higher-order $O(p^2)$.

State	probability
$\alpha\|000\rangle + \beta\|111\rangle$	$(1-p)^3$
$\alpha\|100\rangle + \beta\|011\rangle$	$p(1-p)^2$
$\alpha\|010\rangle + \beta\|101\rangle$	$p(1-p)^2$
$\alpha\|001\rangle + \beta\|110\rangle$	$p(1-p)^2$

Table 2.1: This table shows the possible state outcomes after the noise

Classically, single bit-flip errors are corrected by measuring the three bits and taking a majority vote. Here, this is not possible, since measuring the qubits would project the system into one of the basis states with probability $|\alpha|^2$ or $|\beta|^2$, destroying the superposition state that we are trying to protect.

There is a two stage error-correction procedure which can be used to recover the correct quantum state without loosing the information: detect the error and correct it.

In order to detect the error, we have to perform a measurement that reveals the error without revealing any information about the encoded state. Error detection measurements are called syndrome measurements and the outcomes are called syndromes.

In the quantum 3 bit-flip code one possible way to detect the error is to measure the correlation between two qubits, using two commuting observables Z_1Z_2 and Z_2Z_3 with eigenvalues ± 1.

As shown in figure 2.1 this involves the use of two more qubits, these are called ancilla qubits. The basic idea is to extract the syndrome performing two parity checks with the commuting observables Z_1Z_2 and Z_2Z_3 on the data block, and store the information in the ancilla qubits. The two ancillas are then coupled to the data block as shown in figure 2.1, using CNOT gates. We are left with:

Error Location	Final State, $\|data\rangle\|ancilla\rangle$
No Error	$(\alpha\|000\rangle + \beta\|111\rangle)\|00\rangle$
Qubit 1	$(\alpha\|100\rangle + \beta\|011\rangle)\|10\rangle$
Qubit 2	$(\alpha\|010\rangle + \beta\|101\rangle)\|11\rangle$
Qubit 3	$(\alpha\|001\rangle + \beta\|110\rangle)\|01\rangle$

Table 2.2: This table shows which error is associated with the ancillae

A measurement of the ancillas reveals where (or if) an error has occurred, without directly measuring any of the data qubits.

Then, we use the value of the error syndrome to identify what procedure to use to

recover the initial state: This bit-flip code has a correctable error set with four error

Syndrome measurement: (Z_1Z_2, Z_2Z_3)	Ancilla state	Correction
$(+1, +1)$	$\|00\rangle$	Clean state, no correction needed
$(-1, +1)$	$\|10\rangle$	Bit flip on qubit 1
$(-1, -1)$	$\|11\rangle$	Bit flip on qubit 2
$(+1, -1)$	$\|01\rangle$	Bit flip on qubit 3

Table 2.3: This table shows which error correction should be applied after a specific ancillae state and syndrome measurement.

operators: $\{I, X_1, X_2, X_3\}$. However, the full error set of this error model contained eight error operators. The three errors of weight 2 and one error of weight-3 are uncorrectable errors. They produce states in the same four sub-spaces above, and it is easy to see that in the case of those high-weight errors the correction procedure will produce the erroneous state $|\psi'_L\rangle = \alpha|111\rangle + \beta|000\rangle$. Moreover, the weight 3 error will not even be recognized as an error: it is an undetectable error. This is a general property of QECCs: no QECC can correct every possible error. (This is also true of classical error correcting codes.) In practice, the goal is to choose a code that can correct the most likely errors.

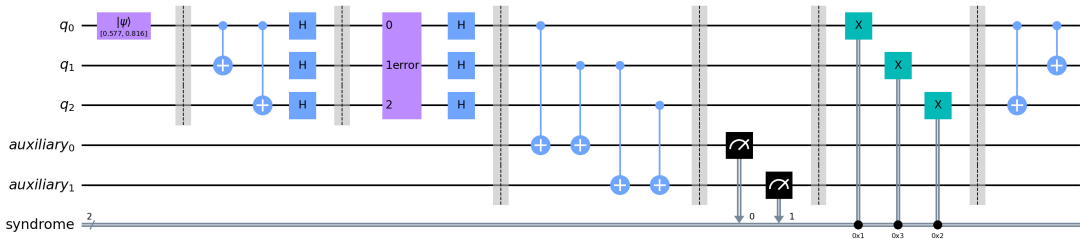


Figure 2.2: Circuit for the 3 phase flip error correction code, built with CNOT gates, which allows for the correction of one Z error on the encoded state. The thin horizontal lines are qubits, while the thick line indicates classical bits. All the different stages of the 3 bit-flip code are separated from grey barriers. From left to right: 1. Initialise the state, 2.Encoding procedure and change computational basis, 3.Error and immediately after applying the Hadamard gate we get back to the original computational basis, 4. Syndrome measurement, 5. Store results in a classical register 6. Error correction, 7.Decoding.

Similarly, we can implement also the phase flip error code. In fact, a phase flip error in the computational basis $(|0\rangle, |1\rangle)$ can be seen as a bit flip in a new set of basis $(|+\rangle, |-\rangle)$, as shown in section 1.2.

This suggests using the states $|+\rangle, |-\rangle$ to create the logical encoded qubits:

$$|0\rangle_L = |+++\rangle,$$
$$|1\rangle_L = |---\rangle,$$

these are the zero and the one states for protection against phase flip errors. To change basis in quantum circuit we can apply an Hadamard gate at appropriate points in the procedure. The 3 phase-flip circuit is shown in figure 2.2

We can improve our error analysis calculating the fidelity of this quantum error correction code, giving a measure of how reliable is this code against specific type of errors. To calculate the fidelity is better to use the density matrix formalism, recalling the fact that the fidelity between a pure and a mixed state is given by :$\mathcal{F}(|\psi\rangle, \rho) = \langle\psi|\rho|\psi\rangle$. The object of quantum error-correction is to increase the fidelity with which quantum information is stored (or communicated) up near the maximum possible fidelity of one. Let's compare the minimum fidelity achieved by the three qubit bit flip code with the fidelity when no error-correction is performed. Suppose the quantum state of interest is $|\psi\rangle$. Without using the error-correcting code the state of the qubit after being sent through the channel is:

$$\rho = (1-p)|\psi\rangle\langle\psi| + pX|\psi\rangle\langle\psi|X.$$

Then the fidelity is given by

$$F = \langle\psi|\rho|\psi\rangle = (1-p) + p\langle\psi|X|\psi\rangle\langle\psi|X|\psi\rangle.$$

Then the minimum fidelity is $F = 1-p$ (No error correction code applied). On the other hand, the state after both the noise and error-correction is:

$$\rho = \left[(1-p)^3 + 3p(1-p)^2\right]|\psi\rangle\langle\psi|.$$

The omitted terms represent contributions from bit flips on two or three qubits. All the omitted terms are positive operators, so the fidelity we calculate will be a lower bound on the true fidelity. We see that $F = \langle\psi|\rho|\psi\rangle \geq (1-p)^3 + 3p(1-p)^2$. Since the fidelity is at least $1 - 3p^2 + 2p^3$, so the fidelity is improved provided $p < 1/2$, as it is possible to see in figure 2.3.

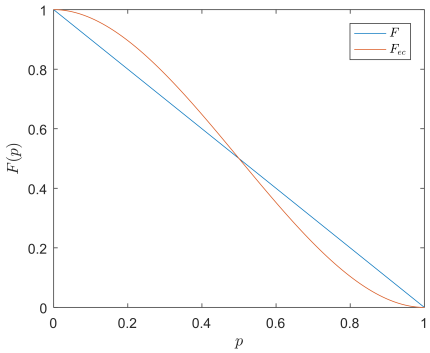


Figure 2.3: In red the fidelity with the error correction procedure, in blue the fidelity without error correction

2.2.1 Simulation of the 3 bit and phase flip code

In this subsection, I will show the computational results of the circuits showed in figures 2.1 and 2.2. I implemented these codes in python, in particularly using the qiskit library [7]. My codes can be consulted in my GitHub.

3 bit-flip code

First, I prepared the "information qubit" (q_0) in the initial state[1]:

$$|\psi\rangle = \frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle\,, \tag{2.1}$$

where the generic α is replaced by $\frac{1}{\sqrt{3}}$ and β by $\sqrt{\frac{2}{3}}$.

I encoded this qubit as shown in the circuit in figure 2.1 creating the encoded state: $|\psi\rangle = \frac{1}{\sqrt{3}}|000\rangle + \sqrt{\frac{2}{3}}|111\rangle$.

Then, a faulty gate that simulates the bit-flip error channel was created, in such a way that satisfies the noise conditions written at the beginning of section 2.2.

In addition, I set the error probability to $p = 0.1$, and ran the simulations with 10^5 shots [2].

To illustrate that the faulty gate acts as an error channel, projecting the encoded state into other perpendicular states, we can measure the outcomes just after the faulty gate. We should get the same results as in table 2.1. Therefore, from the measurement

[1] This choice is totally arbitrary, I chose this one only to emphasise more the results

[2] This is because each time that we do a measurement, a circuit simulation the state collapse, so to get the probability i need to simulate the same circuit more times

we should expect to find the initial state (33% of the time in $|000\rangle$ and around 67% of the time in $|111\rangle$) with probability $p_{noerr} = (1-p)^3 = 0.729$. In fact, as it is possible to see from figure 2.4, if we sum up the probability of $|000\rangle$ and $|111\rangle$ we get 0.737 which is close to the expectation.

Instead the probability that an error has occurred on the initial state is $p(1-p)^2 = 0.081$. This means that we should get a different state from the initial one with probability $p(1-p)^2 = 0.081$, but with the same amplitudes (α,β) of the initial one. Let us consider the case where a the bit flip error occurred on a qubit ($|\psi\rangle_F = \alpha|100\rangle + \beta|011\rangle$), hence we should measure this state with probability $p(1-p)^2 = 0.081$. As before, it is possible to see this from figure 2.4. In fact, if we sum up the probability of $|100\rangle$ and $|011\rangle$ we get 0.084 which is close to what we expected. The small fluctuation around the expected values are stochastic errors and because a $O(p^2)$ error occurred.

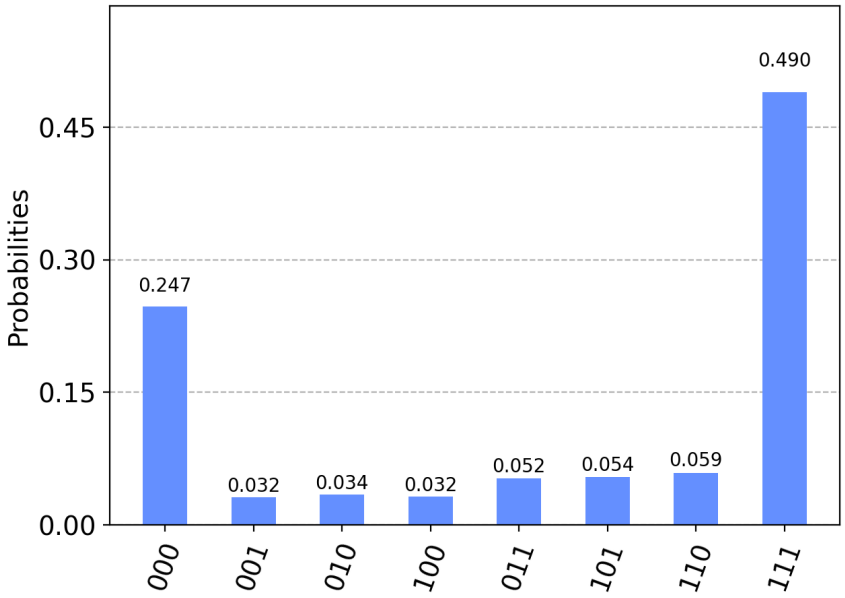


Figure 2.4: This bar graph shows the outcome state and their probabilities after a noisy bit-flip gate. On the x axes there are the possible measurement outcomes, on the y axes the associated probability of measuring them. Along the x axes each histogram bar is labelled with the state. The displayed qubits for each bar are written in the following order: "$q_2q_1q_0$". For example, the second bar shows the probability of "$q_2q_1q_0$" = $|001\rangle$.

Now that we are sure that the noise works as expected, we should look at the recovery and the decoding. The recovery needs two ancillae (in figure 2.1 are labelled with "auxiliary"), which detect the error without collapsing the information. Thank the

ancilla results we can correct the error. The simulation results at the end of the circuit are shown in figure 2.5.

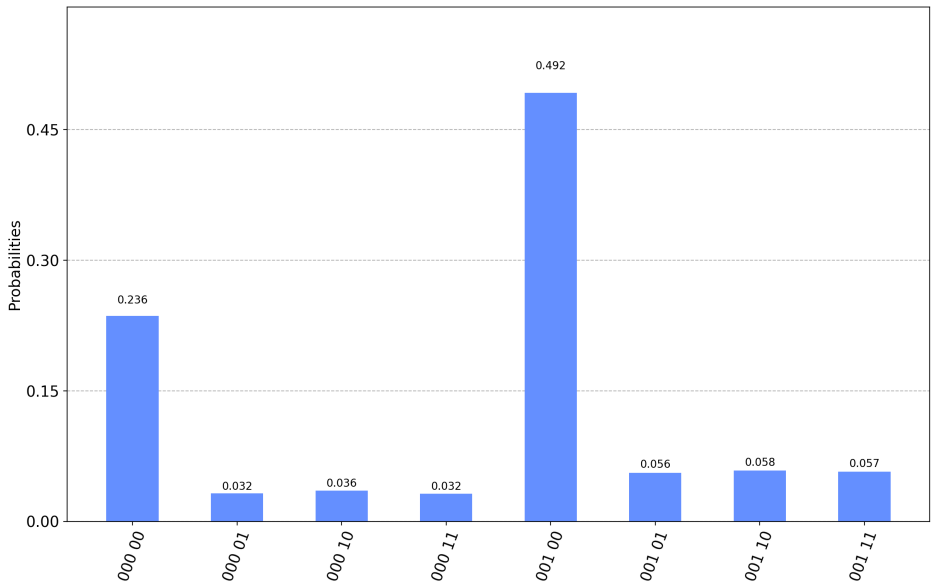


Figure 2.5: This bar graph shows the outcome state and their probabilities after the recovery and the decoding procedures. On the x-axes, there are the possible measurement outcomes with the ancillae that indicate which error occurred, on the y-axes the associated probability. Along the x-axes, each histogram bar is labelled with the state. The displayed qubits for each bar are written in the following order: "$q_2q_1q_0 \quad a_1a_0$", where a_1a_0 indicates the ancilla qubit. Is it possible to see the q_0 state has been recovered if an error occurred.

From the results, is possible to see that all the errors have been recovered[3]. The results are displayed as follow "$q_2q_1q_0 \quad a_1a_0$", with the ancilla measurements "a_1a_0" . In figure 2.5 we see that we recovered successfully the initial state. If we sum up all the probabilities to find the "information qubit" (q_0) in $|0\rangle$, independently of the occurred error, we get 0.336 which is very close to 0.333 as expected. The same works for the probability to find the qubit (q_0) in $|1\rangle$, which for the initial state was approximately 0.667%, here we get 0.663.

[3]To be precise the errors of higher-order (p^2) have been corrected badly, but they are negligible compared to the order of p

3 phase flip code

Similarly, I implemented the 3 phase flip code. The set-up is the same as the 3 bit-flip code, but I applied a Hadamard gate before and after the faulty gate. Recalling that the Hadamard gate changes computational basis, then a Z error may occur, and if we want to see the Z error as a bit-flip we need to change basis again, i.e applying a Hadamard gate again. The results just after the faulty gate are shown in figure 2.6. and the results after the recovery are shown in figure 2.7

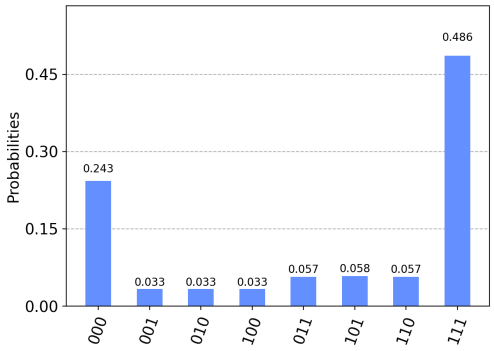


Figure 2.6: This bar graph shows the outcome state and their probabilities after a noisy phase-flip gate between two Hadamards, this allows us to change bases and interpret the phase error as a bit-flip error. On the x axes there are the possible measurement outcomes, on the y axes the associated probability of measuring them. Along the x axes each histogram bar is labelled with three qubits. The displayed qubits for each bar are written in the following order: "$q_2q_1q_0$". For example, the second bar shows the probability of "$q_2q_1q_0$" = $|001\rangle$.

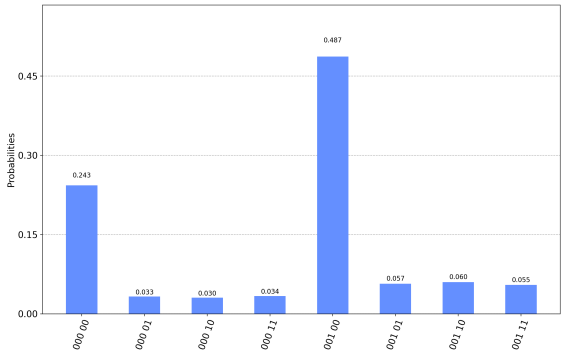


Figure 2.7: This bar graph shows the outcome state and their probabilities after the recovery and the decoding procedures. On the x-axes, there are the possible measurement outcomes with the ancillae that indicate which error occurred, on the y-axes the associated probability. Along the x-axes, each histogram bar is labelled with the state. The displayed qubits for each bar are written in the following order: "$q_2q_1q_0$ a_1a_0", where a_1a_0 indicates the ancilla qubit. Is it possible to see the q_0 state has been recovered if an error occurred.

The considerations and the analysis done in the bit-flip code paragraph are analogous for the phase-flip code. The main differences between these two codes are the encoding procedure, as it is possible to see in 2.2, which implements a change of basis in such a way that a Z error can be considered as a bit-flip error in the new basis and then the error is translated in the computational basis using the Hadamards again. In fact: $HZH = X$, simulating a bit-flip, and the recovery follow as the 3 bitflip code.

2.3 Condition for quantum error correction

In the previous sections we saw how the noise impact on the system, now we are looking for some recovery procedure to correct an eventual error. However, there are some conditions to be satisfied in order to create a quantum error correctin code. These conditions are called Knill-Laflamme conditions [3].

In any code, we must never confuse the logical encoded qubits $|0\rangle_L$ with $|1\rangle_L$, even in the presence of errors. Suppose, in fact, that two errors E_a and E_b map the initially orthogonal codewords $|0\rangle_L \to E_a |0\rangle_L$ and $|1\rangle_L \to E_b |1\rangle_L$ such that the faulty states have non-zero overlap : ${}_L\langle 1| E_b^\dagger E_a |0\rangle_L \neq 0$. When trying to detect the error that happened it is possible (non-zero probability) that we can not distinguish between the events where the system started. In other words, no measurement can decide with certainty whether the initial state was $|0\rangle_L$ or $|1\rangle_L$ If two codewords are orthogonal, and they are acted upon by correctable errors, the result will remain orthogonal. This is a necessary condition given a set $\mathcal{E}$ of correctable errors [4]:

$$\langle\psi_i| E_b^\dagger E_a |\psi_j\rangle = 0 \quad \forall E \in \mathcal{E} \text{ and } \forall |\psi_i\rangle \neq |\psi_j\rangle \in \mathcal{C},$$

where $\mathcal{C}$ indicate the codespace.

This is not enough, another condition is required:

$$\langle\psi_i| E_b^\dagger E_a |\psi_i\rangle = \langle\psi_j| E_b^\dagger E_a |\psi_j\rangle \quad \forall E \in \mathcal{E} \text{ and } \forall |\psi_i\rangle, |\psi_j\rangle \in \mathcal{C}.$$

This second condition is that the outcome of the syndrome measurement must not give any information about the codewords, otherwise the superposition state will collapse. whatever the state encoded in the subspace is, errors occurring on this state must not reveal anything about the state (otherwise we could learn something about the state, thereby destroying quantum information). In other words, the 'symmetric' inner product cannot depend on what exactly the 'current' codeword. This two conditions, called also Knill-Lafflame conditions, can be combined in one yielding to the following theorem:

Theorem 2. *Suppose $\mathcal{E}$ is a linear space of errors acting on the Hilbert space $\mathcal{H}$. Then a subspace C of $\mathcal{H}$ forms a quantum error-correcting code correcting the errors $\mathcal{E}$ iff:*

$$\langle\psi_j| E_b^\dagger E_a |\psi_i\rangle = C_{ab}\delta_{ij}, \tag{2.2}$$

for all $E \in \mathcal{E}$.

[4] A set $\mathcal{E}$ of correctable errors exist an operator $\mathcal{R}$ such that $|\psi\rangle = \mathcal{R}E|\psi\rangle \quad \forall E \in \mathcal{E}$

The matrix's elements C_{ab} do not depend on the codewords $|\psi\rangle$. If C_{ab} has maximal rank, we say that the code is non-degenerate. Otherwise, if C_{ab} has non-maximal rank, we say that the code is degenerate. A non-degenerate code refers to the case that each error in the set $\mathcal{E}_c$ corresponds to a unique syndrome, while in the case of degenerate code, there exist two errors in $\mathcal{E}_c$ whose syndromes are the same. For example, in the 9-qubit code is a degenerate code since the errors Z_1, Z_2, and Z_3 all share the same syndromes. Anyhow, for that case, applying a recovery Z operation on any of the error qubits that share the same syndrome can correct these errors.

2.4 Stabilizer formalism

In quantum error correction codes the description of quantum states becomes a difficult task when the number of qubits starts rising, all the possible quantum states where the information can be projected by the noise grow exponentially in the number of qubits (2^n, where n is the number of physical qubits). The stabilizer formalism is a powerful tool that allows us to make general rules to implement preparation circuits, correction circuits and fault-tolerant logical gate operations once the stabilizer structure of the code is specified.

For example, the three-qubits code works by de-localising the information. The resultant logical state is then encoded in a two-dimensional subspace (the codespace $\mathcal{C}$) of the expanded Hilbert space. If an X-error occurs, the logical state is rotated to an orthogonal error space. This is an event that can be detected via measurements of two commuting operators (stabilizers) and the results are saved in some ancilla qubits, in the case of the three-qubit code the generators of the stabilizer group were $\{Z_1Z_2I, IZ_2Z_3\}$.

Consider $\mathcal{S} = \{M\}$ to be a set of commuting error operators ($[M_i, M_j] = 0 \quad \forall i, j$), this set is called Abelian. Since the operators all commute, they have simultaneous eigenstates. Let $\mathcal{C} = \{|u\rangle\}$ be an orthonormal set of simultaneous eigenstates all having eigenvalue $+1$:

$$M|u\rangle = |u\rangle \quad \forall u \in \mathcal{C}, \quad \forall M \in \mathcal{S}. \tag{2.3}$$

The set $\mathcal{C}$ is called codespace and its elements $|u\rangle$ are called code vectors or quantum codewords. A general state in the codespace is an encoded state or logical state, and it can be expressed as a superposition of the code vectors:

$$|\psi\rangle_L = \sum_{u \in \mathcal{C}} a_u |u\rangle,$$

On the other hand, a suitable stabilizer set of commuting operators acting on a single qubit can be a subgroup of the Pauli group:

$$\mathcal{P} = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}\,.$$

This set of matrices forms a group under the operation of matrix multiplication. The reason why that the multiplicative factor ± 1 and $\pm i$ are included is to ensure that $\mathcal{P}$ is closed under multiplication, thus forming a legitimate group. We can also generalize the concept of stabilizers for a system of multiple qubits. Given n qubits a stabilizer set is a subgroup of the Pauli group $\mathcal{P}_n$, which is created by taking the n-fold tensor product of $\mathcal{P}$, i.e.

$$\mathcal{P}_n = \mathcal{P}^{\otimes n} = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}^{\otimes n}\,.$$

We can now define stabilizers more precisely: a subgroup $\mathcal{S} \subseteq \mathcal{P}_n$ is called stabilizer group of a system of n qubits with k logical qubits, and $\mathcal{C}$ is the vector space stabilized by $\mathcal{S}$.

One feature of the stabilizer codes is that we can start from a stabilizer set and then deduct a codeword. A perfect example, (again), is the three-qubit code ($n = 3$), we can start defining a stabilizer set $\mathcal{S} \equiv \{III, Z_1Z_2I,\ IZ_2Z_3, Z_1IZ_3\}$. The subspace fixed by the operator Z_1Z_2I is spanned by $|000\rangle$, $|001\rangle, |110\rangle$ and $|111\rangle$, and the subspace fixed by IZ_2Z_3 is spanned by $|000\rangle, |100\rangle, |011\rangle$ and $|111\rangle$. The elements $|000\rangle$ and $|111\rangle$ are common to both these lists. Consequently, $\mathcal{C}$ must be the sub-space spanned by the states $|000\rangle$ and $|111\rangle$. Looking at this example we noticed that the sub-spaces have been stabilized by only two of the operators in $\mathcal{S}$. In fact, mathematically a group can be described by its generators. A set of elements $\{g1, ..., gl\}$ in a group G is said to generate the group G if every element of G can be written as a product of some of its elements. In the example $\mathcal{S} = \langle Z_1Z_2I, IZ_2Z_3\rangle$ as $Z_1IZ_3 = (Z_1Z_2I)(IZ_2Z_3)$ and $III = (Z_1Z_2I)^2$. The great advantage of using generators to describe groups is that they provide a compact means of describing the group. Nevertheless, not just any subgroup $\mathcal{S}$ of the Pauli group can be used as the stabilizer for a nontrivial vector space. For example, consider the subgroup of $\mathcal{P}_1$ consisting of $\{\pm I, \pm X\}$, the only solution to $(-I)|\psi\rangle = |\psi\rangle$ is $|\psi\rangle = 0$, and thus $\{\pm I, \pm X\}$ is the stabilizer for the vector space with only the null vector. Hence, two conditions are necessary in order not to span a trivial vector space :

1. the stabilizer subgroup is Abelian (i.e the elements of $\mathcal{S}$ commute)
2. $-I$ and $-iI$ are not an element of the stabilizer set.

One can show that if a stabilizer group $\mathcal{S}$ has $n - k$ independent generators, then

$\dim[\mathcal{S}] = 2^{n-k}$ and $\dim[\mathcal{C}] = 2^k$, where n indicates the number of physical qubits utilized and k the logical qubits represented [8] [9]. Whenever possible, $\mathcal{S}$ is typically represented through its independent generators $\mathcal{S} = \langle g_1, g_2, \ldots, g_{n-k} \rangle$.

To understand how stabilizer codes detect and correct errors it is helpful to assume that the correctable set of errors $\mathcal{E} = \{E\}$ consist of operators from the Pauli group.

Suppose that an error operator E acts on the state. It anticommutes with some of the stabilizer generators, and commutes with others[5].

$$M(E\,|\psi\rangle_L) = \pm EM\,|\psi\rangle_L = \pm(E\,|\psi\rangle_L). \tag{2.4}$$

The action of E changes the codeword to a new eigenstate of the stabilizer generators, where the eigenvalue is still $+1$ for all the generators that commute with E, but is -1 for those generators that anticommute with E.

The measured eigenvalues $\lambda = \pm 1$ for each generator are called error syndrome. To extract the syndrome we measure all the observables in the stabilizer. To do this, it is sufficient to measure any set of $n-k$ linearly independent M in $\mathcal{S}$. Note that such a measurement has no effect on a state in the encoded subspace, since it is already an eigenstate of all these observables. The measurement projects a noisy state into an eigenstate of each M, with eigenvalue $\lambda = \pm 1$. The error can be deduced from the syndrome, hence, to detect which error affected our logical qubit we need to store the results from the syndrome measurements.

The syndrome extraction can be done most simply by attaching an $n-k$ qubit ancilla A to the system, and storing in it the eigenvalues.

An efficent way to store them is the following:

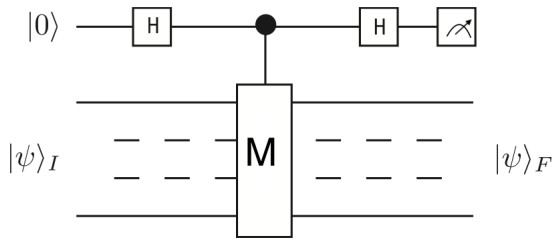


Figure 2.8: Quantum Circuit required to project an arbitrary state, $|\psi\rangle$ into an ± 1 eigenstate of the stabilizer M. The measurement result is store then in the ancilla.

For some arbitrary input state $|\psi\rangle_I$, prepare an ancilla in $(|0\rangle + |1\rangle)/\sqrt{2}$. Then, operate controlled-M with ancilla as control, and system as target (We work directly on

[5] Commute: $[M, E] = ME - EM = 0$. Anticommute: $\{M, E\} = ME + EM = 0$

the encoded state), and then apply an Hadamard rotations on the ancilla.

After this procedure the final state is:

$$
\begin{aligned}
|\psi\rangle_F &= \frac{1}{2}\left(|\psi\rangle_I + M|\psi\rangle_I\right)|0\rangle + \frac{1}{2}\left(|\psi\rangle_I - M|\psi\rangle_I\right)|1\rangle \\
&= \frac{1}{2}\left(|\psi\rangle_I + \lambda|\psi\rangle_I\right)|0\rangle + \frac{1}{2}\left(|\psi\rangle_I - \lambda|\psi\rangle_I\right)|1\rangle.
\end{aligned}
$$

If M is applied on the encoded states it gives or $+1$ or -1 as eignestate depending on which error occurred. Hence, if the stabilizer commute with the error ($\lambda = +1$) the ancilla final state will be $|0\rangle$, and viceversa. The syndrome measurements given by the application of the stabilizers are thus stored in the ancilla states.

The extended network for an error correction code is shown in figure 2.9

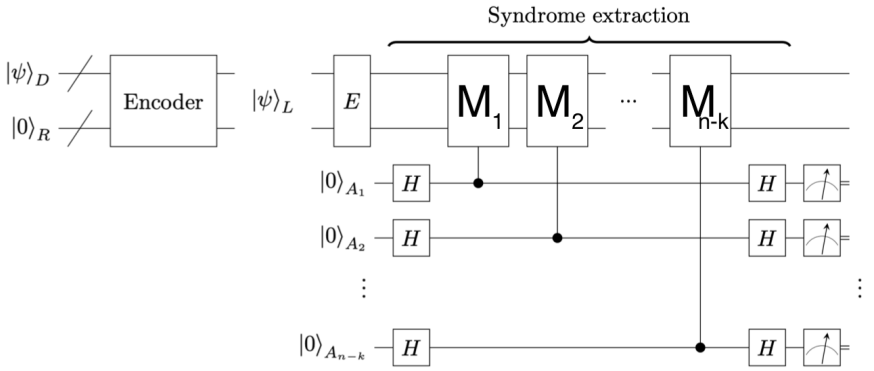


Figure 2.9

Repeating this process for all the $n-k$ operators M which span $\mathcal{S}$, the effect is to couple system with the ancilla as follow:

$$
|0\rangle_a \sum_i \left(E_i|\phi\rangle_L\right) \rightarrow \sum_i |s_i\rangle_a \left(E_i|\phi\rangle_L\right).
$$

The syndromes s_i are $(n-k)$ -bit binary strings. So far the treatment is completely general. Now suppose the E_i all have different syndromes. Then a projective measurement of the ancilla will collapse the sum to a single term taken at random: $|s_i\rangle_a \left(E_i|\phi\rangle_L\right)$, and will yield s_i as the measurement result. The measurement result of the ancilla determines which eigenstate $|\psi\rangle_L$ is projected into. Since there is only one E_i with this

syndrome, we can deduce the operator E_i and correct it.

This is the general prescription for correcting errors with a stabilizer code. One measures the values ± 1 of the stabilizer generators; and from the resulting error syndrome, one deduces which error occurred, and correct it (which for Pauli operators means just applying the error again). If the true error operator was actually a linear combination of Pauli operators, measuring the stabilizer generators will project the state into a joint eigenspace, and one proceeds exactly as if the error had been a Pauli operator. Just like linear codes, for a small stabilizer code one can use a look-up table of error syndromes; for a larger code, a decoding algorithm is needed.

Let us see how this works for the bit-flip code. It has generators $g_1 = ZZI$ and $g_2 = IZZ$. The three weight-one errors are $E_1 = XII, E_2 = IXI, E_3 = IIX$. We can see that E_1 anticommutes with g_1 and commutes with g_2 so we will measure the following ancillae: $|0\rangle_{A_1} |1\rangle_{A_2}$; E_2 anticommutes with both g_1 and g_2, with: $|1\rangle_{A_1} |1\rangle_{A_2}$ as ancillae measurement; and E_3 anticommutes with g_2 and commutes with g_1, with $|1\rangle_{A_1} |0\rangle_{A_2}$ as ancillae measurements. Since g_1 and g_2 are commuting observables, we can measure them to diagnose which error happened (or no error). Then, since Pauli operators are unitary and square to the identity, we can then undo the effects of the error by applying the appropriate Pauli operator again.

2.5 Quantum code distance

A quantum correction code can be identified by 3 parameters using the following notation: $[[n, k, d]]$[6]. This means that a QECC encodes k logical qubits using n physical qubits , in such a way that any operation which maps some encoded state to another encoded state must act on at least d qubits. So far we saw the physical and encoded qubits, but we never focused on the third parameter, so let's define what it is the code distance better, using the stabilizer formalism. As shown in the previous chapters, the eigenvalue of an operator $M \in \mathcal{S}$ from the stabilizer detects errors which anticommute with M. On the other hand, if any of those errors which commute with M occurs, the code it is not be able to detects the error anymore. This is because if we measure the eigenvalue of all the generators of the stabilizer they are all still $+1$, and also the correct codeword has eigenvalue $+1$ so there is no way that we can tell if any error occurs. In fact, stabilizer maps codewords in codewords, it is not that we don't measure the right thing actually there is no way to tell.

[6]The doublebrackets are used simply to denote that the code being referred to is a quantum error correction code rather than a classical code.

In particular let's define a stabilizer set $\mathcal{S}$, and let T(S) be the corresponding QECC. Define then,

$$N(S) = \{N \in P_n \text{ s.t. } MN = NM \quad \forall M \in S\},$$

and errors in this space are undetectable errors. Hence, we can say that a general QECC detects any error not in N(S)\S (i.e., errors which commute with the stabilizer are not detected). We need also to remove all the error in S because "errors" in S leave all codewords fixed, so are not really errors (Degenerate QECC, like the Shor code) In this way, we are left with all those errors which anticommute with some M and can be detected. Starting from this point we can define the distance d. The distance d of T(S) is the weight of the smallest Pauli operator N in N(S)\S.

Intuitively, this can be though as the number of qubit that we need to flip to obtain another codeword. In other words, as is the case for classical codes, the distance of a quantum code is defined as the minimum size error that will go undetected.

This definition of distance tells us not only which errors can be detected but also how many of them can we correct. We can tell how many errors the code corrects by looking at operators that commute with the stabilizer.

We say that a QECC can correct t errors if the set of errors that allow the recovery (those errors not in N(S)\S) of weight t or less, must satisfy the sufficient conditions for the code to be able to correct the error:

$$\langle i| E_a^\dagger E_b |j\rangle = C_{a,b}\delta_{ij},$$

where $E_a^\dagger E_b \notin$ N(S)\S, and the states i and j are codewords.

Therefore, to correct t errors, we should look to all these pairs $E_a^\dagger E_b$ that show up in the criterion. Each $E_{a,b}$ acts on t qubit (correspondingly to the weight), but $E_a^\dagger E_b$ acts on $2t$ qubits. Then, the smallest thing in $N(S)\backslash S$ should be at least of weight $2t+1$.

To sum up, if we want to correct error with weight t, we need a code distance $d \geq 2t+1$. Let's take a fast example, (again), the bit-flip code. In this code the smallest operator that transform $|0\rangle_L$ to $|1\rangle_L$, or vice versa, it is $\bar{X} = X_1X_2X_3$. If it were the case that qubits were only susceptible to $X - errors$, then the three-qubit code would have distance $d = 3$, cause each pauli operator has weight 1 and to obtain another codeword you need 3 of them. However, as qubits are also susceptible to phase-flip errors like Z errors. The action of Z maps a codeword to another one, hence, for the bit-flip code a Z error is undetectable. Hence the real distance for this code is $d = 1$.

2.6 Shor QEC code, $[[9, 1, 3]]$

The Shor code is an example of a distance-three degenerate code for which it is possible to apply a successful recovery operation for any single-qubit error. This code uses 9-qubits, and it was the first QECCs that was capable of correcting any arbitrary error (Phase and bit flip) on a single qubit, protecting one logical qubit state. The Shor code can be constructed by concatenating the bit-flip code and the phase-flip code. Code concatenation involves embedding the output of one code into the input of another. One can build the encoded Shor states from the codespace of the bit-flip and the phase-flip code.

Consider the codespace of a single bit-flip code:

$$\mathcal{C}_{3b} = \text{span}\left\{|0\rangle_{3b} = |000\rangle, |1\rangle_{3b} = |111\rangle\right\}, \quad \mathcal{S}_{3b} = \langle Z_1Z_2, Z_2Z_3\rangle$$

where $\mathcal{S}_{3b}$ are the code stabilizers. Similarly, codespace for phase-flips $\mathcal{C}_{3\text{p}}$ is defined

$$\mathcal{C}_{3\text{p}} = \text{span}\left\{|0\rangle_{3\text{p}} = |+++\rangle, |1\rangle_{3\text{p}} = |---\rangle\right\}, \quad \mathcal{S}_{3\text{p}} = \langle X_1X_2, X_2X_3\rangle$$

Then to build the nine-qubit code, the bit-flip code is embedded into the codewords of the phase-flip code. This concatenation maps the $|0\rangle_{3\text{p}}$ codeword of the phase-flip code to a nine-qubit codeword $|0\rangle_9$ as follows

$$|0\rangle_{3\text{p}} = |+++\rangle \overset{\text{concatenation}}{\longrightarrow} |0\rangle_9 = |+\rangle_{3\text{b}}|+\rangle_{3\text{b}}|+\rangle_{3\text{b}},$$

where $|+\rangle_{3\ \text{b}} = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)$ is a logical state of the bit-flip code.

Similarly, the concatenation maps the $|1\rangle_{3\text{p}}$ codeword of the phase-flip code to

$$|1\rangle_{3\text{p}} = |---\rangle \overset{\text{concatenation}}{\longrightarrow} |1\rangle_9 = |-\rangle_{3\ \text{b}}|-\rangle_{3\ \text{b}}|-\rangle_{3\ \text{b}},$$

where $|-\rangle_{3b} = \frac{1}{\sqrt{\Omega}}(|000\rangle - |111\rangle)$. The code defined by the codewords $|0\rangle_9$ and $|1\rangle_9$ is the nine-qubit Shor code with paramaters $[[9, 1, 3]]$, and the encoding procedure in a quantum circuit is showed in 2.10.

Finally, the qubit state (the sending information) $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is encoded in

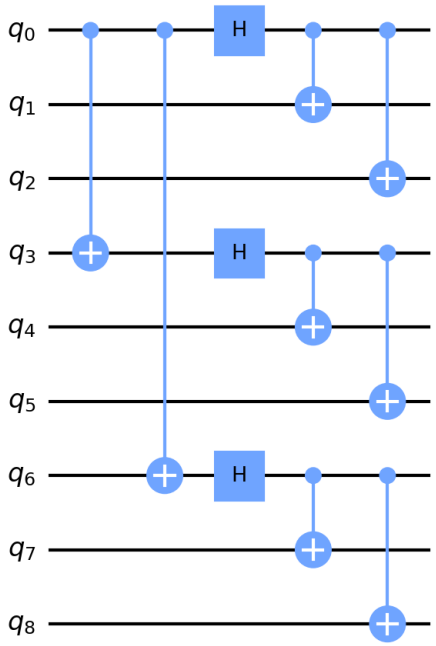


Figure 2.10: Quantum circuit that shows the encoding procedure for the Shor code

$|\psi_L\rangle = \alpha\,|0_L\rangle + \beta\,|1_L\rangle$. The codespace basis are:

$$|0_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle),$$
$$|1_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle),$$

Once one has find the codespace, it can also look for a stabilizer set. Since this code is a concatenation of the bit-flip code and the phase-flip code one might use the same stabilizers but with more accuracy. A suitable stabilizer set is spanned by:

$$\begin{aligned}\mathcal{S}_{[[9,1,3]]} = \langle & Z_1Z_2, Z_2Z_3, Z_4Z_5, Z_5Z_6, Z_7Z_8, Z_8Z_9 \\ & X_1X_2X_3X_4X_5X_6, X_4X_5X_6X_7X_8X_9\rangle\,.\end{aligned} \tag{2.5}$$

The first six terms are the stabilizers of the bit-flip codes in the three-blocks of the code. The final two stabilizers derive from the stabilizers of the phase-flip code.

Hence, using the stabiliser formalism we can proceed for the syndrome measurement.

Table 2.4 shows the syndromes for all single-qubit errors in the nine-qubit code.

Error	Syndrome, S	Error	Syndrome, S
X_1	10000000	Z_1	00000010
X_2	11000000	Z_2	00000010
X_3	01000000	Z_3	00000010
X_4	00100000	Z_4	00000011
X_5	00110000	Z_5	00000011
X_6	00010000	Z_6	00000011
X_7	00001000	Z_7	00000001
X_8	00001100	Z_8	00000001
X_9	00000100	Z_9	00000001

Table 2.4: The syndrome table for single-qubit X and Z errors on the nine-qubit code. The nine-qubit code is a degenerate code, as certain Z errors share the same syndrome

Each of the X errors produce unique syndromes. In contrast, Z -errors that occur in the same block of the code have the same syndrome. Fortunately, this degeneracy in the code syndromes does not reduce the code distance. To see why this is the case, consider the single-qubit errors Z_1 and Z_2, both of which map to the syndrome '00000010'. The decoder therefore has insufficient information to differentiate between the two errors, and will output the same recovery operation for either. For the purposes of this example, we will assume that the recovery operation the decoder outputs is $\mathcal{R} = Z_1$. For the case where the error is $E = Z_1$, the recovery operation restores the logical state as $\mathcal{R}E|\psi\rangle_9 = Z_1Z_1|\psi\rangle_9 = |\psi\rangle_9$. In the event where $E = Z_2$, the recovery operation still restores the logical state as $\mathcal{R}E = Z_1Z_2$ is in the stabilizer of $\mathcal{C}_{[[9,1,3]]}$, and therefore acts on the logical state as follows $Z_1Z_2|\psi\rangle_9 = |\psi\rangle_9$. The same arguments can be applied to the remaining degenerate errors of the code. As a result, the nine-qubit code has the ability to correct all single-qubit errors and has distance $d = 3$ In this code the process of correcting a X error or a Z error is totally independent: the inner layer of the code corrects bit flip errors taking the majority within each set of three, so On the other hand, the outer layer corrects phase flip errors: We take the majority of the three signs, Since these two error correction steps are independent, the code also works if there is both a bit flip error and a phase flip error, and since $Y = iZX$, a Y error can be thought as a single X error and a single Z error acting on the same qubit, up to an irrelevant global phase.

So this code can correct any Pauli error acting on a single qubit. However, even this is not the limit. Note that any operator on a single qubit can be written as a linear

combination $E = aI + bX + cY + dZ$ for some complex numbers a, b, c, d. This allow us, using this code, to correct also a continuous tipe of errors. For instance, suppose the action of a general phase error on the codeword:

$$R_{\theta/2} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix} = e^{i\theta/2} \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix}$$

(with an overall phase that does not matter), we can write it as

$$R_{\theta/2} = \cos\frac{\theta}{2} I - i \sin\frac{\theta}{2} Z.$$

It turns out that our earlier error correction procedure will also correct this error. This can suggest a bigger result expressed by the following theorem [10]:

Theorem 3. *If a quantum code corrects errors A and B, it also corrects any linear combination of A and B. In particular, if it corrects all weight t Pauli errors, then the code corrects all t-qubit errors.*

2.7 CSS codes

CSS codes are a very special class of stabilizer codes with special properties. In fact, principles from the theory of classical error correction codes can be adapted for the construction of quantum error correction codes. Those quantum error correction codes can be driven from the classical codes and are called CSS codes. Before introducing a CSS code construction, is better to do a brief recall of classical error correction codes, and we will take the Hamming code as a landmark. There are two ways of representing a classical linear code and find its codewords: either as the image of a matrix G called the generator matrix or as the kernel of matrix H called the parity check matrix. More precisely we can define a linear classical error correcting code $\mathcal{C}$ of n-bit vectors by: $x \in \mathcal{C} \iff Hx = 0$ with the linear property: $x, y \in \mathcal{C} \to x + y \in \mathcal{C}$. Moreover, $HG^T = 0$ because each row of the generator matrix is a codeword and must satisfy the parity check; the other codewords are all linear combinations of the rows of the generator matrix. We can deduce a similarity between the parity check matrix and the stabilizer set. In fact, in classical codes if an error occur it affects the codeword as follow $x + e_1$, and no longer satisfy the parity check $H(x + e_1) = He_1 \neq 0$. An analogous situation corresponds when an error occurs and the condition (2.3) is no longer satisfied. Hence, we can see the rows of H as indicators of which stabilizers are needed. A well known

classical correcting code is the Hamming code which is [7,4,3]. The parity check of this code is:

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

If we replace each 1 in this matrix by the operator Z, and 0 by I, we are just specifying three operators that implement the parity check measurements. The statement that the classical Hamming code corrects one error is the statement that each bit flip error of weight one or two anticommutes with one of these three operators.

$$H_1 = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \rightarrow \mathcal{S}_{bf} = \left\langle \begin{matrix} I & I & I & Z & Z & Z & Z \\ I & Z & Z & I & I & Z & Z \\ Z & I & Z & I & Z & I & Z \end{matrix} \right\rangle$$

However, qubits are subjected to phase-flips as well. We can use again the same procedure; we replace each 1 by X instead of Z and we can correct a phase error. We again get three operators, and they will anticommute with any weight one or two Z error.

$$H_2 = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \rightarrow \mathcal{S}_{pf} = \left\langle \begin{matrix} I & I & I & X & X & X & X \\ I & X & X & I & I & X & X \\ X & I & X & I & X & I & X \end{matrix} \right\rangle$$

Thus, if we make a stabilizer out of the three Z operators and the three X operators, we get a code that can correct any single qubit error. X errors are picked up by the first three generators, Z errors by the last three, and Y errors are distinguished by showing up in both halves.

$$\mathcal{S}: \begin{matrix} I & I & I & Z & Z & Z & Z \\ I & Z & Z & I & I & Z & Z \\ Z & I & Z & I & Z & I & Z \\ I & I & I & X & X & X & X \\ I & X & X & I & I & X & X \\ X & I & X & I & X & I & X \end{matrix} \tag{2.6}$$

We solved a difficult problem as finding a stabilizer set of a quantum error correcting code with a minimum effort using the knowledge from the classical theory. But there is

one thing that we need to pay attention when we use this procedure: the stabilizer set must be Abelian, i.e. all the elements commute. In the previous example, the stabilizer set is Abelian and the corresponded code is the Steane code $[[7, 1, 3]]$.

This example uses the same classical code for both the X and Z generators, but there was no reason to do so. We could have used any two classical codes C_1 and C_2. The only requirement is that the X and Z generators commute. This corresponds to the statement that $C_2^{\perp} \subseteq C_1$ where $C_2^{\perp}$ is the dual code of C_2[7]. If C_1 is an $[n, k_1, d_1]$ code, and C_2 is an $[n, k_2, d_2]$ code, then the corresponding quantum code is an $[[n, k_1 + k_2 - n, \min(d_1, d_2)]]$ code.

The codewords of a CSS code, in this case the Steane code, include two entangled states obtained from the classic codewords of each coset of C_1 relative to $\mathrm{C}_2^{\perp}$: the logical 0 is given from all the codewords with even weight $\mathrm{C}_2^{\perp} \oplus (0000000)$:

$$|0\rangle_L = \tfrac{1}{\sqrt{8}}[|0000000\rangle + |1010101\rangle + |0110011\rangle + |1100110\rangle \\ +|0001111\rangle + |1011010\rangle + |0111100\rangle + |1101001\rangle]$$

To determine the other logical codeword we need to find an element of C_1 that is not in $C_2^{\perp}$. An example of such an element is (1111111), giving then the other logical qubit, which this time is formed by all the codewords with odd weight:

$$|1\rangle_L = \tfrac{1}{\sqrt{8}}[|1111111\rangle + |0101010\rangle + |1001100\rangle + |0011001\rangle \\ +|1110000\rangle + |0100101\rangle + |1000011\rangle + |0010110\rangle]$$

2.7.1 Simulation of the Seven qubit code

In this subsection, I will show and discuss the computational results of the seven qubit code. I implemented this code in python, in particular using the qiskit library [7]. This code can be found in my GitHub page as well.

2.8 Quantum Hamming bound

Even if CSS code construction is a very straightforward and elegant procedure, it not guarantees that we can create the best QECC code from classical codes. In fact, exists a better QECC: the $[[5, 1, 3]]$. Can we do better than this? Actually no, there is an

[7] $C_2^{\perp}$ is the dual code of C_2, intuitively this means that in $C_2^{\perp}$ the generator matrix and the parity matrix are switched respect C_2. In other words it consists of those codewords which are orthogonal to the codewords of C_2)

upper bound that we cannot cross over if we want to construct an error correction code. This bound is the Hamming bound, which unfortunately only applies to non-degenerate codes, but it gives us an idea of what more general bounds may look like [6]. Suppose a non-degenerate code is used to encode k qubits in n physical qubits in such a way that it can correct errors on any subset of t or fewer qubits. Suppose j errors occur, where $j \leq t$. There are $\begin{pmatrix} n \\ j \end{pmatrix}$ set of locations where errors may occur. With each such set of locations there are three possible errors (the 3 pauli matrices X,Y,Z) that may occur on a single qubit for a total of 3^j possible errors. The total number of errors that may occur on t or fewer qubits is therefore:

$$\sum_{j=0}^{t} 3^j \begin{pmatrix} n \\ j \end{pmatrix}$$

where $j = 0$ corresponds to the case where there is no error. In order to encode k qubits in a non-degenerate way each of these errors acting on each basis codeword produces a linearly independent state, must correspond to an orthogonal 2k-dimensional subspace. All these subspaces must be fit in the 2^n-dimensional available to n qubits, leading to the inequality:

$$\left(\sum_{j=0}^{t} 3^j \begin{pmatrix} n \\ j \end{pmatrix} \right) 2^k \leq 2^n \tag{2.7}$$

This relation is called the quantum Hamming bound.

To understand the power of this relation consider, for example, the case where we wish to encode one qubit in n qubits in such a way that errors on one qubit are tolerated. Substituting $t = 1, k = 1$ in 2.7, we obtain the relation:

$$2(1 + 3n) \leq 2^n$$

This is satisfied only for $n \geq 5$. In fact, the $[[5, 1, 3]]$ is the best quantum error correction code. There is no code that encodes one qubit in fewer than five qubits in such a way as to protect from all possible errors on a single qubit. A non-degenerative code for large n and t, $R = \frac{k}{n}$ (Rate) fixed the best nondegenerate quantum codes satisfy:

$$\frac{k}{n} \leq 1 - \frac{t}{n} log_2(3) - H\left(\frac{t}{n}\right)$$

where H is the entropy function ($H(x) = -xlog_2(x) - (1 - x)log_2(x)$) The general behaviour hof the hamming bound as a function of $\frac{t}{n}$ is shown in figure 2.11.

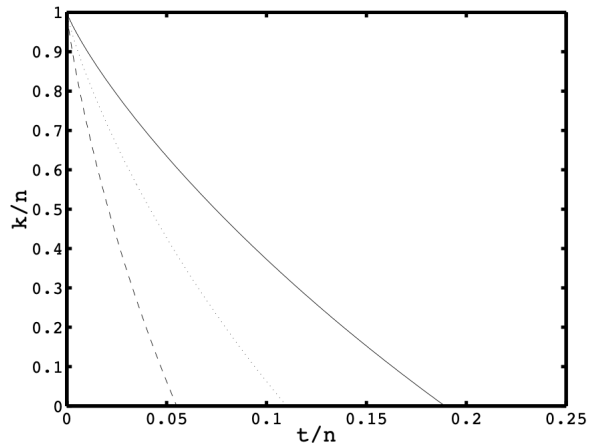


Figure 2.11: This figure shows the hamming bound. Indicates how many qubit it is possible to encode to correct t errors usin n physical qubit. The dotted and the dashed line are different tipe of bounds [6]

.

Chapter 3

Fault-Tolerant computation

In this last chapter I will discuss how to make a circuit even more resistant to error propagation through fault-tolerant procedures. Finally, I will present an important result: the threshold theorem, which states that arbitrarily long reliable fault-tolerant quantum computation is possible if the error rate of a physical system is below some threshold value.

3.1 Introduction to fault-tolerant computation

So far we had a sort of communication scenario where Alice wants to send information to Bob and the error occurs in the noisy channel. Then we developed good algorithms to contrast errors that happen in between, protecting the information. To do that we supposed that Alice and Bob were dealing with perfect quantum computers with ideal gates and ideal operations (i.e not subjected to errors). However, nature is far from this. The error can occur while we are doing any quantum operations, also during the quantum error correction procedure. Noise afflicts each elements used to build circuits: the state preparation procedures, quantum logic gates, measurement of the output, and even the simple transmission of quantum information along quantum wires.

To face this problem we can use fault-tolerant computation, which computes directly on encoded quantum states in such a manner that decoding is never required. Hence, each qubit in the original circuit is replaced with an encoded block of qubits, using an error-correcting code such as the seven qubit Steane code, and each gate in the original circuit is replaced with a procedure for performing an encoded gate acting on the encoded state. One might think performing error-correction periodically after each operation, but this is not sufficient to prevent the errors formations.

In fact, when we perform multiple qubits gates during a quantum computation, any existing error can propagate to other qubits, even if the gate itself can be considered ideal. For instance, a CNOT gate can propagate forward (from control qubit to the target qubit) a bit-flip error. In figure 3.1 a bit-flip error occurs on the control qubit just before the CNOT gate. This single error will cascade forward such that the X error propagates on both output qubits after the gate.

We started with 1 error and we ended up with 2 errors. Another example, that

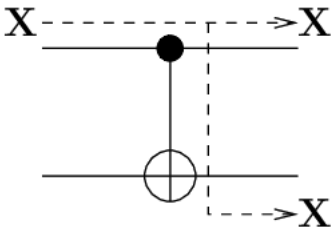


Figure 3.1: A CNOT gate can cause a X error to propagate forward so that instead of affecting one qubit, it affects two. This is also true when encoded qubits are used, and an encoded CNOT is implemented.

has not any classical analogy, is the backwards (from target to control) propagation. Imagine having a CNOT and a Z error occurs just before on the target qubit, then after the CNOT we will have 2 phase errors on both qubits as shown in figure 3.2 .

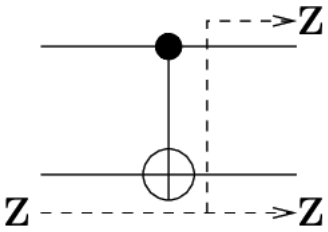


Figure 3.2: A CNOT gate can cause a Z error to propagate backwards so that instead of affecting one qubit, it affects two. This is also true when encoded qubits are used, and an encoded CNOT is implemented.

This is clearly a problem if we build an arbitrary circuit that implements CNOT gates and a QECC that corrects up to 1 error is not able to correct the propagated error anymore. Thus, by error propagation, a correctable local error is converted into a non-correctable one. The procedure described is thus not fault tolerant.

Hence, to achieve a fault tolerant computation we first need to implement fault

tolerant logic operations, which prevent the building up of errors in the same error correction block. A solution is to perform logic gates transversally - i.e. a transversal gate interacts the ith qubit in one block with the ith qubit of the other block. Recall that single encoded qubit gates involving only one block are transversal, because the ith qubit interacts with itself. For instance, in the Steane code the logical $\bar{X} = X^{\otimes 7}$ and $\bar{Z} = Z^{\otimes 7}$ operations on a single encoded qubit are the first examples of valid transversal codeword operations. For any stabilizer codes, a large class of operations can be performed on logical data in a fault-tolerant way. Furthermore, the performed operations must map codewords to codewords.

More precisely, if a given logical state $|\psi\rangle_L$ is stabilized by $M \in \mathcal{S}$, and the logical operation U is applied, the new state $U|\psi\rangle_L$ is stabilized by $UMU^\dagger \in \mathcal{S}$, i.e,

$$(UMU^\dagger)U|\psi\rangle_L = UM|\psi\rangle_L = U|\psi\rangle_L \tag{3.1}$$

Hence, for U to preserve the codespace, $UMU^\dagger$ must be in the stabiliser set. This properties is satisfied for any $U \in \mathbf{C}$ where $\mathbf{C}$ is called the Clifford group:

Definition 3.1. *Let $\mathcal{P}_n$ be the Pauli group on n qubits and let $\mathcal{U}(2^n)$ be the unitary group. The Clifford group $\mathbf{C}_n$ on n qubits is:*

$$\mathbf{C}_n = \left\{ U \in \mathcal{U}(2^n) : UPU^\dagger \in \mathcal{P}_n \text{ for all } P \in \mathcal{P}_n \right\} \tag{3.2}$$

The Clifford group on n qubits can be generated by the Hadamard gate H, $R_{\pi/4}$ gate, and CNOT gate, where

$$H = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad R_{\pi/4} = e^{-i\pi/4}\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad \text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

From this we can notice also how errors change after applying quantum gates: the operation of U on an erroneous codeword $E|\psi\rangle$ gives,

$$UE|\psi\rangle = \left(UEU^\dagger\right)U|\psi\rangle \tag{3.3}$$

Observe that $U|\psi\rangle$ is itself a state, that comes from the operation of U on the state $|\psi\rangle$. Here we can see that the error E before the operation U becomes $UEU^\dagger$ after the operation of U. So now our goal is to implement the generators of clifford group in a

fault-tolerant way. Let us think this in the context of the seven qubit code $[[7, 1, 3]]$ with its stabiliser set (2.6).

Fault-tolerant Hadamard and $R_{\pi/4}$

A logical Hadamard gate $\bar{H}$ interchanges $\bar{Z}$ logical and $\bar{X}$ logical under conjugation, just as the Hadamard gate H interchanges Z and X under conjugation. $\bar{H} = H^{\otimes 7}$ accomplishes this task, so that a Hadamard on the encoded qubit can be implemented as in figure 3.3

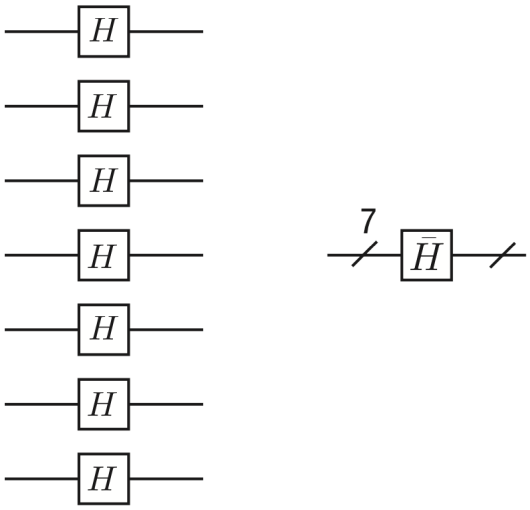


Figure 3.3: This figure shows the logical hadamard gate. On the right is showed a more compact notation: the 7 and the slash on the wire indicate that we are operating on an encoded state

This is fault-tolerant because the failure of a single component in the circuit can cause at most one error in the block of qubits output from the procedure. To see that this is true suppose that a Z error occurs on the first qubit just before the encoded H gate was applied. The combined operation on the qubit is HZ. Then, we can use the equation 3.3, which describes how an error spreads after applying a gate. Hence it gives $HZ = (HZH^\dagger)H = XH$, so such an error is equivalent to o first the application of H and then the occurring of X.

Moreover, $\bar{H}$ maps stabilisers in other stabiliser, let us take the first stabilizer $M_1 = ZZZZIII \in \mathcal{S}$ and we see that $\bar{H}M_1\bar{H} = M_4 = XXXXIII$ which is still in the stabilizer.

A very similar construction can be implemented also for the $\frac{\pi}{4}$ logical operation.

The $R_{\pi/4}$ gate leaves Z unaltered, while X is mapped to $Y = iXZ$. However, applying $R^{-}_{\pi/4} = R^{\otimes 7}_{\pi/4}$ takes $\bar{Z}$ to $\bar{Z}$ under conjugation, and $\bar{X}$ to $-\bar{Y}$. The minus sign in front of the $-\bar{Y}$ may be fixed up by applying $\bar{Z}$. Thus, applying the operation $ZR_{\pi/4}$ to each qubit in the code affects an encoded phase gate, which is transversal and thus fault-tolerant.

Fault-tolerant CNOT gate

A two qubit logical CNOT operation can also be applied in the same transversal way. For unencoded qubits, a CNOT operation performs the following mapping on the two qubit stabilizer set,

$$\begin{aligned} X \otimes I &\to X \otimes X \\ I \otimes Z &\to Z \otimes Z \\ Z \otimes I &\to Z \otimes I \\ I \otimes X &\to I \otimes X. \end{aligned}$$

Where the first operator corresponds to the control qubit and the second operator corresponds to the target. This can be extended to the logical space. We can apply the logical CNOT operation between two encoded states:

$$\begin{aligned} \bar{X} \otimes I &\to \bar{X} \otimes \bar{X}, \\ I \otimes \bar{Z} &\to \bar{Z} \otimes \bar{Z}, \\ \bar{Z} \otimes I &\to \bar{Z} \otimes I, \\ I \otimes \bar{X} &\to I \otimes \bar{X}. \end{aligned}$$

The issue of Fault-tolerance with these logical operations should be clear. The $\bar{X}, \bar{Z}, \bar{H}$ and $\bar{R}_{\frac{\pi}{4}}$ gates are fault-tolerant since the logical operation is performed through seven single qubit gates. The logical CNOT is also fault-tolerant since each two-qubit gate only operates between the corresponding qubits in each logical block as shown in figure 3.4 . Hence if any gate is inaccurate, then at most a single error will be introduced in each block, and we are able to correct 1 error per block. A fault-tolerant CNOT is also a good example to show that the error probability goes from p to cp^2.

Nevertheless, the Clifford group is not an universal set of operation. The Gottesman-Knill theorem states that if a quantum circuit consists of only these elementary operations, the operation of the circuit can be efficiently simulated by a classical computer [11]. This means that if we want to construct a quantum computer that fully exploits the power of quantum computation, only aforementioned operations are not sufficient.

In order to achieve universality one of the following gates are generally added to the

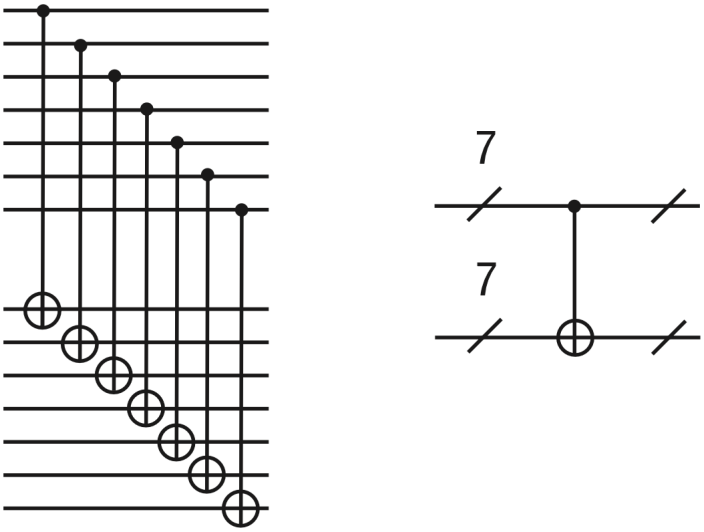


Figure 3.4: This figure shows a transversal fault-tolerant CNOT between two encoded qubits.

available set: $R_{\pi/8}$ or the Toffoli gate. Although both gates can be implemented in a fault-tolerant way through a more complicate procedure called "Magic state injection", in the following we will focus only on the fault-tolerant implementation of $R_{\pi/8}$ gate.

The basic idea of this procedure is based on the concept of teleportation, which is briefly described in the appendix. Let us begin by not considering about fault-tolerance, and simply attempt to perform some gate $\bar{U}$ on an encoded state. We will consider the case of a single-qubit gate $\bar{U}$ first. [12]

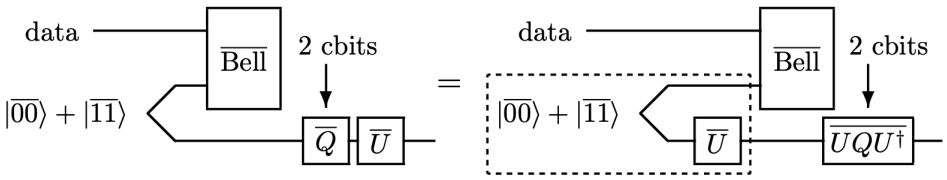


Figure 3.5: Gate teleportation of $\bar{U}$. The process of teleporting the state followed by $\bar{U}$ is the same as teleporting the state through a special ancilla with an appropriately modified correction operation.

Suppose we were to perform quantum teleportation (this means to apply the require pauli $\bar{Q}$ gates once we get the instructions from classical measurements) and then follow it, somehow, by an implementation of $\bar{U}$. Thus on input state $|\psi\rangle$, the overall output state would be $U|\psi\rangle$. Now imagine that the receiver (Bob), who controls the output block,

perform $\bar{U}$ earlier than intended, before the senders's (Alice) measurement outcome instructions. Eventually Alice performs the logical Bell measurement and sends Bob the two classical bits describing the outcome, corresponding to a logical Pauli $\bar{Q}$. To complete the teleportation procedure, Bob needs to do something different now, he can't do the same pauli operation as before: first, he must undo the $\bar{U}$ he performed prematurely, then perform $\bar{Q}$, and then finally redo $\bar{U}$, now in the correct place. That is, he should implement the gate $\bar{U}Q\bar{U}^\dagger$. This procedure is pictured in figure 3.5. It may not seem like we have gained anything by doing this, but for some special gates U, we have. We can imagine the state $(I \otimes \bar{U})(|\overline{00}\rangle + |\overline{11}\rangle)$ as a special ancilla state, a replacement for the EPR pair normally used in teleportation, and we can prepare it separately. Since it is a fixed ancilla state, independent of the data, we can apply some special tricks to prepare it.

We still have to perform the gate $\bar{U}\bar{Q}\bar{U}^\dagger$, which cannot be done ahead of time on the ancilla, since Q depends on the outcome of a logical Bell measurement on the data block. However, the gate $\bar{U}\bar{Q}\bar{U}^\dagger$ might be simpler to perform than $\bar{U}$ was. For instance, when $U \in \mathbf{C}_1, \bar{U}\bar{Q}\bar{U}^\dagger \in \mathcal{P}_1$ that is the defining property of the Clifford group. For some gates $U \notin \mathbf{C}_1$, it is nonetheless still true that $\bar{U}\bar{Q}\bar{U}^\dagger \in \mathbf{C}_1$ for any $\bar{Q} \in \mathcal{P}_1$. For instance, the $\pi/8$ rotation $R_{\pi/8}$ has this property:

$$R_{\pi/8} X R_{\pi/8}^\dagger = \begin{pmatrix} 0 & e^{-i\pi/4} \\ e^{i\pi/4} & 0 \end{pmatrix} = e^{i\pi/4} XP^\dagger$$
$$R_{\pi/8} Z R_{\pi/8}^\dagger = Z$$

We sometimes call the set of unitary operators with this property, of conjugating Pauli operators into Clifford group operators, $C_3.C_1$ is the Pauli group $\mathcal{P}_n$, and C_2 is the Clifford group $\mathcal{C}_1$. One can define a set $C_k = \left\{U \mid UQU^\dagger \in C_{k-1} \forall Q \in C_1\right\}$, and the teleportation construction tells us how, given appropriate ancilla states, to perform a gate from C_k once we know how to perform gates from C_{k-1}.

This whole procedure gives us an indication of how to perform a universal set of fault-tolerant gates. For the 7-qubit code, and some similar CSS codes, we already know how to perform all logical Clifford group operations. The Bell measurement is a Clifford group operation, and now we have seen that $R_{\pi/8} Q R_{\pi/8}^\dagger$ is also a Clifford group operation for $Q \in \mathcal{P}$.

The correction required after teleportation is a Clifford group gate, for which we already know a fault-tolerant procedure. This yield to the possibility of having universal quantum computation procedures in a complete fault-tolerant way.

3.2 Threshold theorem

The threshold theorem is one of the most remarkable results in fault-tolerant computation. It says that if the error rate of a physical system is below some threshold value, arbitrarily long reliable fault-tolerant quantum computation is possible.

One possible way to prove the threshold theorem and build a reliable quantum computation is by using fault tolerant protocols and concatenating codes. If an error occurs during a logical gate operation, then fault-tolerance ensures that this error will only propagate to at most one error in each encoded block, after which a cycle of error correction will remove the error. Hence if the failure probability of unencoded qubits per time step is p, then a single level of error correction will ensure that the logical step fails only when two (or more) errors occur. Hence the failure rate of each logical operation, to leading order, is now $p_L = cp^2$, where p_L is the failure rate (per logical gate operation) of a first level logical qubit, and c is the upper bound for the number of possible 2-error combinations which can occur at a physical level within the circuit[1]. Hence, thanks to fault tolerance protocols we can decrease the error rate from $p \to cp^2$.

Then, we can reduce the effective error rate achieved even more by concatenation of codes. This idea replaces each physical qubit at layer k by an encoded qubit. An example of concatenation using the bit-flip code is shown in figure 3.6 [2].

If we perform error correction at each layer of concatenation, then if the bare qubit experiences an error probability of p, then after one layer of error correction the logical error probability is cp^2, and after two layers, the logical error probability is $c\left(cp^2\right)^2$ (since two blocks have to fail and each block has failure probability cp^2). With each concatenation layer, even though the number of qubits is grows exponentially in k, we get a loss in error probability. That is:

$$\begin{array}{ccccccccc} \text{level } 0 & & \text{level } 1 & & \text{level } 2 & & \dots & & \text{level } k \\ p & \to & cp^2 & \to & c^{-1}\left(cp^2\right)^2 & \to & \dots & \to & c^{-1}(cp)^{2^k} \end{array}$$

And we are guaranteed that the error probability decreases with level of concatenation

[1] In this specific case c is the number of possible 2-error combinations, because we used a QECC that can correct 1 error. If we use different QECC codes that corrects t-errors the upper bound c becomes: $c = \begin{pmatrix} G \\ t+1 \end{pmatrix}$ where G is the total number of gates.

In addition, the logical step fails only when $t+1$ (or more) errors occur. Hence, the failure rate of each logical operation gate, to leading order, is $p_k = cp^{(t+1)}$

[2] Of course the bit-flip code is insufficient in general since it doesn't correct phase errors. However, everything we shall demonstrate with the bit-flip code also holds for more general quantum codes, as the 7 qubit code.

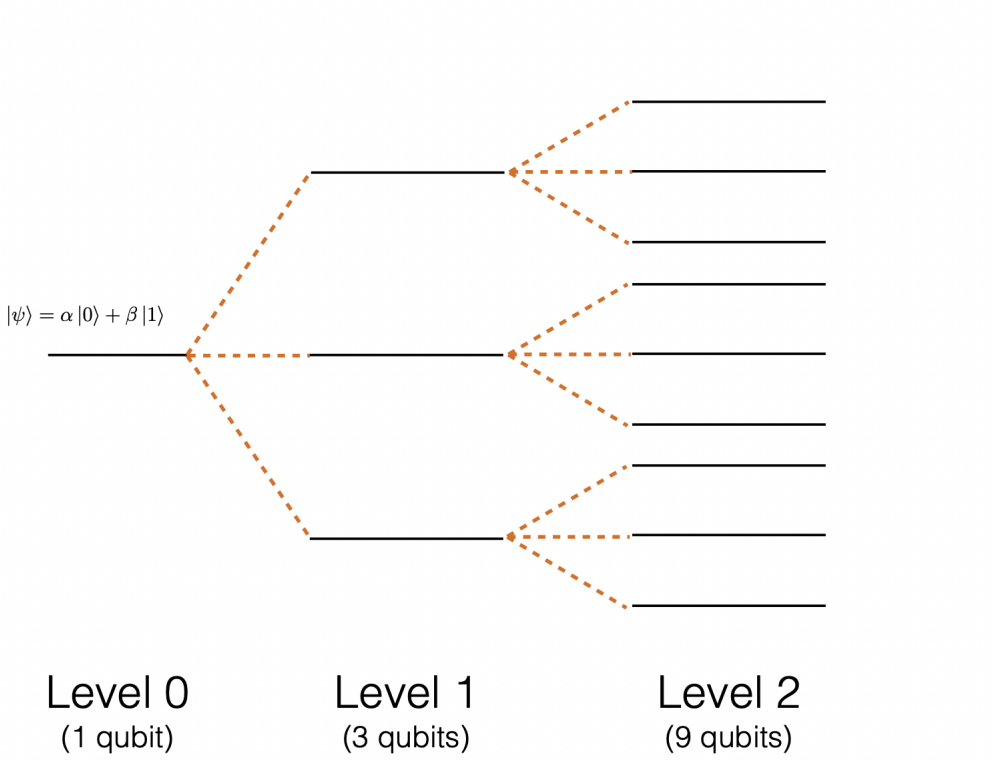


Figure 3.6: An illustration of concatenation of the bit flip code. Each qubit at level k is encoded into three qubits at level $k+1$.

if

$$p > cp^2. \tag{3.4}$$

Now let us look at the cost of doing this concatenated coding. If the size of the original (unencoded) circuit we want to execute has T gates, and the cost of implementing each gate at the logical level (this depends on the code being used) is at most N physical gates, then the number of gates needed at level 1 of concatenation is at most NT. And for each level of increasing concatenation we must use N times as many gates:

$$\begin{array}{ccccccccc} \text{level } 0 & & \text{level } 1 & & \text{level } 2 & & \ldots & & \text{level } k \\ T & \rightarrow & NT & \rightarrow & N^2T & \rightarrow & \ldots & \rightarrow & N^kT \end{array}$$

Note that the number of gates needed grows exponentially in k, but recall that the error probability reduced doubly exponentially in k. This suggests that we can win with this approach. But to see this convincingly, let us see how many gates we need to

achieve a given error probability in the computation. Suppose we wish to achieve a final accuracy of $\epsilon \geq 0$ in our simulation of our algorithm. Then we want that the total error rate is below ϵ:

$$Tp_k = Tc^{-1}(cp)^{2^k} < \epsilon,$$

for some concatenation level k. Here T is the number of logical quantum gates needed to perform the computation (which is the same as the number of gates necessary at the unencoded level), and p_k is the error probability per logical gate at concatenation level k.

We can solve for k to get

$$\begin{aligned}(cp)^{2^k} &< \frac{\epsilon c}{T} \\ 2^k &< \log_{cp}\left(\frac{\epsilon c}{T}\right) = \frac{\log_2\left(\frac{\epsilon c}{T}\right)}{\log_2(cp)} \\ k &< \log_2\left[\frac{\log_2\left(\frac{\epsilon c}{T}\right)}{\log_2(cp)}\right].\end{aligned}$$

From this upper bound on the concatenation level we can also form an upper bound the number of physical gates we will need to achieve this level ϵ of logical error.

We calculated above that at concatenation level k we need $N^k T$ gates. Then,

$$N^k T < N^{\log\left(\frac{\log\left(\frac{\epsilon c}{T}\right)}{\log(cp)}\right)} T.$$

Using $\log_a x = \frac{\log_b x}{\log_b a}$, we simplify

$$N^{\log(\cdot)} = \left(N^{\log_N(\cdot)}\right)^{\frac{1}{\log_N 2}} = (\cdot)^{\log N}$$

Then, the number of gates needed is upper bounded by

$$\#\text{gates} = T\left(\frac{\log\left(\frac{T}{\epsilon c}\right)}{\log\left(\frac{1}{cp}\right)}\right)^{\log N} \sim O(T\,\mathrm{poly}(\log T/\epsilon)).$$

The original T gates are scaled by the factor in braces. This factor is a polynomial in the log (is polylog) in $\frac{1}{\epsilon}$ (inverse error) and in T (the number of gates in the unencoded computation). Therefore the cost of achieving arbitrary error by concatenation scales very favorably, and this is the power of the fault tolerance theorem: we can achieve an

arbitrary error probability ϵ with resources that scale only polylogarithmically in the inverse of the desired error and the complexity of the computation to be performed (T).

Now, let us go back to Eq. **??**, which defines the crucial condition for concatenation to be effective: $p > cp^2$. We can compute the value of p that is at the boundary defined by this condition, $p = cp^2$. This value of p is called the threshold probability, and is easily seen to be

$$p_{\text{th}} = c^{-1},$$

because we want that the error probability decreases further the concatenation goes on. Thus if the physical error probability is below this threshold probability $p < p_{\text{th}}$, then concatenation and error correction is beneficial. Finally, writing the logical error probability at concatenation level k in terms of p_{th} gives us

$$p_k = c^{-1}(cp)^{2^k} = p_{\text{th}} \left(\frac{p}{p_{\text{th}}} \right)^{2^k}.$$

So what is the value of p_{th}? This depends heavily on the type of error correction code used, because c depends on the the type of constructions used for gates, measurements, state preparation, etc. Nevertheless, a rough estimate for the threshold probability for the Steane code is given in [13] and the threshold is $p_{th} \approx 10^{-4}$. Thus, if we can get the error rate per gate below this value we can do arbitrary long reliably quantum computation. So if quantum computing architectures can get their fundamental errors (in gates, measurements, state preparation, and idle periods) down to this level, then theory comes in and we are in a safe space to compute.

A more detailed derivation of the threshold theorem can be found in [10]

An active research field in quantum computing is to find codes that will increase p_{th} to larger values.

During the years has been developed, some much more complicated quantum error codes, such as the survace or color codes, which use intensively concept from QEC and topology. These codes also have had a lot of positive feedback and opens up to a larger class of codes and can operate under an higher threshold.

Chapter 4

Conclusions

We have seen that during quantum computation decoherence and errors are inevitable due to the noise and very often if no protocol for error correction is applied they can lead to incorrect results.

In fact, in the first chapter, I showed what are the causes of the decoherence and source of errors, after a brief introduction to quantum computing. In addition, I showed some basic tools that are helpful to get a good analysis of the noise such as CPTP maps and fidelity.

Then, I showed that in quantum mechanics is impossible to clone an arbitrary state, preventing us from using the same "copy and paste" technique that is widely used in classical error correction.

In order to approach some quantum error-correcting codes and techniques, I started from the 3 qubit code as an example to show the basic principles in quantum error correction followed by the conditions under which it can create a quantum error correction code.

However, life is not so simple such as the 3 qubit code, more powerful tools have to be developed when the numbers of qubits start rising. I showed the basic principle of the stabiliser formalism and then showed some more codes that prevent the computation to fail.

In the last chapter, I showed what is fault-tolerant computation and why it is important to achieve it. In addition, I presented one of the most important results of fault-tolerant: the threshold theorem, which states that is possible to have a reliable quantum computation if the error rate of a physical system is below a threshold value. These are only the basics of this big field in quantum computing and not only, but further applications and new codes have also been developed during the years, one of the

most promising quantum error correction codes are the topological codes, which have been thought of first by Kitaev in 1997.

In the end, I have to quote the 'Quantum Error Correction Sonnet' written by Daniel Gottesmann:

We cannot clone, perforce; instead, we split
Coherence to protect it from that wrong
That would destroy our valued quantum bit
And make our computation take too long.

Correct a flip and phase – that will suffice.
If in our code another error's bred,
We simply measure it, then God plays dice,
Collapsing it to X or Y or zed.

We start with noisy seven, nine, or five
And end with perfect one. To better spot
Those flaws we must avoid, we first must strive
To find which ones commute and which do not.

With group and eigenstate, we've learned to fix
Your quantum errors with our quantum tricks.
– 'Quantum Error Correction Sonnet', by Daniel Gottesman

Appendix

No-cloning theorem proof

Theorem 4. *There is no unitary operator U on $\mathcal{H} \otimes H$ such that for all normalised states $|\psi\rangle_A$ and $|s\rangle_B$ in H*

$$U\left(|\psi\rangle_A|s\rangle_B\right) = e^{i\alpha(\psi,s)}|\psi\rangle_A|\psi\rangle_B,$$

for some real number α depending on ψ and s.

PROOF: Suppose we have two quantum systems A and B. We start from an unknown but pure quantum state in A, $|\psi\rangle_A$. This is the state which is going to be copied in B. We assume that the target slot starts out in some standard pure state, $|s\rangle$. Thus, the initial state is:

$$|\psi\rangle \otimes |s\rangle.$$

Then, we want an unitary time-evolution operator U that effects the copying procedure, ideally,

$$|\psi\rangle \otimes |s\rangle \xrightarrow{U} U(|\psi\rangle \otimes |s\rangle) = e^{i\alpha_\psi}|\psi\rangle \otimes |\psi\rangle.$$

It looks like that we actually performed the copying procedure for the state $|\psi\rangle$, however, this procedure has to work for any state and not only for a specific one. Suppose then, this copying procedure works for two particular pure states, $|\psi\rangle_A$ and $|\varphi\rangle_A$. Then we have

$$\begin{array}{c} U(|\psi\rangle \otimes |s\rangle) = e^{i\alpha_\psi}|\psi\rangle \otimes |\psi\rangle, \\ U(|\varphi\rangle \otimes |s\rangle) = e^{i\alpha_\varphi}|\varphi\rangle \otimes |\varphi\rangle \end{array}.$$

The inner product of these two equations gives

$$\begin{array}{c} \langle s|\,\langle\varphi|\,U^\dagger U|\psi\rangle|s\rangle = e^{i(\alpha_\psi - \alpha_\varphi)}(\langle\varphi|\,\langle\varphi|)(|\psi\rangle\,|\psi\rangle), \\ \langle\psi \mid \varphi\rangle = e^{i(\alpha_\psi - \alpha_\varphi)}(\langle\psi \mid \varphi\rangle)^2. \end{array}$$

We can then take the absolute value of the last expression, and the phase factor vanishes:

$$|\langle\psi \mid \varphi\rangle| = |\langle\psi \mid \varphi\rangle|^2.$$

But, $x = x^2$ has only two solutions, $x = 0$ and $x = 1$, so either $|\psi\rangle = e^{i\gamma}|\varphi\rangle$ or $|\psi\rangle \perp |\varphi\rangle$. This means that we can theoretically clone only a precise state and its orthogonal states. However, this cannot be the case for two arbitrary states. Therefore, a single universal U cannot clone a general quantum state. This proves the no-cloning theorem.

■

Quantum Teleportation

In the teleportation protocol, the two parties (guess who) share the entangled Bell State and it is implemented via a CNOT gate between the state to be sent (suppose Alice is sending the state) and the part of entangled Bell state Alice has. The CNOT gate creates another entangled state whose measurement Alice will send to the second party, say it's Bob, to perform the measurements accordingly. So if you assume the state to be sent is 1 qubit state after CNOT you will have a 3 qubit state (as Bell sate is 2 qubit state). Now Alice will measure the middle qubit which will be the part of classical information she will communicate to Bob.

So, you see the controlled-NOT acts as an entangling operator without affecting the first qubit (here $|\psi\rangle$) and changing the entangled Bell state and hence its measurement outcome. A general description of this cool phenomena is shown in figure: Quantum

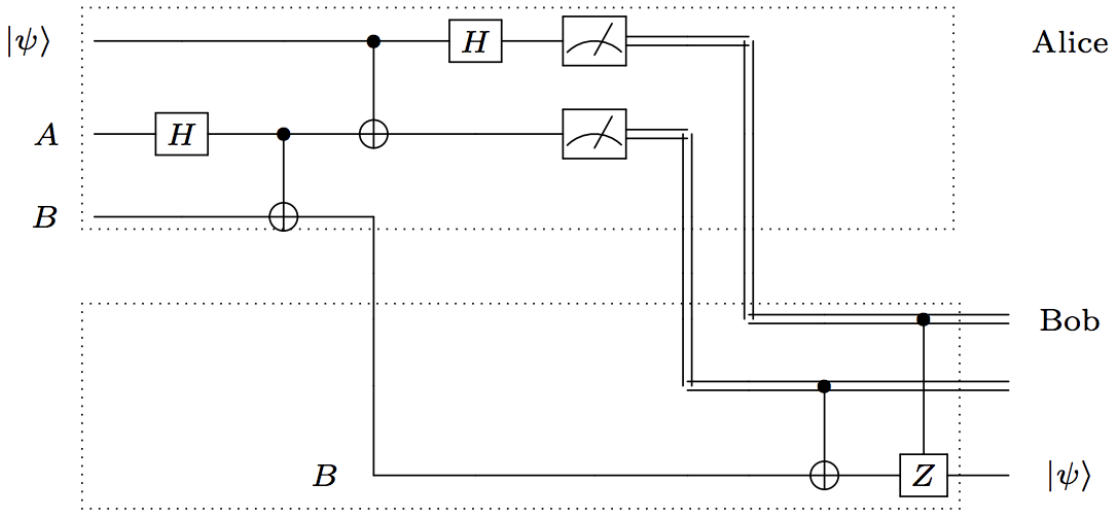


Figure 4.1: Basic circuit which implements quantum teleportation

teleportation is very useful in quantum communication because it allows to send classical reliable bits instead of a qubit.

Bibliography

[1] Franco Dalfovo. *Lecture videos and notes of Quantum mechanics course*. 2020/21.

[2] Julia Kempe. *Approaches to Quantum Error Correction*. 2006. arXiv: quant-ph/0612185 [quant-ph].

[3] Emanuel Knill, Raymond Laflamme, and Lorenza Viola. "Theory of Quantum Error Correction for General Noise". In: *Physical Review Letters* 84.11 (2000), 2525–2528. ISSN: 1079-7114. DOI: 10.1103/physrevlett.84.2525. URL: http://dx.doi.org/10.1103/PhysRevLett.84.2525.

[4] W. K. Wootters W. H. Zurek. "A single quantum cannot be cloned". In: *Nature* 299 (Oct. 1982), 802–803. DOI: https://doi.org/10.1038/299802a0.

[5] Vladimir Buzek and Mark Hillery. *Universal optimal cloning of qubits and quantum registers*. 1998. arXiv: quant-ph/9801009 [quant-ph].

[6] Andrew Steane. "A Tutorial on Quantum Error Correction A Tutorial on Quantum Error Correction 2". In: 162 (Jan. 2006). DOI: 10.3254/1-58603-660-2-1.

[7] MD SAJID ANIS et al. *Qiskit: An Open-source Framework for Quantum Computing*. 2021. DOI: 10.5281/zenodo.2573505.

[8] Daniel Gottesman. *Stabilizer Codes and Quantum Error Correction*. 1997. arXiv: quant-ph/9705052 [quant-ph].

[9] Todd A. Brun. *Quantum Error Correction*. 2019. arXiv: 1910.03672 [quant-ph].

[10] Daniel Gottesman. *Quantum Error Correction, Lecture notes*. https://www2.perimeterinstitute.ca/personal/dgottesman/QECC2007/. 2007.

[11] Daniel Gottesman. *The Heisenberg Representation of Quantum Computers*. 1998. arXiv: quant-ph/9807006 [quant-ph].

[12] Daniel Gottesman. *An Introduction to Quantum Error Correction and Fault-Tolerant Quantum Computation*. 2009. arXiv: 0904.2557 [quant-ph].

[13] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge Library Collection. Cambridge University Press, 2010. DOI: 10.1017/CBO9780511976667.

[14] Philipp Hans Juergen Hauke. *Lecture videos and notes*. 2020/21.

[15] Simon Devitt, William Munro, and Kae Nemoto. "Quantum Error Correction for Beginners". In: *Reports on progress in physics. Physical Society (Great Britain)* 76 (June 2013), p. 076001. DOI: 10.1088/0034-4885/76/7/076001.

[16] Joschka Roffe. "Quantum error correction: an introductory guide". In: *Contemporary Physics* 60.3 (2019), 226–245. ISSN: 1366-5812. DOI: 10.1080/00107514.2019.1667078. URL: http://dx.doi.org/10.1080/00107514.2019.1667078.

[17] Philipp Kammerlander. *Quantum Information Processing Concepts, Lecture notes*. 2021.

[18] Daniel Gottesman and Beni Yoshida. *IQC - Quantum Error Correction - Curse*. en. Lectures see, https://pirsa.org. 2018. URL: https://www2.perimeterinstitute.ca/personal/dgottesman/QECC2018/index.html.