Universidade de São Paulo

Instituto de Física de São Carlos

Informação Quântica / Quantum Information

Quantum error correction algorithms - a theoretical overview

Alexandre de Taunay Voloch (12557532)

July 2024

Abstract

Quantum error correction (QEC) is vital in advancing the reliability and feasibility of quantum computing. In this article, we provide an overview of the fundamental quantum mechanical concepts behind quantum error correction algorithms, as well as explain their role in protecting quantum information against decoherence and operational imperfections. We begin with a basic introduction to quantum mechanics and the inherent properties of qubits, before delving into classical and quantum logic gates and the error correcting algorithms. Finally, we discuss recent advances in the field in significantly reducing error rates, and possible future developments.

This article was written for the undergrad course Informação Quântica (Quantum Information) - 7600074, taken by the author in the first semester of 2024.

Contents

1 Basic introduction to quantum mechanics

Quantum mechanics is the field of physics which aims to understand and deal with physical properties of material reality which appear in very small (atomic-level) scales. Actually, this is not a proper characterization, since there are many macroscopic quantum effects (e.g. superconductors, Bose-Einstein fluid, lasers etc). However, in the context of quantum computing, especially its theoretical modeling, practically all the physics is in very small microscopic scales, and so that is what we will focus on here. The main system used to model quantum bits (hereafter called qubits) is that of a two-level system (i.e. a 2-dimensional Hilbert space). Normally, we model this as a spin-1/2 particle (the meaning of which will be explained below), however it can, and will be proven that all two-level systems can be modeled using the Pauli matrices which appear when studying a spin-1/2 particle such as an electron.

Let us begin with the basics of quantum mechanics. The Schrödinger equation says that, for a given quantum state $|\psi(t)\rangle$, its evolution along time is given by

$$i\hbar\frac{\partial\,|\psi(t)\rangle}{\partial t} = H(t)\,|\psi(t)\rangle \tag{1}$$

Where

$$|\psi(t)\rangle = U(t,t_0)\,|\psi(t_0)\rangle \tag{2}$$

$U(t,t_0)$ being the propagator or evolution operator of the system; in the case of a Hamiltonian which does not depend on time, it is of the form $U(t,t_0) = e^{\frac{-i}{\hbar}H(t-t_0)}$. However, this is not generally the case, especially in systems of interest for quantum computing, in which we desire to apply specific potential "pulses", which vary over time, in order to achieve a specific result.

1.1 Spin and two-level systems.

The general formula for turning a classical problem into its quantum equivalent is [1] to replace the independent variables x and p of classical mechanics with their quantum correspondents, the Hermitian operators X and P, which satisfy the commutation relation $[X,P] = XP - PX = i\hbar$.

When applying this treatment for the angular momentum $\vec{L}$, which is (in classical as well as quantum mechanics) the generator of rotations in space, it is found that any generator of rotations, in order to be valid in quantum mechanics, must obey the equation

$$\vec{L}\times\vec{L} = i\hbar\vec{L} \tag{3}$$

Of course, it makes no sense to think of $\vec{L}$ as purely a vector - it is actually an operator. Otherwise the vector product of it with itself would always result in 0. Now, the above equation is extremely insightful and useful, for it shows that the components of the angular momentum operator (i.e. L_x, L_y, L_z) do not commute with each other. Specifically,

$$[L_x, L_y] = i\hbar L_z \tag{4a}$$
$$[L_y, L_z] = i\hbar L_x \tag{4b}$$
$$[L_z, L_x] = i\hbar L_y \tag{4c}$$

This makes us able to define the raising and lowering operators L_+ and L_-, such that $[L_+, L_-] = 1$. Furthermore, we also have that L^2 and L_z commute, therefore, we can try to find eigenstates of the type $|\alpha\beta\rangle$ which are eigenstates of both those operators. In doing so, using the algebra which we have just stated, we come to the following eigenvalues:

$$L^2 |\alpha\beta\rangle = \hbar^2 l(l+1) |\alpha\beta\rangle \tag{5a}$$
$$L_z |\alpha\beta\rangle = \hbar m |\alpha\beta\rangle \tag{5b}$$

Where l, m are half- (or full-) integers, $l \geq 0$, and $l \geq m \geq -l$.

Now, what is interesting about this solution is that the specifically semi-integer values of l (i.e. $\frac{1}{2}, \frac{3}{2}, ...$) cannot be arrived at through any solution for a spacial rotation problem, where an object is rotating through space, with spatial coordinates attached to this rotation. This is, of course, the only way that we know to think about rotation, since that is how classical objects rotate. Attached to these semi-integer values of l is a new property of matter called *spin*, which has no classical analogue and is an intrinsic characteristic of certain particles, such as electrons.

It turns out that a specific type of particle can only have one value of l, and that electrons have $l = \frac{1}{2}$. This is what we call a spin-1/2 particle. Now, since m is between $-l$ and l, it can, in this case, only have two values, that is $\pm\frac{1}{2}$. It is what we call a **two-level system**.

It is obvious why two-level systems would be extremely useful for quantum computation. Classical computers work using bits, which represent a binary state of a system (either on or off). Arranging many bits in parallel, it is possible to store arbitrarily large information using binary encoding. In order to process this information, logic gates are used. This is the basis of classical computing. Now, in order to extend this logic to a quantum computer, if we are to have any analogue at all to the classical computing bit, the obvious thing to do is to use two-level systems, which can be in one of two states, analogous to the "on" or "off" of classical bits.

In the context of spin, the eigenstates of spin-1/2 S^2 and S_z are the kets $\left|\frac{1}{2},-\frac{1}{2}\right\rangle, \left|\frac{1}{2},+\frac{1}{2}\right\rangle$, in the $|s,m\rangle$ basis. It is customary to denote these as either $|0\rangle, |1\rangle$, or $|-\rangle, |+\rangle$ or $|\downarrow\rangle, |\uparrow\rangle$. Measurements of S_z will result in $\pm\frac{\hbar}{2}$, according to eq. (5b).

It is convenient to use as basis these eigenkets of S_z. In this basis, each of the components of $\vec{S}$, that is, S_x, S_y, S_z and its amplitude S^2 are expressed using multiples of the Pauli matrices:

$$S_x = \frac{\hbar}{2}\sigma_x = \frac{\hbar}{2}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{6a}$$

$$S_y = \frac{\hbar}{2}\sigma_y = \frac{\hbar}{2}\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{6b}$$

$$S_z = \frac{\hbar}{2}\sigma_z = \frac{\hbar}{2}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{6c}$$

Of course, $S^2 = \frac{3\hbar^2}{4}\mathbb{I}$. Now, these four matrices, with $\sigma_0 = \mathbb{I}$, form a basis for generators of the SU(2) group. In particular, they form a basis for any hermitian 2x2 matrix, that is, any measurable operator in a two-level system. Thus, **all two-level systems** can be modeled using just the σ_i operators we have just discovered from spin. This is very important, since it means that any algebra we perform using a theoretical model of spin-1/2 particles, can also be implemented on any platform which uses two-level systems or anything analogous to spin-1/2, like quantum dots (semiconductor nanocrystals), photon polarization (left or right polarization), or trapped ions.

In the context of quantum computing, we usually assign to the lower energy level of the system the ket $|0\rangle$, and to the higher energy level, $|1\rangle$.

Now let us analyze some of the interesting effects which arise from the quantum mechanical description of qubits and discuss how we might take advantage of them for quantum computing.

1.2 Superposition

The main effective or operational difference between classical and quantum mechanics is that the latter allows for wave-like behaviours in particle-like objects (like electrons), and from this arises the phenomena of superposition and interference. Since the only requirement for kets is that they be members of their specific Hilbert space (i.e. that they be vectors in that space), any linear combination of multiple kets is also a ket belonging to that same vector space. To illustrate: a classical bit has only two possible states, which we might represent by $|0\rangle$ and $|1\rangle$. There cannot be a classical "superposition" of these two states, whatever that would mean - a light can either be on or off, a bit can either be 1 or 0.

A qubit, however, need not submit to these rules and have a state such as

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

where α and β are complex numbers that describe the probability amplitudes of the respective states. Thus, on applying a logic gate (an operator) to $|\psi\rangle$, we would be able to compute the output on both $|0\rangle$ and $|1\rangle$ simultaneously. This is a simple example since we are only operating on one qubit. Imagine, however a state $|\psi\rangle$ which is a superposition of N other basis kets:

$$|\psi\rangle = \sum_n |\phi_n\rangle$$

We would then theoretically be able to perform parallel computation on all N kets simultaneously. This is extremely useful, though not as miraculous as it initially seems, since we can only measure a specific state of the qubit (either 0 or 1), not a superposition. However, since the output of our measurement will be statistically influenced by the processing we have performed on $|\psi\rangle$, we can still, over many measurements, generate useful results, especially utilizing another of QM's distinctive characteristics, **interference.**

1.3 Interference

Due to the principle of superposition, the probability amplitudes α and β of different states can interact with each other, creating patterns of interference. This is, again, in stark contrast to classical mechanics. A very famous example is that of the double-slit experiment, which can be realized both for light (which was already known to be wave-like by the early 20th century) and individual electrons[2] (which conclusively proves their wave-like properties.) The wave function of the electron at different locations will interact with itself and generate an interference pattern in the collection area of the experiment. This is, of course, very different to what one would expect for a point-like "particle", which would have no interference. See figure 1.

For example, if we had two qubits $|\phi_0\rangle = \alpha |0\rangle + \beta |1\rangle$ and $|\phi_1\rangle = \alpha' |0\rangle - \beta |1\rangle$, then

$$|\phi_0\rangle + |\phi_1\rangle = (\alpha + \alpha') |0\rangle + (\beta - \beta) |1\rangle = (\alpha + \alpha') |0\rangle$$

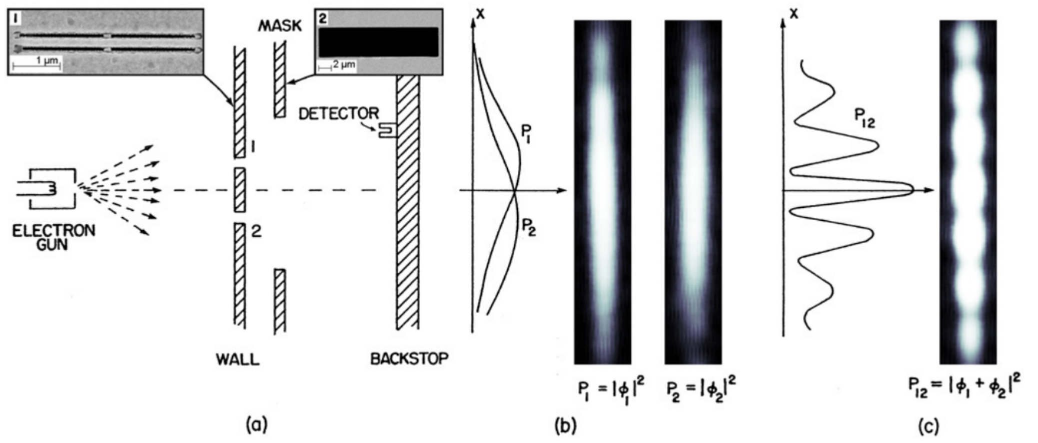


Figure 1: Double-slit experiment. (a): experimental setup. (b): expected pattern for each of the slits. (c) actual pattern, which is not a superposition of the patterns in (b) but totally different because of the effect of interference. Source: [2]

Which means that the probability amplitudes for $|1\rangle$ canceled out due to interference! If we were to measure some sort of sum of these two kets, we would never measure the qubit to be in the state $|1\rangle$. This is the magic of interference.

If we exploit both these features, we can generate algorithms much faster than their classical counterparts, for example, the Grover algorithm [3] for unstructured search, and the Shor algorithm [4] for integer factorization, which utilize heavily both superposition and interference in order to isolate probabilistically the desired result and let other probabilities go to 0, just like in the simple example given above.

1.4 Entanglement

There is one more feature to analyze and that is of quantum entanglement. It occurs, basically, because of the collapse of the wave function due to measurement [1]. There is a great deal of (philosophical) controversy around this phenomenon, especially since it allows the breaking of causality. However, we will not touch on the controversy and analyze only the phenomenon itself. The postulate of the collapse of the wave function says that, upon measuring a variable (represented by a Hermitian operator Ω) of a wave function $|\psi\rangle = \sum_n c_n |\omega_n\rangle$, the same immediately "collapses" (losing its superposition in the $\{|\omega_n\rangle\}$ basis, basically) and assumes the value of $|\psi'\rangle = |\omega_i\rangle$ (save a global phase of modulus 1), assuming that the corresponding eigenvalue ω_i

was obtained during measurement.

Now, suppose that we have two qubits and the total wave function (using the tensor product to create a new $2^2 = 4$-dimensional Hilbert space) is

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$$

This arrangement can be created using many different techniques and arises naturally through many quantum processes. For example, the EPR[5] or EPR-Bohm[6] formulation of entanglement. Now, let us suppose that both qubits or particles are very far away from each other. Let us measure qubit 1. It returns the value corresponding to $|0\rangle_1$. This means that the wave-function (for both particles) has now collapsed, since the only possible state for particle 2, since we have measured 0 in particle 1, is for it to be in state $|1\rangle_2$. So, although we did not touch particle 2 at all, it was still influenced directly by the measurement we did on 1, and had its wave-function collapsed - even if we measured 2 before the corresponding time that light (or information) from particle 1 could arrive, which contradicts the principle of relativity.

The initial and natural response to this is that quantum mechanics must somehow be incomplete and that there are hidden variables that each particle had at the moment of entanglement which determined that they would, in fact, only have the state $|01\rangle$ and not the opposite. A hidden-variable theory can be tested through the Bell inequalities [7] and has been proven to be untenable if it preserves locality.

1.5 The density operator

Up to now, we have only talked about isolated systems with no interaction with the environment. This is, however, a gross idealization and any real model for a quantum computation must take into account the environmental variables which affect the system of interest. We accomplish this through use of the density operator.

The density operator, or density matrix, generalizes the concept of a state vector and is crucial for describing qubits in mixed states, which represent a statistical ensemble of different possible states. This is described mathematically as

$$\rho = \sum_i p_i |\psi_i\rangle \langle\psi_i| \tag{7}$$

Where p_i are the probabilities associated with each $|\psi_i\rangle$ in the ensemble, and are real positive numbers and $\sum_i p_i = 1$.

The density operator is useful in two major cases[1]: first, when the experimental preparation of the system is such that there are numerous possible states $|\psi_i\rangle$ in the ensemble; and second, when two physical systems are entangled with one another, and it is desirable to describe only one of the physical systems while disregarding the other. This is the case of, for example, interaction of a quantum system with its "environment" (which, of course, is also a quantum system - only one that we choose to disregard.) In real-world quantum systems, qubits often exist in mixed states due to interactions with the environment or incomplete knowledge about the state, making the density operator essential for predicting system behavior and implementing error correction.

Density operators have analogues to all the properties of normal state vectors like kets. They also have, however, some properties of their own which are quite useful for our purposes.

A **pure state** is one where all the p_i's in eq. 7 are 0, except for one which is 1. This means that the ensemble actually only has one state: $\rho_{pure} = |\psi\rangle\langle\psi|$ The purty of a density operator can be measured by taking the trace of its square. A pure state has purity 1:

$$\mathrm{Tr}\left(\rho_{pure}^2\right) = 1$$

When a state is completely mixed, its purity is $\frac{1}{d}$ where d is the dimensionality of the Hilbert space[8].

A question which might arise is: what is the difference between a pure state with superpositions and a mixed state? The difference is that, in the first case, the density operator will have non-diagonal terms. For example, let $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. Then the density operator $\rho_{pure} = |\psi\rangle\langle\psi|$ associated with it will be

$$\rho_{pure} = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Whereas a maximally mixed state, $\rho = \frac{1}{2}|0\rangle\langle 0| + \frac{1}{2}|1\rangle\langle 1|$, would be represented matricially as

$$\rho = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Note that the diagonal terms have disappeared. This is a sign of a mixed state or of **decoherence**. Since we only have a mixed ensemble of non-superposed states, this means that we lose the effects mentioned in previous sections such as interference and superposition, and thus lose any quantum advantage we might have gotten from our computation. Decoherence specifically is what happens to the density matrix of a system as it interacts with an environment over time. The interactions causes the system to gradually transition from its original superposed state into an ensemble of states with basically only classical properties. This poses a significant challenge for maintaining the stability of quantum information in a quantum computer. Understanding and mitigating decoherence is vital for the development of real quantum computing technology, and will be touched upon later in this article.

2 Quantum gates and circuits

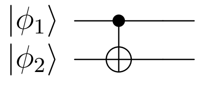
Classical computing utilizes bits and logic gates in specific arrangements, called circuits, to achieve a desired operation on the bits. Analogously, in quantum computing we utilize logic gates to perform operations on the qubits. However, quantum gates must differ from classical gates in a key aspect, and that is their reversibility.

Classical gates need not be reversible. For example, it is trivial to delete multiple bits or to clone one bit into many. This is not possible in quantum computation[9][10]. Furthermore, quantum logic gates must be described by unitary linear operators, meaning they must be reversible in time. This is necessary because only unitary operators preserve the magnitude of a vector over time, and if this were not the case there would not be conservation of probability, which is necessary for quantum mechanics.

Let us analyze some of the more elementary quantum logic gates, noting that they are all reversible (i.e. the same number of inputs as that of outputs, with one output corresponding exactly to one input.)

2.1 CNOT gate

The CNOT gate is analogous to the NOT gate of classical computing and is represented by the following diagram:



In this case, $|\phi_1\rangle$ is the control bit and $|\phi_2\rangle$ is the target bit. The effect of the CNOT gate is to flip the second qubit if the first qubit is $|1\rangle$, otherwise it leaves the second qubit unaffected. The CNOT gate can also be seen as the sum modulo 2 (XOR) operation between the two bits:

$$|\phi_1\rangle, |\phi_2\rangle \rightarrow |\phi_1\rangle, |\phi_1 \oplus \phi_2\rangle$$

And can be written matricially (in the $|\phi_1\phi_2\rangle$ basis) as

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

2.2 Hadamard Gate

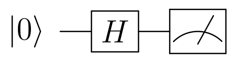
The Hadamard gate takes only one qubit and basically returns the corresponding eigenstate of σ_x, that is, performs a "rotation" on the qubit. It can be represented matricially as

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

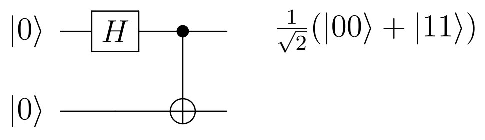
Note that

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

The Hadamard gate is written schematically as H. For example, in the circuit below, a qubit is operated on by H and then measured.



As an example, using the Hadamard and CNOT gates, it is possible to create an entangled system (a Bell state):

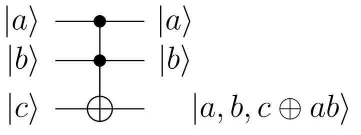


2.3 Toffoli and other gates

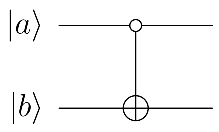
Using the notation we have just described, especially in regards to the CNOT gate, we can describe any number of different gates. For example, the Toffoli gate

$$|a, b, c\rangle \to |a, b, c \oplus ab\rangle$$

Can be drawn schematically as



Another gate (XNOR) which is the opposite of the CNOT gate in that it flips the second qubit only if the first qubit is 0, can be implemented using a σ_x gate (which flips $|0\rangle$ to $|1\rangle$ and vice-versa as can be easily inferred from eq. 6a) on the 1st bit and then performing a CNOT operation. It can be drawn schematically too as



Note the white dot on the control qubit in contrast to the black dots in the previous figures. More complex gates with black and white dots can be understood by returning to these basic figures. In further sections we will use these and the more complex gates as well.

3 Quantum error correction

3.1 Basic principles

Error correction is necessary in any field of computation due to inherent imprecisions in the physical substrate of the processor. In classical computing, the most common type of error is that of a bit flip, and it is by far the most relevant, since classical bits can basically only flip (they have no phase or superposition attached to them.) So, for a classical bit, the obvious technique to diagnose and remedy bit flip errors is to employ redundancy, that is, to make multiple copies of the information before it's processed, then to perform the (possibly error-inducing) processing on each of those copies, and then check at the end if there are any divergences. If so, pick whatever the majority of bits says is the correct bit.

For example, if there were five redundant copies of a classical bit, one was set to 0 and the other four to 1, the error correcting algorithm should decide that the correct value for that bit is 1, not 0. This is, of course, a very simple algorithm, but any more complex algorithm will also operate from the same basic principles. These operations - making copies of the information in the beginning, and checking (measuring) each bit at the end - are simply not possible in quantum computing without severely handicapping the system. The former (copying) is not possible using unitary linear operators, and so is not useful for any type of quantum computing. And the latter (measurement) would collapse the state vector and is also not useful since collapsing entails the loss of any superposition information stored in that qubit.

Furthermore, there are other characteristics of a qubit - such as the relative phase between its two components - which may be altered by errors. This is called a phase flip or phase change:

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

These are the two basic types of errors we may encounter - bit flips and phase flips. Of course, this is still within the ideal quantum computer. A real quantum computer will be subject to numerous other factors, such as decoherence over time (which is rendered as "noise" in the qubit's density matrix because of the partial trace over the environment), and other technical issues which arise from the specific implementation of the quantum computer. In the next section we will talk about simple code that corrects bit flips, show how this is implemented using the quantum gates discussed earlier, and then touch on code for phase flips and both at the same time (Shor's code). Then we will talk about more advanced strategies for error correction.

3.2 Bit flip correction

An example of bit flip correcting code is the three-qubit flip code which was introduced by A. Peres in 1985 [11]. This is the usual textbook error correcting code, because of its simplicity. Even though it is not that useful for real-life quantum computing, we will dedicate some time to it as it elucidates many aspects which are common to the more advanced algorithms.

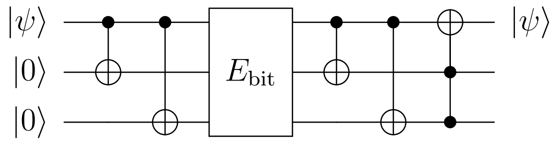


Figure 2: Peres' error correcting code for 3 qubits.

The general idea of the Peres algorithm is to encode a single qubit of information onto a highly entangled state across three qubits. Thus, and (single) bit-flip error can be corrected using the majority vote across the three. There is an additional (but optional) syndrome diagnosis code which is shown in some references and can be used to check which of the qubits was specifically flipped. Here, however, we will omit this intermediary portion since it is unnecessary.

The encoding portion of the algorithm is done using two CNOT gates. Assuming that the original qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, the transformation is

$$|\psi, 0, 0\rangle \to \alpha|000\rangle + \beta|111\rangle \tag{8}$$

Now, we assume that a bit flip error (i.e. σ_x) can occur on any one of these three qubits, but not on more than one. Then, we run the correcting code: two CNOT gates and one Toffoli gate. The two CNOT gates flip their respective target qubits if the first qubit is $|1\rangle$. In practice, there are four possible outcomes on the second and third qubits:

00	No error
01	Error on third qubit
10	Error on second qubit
11	Error on first qubit

Table 1: Possible outcomes for qubits 2 and 3 after the final two CNOT operations.

The final step is to use the Toffoli gate, which is better understood schematically - it flips the first bit only if the 2nd and 3rd bits are both 1, which corresponds to the last line on table 1. Otherwise, qubit 1 (which we are interested in) is not changed. We can also, if we so choose, measure qubits 2 and 3 in order to find out what error there was.

The algebra behind this problem is quite simple. We assume that a single bit flip can occur, independently of the others, with a probability p, meaning the qubit is unchanged with probability $1 - p$. The action of the E_{bit} error-generating gate on a single qubit can thus be written $E_{bit}(\rho) = (1-p)\rho + pX\rho X$, where $X = \sigma_x$ is the bit flip gate.

Now, after passing through the E_{bit} gate with our three-qubit system (eq. 8), we find that no qubit is flipped with probability $(1-p)^3$, one qubit is flipped with probability $3p(1-p)^2$, two with probability $3p^2(1-p)$, and all three with p^3. Of course, our error-correcting code is valid only for the first and second cases, returning incorrect results for the other two. We can thus create a function of its minimum fidelity, which can be shown to be

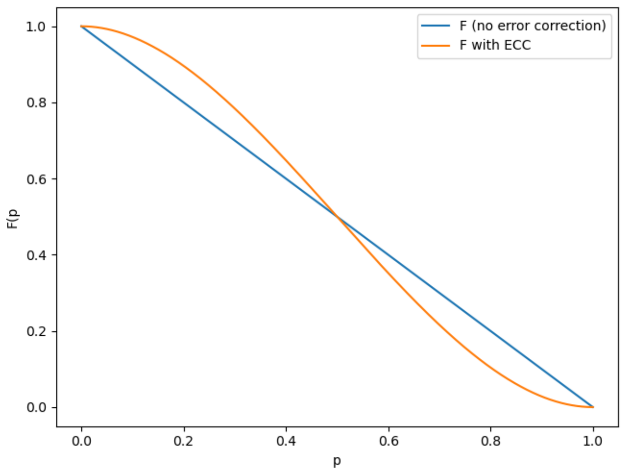


Figure 3: Plot of the different fidelities with and without error correction. Notice that, for $p > 0.5$, the error correction code is less faithful. However, for small $p < 0.5$, ECC is of benefit.

$$F(|\psi\rangle \geq 1 - 3p^2 + 2p^3 \tag{9}$$

Comparing this to the minimum fidelity without error correction at all, which would of course just be $1 - p$, we arrive at the very well-known chart seen in figure 3.

Thus ends our exposition of the single bit-flip correction code. Of course, this is a very simple and idealized situation, and real error correction algorithms used in actual realized quantum computers are much more advanced. We will talk about those now. First, though, let us analyze a similar algorithm for correcting phase flipping.

3.3 Phase flip and Shor's code

A phase flip (or sign flip) can be corrected using the above code by transforming the encoded bits into the σ_x basis of $|+\rangle, |-\rangle$ (by using a Hadamard gate) before and after entering them into the error-generating circuit, as can be seen in figure 4. Note that the error-generator is now E_{phase}, meaning that it generates only phase flips and not bit flips.

To see that a sign flip in the Hadamard basis is equivalent to a bit flip in the σ_z basis, one need only notice that the Hadamard basis qubits are $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$, which, under sign flip, become $\frac{1}{\sqrt{2}}(|0\rangle \mp |1\rangle)$, equivalent to $|\mp\rangle$

instead of $|\pm\rangle$ as was the case before the sign flip.

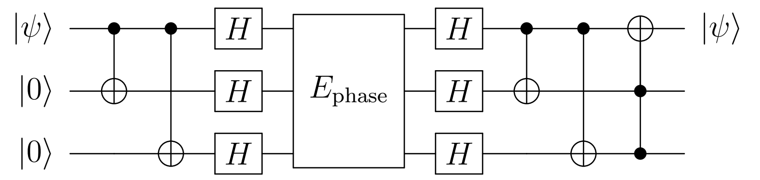


Figure 4: The sign flip correction code.

If the error channel can induce both bit or sign flipping, independently (one of each can occur, or both, or neither), then we can use Shor's error correcting code [12] to correct both types of errors. We will not explain this code in-depth. In Shor's original paper, to correct k qubits from both sign and bit flipping, we need $9k$ total qubits, $8k$ being used as ancillas. So, this is quite a costly error-correcting code, in terms of qubits and logic gates. The algorithm still assumes that the errors on each qubit are independent of each other - that is, decoherence occurs independently on each qubit. According to Shor, this is not an unreasonable physical assumption.

3.3.1 Steane, CSS codes and beyond

A generalization of the two codes we have just seen is the class of ECC called CSS codes, named after A. R. Calderbank, Peter Shor, and Andrew Steane. Steane himself came up with a similar bit- and phase-flip code with $7k$ qubits instead of Shor's $9k$. CSS codes are actually a mathematical class, which we will not be discussing technically since this already goes much deeper into the field of quantum computing than is the purpose of this article. It is sufficient to note that these types of codes are widely used in fault-tolerant quantum computing.

Beyond this point, the codes become much more technical and difficult to understand for the lay reader. For example, surface and topological codes. We will not touch upon these in great detail. However, they are where the cutting edge in quantum error correction lies, and we hope this article can lay the foundations for understanding those more complicated subjects in the future.

3.4 Physical vs. logical qubits

Most quantum computers today do not use single physical qubits for each "qubit"' of information but instead use many physical qubits in an error-correcting scheme. This is very useful since it promotes fault-tolerant quantum computing. In contemporary QC, most physical qubits consist either of superconducting materials, trapped ions, or photons, all of which possess high failure and error rates. However, it is possible to combine dozens or hundreds of these physical, fallible qubits into a single logical qubit with a much smaller error rate. This is, in practice, what most QC enterprises are trying to accomplish today. For more information, see [13], where the software companies Microsoft and Quantinuum have created four logical qubits out of thirty physical ones (using the Steane code we mentioned above), with error rates up to 800x lower than the corresponding physical error rates. However, the aforementioned article estimates that, for useful quantum computing results, tens or hundreds of logical qubits are needed, and so we are still quite far away from this.

4 Conclusion

We hope to have given, in this article, an overview of the more commonly known and utilized quantum error correcting algorithms, as well as the theoretical background necessary to fully understand them as well as quantum computing in general. As we have seen, these codes are not just theoretical constructs (as they may seem to be) but are actually essential for the realization of robust quantum computers. Despite significant progress, there is still a long way to go until we have scalable and fault-tolerant quantum computing. Hopefully, not too long until then!

References

[1] Ramamurti Shankar. *Principles of Quantum Mechanics*. 2nd ed. Vol. 2. Discussion of quantum postulates. Springer, 2012. Chap. 4.

[2] Roger Bach et al. "Controlled double-slit electron diffraction". In: *New Journal of Physics* 15.3 (Mar. 2013), p. 033018. ISSN: 1367-2630. DOI: 10.1088/1367-2630/15/3/033018. URL: http://dx.doi.org/10.1088/1367-2630/15/3/033018.

[3] Lov K. Grover. *A fast quantum mechanical algorithm for database search*. 1996. arXiv: quant-ph/9605043 [quant-ph]. URL: https://arxiv.org/abs/quant-ph/9605043.

[4] P.W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700.

[5] Albert Einstein, Boris Podolsky, and Nathan Rosen. "Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?" In: *Physical Review* 47.10 (1935), pp. 777–780. DOI: 10.1103/PhysRev.47.777.

[6] David Bohm. *Quantum Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1951.

[7] John S Bell. "On the Einstein Podolsky Rosen paradox". In: *Physics Physique Fizika* 1.3 (1964), pp. 195–200.

[8] Wikipedia contributors. *Density matrix — Wikipedia, The Free Encyclopedia*. [Online; accessed 29-June-2024]. 2024. URL: https://en.wikipedia.org/wiki/Density_matrix.

[9] William K Wootters and Wojciech H Zurek. "A single quantum cannot be cloned". In: *Nature* 299 (1982), pp. 802–803. DOI: 10.1038/299802a0.

[10] Arun K Pati and Samuel L Braunstein. "Impossibility of deleting an unknown quantum state". In: *Nature* 404 (2000), pp. 164–165. DOI: 10.1038/35004532.

[11] Asher Peres. "Reversible Logic and Quantum Computers". In: *Physical Review A* 32.6 (1985), pp. 3266–3276. DOI: 10.1103/PhysRevA.32.3266.

[12] Peter W. Shor. "Scheme for reducing decoherence in quantum computer memory". In: *Physical Review A* 52.4 (1995), R2493–R2496. DOI: 10.1103/PhysRevA.52.R2493.

[13] Moor Insights. *Microsoft and Quantinuum Improve Quantum Error Rates by 800x*. Forbes. Accessed: 2024-06-29. 2024. URL: `https://www.forbes.com/sites/moorinsights/2024/04/18/microsoft-and-quantinuum-improve-quantum-error-rates-by-800x/`.