A Basic Overview of Quantum Computing Fundamentals and Error Correction

A Preprint

Anthony Thonnard
Department of Computer Science
Cranberry-Lemon University
Pittsburgh, PA 15213
`hippo@cs.cranberry-lemon.edu`

Benjamin Dobbins
Department of Electrical Engineering
Mount-Sheikh University
Santa Narimana, Levand
`yo-mama@ee.mount-sheikh.edu`

November 8, 2023

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1 Qubits

Qubits are the fundamental units of information in quantum computers, and are essentially the quantum analogue of classical bits. Classical bits are represented as a 0 or a 1, denoted by the presence of an electrical current. They store binary information in a classical computer, and can be combined and abstracted to store more complex information. Qubits are similarly limited to two possible physical states, also described as 0 or 1, but because of their quantum mechanical properties, they can also be in a superposition of those two states. Consequently, while bits must only be a 0 or a 1 at all times (i.e. the electrical current is present or it is not), qubits can be in an unknown state, in which it cannot be determined whether the qubit is a 0 or a 1 unless it is being observed. This is useful because we are able to manipulate the probabilities of the qubit being in either state, and thus we can perform operations on the qubit that are not possible with classical bits. Furthermore, quantum entanglement can be applied to link multiple qubits together, creating a mutual dependence between their probabilities of taking each state. As a result, the computer's ability to make complex calculations in shorter amounts of time is exponentially increased.

1.1 Qubits and Probabilities

1.1.1 The Mathematical Representation of Qubits

Qubits are represented mathematically as vectors, more specifically as unit normal column vectors in a two-dimensional complex vector space. The two basis vectors of this vector space are denoted as $|0\rangle$ and $|1\rangle$, and are represented as follows:

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

The unit normal nature of these vectors means that they always have a magnitude of 1 (the square root of the sum of the squared components is 1). This magnitude can be thought of as the probability that the qubit will take any state when observed (always must be 1 since a state must be observed). While that information alone may seem useless, it entails that the square of each component of the vector is equal to the probability of the system being in that state. The first row of the vector represents the $|0\rangle$ component, and the second row represents the $|1\rangle$ component. For the basis vector $|0\rangle$, the probability of being in the $|0\rangle$ state is 1^2, and the probability of being in the $|1\rangle$ state is 0^2. Similarly, for the basis vector $|1\rangle$, the probability of being in the $|0\rangle$ state is 0^2, and the probability of being in the $|1\rangle$ state is 1^2. Consider the qubit below:

$$q = \begin{bmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} \end{bmatrix}$$

In this case, the probability of q being in the $|0\rangle$ state would be $(\frac{1}{2})^2 = \frac{1}{4}$, while the probability of q being in the $|1\rangle$ state would be $(\frac{\sqrt{3}}{2})^2 = \frac{3}{4}$. **ADD REFERENCE AND EXPLANATION FOR COMPLEX VECTOR SPACE**.

1.2 Quantum Coin Toss

1.3 Multi-Qubit Systems

A one-qubit system is about as useful as a single bit in a classical computer, so it is necessary to combine multiple qubits together to create a more complex system. This can be accomplished by taking the tensor product of two qubits/systems of qubits. The tensor product of two vectors is a vector that represents the combined system of the two vectors. For example, the tensor product of the two qubits $|0\rangle$ and $|1\rangle$ would be:

$$|0\rangle \otimes |1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ 0 \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = |01\rangle$$

Notice that the tensor product of two vectors of length n and m will result in a vector of length $n \cdot m$. The four-dimensional vector above represents the two qubit system where the first qubit is in the 0 state and the second qubit is in the 1 state (hence the $|01\rangle$ notation). **ADD REFERENCE FOR TENSOR/KROENECKER PRODUCT**. We can also take the tensor product of larger qubit systems, take the following example:

$$|01\rangle \otimes |10\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \\ 1 \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \\ 0 \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \\ 0 \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = |0110\rangle$$

1.3.1 Example of a Multi-Qubit System

1.4 The Bloch Sphere

1.4.1 Bloch Sphere Representations

1.5 Physical Qubits

A physical qubit is a physical system that can be used to represent a qubit. There are many different types of physical qubits, but the most common are the spin of an electron, the polarization of a photon, and the energy levels of an atom. The commonalities shared by these systems are that they all have two possible observed states, yet they can all be in a superposition of those states. For the spin of an electron, the $|0\rangle$ state would be spin up, and the $|1\rangle$ state would be spin down. These physical qubits based on spin can be modeled by the Bloch sphere, which is a unit sphere that represents the possible states of a qubit. The north pole of the sphere represents the $|0\rangle$ state, and the south pole represents the $|1\rangle$ state. Any point outside of the poles represents the qubit in a state of superposition, and the point on the sphere represents the **fill this in later, think it's the probabilities, but not sure**. The polarization of a photon, or the direction of its electromagnetic field, can be used with horizontally polarized representing the $|0\rangle$ state, and vertically polarized representing the $|1\rangle$ state. The energy levels of an atom are used in trapped ion quantum computers, and the $|0\rangle$ state is represented by the ground state of the atom, while the $|1\rangle$ state is represented by the excited state of the atom.

Physical qubits are used to store all of the information in a quantum computer, meaning ideally they would be resistant to errors, to prevent loss of information. However, in reality, physical qubits are extremely susceptible to errors, and thus error correction is necessary to ensure the accuracy of the information stored in qubits. The most common types of errors are bit-flip and phase-flip errors, which will be discussed in more depth in section **Add reference to section** 3.3. Since physical qubit systems are so unfathomably delicate, these errors are prone to occur quite often and are caused by a multitude of different factors. This is a major reason why error correction is so important; there is an abundance of noise and errors when working with these systems, which leads to information loss. The most common causes of errors are **find common causes. One can be qubits interacting with each other, causing interference in their wave functions**. Because one of the main contributors to noise is qubits interfering with each other, the probability of information loss increases exponentially as the number of qubits in a system increases. Since it is inherently impossible to eliminate the noise in quantum computers, it is necessary to devise a system that allows correction of these errors, which is the purpose of logical qubits.

1.5.1 Trapped Ion Qubits

1.6 Logical Qubits and Limitations of Quantum Computers

Logical qubits will be discussed in further detail in section 3.1, but for now it is important to understand that logical qubits are a method of representing ideal, error-resistant qubits. Logical qubits are built using a system of multiple physical qubits that are entangled together in such a way that some of the qubits, called auxiliary qubits, can detect and/or correct errors in the other qubits, called data qubits. As is shown in section 3.1, simpler logical qubits that only detect/correct certain types of errors can be built using smaller amount of qubits, but more practical logical qubits require at least **number** physical qubits to detect multiple different types of errors. The number of physical qubits required to build a logical qubit is called the overhead of the logical qubit, and is a major factor in the practicality of a logical qubit. **Say something about how there really aren't any "practical" logical qubits yet, since they all have such high overheads, computers cannot contain very many qubits yet, and there are issues with gate operations on logical qubits**.

2 Quantum Circuits

2.1 Basic Quantum Gates

In classical computers, logic gates are used to manipulate the states of bits. More information about classical logic gates can be found **HERE** Similarly, qubits and qubit systems are altered using quantum gates. Quantum gates are represented as matrices, and are applied to qubits by multiplying the gate matrix by the qubit vector. The result of this multiplication is the new state of the qubit.

2.1.1 X Gate

The X gate is the quantum analogue of the NOT gate in classical computers. It is represented by the following matrix:

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Similar to classical computers, if the X gate is applied to a qubit in the $|0\rangle$ state, that qubit will then be in the $|1\rangle$ state and vice versa. This can be clearly shown using the mathematical representation of the qubit and the gate:

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

Add something about how the X gate flips the qubit along the x-axis of the Bloch sphere (or whatever axis it was, I forget)

2.1.2 Y Gate

The Y gate is represented by the following matrix:

$$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

Fill in information about what the Y gate does. Applying it to the $|0\rangle$ state and the $|1\rangle$ state, we get:

$$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ i \end{bmatrix}$$
$$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -i \\ 0 \end{bmatrix}$$

2.1.3 Z Gate

The Z gate is represented by the following matrix:

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Fill in information about what the Z gate does. Applying it to the $|0\rangle$ state and the $|1\rangle$ state, we get:

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$
$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

2.1.4 Hadamard Gate

The Hadamard gate is represented by the following matrix:

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

The hadamard gate is used to put a qubit into a superposition of the $|0\rangle$ and $|1\rangle$ states. This is extremely useful because it is what allows us to capitalize on the quantum mechanical properties of qubits. Applying the hadamard gate to the $|0\rangle$ state and the $|1\rangle$ state, we get:

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$
$$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}$$

Notice that in each of those cases the magnitude of the resulting vector is still 1, yet the probabilities of the qubit being in the $|0\rangle$ and $|1\rangle$ states are now both $\frac{1}{2}$. The hadamard gate can also be applied again to remove these superpositions and return the qubits to their original states:

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \times \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$
$$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \times \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

2.1.5 CNOT/CX Gate

The CNOT gate is another one of the most important gates in quantum computing, since facilitates the entanglement of qubits, which is a key component of quantum computing. It does this by essentially projecting the state of one qubit onto another qubit (**double check this**). The CNOT gate is the first gate we have discussed that acts on multiple qubits instead of just one. Therefore, we must recall the tensor product of qubits that we discussed in section 1.1.1. We can use the tensor product to represent two qubits in a system, and then apply the CNOT gate to that system. Consider the following two-qubit systems:

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = |10\rangle$$

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = |01\rangle$$

If we apply the CNOT gate to the $|10\rangle$ system, we get:

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

We end up with the $|11\rangle$ state, meaning both qubits in the system are now in the $|1\rangle$ state. This is because, as previously described, the CNOT gate projects the state of the first qubit onto the second qubit. Since the first qubit was in the $|1\rangle$ state, the second qubit is now also in the $|1\rangle$ state. Knowing this, we can now make the prediction that applying the CNOT gate to the $|01\rangle$ system will result in the $|00\rangle$ system, since the first qubit is in the $|0\rangle$ state. However, this is not the case:

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

explain why this is, maybe give another example with a two qubit system in superposition

2.1.6 CZ Gate

The CZ gate is represented by the following matrix:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

explain what the cz gate does

2.2 Basic Rotation Circuit

2.3 Combining Quantum Gates

2.4 Building Quantum Gates

In section 2.1, we discussed some of the most common quantum gates and their functions. However, we omitted an explanation regarding the underlying principles governing the functions of these gates. This absence of clarification could potentially obscure the rationale behind their compositions. In this section, we will address this oversight, demonstrating not only how these gates function, but also why they work. The answer, unsurprisingly, involves the fundamental concept that mathematically relates qubits to gates: matrix multiplication.

2.4.1 Building Gates by Manipulating Basis States

When building a quantum logic gate, it is customary to focus on the gate's effect on the basis states of the system. Any n qubit system has 2^n basis states, each of which can be represented as a column vector with 2^n rows. Furthermore, each basis state is a vector with a single 1 and $2^n - 1$ 0's. Knowing this, consider the general case of each basis state of an n qubit system passing through a gate:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{12^n} \\ a_{21} & a_{22} & \cdots & a_{22^n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{2^n1} & a_{2^n2} & \cdots & a_{2^n2^n} \end{bmatrix} \times \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{i2^n} \end{bmatrix} = \begin{bmatrix} X_{i1} \\ X_{i2} \\ \vdots \\ X_{i2^n} \end{bmatrix}$$

Here, i represents the i^{th} basis state in the system (therefore $i \in \{1, 2, ..., 2^n\}$). For example, $x_{11}, x_{12}, ..., x_{12^n}$ are the elements of the first basis state in the system and $X_{11}, X_{12}, ..., X_{12^n}$ are the elements of the corresponding output state for the first basis state.

The system of equations for such an event can be generalized as follows:

$$\begin{aligned} a_{11}x_{i1} + a_{12}x_{i2} + ... + a_{12^n}x_{i2^n} &= X_{i1} \\ a_{21}x_{i1} + a_{22}x_{i2} + ... + a_{22^n}x_{i2^n} &= X_{i2} \\ &\vdots \\ a_{2^n1}x_{i1} + a_{2^n2}x_{i2} + ... + a_{2^n2^n}x_{i2^n} &= X_{i2^n} \end{aligned}$$

Now, consider some $j \in \{1, 2, ..., 2^n\}$. It is always true that for the i^{th} basis state in any n qubit system, $x_{ii} = 1$ and $x_{ij} = 0$ for all $j \neq i$. For example, the first row of the first basis state of a system (where all qubits are $|0\rangle$) is always 1, so in that case $x_{11} = 1$ and all x_{1j} where $j \neq 1$ are 0.

Thus, if the gate were to be applied to the first basis state of the system, where $x_{11} = 1$ and $x_{1j} = 0$ when $j \neq 1$, the above system of equations can be simplified to:

$$\begin{aligned} a_{11} &= X_{11} \\ a_{21} &= X_{12} \\ &\vdots \\ a_{2^n1} &= X_{12^n} \end{aligned}$$

Take a moment to understand why this happens by plugging the proper values into the system of equations above.

Expanding this to the general case, if the gate were to be applied to the i^{th} basis state of the system, where $x_{ii} = 1$ and $x_{ij} = 0$ when $j \neq i$, then the system of equations would be reduced to:

$$\begin{aligned} a_{1i} &= X_{i1} \\ a_{2i} &= X_{i2} \\ &\vdots \\ a_{2^ni} &= X_{i2^n} \end{aligned}$$

where $X_{i1}, X_{i2}, ..., X_{i2^n}$ represent the desired output state for the basis state with $x_{ii} = 1$ (the i^{th} basis state in the system).

Notice that $a_{1i}, a_{2i}, ..., a_{2^ni}$ are now equivalent to the desired output state for the i^{th} basis state of the system. Also notice that these are the values that compose the i^{th} column of the gate matrix. This is a crucial observation, because it provides a heuristic for building quantum gates.

The heuristic is as follows: the i^{th} column of the gate matrix is the desired output vector for the i^{th} basis state of the system.

Hence, the completed gate matrix for the general example would be:

$$\begin{bmatrix} X_{11} & X_{21} & \cdots & X_{2^n1} \\ X_{12} & X_{22} & \cdots & X_{2^n2} \\ \vdots & \vdots & \ddots & \vdots \\ X_{12^n} & X_{22^n} & \cdots & X_{2^n2^n} \end{bmatrix}$$

Given the many systems of equations and substitutions involved in the general derivation, it may seem convoluted or difficult to follow. For clarity, let's try a simple example and build a quantum gate for a two qubit system. These are the basis states for such a system:

$$|00\rangle = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} \; |01\rangle = \begin{bmatrix}0\\1\\0\\0\end{bmatrix} \; |10\rangle = \begin{bmatrix}0\\0\\1\\0\end{bmatrix} \; |11\rangle = \begin{bmatrix}0\\0\\0\\1\end{bmatrix}$$

Suppose there exists a desired output state for each of the four basis states as follows:

$$\begin{aligned}|00\rangle &\rightarrow |A\rangle\\ |01\rangle &\rightarrow |B\rangle\\ |10\rangle &\rightarrow |C\rangle\\ |11\rangle &\rightarrow |D\rangle\end{aligned}$$

We can then create an arbitrary gate and some equations to represent these transformations:

$$\begin{bmatrix}a_1 & b_1 & c_1 & d_1\\ a_2 & b_2 & c_2 & d_2\\ a_3 & b_3 & c_3 & d_3\\ a_4 & b_4 & c_4 & d_4\end{bmatrix} \times \begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \begin{bmatrix}A_1\\A_2\\A_3\\A_4\end{bmatrix}$$

$$\begin{bmatrix}a_1 & b_1 & c_1 & d_1\\ a_2 & b_2 & c_2 & d_2\\ a_3 & b_3 & c_3 & d_3\\ a_4 & b_4 & c_4 & d_4\end{bmatrix} \times \begin{bmatrix}0\\1\\0\\0\end{bmatrix} = \begin{bmatrix}B_1\\B_2\\B_3\\B_4\end{bmatrix}$$

$$\begin{bmatrix}a_1 & b_1 & c_1 & d_1\\ a_2 & b_2 & c_2 & d_2\\ a_3 & b_3 & c_3 & d_3\\ a_4 & b_4 & c_4 & d_4\end{bmatrix} \times \begin{bmatrix}0\\0\\1\\0\end{bmatrix} = \begin{bmatrix}C_1\\C_2\\C_3\\C_4\end{bmatrix}$$

$$\begin{bmatrix}a_1 & b_1 & c_1 & d_1\\ a_2 & b_2 & c_2 & d_2\\ a_3 & b_3 & c_3 & d_3\\ a_4 & b_4 & c_4 & d_4\end{bmatrix} \times \begin{bmatrix}0\\0\\0\\1\end{bmatrix} = \begin{bmatrix}D_1\\D_2\\D_3\\D_4\end{bmatrix}$$

The next step is to create systems of equations for each of these events. First, consider the system of equations for the gate being applied to $|00\rangle$ ($i = 1$ in the general example):

$$\begin{aligned}a_1(1) + b_1(0) + c_1(0) + d_1(0) = a_1 = A_1\\ a_2(1) + b_2(0) + c_2(0) + d_2(0) = a_2 = A_2\\ a_3(1) + b_3(0) + c_3(0) + d_3(0) = a_3 = A_3\\ a_4(1) + b_4(0) + c_4(0) + d_4(0) = a_4 = A_4\end{aligned}$$

Next, for the gate applied to $|01\rangle$ ($i = 2$):

$$\begin{aligned}a_1(0) + b_1(1) + c_1(0) + d_1(0) = b_1 = B_1\\ a_2(0) + b_2(1) + c_2(0) + d_2(0) = b_2 = B_2\\ a_3(0) + b_3(1) + c_3(0) + d_3(0) = b_3 = B_3\\ a_4(0) + b_4(1) + c_4(0) + d_4(0) = b_4 = B_4\end{aligned}$$

Then, for the gate applied to $|10\rangle$ ($i = 3$):

$$\begin{aligned}a_1(0) + b_1(0) + c_1(1) + d_1(0) = c_1 = C_1\\ a_2(0) + b_2(0) + c_2(1) + d_2(0) = c_2 = C_2\\ a_3(0) + b_3(0) + c_3(1) + d_3(0) = c_3 = C_3\\ a_4(0) + b_4(0) + c_4(1) + d_4(0) = c_4 = C_4\end{aligned}$$

Finally, for the gate applied to $|11\rangle$ ($i = 4$):

$$\begin{aligned}a_1(0) + b_1(0) + c_1(0) + d_1(1) = d_1 = D_1\\ a_2(0) + b_2(0) + c_2(0) + d_2(1) = d_2 = D_2\\ a_3(0) + b_3(0) + c_3(0) + d_3(1) = d_3 = D_3\\ a_4(0) + b_4(0) + c_4(0) + d_4(1) = d_4 = D_4\end{aligned}$$

Given the above systems of equations, the pattern relating columns of the gate matrix to the desired output vectors for the basis states becomes clear. In this case, the constructed gate matrix for these transformations is:

$$\begin{bmatrix}A_1 & B_1 & C_1 & D_1\\ A_2 & B_2 & C_2 & D_2\\ A_3 & B_3 & C_3 & D_3\\ A_4 & B_4 & C_4 & D_4\end{bmatrix}$$

Now that we know how to build a gate for an n qubit system, let's apply this knowledge to build some of the gates we have already learned.

2.4.2 Building the X (NOT) Gate

As previously discussed, gates are built by considering the desired output states of all basis states of the system when passed through the gate. The X gate is the quantum analogue of a NOT gate, so it should negate the basis states of the system. Furthermore, a NOT gate can only negate a single bit, so the X gate should only negate a single qubit. Thus, the X gate can only operate on a 1 qubit system and should have the following desired output states for each basis state:

$$|0\rangle \rightarrow |1\rangle : \begin{bmatrix}1\\0\end{bmatrix} \rightarrow \begin{bmatrix}0\\1\end{bmatrix}$$

$$|1\rangle \rightarrow |0\rangle : \begin{bmatrix}0\\1\end{bmatrix} \rightarrow \begin{bmatrix}1\\0\end{bmatrix}$$

Recall that according to the heuristic developed in the last section, the first column of the gate should be the desired output for the first basis state ($|0\rangle$), and the second column should be the desired output for the second basis state ($|1\rangle$). Thus, the gate matrix for the X gate should be:

$$\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$$

This is the correct X gate matrix, as shown in section 2.1.1. However, for the sake of completeness, let's verify again why the heuristic works as expected.

An X gate is a 1 qubit gate, so the equation for applying the gate would be:

$$\begin{bmatrix}a_1 & b_1\\ a_2 & b_2\end{bmatrix} \times \begin{bmatrix}x_1\\x_2\end{bmatrix} = \begin{bmatrix}X_1\\X_2\end{bmatrix}$$

The system of equations for this would be:

$$\begin{aligned}a_1x_1 + b_1x_2 = X_1\\ a_2x_1 + b_2x_2 = X_2\end{aligned}$$

Now, consider the situation where the input state is $|0\rangle$ and the output should be $|1\rangle$. Our equations become:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$a_1(1) + b_1(0) = a_1 = X_1 = 0$$
$$a_2(1) + b_2(0) = a_2 = X_2 = 1$$

Clearly, the first column of the gate is the desired output for the $|0\rangle$ state, which is $|1\rangle$. Now for the situation where the input state is $|1\rangle$ and the output should be $|0\rangle$:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$a_1(0) + b_1(1) = b_1 = X_1 = 1$$
$$a_2(0) + b_2(1) = b_2 = X_2 = 0$$

Again, the heuristic holds true, and the second column of the gate is the desired output for the $|1\rangle$ state, which is $|0\rangle$. We now have that $a_1 = 0$, $a_2 = 1$, $b_1 = 1$, and $b_2 = 0$. This gives us the matrix:

$$\begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Thus, we have constructed the X gate (2.1.1).

2.4.3 Building Other Common Quantum Gates

Now that we understand how quantum gates are constructed, we can understand why the gates we have already learned work as expected. Consider the CNOT gate, or controlled NOT gate. As previously discussed, the CNOT gate projects the state of the first qubit onto the second qubit. In other words, the second qubit should be flipped if and only if the first qubit is in the $|1\rangle$ state. It is a gate that operates on two qubits, so the gate matrix should be a 4×4 matrix. Given the logic for how the gate should operate, the basis states should transform as follows:

$$|00\rangle \to |00\rangle : \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \to \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$|01\rangle \to |01\rangle : \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \to \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$|10\rangle \to |11\rangle : \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \to \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$|11\rangle \to |10\rangle : \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \to \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Therefore, based on the heuristic, the gate matrix should be:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

2.5 Grover's Algorithm

2.5.1 Why Grover's Algorithm Works

2.6 Circuit Calculator

3 Logical Qubits and Error Correction

3.1 Logical Qubits

3.2 Where Logical Qubits are Used

3.3 Simple Error Correction Circuits

3.3.1 Bit-Flip Code

3.3.2 Phase-Flip Code

3.4 Shor's Code

3.5 Logical Gates

3.6 Bacon-Shor Code

3.6.1 Ancilla Qubits

3.6.2 Why Ancilla Qubits Do Not Disrupt the System