4-15-2019

Quantum Advantages Quantum Algorithm for Finding the Minimum

Binam Bajracharya
Lake Forest College, bajracharyab@lakeforest.edu

Quantum Advantages Quantum Algorithm for Finding the Minimum

Abstract
Theories about quantum computers and how they would work have been around for a few decades. Peter Shor and Lov Grover even came up with algorithms during the 1990s when the idea of building a quantum computer was far-fetched. We are now at a point where our processors are getting faster and smaller, only a few nanometres thick. This obviously has its limits. Big companies like Google, Microsoft, and IBM are at a point where their quantum computers are equivalent to the room sized computers that we see in some old pictures. Here we build up the necessary background required to understand how quantum computers work and what advantages a quantum computer has over a classical computer. We will also be looking at the famous Grover's quantum search algorithm and will be using the ideas from that to create our own simple algorithm that finds the minimum value from a given set of data.

Document Type
Thesis

Distinguished Thesis
Yes

Degree Name
Bachelor of Arts (BA)

Department or Program
Mathematics and Computer Science

First Advisor
Robert L. Holliday

Second Advisor
Nathan Mueggenburg

Third Advisor
Craig D. Knuckles

Subject Categories
Atomic, Molecular and Optical Physics | Mathematics

Printed Name: Binam Bajracharya

Thesis Title: Quantum Advantages Quantum Algorithm for Finding the Minimum

LAKE FOREST COLLEGE

Senior Thesis

Quantum Advantages

Quantum Algorithm for Finding the Minimum

by

Binam Bajracharya

April 15, 2019

The report of the investigation undertaken as a
Senior Thesis, to carry two courses of credit in
the Department of Mathematics and Computer Science

Abstract

Theories about quantum computers and how they would work have been around for a few decades. Peter Shor and Lov Grover even came up with algorithms during the 1990s when the idea of building a quantum computer was far-fetched. We are now at a point where our processors are getting faster and smaller, only a few nanometres thick. This obviously has its limits. Big companies like Google, Microsoft, and IBM are at a point where their quantum computers are equivalent to the room sized computers that we see in some old pictures. Here we build up the necessary background required to understand how quantum computers work and what advantages a quantum computer has over a classical computer. We will also be looking at the famous Grover's quantum search algorithm and will be using the ideas from that to create our own simple algorithm that finds the minimum value from a given set of data.

DEDICATION

I want to dedicate this work to my parents, Birendra and Juni Bajracharya, who have supported me through everything and made me the person I am today.

I want to also dedicate this work to my friends. To my best friends, Lumi and Pujyata, who have been there throughout all the difficult times and always helping me get back up when I am down, even if they're in the other side of the world. To Hannah for always making me believe that I can achieve higher and without whom I would not have written this thesis. And to Sean, Ayesha, and all the people in Cleveland Young that I've had the pleasure to live with and keeping me going for the past 4 years.

Acknowledgements

I would like to thank all my committee members, Professor Robert L. Holliday, Professor Craig D. Knuckles, and Professor Nathan Mueggenburg for agreeing to be on my committee and taking the time and effort to guide me and going through all the different iterations of my thesis. I would like to thank all the professors in the Department of Mathematics & Computer Science and the Department of Physics for sparking the joy of learning.

Contents

1 Introduction……………………………………………………………..…………1

2 Preliminaries…………………………………………………………………....3

- 2.1 What is a Quantum Computer..……………………………………………..3
- 2.2 Qubits……………..……………………………………………………...4
  - 2.2.1 Bloch Sphere…………………..…………………………….9
- 2.3 Quantum Circuits……………………………………………………..10
  - 2.3.1 Single Qubit Gates…………………………………………….11
  - 2.3.2 Multiple Qubit Gates…………………………………………..15
- 2.4 Measurement……………………………………………………………….17
- 2.5 Entanglement……………………………………………………………….19
- 2.6 Decoherence………………………………………………………………..22
- 2.7 Computational Complexity……………………………………………...23

3 Quantum Algorithms……………………………………………………..…26

- 3.1 Grover's Algorithm……………………………………………………….27
  - 3.1.1 The Algorithm……………………………………………………28
    - 3.1.1.1 Grover Iteration……………………………………...28
- 3.2 Finding Minimum Using Grover Algorithm…………………………….30
  - 3.2.1 The Algorithm……………………………………………………31
- 3.3 Alternate Minimum Algorithm………………………………………….32
  - 3.3.1 The Algorithm……………………………………………………32
  - 3.3.2 Worked Example…………………………………………………34
- 3.4 Proof of Alternate Minimum Algorithm...……………………………….40
- 3.5 Observation and Final Remarks………………………………………….41

List of Figures

FIG. 1: Geometric representation of the angle between two vectors defined using an inner product…………………..…………………………………………………..………8

FIG. 2: Bloch Sphere Representation of a Single Qubit………………………………9

FIG. 3: Quantum circuit of the *X*-gate acting on a single qubit……………………...12

FIG. 4: Quantum circuit of the Hadamard-gate acting on a single qubit....................14

FIG. 5: Quantum circuit of the CNOT acting on two qubits…………………………16

FIG. 6: Equivalent quantum circuits where a measurement is taken, and *U* is operated……………………………………………………………………………18

FIG. 7: Probability distribution achieved by squaring the magnitudes of each state…………………………………………………………………………………...39

1. Introduction

After decades of amazing progress in computer technology we are approaching a wall in computing power. Ever since the introduction of microprocessors in 1971 [1], we have been able to exponentially decrease the size of transistors used in microprocessors and increase processing power. A transistor, to put it simply, is a “switch” that controls the flow of electrons. It can either be in an on or an off state depending on whether there is current flowing through it, and this forms the basis of all computer devices today. The smallest transistor we can create today is 14 nm ($1nm = 10^{-9}m$). A red blood cell is 8000 nm for scale. [2] When we approach the size of an atom things tend not to work the way we expect them to. We leave behind the realm of classical physics, where things are typically intuitive, and enter the world of quantum mechanics. One difficulty that arises is quantum tunneling. [1] This is the quantum mechanical phenomenon where a subatomic particle, in our case, an electron, passes through a potential barrier that it typically would not be able to cross under the conditions of classical mechanics. When the barrier gets small enough the particle no longer obeys the laws of classical mechanics, where things work as expected and enters the quantum world. [2] So, when a transistor is made too small, the electron, with a very small probability, passes through the barrier and onto the other side when the “switch” is turned off. Even though this occurs with a small probability, we cannot ignore this possibility when we’re doing big, complicated calculations.

Quantum tunneling is not something that can be easily replicated using classical physics since what we’re describing is essentially passing a particle through a barrier without changing the state of the particle or the barrier. Instead of finding a workaround for this problem, we can use these unique quanta mechanical phenomenon to our advantage. We can figure out new and different ways to solve problems that we were not capable of doing in a classical computer and make existing solutions better.

2. Preliminaries

2.1 What is a Quantum Computer?

A **quantum computer** is a device that manipulates quantum states in a controlled manner, the way a common digital computer manipulates the state of its bits [3]. The computing is done using quantum mechanical phenomena, such as superposition and entanglement, on quantum-bits or **qubits**. Just as memory is made up of bits in a classical computer, where each bit is represented by either 1 or 0, memory in quantum computers is made up of qubits, where each qubit is represented by either a $|1\rangle$, $|0\rangle$, or both at once, due to a phenomenon called **superposition** [3]. Superposition in classical physics refers to the interaction of two or more waves and how they add to create constructive interference and subtract to create destructive interference. In the quantum world, particles can also have wave-like behaviour. Particles can exist in different states, they can be spinning up or spinning down and can have different energies or be moving with different speeds just like the $|1\rangle$ and $|0\rangle$ states (represented by Dirac's bra-ket notation). Qubits can be thought of as a wave where all these states fluctuate and exist simultaneously. They are represented by $c_o|0\rangle + c_1|1\rangle$ where c_o and c_1 represent complex numbers and represent the amplitude of $|0\rangle$ and $|1\rangle$ respectively.

Superposition is impressive, and no classical computer can achieve it. But in quantum computing, superposition is most useful when it is in an entangled state.

Quantum entanglement is a quantum mechanical phenomenon in which the quantum states of two or more particles are described with reference to each other, even though the particles might be spatially apart [4]. The correlation between the outcomes of the two qubits is detected only after the qubits have been measured and compared. Entanglement is a property that plays a vital role in essential effects such as quantum teleportation, fast quantum algorithms, and error-correction [3]. There is no comparable classical equivalent for this.

The ability of qubits to exist in entangled states is responsible for the increase in computing power of quantum computers. We can see an increase in space and power when we compare it to classical bits. A classical computer with N-bits can have any one of the possible 2^N values whereas a quantum computer of N-qubits can be all the possible 2^N values at the same time. This is due to the quantum communication protocol called superdense coding which combines all the basic ideas of quantum mechanics [3].

2.2 Qubits

As mentioned earlier, a qubit consists of these two states $|0\rangle$ and $|1\rangle$ read as ket 0 and ket 1 respectively. These form the computational basis corresponding to the following vectors:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{1}$$

Through superposition of states we can also have states besides $|0\rangle$ and $|1\rangle$, such as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$. We call this the **normalization condition.** [4] When we measure or observe a qubit with state $|\psi\rangle$ we either get the state $|0\rangle$ with a probability of $|\alpha|^2$ or $|1\rangle$ with a probability of $|\beta|^2$. Quantum computation is performed by increasing the probability of observing a correct state to a high value so that the correct answer may be found with a high amount of certainty. Thus, unlike classical computers, quantum computers are not completely deterministic. For a system to be *deterministic* it needs to give the same set of outputs every time the same set of input is given to a program. Quantum computers are non-deterministic because even if an algorithm has a high chance of getting the same value each time, there always will be a small chance that it is not the same value. [4]

As with all vectors, $|0\rangle$ and $|1\rangle$, have a direction and a magnitude. Using these as our basis vectors, we can construct all other vectors from a linear combination of these. We have been looking at just 1 qubit until now. To create larger numbers, we need the **tensor product,** which is one of the operations from linear algebra. Given two vector spaces V and W, the tensor product is represented by $V \otimes W$ where $V \otimes W$ is spanned by the elements of the form $v \otimes w$ that satisfy the following conditions for every scalar α. [5]

$$\begin{aligned} (v_1 + v_2) \otimes w &= v_1 \otimes w + v_2 \otimes w \\ v \otimes (w_1 + w_2) &= v \otimes w_1 + v \otimes w_2 \\ \alpha(v \otimes w) &= (\alpha v) \otimes w = v \otimes (\alpha w) \end{aligned} \tag{2}$$

$$0 \otimes w = v \otimes 0 = 0$$

This helps us combine vector spaces to form larger vector spaces.

$$\begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \otimes \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \\ x_1 \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} x_0 & y_0 \\ x_0 & y_1 \\ x_1 & y_0 \\ x_1 & y_1 \end{pmatrix} \tag{3}$$

We observe that the dimension of the tensor product vector is equal to the product of the dimensions of the vectors:

$$\dim(V \otimes W) = \dim(V) \times \dim(W) \tag{4}$$

We can see how this can be useful through an example:

$$|10\rangle = |1\rangle \otimes |0\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \tag{5}$$

Quantum registers can store individual numbers like so:

$$|4\rangle = |100\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{6}$$

Quantum registers can also store multiple numbers simultaneously say 4 and 7:

$$\frac{1}{\sqrt{2}}(|4\rangle + |7\rangle) = \frac{1}{\sqrt{2}}(|100\rangle + |111\rangle) \tag{7}$$

where the register has a 50% chance of being either 4 or 7. This is an example of superposition of states. In general, a superposition can be mathematically represented by:

$$\sum_{i=0}^{2^{n-1}} a_i \, |S_i\rangle \tag{8}$$

Here n is the number of qubits, $|S_i\rangle$ is a state vector, and a_i is the amplitude of the state vector.

A quantum state provides a probability distribution for the possible outcomes of the system. There are two types of states, that are the building blocks of quantum states. One state is the *pure quantum state.* These cannot be formed using any other states. The other is the *mixed quantum state.* We can define mixed states as a distribution over quantum states, so $\{p_{i,}\psi_i\} = p_{1,}|\psi_1\rangle, \dots, p_{n,}|\psi_n\rangle$ where p_i is the probability that the superposition is $|\psi_i\rangle$. A mixed state cannot be defined with just a single ket and needs to be defined using a density matrix, ρ. Here, $\rho = \sum_s p_s |\psi_s\rangle\langle\psi_s|$ where p_s is the probability of the vector $|\psi_s\rangle\langle\psi_s|$ [3] A *Hermitian conjugate* is a transposed complex conjugate of a vector. [6] Before we proceed further, let us define some terms:

Definition 1. A ***metric space*** is a set S with a global distance function g that, for every two points x, y in S, gives the distance between them as nonnegative real number $g(x, y)$. A metric space must also satisfy:

1. $g(x,y) = 0$ iff $x = y$
2. $g(x,y) = g(y,x)$
3. The triangle inequality $g(x,y) + g(y,z) \geq g(x,z)$ [7]

Definition 2. ***Cauchy sequence*** is a sequence $a_1, a_2, \ldots$ such that the metric $g(a_m, a_n)$ satisfies:

$$\lim_{\min(m,n)\to\infty} g(a_m, a_n) = 0 \tag{9}$$

Definition 3. A ***complete metric space*** is a metric space in which every Cauchy sequence is convergent. [8]

Definition 4. An *inner product* is a generalization of the dot product. In a vector space, it is a way to multiply vectors together, with the result of this multiplication being a scalar. [9]

There is a more precise definition for inner product but for this paper we only need to understand the geometric interpretation of this definition.

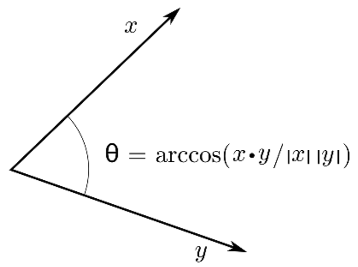


FIG. 1: Geometric representation of the angle between two vectors defined using an inner product [9]

Definition 5. A ***Hilbert space*** H is a real or complex inner product space that is also a complete metric space with respect to the distance function induced by the inner product. [10] To say that H is a complex inner product space means that H is a complex vector space on which there is an inner product $\langle x, y\rangle$ associating a complex number to each pair of elements x, y of H.

2.2.1 Bloch Sphere

Using these definitions, we have created a space where we can geometrically visualize a qubit. It is a sphere with a radius of 1 and a point on its surface representing the state of a qubit. Even though we can only use this model to visualize a single qubit, it provides an intuitive way to interpret its state, and gives some idea about quantum information and qubit manipulation.

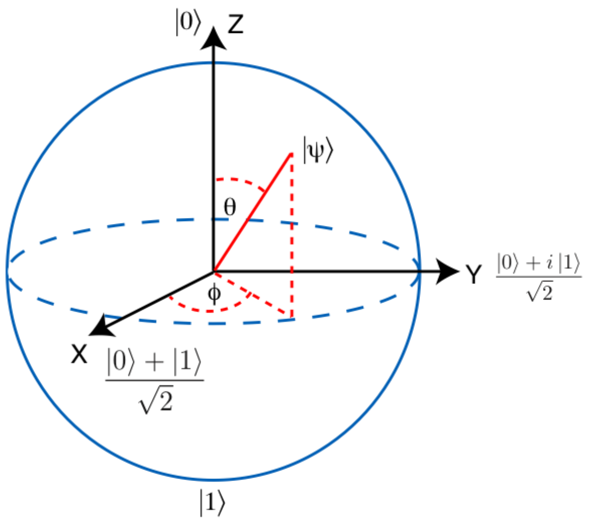

$|0\rangle$
Z
$|\psi\rangle$
θ
Y
$\frac{|0\rangle + i|1\rangle}{\sqrt{2}}$
ϕ
X
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
$|1\rangle$

FIG. 2: Bloch Sphere Representation of a Single Qubit [4]

The qubit $|\psi\rangle = a|0\rangle + b|1\rangle$ can be represented as a point (θ, ϕ) on the unit sphere. The angles θ and ϕ define a point in $\mathbb{R}^3$ as shown in FIG. 2. We can define the angles by letting $a = \cos(\theta/2)$ and $b = e^{i\phi}\sin(\theta/2)$. We now have,

$$|\psi\rangle = \cos(\theta/2)|0\rangle + e^{i\phi}\sin(\theta/2)|1\rangle \tag{10}$$

Looking at equation (10) we notice that there is a one-to-one relation between qubit states represented by the angles in $\mathbb{R}^3$ and the points on the surface of the unit sphere represented by $|\psi\rangle$ in $\mathbb{C}^2$. The state on the top of the sphere represents $|0\rangle$ and the bottom represents $|1\rangle$ which corresponds to the spin-up and the spin-down states of an electron. The qubit is in superposition when the vector points somewhere between $|0\rangle$ and $|1\rangle$. All the points on the surface of the sphere correspond to superposition of pure states of the system, and the points inside the Bloch sphere represents mixed states. [3]

We have no way of representing multiple qubits using a Bloch sphere. Since we know that one qubit exists on a complex vector space of dimension 2, $\mathbb{C}^2$. We also know that the standard basis is the set of all binary strings in the set $\{0, \ldots, 2^n - 1\}$. So, the complex vector space of an n qubit system has dimension of 2^n, denoted by $\mathbb{C}^{2^n}$. [3]

2.3 Quantum Circuits

A quantum circuit is a model in quantum information theory for quantum computation in which computation in carried out as a sequence of quantum gates that are reversible transformations on an n-qubit register. [11] These transformations should be able to change the state of a given qubit just as we manipulate classical bits with logic gates like AND, OR, or NOT.

2.3.1 Single Qubit Gates

Consider an operation on a single bit, the classical NOT gate which is defined by the truth table in which $0 \rightarrow 1$ and $1 \rightarrow 0$. The goal now is to do the same thing with a single qubit where we change the state of $|0\rangle$ to $|1\rangle$ and vice versa. This would be a useful tool to have in a quantum computer. However, just switching $|0\rangle$ to $|1\rangle$ is not enough for a qubit since qubits contain more information than a classical bit. Qubits have amplitude because of their ability to exist in superposition. The quantum NOT gate acts linearly on a qubit which means that it takes the state

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

and interchanges the role of $|0\rangle$ and $|1\rangle$,

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

We can represent this operation in form of a matrix which follows the linearity of quantum gate,

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

So, the quantum state $\alpha|0\rangle + \beta|1\rangle$ is written in vector form

$$\begin{bmatrix}\alpha\\ \beta\end{bmatrix},$$

where the top represents the amplitude for $|0\rangle$ and the bottom represents the amplitude for $|1\rangle$. [4] Applying the not gate is just doing a simple matrix multiplication of these vectors which gives us the following output,

$$X\begin{bmatrix}\alpha\\ \beta\end{bmatrix}=\begin{bmatrix}\beta\\ \alpha\end{bmatrix}$$

with the top entry still corresponding to the amplitude of $|0\rangle$ and the bottom still representing the amplitude of $|1\rangle$. Notice that now the amplitudes have switched, which means the states have swapped which is exactly the purpose of the quantum not gate. Geometrically the quantum NOT gate, or the X gate, is a π-rotation around the X axis and has the property that $X \to X$, and $Z \to -Z$. This is also known as the Pauli-X gate, which looks like the following in a quantum circuit diagram:

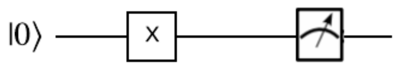


FIG. 3: Quantum circuit of the X-gate acting on a single qubit and being measured

Like the Pauli-X gates, the Pauli-Y gate acts on a single qubit as well. Considering the Bloch sphere, it equates to a rotation of π radians on the Y-axis. It maps $|0\rangle$ to $i|1\rangle$ and $|1\rangle$ to $-i|0\rangle$. [4]It is represented by the matrix:

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

A Pauli-Z gate also acts on a single qubit and is a π radians rotation around the Z-axis of a Bloch sphere. This is also known as the phase shift gate since it flips the amplitude of a state. [4] If the state is $|0\rangle$, it leaves it unchanged but if the state is $|1\rangle$ then it changes it to $-|1\rangle$. It is represented by the matrix:

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

Mathematically, classical logic gates are described using Boolean algebra [12]. There is an important restriction to what a quantum logic gate can be. Remember the normalization condition for a quantum state $\alpha|0\rangle + \beta|1\rangle$ requires $|\alpha|^2 + |\beta|^2 = 1$. This means that the logic gate matrix G must be a unitary matrix. That is $GG^H = I$, where G^H is the adjoint matrix, the conjugate transpose, $G^H \equiv \overline{G^T}$, and I is the identity matrix. This shows another unique property of quantum logic gate that most classical logic gates do not have. Quantum logic gates are reversible. This is clear from the above fact that matrices correspond to logic gates are unitary. [4]

Suppose we have a quantum state Q, represented in matrix form and a logic gate G. Applying the gate to the state would result in $QG = Q'$. As described above, the quantum gate G is unitary and has an adjoint matrix G^H associated with it. If we apply G^H to the result $Q'G^H = Q$, we get back the initial state vector Q. [3] This is another unique property of quantum computing that classical computing does not have. Quantum computation is reversible.

One of the key concepts of quantum mechanics we use is superposition. We can create a superposition using a quantum logic gate. The Hadamard gate acts on a single qubit. It maps the basis state $|0\rangle$ to $\frac{|0\rangle+|1\rangle}{\sqrt{2}} = |+\rangle$ and $|1\rangle$ to $\frac{|0\rangle-|1\rangle}{\sqrt{2}}=|-\rangle$. [3] This causes the measurement of this state to have an equal probability of 0 or 1. This represents a π rotation on the diagonal $X + Z$ axis of the Bloch sphere. It is represented by the matrix:

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

Note that the -1 in the bottom right of the matrix makes the operation reversible. Since a Hadamard matrix is a unitary matrix, multiplying H with $|+\rangle$ results in the state $|0\rangle$ and multiplying H with $|-\rangle$ results in the state $|1\rangle$. This makes this process reversible even though when observing the state of both $|+\rangle$ and $|-\rangle$, this is an equal chance of getting either 0 or 1. [4]

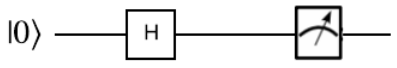


FIG. 4: Quantum circuit of the Hadamard-gate acting on a single qubit and being measured

2.3.2 Multiple Qubit Gates

Controlled gates operate on 2 or more qubits. The controlled gate operation on a qubit is determined by the state of another qubit. One of these controlled gates is the controlled NOT gate (CNOT or cX) which is based on the classical XOR gate. This gate acts on two qubits and flips the bit of the second bit only if the first bit is $|1\rangle$. [4] CNOT is represented by the following matrix:

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

Consider applying the CNOT gate to the multiple qubit state $|10\rangle$,

$$cX|10\rangle = cX\begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |11\rangle$$

and consider applying the same gate to the qubit $|01\rangle$,

$$cX|01\rangle = cX\begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |01\rangle$$

Note that the CNOT matrix is just the identity matrix with the last two rows switched. This fact, along with how tensor products work allows the bit flip to take place only when the first bit is 1.

The CNOT gate is represented in a quantum circuit as connecting two qubits together with a line.

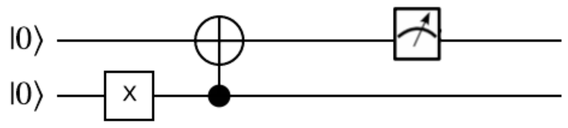


FIG. 5: Quantum circuit of the CNOT acting on the first qubit where the second qubit is the control bit

Here the first qubit has a state $|0\rangle$ and the second qubit state has been flipped to $|1\rangle$. The second qubit here is set up to be the control gate that determines whether to flip the first qubit. In this case, since the control bit is set to be $|1\rangle$ the first qubit is flipped, and we measure a 1 when this qubit is measured.

This idea of controlled gates can be further generalized. Let U be a gate that operates on a single qubit where, $U = \begin{pmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{pmatrix}$. The matrix representing the controlled U is

$$C(U) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{pmatrix},$$

where the gate U operates on the last bit if and only if the first bit is 1. So, we could apply any of the single qubit gates defined in the previous section. [3]

2.4 Measurement

Measurement is not a quantum gate since it is irreversible. It takes a quantum state and collapses it to one of the base vectors with the probability equal to the square of the base vector's amplitude. It is an irreversible operation because it sets the quantum state equal to the base vector that it represents which is a definite singular value.

For quantum computation of a multi-qubit system, the measurement can be treated as a series of single-qubit measurements. For example, a two-qubit state can be expressed as

$$\mathrm{a}|00\rangle + b|01\rangle + c|10\rangle + d\,|11\rangle \tag{11}$$

where $a, b, c, d \in \mathbb{C}$ and $|a|^2 + |b|^2 + |c|^2 + |d|^2 = 1$. Measuring this two-qubit state we break down the measurement into the first qubit and the second qubit separately in terms of the standard basis vector. To see how this works let us rewrite the state of this vector:

$$\begin{aligned} &\mathrm{a}|00\rangle + b|01\rangle + c|10\rangle + d\,|11\rangle \\ &= |0\rangle \otimes (a|0\rangle + b|1\rangle) + |1\rangle \otimes (c|0\rangle + d|1\rangle) \\ &= u|0\rangle \otimes (a/u|0\rangle + b/u|1\rangle) + v|1\rangle \otimes (c/v|0\rangle + d/v|1\rangle) \end{aligned} \tag{12}$$

where $u = \sqrt{|a|^2 + |b|^2}$ and $v = \sqrt{|c|^2 + |d|^2}$ where $|u|^2 + |v|^2 = 1$. This rewritten state decomposes the vector into its two bits so that the measurement can be done for the two qubits separately. Measurement of the first qubit returning $|0\rangle$ is $u^2 = |a|^2 + |b|^2$ which would collapse the state to $|0\rangle \otimes (a/u|0\rangle + b/u|1\rangle)$ where we would measure the second qubit. Here, the probability that the qubit is $|0\rangle$ is a/u and the probability that the qubit is $|1\rangle$ is b/u. Measuring the second qubit works the same, and the concept works for any n number of qubits. [11]

There are two important principles of measurement which apply to a quantum circuit:

1. *Principle of deferred measurement:* Measurement can always be moved from an intermediate stage of a quantum circuit to the end of a circuit. [3]This means that delaying measurements until the end of a quantum computation doesn't affect the probability distribution outcome. The circuits below are equivalent. On the left-hand side, the operation takes place first and then it is measured while on the right-hand side, the measurement happens first followed by a classical operation on the measured bit. In both cases we're measuring the value of the control bit and in both cases the control bit is unchanged.

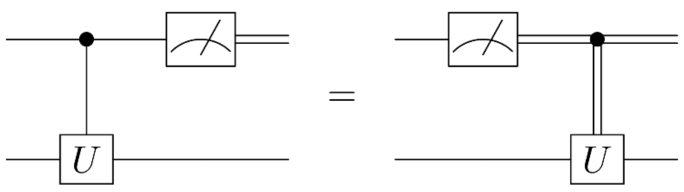


FIG. 6: Equivalent quantum circuits where a measurement is taken, and U is operated

The differed measurement allows shifting the measurement around so that it happens at better times. For example, measuring qubits as early as possible can reduce the maximum number of simultaneously stored qubits, giving us better space complexity and allows us to run an algorithm on a smaller quantum computer.

2. *Principle of implicit measurement:* Without loss of generality, any unmeasured qubits at the end of a quantum circuit may be assumed to be measured. [3] To understand this, consider two qubits and where the first one is being measured at the end of the circuit. The statistics of the observed value for the first qubit is determined by its reduced density matrix. If a measurement was also performed on the second qubit, measuring the second qubit would not influence the statistics of the measurement of the first qubit.

2.5 Entanglement

Measurement also gives another way of thinking about entangled particles. Particles are entangled if the measurement of one influences the measurement of the other. Mathematically, if the product state of two qubits cannot be factored, they are said to be entangled. Given a product state,

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix}$$

the qubit state of this should be in the form $\binom{a}{b} \otimes \binom{c}{d}$ but is not factorable since it is in an entangled state. We can show why this does not work. If we try to factor it to its two qubit states, since it is a tensor product, we get:

$$ac = \frac{1}{\sqrt{2}}, \qquad ad = 0, \qquad bc = 0, \qquad bd = \frac{1}{\sqrt{2}}$$

Since $ad = 0$, either a or d should equal 0. If $a = 0$ then $ac = 0$ but it does not. If $d = 0$ then $bd = 0$ but it does not either. This product state cannot be factored. This means that we cannot separate these qubits and their values only make sense when they're together. In this product state the qubits have a 50% chance of either collapsing into $|00\rangle$ or $|11\rangle$. [11]

Considering the same state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, we can see how this state would collapse.

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle) + \frac{1}{\sqrt{2}}(|1\rangle \otimes |1\rangle) \tag{13}$$

Since tensor product is an operation over a Hilbert space,

$$\frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle) + \frac{1}{\sqrt{2}}(|1\rangle \otimes |1\rangle) = \frac{1}{\sqrt{2}}|0\rangle\otimes|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\otimes|1\rangle \tag{14}$$

or,

$$\frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle) + \frac{1}{\sqrt{2}}(|1\rangle \otimes |1\rangle) = |0\rangle\otimes\frac{1}{\sqrt{2}}|0\rangle + |1\rangle\otimes\frac{1}{\sqrt{2}}|1\rangle \tag{15}$$

In equation (14) the probability that the first qubit is measured to be $|0\rangle$ is $1/2$ if the second bit has not been measured. However, if the second qubit is measured, first as in equation (15), then the probability that the first qubit is $|0\rangle$ is either 0 or 1 depending on whether the second qubit measured is $|0\rangle$ or $|1\rangle$ respectively. Since the measurement of the second qubit affects the value of the first qubit, by definition, the state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ is an entangled state. Now consider the state $\frac{1}{\sqrt{2}}(|10\rangle + |11\rangle)$. This state is not entangled since,

$$\frac{1}{\sqrt{2}}(|10\rangle + |11\rangle) = |1\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{16}$$

The measurement of the first qubit returns $|1\rangle$ regardless of the measurement of the second bit. Furthermore, the measurement of the second bit will have a fifty-fifty chance of collapsing to either $|0\rangle$ or $|1\rangle$ regardless of whether the first qubit was measured or not. [3]

2.6 Decoherence

Quantum noise or quantum decoherence is the loss of information due to environmental disturbances. Removing quantum decoherence is the greatest challenge facing quantum computation right now. We can describe decoherence using the Bloch sphere from the previous section. The pure states that we defined earlier have a Bloch vector of length 1, which means the states are on the surface of the Bloch sphere. The pure state can be represented by a density matrix $\rho = |\psi\rangle\langle\psi|$. Decoherence causes a change in the quantum states from pure state to mixed state which can have a density matrix. This is written as a sum of pure states $\rho = \sum_s p_s |\psi_s\rangle\langle\psi_s|$ which is a Bloch sphere that sits inside a Bloch sphere $|\langle X\rangle|^2 + |\langle Y\rangle|^2 + |\langle Z\rangle|^2 < 1$. [4]

A state decoheres when the system interacts with the external world. So, it is necessary to isolate the system from the environment to reduce decoherence. These interferences, depending on which implementation of quantum computer is being used, could be anything. If we consider a quantum computer that uses ions as qubits, possible sources of interference could be thermal energy of the environment, other particles interacting with the ion, or even the electric and magnetic fields required to run the machine itself. All of these would cause an ion to change its spin states at random and eventually collapse. This makes scaling difficult because increasing the number of qubits would also increase decoherence. [3]

2.7 Computational Complexity

To understand the power of quantum computing, it helps to compare the computational power of quantum computers in terms of classical computers. In computer complexity theory we classify decision problems into different classes. Decision problems that can be solved in polynomial time by a deterministic Turing machine are classified in the class **P**. The set of decision problems solvable in polynomial time by a non-deterministic Turing machine are in the class NP. Thus, **NP** problems are those that require a non-deterministic Turing machine in order to be solved efficiently but take exponential time on a deterministic machine. The class of NP-complete problems (**NPC**) are the hardest problems in NP. Every NP problem can be reduced to a problem in NPC. If one NPC problem is found to be in P, then all the problems in NP would also be in P, which would solve the classic open problem in computer science of whether P = NP. Most theoretical computer scientists believe that P≠NP, but that is as million-dollar question that no one has been able to answer. [12]

Given an input, a deterministic Turing machine will always have a linear ordered sequence of steps that always leads to the same output. A non-deterministic Turing machine does not have this restricted and ordered computation. A non-deterministic machine can make several moves at a given moment, which allows the computation to branch out into many possible outputs for the same input. There can be none, one, or more than one next configuration, any of which can ultimately lead the computation to an acceptable state, if one exists. It might sound like a quantum computer might be a non-deterministic machine given the qubits can exist in superposition of states. The key reason why a quantum computer is not a non-deterministic Turing machine is because the model of quantum computation is based on a linear combination of basis states that is probabilistic, which means that the accuracy of its output is bounded by some probability that is less than 1. The definition states that the computation model for a non-deterministic machine either returns nothing or branches out. [11]That is, it is not linear. Therefore, we can assign quantum computers to its own class of Turing machine, Quantum Turing Machine.

This brings us to another important complexity class called Bounded-error Probabilistic Polynomial time (**BPP**). This class contains decision problems that can be solved in polynomial time by a probabilistic Turing machine, with an accompanying probability of the solution being wrong. [11] Probabilistic Turing machines are those with access to some source of truly random input which is used to make random decisions. The downside of using randomness is that there will always be a probability that the algorithm will produce a wrong input. In BPP, the error of the solution is bounded in that the probability that the answer is correct must be at least two-thirds. The two-thirds chance of being the correct answer in the definition of a BPP is picked arbitrarily. These types of problems come closer to our realm of quantum computation.

Quantum computation introduces several new complexity classes to the polynomial hierarchy, with Bounded-error Quantum Polynomial time (**BQP**) being the most studied. BQP is the quantum extension of BPP. BQP contains a class of problems solvable in polynomial time by an innately probabilistic quantum Turing machine, with the same error constraint defined for BPP, that the probability that the answer is correct is at least two-thirds. But unlike BPP it is suspected that $P \subset BQP$, which would mean that quantum computers may be capable of solving some problems in polynomial time that cannot be solved efficiently by a classical Turing machine. [12]

3. Quantum Algorithms

Quantum algorithms are algorithms which run on a realistic model of quantum computation. The model widely used and the one we have been using is the quantum circuit model. Any classical algorithm can also be performed on a quantum computer, but it wouldn't make sense to do so. Quantum algorithms refer to those algorithms that use the properties of quantum mechanics, such as quantum superposition or entanglement. [4]

One of the most significant developments in quantum computation occurred in 1997. Peter Shor [13] developed a quantum algorithm that performs prime factorization of integers in polynomial time compared to exponential time in a classical computer. Another algorithm that is important in quantum computation is Lov Grover's quantum database search. [14] This is the algorithm that we will be exploring and will take some key ideas to come up with our own algorithm.

When we examine these algorithms, we need to assume that they will be running in a perfect quantum computer. Most of these algorithms have not been tested out on an actual quantum computer since making quantum computers with larger number of qubits is difficult due to decoherence.

3.1 Grover's Algorithm

Grover's algorithm searches for a specified entry in an unordered database of $N = 2^n$ items where n is the number of qubits. Just like Shor's algorithm, Grover's algorithm also provides a polynomial time speedup compared to its classical counterpart. The best classical search algorithm over unordered data requires $O(N)$ time. Grover's algorithm can perform a quantum search in only $O(\sqrt{N})$ time which would be a quadratic speedup. [14]

Grover's algorithm uses the advantages described in the previous sections to improve on the slower runtimes of classical algorithms. It uses quantum superposition and a key technique in quantum computation known as *amplitude amplification*. This exploits the fact that qubit states have amplitudes associated with them. This applies a Z-gate on the desired state performing a phase shift to a negative amplitude. Even though the amplitude for the state turns negative, the probability of that state stays the same since the probability is the square of the amplitude, so the sign is ignored. After this step we increase the probability of the qubit to be in the state with the differing phase. Amplitude amplification is unique to quantum computing because of the property that different states could have different amplitudes for which there is no counterpart in classical computing. [14]

3.1.1 The Algorithm

Grover's algorithm starts with a quantum register of n qubits, where n is the number of qubits necessary to represent all the records of size $N = 2^n$. These are all initialized to $|0\rangle$:

$$|0\rangle^{\otimes n} = |0\rangle$$

We now take all these qubits and put them into an equal superposition of states by applying the Hadamard gate on all of them. This takes $\Theta(\mathrm{n}) = \Theta(\mathrm{log2^n}) = \Theta(\mathrm{logN})$ operations.

$$|\psi\rangle = H^{\otimes n}|0\rangle^{\otimes n} = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle \tag{17}$$

All these possible states are entries in Grover's database and any one of them could be equal to our search key. Giving all the states the same amplitude lets us consider every element at once.

3.1.1.1 Grover Iteration

The next step is to perform a series of transformations referred to as Grover iteration which performs the amplitude amplification. This does the most work in the algorithm since it will be repeated $\frac{\pi}{4}\sqrt{N}$ times. The quantum computer can be in multiple states at the same time and can carry out multiple tasks on those state at the same time as well.

The specifics of why the iteration is done for $\frac{\pi}{4}\sqrt{N}$ times is described in the paper *Tight bounds on quantum searching* where they do a tight analysis of the Grover's search algorithm. [14]

The first step of Grover iteration is to send the data through a *quantum oracle,* $\boldsymbol{\mathcal{O}}$, that modifies the configuration of the system depending on what the search key is. An oracle is a black-box function which is a system that can be viewed only in terms of its inputs and outputs, with no knowledge of its internal workings. A quantum oracle is a quantum black-box which can observe and modify a qubit without collapsing it to a classical state. Essentially it can recognize the correct state. If the qubit is in the correct state, then an Z-gate is applied to it which will rotate its phase by π radians. Otherwise it does nothing. The phase shift changes the sign of the amplitude marking the correct state. We should note that change in the sign does not matter to the probability of the state. The oracle's effect on $|x\rangle$ can be written as:

$$|x\rangle \overset{\mathcal{O}}{\rightarrow} (-1)^{f(x)}|x\rangle \tag{18}$$

where $f(x) = 1$ if x is the correct state and $f(x) = 0$ otherwise. [12]

The next part of the iteration is the diffusion transform, which performs an inversion about the average. This means that the amplitude of each of the state is transformed such that it is as far above the average as it was below the average prior to the transformation. Another Hadamard gate, $|H\rangle^{\otimes n}$, is applied here and is followed by a phase shift that flips every state except $|0\rangle$ and then ends it with another Hadamard gate. The entire Grover iteration can be represented by:

$$2[|\psi\rangle\langle\psi| - I]\mathcal{O} \tag{19}$$

where I is an identity. Once Grover iteration has been performed an adequate number of times, a classical measurement is performed that collapses the states and determines the result.

3.2 Finding Minimum Using Grover Algorithm

Let's define the problem first. Let $T[0, \ldots, N-1]$ be an unsorted array of N items, each holding a value from an ordered set. Assume that all values are distinct. Find the index y such that $T[y]$ is the minimum value of the unsorted array. This requires at least $O(N)$ time in a classical computer. Christoph Dürr and Peter Høyer, in 1996, [15] came up with an algorithm that solves the problem in $O(\sqrt{N})$ time using Grover's algorithm. This algorithm obviously works for finding the maximum as well.

This algorithm calls on Grover's algorithm as a subroutine to find the index of a smaller item than the value determined by a threshold index. The result is then chosen as the new index and the process is repeated until the probability of the threshold index is sufficiently large enough.

3.2.1 The Algorithm

1. Choose the threshold index $0 \leq y \leq N-1$ at random.
2. Repeat the following and interrupt it when the total running time is more than $22.5\sqrt{N} + 1.4\lg^2 N$ then go to stage 2.c.
    a. Initialize the memory as $\sum_j \frac{1}{\sqrt{N}} |j\rangle|y\rangle$ and mark every item j for which $T[j] < T[y]$.
    b. Apply Grover's algorithm
    c. Observe the first register: let y' be the outcome. If $T[y'] < T[y]$, then set threshold index y to y'.
3. Measure y.

In their paper, *A quantum algorithm for finding the minimum,* Christoph Dürr and Peter Høyer proved a theorem that states, *this algorithm finds the index of the minimum value with probability at least $\frac{1}{2}$. Its running time is $O(\sqrt{N})$.* [15]

This is impressive, but we can do better than this with an algorithm that is more intuitive for the same run time. This algorithm does not require understanding the complexities of Grover's algorithm, even though it does take some inspiration from it.

3.3 Alternate Minimum Algorithm

Let's define the problem in a slightly different manner this time. Given an array with N elements $[x_0, x_1, \dots, x_i, \dots, x_{N-1}]$ where i is the index and $0 \leq i \leq N-1,\ N \leq 2^n$. Find the index i such that x_i is the minimum element in the array.

This algorithm uses Grover iteration which is denoted by:

$$2[|\psi\rangle\langle\psi| - I]\mathcal{O}$$

For the sake of simplicity, we are going to break down Grover iteration into simple mathematical operations.

3.3.1 The Algorithm

1. Take all the data and apply the Hadamard gate to all the qubits:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |x_i\rangle\, |i\rangle$$

2. Now consider the first digit of each of the states. Given a state $|101\rangle$ we will be looking at the left most $|1\rangle$. We also want to keep track of how many of the first digits are $|0\rangle$s since this determines the number of iterations of step 2. Let the number of iterations be j. The number of iterations here is set to j because we only care about the numbers that have the digit $|0\rangle$ in the beginning. Doing so prevents an increase in amplitude of large numbers. We will see this in a worked example below. In a case where all the states have $|1\rangle$ or $|0\rangle$ as their first digit, we

set j as the number of $|0\rangle$s in the next digit place and so on. During the current, if all the states have the same number, either all $0s$ or all $1s$, in their digit place then we skip this iteration and continue the remaining iterations starting with the next digit place. We do this so that we only iterate through the relevant digits and do not perform any iterations.

a. Here we will be using the same quantum oracle from Grover's algorithm.

$$|x\rangle \overset{O}{\rightarrow} (-1)^{f(x)}|x\rangle|i\rangle$$

where $f(x) = 1$ and increase counter j by 1 when the first digit is $|0\rangle$ and $f(x) = 0$ otherwise. This marks all the states where there first digit is $|0\rangle$, shifting their phase by -1. If a digit place has all $|1\rangle$s or all $|0\rangle$s in all the states, then do not iterate through that and iterate through the next digit place.

b. We need to calculate a new value here, call it μ, where μ is the average of all the states:

$$\mu = \frac{1}{N}\sum_{i=0}^{N-1} \alpha_i|x_i\rangle\,|i\rangle$$

Here α_i is the amplitude of the state $|x_i\rangle$ after the phase shift from 2.a.

c. We now do the amplitude amplification to all the states. Now change each amplitude, α_i, to the new amplitude α_i':

$$\alpha_i' = 2\mu - \alpha_i$$

This makes sure that the sum of all the α_i^2 remains 1 while changing the amplitude of the desired states.

d. This step is just to clean things up. Perform a phase shift, Z-gate, for any amplitude that is negative so that we can normalize the amplitude for the next steps.

Repeat step 2 j times or n bit times, whichever comes first.

3. Measure $|i\rangle$.

3.3.2 Worked Example

Consider a system with $n = 3$ qubits and an unordered array with $N = 5$ elements. Let the elements be $[111, 101, 010, 001, 110]$ represented by the state:

$$|\psi\rangle = \sum_{i=0}^{N-1} \alpha_i |x_i\rangle\,|i\rangle \tag{20}$$

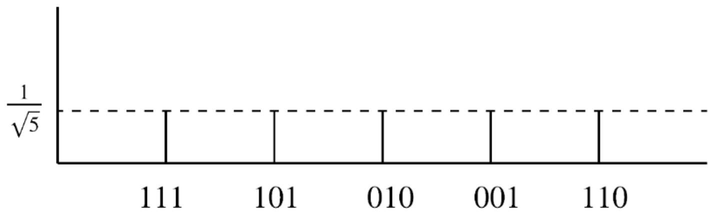
$$|\psi\rangle = \alpha_0|111\rangle|0\rangle + \alpha_1|101\rangle|1\rangle + \alpha_2|010\rangle|2\rangle + \alpha_3|001\rangle|3\rangle + \alpha_4|110\rangle|4\rangle$$

where α_i is the amplitude of the state and i is the index. The first step according to the algorithm is to apply a Hadamard gate to all the qubits:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} \alpha_i |x_i\rangle\,|i\rangle \tag{21}$$

$$|\psi\rangle = \frac{1}{\sqrt{5}}|111\rangle|0\rangle + \frac{1}{\sqrt{5}}|101\rangle|1\rangle + \cdots + \frac{1}{\sqrt{5}}|110\rangle|4\rangle$$

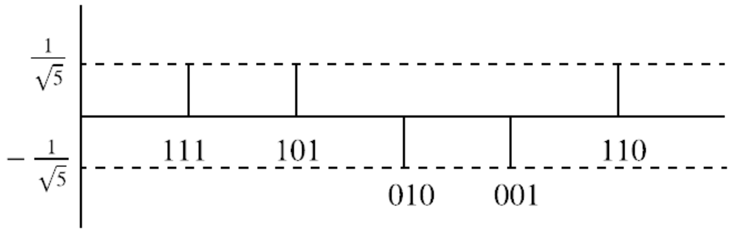
We can also represent this in an amplitude graph:



The next step is to apply the oracle, $\mathcal{O}$, defined in the step 2.a. Doing this flips the phases of all the numbers with first qubit $|0\rangle$. There are two qubits with $|0\rangle$ as its first digit so we will iterate twice through step 2.

First Iteration:

$$|\psi\rangle = \frac{1}{\sqrt{5}}|111\rangle|0\rangle + \frac{1}{\sqrt{5}}|101\rangle|1\rangle - \frac{1}{\sqrt{5}}|010\rangle|2\rangle - \frac{1}{\sqrt{5}}|001\rangle|3\rangle + \frac{1}{\sqrt{5}}|110\rangle|4\rangle \quad (22)$$

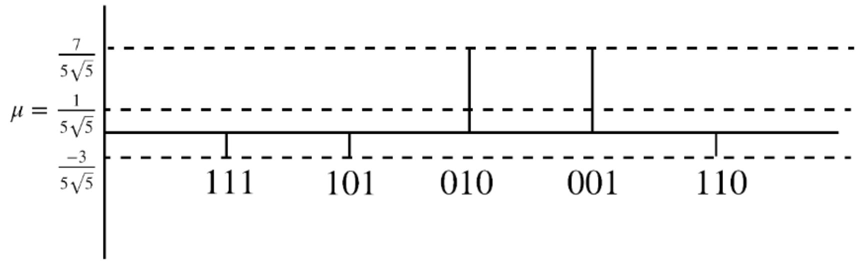


This has marked the state $|010\rangle$ and $|001\rangle$ to be amplified. We now need to find the average amplitude, μ, of all the states after the oracle has been applied. In this case:

$$\mu = \frac{1}{N}\sum_{i=0}^{N-1} \alpha_i |x_i\rangle\, |i\rangle$$
$$\mu = \frac{1}{5}\left(\frac{1}{\sqrt{5}} + \frac{1}{\sqrt{5}} - \frac{1}{\sqrt{5}} - \frac{1}{\sqrt{5}} + \frac{1}{\sqrt{5}}\right) = \frac{1}{5\sqrt{5}} \tag{23}$$

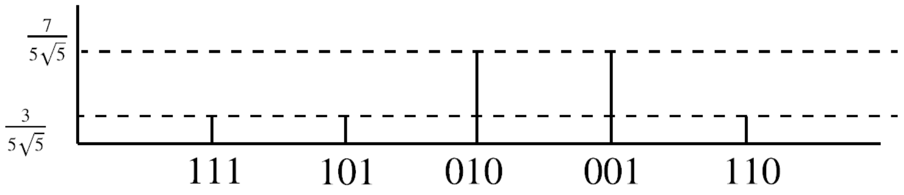
We now go to step 2.c. where we do an inversion over the average.

$$|\psi\rangle = -\frac{3}{5\sqrt{5}}|111\rangle|0\rangle - \frac{3}{5\sqrt{5}}|101\rangle|1\rangle + \frac{7}{5\sqrt{5}}|010\rangle|2\rangle + \frac{7}{5\sqrt{5}}|001\rangle|3\rangle - \frac{3}{5\sqrt{5}}|110\rangle|4\rangle$$



Now we normalize the amplitudes, getting the state ready for the second iteration. We do a phase shift for all the amplitudes that are negative.

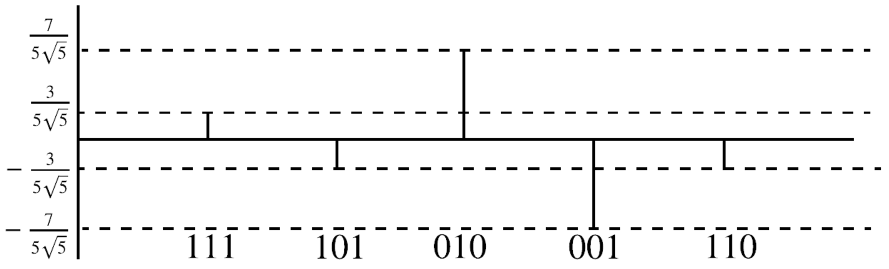
$$|\psi\rangle = \frac{3}{5\sqrt{5}}|111\rangle|0\rangle + \frac{3}{5\sqrt{5}}|101\rangle|1\rangle + \frac{7}{5\sqrt{5}}|010\rangle|2\rangle + \frac{7}{5\sqrt{5}}|001\rangle|3\rangle + \frac{3}{5\sqrt{5}}|110\rangle|4\rangle$$



Second Iteration:

We restart step 2 for this state. We now flip the amplitudes if the second digit is $|0\rangle$, which gives us:

$$|\psi\rangle = \frac{3}{5\sqrt{5}}|111\rangle|0\rangle - \frac{3}{5\sqrt{5}}|101\rangle|1\rangle + \frac{7}{5\sqrt{5}}|010\rangle|2\rangle - \frac{7}{5\sqrt{5}}|001\rangle|3\rangle + \frac{3}{5\sqrt{5}}|110\rangle|4\rangle$$

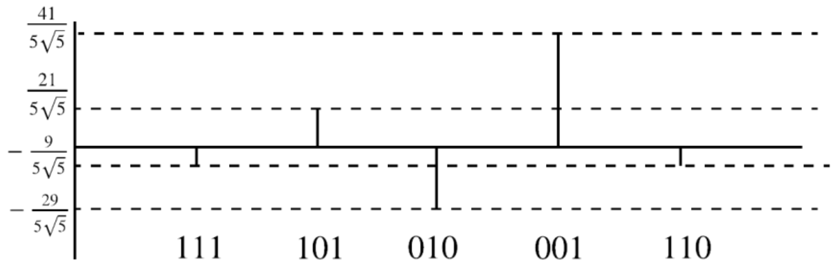


We now find the new average of amplitudes.

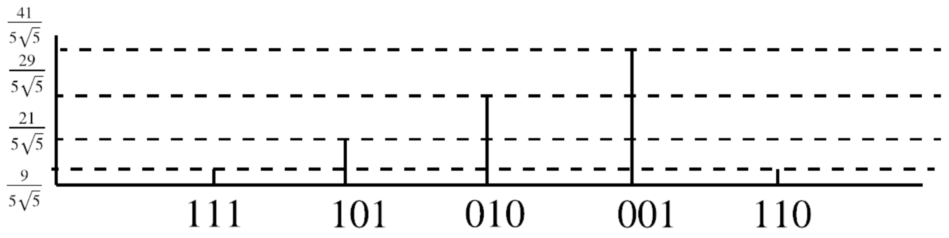
$$\mu = \frac{1}{5}\left(\frac{3}{5\sqrt{5}} - \frac{3}{5\sqrt{5}} + \frac{7}{5\sqrt{5}} - \frac{7}{5\sqrt{5}} + \frac{3}{5\sqrt{5}}\right) = \frac{3}{25\sqrt{5}} \quad (24)$$

The next step is inversion over the average:

$$|\psi\rangle = -\frac{9}{25\sqrt{5}}|111\rangle|0\rangle + \frac{21}{25\sqrt{5}}|101\rangle|1\rangle - \frac{29}{25\sqrt{5}}|010\rangle|2\rangle + \frac{41}{25\sqrt{5}}|001\rangle|3\rangle - \frac{9}{25\sqrt{5}}|110\rangle|4\rangle$$



We now normalize the amplitudes again and terminate step 2.



We now go to the last step which is measurement of the index, $|i\rangle$, of each state. We can find the probability of each of the indices by squaring the amplitudes of each of them. If we do this measurement multiple times we get the following probability distribution:

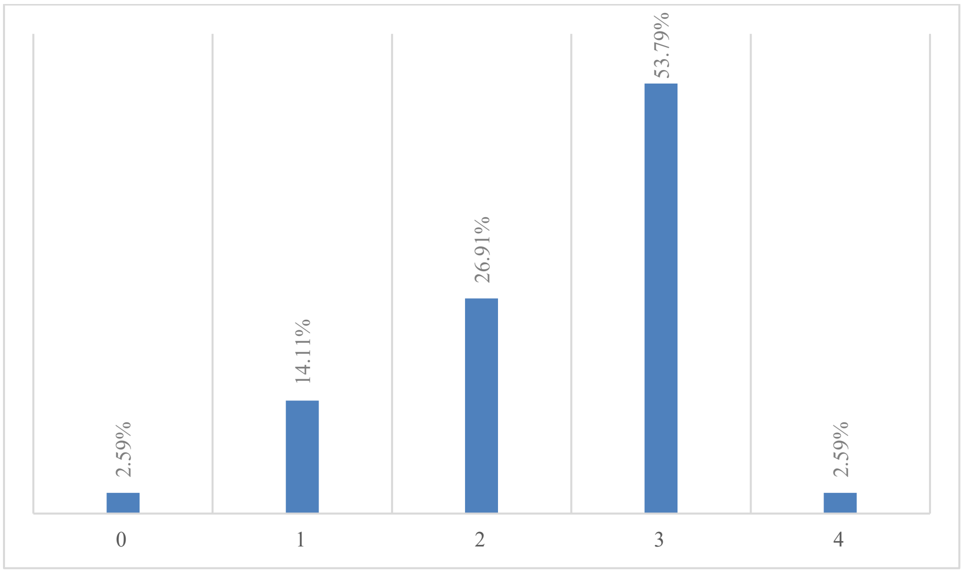


FIG. 7: Probability distribution achieved by squaring the magnitudes of each state

The third index has the highest probability which is what we wanted since the third index corresponds to the value 001. We can see that the probability of it being the right answer is 53.79%. This follows Christoph Dürr and Peter Høyer's theorem that their algorithm gives the minimum value at least half the time.

The runtime of this algorithm depends on the number of states that have 0 as the first digit. The worst case is when every state, but one has a 1 as its first digit. The most it would iterate over is $O(n) = O(lgN)$ steps. Recall that the amplitudes are being manipulated for each step simultaneously at a given time. Step 2 of the algorithm is like the Grover iteration which does two Hadamard transforms and applies $O(n)$ gates to perform the conditional phase shifts. This makes the total runtime of Grover iteration to be $\Theta(2n)$. Grover states in his paper that it is not possible to identify the desired element in fewer than $\Omega(\sqrt{N})$ steps. Similarly, we conclude that the runtime of this alternate minimum algorithm is $\Omega(\sqrt{N})$.

3.4 Proof of Alternate Minimum Algorithm

Theorem *The alternate minimum algorithm will always find a minimum value.*

Consider an array $T = [x_0, x_1, x_2, \dots, x_N]$ with N elements where all x_i are unique and unordered in T.

Also consider an algorithm that perfectly finds the index of the minimum element. Say the algorithm returns the index m such that $T[m] = x_m$ where x_m is the minimum element.

Now consider the alternate minimum algorithm and execute on T. Suppose this algorithm returns the index k such that $T[k] = x_k$.

Case 1: If $m = k$, then:

The algorithm works, and we're done.

Case 2: If $T[m] > T[k]$, then:

In this case, we have found a new minimum. This is a contradiction since T has as unique minimum.

Case 3: If $T[m] < T[k]$, then:

According to the problem, the minimum index should have the highest amplitude/probability. In this case the index k has the highest probability. We can track the amplitude of m and k through the algorithm. When the algorithm first runs there must be a first point where the c^{th} digit for m is 0 and k is 1 since, in this case $T[m] < T[k]$. So,

$$T[m] = \underbrace{\dots\dots}_{a}\underbrace{0}_{c}\underbrace{\dots\dots}_{b} \qquad T[k] = \underbrace{\dots\dots}_{a}\underbrace{1}_{c}\underbrace{\dots\dots}_{b}$$

where a is the number of digits before the digit in the c^{th} position and the first a digits are the same for both the numbers. b is the number of digits after the c^{th} position and is not the same for both the numbers. According to the algorithm this suggests that the amplitude of position m should have been amplified and the amplitude of k should have decreased. Which means that the amplitude k should be less than the amplitude of m, so k should not have been picked.

3.5 Observations and Final Remarks

The alternate minimum algorithm is an intuitive method of finding the minimum or a maximum of an array. Unlike Christoph Dürr and Peter Høyer's algorithm for finding the minimum, it does not need to compare the values of the actual states with each other. It also does not necessarily produce just one highest probability with all other lower probabilities equal as Grover's algorithm does. We should also note that the algorithm is reversible up until the measurement. All the transformations, like the Hadamard gate, oracle function, and the diffusion transformation, are reversible.

One caveat of this algorithm is that it only works for a subset of possible cases. The current design of the algorithm requires that the number of binary numbers with first digit 0 should be strictly less than the number of binary numbers with first digit 1. If we run the algorithm on a set that has the same number of both $0s$ and $1s$ in the first digit then the average μ is 0. If $\mu = 0$, then there would be no change in amplitude for any of the states while performing the inversion over the average. This would ignore the first digit of all the states, the digit that has the biggest impact on finding the minimum. If we run the algorithm on a set that has more $0s$ than $1s$ in its first digit, the average, μ will be negative. If $\mu < 0$, then performing the inversion over the average would decrease the amplitude of the states with negative amplitude and increase the amplitude of the states with positive amplitude.

One disadvantage of not comparing the values is that in cases where the number of qubits is large and the number of elements in the unsorted array is small, then the algorithm examines all the n qubits in the worst case instead of doing just 3 comparisons. Another downside of this algorithm is that it uses extra qubits to keep track of how many iterations it needs to do. Further improvements need to be done so that we no longer need to keep track of the number of iterations we need to do. A possible solution might be to use weighted increase or decrease in the amplitude so that the sum of their probabilities remains 1. The weight would depend on which digit place we are iterating over.

In Grover's algorithm, the Grover iteration can be run multiple times so that the probability that the searched item is found is higher. We can do something similar with the alternate minimum algorithm. After the algorithm runs the first time, instead of measuring the value we can run the algorithm again on the superposition of states that resulted from running the algorithm the first time. We can do this multiple time so that we amplify the amplitude of the results. This algorithm also takes care of a problem with multiple solutions. Assume the solution lies in indices 4 and 5. After running the algorithm, the probability that it is 4 or 5 would be the same. If the probability is high enough there would be a fifty-fifty percent chance that it is either 4 or 5.

Currently we have some limitations running this and other algorithms in a quantum computer. The error free quantum computers we do have right now are made up of only a few qubits. This is not enough to create a simple data structure like an array. It is also difficult to store any data in a quantum computer. Without being able to store data and trying to run it in a quantum computer we wouldn't be able to build an unordered list of any size $N \leq 2^n$. We would have to consider all $N = 2^n$ states and it would be difficult to associate them with an index. Further research is being done for storing quantum data storage.

References

[1] J. Strickland, "How small can CPUs get?," [Online]. Available: https://computer.howstuffworks.com/small-cpu1.htm.

[2] P. Dettmer, "Quantum Computers Explained – Limits of Human Technology," Kurzgesagt, 8 December 2015. [Online]. Available: https://www.youtube.com/watch?v=JhHMJCUmq28.

[3] M. A. Nielsen and L. I. Chuang, Quantum Computation and Quantum Information 10th Anniversary Edition, Cambridge University Press, 2010.

[4] I. R. a. I. Q. team, "IMB Q Experience Documentation," IBM, 2017. [Online]. Available: https://quantumexperience.ng.bluemix.net/qx/tutorial?sectionId=beginners-guide&page=introduction.

[5] T. Rowland, "Vector Space Tensor Product," A Wolfram Alpha Web Resource, created by Eric W. Weisstein, [Online]. Available: http://mathworld.wolfram.com/VectorSpaceTensorProduct.html.

[6] E. W. Weisstein, "Conjugate Transpose," MathWorld--A Wolfram Web Resource, [Online]. Available: http://mathworld.wolfram.com/ConjugateTranspose.html.

[7] E. W. Weisstein, "Metric Space," MathWorld--A Wolfram Web Resource, [Online]. Available: http://mathworld.wolfram.com/MetricSpace.html.

[8] E. W. Weisstein, "Complete Metric Space," MathWorld--A Wolfram Web Resource, [Online]. Available: http://mathworld.wolfram.com/CompleteMetricSpace.html.

[9] "WolframAlpha," [Online]. Available: https://www.wolframalpha.com/input/?i=hilbert+space.

[10] E. W. Weisstein, "Hilbert Space," MathWorld--A Wolfram Web Resource, [Online]. Available: http://mathworld.wolfram.com/HilbertSpace.html.

[11] W. P. Eleanor G. Rieffel, "An Introduction to Quantum Computing for Non-Physicists," *ACM Computing Surveys,* 2000.

[12] E. Strubell, "An Introduction to Quantum Algorithms," *COS498 Chawathe Spring,* 2011.

[13] P. W. Shor, "Algorithms for quantum computationion: discrete lograithms and factoring," *35th annual symposium of fundamentals of comp. science,* 1994.

[14] L. K. Grover, "A Fast Quantum Mechanical Algorithm for Database Search," *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing,* 1996.

[15] P. H. Christoph Durr, "A quantum algorithm for finding the minimum," *quant-ph/9602016,* 1996.

[16] E. W. Weisstein, "Cauchy Sequence," MathWorld--A Wolfram Web Resource, [Online]. Available: http://mathworld.wolfram.com/CauchySequence.html.

[17] D. Terr, "Bloch Sphere," MathWorld--A Wolfram Web Resource, [Online]. Available: http://mathworld.wolfram.com/BlochSphere.html.

[18] G. B. P. H. A. T. Michel Boyer, "Tight bounds on quantum searching," *PhysComp,* 1996.