Quantum Computing and its Implementation

Ashish Kumar, Paula Medina, Lewis westfall, Avery M. Leider, and Charles C. Tappert
Seidenberg School of Computer Science and Information Systems, Pace University
Pleasantville, NY 10570, USA
Email: {ak33894n,ph79486,lwestfall, aleider, ctappert}@pace.edu

***Abstract*—Quantum computers have achieved a solid base in both physics and maths. However, we still need a breakthrough when we talk about technology. We need more development in the quantum algorithm for them so we can achieve more accurate results. To work on this algorithm one needs to have information on the hypothesis of Quantum computing. In this paper we will be talking about Quantum computation and how elements like linear algebra and phenomena like superposition and entanglement needed to understand the quantum computing process. Some algorithm that works on computation theory will also be discussed. Ex: Shor's algorithm.**

***Index Terms*—Quantum Computing, Qubits, Linear Algebra, Shor's Algorithm**

I. Introduction

Quantum computing is one of the leading applications of quantum physics. It has the potential to solve some of the most complex problems which are beyond the reach of even today's most powerful supercomputers, quantum computers are not going to replace classical computers but a different way of operating enables them to perform some calculations that classical computers cannot perform efficiently [1]. Unlike classical computer which works on binary bits i.e. Zeros and ones Quantum computers work on 0, 1, or any value in between including having an imaginary component two or more qubits can be entangled so that they function as a single unit and not as individual qubits.

Hilbert space is one of the basic mathematical operators used in Quantum Mechanics. The mathematical concept of Hilbert space named after David Hilbert, it generalizes the notion of Euclidean space. It extends the method of vector algebra and calculus from two dimensional and three-dimensional space to spaces with any finite or infinite number of dimensions [2]. Hilbert space uses all properties of vector space with some additional properties. The additional property it has is inner product operation. The inner product is an operation that takes two vectors and results in a scalar.

$$\langle \cdot , \cdot \rangle : V \times V \to F$$

Fig. 1. Inner Product

For Hilbert space to have a valid inner products operation it has to obey certain constraints. Constraints are as below

1) Conjugates Symmetry:

$$\langle x, y \rangle = \overline{\langle y, x \rangle} \tag{1}$$

2) Linearity w.r.t to second vector:

$$\begin{gathered} \langle ax, y \rangle = a\langle x, y \rangle \\ \langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle \end{gathered} \tag{2}$$

3) Antilinear w.r.t first vector:

$$\langle x, x \rangle > 0, \quad x \in V \setminus \{\mathbf{0}\}. \tag{3}$$

Two types of Hilbert spaces that are important in Quantum Mechanics are:

A. Finite-Dimensional Hilbert Space

Examples of these vector spaces are $\mathbb{R}^{\ltimes}$ and $\mathbb{C}^{\ltimes}$ where $\mathbb{C}^{\ltimes}$ are set of n tuples of Complex numbers and $\mathbb{R}^{\ltimes}$ is the set of n tuples of Real numbers. These are finite because vectors necessary to form basis in these vector spaces is finite. Like any typical Hilbert space we need to have well defined inner product. Inner product on $\mathbb{R}^{\ltimes}$

$$X_1 = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}, X_2 = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{4}$$

The inner product of

$$(X_1 * X_2) = ({X_1}^T * X_2) = (a_1)(b_1) + ... + (a_n)(b_n) \tag{5}$$

Inner product on $\mathbb{C}^{\ltimes}$

$$Z_1 = \begin{bmatrix} a_1 + b_1 i \\ a_2 + b_2 i \\ \vdots \\ a_n + b_n i \end{bmatrix}, Z_2 = \begin{bmatrix} c_1 + d_1 i \\ c_2 + d_2 i \\ \vdots \\ c_n + d_n i \end{bmatrix}, i^2 = -1 \tag{6}$$

The inner product of complex number is similar to inner product of real number except for the first vector we are using conjugate transpose we are taking conjugate of Z_1 and then taking transpose of resulting conjugate vector

$$(Z_1 * Z_2) = ({Z_1}^{\dagger} * Z_2) = \begin{bmatrix} a_1 + b_1 i & \dots \end{bmatrix} \begin{bmatrix} c_1 + d_1 i \\ c_2 + d_2 i \\ \vdots \\ c_n + d_n i \end{bmatrix} \tag{7}$$

B. Infinite-Dimensional Hilbert Space

Example of these is the Vector space of complex valued functions which has the following inner product where the inner product of two functions

$$\langle \phi, \psi \rangle = \int_{-\infty}^{\infty} \phi \psi dx \tag{8}$$

Because of this inner product we cannot use any function in our function space we have to use a space whole class of functions called square integral functions.The end result of equation 8 will be infinite so in order for result to have finite solution we use Squared integral function.

$$\int_{-\infty}^{\infty} |\phi|^2 dx = finite \tag{9}$$

US Government at this point is ready to give money to research on challenge currently been faced in Hilbert space. The challenge being that as the number of qubits in quantum system increases, the Hilbert space that defines the system grows exponentially, and the resources needed for complete characterization correspondingly grow exponentially [3].

Shor's algorithm is one of the best examples of Quantum computing. Peter Shor came up with this idea in the nineties. It suggests that quantum mechanics allows factorization to be done in polynomial time instead of exponential time which could have a dramatic impact on the field of data security. This is so amazing because if we can get factorization done in polynomial time frame as compared to present exponential time then it would mean that the data security which is based on idea of prime factorization with the large number is not going to have a lot of difficulty in maintaining security that's why it is interesting and it also provides better ways of creating cryptography [4]. Google in 2019 announced its latest 54-cubit quantum computer named Sycamore, able to achieve Quantum supremacy. Sycamore processor was able to perform a calculation in 200 seconds that would have taken the world's most powerful supercomputer 10,000 years [5]. Quantum supremacy is a point where a quantum computer will compute a real-time problem that runs faster than a classical computer [6]. It does not mean that classical computers will no longer be needed. Just that for some procedures quantum computers will be able to do them in a 'reasonable' amount of time while a classical computer cannot. Classical computers will still be needed to prepare data and algorithms for use by the quantum computer and to process the output from the quantum computer. Classical computers will still be used for most of the regular processing. There are many companies Microsoft, Intel working on designing Quantum Computers.

Our project is to present key elements needed to understand the process of Quantum computers

II. Literature Review

Jozef Gruska [7] book, on quantum computing starts with basic concepts, models, methods and results presented in a systematic way. This book emphasis more on computational aspects, models, methods and problems than on the details on technological features which are of importance for implementation of quantum information processing systems.

Nielsen and Chuang [8]This book provides an introduction to the main ideas and techniques of the field of quantum computation and quantum information. Rapid progress in this field and its cross-disciplinary nature have made it difficult for newcomers to obtain a broad overview of the most important techniques and results of the field.This book contains twofold. First, it introduce the basic concept related to computer science, mathematics and physics necessary to understand quantum computation and quantum information and second purpose of the book is to develop in detail the central results of quantum computation and quantum information.A review in the November 2001 edition of Foundations of Physics says, ”Among the handful of books that have been written on this new subject, the present volume is the most complete and comprehensive [9].

III. Project requirements

Classical computers encode information in bits and each bit can represent 0 or 1 this bit act as on-off switch that ultimately translate into computing functions to perform a simple calculation like solving a maze a classical computer will test each possible route one at time to find the correct one, just as classical computer has bits quantum computer have qubits [10]. Quantum bit is the foundation for Quantum computers. Qubits make use of two key principles of quantum physics superposition and entanglement [11]. Superposition means that each qubit can represent a zero a one or both at the same time and entanglements happen when two qubits in a superposition are correlated with one another meaning state of one whether it is 0 or 1 or both depends on the state of another using this two principles qubits can act as much more sophisticated version of switches helping quantum computers solve difficult problems that are virtually impossible for classical computers.

A. Linear Algebra

The language quantum mechanics is linear algebra. linear algebra is a study of linear transformation and the entities they act on vectors [1].

Because the qubit is represented by a vector and the operations on the qubit or multiple qubits are done by matrix manipulation, some knowledge of linear algebra is required. Don't worry, the amount you need is only a small part of linear algebra and also the easier part [6].

The essentials [5]

Scalar A number, which can be complex, eg. a + bi

Vector A vector has a magnitude and a direction. A column vector is a n x 1 matrix, while a row vector is a 1 x n matrix. Each element of a vector is a scalar and represents a dimension.

- Vector is a mathematical concept.

- Ket - $|\varphi\rangle$ - is a physics concept and is a vector in Hilbert space and is a column vector.
- Bra - $\langle\varphi|$ - is the conjugate transpose of a ket and is a row vector.

$$|\varphi\rangle = \left[\begin{array}{c} a \\ b \end{array} \right] \tag{10}$$

$$\langle\varphi| = \left[\begin{array}{cc} a & b \end{array} \right] \tag{11}$$

Matrix A matrix is n x m and every element is a scalar. Below is a 3 x 4 matrix.

$$\left[\begin{array}{cccc} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \end{array} \right] \tag{12}$$

Vector and Matrix Addition To do vector or matrix addition, simply add the corresponding element.

$$\left[\begin{array}{c} a_0 \\ a_1 \\ a_2 \end{array} \right] + \left[\begin{array}{c} b_0 \\ b_1 \\ b_2 \end{array} \right] = \left[\begin{array}{c} a_0 + b_0 \\ a_1 + b_1 \\ a_2 + b_2 \end{array} \right] \tag{13}$$

$$\left[\begin{array}{cc} a_{00} & a_{01} \\ a_{10} & a_{11} \end{array} \right] + \left[\begin{array}{cc} b_{00} & b_{01} \\ b_{10} & b_{11} \end{array} \right] = \left[\begin{array}{cc} a_{00} + b_{00} & a_{01} + b_{01} \\ a_{10} + b_{10} & a_{11} + b_{11} \end{array} \right] \tag{14}$$

Multiplication by a scalar To multiply by a scalar, simply multiply each element by the scalar.

$$s \left[\begin{array}{c} v_1 \\ v_2 \end{array} \right] = \left[\begin{array}{c} sv_1 \\ sv_2 \end{array} \right] \tag{15}$$

$$s \left[\begin{array}{cc} a_1 & b_1 \\ a_2 & b_2 \end{array} \right] = \left[\begin{array}{cc} sa_1 & sb_1 \\ sa_2 & sb_2 \end{array} \right] \tag{16}$$

Transpose The superscript T identifies the transpose operation. The transpose is done by swapping the top right element with the bottom left element and similarly swapping each element on one side of the diagonal with the corresponding element on the other side of the diagonal. The diagonal itself is not changed.

$$\left[\begin{array}{ccc} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{array} \right]^T = \left[\begin{array}{ccc} a_{00} & a_{10} & a_{20} \\ a_{01} & a_{11} & a_{21} \\ a_{02} & a_{12} & a_{22} \end{array} \right] \tag{17}$$

Taking the transpose of a column vector gives a row vector and the transpose of a row vector is a column vector.

$$\left[\begin{array}{c} a \\ b \end{array} \right]^T = \left[\begin{array}{cc} a & b \end{array} \right] \tag{18}$$

$$\left[\begin{array}{cc} a & b \end{array} \right]^T = \left[\begin{array}{c} a \\ b \end{array} \right] \tag{19}$$

Conjugate The superscript * identifies the conjugate operation. The conjugate is taken by changing the sign of the imaginary part of each element. Minus (-) to plus (+) and plus to minus. If the vector or matrix has no imaginary part, then there is no change.

$$\left[\begin{array}{cc} a+bi & c-di \\ m+ni & s+ti \end{array} \right]^* = \left[\begin{array}{cc} a-bi & c+di \\ m-ni & s-ti \end{array} \right] \tag{20}$$

Conjugate Transpose The superscript dagger (†) indicates the conjugate transpose operation. First the conjugate of the vector or matrix is taken and then the transpose.

$$\left[\begin{array}{cc} a+bi & c-di \\ m+ni & s+ti \end{array} \right]^\dagger = \left[\begin{array}{cc} a-bi & m-ni \\ c+di & s-ti \end{array} \right] \tag{21}$$

Product of a vector and a matrix The result of this operation is vector with the size and shape of the original vector. The size of the vector can be 1xn or nx1 and the size of the matrix must be nxn.

$$\left[\begin{array}{cc} a & b \\ c & d \end{array} \right] \left[\begin{array}{c} e \\ f \end{array} \right] = \left[\begin{array}{c} ae+bf \\ ce+df \end{array} \right] \tag{22}$$

$$\left[\begin{array}{cc} a & b \end{array} \right] \left[\begin{array}{cc} e & g \\ f & h \end{array} \right] = \left[\begin{array}{cc} ae+bf & ag+bh \end{array} \right] \tag{23}$$

Inner product The inner product of two vectors is a scalar. The inner product of two matrices of size n x m and m x p is a matrix of size n x p. Both are represented by $\langle A \mid B \rangle$.

$$\left[\begin{array}{cc} a & b \end{array} \right] \left[\begin{array}{c} c \\ d \end{array} \right] = ac + bd \tag{24}$$

$$\left[\begin{array}{cc} a & b \\ c & d \end{array} \right] \left[\begin{array}{cc} e & g \\ f & h \end{array} \right] = \left[\begin{array}{cc} ae+bf & ag+bh \\ ce+df & cg+dh \end{array} \right] \tag{25}$$

Tensor product

$$|A\rangle \otimes |B\rangle = |A\rangle\,|B\rangle = |AB\rangle \tag{26}$$

$$A = \left[\begin{array}{cc} a & b \\ c & d \end{array} \right], B = \left[\begin{array}{cc} e & g \\ f & h \end{array} \right] \tag{27}$$

$$\begin{aligned} |AB\rangle &= \left[\begin{array}{cc} a & b \\ c & d \end{array} \right] \otimes \left[\begin{array}{cc} e & g \\ f & h \end{array} \right] \\ &= \left[\begin{array}{cc} a \left[\begin{array}{cc} e & g \\ f & h \end{array} \right] & b \left[\begin{array}{cc} e & g \\ f & h \end{array} \right] \\ c \left[\begin{array}{cc} e & g \\ f & h \end{array} \right] & d \left[\begin{array}{cc} e & g \\ f & h \end{array} \right] \end{array} \right] \\ &= \left[\begin{array}{cccc} ae & ag & be & bg \\ af & ah & bf & bh \\ ce & cg & de & dg \\ cf & ch & df & dg \end{array} \right] \end{aligned} \tag{28}$$

All this linear algebra are needed for Quantum computing

IV. Methodology

Quantum computation can be done by applying a network of quantum logic gates which also known as Quantum circuit. Quantum circuit consists of n bits of information which has 2^n possible states. Each state is called Bell states, Bell states tells about the probability of a pair of entangled qubits either in the first configuration or the second configuration in any of the given four Bell states.

The five quantum gates we will use are the identity (I), Pauli-X (X), Pauli-Y (Y), Pauli-Z (Z), and the Hadamard (H).

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

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

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

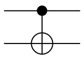
$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

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

We use one multiple qubit gate, the controlled not or CNOT gate. CNOT operator is a 2 qubit unitary transformation where one qubit acts as a control and another act as a target. If the control qubit is set to 1 then the target qubit is flipped The matrix operation for the CNOT is

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

and In Quantum circuit diagram CNOT gate black dot-connecting to qubit wire is for control bit and circle connecting to a wire is target qubit



The ket values $|0\rangle$ and $|1\rangle$ correspond to the column vectors $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ respectively.

To create entangled pair of qubits we combine the use of Hadamard gate with CNOT gate.Below figure shows circuit for Entanglement [12].

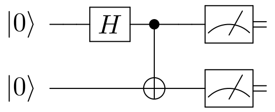


Fig. 2. Quantum Circuit for Entanglement

The diagram below shows the effect of combining Hadmard gate and CNOT gate to generate first Bell states given both qubit starts in the $|0\rangle$ initial state. If we wanted to generate different Bell states then all we need to do is change the initial state of the qubits.

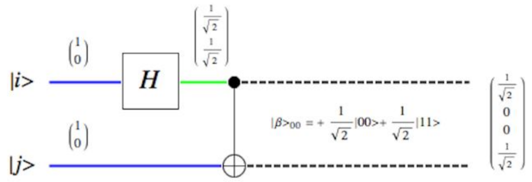

|i>
|j>
$\binom{1}{0}$
$\binom{1}{0}$
H
$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}$
$|\beta\rangle_{00} = + \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$
$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix}$

Fig. 3. Generating Bell State

Suppose if we want to recover the initial state of qubits that is before entanglement all we need to do is mirror the use of Controlled not and Hadamard gates as shown in the figure below.

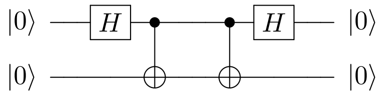


Fig. 4. Reversible Quantum Circuit

By doing this we can undo the effects of Hadamard and controlled gate that puts the qubits an entangled state. This is possible because quantum gate operators are unitary transformations that are reversible. when we reapply these operators it returns us the original initial qubit condition. This means that if we were to give someone entangled qubits in some unknown Bell state all they need to do to recover the initial condition is to apply Controlled Not and Hadamard gates to the pair of qubits.

References

[1] daytonellwanger, “Introduction to quantum computing (11) - the qubit,” https://www.youtube.com/watch?v=Co1gWMjVPBI&list=PLIxlJjN2V90w3KBWpELOE7jNQMICxoRwc&index=13, 2017.

[2] Wikipedia contributors, “Hilbert space — Wikipedia, the free encyclopedia,” https://en.wikipedia.org/w/index.php?title=Hilbert_space&oldid=944890999, 2020, [Online; accessed 14-March-2020].

[3] “Quantum characterization of intermediate scale systems.” [Online]. Available: https://drive.google.com/file/d/14vi4yCFIvE35AqLKfK5cA1tTAKtFbAPF/view

[4] P. D. Goswami, “Basics of shor’s algorithm,” https://www.youtube.com/watch?v=FA21Dj2l3Ac, 2017.

[5] Ronald Frank PhD, *Linear Algebra Review*, class notes.

[6] L. Westfall and A. Leider, “Teaching quantum computing,” in *Proceedings of the Future Technologies Conference*. Springer, 2018, pp. 63–80.

[7] J. Gruska, *Quantum computing*. Citeseer, 1999, vol. 2005.

[8] M. A. Nielsen and I. Chuang, “Quantum computation and quantum information,” 2002.

[9] S. P. Gudder, “Book review: Quantum computation and quantum information. by michael a. nielsen and isaac l. chuang. cambridge university press, cambridge, united kingdom, 2000, i–xxv+ 676 pp., (hardcover),” 2001.

[10] I. T. Academy, “How it works: Quantum computing,” https://www.youtube.com/watch?v=WVv5OAR4Nik, 2017.

[11] C. G. Almudever, L. Lao, X. Fu, N. Khammassi, I. Ashraf, D. Iorga, S. Varsamopoulos, C. Eichler, A. Wallraff, L. Geck *et al.*, “The engineering challenges in quantum computing,” in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. IEEE, 2017, pp. 836–845.

[12] Steve Atkin, “Demystifying Superdense Coding,” 2018. [Online]. Available: https://medium.com/qiskit/demystifying-superdense-coding-41d46401910e