SRI VENKATESWARA INTERNSHIP PROGRAM FOR RESEARCH IN ACADEMICS (SRI – VIPRA)

SRI-VIPRA

Project Report of 2025: SVP-2531

Quantum Information and Computation : Algebra and Algorithms

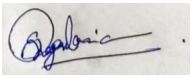
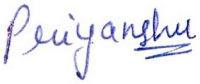
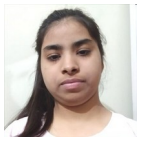
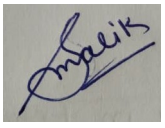
IQAC
Sri Venkateswara College
University of Delhi
Benito Juarez Road, Dhaula Kuan, New Delhi
New Delhi -110021

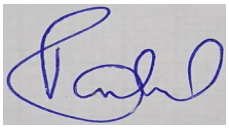
SRIVIPRA PROJECT 2025

Title : Quantum Information and Computation : Algebra and Algorithms

Name of Mentor :	**Ram Lal Awasthi**	**Photo** 
Name of Department :	**Physics**	
Designation :	**Assistant Professor**	

List of students under the SRIVIPRA Project

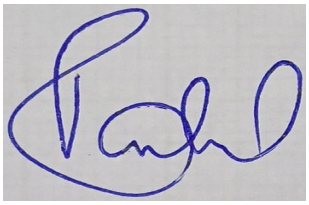
S.No	Photo	Name of the student	Roll number	Course	Signature
1		S. Nagasubramanian	1823009	B.Sc. (H) Physics	
2		Keshav	1823032	B.Sc. (H) Physics	
3		Priyanshu Sharma	1823062	B.Sc. (H) Physics	
4		Komal Lamba	1823901	B.Sc. (H) Physics	
5		Shiksha	1823902	B.Sc. (H) Physics	



Signature of Mentor

Certificate of Originality

This is to certify that the aforementioned students from Sri Venkateswara College have participated in the summer project SVP-2531 titled "**Quantum Information and Computation : Algebra and Algorithms**".The participants have carried out the research project work under my guidance and supervision from 1st July, 2024 to 30th September 2025. The work carried out is original and carried out in an offline mode.



Signature of Mentor

Acknowledgements

We would like to thank the Principal SVC (Prof. V. Ravi), the SRI-VIPRA 2025-26 Team, IQAC – SVC and TIC, Department of Physics (Dr. Narender Kumar) for all the support and the opportunity.

We would also like to express our gratitude towards Dr. Ram Lal Awasthi for the supervision of this project.

Keshav,
Komal Lamba,
S. Nagasubramanian,
Priyanshu Sharma,
Shiksha.

SVP-2531 : Quantum Information and Computation : Algebra and Algorithms

A SRI-VIPRA 2025 project report

by

S. Nagasubramanian (1823009),

Keshav (1823032),

Priyanshu Sharma (1823062),

Komal Lamba (1823601),

and

Shiksha (1823602)

in supervision of

Dr. Ram Lal Awasthi

Submitted to Sri Venkateswara College.

Department of Physics

Sri Venkateswara College

University of Delhi

Dhaula Kuan, New Delhi, Delhi 110021

Certificate

This is to certify that the project report titled "**SVP-2531 : Quantum Information and Computation : Algebra and Algorithms**" submitted to the **Sri Venkateswara College** as the part of Sri-Vipra 2025, is the record of bona fide research work done by S. Nagasubramanian (1823009), Keshav (1823032), Priyanshu Sharma (1823062), Komal Lamba (1823601) and Shiksha (1823602) in my supervision.

Dr. Ram Lal Awasthi

(Supervisor)

Assistant Professor

Sri Venkateswara College

University of Delhi

New Delhi-110021

ABSTRACT

The field of quantum information and computation has rapidly emerged as one of the most exciting and potentially life changing fields in the past few decades. We know that the role of any computer, theoretical or real, is to compute or automatically carry out sequences of arithmetic and logical operations based on their programming. While classical computers have a time advantage over normal analytic solutions, and they have advantage of repetitivity and scalability, they are still limited in comparision to quantum computers. The quantum mechanics principles which make the realization of quantum computers possible and promise to be exponentially fast are : Superposition and Entanglement. Other properties like Quantum Teleportation and No cloning theorem are other useful quantum principles operational in working of quantum computers as well as in information sharing and communication. Quantum computers have advantage over classical computers for their use of supeposition of states and entanglement. The computation capacity scales exponentially with scaling of hardware.

In this report we will be presenting our basic understanding of the Quantum mechanics principle and useful basic Quantum Algorithms. We will also brief about realization of quantum computers.

Key Words: Qubit, Entanglement, Superposition, Teleportation, Quantum Information, Quantum Computation, Quantum Gates, Douesch-Josza's Algorithm, Simon's Algorithm, Gorver's Algorithm, Shor's Algorithm.

Contents

List of Figures

List of Tables

CHAPTER 1

Superposition, Qubit, Entanglement and Representation

1.1 Introduction

Quantum Mechanics since the inception of the idea in 1920's has made a long and proud journey of explaining not only the the dynamics of subatomic systems but also the working of large scale structure. The underlying mathematics of Quantum Mechanics is simple Linear Algebra, and underlying laws of nature are superposition of states. The entanglement of states is another principle which makes scaling of quantum computing possible.

1.2 Superposition Principle

Superposition principle is nothing but an equivalent concept to vector. Suppose you have a general vector $\vec{A}$ in 3 real dimension (Euclidean space), you would very much like to present it in three component notation, say (A_1, A_2, A_3). These numbers present in this tuple are the projection of the vector $\vec{A}$, 'in a given basis'. We are often comfortable to pick the basis vector of cartesian type. Or, in matrix notation to choose the basis vectors

like

$$\hat{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \hat{y} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad \hat{x} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{1.1}$$

Then this vector $\vec{A}$ can be written as

$$\vec{A} = A_x \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + A_y \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + A_z \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{1.2}$$

where having known the basis are cartesian, we have changed the index from $1, 2, 3$ to a, y, z. If this vector is evolving with time either coefficients could be time dependent (Schrodinger's representation) or the basis could be time dependent (Heisenberg's representation).

Now, if we generalize the space from real space to complex space and transform the above representation to Dirac's Bra-(C)-Ket notation, we would write a state $|\psi\rangle$ as follows:

$$|\psi\rangle = \sum_i |i\rangle \langle i| \psi\rangle \tag{1.3}$$

This i, representing the basis count may be discrete finite, infinite and continuous index. A slight change in representation and definition changes would do the job for continues case. In this report we will hardly need infinite and continuum space. So, for a qubit - having two orthonormal states - the basis may be $|0\rangle$ and $|1\rangle$ in which case an arbitrary state is written as

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle, \quad \text{s.t.} \quad |\alpha|^2 + |\beta|^2 = 1. \tag{1.4}$$

The condition on complex numbers α and β is to ensure that not only basis states are orthonormal but also any superposition state is also orthonormal. These basis states can be any pair of the infinitely many possible orthonormal choices, out of which, a set of two is written below.

$$\left[|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right]; \quad \left[|0\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}, |1\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right] \tag{1.5}$$

In quantum mechanics $|\psi\rangle \equiv e^{i\theta} |\psi\rangle$. This overall phase on the whole state is redundent and never appeard in any physical interpretation. So, we can always fix (throwout)

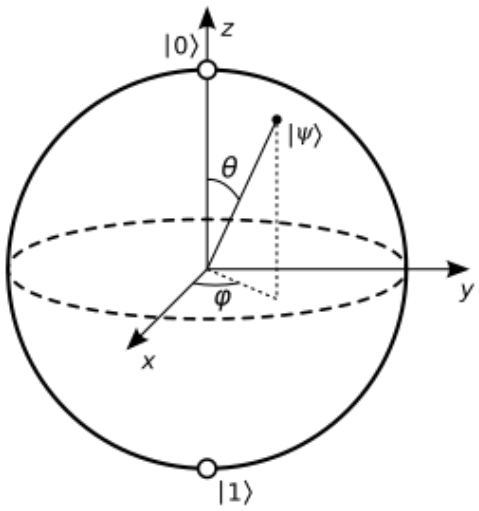


Figure 1.1: A state is represented by a vector from origin to a point on the sphere and written as Eqn.(1.7).

it. This way we get two conditions on total four independent parameters. Thus, only two parameters are free for them to represent a vector pointing to the surface of a unit sphere, called Bloch sphere.

1.3 Bloch Sphere

The Bloch Sphere (Felix Bloch) provides a beautiful geometric representation for the state of a single qubit. It employs the spherical coordinate system, wherein each point on the sphere corresponds to a unique state of the qubit. A complex number $z = x + iy \equiv r\, e^{i\theta}$ has two variables (x, y) or (r, θ). If we impose unit magnitude complex number we have $|z|^2 = x^2 + y^2 = 1 = r$, leaving only one variable. So, a quantum state $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ having two complex variables α and β or four real variables, if is imposed to have probabilistic interpretation $\langle\psi\,|\psi\rangle = 1$, is left with three variable. Out of which one variable can be taken out as overall phase of the states

$$\begin{aligned} |\psi\rangle &= \alpha\,|0\rangle + \beta\,|1\rangle \\ &= a\,e^{ib}\,|0\rangle + c\,e^{id}\,|1\rangle \\ &= e^{ib}\left(a\,|0\rangle + c\,e^{i(d-b)}\right) \end{aligned} \tag{1.6}$$

this, or any such, overall phase e^{ib} can be ignored and we are left with three variables a, c and $d-b$ constrained by $|a|^2 + |c\, e^{i(d-b)}|^2 = |a|^2 + |c|^2 = 1$. We can thus assign $a = \cos\theta$ and $c = \sin\theta$. Redefining $d - b = \phi$, we can write

$$e^{i\zeta}\,|\psi\rangle \equiv |\psi\rangle = \cos\theta\,|0\rangle + \sin\theta e^{i\phi}\,|1\rangle \tag{1.7}$$

Note that if angle and phase are $\pi-\theta$ and $\pi+\phi$ then $\cos(\pi-\theta) = -\cos\theta$ and $\sin(\pi-\theta) = \sin\theta$, but $e^{i(\pi+\phi)} = -e^{i\phi}$. Therefore, $|\psi\rangle_{\theta,\phi} = -\,|\psi\rangle_{\pi-\theta,\pi+\phi}$. An overall negative sign is also an overall phase on the state so the two states are indifferent. Thus, we effectively have only two variables $\theta \in [0, \pi/2]$ and $\phi \in [0, 2\pi]$. But, then this definition seems failing in geometrical representation on Bloch sphere that one never reaches state $|1\rangle$. The correct representation on a boch sphere then should be

$$|\psi\rangle = \cos\frac{\theta}{2}\,|0\rangle + e^{i\phi}\sin\frac{\theta}{2}\,|1\rangle\,, \qquad \text{s.t.}\;\; \theta \in [0,\pi]\;\&\;\phi \in [0, 2\pi] \tag{1.8}$$

The various superposition states and quantum gate operations can hence be visualised as rotations on the sphere. Note that $\theta = 0$ belongs to pure $|0\rangle$ state and $\theta = 180$ to pure $|1\rangle$ state for arbitrary value of ϕ. So, the states $|0\rangle$ and $|1\rangle$ are orthogonal to each other, shown inverted on bloch sphere.

1.4 Entanglement

Single qubit though has infinite possibilities. But, it can not encode more than one logical variable. It can contain only one unit of quantum information at a time. Storing or processing of more than one variable is not possible with single qubit. A non-trivial computation required to combine and correlate several pieces of informations and apply logical operations on them. Therefore, one needs more than one qubits to perform a computation.

Let us start with two qubit system. Both of them live in their own $SU(2)$ space with states defined in their own basis. How we would write a state representing both of them? Best solution is perhaps a tensor product of the states. Let us have two qubits $|\psi\rangle$ and $|\phi\rangle$ as following:

$$|\psi\rangle = \alpha_1\,|a_1\rangle + \beta_1\,|b_1\rangle\,, \quad |\phi\rangle = \alpha_2\,|a_2\rangle + \beta_2\,|b_2\rangle\,. \tag{1.9}$$

where $\{|a_i\rangle, |b_i\rangle\}$ for $i = 1, 2$ are the basis sets of vector in $SU(2)_i$ spaces. So, the composite state is:

$$\begin{aligned} |\psi\rangle \otimes |\phi\rangle &= \alpha_1\alpha_2 |a_1\rangle \otimes |a_2\rangle + \alpha_1\beta_2 |a_1\rangle \otimes |b_2\rangle \\ &\quad + \beta_1\alpha_2 |b_1\rangle \otimes |a_2\rangle + \beta_1\beta_2 |b_1\rangle \otimes |b_2\rangle \end{aligned} \tag{1.10}$$

Having understood that the two qubits live in two different spaces. Also, states belonging to same space can't be multiplied, we can simplify our presentation of two qubit system by dropping $\otimes$ and writing $|\psi\rangle |\phi\rangle$ or even combining the two to kets to one and writing them as

$$|\psi\, \phi\rangle = \alpha_1\alpha_2 |a_1\, a_2\rangle + \alpha_1\beta_2 |a_1\, b_2\rangle + \beta_1\alpha_2 |b_1\, a_2\rangle + \beta_1\beta_2 |b_1\, b_2\rangle \tag{1.11}$$

where $|\psi\, \phi\rangle \neq |\phi\, \psi\rangle$. One can define an orthonormal basis for the system as

$$\{|a_1\, a_2\rangle, |a_1\, b_2\rangle, |b_1\, a_2\rangle, |b_1\, b_2\rangle\} \tag{1.12}$$

Where the first element of each basis vector corresponds to qubit one and second corresponds to qubit two. In this case the vector space for both objects is $SU(2)$. In general, both spaces could be different from $SU(2)$ and distinct from each other. Call $\alpha_1\alpha_2 = r_1$, $\alpha_1\beta_2 = r_2$, $\beta_1\alpha_2 = r_3$ and $\beta_1\beta_2 = r_4$. Remember that

$$|r_1|^2 + |r_2|^2 + |r_3|^2 + |r_4|^2 = 1 \tag{1.13}$$

Also, most importantly, that $\boxed{r_1 r_4 = r_2 r_3}$. As long as this condition is satisfied, the two states as not entanglesd. That means that any measurement or disturbance to one state does not affect the information contained by other.

But, if $r_1 r_4 \neq r_2 r_3$, the two states are entangled. That means that the measurement of one qubit will affect the outcome of the other. Let us see it through one example: Let the composite state is

$$|\psi\, \phi\rangle = -\frac{\sqrt{3}}{4} |0\, 0\rangle + \frac{1}{4} |0\, 1\rangle - \frac{3}{4} |1\, 0\rangle + \frac{\sqrt{3}}{4} |1\, 1\rangle \tag{1.14}$$

Note that $\langle\psi\phi|\psi\phi\rangle = 1$ and $r_1\, r_4 = r_2\, r_3$ is satisfied. So it does not seem like an entangled system. Let us check: Suppose first qubit is being measured.

$$|\psi\, \phi\rangle = |0\rangle \left(-\frac{\sqrt{3}}{4} |0\rangle + \frac{1}{4} |1\rangle \right) + |1\rangle \left(-\frac{3}{4} |0\rangle + \frac{\sqrt{3}}{4} |1\rangle \right) \tag{1.15}$$

For making terms inside brackets a unit vector we require to pull out something from it. Thus getting

$$\begin{aligned} |\psi\,\phi\rangle &= \frac{1}{2}|0\rangle\left(-\frac{\sqrt{3}}{2}|0\rangle+\frac{1}{2}|1\rangle\right)+\frac{\sqrt{3}}{2}|1\rangle\left(-\frac{3}{2}|0\rangle+\frac{1}{2}|1\rangle\right) \\ &= \left(\frac{1}{2}|0\rangle+\frac{\sqrt{3}}{2}|1\rangle\right)\left(-\frac{\sqrt{3}}{2}|0\rangle+\frac{1}{2}|1\rangle\right) \end{aligned} \tag{1.16}$$

means the state we started with was not entangled. We have tensor product of two qubits representing two different independent systems/properties. Measurement of one qubit has no effect on other. On the other hand suppose we have a state

$$|\psi\,\phi\rangle = -\frac{\sqrt{3}}{4}|0\,0\rangle+\frac{1}{\sqrt{2}}|0\,1\rangle-\frac{1}{\sqrt{8}}|1\,0\rangle+\frac{\sqrt{3}}{4}|1\,1\rangle \tag{1.17}$$

and we follow the above procedure in an attempt to see if two qubits get segeragated or not. Note, though, that $\frac{-\sqrt{3}}{4}\times\frac{\sqrt{3}}{4}\neq\frac{1}{\sqrt{2}}\times\frac{-1}{\sqrt{8}}$.

$$\begin{aligned} |\psi\,\phi\rangle &= |0\rangle\left(-\frac{\sqrt{3}}{4}|0\rangle+\frac{1}{\sqrt{2}}|1\rangle\right)+|1\rangle\left(-\frac{1}{\sqrt{8}}|0\rangle+\frac{\sqrt{3}}{4}|1\rangle\right) \\ &= \frac{\sqrt{11}}{4}|0\rangle\left(-\frac{\sqrt{3}}{\sqrt{11}}|0\rangle+\frac{2\sqrt{2}}{\sqrt{11}}|1\rangle\right)+\frac{\sqrt{5}}{4}|1\rangle\left(-\frac{2}{\sqrt{10}}|0\rangle+\frac{\sqrt{3}}{\sqrt{5}}|1\rangle\right) \end{aligned} \tag{1.18}$$

For this state the terms inside the brackets are different. When first qubit is measured to be $|0\rangle$, the combined system jumps to unentangled state $|0\rangle\left(-\frac{\sqrt{3}}{\sqrt{11}}|0\rangle+\frac{2\sqrt{2}}{\sqrt{11}}|1\rangle\right)$ and second qubit is measured to be $|0\rangle$ or $|1\rangle$ with different probability from the case when first qubit is measured to be $|1\rangle$. That means outcome of first measurement affects the second measurement. The outcome is no different in the case when second qubit is measured before the first. The outcome of second qubit will impact the outcome of first qubit.

In summary, If the joint wavefunction of two or more systems interact in such a way that their join wavefunction can not be described seperately i.e. $|\psi\,\phi\rangle\neq|\psi\rangle\otimes|\phi\rangle$, they are entangled.

Can Entanglement be used for instantaneous communication? Answer is NO! The reason is that no matter a measurement of first qubit is done earlier or later than second qubit the measurement outcomes ($|0\rangle$ or $|1\rangle$) will occur with same probability ratio. The observer of measurement of first qubit will never know whether measurement of second qubit has happened or not. So, no information is being shared.

CHAPTER 2

Classical and Quantum Gates and their matrix representation

2.1 Introduction

While the modern classical computers work in gate based model. Quantum computers have beed developed both in Gate based quantum circuit and other methods of computation for example : Quantum Annealing (designed specifically for optimization problems), Adiabatic Quantum Computing (Adiabatic evolution of Hamiltonian to get ground state of system), Measurement-based quantum computing, Quantum walk, Topological quantum computing are the major choices popular among developers of quantum computers. Since, both classical and quantum computers can fundamentally work using gate based principle. We have mainly adopted to study the gate bases computing in this report. The type of gate and the underlying principles are very different in quantum computers compaired to classical computers. Gates are the basic building blocks for computation, performing logical operations.

Classical computers work with binary numbers following Boolean algebra for logical/ mathematical operations. Information is stored in the form of Bit (True/False, High/Low, 0/1 etc.), being the smallest unit of information. In digital computers it exists and stored in the form of electrical charges. Gate are electronic circuits which perform Binary logical operations on one or more binary inputs and gives one binary output. We will discuss

the Gate operations using operations in Linear Algebra for for Classical and Quantum Gates.

In Quantum computing, Qubit is the basic unit of information, a quantum state. A qubit can be much more than $|0\rangle$ or $|1\rangle$ state. Having two linearly independent states as basis, there may exist infinitely many linear combination of the two states $|0\rangle$ and $|1\rangle$. We denote it as $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ with constraint $|\alpha|^2 + |\beta|^2 = 1$, for probabilistic interpretation of quantum mechanics. This phenomena is called superposition of states. In general α and β are complex in nature. Or, we can say that, the state $|\psi\rangle$ lives in a two dimensional unitary space $SU(2)$. Thus the gate operations in quantum computing are in principle unitary operations.

2.2 Classical Gate

We will adopt Dirac Bra-Ket representation to denote classical bit 0 and 1. Let us represent these states in matrix representation as:

$$|0\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \& \quad |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{2.1}$$

NOT gate being the simplest one is easily understandable in both Truth table and Matrix representation.Depicted below :

I	O
0	1
1	0

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{2.2}$$

which is equivalent to saying $\hat{\mathrm{N}}\,|0\rangle = |1\rangle$ and $\hat{\mathrm{N}}\,|1\rangle = |0\rangle$. The operator $\hat{N}$ is the matrix $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Note that the representation for Not operation in Eqn.(2.2) is valid in the basis of Eqn.(2.1). If we change the basis to

$$|0\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad \& \quad |1\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \tag{2.3}$$

Then the Not operator would be Simply the matrix $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ so that

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \tag{2.4}$$

In the same lines of arguments we can write the matrix algebra for other classical gates. One difference though is that in other gates we have more than one input. With two inputs we have four possible cases as shown in truth table. Thus states for these four possibilities can be represented as

$$|00\rangle \equiv |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \equiv \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad |01\rangle \equiv |0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \tag{2.5}$$

$$|10\rangle \equiv |1\rangle \otimes |0\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \quad |11\rangle \equiv |1\rangle \otimes |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{2.6}$$

Thus the Truth table and Matrix operation for **And** gate look like:

I_1	I_2	O
0	0	0
0	1	0
1	0	0
1	1	1

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{2.7}$$

That is equivalent to saying $\hat{\mathrm{A}}\,|0,0\rangle = |0\rangle$, $\hat{\mathrm{A}}\,|0,1\rangle = |0\rangle$, $\hat{\mathrm{A}}\,|1,0\rangle = |0\rangle$ and $\hat{\mathrm{A}}\,|1,1\rangle = |1\rangle$.

and the Truth table and Matrix operation for **OR** gate look like:

I_1	I_2	O
0	0	0
0	1	1
1	0	1
1	1	1

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{2.8}$$

That is equivalent to saying $\hat{\mathrm{A}}\,|0,0\rangle = |0\rangle$, $\hat{\mathrm{A}}\,|0,1\rangle = |1\rangle$, $\hat{\mathrm{A}}\,|1,0\rangle = |1\rangle$ and $\hat{\mathrm{A}}\,|1,1\rangle = |1\rangle$. This way we can also describe remaining of common classical gate operations in matrix form.

2.3 Quantum gates

In the Sec.2.2 we discussed about classical gates but we used Dirac's Bra-Ket notation to represent the classical states. Also, we used outer or tensor product of the states to write a composite basis for two bit input operation. This much of knowledge is used in quantum gates as such. In fact, if we don't use the the superposition of the states and apply, say, **Not** gate on $|0\rangle$ or $|1\rangle$ the operation will exactly mimic the behavior of classical computer. Thus, in principle, quantum computer can calculate anything which classical computer can. While no cloning theorem may seem like a problem but we can always design a classical algorithms where we don't need to clone anything. There is a fundamental difference between bit and qubit. As we stated earlier, the qubit can live in superposition state $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$, which is not possible in case of classical bit.

We will also call the two qubit basis state as $|00\rangle = |0\rangle$, $|01\rangle = |1\rangle$, $|10\rangle = |2\rangle$ and $|11\rangle = |3\rangle$.

2.3.1 Three qubit basis states:

$$\begin{cases} |000\rangle \equiv |0\rangle, \quad |001\rangle \equiv |1\rangle, \quad |010\rangle \equiv |2\rangle, \quad |011\rangle \equiv |3\rangle, \\ |100\rangle \equiv |4\rangle, \quad |101\rangle \equiv |5\rangle, \quad |110\rangle \equiv |6\rangle, \quad |111\rangle \equiv |7\rangle \end{cases}$$

$$|0\rangle \otimes |2^3 - 1\rangle, \qquad I_8$$

2.3.2 Single Qubit Quantum Gates

A single qubit gate is represented by a **unitary operator** U :

$$UU^\dagger = I, \qquad U^{-1}U^\dagger, \qquad \det U = 1$$

The columns satisfy the 'orthonormality condition' :

$$\langle u_i|u_i\rangle = 1, \qquad \langle u_i|u_j\rangle = 0 \;\; (i \neq j)$$

Any single qubit quantum gate can be expressed as :

$$U = e^{i\alpha} R_{\hat{n}}(\theta) = e^{i\alpha} \exp\left(-i\,\frac{\theta}{2}\,(\vec{\sigma}\cdot\hat{n}\right)$$

$$U = e^{i\alpha}\left[I\cos\frac{\theta}{2} - i(\vec{\sigma}\cdot\hat{n})\sin\frac{\theta}{2}\right]$$

For $\alpha = \frac{\pi}{2}$, $\theta = \pi$:

$$U = e^{i\pi/2}\left[I\cos\frac{\pi}{2} - i(\vec{\sigma}\cdot\hat{n})\sin\frac{\pi}{2}\right] = \vec{\sigma}\cdot\hat{n}$$

$$\vec{\sigma}\cdot\hat{n} = \sigma_x n_x + \sigma_y n_y + \sigma_z n_z$$

The general form :

$$U = \vec{\sigma} \cdot \hat{n} = \sigma_x n_x + \sigma_y n_y + \sigma_z n_z$$

$$\begin{aligned}
\hat{n} = (1,0,0) \quad &\Rightarrow \quad U = \sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \\
\hat{n} = (0,1,0) \quad &\Rightarrow \quad U = \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \\
\hat{n} = (0,0,1) \quad &\Rightarrow \quad U = \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}
\end{aligned} \tag{2.9}$$

These are the Pauli Gates(X, Y, Z).

2.3.2.0.1 HADAMARD GATE

$$\hat{n} = \left(\tfrac{1}{\sqrt{2}}, 0, \tfrac{1}{\sqrt{2}}\right) \quad \Rightarrow \quad U = \tfrac{1}{\sqrt{2}}\sigma_x + \tfrac{1}{\sqrt{2}}\sigma_z = \tfrac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = H$$

$$H^2 = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

If you apply the Hadamard gate twice, you return to the same state.

2.3.3 Pauli X Gate (Quantum NOT gate)

The Pauli - X gate is :

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

It acts as :

$$X = |0\rangle \langle 1| + |1\rangle \langle 0|$$

$$X|0\rangle = (|0\rangle \langle 1| + |1\rangle \langle 0|)|0\rangle$$

$$X|0\rangle = |1\rangle$$

$$X\,|1\rangle = (|0\rangle\,\langle 0| + |1\rangle\,\langle 0|)\,|1\rangle$$

$$X|1\rangle = |0\rangle$$

This corresponds to a rotation by 180° around the x-axis on the Bloch sphere.

Action:

$$X|0\rangle = |1\rangle, \qquad X|1\rangle = |0\rangle$$

Quantum Circuit Representation:

2.3.4 Pauli Y Gate

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

It acts as :

$$Y = -i\,|0\rangle\,\langle 1| + i\,|1\rangle\,\langle 0|$$

$$Y\,|0\rangle = (-i\,|0\rangle\,\langle 1| + i\,|1\rangle\,\langle 0|)\,|0\rangle$$

$$Y\,|0\rangle = i\,|1\rangle$$

$$Y\,|1\rangle = (-i\,|0\rangle\,\langle 1| + i\,|1\rangle\,\langle 0|)\,|1\rangle$$

$$Y\,|1\rangle = -i\,|0\rangle$$

Quantum Circuit Representation:

2.3.5 Pauli Z Gate

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

It acts as :

$$Z = |0\rangle\langle 0| - |1\rangle\langle 1|$$

$$Z|0\rangle = (|0\rangle\langle 0| - |1\rangle\langle 1|)|0\rangle$$

$$Z|0\rangle = |0\rangle$$

$$Z|1\rangle = (|0\rangle\langle 0| - |1\rangle\langle 1|)|1\rangle$$

$$Z|1\rangle = -|1\rangle$$

Quantum Circuit Representation:

2.3.6 Hadamard Gate

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

$$H = \frac{1}{\sqrt{2}}\big[\,|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|\big]$$

$$H|0\rangle = (\frac{1}{\sqrt{2}}\big[\,|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |0\rangle\langle 0|\big])|0\rangle$$

$$H|0\rangle = \frac{1}{\sqrt{2}}\big[\,|0\rangle + |1\rangle\big]$$

$$H|1\rangle = (\frac{1}{\sqrt{2}}\big[\,|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|\big])|1\rangle$$

$$H|1\rangle = \frac{1}{\sqrt{2}}\big[\,|0\rangle - |1\rangle\big]$$

Circuit Representation:

$|0\rangle$ —H— $\frac{1}{\sqrt{2}}\big[\,|0\rangle + |1\rangle\big]$ $\qquad$ $|1\rangle$ —H— $\frac{1}{\sqrt{2}}\big[\,|0\rangle - |1\rangle\big]$

This is also a superposition state

$$H = \frac{1}{\sqrt{2}}\big[X + Z\big]$$

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

$$H\,|0\rangle = \frac{1}{\sqrt{2}}\big[\,|0\rangle + |1\rangle\big], \qquad H\,|1\rangle = \frac{1}{\sqrt{2}}\big[\,|0\rangle - |1\rangle\big]$$

general representation

$$H\,|x\rangle = \frac{1}{\sqrt{2}}\big[\,|0\rangle + (-1)^x\,|x\rangle\big]$$

2.4 Quantum Algorithms

Quantum Algorithms are pseudocode for applying sequence of unitary operation to manipulate qubit(s) through a series of quantum transformations to reach to an outcome which is interpreted for solution of problem. The basic framework of any quantum algorithm is as follows:

1. **Initialization:** Qubits are prepared in a known classical state, e.g. $|0\rangle^{\otimes n}$.
2. **Superposition:** The system is spread into a superposition of many states, using Hadamard gates among others.
3. **Unitary Operations:** This superposition is then acted upon with (several) unitary operations that encode the desired information based on the problem at hand.
4. **Measurement:** Finally the measurement of the qubits is done.

Some of the prominent quantum algorithms include Deutsch's Algorithm and the Deutsch-Jozsa Algorithm, Simon's Algorithm, Grover's Algorithm (finding a needle in a haystack), Shor's Algorithm (prime factorization of integers).

CHAPTER 3

Deutsch's and Deutsch-Jozsa Algorithm

3.1 Introduction

Deutsch's Algorithm is one of the earliest quantum algorithms demonstrating how quantum computation can surpass classical computation. By taking advantage of quantum superposition and interference, which allow for the simultaneous evaluation of multiple inputs and constructive or destructive interference to reveal the nature of the function, it effectively ascertains whether a one-bit function is constant or balanced using a single quantum query. This demonstrates the basic benefit of quantum algorithms over traditional deterministic techniques that necessitate numerous evaluations in terms of lowering computational complexity.

Utilizing basic concepts from quantum mechanics, including superposition, entanglement, and interference, quantum computing performs calculations in a manner distinct from classical computing. Quantum bits, or qubits, can exist in a superposition of both states simultaneously, unlike classical bits, which only represent 0 or 1. This allows quantum computers to process multiple inputs at once. This benefit is demonstrated by the groundbreaking quantum algorithm known as Deutsch's algorithm, which was put forth by David Deutsch in 1985. With just one quantum query, it resolves the issue of figuring out if a one-bit function is balanced or constant. To solve the same problem, classical deterministic algorithms need to run several queries. Deutsch's algorithm offers an early illustration of quantum computational speedup by utilizing quantum parallelism

and interference, opening the door for more intricate quantum algorithms.

Problem Statement

Consider a function ($f : 0, 1 \to 0, 1$). The problem is to determine whether (f) is *constant* (i.e., (f(0) = f(1))) or *balanced* i.e., ($f(0) \neq f(1)$). Classically, in the worst case, two evaluations are required to identify the type of (f).

3.2 Deutsch's Algorithm

Deutsch's algorithm provides a solution using a quantum computer with just one evaluation query of (f).

3.2.1 Algorithm Outline

We represent the input bits as qubits. starting with two qubits initialized in the state ($|0\rangle|1\rangle$):

1. Apply a Hadamard gate (H) to each qubit to create superposition:

$$|\psi\rangle_0 \equiv |0\rangle\,|1\rangle \xrightarrow{H\otimes H} \frac{1}{2}\Big[\,|0\rangle + |1\rangle\,\Big]\Big[\,|0\rangle - |1\rangle\,\Big] \equiv |\psi\rangle_1 \tag{3.1}$$

2. Apply the oracle U_f representing the function f, which performs:

$$U_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle, \tag{3.2}$$

   where $\oplus$ is addition modulo 2.

   The resulting state becomes:

$$|\psi\rangle_2 = \frac{1}{2}\Big[(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\Big]\Big(|0\rangle - |1\rangle\Big). \tag{3.3}$$

3. Apply a Hadamard gate H to the first qubit:

$$\begin{aligned} |\psi\rangle_3 &= \frac{1}{\sqrt{2}}\left[(-1)^{f(0)}\frac{|0\rangle+|1\rangle}{\sqrt{2}}+(-1)^{f(1)}\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]\Big(|0\rangle-|1\rangle\Big) \\ &= \frac{1}{2}\Big(\left[(-1)^{f(0)}+(-1)^{f(1)}\right]|0\rangle+\left[(-1)^{f(0)}-(-1)^{f(1)}\right]|1\rangle\Big)\Big(|0\rangle-|1\rangle\Big). \end{aligned} \tag{3.4}$$

4. Measure the first qubit:

   - If the measurement is $|0\rangle$, then f is constant.
   - If the measurement is $|1\rangle$, then f is balanced.

3.2.2 Pseudocode of the Algorithm

Algorithm 1 Deutsch's Algorithm

Initialize qubits in state $(|0\rangle|1\rangle)$

Apply Hadamard gate on both qubits

Apply oracle (U_f)

Apply Hadamard gate on the first qubit

Measure the first qubit

if measurement result is 0 **then**

    return (f) is constant

else

    return (f) is balanced

end if

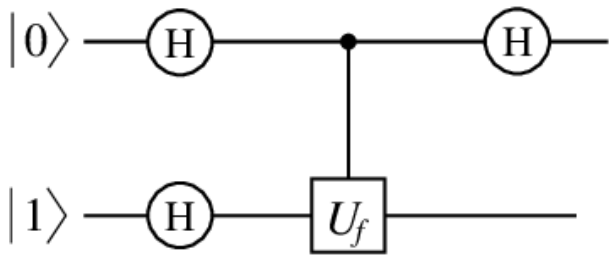


Figure 3.1: This is the circuit diagram of deutsche algorithm.

3.2.3 Significance

Since it was the first to show that quantum computing can solve particular problems with fewer queries than traditional deterministic algorithms, Deutsch's algorithm is noteworthy. It demonstrated how quantum computation could surpass classical techniques by evaluating a one-bit function using only one quantum query. This discovery opened the door for the creation of more sophisticated quantum algorithms that more effectively solve progressively difficult problems, such as the Deutsch-Jozsa algorithm, Simon's algorithm, and Shor's algorithm. By specifically taking advantage of basic quantum phenomena like superposition and interference, Deutsch's algorithm achieves a definite quantum advantage by allowing quantum parallelism and constructive or destructive interference to disclose information about the function.

3.3 The Deutsch-Jozsa Algorithm

The Deutsch-Jozsa algorithm is one of the very first examples of quantum algorithms, developed by British physicist David Deutsch - regarded as the father of quantum computing - along with Australian mathematician Richard Jozsa. The algorithm is a modification and generalisation of the Deutsch algorithm developed solely by Deutsch earlier.

The Deutsch's algorithm is a special case of the Deutsch-Jozsa algorithm. Both these algorithms serve the purpose to solve a category of problems known as "the oracle problems". While these algorithms might not be solving practical problems, they succinctly demonstrated the advantage of quantum computers.

The oracle is an entity capable of solving some decision problem or likewise and is visualised through a black box approach wherein the black box is able to solve certain problems in a single operation.

3.3.1 Problem Statement

We consider a function that takes in a n-string qubit (n-bit binary values) as its input and produces a single qubit (a binary 0 or 1) as its output. The function is unknown and within a black box set-up. We need to develop an algorithm to evaluate the nature of the function – whether it is balanced or constant (biased). Hence our algorithm essentially is an oracle, in predicting the nature of the contained function.

The problem originally considers the function to be Boolean, with n = 1 : Deutsch's Algorithm.

For Deutsch-Jozsa algorithm,

$$f : \{0,1\}^n \rightarrow \{0,1\}$$

3.3.2 The Classical Approach

The problem essentially aims to determine the nature of the function, i.e., there are essentially two outcomes among which we wish to decide where the function belongs or what the function is.

One may consider a fair coin to be analogous to the balanced scenario and a biased coin to be analogous to the constant (biased) scenario.

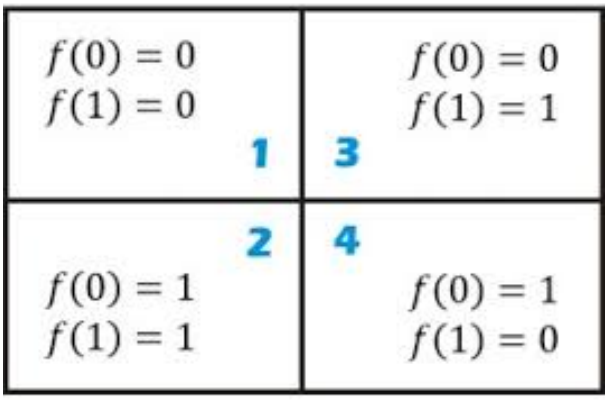


Figure 3.2: Examining the nature of a single bit input function. For more detail.

If the function returns 0 and 1 an equal number of times, it is balanced, while if

returns only one of 0 and 1 throughout, it is said to be constant or biased.

The corresponding classical approach hence mandates that the only method of evaluation of the nature of the function is by taking upto 2 queries in the case of the Deutsch Algorithm (one query each to input both 0 and 1 separately) and likewise, $2^n + 1$ queries for Deutsch-Jozsa algorithm. Hence the classical approach demands a high time and computational cost.

3.3.3 The Quantum Approach

Au contraire, by utilising qubits, the quantum approach employs quantum parallelism - an ability harnessed throught the property of superposition - due to which one can potentially input all the different states at once in the algorithm which is exponentially quicker and is able to solve the problem for both algorithms in a single query.

3.3.4 Step-by-Step Functioning

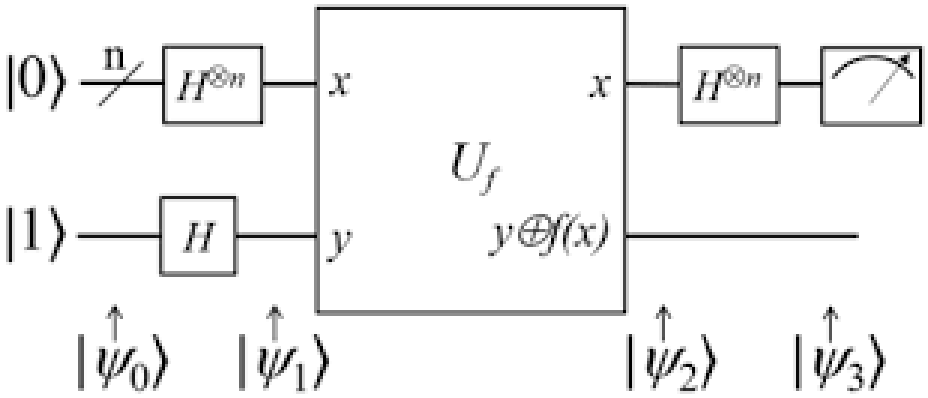


Figure 3.3: Circuit Diagram for the Deutsch-Jozsa Algorithm. A wiki page.

1. The algorithm is initialised by setting the initial input states. $|\psi_0\rangle = |0\rangle^{\otimes n} |1\rangle$

2. These are put into superposition using the Hadamard gates.

$$|\psi_1\rangle = \sum_{x\in\{0,1\}^n} \frac{1}{\sqrt{2^{n+1}}} |x\rangle \left(|0\rangle - |1\rangle \right)$$

3. The unitary transformation is applied to the states at this juncture.

$$|\psi_2\rangle = \sum_{x\in\{0,1\}^n} \frac{1}{\sqrt{2^{n+1}}} (-1)^{f(x)} |x\rangle \left(|0\rangle - |1\rangle \right)$$

4. Hadamard gates are again employed to introduce interference after which we finally attain the final state which is measured.

$$|\psi_3\rangle = \sum_{z\in\{0,1\}^n} \sum_{x\in\{0,1\}^n} \left(\frac{1}{2^n\sqrt{2}} (-1)^{x\cdot z + f(x)} \right) |z\rangle \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

3.3.5 Result and Inferences

Measuring the first n qubits allows us to determine the nature of the function (it's balanced-ness or biasedness) with certainty.

If the function is constant, here we measure $|0\rangle^{\otimes n}$ certainly or with probability one. Else, we have zero probability for measuring the particular state for this function in the balanced state

The Deutsch-Jozsa Algorithm crucially uses the phase kickback mechanism to its advantage. The phase kickback mechanism refers to the fact that the controlled operations end up affecting their controls besides affecting their targets, in a way akin to phasing operations. As the work by Cleve et al., 1997 illustrates, the phase of the target qubit on which the controlled operation is applied is conditioned in such a way that it depends on the state of the control qubit – the phase of target qubit is being kicked back to the first qubit. It requires that the control qubits must be in a superposition, the wavefunction must be an eigenvector of the controlled operator, akin to the scenario of a boat and a river.

In the algorithm, if the function ends up being constant, the oracle flips the sign of the amplitude of the input states making the measurement of the all-zero state highly probable using constructive interference while when it is balanced, no kickback is introduced and there is an equal probability of measuring any of the input states.

The algorithm acts as an excellent initial foray and example into the field of quantum computations and the area of quantum algorithms since it easily demonstrates the quantum advantage. However, due to it working in the highly idealised scenario (that is the existence of only balanced and biased states in the problem and excluding the numerous intermediate possibilities), it has limited direct applications - though it is a part of multiple programs and experiments related to cryptography among others. It is largely hence a proof-of-concept algorithm.

3.3.6 A Foray Into Qiskit-powered coding

Qiskit is an open-source package developed by IBM Research for quantum computing based on Python. It has emerged as a powerful tool to create and execute quantum programs for various purposes. Among its multiple components, Qiskit Runtime is particularly of interest since it is cloud-based and optimal to be used by anyone at any point of time anywhere in the world.

This access to cloud-based quantum computation is made possible by the IBM Quantum Platform - which can be used to run algorithms and experiments, and to explore simulations.

Herewith, are certain snapshots of my first foray into the world of quantum computing using Qiskit through tutorials accessible and available at IBM Quantum Platform.

3.3.7 Conclusion

Deutsch's algorithm is a fundamental illustration of how quantum computation can outperform classical techniques, even though it solves a simplified problem. By evaluating a function with fewer queries than any classical method, it illustrates quantum parallelism.

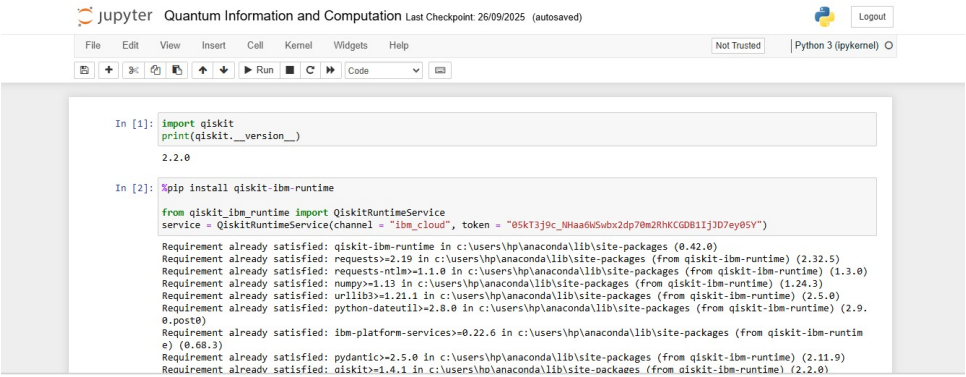


Figure 3.4: Importing and installing qiskit for the first time

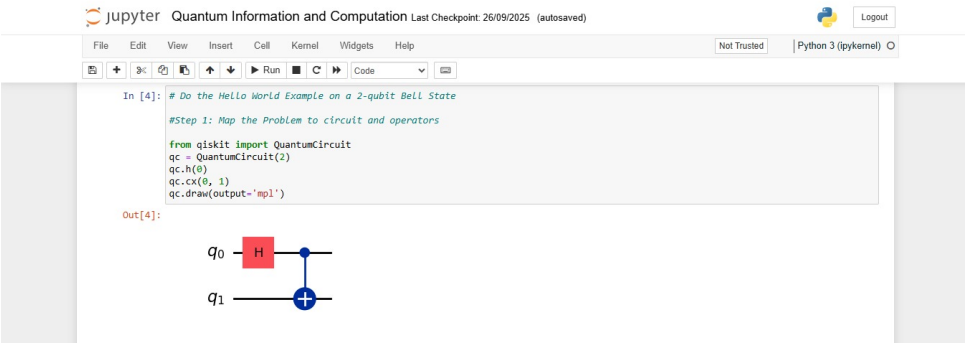


Figure 3.5: A Basic Hello World Diagram for a 2-qubit bell state using QuantumCircuit

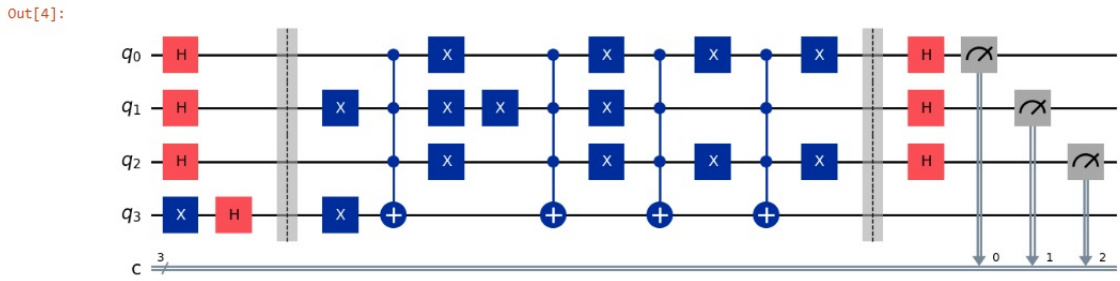


Figure 3.6: Circuit Diagram for Deutsch-Jozsa Algorithm realised using Qiskit via Python Code

It introduced important ideas like superposition and interference, which are essential to more complex quantum algorithms, even though they were not immediately useful. It is important because it shows that certain problems can be solved more effectively by quantum systems, opening the door for the creation of strong algorithms like Shor's and Grover's. As a result, Deutsch's algorithm represents a significant advancement in the theory and applications of quantum computing.

CHAPTER 4

Simon's Periodicity Algorithm

Introduction

- This is the first quantum algorithm proposed by Daniel Simon in 1994.
- This is the first proof that quantum computers can solve same problem exponentially faster than classical computers
- It uses superposition and interference to find a hidden string.

Definition

Given a black-box(oracle) f which maps n-bit inputs to output but we don't know its inside wiring:

$$f : \{0,1\}^n \rightarrow \{0,1\}^n$$

There is a hidden n-bit string $c = c_0, c_1 \cdots c_{n-1}$ such that for all strings $x, y \in \{0,1\}^n$

$$f(x) = f(y) \quad \text{iff} \quad x = y \oplus c$$

Here $\oplus$ denotes bitwise XOR in binary in another words the value of function repeat themselves in some pattern and the pattern is determined by c which is called as period of function . That's why we say the function has a periodicity shifting input by c does

not change the output.

Properties

- If $c = 00\ldots0$,then f is one -to-one function(each input has unique output)
- If c$\neq$ 0, then f is two-to-one function(every output comes from exactly two inputs that differ by c or we can say that for two inputs produce one same output

Goal: Find hidden string c .

Example.1

Let n=3,consider c=101.Then we are going to have the following requirements on f.

x	$x \oplus c$	$f(x) = f(y)$
000	101	$f(000) = f(101)$
001	100	$f(001) = f(100)$
010	111	$f(010) = f(111)$
011	110	$f(011) = f(110)$
100	001	$f(100) = f(001)$
101	000	$f(101) = f(000)$
110	011	$f(110) = f(011)$
111	010	$f(111) = f(010)$

- (000,101)
- (001,100)
- (010,111)
- (111,110)

Those are the pairs with same output because that's exactly how Simon's function is defined. c=101 meaning for any input x there exists another input y such their XOR equals c and f(x)=f(y)

Classical computer vs Quantum computer

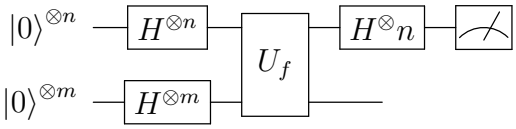
- In classical computers we give different inputs to function f and observe corresponding inputs we have two inputs that have same output then we stop, to find c,one would try all possible input pairs and check outputs.This takes $0(2^{n/2})$ queries.

  In Example 1, n = 3, so it has $2^3 = 8$ inputs, since every output is shared between 2 inputs there are $2^n - 1$ =4 unique outputs.

  It requires checking up to half of the pairs about 4 to 8 queries or requires 7 comparisons in the the worst case.

- In quantum computers, Simon's algorithm finds c using parallelism, checking all inputs at a once. So, For n=3 it find c in about n=3 queries It uses interference pattern and measurement to deduce c with only $0(n)$ queries.

Quantum Circuit Representation



The goal is to determine C.

Initial state:$|0\rangle$

We start with two n-qubit registers in the state: $|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n}$

step 1:Hadamard on the first register

Apply $H^{\otimes n}$ to the first register to create a uniform superposition:

$|\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in 0,1^n} |x\rangle |0\rangle^{\otimes n}$

step 2: oracle U_f

Apply the oracle: $U_f |x\rangle |0\rangle = |f(x)\rangle$. This yields: $|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in 0,1^n} |x\rangle |f(x)\rangle$.

step 3: Measure the second register

Measuring the second register collapses the state to uniform superposition of two inputs differing by c:

$$|\psi_3\rangle = \frac{1}{\sqrt{2}} (|x\rangle , + |x \oplus c\rangle) |f(x)\rangle .$$

step 4: Hadamard on first register again

Apply $H^{\otimes n}$ to the first register: $H^{\otimes n} |x\rangle = \frac{1}{\sqrt{2^n}} \sum_z (-1)^{x \cdot} |z\rangle$, where $x \cdot z$ is the bitwise dot product modulo 2.

So we obtain: $|\psi_4\rangle = \frac{1}{\sqrt{2^n n-1}} \sum_z \left[(-1)^{x \cdot z} + (-1)^{(x \oplus c) \cdot z}\right] |z\rangle |f(x)\rangle$.

Here z=corresponds to the state of top qubits after the final $H^{\otimes n}$ applied.It will give information about hidden period c,because only these z satisfying z.c=0

- = indicates that the amplitude of the state is negative and phase gives the dot product condition z.c=0

z ranges over all n bit strings.

The amplitude is nonzero only for those z satisfying: $c \cdot z = 0 \pmod 2$.

Measuring the first register gives a string z orthogonal to c (mod 2). After collecting $n-1$ such equations, we can solve for c.

For each input x and for each z,we are assured by the oracle who gave us the item f that the set $\{z, f(x)\}$ is same set as $\{z, f(x \oplus c)\}$. The coefficient for this ket is then

$\frac{(-1)^{\langle z,x\rangle}+(-1)^{\langle z,x\oplus c\rangle}}{2}$

We saw $\langle -,-\rangle$ is an inner product and from

$\langle x, y \oplus' c\rangle = \langle x, y\rangle \oplus \langle x, y'\rangle$

Hence $\langle z, x \oplus c\rangle = \langle z, x\rangle \oplus \langle z, c\rangle$ So $\frac{(-1)^{\langle z,x\rangle z,x+c\rangle}}{2} = \frac{(-1)^{\langle z,x\rangle}+(-1)^{\langle z,c\rangle}}{2}$ (x)

Now we have two cases:

Case 1: When $\langle z, c\rangle = 1$

That means inner product (dot product mod 2) of z and c is odd,equal to 1.

so $\frac{(-1)^{\langle z,x\rangle}+(-1)^{\langle z,x\rangle}(-1)}{2} = \frac{(-1)^{\langle z,x\rangle}-(-1)^{\langle z,x\rangle}}{2} = 0$ So numerator=0 $\Rightarrow$ denominator = 0. Thus this outcome z can never appear (forbidden).

Case 2: When $\langle z, c\rangle = 0$

Inner product is even (0 mod 2). $(-1)^{\langle z,c\rangle} = (-1)^0 = 1$

$\frac{(-1)^{\langle z,x\rangle}(1)}{2} = \frac{2(-1)^{\langle z,x\rangle}}{2} = (-1)^{\langle z,x\rangle}$ So amplitude is ± 1. Hence z can appear and it is in allowed region.

CHAPTER 5

Grover's Search Algorithm

5.1 Introduction

The advent of quantum computing represents a paradigm shift in the way we process information. Classical computers operate on bits, which can exist only in states 0 or 1. In contrast, quantum computers make use of *qubits*, which, due to the principles of superposition and entanglement, can exist in linear combinations of these states. This fundamental difference allows quantum computers to perform certain tasks more efficiently than their classical counterparts.

The idea of harnessing quantum mechanics for computation was first formalized in the 1980s by pioneers such as Richard Feynman and David Deutsch, who demonstrated that quantum systems could simulate physical processes exponentially faster than classical machines. This insight led to the birth of quantum computation as a research field, motivating the development of quantum algorithms that could take advantage of quantum parallelism.

Among the earliest and most influential quantum algorithms are Shor's algorithm and Grover's algorithm. Shor's algorithm, discovered in 1994, solves integer factorization in polynomial time, posing a direct challenge to widely used cryptographic protocols. Grover's algorithm, introduced by Lov K. Grover in 1996, addresses the problem of searching an unsorted database. While classical search requires, on average, $O(N)$ queries

to locate a desired element in a list of N elements, Grover's algorithm achieves this task in only $O(\sqrt{N})$ steps. This quadratic speedup, although less dramatic than the exponential speedup of Shor's algorithm, is nevertheless significant, particularly for large-scale search problems.

Grover's algorithm is conceptually elegant. It consists of three main steps: initialization into a uniform superposition of all possible states, application of the oracle that identifies the desired state by flipping its phase, and the diffusion operator (inversion about the mean) that amplifies the probability of the desired state. Repeating these steps a number of times proportional to $\sqrt{N}$ maximizes the likelihood of measuring the desired state.

Beyond unstructured search, Grover's algorithm has wide-ranging applications, as it forms the basis for a family of *amplitude amplification* techniques. These generalizations allow for speedups in diverse areas such as optimization, constraint satisfaction, and graph traversal. Thus, Grover's algorithm is considered one of the foundational algorithms in quantum computing, showcasing how quantum principles can provide advantages even for problems where no exponential speedup is possible.

5.2 Grover's Search Algorithm

5.2.1 Motivation

Searching for a single desired item among m possibilities is like finding a needle in a haystack for large m. Classically, one must check on average $m/2$ items, and in the worst case all m items. Grover's algorithm achieves this task with only $\sqrt{m}$ queries. Although this is not an exponential speedup, it still represents a significant improvement and has many applications, especially in database search.

5.2.2 Problem Setup

We are given a Boolean function

$$f : \{0,1\}^n \to \{0,1\}$$

with the promise that there exists exactly one input x_0 such that

$$f(x) = \begin{cases} 1, & \text{if } x = x_0, \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

Our task is to determine x_0. A classical approach requires up to 2^n evaluations, but Grover's algorithm needs only $2^{n/2}$.

5.2.3 Oracle Representation

The function f is implemented as a unitary operator U_f acting on two registers:

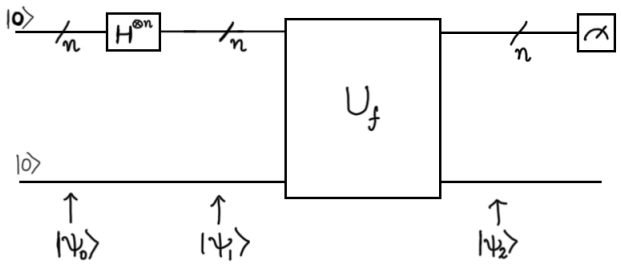
$$U_f \left|x, y\right\rangle = \left|x, y \oplus f(x)\right\rangle .$$

For instance, when $n = 2$ and f is the unique function that "picks out" the string 11, the matrix representation is

$$U_f = \begin{array}{c} \\ 00,0 \\ 00,1 \\ 01,0 \\ 01,1 \\ 10,0 \\ 10,1 \\ 11,0 \\ 11,1 \end{array} \begin{array}{c} \begin{array}{cccccccc} 00,0 & 00,1 & 01,0 & 01,1 & 10,0 & 10,1 & 11,0 & 11,1 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

Initial Superposition

As a first attempt at solving this problem we start with $|x\rangle = |0\rangle^{\otimes n}$, apply a Hadamard transform to place $|x\rangle$ into equal superposition all possible strings, and then query the oracle:



In terms of matrices this becomes,

$$U_f(H^{\otimes n} \otimes I)\,|\mathbf{0}, 0\rangle\,. \tag{2.2}$$

This creates the following states:

$$|\varphi_0\rangle = |\mathbf{0}, 0\rangle\,, \tag{2.3}$$

$$|\varphi_1\rangle = \left[\frac{\sum_{x\in\{0,1\}^n} |\mathbf{x}\rangle}{\sqrt{2^n}}\right] |0\rangle\,, \tag{2.4}$$

$$|\varphi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |\mathbf{x}, f(\mathbf{x})\rangle\,. \tag{2.5}$$

Measuring the top qubits will, with equal probability, give one of the 2^n binary strings. Measuring the bottom qubit will give $|0\rangle$ with probability $\frac{2^n-1}{2^n}$, and $|1\rangle$ with probability $\frac{1}{2^n}$. If you are lucky enough to measure $|1\rangle$ on the bottom qubit, then, because the top and the bottom are entangled, the top qubit will have the correct answer. However, it is improbable that you will be so lucky. We need something more.

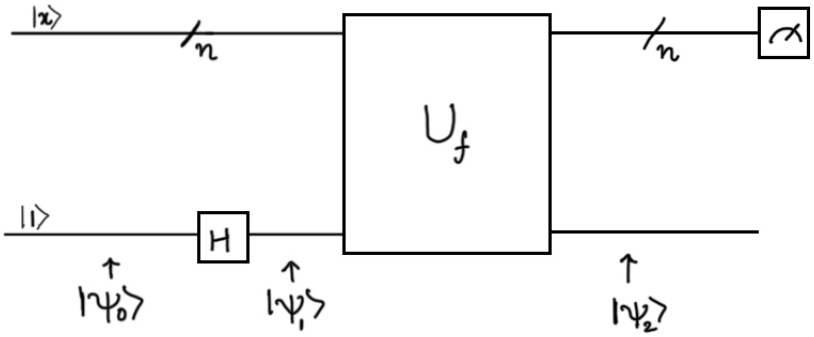
5.2.4 Phase Inversion

To amplify the desired state, Grover's algorithm uses two key ideas. The first is *phase inversion*, which flips the sign of the amplitude of the desired string while leaving all

others unchanged. This is accomplished by preparing the ancilla (bottom) qubit in the superposition

$$\frac{|0\rangle - |1\rangle}{\sqrt{2}},$$

and then evaluate U_f. For an arbitrary $\mathbf{x}$ this looks like :



In terms of matrices this becomes

$$U_f(I_n \otimes H)\,|\mathbf{x}, 1\rangle\,. \tag{2.6}$$

Since U_f and H are unitary operations, it is obvious that phase inversion is a unitary operation. The states are

$$|\varphi_0\rangle = |\mathbf{x}, 1\rangle\,, \tag{2.7}$$

$$|\varphi_1\rangle = |\mathbf{x}\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] = \left[\frac{|\mathbf{x}, 0\rangle - |\mathbf{x}, 1\rangle}{\sqrt{2}}\right], \tag{2.8}$$

$$|\varphi_2\rangle = |\mathbf{x}\rangle \left[\frac{|f(\mathbf{x}) \oplus 0\rangle - |f(\mathbf{x}) \oplus 1\rangle}{\sqrt{2}}\right] = |\mathbf{x}\rangle \left[\frac{|f(\mathbf{x})\rangle - |\overline{f(\mathbf{x})}\rangle}{\sqrt{2}}\right]. \tag{2.9}$$

$$|\varphi_2\rangle = (-1)^{f(\mathbf{x})}\,|\mathbf{x}\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] = \begin{cases} -1\,|\mathbf{x}\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], & \text{if } x = x_0, \\ +1\,|\mathbf{x}\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], & \text{if } x \neq x_0. \end{cases} \tag{2.10}$$

How does this unitary operation act on states? If $|x\rangle$ starts off in an equal superposition of four different states, i.e.,

$$\left[\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}\right]^T,$$

and f chooses the string "11," then after performing a phase inversion, the state looks like

$$\left[\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}, -\tfrac{1}{2}\right]^T.$$

Measuring $|x\rangle$ does not give any information: both $\left|\frac{1}{2}\right|^2$ and $\left|-\frac{1}{2}\right|^2$ are equal to $+\frac{1}{4}$. Changing the phase from positive to negative separates the phases, but does not separate them in a way useful to us. We need something else.

5.2.5 Inversion About the Mean (Diffusion Operator)

Phase inversion alone does not increase measurement probability. The second ingredient is *inversion about the mean*, which redistributes amplitudes so that the desired state's amplitude grows at the expense of the others.

If the amplitudes of the system are $\{v\}$ with mean a, then inversion about the mean is defined by

$$v' = -v + 2a. \tag{2.11}$$

For example, let the amplitudes be $\{37, 49, 80, 54, 15\}$. Their average, a $= 47$. Here the first amplitude $v = 37$ is $47 - 37 = 10$ units below the average, so if we apply the inversion about the mean operation, we get $v' = -37+2(47) = 57$ which is 10 units above the average. Similarly applying the inversion about mean for other amplitudes gives us $\{57, 45, 14, 40, 79\}$ as the inverted amplitudes. Note : the average remains the same, i.e., 47.

This can be expressed in matrix form as

$$V' = (-I + 2A)V, \tag{2.12}$$

where A is the averaging matrix

$$A = \frac{1}{2^n}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix}. \tag{2.13}$$

Note : $A^2 = A$

Multiplying any 2^n state vector (i.e. the state vector for n qubits) by A will give a state where each amplitude will be the average of all the amplitudes. Hence, applying $-I+2A$ (i.e. the inversion about mean operation) to a state will give inverted amplitudes about mean. It can be shown that $(-I + 2A)$ is unitary, hence suitable for a quantum algorithm :

$$(-I + 2A)(-I + 2A) = I - 2A - 2A + 4A^2 = I - 4A + 4A = I$$

5.2.6 Combining the two Operations : A Powerful Tool

When considered separately, phase inversion and inversion about the mean are each innocuous operations. However, when combined, they are a very powerful operation that separates the amplitude of the desired state from those of all the other states.

Let us do an example that shows how both these techniques work together. Consider the vector

$$[8, 8, 8, 8, 8, 8, 8, 8]^T. \tag{2.14}$$

We are always going to perform a phase inversion to the second of the eight numbers. There is no difference between the second number and all the other numbers. We start by doing a phase inversion to the second number and get

$$[8, -8, 8, 8, 8, 8, 8, 8]^T. \tag{2.15}$$

The average of these seven numbers is $a = 6$. Calculating the inversion about the mean we get

$$-v + 2a = -8 + (2 \times 6) = 4 \tag{2.16}$$

and

$$-v + 2a = 8 + (2 \times 6) = 20. \tag{2.17}$$

Thus, our eight numbers become

$$[4, 20, 4, 4, 4, 4, 4, 4]^T. \tag{2.18}$$

The difference between the second element and all the others is $20 - 4 = 16$.

Let us do these two operations again to our eight numbers. Another phase inversion on the second element gives us

$$[4, -20, 4, 4, 4, 4, 4, 4]^T. \tag{2.19}$$

The average of these numbers is $a = 1$. Calculating the inversion about the mean we get

$$-v + 2a = -4 + (2 \times 1) = -2 \tag{2.20}$$

and

$$-v + 2a = 20 + (2 \times 1) = 22 \tag{2.21}$$

Hence, our eight numbers become

$$[-2, 22, -2, -2, -2, -2, -2, -2]^T. \tag{2.22}$$

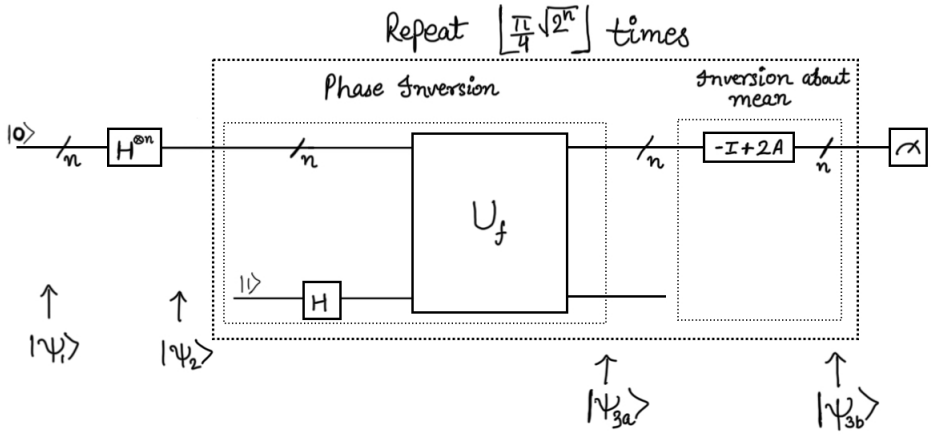
The difference between the second element and all the others is $22 + 2 = 24$. We have further separated the numbers. This was all done with unitary operations.

These operations should be done $\lfloor \frac{\pi}{4}\sqrt{2^n} \rfloor$, i.e., $floor(\frac{\pi}{4}\sqrt{2^n})$ times where N is the size of the function's domain. If you do it more than that, the process will "overcook" the numbers. The proof that $\lfloor \frac{\pi}{4}\sqrt{2^n} \rfloor$ times is needed is beyond this report.

5.2.7 Algorithm :

1. Initialize in $|0\rangle^{\otimes n}$.
2. Apply $H^{\otimes n}$ to form the uniform superposition.
3. Repeat the following "Grover Iteration" $\lfloor \frac{\pi}{4}\sqrt{2^n} \rfloor$ times:
   (a) Apply phase inversion using U_f.
   (b) Apply inversion about the mean.
4. Measure the Qubits.

This can be viewed as :



5.2.8 Example

Let us look at an example of an execution of this algorithm. Let f be a function that picks out the string "011". The states after each step will be

$$|\varphi_1\rangle = \begin{bmatrix} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T . \tag{2.23}$$

$$|\varphi_2\rangle = \begin{bmatrix} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\ \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} \end{bmatrix}^T . \tag{2.24}$$

$$|\varphi_{3a}\rangle = \begin{bmatrix} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\ \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & -\frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} \end{bmatrix}^T . \tag{2.25}$$

The average of these numbers is

$$a = \frac{7 * \frac{1}{\sqrt{8}} - \frac{1}{\sqrt{8}}}{8} = \frac{6}{8\sqrt{8}} = \frac{3}{4\sqrt{8}}. \tag{2.26}$$

Calculating the inversion about the mean we have

$$-v + 2a = -\tfrac{1}{\sqrt{8}} + \left(2 \times \tfrac{3}{4\sqrt{8}}\right) = \tfrac{1}{2\sqrt{8}}, \tag{2.27}$$

and

$$-v + 2a = \tfrac{1}{\sqrt{8}} + \left(2 \times \tfrac{3}{4\sqrt{8}}\right) = \tfrac{5}{2\sqrt{8}}. \tag{2.28}$$

Thus, we have

$$|\varphi_{3b}\rangle = \begin{array}{c} \begin{array}{cccccccc} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \end{array} \\ \left[\begin{array}{cccccccc} \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{5}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} \end{array}\right]^T \end{array}. \tag{2.29}$$

A phase inversion will give us

$$|\varphi_{3a}\rangle = \begin{array}{c} \begin{array}{cccccccc} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \end{array} \\ \left[\begin{array}{cccccccc} \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & -\frac{5}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} \end{array}\right]^T \end{array}. \tag{2.30}$$

The average of these numbers is

$$a = \frac{7 * \frac{1}{2\sqrt{8}} - \frac{5}{2\sqrt{8}}}{8} = \frac{1}{8\sqrt{8}}. \tag{2.31}$$

Calculating for the inversion about the mean we have

$$-v + 2a = -\tfrac{1}{2\sqrt{8}} + \left(2 \times \tfrac{1}{8\sqrt{8}}\right) = -\tfrac{1}{4\sqrt{8}}, \tag{2.32}$$

and

$$-v + 2a = \tfrac{5}{2\sqrt{8}} + \left(2 \times \tfrac{1}{8\sqrt{8}}\right) = \tfrac{11}{4\sqrt{8}}. \tag{2.33}$$

Hence, we have

$$|\varphi_{3b}\rangle = \begin{array}{c} \begin{array}{cccccccc} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \end{array} \\ \left[\begin{array}{cccccccc} -\frac{1}{4\sqrt{8}} & -\frac{1}{4\sqrt{8}} & -\frac{1}{4\sqrt{8}} & \frac{11}{4\sqrt{8}} & -\frac{1}{4\sqrt{8}} & -\frac{1}{4\sqrt{8}} & -\frac{1}{4\sqrt{8}} & -\frac{1}{4\sqrt{8}} \end{array}\right]^T \end{array}. \tag{2.34}$$

For the record,

$$\tfrac{11}{4\sqrt{8}} = 0.97227 \quad \text{and} \quad -\tfrac{1}{4\sqrt{8}} = -0.08839.$$

Squaring the numbers gives us the probability of measuring those numbers. When we measure the state in Step 4, we will most likely get the state

$$|\varphi_4\rangle = \begin{bmatrix} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}^T, \tag{2.35}$$

which is exactly what we wanted.

5.2.9 Time Complexity and Generalization

Classically, searching requires $O(n)$ queries. Grover's algorithm completes the task in $O(\sqrt{n})$ queries — a quadratic speedup.

If there are t desired elements, the algorithm still works, but the number of iterations becomes $O(\sqrt{2^n/t})$.

5.3 Simulation using Qiskit

The following Python code implements Grover's algorithm for $n = 3$ qubits with the oracle marking the state $|111\rangle$, simulated using Qiskit. The code builds the circuit, applies two Grover iterations (oracle + diffusion), performs the measurement, and then visualizes the results by saving the histogram and circuit diagram as PNG files.

```
import math
from qiskit import QuantumRegister, ClassicalRegister,
   QuantumCircuit, transpile
from qiskit_aer import Aer
from qiskit.visualization import plot_histogram
import matplotlib.pyplot as plt


# ------------------------
# Building Grover circuit (manual, with 2 iterations)
# ------------------------
```

```
qreg_q = QuantumRegister(3, 'q')
creg_c = ClassicalRegister(3, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)

# Step 1: Uniform superposition
circuit.h(qreg_q)
circuit.barrier()

# Defining oracle for |111>
def oracle(circ, q):
    circ.h(q[2])
    circ.ccx(q[0], q[1], q[2])
    circ.h(q[2])

# Defining diffusion operator
def diffusion(circ, q):
    circ.h(q)
    circ.x(q)
    circ.h(q[2])
    circ.ccx(q[0], q[1], q[2])
    circ.h(q[2])
    circ.x(q)
    circ.h(q)

# ------------------------
# Step 2 & 3: Applying Grover iterations
# ------------------------
iterations = 2
for i in range(iterations):
    oracle(circuit, qreg_q)
    circuit.barrier()
    diffusion(circuit, qreg_q)
    circuit.barrier()
```

```
# Step 4: Measurement
circuit.measure(qreg_q, creg_c)

# ------------------------
# Simulation
# ------------------------
simulator = Aer.get_backend("qasm_simulator")
compiled = transpile(circuit, simulator)
job = simulator.run(compiled, shots=1024)
result = job.result()
counts = result.get_counts()

print("Counts:", counts)

# ------------------------
# Visualization
# ------------------------
# Histogram
hist = plot_histogram(counts, title="Grover's Algorithm Results (
    n=3, oracle=|111>, 2 iterations)")
hist.savefig("grover_hist.png")
print("Histogram saved as grover_hist.png")

# Circuit diagram
circ_fig = circuit.draw("mpl")
circ_fig.savefig("grover_circuit.png")
print("Circuit diagram saved as grover_circuit.png")

# ASCII fallback
print(circuit.draw())
```

Output :

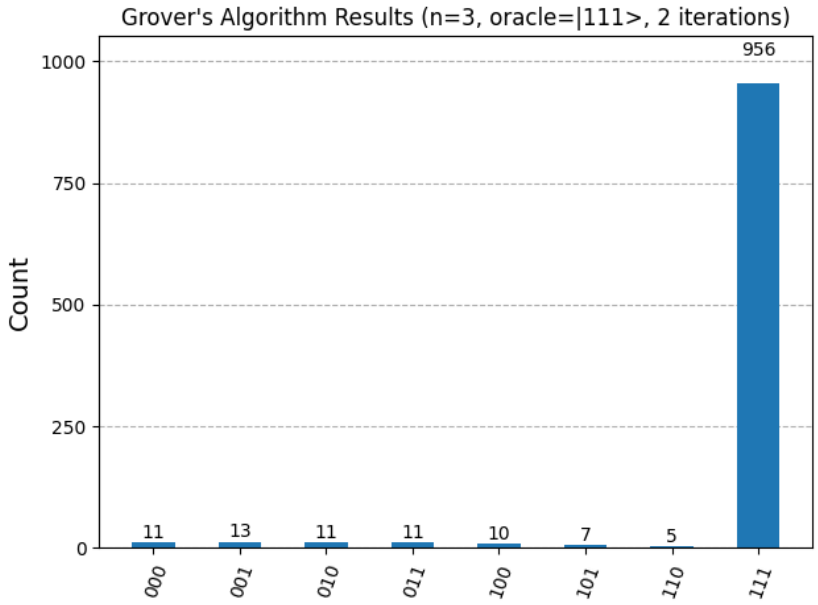


Figure 5.1: Histogram of measurement results from Grover's algorithm for $n = 3$ qubits with oracle state $|111\rangle$. The peak at "111" confirms the success of the algorithm.

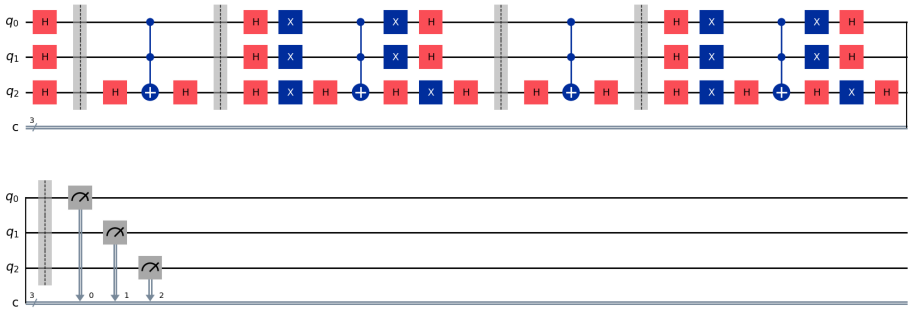


Figure 5.2: Circuit diagram of Grover's algorithm for $n = 3$ with 2 iterations. The barriers clearly separate the oracle and diffusion steps.

5.4 Future Prospects and Potential Applications

Quantum computing, though still in its early stages of development, holds immense promise for transforming both theoretical and applied domains of science and technology. As hardware improves and error-correcting codes mature, the ability to run larger-scale and more reliable quantum algorithms will steadily increase. Among the foundational algorithms, Grover's algorithm is expected to play a central role in providing practical quantum advantage for search and optimization tasks.

Grover's algorithm demonstrates a quadratic speedup over classical brute-force search, reducing the required number of queries from $O(N)$ to $O(\sqrt{N})$. While not exponential, this speedup is still highly impactful in problems where the search space grows rapidly with system size. As quantum hardware scales, Grover's search is anticipated to find applications across a broad spectrum of fields:

- **Cryptography:** Grover's algorithm can quadratically speed up brute-force attacks on symmetric-key cryptosystems, effectively halving the security level of encryption schemes such as AES. For example, a 256-bit key would offer only 128-bit security against a quantum adversary. This motivates the design of post-quantum cryptographic protocols that remain secure in the presence of quantum attacks.

- **Optimization Problems:** Many real-world problems—such as scheduling, logistics, and resource allocation—can be reduced to unstructured search or constraint satisfaction. Grover's algorithm provides a generic tool for accelerating such searches, especially when combined with problem-specific heuristics.

- **Machine Learning:** In quantum machine learning, Grover's algorithm underlies *amplitude amplification* techniques that can speed up training processes and enhance sampling methods for large datasets.

- **Database Search and Pattern Matching:** Although physical databases are not typically queried in quantum superposition, abstract problems such as text search, DNA sequence alignment, and graph traversal can benefit from Grover's quadratic speedup.

- **Mathematical Applications:** Grover's method has been extended to solve NP-complete problems, linear system solving, and satisfiability testing by providing improved subroutines for candidate solution search.

Looking ahead, the future of Grover's algorithm depends not only on algorithmic refinements but also on the advancement of fault-tolerant quantum hardware. Hybrid approaches that combine Grover's algorithm with classical pre-processing are expected to provide near-term quantum advantage in optimization and search-related domains. As quantum devices grow in scale and reliability, Grover's algorithm will continue to serve as a practical benchmark for assessing the capabilities of quantum computers, while also offering tangible advantages for both scientific research and industry.

CHAPTER 6

Realizing Quantum Computers

6.1 Introduction

The emergence of digital classical computers was predated only by a few years by the development of the modern quantum theory in the 1920s. It was realised then, that quantum mechanical models tend to be significantly more efficient in terms of time - hence, bringing the confluence of quantum mechanics and computer science as the idea of quantum computers came into being. These computers use superposition and interference to solve problems and reduce computational costs by replacing the classical bit with the qubit.

As proposed by eminent physicist Richard Feynman in 1982, one of the larger goals in quantum computing is to achieve quantum supremacy. It is a mission to demonstrate the ability of a programmable quantum computer to solve a problem that no classical computer can solve in any feasible amount of time. However, achievement of true and complete quantum supremacy remains as a theoretical idea till date although multiple claims have been made by different organizations including Google and IBM.

6.2 Realizing Quantum Computers

American computer engineer Howard Aiken once stated in 1947 that just six electronic digital computers would satisfy the computing needs of the US.

Safe to say that it was a really bad estimate !!

As we keep evolving in the field of computers, and more advanced microprocessors are generated every day – it might be pertinent to mention Moore's Law – wherein the number of transistors on a microprocessor continue to double every 18 months – leading to circuits on microprocessor which are measured on the atomic scale.

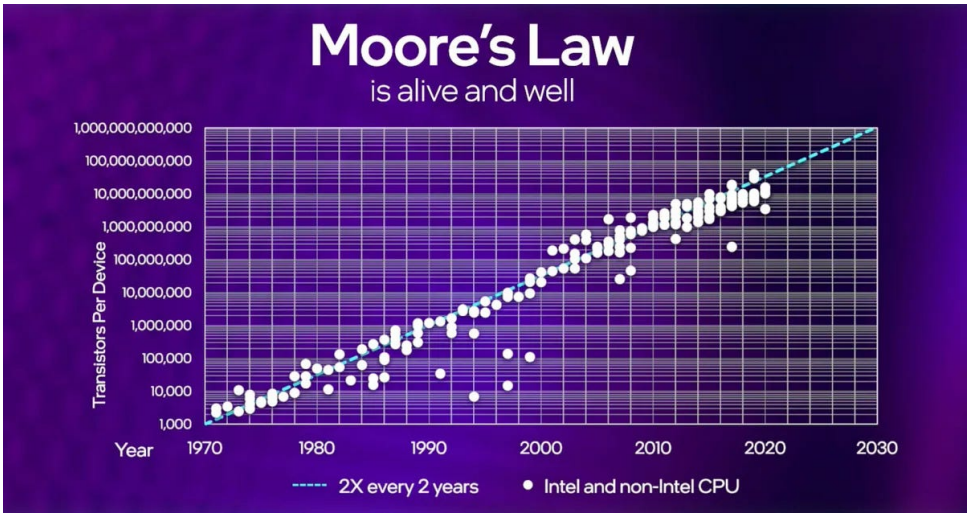


Figure 6.1: Moore's Law

Thus, the next logical step was to move towards creation of quantum computers.

6.3 David DiVincenzo's Criteria for Quantum Computers

Primarily, a quantum computer requires that its qubits must be given such a physical representation where they retain their properties as well as evolve as desired in a particular system.

Furthermore, qubits must be preparable in terms of initial states, and it must be possible to measure the final output.

1. Scalability with well characterised qubits
2. Initialising qubits into a simple fiducial state
3. Long coherence times
4. A universal set of quantum gates
5. A qubit-specific measurement capability
6. Ability to interconvert stationary and flying qubits
7. Ability to faithfully transmit flying qubits between specified locations

In totality, a quantum computer must be well isolated, yet its qubits should be accessible. This also proves to be one of the constraining or limiting factors for the actual design and implementation of most kinds of quantum computers.

6.4 Basic Hardware Outline for Quantum Computers

A quantum computer's hardware structure can in essence be divided into three primary architectural layers – which are

1. Quantum Data Plane: It comprises qubits and qubit processing units, which are basically the various physical systems by means of which qubits can be realised. Additionally, control structures necessary for the maintenance of these qubits are housed here.
2. Control and Measurement Plane: The layer involves mechanisms for converting digital signals into analog signals to manipulate the qubits, and this is where operations are performed on the qubit and measurements performed, with the aim to suppress noise paramount

Figure 6.2: A Typical Quantum Computer

3. Control Processor Plane and Host Processor: This comprises of a conventional classical host that runs the quantum software, whose instructions are taken and translated for the control and measurement plane by the control processor

6.5 Types of Quantum Computers

6.5.1 Outline

The most crucial aspect that differentiates the various kinds of quantum computers is the method by which the qubit is controlled.

1. Superconducting circuits allow the flow of electrons with almost no resistance at very low temperatures.
2. Ion traps use optical or magnetic fields and their action on the ions.
3. Optical traps use light waves to trap or control particles.
4. Quantum dots made of semiconductor materials are used to contain and manipulate electrons
5. Semiconductor impurities contain the electrons by using unwanted atoms found in the semiconductor material.

6.5.2 Quantum Computers based on the Method of Qubit Control

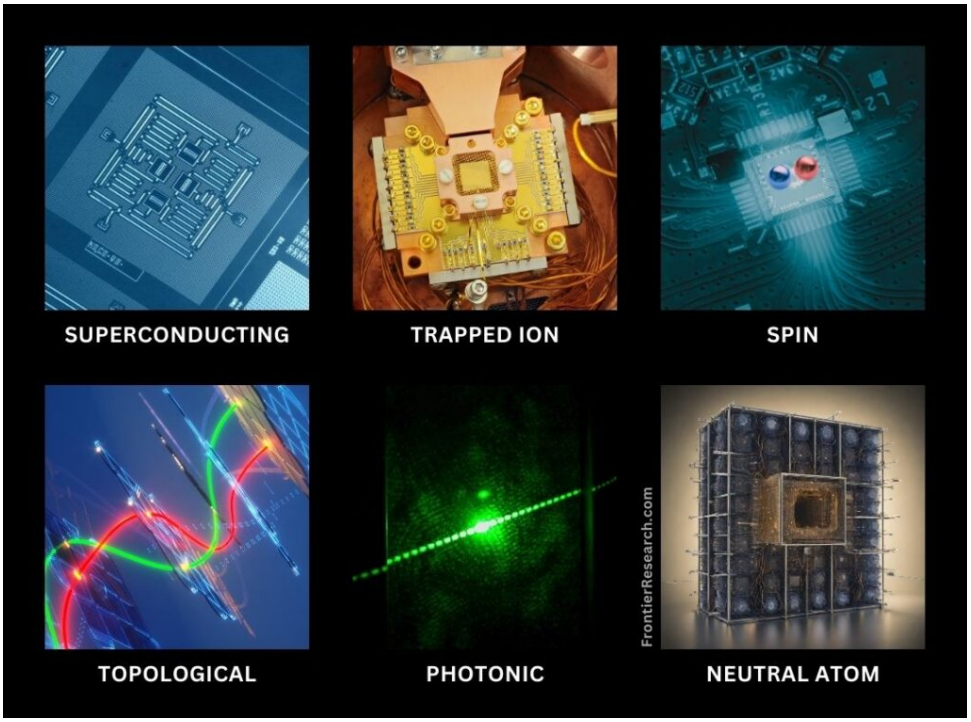


Figure 6.3: Types of Quantum Computer

- Superconducting quantum computer
- Trapped ion quantum computer
- Neutral atom quantum computer
- Nuclear magnetic resonance quantum computer
- Cavity QED-based quantum computer
- Spin qubit quantum computer
- Kane quantum computer
- Diamond-based nitrogen vacancy quantum computer
- Electron-on-helium quantum computer
- Metallic-like carbon nanospheres-based quantum computer

- Transistor-based quantum computer
- Bose-Einstein condensate based – quantum computer
- Nonlinear Optical Quantum-Computer
- Vibrational Quantum Computer
- Fullerene-based Electron spin resonance quantum computer
- Engineered Quantum Well
- Single molecule magnets

6.5.3 Quantum Computers based on their Utility

1. Experimental Quantum Computers: These computers are supposed to experiment with enhancing the capabilities of the quantum computers such as increasing qubit counts and enhancing qubit stability. These are basically focusing on R&D to enhance error corrections being majorly used in national labs.
2. Commercial Quantum Computers: The more developed systems being employed for practical and real world applications such as cryptography and optimization problems across fields such as logistics, finance, AI and pharmaceuticals including IBM Cloud.
3. Educational Quantum Computers: Designed for scientific experimentation and testing on a smaller scale and are more compact and affordable – being used by universities and other institutions
4. Cloud-Based Quantum Computers: Providing users remote accessibility without ownership of the hardware

6.6 Prominent Challenges

Some of the major challenges and hinderances that shall be problems to prioritise solving include:

1. Ionizing radiation such as cosmic rays nevertheless pose a threat to cause systems to decohere within milliseconds

2. Current quantum computers generate very limited entanglement before being overwhelmed by noise

3. A 400 qubit quantum computer may come into conflict with the cosmological information bound implied by the holographic principle – as argued by Paul Davies. The holographic principle postulates that information describing a volume of space can be encoded on a lower dimensional boundary to that region – suggesting that the 3-D universe could be a holographic projection from a 2-D surface at its boundary

Estimates suggest there are less than thousand quantum computers in existence at present – with the current key players being companies such as IBM, Google Quantum AI, SpinQ, Microsoft, D-Wave and Honeywell.

McKinsey estimates suggest at the rapid advancements in the field with 5000 operational quantum computers expected to exist but the required hardware and software advancements are to take a lot of time.

Bibliography

[1] Noson S. Yanofsky, Micro A. Mannucci, "Quantum Computing for Computer Scientists", pp. 195–204

[2] Wikipedia, "Grover's Algorithm", url : https://en.wikipedia.org/wiki/Grover%27s_algorithm

[3] L. K. Grover, "A fast quantum mechanical algorithm for database search," *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, pp. 212–219, 1996.

[4] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997.

[5] National Institute of Standards and Technology (NIST), "Post-Quantum Cryptography Standardization," Available at https://csrc.nist.gov/projects/post-quantum-cryptography

[6] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, "Quantum Amplitude Amplification and Estimation," *Contemporary Mathematics*, vol. 305, pp. 53–74, 2002.

[7] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum Machine Learning," *Nature*, vol. 549, pp. 195–202, 2017.