Anshul Choudhary - 21110028
Anurag Vishal - 21110030
Jivitesh Soneji - 21110088
Anish Karnik - 21110098
Mithil Pechimuthu - 21110129

Application of Quantum Walks in Graph Algorithms

MA201 - Mathematics III
March 2023

Acknowledgments

I want to express my profound gratitude to Professor B. Prasanna Venkatesh, Assistant Professor of Physics, for guiding us from the beginning. Your support and help have helped us explore and find an interest in a new topic. I would also like to thank Professor Tanya Kaushal Srivastava, Assistant Professor of Mathematics, for allowing us to study the applications of probability and statistics in real life through a project. The successful completion of this project does not have a singular cause but is the culmination of the whole team's hard work. I want to thank all my team members for completing this project and helping me out, overlooking my weaknesses. I thank IBM for allowing us to use their IBM Quantum facility to set up and run our quantum circuit for Grover's algorithm.

Contents

Chapter 1

Introduction

Algorithms are steps that, when followed, provide a definitive result from an arbitrary set of inputs. Since Algorithms are repetitive, we can use a computer to perform algorithms that solve our day-to-day problems. However, the time to compute the results grows as the information grows. A talisman to guide us through this issue was born in 1980 in the form of quantum mechanical computers[1]. The first problem that quantum computers (in 1994) solved faster than classical computers was the factorisation of an integer number [2]. Another problem fundamental to computer science as factorisation of an integer number is the problem of finding a unique item from an unordered (unsorted) set of N items. In this project (paper), we will introduce and analyse Grover's Algorithm [3].

Searching algorithms are a fundamental concept in computer science that are used to find specific elements or values within a collection of data. The purpose of a searching algorithm is to efficiently locate a target element in a given set of data. Classical search algorithms, such as breadth-first search, depth-first search, and A* search has several limitations. Search algorithms can take a lot of time to find a solution, especially if the search space is large. The time complexity of these algorithms can be exponential in the worst case, making them unsuitable for solving complex problems. There is a need for quantum search algorithms because traditional classical search algorithms become exponentially slower as the size of the search space grows. This limits the ability to process large amounts of data efficiently, which is a significant challenge in many fields, including computer science, physics, and chemistry. Additionally, quantum search algorithms can be used for problems that are difficult or impossible to solve using classical computers, such as simulating quantum systems, factorization of large numbers, and cryptography.

Chapter 2

Motivation

The need for quantum search algorithms arises from the limitations of classical search algorithms in handling large-scale data processing and the potential for quantum computing to provide exponential speedup for certain computational problems.

Grover's algorithm is one of the few dozen quantum algorithms known to date [4]. The classical solution to the problem of database searching will be to look at every element one by one, and if it has the unique property we were looking for, we return the item. If the chosen element is not the one we want, we mark it, so we don't look at it again.

Another motivation for using Grover's algorithm is that it is a fundamental algorithm in quantum computing and has been used as a building block for other quantum algorithms, such as Shor's algorithm for factoring large numbers. Therefore, understanding Grover's algorithm is important for developing and understanding quantum computing more broadly.

Lemma 2.1. *We must look into the database on an average, (N+1)/2 number of times to get the unique element.*

Proof. Expected no times to look into the database to find the element X: R.V. that represent the number of times we looked into the database.

$$\begin{aligned} E(x) &= \frac{1}{N} + 2\left(1-\frac{1}{N}\right)\left(\frac{1}{N-1}\right) + 3\left(1-\frac{1}{N}\right)\left(\frac{N-2}{N-1}\right)\left(\frac{1}{N-2}\right) + 4(\dots \\ &= \frac{1}{N} + \frac{2}{N} + \frac{3}{N} + \frac{4}{N} \\ &= \frac{1}{N}\left(\frac{N(N+1)}{2}\right) \\ &= \frac{N+1}{2} \end{aligned}$$

□

Note: Random Variable X has a uniform distribution (discrete).

So, the classical algorithm has a time complexity of O(N) (worst case scenario). However, Grover's quantum search algorithm quadratically improves the search time. This

improvement interested us in exploring the domain of quantum algorithms. The algorithm looks at all the elements at once and performs computations.

Chapter 3

Grover's Search Algorithm

3.1 The Problem Statement

As stated in Grover's paper[3], the abstract problem: Let a system have $N = 2n$ states labelled $S1, S2, ...SN$. Let an n-bit string represent these 2^n states. Let there be a unique state, say Sv, that satisfies the condition $C(Sv) = 1$, whereas for all other states S, $C(S) = 0$ (assume that for any state S, the condition $C(S)$ is evaluated in unit time). The problem is to identify the state Sv.

An even more academic description of the problem is as follows. In an unstructured search problem, given a set of N elements forming a set $X = x1, x2, ..., xN$ and given a boolean function $f : X \rightarrow \{0, 1\}$, the goal is to find an element $x\star$ in X such that $f(x\star) = 1$[**?**].

3.2 The Algorithm

[5]Let us first look at the basic steps a classical probabilistic algorithm follows. We examine the probability distributions over all the various states with a certain probability of being in each state. We then analyse the possibility of transitions from one state to another. It is usually done by multiplying a state transition matrix by the state probability matrix. This allows us to know the distribution at any time [3].

Similarly, quantum probabilistic algorithms use the same concept, except that we don't need the probability vector; instead, we need each state's amplitude (a complex number). The system's time evolution is obtained by pre-multiplying this amplitude vector by the transition matrix. We can make the probability vector by squaring the amplitude in each state.

We can sum up all these steps as follows:

1. Create an amplitude matrix so that each state's amplitude is equal. Note that the sum of their squares must equal one for obvious reasons.
2. Build the transformation matrix. It is usually a Walsh-Hadamard transformation matrix. For example, the transformation matrix for a "coin flip" operation on a

single qubit is given by

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

3. Then, we need to selectively rotate and amplify different states (note the probabilities shouldn't change by this transformation). For example[3]:

$$n : \begin{bmatrix} e^{j\phi_1} & 0 & 0 & 0 \\ 0 & e^{j\phi_2} & 0 & 0 \\ 0 & 0 & e^{j\phi_3} & 0 \\ 0 & 0 & 0 & e^{j\phi_4} \end{bmatrix}$$

The actual Grover's Algorithm

(i) Initialize the system to the distribution: $\left(\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}} \cdots \frac{1}{\sqrt{N}}\right)$, i.e. there is the same amplitude to be in each of the N states. This distribution can be obtained in $O(\log N)$ steps, as discussed in section 1.2.

(ii) Repeat the following unitary operations $O(\sqrt{N})$ times (the precise number of repetitions is important as discussed in [BBHT96]):

(a) Let the system be in any state S:

In case $C(S) = 1$, rotate the phase by π radians;

In case $C(S) = 0$, leave the system unaltered.

(b) Apply the diffusion transform D which is defined by the matrix D as follows:

$D_{ij} = \frac{2}{N}$ if $i \neq j$ & $D_{ii} = -1 + \frac{2}{N}$.

This diffusion transform, D, can be implemented as $D = WRW$, where R the rotation matrix & W the Walsh-Hadamard Transform Matrix are defined as follows:

$R_{ij} = 0$ if $i \neq j$;

$R_{ii} = 1$ if $i = 0$; $R_{ii} = -1$ if $i \neq 0$.

As discussed in section 1.2:

$W_{ij} = 2^{-n/2}(-1)^{\dot{i} \cdot \dot{j}}$, where $\dot{i}$ is the binary representation of i, and $\dot{i} \cdot \dot{j}$ denotes the bitwise dot product of the two n bit strings $\dot{i}$ and $\dot{j}$.

(iii) Sample the resulting state. In case $C(S_v) = 1$ there is a unique state S_v such that the final state is S_v with a probability of at least $\frac{1}{2}$.

Figure 3.1

Step 1 can be achieved using a Hadamard gate.

Step 2a can be achieved using an oracle.

$$U_\omega|x\rangle = \begin{cases} |x\rangle & \text{if } x \neq \omega \\ -|x\rangle & \text{if } x = \omega \end{cases}$$

This oracle will be a diagonal matrix, where the entry that correspond to the marked item will have a negative phase. For example, if we have three qubits and $\omega = 101$, our oracle will have the matrix:

$$U_\omega = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \leftarrow \omega = 101$$

Figure 3.2

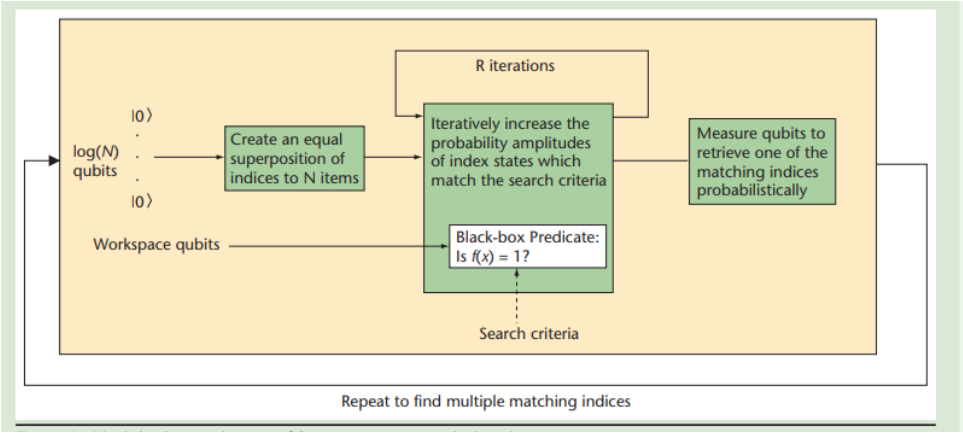

Figure 3.3: [6]

3.3 Proof of Correctness

Grover's algorithm can be mathematically proven using quantum mechanics and linear algebra. The algorithm involves the application of the Grover iteration, which is a sequence of operators that can be described using matrix multiplication.

Suppose we have an unstructured database of N items, and we are searching for a specific item marked by the function $f(x) = 1$. We can represent the database as a vector $|x\rangle$, where $x = 0, 1, 2, ..., N-1$. The initial state of the quantum computer is a superposition of all possible states:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum |x\rangle$$

We can represent the superposition using a matrix as follows:

$$\psi\rangle = \frac{1}{\sqrt{N}} \sum A|x\rangle$$

where A is an N x N matrix with all elements equal to $1/\sqrt{N}$. This is done by the Hadamard operator.

The Oracle operator U_f flips the phase of the marked state, and can be represented by the following matrix:

$$U_f = I - 2|w\rangle\langle w|$$

where I is the identity matrix, and $|w\rangle$ is the state that corresponds to the marked item. The state $|w\rangle$ can be written as:

$$w\rangle = \frac{1}{M}|x\rangle f(x)$$

where M is the number of marked items. ($M \in [0, N]$)

The Diffusion operator Us reflects the superposition about the mean state, and can be represented by the following matrix:

$U_s = 2|\psi\rangle\langle\psi| - I$, where I is the identity matrix.

The Grover iteration involves applying the Oracle operator followed by the Diffusion operator. The Grover iteration can be represented by the following matrix multiplication:

$$U_g = U_s U_f$$

After applying the Grover iteration for some k times, the state of the quantum computer can be represented by:

$$|\psi_k\rangle = U_g^k|\psi\rangle$$

We can prove that after k iterations, the probability of measuring the marked item is maximised when k is approximately equal to N2.

To see why, we first consider the amplitude of the marked state after k iterations:

$$A_k = \langle w|\psi\rangle$$

Using the matrix representation of the Grover iteration, we can derive the following recursive formula:

$$A_{k+1} = \left(1 - \frac{2}{M}\right) A_k + \left(2 \times \frac{M-1}{M}\right) A_{k-1}$$

where $A_0 = \frac{1}{\sqrt{N}}$ and $A_1 = \frac{2}{M} - \frac{1}{\sqrt{N}}$

We can solve this recursive formula to obtain:

$$A_k = \frac{\sin(2k+1)\theta}{\sin\theta}$$

where $\theta = \cos^{-1}(1 - \frac{2}{M})$ and $M = O(\sqrt{N})$.

The maximum amplitude is achieved when $sin[(2k+1)] = 1$, which occurs when k is approximately equal to N_2. Therefore, by applying the Grover iteration for $\mathrm{O}(\sqrt{N})$ times, the probability of measuring the marked item can be amplified to near certainty, and Grover's algorithm can search an unstructured database in $\mathrm{O}(\sqrt{N})$ time complexity [7].

Chapter 4

Implementation [8][9][10][11]

We used Python and the Qiskit programming language to implement Grover's algorithm. Because the grover's algorithm is a quantum algorithm, it must be written on a quantum computer in a specific programming language (here Qiskit). To work in quantum computing, we used IBM Quantum. We've also written code using Python and its various modules to demonstrate the exact process the data travels.

4.1 The Process:

1. Initialize: Begin by creating a uniform superposition of all possible bit strings using Hadamard gates. As a result, all inputs will have the same amplitude.
2. Oracle: The item bits are then processed by an oracle. There are only two outcomes from the oracle. Its amplitude will be flipped to negative if it detects the target item. The amplitudes of all other items will remain positive.
3. Amplification: We use an amplifier to magnify the amplitude difference by flipping each amplitude around the mean amplitude. Only the amplitude of the target item was changed to negative. In other words, the mean amplitude would remain positive, with a value only slightly lower than the amplitudes of the other items.
4. Repeat: Repeat steps 2 and 3.
5. Measure:

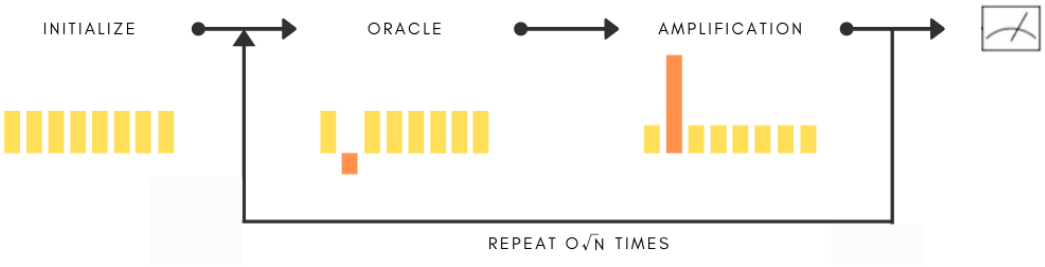

Figure 4.1

4.2 Making the Circuit

The oracle is the black box which takes the input and flips the output. The X gate flips the input. Hence, if we want to make Grover's circuit for $|001\rangle$, we need to apply the X gates in such a way that the output flips on input 001. We need to perform mathematical calculations for that and it can also be done using online simulators like the IBM Quantum.

For example let us make a circuit for $|11\rangle$.

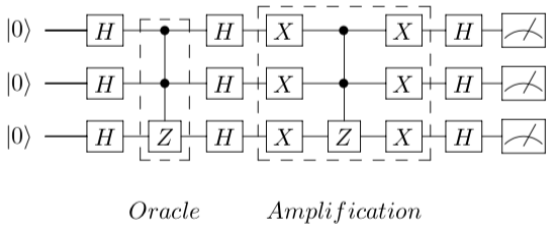

Figure 4.2

4.3 Oracle $U\omega$

The oracle $U\omega$ in this case acts as follows:

$$U\omega|s\rangle = U\omega(1/2)(|00\rangle + |01\rangle + |10\rangle + |11\rangle) = (1/2)(|00\rangle + |01\rangle + |10\rangle - |11\rangle)$$

which you may recognise as the controlled-Z gate. Thus, for this example, our oracle is simply the controlled-Z gate:

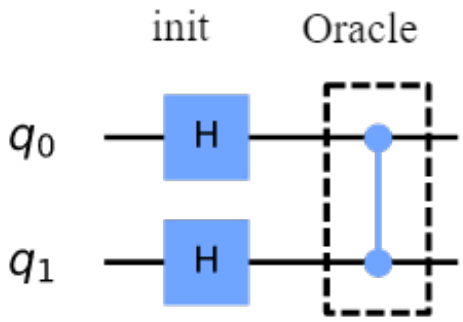

Figure 4.3

4.4 Reflection Us

To complete the circuit, we must implement the additional reflection $Us = 2|s\rangle\langle s| - I$. We want to add a negative phase to every state orthogonal to $|s\rangle$ because this is a reflection about $|s\rangle$.

One method is to use the operation that transforms the state $|s\rangle \rightarrow |0\rangle$, which we know is the Hadamard gate applied to each qubit: $H \otimes n|s\rangle = |0\rangle$. Then we apply a circuit that adds a negative phase to the states orthogonal to $|0\rangle$:

$$U0(1/2)(|00\rangle + |01\rangle + |10\rangle + |11\rangle) = (1/2)(|00\rangle - |01\rangle - |10\rangle - |11\rangle)$$

Except for $|00\rangle$, the signs of each state are flipped. The following circuit is one way of implementing $U0$ that can be easily verified:

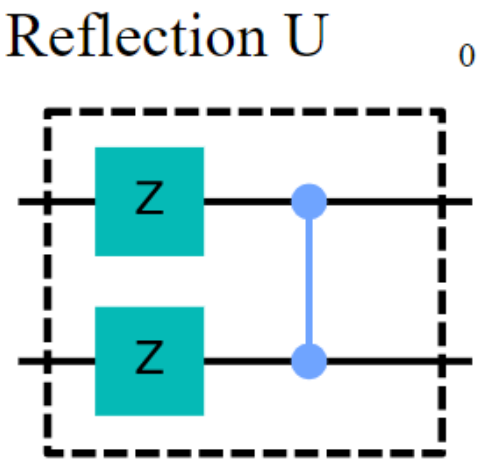

Figure 4.4

Finally, we do the operation that transforms the state $|0\rangle \rightarrow |s\rangle$ (the H-gate again). The complete circuit will be then:

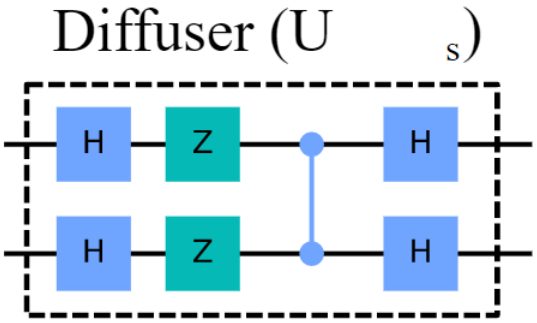

Figure 4.5

The diffuser connected with the oracle together is called the grover's circuit. The whole circuit looks like:

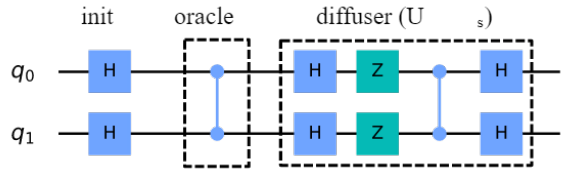

Figure 4.6

4.5 Mathematical Interpretation

1. The amplitude amplification procedure starts out in the uniform superposition $|s\rangle$, which is easily constructed from $|s\rangle = H \otimes n|0\rangle n$.

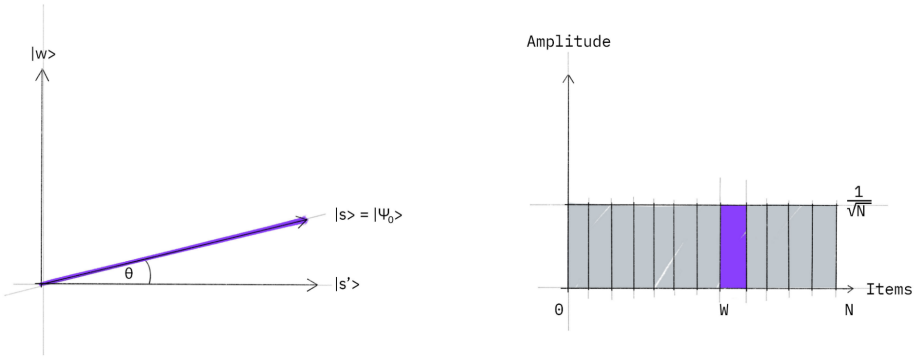

Figure 4.7

2. We apply the oracle reflection Uf to the state $|s\rangle$.

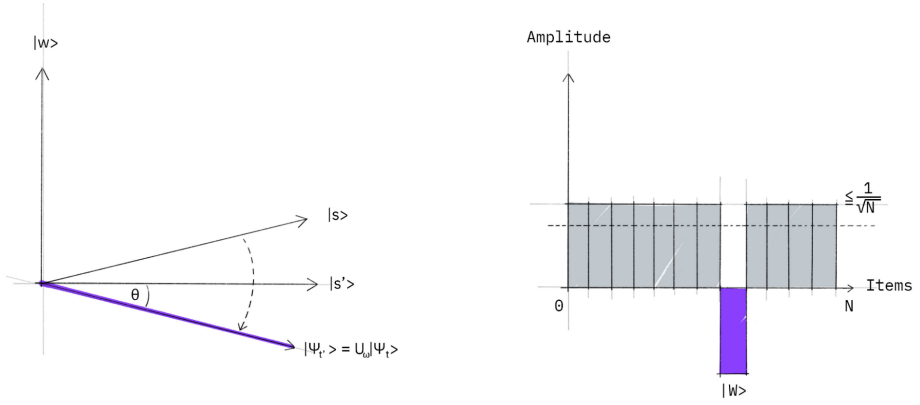

Figure 4.8

3. We now apply an additional reflection (Us) about the state $|s\rangle : Us = 2|s\rangle\langle s| - I$. This transformation maps the state to $(Us)(Uf)|s\rangle$ and completes the transformation.

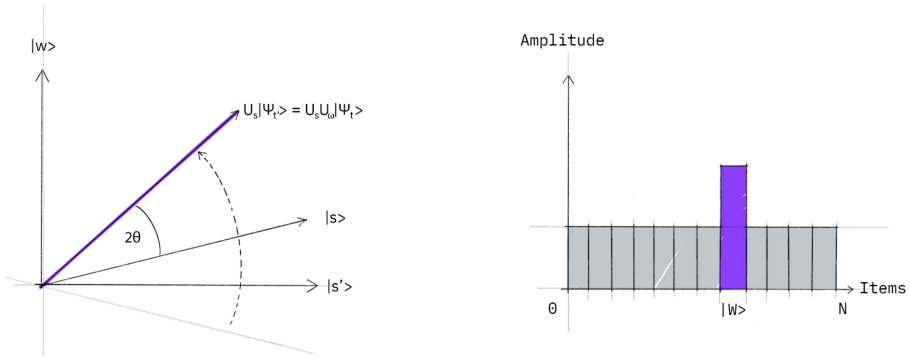

Figure 4.9

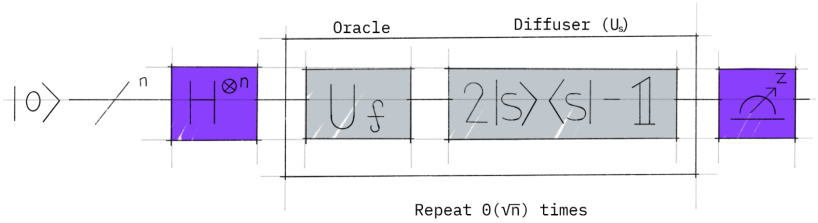

Figure 4.10

The implementation of the algorithm is done using python and the codes were saved in GitHub. The link for GitHub Repository for Grover's Algorithm is given below:
`https://github.com/anish-karnik/Grover-Algorithm-Implementation`

Results

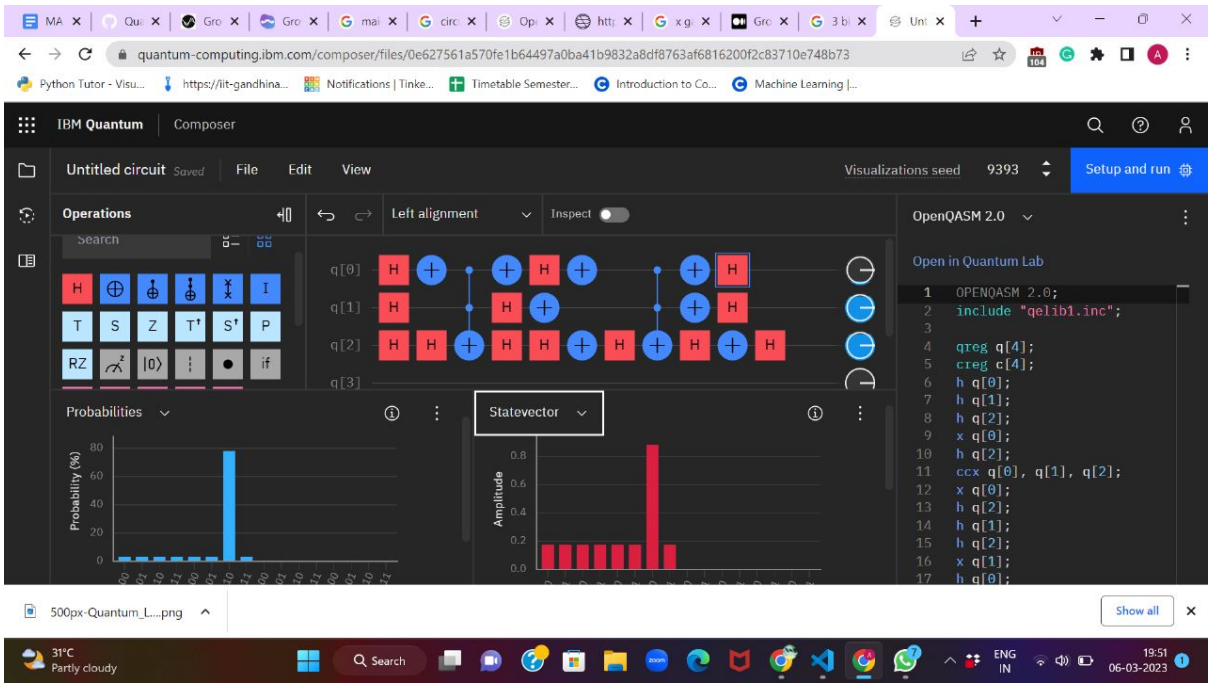

Figure 4.11

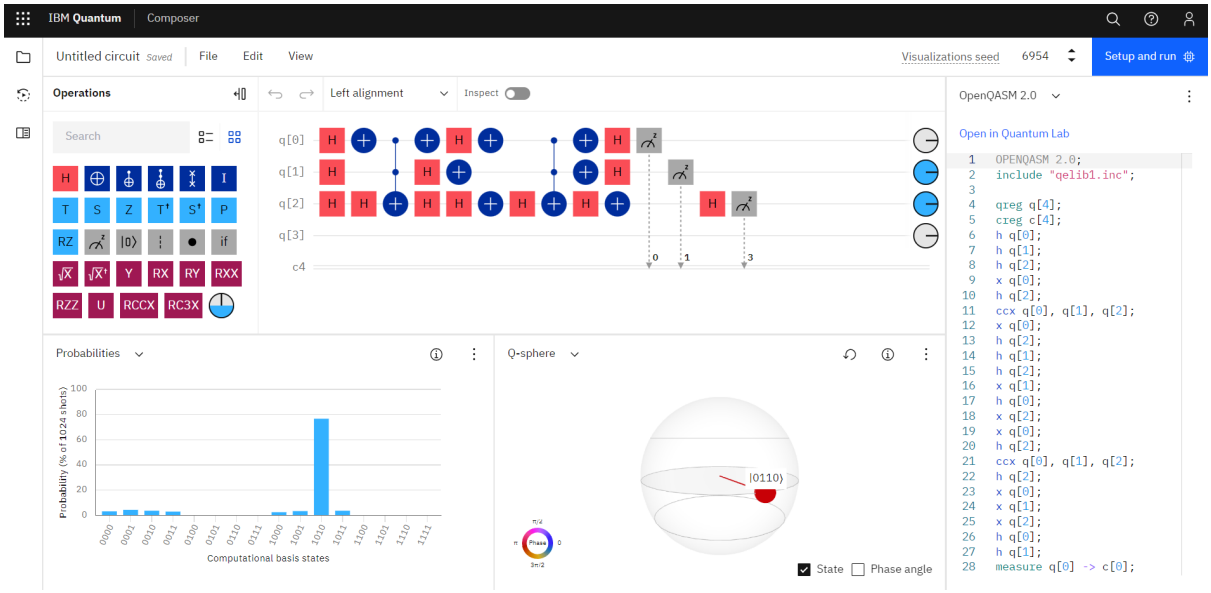

Figure 4.12

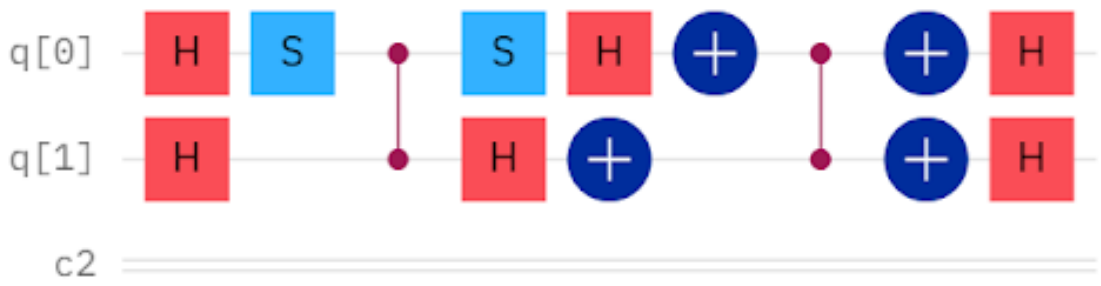

Figure 4.13

Chapter 5

Role of Probability and Statistics

The very essence of quantum mechanics is the probability of a quantum particle being in a particular state. Now, Grover's Algorithm being a quantum search algorithm, we deal with the likelihood of success and increase it with every iteration. To analyse the average time complexity of the classical search algorithm, we compute the expected value of the time taken to run over various inputs. It can be done only after analysing the random variable's probability distribution, denoting the time taken by the algorithm on inputs of different sizes. Moreover, we use specific statistical methods to compute the probability of success after every iteration. Some gates ensure that the probability distribution at various states doesn't change when the quantum system passes through. The operations and transformations done on the state vector are probabilistic, as the probability of being in that particular state is just the square of the amplitudes in the state vector.

Chapter 6

Conclusion

- This algorithm is easier to implement than other quantum algorithms because it uses Walsh-Hadamard transform and the conditional shift operation, both of which are relatively easy to implement than other operations used in other quantum algorithms [3].
- The lower bound of the running time of Grover's algorithm has been computed to be $\Omega(\sqrt{N})$. Since the algorithm has a worst case running time of $O(\sqrt{N})$, it is just a constant factor off from the best case scenario [3].
- In the Grover's algorithm we are not looking at how much time it will take to process queries from the oracle. It can be the case that if the query processing takes considerable time, then even though we ask $\sqrt{N}$ queries, the total time complexity might become as bad as the classical algorithm, i.e. O(N) [6].
- It can be combined with other efficient classical algorithms to add more improvements given that the search database has some special properties.
- Its applications include Web search engines, searching large databases for real-time processing of credit card transactions, and analysis of high-volume astronomical observations. Such databases explicitly store numerous pieces of classical information. Another class of existing applications is illustrated by code breaking and Boolean satisfiability, where the input is a mathematical function p(x) specified concisely by a formula, algorithm, or logic circuit. We seek the bits of x such that $p(x) = 1$, which might represent a correct password or encryption key. The database of all possible values of x is implicit and doesn't require large amounts of memory[6].

Work Distribution

Anshul Choudhary: Implementation + Report content + Ideation

Anurag Vishal: Report draft + report content

Jivitesh Soneji: Final Report structure + Implementation + Ideation

Anish Karnik: Implementation + Report content + Ideation

Mithil Pechimuthu: Report Draft + Report content + Ideation

Bibliography

[1] P. Benioff, "The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines," *J. Stat. Phys.*, vol. 22, no. 5, pp. 563–591, May 1980.

[2] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*. IEEE Comput. Soc. Press, 2002.

[3] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*. New York, New York, USA: ACM Press, 1996.

[4] "5 quantum algorithms that could change the world." [Online]. Available: https://www.amarchenkova.com/posts/5-quantum-algorithms-that-could-change-the-world

[5] SoniaLopezBravo, "Theory of grover's search algorithm," https://learn.microsoft.com/en-us/azure/quantum/concepts-grovers, accessed: 2023-3-11.

[6] G. Viamontes, I. Markov, and J. Hayes, "Is quantum search practical?" *Computing in Science & Engineering*, vol. 7, no. 3, pp. 62–70, 2005.

[7] A. Krahn, "Quantum computation and grover's algorithm." [Online]. Available: https://math.uchicago.edu/~may/REU2012/REUPapers/Krahn.pdf

[8] "IBM quantum," https://quantum-computing.ibm.com/, accessed: 2023-3-11.

[9] C. Figgatt, D. Maslov, K. A. Landsman, N. M. Linke, S. Debnath, and C. Monroe, "Complete 3-qubit grover search on a programmable quantum computer," *Nat. Commun.*, vol. 8, no. 1, p. 1918, Dec. 2017.

[10] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge, England: Cambridge University Press, Jun. 2012.

[11] "Grover's algorithm 2022," Nov 2022. [Online]. Available: https://qiskit.org/textbook/ch-algorithms/grover.html