Journal of Quantum Computing
DOI:10.32604/jqc.2021.018114
Article

Tech Science Press

Grover's Algorithm in a 4-Qubit Search Space

Saasha Joshi* **and Deepti Gupta**

Department of Computer Science and Engineering, University Institute of Engineering and Technology, Panjab University, Chandigarh, 160014, India

*Corresponding Author: Saasha Joshi. Email: saashajoshi08@gmail.com

Received: 10 August 2021; Accepted: 15 October 2021

Abstract: This paper provides an introduction to a quantum search algorithm, known as Grover's Algorithm, for unsorted search purposes. The algorithm is implemented in a search space of 4 qubits using the Python-based Qiskit SDK by IBM. While providing detailed proof, the computational complexity of the algorithm is generalized to n qubits. The implementation results obtained from the IBM QASM Simulator and IBMQ Santiago quantum backend are analyzed and compared. Finally, the paper discusses the challenges faced in implementation and real-life applications of the algorithm hitherto. Overall, the implementation and analysis depict the advantages of this quantum search algorithm over its classical counterparts.

Keywords: Grover's algorithm; unsorted search; Oracle; amplitude amplification; quantum computing

1 Introduction

Quantum Computing is a new computational machine concept that works on the laws of quantum physics and quantum mechanics. Where a classical computer uses bits to store information in the form of binary, 0 and 1, quantum computers use quantum bits called qubits. These two-level quantum systems display unique physical characteristics of superposition and entanglement. Qubits in superposition can represent different states at the same time in the form of a linear combination of all the possible states. The possible states of $|0\rangle$ and $|1\rangle$ of a qubit represent the basis vectors in a 2D complex vector space $\mathbb{C}^2$. Their linear combination using Dirac's Notation [1] is given as

$$|\boldsymbol{\psi}\rangle = \boldsymbol{\alpha}|\mathbf{0}\rangle + \boldsymbol{\beta}|\mathbf{1}\rangle \tag{1}$$

where the coefficients α and β are complex numbers satisfying the condition,

$$|\alpha|^2 + |\beta|^2 = 1 \tag{2}$$

with the states $|0\rangle$ and $|1\rangle$ having the probability of $|\alpha|^2$ and $|\beta|^2$, respectively, according to the Born rule [2].

Therefore, unlike a classical bit existing in a single state at a time, a qubit exists in a linear combination of all the possible states and is capable of parallelly calculating the probability of every possible qubit configuration. This makes quantum systems probabilistic up until all the qubits are measured. This is because measuring a superimposed qubit collapses it into any one of its possible states [2]. Quantum computers use this property to algorithmically increase the probability of the desired configuration to a seemingly distinguishable level. Hence, providing quantum algorithms an edge over their classical counterparts in terms of bit requirements and speed.

One of the problems where quantum computers have shown significant progress in terms of speed and space is database searching. Grover's algorithm is a quantum search algorithm that provides a quadratic speedup in the number of queries required to perform an unsorted search [3]. A classical computer can perform a search in an unsorted database of size N, with an average of $\frac{N}{2}$ brute-force queries, i.e., in O(N)

time. Grover's algorithm, on the other hand, can manage to perform the same task in O ($\sqrt{N}$) time [3,4], outperforming its classical counterpart.

In this paper, we implement Grover's algorithm in a search space of n = 4 qubits, which corresponds to a dataset of N = 2^n = 16 states. This implementation uses single-solution Boolean Oracles [1] to find a unique target state (t = x) out of the 16 possible states. In addition to using the 4 qubits, an auxiliary qubit is used to aid the construction of the oracle [5]. These Boolean Oracles are equivalent to the state marking scheme required to perform a classical brute-force search [5].

The algorithm is programmed using the Python-based Qiskit Software Development Kit (SDK) by IBM. Results are obtained by running the algorithm on two publicly available IBM quantum backends– QASM Simulator, with 32 qubit capacity, and IBMQ Santiago Quantum Device, with 5 qubit capacity. The results are compared and analyzed to draw conclusions in the final section.

2 IBM Quantum Experience (IBM QX)

IBM is one of the leading companies in the field of quantum computing. Currently, IBM publicly hosts 9 quantum devices with a qubit count ranging from 1 to 15 and a qubit volume of 8 to 32. The company also provides a publicly available 32-qubit quantum simulator that allows users to simulate quantum circuits with the least error possible. All these devices are hosted over the IBM Cloud facility through their IBM QX platform [6].

The publicly available IBM quantum backends can be accessed using their Python-based SDK-Qiskit. Qiskit can automatically process and submit quantum circuits to the available quantum devices. Furthermore, it can decompose quantum circuits into machine-executable gates and provide functions to calculate the total number of gates being implemented and the overall circuit cost. The IBM quantum computers run for a maximum of 8192 shots, with the default number of shots being 1024 in a single job. The job progress can be monitored at the IBM QX platform which also provides information such as the status, ID, queue number, and the execution time of a particular job.

In this paper, we execute Grover's algorithm on the latest IBMQ Santiago quantum backend with 5 qubits capacity [7]. The backend has a linear topology of qubits with a quantum volume of 32. It uses the Canary r3 processor and is publicly available on the IBM QX platform circa mid-2020.

3 Grover's Algorithm

As shown in Fig. 1, Grover's Algorithm for an unsorted search is implemented in four stages: Initialization, Oracle, Amplitude Amplification, and Measurement. It also consists of repeated application of a subroutine known as the Grover's Operator (G) which includes the Oracle and Amplitude Amplification stages [1]. The algorithm can be procedurally given as:

1. Initialization of qubits to a superposition state using a Hadamard Transform.

$$|\psi\rangle = \mathrm{H}(|0\rangle^{\otimes \mathrm{n}}|1\rangle) \tag{3}$$

2. Application of Grover's Operator (G) for r(N) times.

$$\mathrm{r(N)} \equiv \mathrm{O}(\sqrt{\mathrm{N}}) \tag{4}$$

3. Application of an Oracle operator (U_o) to detect the target state among the N possible states of the database.

$$U_o|x\rangle = (-1)^{f(x)}|x\rangle = \begin{cases} |x\rangle & \text{if x = x} \\ -|x\rangle & \text{if x = x'} \end{cases} \tag{5}$$

$$\mathrm{U_o}\,|\mathrm{x}\rangle = \mathrm{I} - 2|\mathrm{x}\rangle\langle\mathrm{x}| \tag{6}$$

4. Application of a diffuser (U_a) to amplify the amplitude of the target state by performing a phase shift.

$$\mathrm{U_a}|\mathrm{x}\rangle = (-1)^{\mathrm{f(x)}}|\mathrm{x}\rangle = \begin{cases} |\mathrm{x}\rangle & \text{if x = 0} \\ -|\mathrm{x}\rangle & \text{if x = x'} \end{cases} \tag{7}$$

$$U_a |x\rangle = 2|x\rangle\langle x| - I \tag{8}$$

5. Measurement of qubits to obtain the final output.

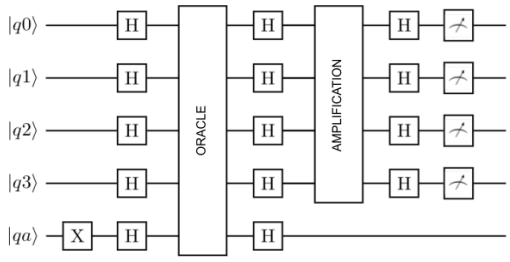


Figure 1: Stages of Grover's Algorithm

4 Implementation of Grover's Algorithm

4.1 Initialization

The initialization stage of the algorithm takes the initial quantum state of $|0\rangle^{\otimes n}|1\rangle$ and sets it into a state of equal superposition. A Hadamard gate applied to each qubit in the quantum circuit can perform this transformation. It changes the basis of a quantum state by transforming qubits from their computational basis, $|0\rangle$ and $|1\rangle$, to polar basis, $|+\rangle$ and $|-\rangle$, thus, putting them into a state of equal superposition. This transform can be represented by the matrix

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{9}$$

$$H|0\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = |+\rangle \tag{10}$$

$$H|1\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = |-\rangle \tag{11}$$

Thus, applying Hadamard transform to the initial quantum state $|\psi\rangle$ results in

$$|\psi\rangle = \sum_{x\in\{0,1\}^n} \frac{|x\rangle}{\sqrt{2^n}} \frac{|0\rangle - |1\rangle}{\sqrt{2}} \tag{12}$$

In the Qiskit implementation of the initialization stage of the algorithm as shown in Fig. 2, the state of equal superposition can be achieved by applying a Hadamard gate (H) to each qubit. The auxiliary qubit, on the other hand, is first initialized to state $|1\rangle$ by applying a NOT gate (X), and then transformed into a superposition. After the application of the Hadamard gate, each probable state configuration has an amplitude of $\frac{1}{\sqrt{N}}$.

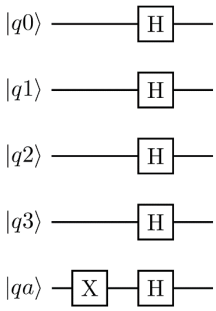


Figure 2: Initialization of Qubits

4.2 Oracle

The next stage in Grover's Algorithm helps in the application of an oracle operator which encodes the target state x′ among the N possible states in a database. This oracle is equivalent to a classical brute-force search method, satisfying the following condition:

$$f(x') = 1 \tag{13}$$

In this paper, we use a Boolean Oracle (U_o) to mark a single target state (t = x′). The Boolean method of oracle application uses an auxiliary qubit initialized to state $|1\rangle$, as discussed in the first stage. This oracle flips the auxiliary state when it encounters the target state. As a result of this flip, the amplitude of the target state is reversed while all the other states retain their original amplitudes. The amplitude of the marked state after the application of an oracle becomes $\frac{-1}{\sqrt{N}}$.

This action of the oracle operator can be given as

$$|x\rangle \rightarrow (-1)^{f(x)}|x\rangle \tag{14}$$

Here,

$$f(x) = \begin{cases} 0 & \text{if x = x} \\ 1 & \text{if x = x'} \end{cases} \tag{15}$$

This results in the oracle operator being represented as

$$U_o|x\rangle = (-1)^{f(x)}|x\rangle = \begin{cases} |x\rangle & \text{if x = x} \\ -|x\rangle & \text{if x = x'} \end{cases} \tag{16}$$

It can be compactly written as

$$U_o\,|x\rangle = I - 2|x\rangle\langle x| \tag{17}$$

The Qiskit implementation of a Boolean Oracle is achieved by using a controlled-Z (CZ) gate which is represented by the matrix,

$$CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{18}$$

A CZ gate is composed of a minimum of two qubits, a controlling qubit, and a target qubit. It performs a conditional flipping of the target qubit if and only if the control qubit has the state $|1\rangle$.

In our paper, we use the first n = 4 qubits of the quantum circuit as the control qubits and the auxiliary qubit as the target qubit. Thus, when implementing a Boolean Oracle, CZ gate performs a phase flip to the target auxiliary qubit when all the control qubits are in a state of $|1\rangle$. This flip of the auxiliary qubit, which is initialized to a superposition state of $|-\rangle$, brings an overall negative phase to the quantum state $|\psi\rangle$, while leaving all the qubits unchanged when encountered with the target state.

The 4-qubit implementation of the algorithm in this paper requires a quadruple-controlled Z (ccccZ) gate to be implemented as shown in Fig. 3. This can be achieved in the following two of the many ways using the Qiskit SDK:

1) A controlled-Z gate can be made using Hadamard gates and controlled-X (CNOT) gates [2]. This is achievable noting the fact that

$$\text{HXH} \equiv \text{Z} \tag{19}$$

A quadruple-controlled X (cccc-X) gate can be constructed using an in-built Multi-Control Toffoli (MCT) operation [5] in Qiskit. Here, the first 4 qubits are used as control gates and the auxiliary qubit acts as the target qubit. This construction is shown in Fig. 3.

2) According to Barenco et al. [8] an arbitrary Unitary gate $\wedge_m (U)$, such as a controlled-Z gate, for m > 2 qubits can be constructed from a series of CNOT gates and controlled-V gates. Here,

$$U = V^{2^{n-2}} \tag{20}$$

Total number of qubits used in our paper is 4 and 1 auxiliary qubit, i.e., n = 5. Therefore, $U = V^8$. This construction of a quadruple-controlled Z gate using a series of controlled-V gates is shown in Fig. 8 and Fig. 9 in Appendix A.

Table 3 in Appendix B displays the single-solution oracles for different marked states.

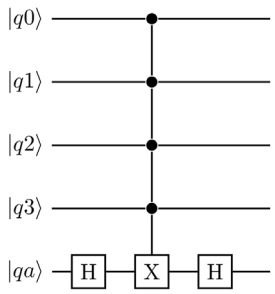


Figure 3: Construction of a ccc-Z gate using HXH gates

4.3 Amplification

After the action of the Boolean Oracle (U_o) in the previous stage, the target state has accumulated a phase of $\frac{-1}{\sqrt{N}}$. As a consequence of this negative phase, the overall mean of the amplitudes of the possible N states in the database is reduced. This average could now be given as

$$\text{Average Amplitude } (\boldsymbol{\mu}) = \frac{(\boldsymbol{N-1})\frac{\boldsymbol{1}}{\sqrt{\boldsymbol{N}}}+\frac{\boldsymbol{-1}}{\sqrt{\boldsymbol{N}}}}{\boldsymbol{N}} \tag{21}$$

Keeping in mind a reduction in the mean, the third stage in the Grover's Algorithm involves implementing a diffuser (U_a). A diffuser performs an inversion of the amplitudes of the N possible states around the new reduced mean. This inversion reverses the negative phase of the target state and helps in separating the target from the rest of the states.

By straightforward computation, the amplitude of target state (A_t) differs from the new average (μ) by a factor of

$$\delta_t = \mu - A_t \tag{22}$$

After inversion around the new mean (μ), the amplitude of the target state becomes

$$A_t = \mu + \delta_t \tag{23}$$

Thus, the state $|\psi\rangle$ after the application of a diffuser (U_a) is given as

$$U_a|\psi\rangle = \sum_x 2\mu - \delta_x|x\rangle \tag{24}$$

From the above equation, the resultant amplitude of the target state (t = x′) comes out to be greater than the amplitudes of rest of the possible N − 1 states.

$$A_t = \frac{3N-4}{N\sqrt{N}} \quad \text{and} \quad A_{N-t} = \frac{N-4}{N\sqrt{N}} \tag{25}$$

That is,

$$A_t > A_{N-t} \tag{26}$$

This action of the diffuser can be represented as a unitary operator in Dirac's notation.

$$U_a|x\rangle = (-1)^{f(x)}|x\rangle = \begin{cases} |x\rangle & \text{if x = 0} \\ -|x\rangle & \text{if x = x'} \end{cases} \tag{27}$$

That can be compactly written as,

$$U_a\,|x\rangle = 2|x\rangle\langle x| - I \tag{28}$$

The Qiskit implementation of the diffuser can be done using NOT gates (X) and a controlled-Z gate. Similar to the construction of Boolean Oracle, the X gates aid in marking the target state from among the

N possible states in the dataset. In the circuit, when encountered with a target state (t = x′), the diffuser flips the phase of this state, making it negative.

In order to flip the phase of the target state, a controlled-Z gate is used. The first 3 qubits of the circuit act as the control qubits for the controlled-Z gate, whereas, the fourth qubit of the circuit, depicted in Fig. 4. as $|q3\rangle$, is used as the target qubit. When the diffuser flips the state of the target qubit, the value of $|q3\rangle$ is changed from $|1\rangle$ to $-|1\rangle$, hence, adding a negative phase to the system.

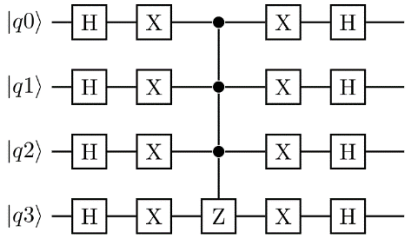


Figure 4: Diffuser for amplitude amplification

4.4 Measurement

The measurement stage of the algorithm measures the final output of the circuit. After measurement, the qubits are no longer in superposition as they tend to lose all their quantum physical properties and collapse to give an outcome of one of their possible states [2]. As a result, qubits cannot be used furthermore after their measurement is done.

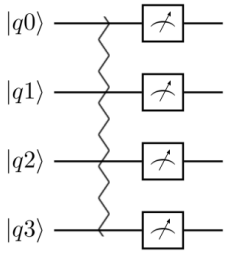


Figure 5: Measuring the first 4 qubits

In Qiskit, qubit measurement is done using the measurement gates, as shown in Fig. 5.

5 Performance

5.1 Iterations of Grover's Operator (G)

The Grover's Operator (G) consisting of the Boolean Oracle (U_o) and the diffuser (U_a), increases the probability of success of detecting the target state from the N possible states in a dataset. As shown in Table 1, a repeated application of this operator can lead to an increase in the amplitude of the target state by a factor of O($\frac{1}{\sqrt{N}}$) after each iteration [3]. As a result, an approximate O($\sqrt{N}$) number of iterations of G can result in the desired probability of the target state to reach O (1).

Table 1: Circuit cost of 4-qubit Grover's Algorithm

Number of qubits	Iterations	Number of gates	Circuit cost
4	1	163	739
	2	320	1472
	3	477	2205

However, increasing the number of iterations does not monotonically increase this probability of success [4]. Thus, it becomes significant to know the maximum number of optimal iterations (m) that G can have to have the desired amplitude of the target state. This value of m can be precisely calculated by having a geometrical visualization of the G operator [9].

Grover's iteration can be visualized as a rotation of the $|\psi\rangle$ vector in a 2-dimensional Hilbert space. By knowing that the state $|\psi\rangle$ is a uniform superposition or a linear combination of N possible states, we can decompose $|\psi\rangle$ into its target and non-target state components. This can be then written as

$$|\psi\rangle = \sum_{x\neq x'} \frac{|x\rangle}{\sqrt{2^n}} + \frac{|x'\rangle}{\sqrt{2^n}} \tag{29}$$

where, $|x'\rangle$ is the target state and $|x\rangle$ is the non-target state. A normalized collection of non-target states can thus be written as

$$|\alpha\rangle = \frac{1}{\sqrt{N-1}} \sum_{x\neq x'} |x\rangle \tag{30}$$

Rewriting the decomposition of state $|\psi\rangle$ with a normalized collection non-target states, $|\alpha\rangle$, we get

$$|\psi\rangle = \frac{\sqrt{N-1}}{\sqrt{N}}|\alpha\rangle + \frac{1}{\sqrt{N}}\,|x'\rangle \tag{31}$$

Given the operator G = $U_o U_a$, the Boolean oracle U_o performs a reflection of the vector $|\psi\rangle$ about the vector |αi in a plane defined by $|\alpha\rangle$ and $|x'\rangle$. By this action of the oracle

$$U_o|\psi\rangle = \frac{\sqrt{N-1}}{\sqrt{N}}\,|\alpha\rangle - \frac{1}{\sqrt{N}}\,|x'\rangle \tag{32}$$

The next operator, U_a, performs a reflection of the vector $U_o|\psi\rangle$ about the vector $|\psi\rangle$ in a plane defined by $|\alpha\rangle$ and $|x'\rangle$. This results in a final state of

$$G|\psi\rangle = \frac{\sqrt{N-1}}{\sqrt{N}}|\alpha\rangle + \frac{1}{\sqrt{N}}\,|x'\rangle \tag{33}$$

This shows that the overall action of the Grover's Operator G is a rotation of vector $|\psi\rangle$ in a space spanned by vectors $|\alpha\rangle$ and $|x'\rangle$. Moreover, with m iterations of the G operator, the state $G^m|\psi\rangle$ still remains in the same space.

In order to reach the target state by iterating $|\psi\rangle$ in the space, the angle of rotation available between $|\psi\rangle$ and $|x'\rangle$ must be traversed. Let this angle be given as $\frac{\theta}{2}$, where

$$\cos\left(\pi - \frac{\theta}{2}\right) = \langle x'|\psi\rangle = \frac{1}{\sqrt{N}} \tag{34}$$

Number of rotations m to traverse an angle of $\frac{\theta}{2}$ between $|\psi\rangle$ and $|x'\rangle$ can therefore be given as

$$r(N) = m = CL\left(\frac{\arccos\frac{1}{\sqrt{N}}}{\theta}\right) \tag{35}$$

Here, CL(a) function gives the integer closest to the argument a [1]. We know, for a small value of θ,

$$\theta \approx \sin\frac{\theta}{2} = \frac{1}{\sqrt{N}} \tag{36}$$

Therefore,

$$r(N) \leq \frac{\pi}{4}\sqrt{N} \tag{37}$$

Hence, $r(N) \equiv O(\sqrt{N})$.

5.2 Probability of Success

A closed-form formula for determining the probability of success of Grover's Algorithm is provided by Boyer et al. [4]. According to their paper, the algorithmic probability (ASP) of measuring a single target state (t = k) from the state $|\psi_m\rangle$ after any given number of iterations (m) is given by

$$|\psi_m\rangle = |\psi(k_m, l_m)\rangle \tag{38}$$

Here,

$$ASP(k_{m+1}) = \left(\frac{N-2}{N}k_m + \frac{2(N-1)}{N}l_m\right)^2 \quad (39)$$

$$ASP(l_{m+1}) = \left(\frac{N-2}{N}l_m - \frac{2}{N}k_m\right)^2 \quad (40)$$

It is important to note that the above formula is only for finding a unique target state from a dataset of N = 2^n probable states.

6 Result

6.1 Theoretical Result Probability

From the equation by Boyer et al. [4], the algorithmic success probability (ASP) of measuring the target state (k) after m = 1, 2, and 3 iterations is

for m = 1,

$$ASP(k_1) = \left(\frac{14}{16}k_0 + \frac{30}{16}l_0\right)^2 = 47.27\% \quad (41)$$

$$ASP(l_1) = \left(\frac{14}{16}l_0 - \frac{2}{16}k_0\right)^2 = 3.52\% \quad (42)$$

as $k_0 = l_0 = \frac{1}{\sqrt{N}}$.

for m = 2,

$$ASP(k_2) = \left(\frac{14}{16}k_1 + \frac{30}{16}l_1\right)^2 = 90.84\% \quad (43)$$

$$ASP(l_2) = \left(\frac{14}{16}l_1 - \frac{2}{16}k_1\right)^2 = 0.61\% \quad (44)$$

as $k_1 = \frac{3N-4}{N\sqrt{N}}$ and $l_1 = \frac{N-4}{N\sqrt{N}}$.

for m = 3,

$$ASP(k_3) = \left(\frac{14}{16}k_2 + \frac{30}{16}l_2\right)^2 = 151.41\% \quad (45)$$

$$ASP(l_3) = \left(\frac{14}{16}l_2 - \frac{2}{16}k_2\right)^2 = 0.26\% \quad (46)$$

as $k_2 = \frac{5N^2-20N+16}{N^2\sqrt{N}}$ and $l_2 = \frac{N^2-12N+16}{N^2\sqrt{N}}$.

6.2 Quantum Simulator (IBM QASM Simulator) Results

Grover's algorithm is first performed on IBM QASM Simulator with a 32-qubit capacity. The performance of the algorithm is tested with varying number of shots (n) and iterations (m).

For m = 1 iterations and n = 1024 shots, the probability of correctly finding the target state (t = x′) is

$$P(x') = \frac{\text{No. of times t is marked}}{\text{Total no. of shots (n)}} \quad (47)$$

$$P(x') = \frac{7689}{16*1024} = 46.92\% \quad (48)$$

Similarly for m = 1 iterations and n = 8192 shots,

$$P(x') = \frac{61890}{16*8192} = 47.28\% \quad (49)$$

Results with subsequent number of shots (n) and iterations (m) performed on the QASM Simulator are given in Table 2. Fig. 6 shows the data from the execution of Grover's Algorithm in a 4-qubit search space, performed on a simulator m = 1 and n = 1024. The x-axis of the map displays the state actually measured by the simulator during the execution and the y-axis displays the target state as marked by the Boolean oracle.

Table 2: Results

Quantum backend	Number of qubits	Number of shots	Iterations	ASP (%)	Average runtime
IBM QASM Simulator	4	1024	1	46.92	1.30
			2	49.48	1.38
			3	85.40	1.41
		8192	1	47.28	1.32
			2	48.66	1.44
			3	85.03	1.41
IBMQ Santiago	4	1024	1	6.02	14.58
			2	4.96	14.47
			3	5.58	14.62
		8192	1	5.93	16.48
			2	4.97	17.38
			3	5.73	18.43

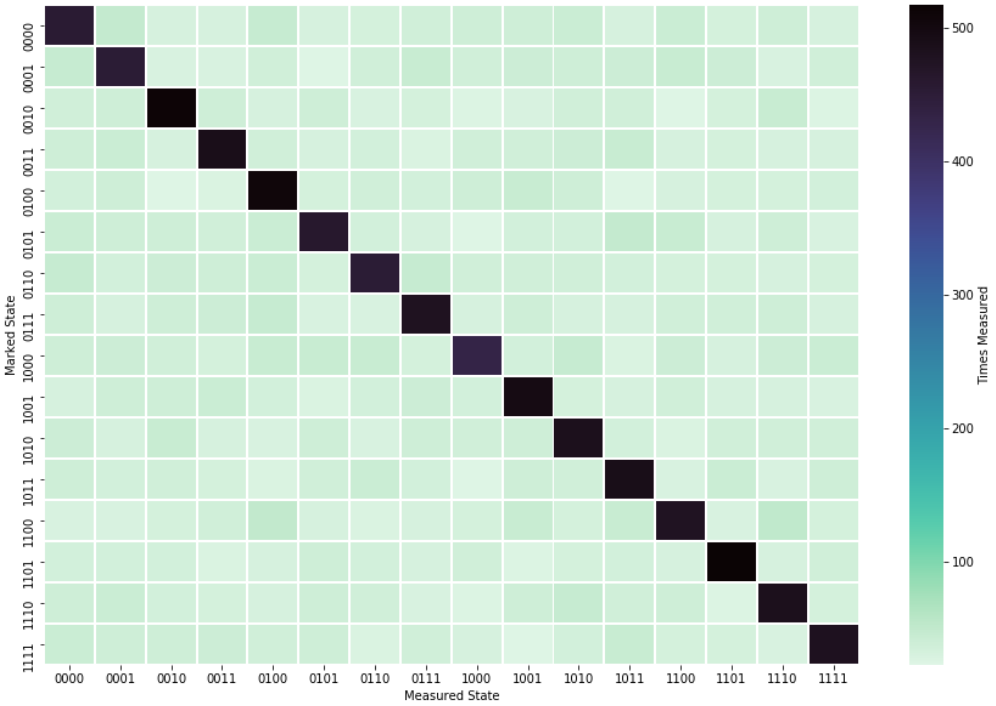


Figure 6: IBM QASM Simulator results for Grover's algorithm

6.3 Quantum Backend (IBMQ Santiago) Results

Next, the algorithm is run on a 5-qubit IBM Quantum Backend-IBMQ Santiago. The performance of the algorithm is tested with varying number of shots (n) and iterations (m).

For m = 1 iterations and n = 1024 shots, the probability of correctly finding the target state (t = x′) is

$$P(x') = \frac{874}{16*1024} = 6.02\% \quad (50)$$

Similarly for m = 1 iterations and n = 8192 shots

$$P(x') = \frac{6973}{16*8192} = 5.93\% \quad (51)$$

Results with subsequent number of shots (n) and iterations (m) performed on the IBMQ Santiago quantum backend are given in Table 2.

Fig. 7 shows the data from the execution of Grover's Algorithm in a 4-qubit search space, performed on IBMQ Santiago backend with m = 1 and n = 1024. The x-axis of the map displays the state measured by the quantum backend during the execution and the y-axis displays the target state as marked by the Boolean oracle.

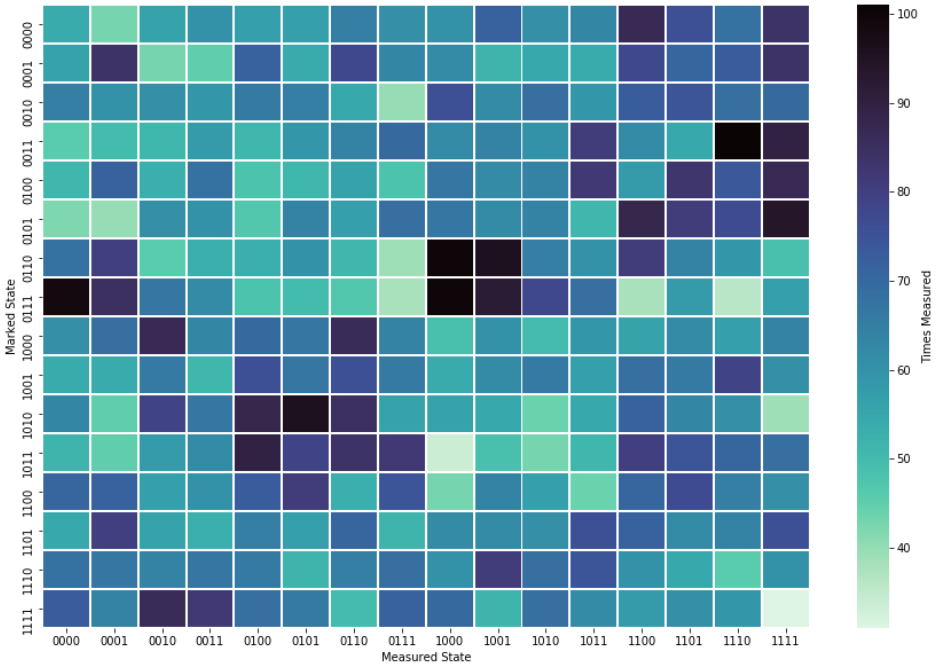


Figure 7: IBMQ Santiago results for Grover's algorithm

7 Conclusion

This paper implements Grover's Algorithm for unsorted search purposes in a search space of 4 qubits. The algorithm is run on two publicly available quantum backends-IBM QASM Simulator and IBMQ Santiago. The backends have a capacity of 32 qubits and 5 qubits, respectively.

The results obtained from the experiment are analyzed and it is found that there is a huge difference between the accuracy of finding the target state among simulators and actual quantum devices. Where the simulator shows an average algorithmic success probability (ASP) of about 47%, the actual quantum device falls short at just 6% ASP.

As discussed in the paper, increasing the number of iterations (m) of the Grover's Operator (G) displays an increase in the ASP of the algorithm. By calculating the maximum number of optimal iterations, $r(N) \leq O(\frac{\pi}{4}\sqrt{N})$,we run the algorithm for m = 1, 2 and 3 iterations. For the first two iterations, the simulator shows results that are consistent with those obtained by a single application of the G operator. In the third iteration, howsoever, the ASP shots up to near 85% for IBM QASM Simulator. This might lead us to conclude that increasing the count of m up to the optimal limit improves the performance of the algorithm. However, it is surprising and erratic that the IBMQ Santiago quantum backend gives a better ASP when run for a single iteration in n = 1024 and 8192 shots. For m = 1 iteration, the quantum backend gives an ASP of around 6%, whereas, for the rest of the vales of m, ASP decreases for around 1%. This result is irregular considering that number of iterations when increased up to the limit of $O(\frac{\pi}{4}\sqrt{N})$ should increase the ASP monotonically.

It should also be noted that the algorithm shows no variation to the different number of shots (n) it has been repeated for. The results for n = 1024, the default for IBM quantum devices, and for n = 8192, the

maximum shots IBM devices can execute, are uniform. The average runtime of the program also remains the same for corresponding values of n.

According to the theoretical results, Grover's algorithm is expected to have an ASP of 47.27% for at least m = 1 iteration. Hence, it is evident that where IBM QASM Simulator runs to achieve this value, the IBMQ Santiago quantum device reaches nowhere near the limit.

The possible reasons for the discrepancies which arose in expected and actual measurements could be attributed to the noise produced by an actual quantum device. The current quantum hardware in the NISQ [10] era is highly susceptible to environmental noise and the disturbances created by the quantum gates. The irregularity of the ASP of the quantum device for different values of m can also be well explained by knowing that the number of gates in a quantum circuit increase proportionally to the number of iterations. Moreover, the current quantum hardware differs in terms of topology and qubit quality which also produces some erroneous results. Thus, in order to obtain optimal results, one might need to develop hardware-tailored quantum algorithms which produce the least amount of noise.

Overall, it can be concluded that the current quantum hardware, such as the ones publicly available, are not yet suitable for the implementation of Grover's algorithm with an increased qubit count. In a search space of 4 qubits, this algorithm provides no useful results when run on the actual IBMQ Santiago quantum backend. The simulators on the other hand do not consider the environmental noise and other factors, and therefore provide results which are consistent with the expected theoretical calculations. Moreover, quantum devices need to be optimized to provide results in a shorter time frame as the simulators.

Acknowledgement: We are grateful to the peoples for the support and encouragement.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. A. Nielsen and I. L. Chuang, "Quantum computation and quantum information," *Mathematical Structures in Computer Science*, vol. 17, no. 6, pp. 1115, 2002.

[2] N. D. Mermin, *Quantum Computer Science (1st ed.)*, vol. 1. Cambridgeshire, England, UK: Cambridge University Press, 2007.

[3] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, New York, NY, USA, pp. 212–219, 1996.

[4] M. Boyer, B. Gilles, H. Peter and T. Alain, "Tight bounds on quantum searching," *Fortschritte der Physik. Progress of Physics*, vol. 46, no. 4, pp. 493–505, 1998.

[5] C. Figgatt, M. Dmitri, K. A. Landsman, M. L. Norbert and D. Shantanu, "Complete 3-qubit Grover search on a programmable quantum computer," *Nature Communications*, vol. 8, no. 1, pp. 1–9, 2017.

[6] The IBM Quantum Experience. [Online] Available: http://www.quantum-computing.ibm.com/.

[7] The IBM Quantum systems overview. [Online] Available: https://quantumcomputing.ibm.com/docs/manage/backends/.

[8] A. Barenco, H. B. Charles, C. Richard, P. D. David and M. Norman, "Elementary gates for quantum computation," *Physical Review A*, vol. 52, no. 5, pp. 3457–3465, 1995.

[9] A. Krahn, "Quantum computation and Grover's Algorithm", *Quantum*, vol. 1, no. 80, pp. 1–12, 2001.

[10] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, no. 79, pp. 1–11, 2018.

Appendix

A. Construction of Controlled-Z Gate (cccc-Z)

Figs. 8 and 9 show the construction of ccc-Z and cccc-Z gates [8] used in the diffuser (U_a) and Oracle (U_o), respectively.

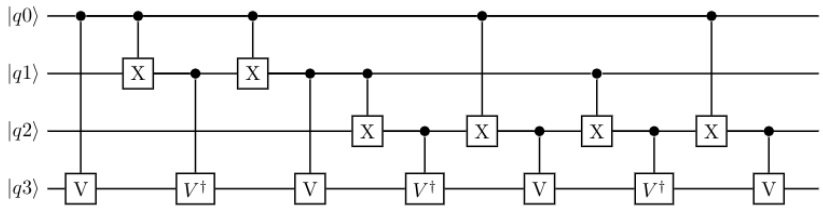


Figure 8: Construction of a ccc-Z gate using controlled-V gates

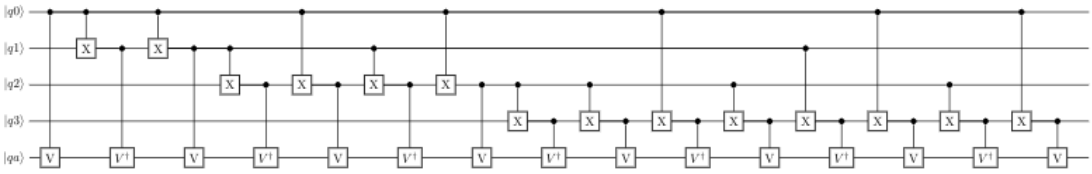
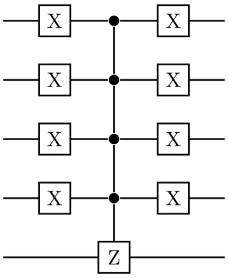
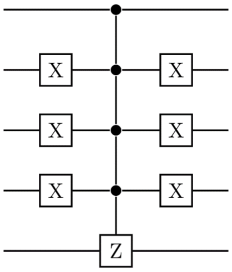
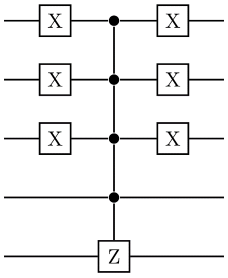
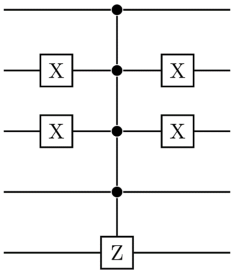


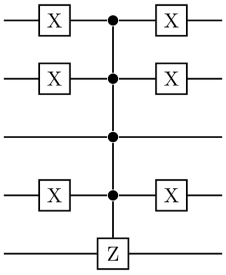
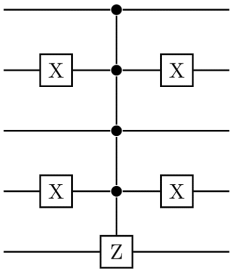
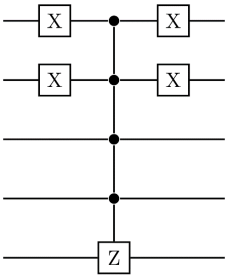
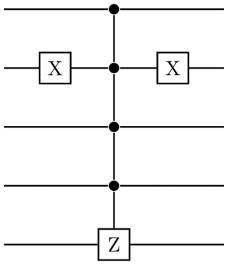
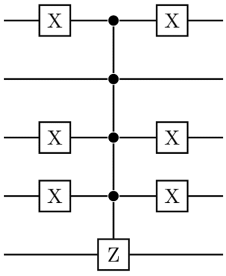
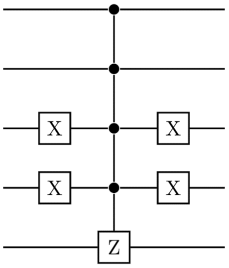
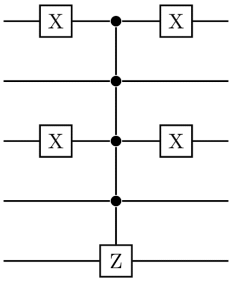
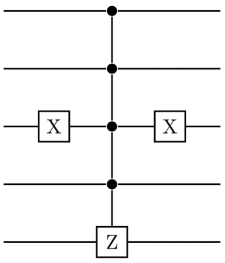
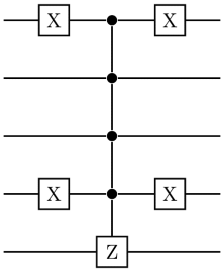
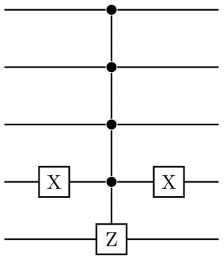
Figure 9: Construction of a cccc-Z gate using controlled-V gates

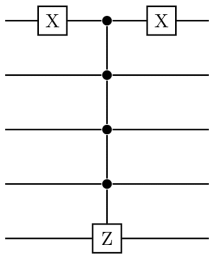
B. Single-Solution Boolean Oracles

Table 3 displays the single-solution Boolean Oracles for the N possible states.

Table 3: Single solution Boolean Oracles

Marked state	Boolean Oracle	Marked state	Boolean Oracle
0000		1000	
0001		1001	

0010		1010	
0011		1011	
0100		1100	
0101		1101	
0110		1110	

0111		1111