# Implementing Grover's Algorithm on the IBM Quantum Computers

Aamir Mandviwalla*, Keita Ohshiro* and Bo Ji

***Abstract*—This paper focuses on testing the current viability of using quantum computers for the processing of data-driven tasks fueled by emerging data science applications. We test the publicly available IBM quantum computers using Grover's algorithm, a well-known quantum search algorithm, to obtain a baseline for the general evaluations of these quantum devices and to investigate the impacts of various factors such as number of quantum bits (or qubits), qubit choice, and device choice. The main contributions of this paper include a new 4-qubit implementation of Grover's algorithm and test results showing the current capabilities of quantum computers. Our study indicates that quantum computers can currently only be used accurately for solving simple problems with very small amounts of data. There are also notable differences between different selections of the qubits in the implementation design and between different quantum devices that execute the algorithm.**

## I. Introduction

Quantum computing is a rapidly growing field with the proven potential to achieve significant speedups over classical algorithms [1]. Considering the rate at which data sets are growing larger and larger in size (reaching and exceeding Petabytes in volume), there is a growing need for algorithms that can process data in less time [2]. As a result, there is an increasing desire throughout the computing world to successfully implement quantum algorithms on a large scale [3].

One important example of quantum algorithms closely related to machine learning is the Harrow-Hassidim-Lloyd (HHL) algorithm, which can be used for inverting matrices [4]. The HHL algorithm is capable of inverting a matrix in exponentially faster time than the best classical algorithm thanks to the ability to represent $N$ bit classical data with $\log N$ quantum bits (or qubits) [4]. This result has been used in many machine learning algorithms due to the usefulness of speedy inversion of matrices in that field. Shor's factoring algorithm, perhaps an even more well-known example, has been proven to be capable of factoring large numbers exponentially faster than the best classical algorithm [5]. This result potentially has big consequences in the field of security given how prevalent the usage of Rivest-Shamir-Adleman (RSA) encryption is, which bases its security on the difficulty of factoring large numbers quickly [6]. On the other hand, the main caveat of all these advances is that working quantum computers are quite new and suffer from numerous issues that prevent them from running complex algorithms with reasonable accuracy. In other words, the physical accuracy of quantum computers has not come close to reaching the theoretical levels.

*Aamir Mandviwalla and Keita Ohshiro are co-primary authors and have equal contributions to this work.

Aamir Mandviwalla (tug16001@temple.edu), Keita Ohshiro (tug79593@temple.edu), and Bo Ji (boji@temple.edu) are with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA.

Our goal in this paper is to study how close we are hardware-wise to realistically being able to exploit the potential benefits of quantum algorithms. Prior work is rather limited for this subject. There exist articles proclaiming that the age of quantum computing is only a year or two away along with less positive articles that do not expect major advancements for at least ten years [7], [8]. Instead of providing a prediction for the future, in this paper we aim to evaluate the accuracy of quantum computers that are publicly available now.

To accomplish this goal, we run tests using Grover's algorithm implemented in a variety of different ways on the publicly available IBM computers: IBM Q 5 Tenerife (5-qubit chip) and IBM Q 16 Rüschlikon (16-qubit chip) [9]. Grover's algorithm is a simple quantum search algorithm with one specific target [10]. For example, in a 2-qubit implementation of the algorithm, the target could be (00), (01), (10), or (11). The theoretical accuracy of the algorithm is extremely high. Depending on the qubit count and method of implementation the accuracy of the algorithm varies, but for each of the cases we are considering (2-qubit, 3-qubit, and 4-qubit implementations) it is possible to achieve a theoretical accuracy of greater than 95%. The high theoretical accuracy of Grover's algorithm allows us to easily test the overall accuracy of IBM's quantum computers by checking how often the correct value is measured over the course of a large number of executions.

The two key contributions of this paper include a new 4-qubit implementation of Grover's algorithm, and our experimental results from testing multiple implementations out. Other papers have provided 2-qubit and 3-qubit implementations of the algorithm in the past, which are essentially the same as our 2-qubit and 3-qubit implementations [11], [12], [13]. The only 4-qubit implementation that has been published before used one Grover's iteration and has a theoretical accuracy of about 47% [14], while our implementation uses 3 iterations and achieves a theoretical accuracy of approximately 96%. In addition, our experimental results are original since there are very few experimental studies available about current quantum computers. These results are a valuable contribution because they give a sense of what the capabilities of current quantum computers are.

The remainder of this paper is structured as follows. In Section II, we provide readers with necessary background needed to understand the paper. Then, we explain the implementation of Grover's algorithm in Section III. Section IV presents our experiments and our experimental results. Finally, we conclude this paper in Section V.

## II. Background

In this section, we provide readers with a background on the important concepts necessary to understand our results. We

978-1-5386-5035-6/18/$31.00 ©2018 IEEE

start with a brief overview of quantum computing as a whole, and then give a more in-depth look at Grover's algorithm. We end with a description of the IBM quantum computing resources we will use to obtain our results.

### A. Quantum Computing

In this subsection, we include a brief overview of quantum computing. At an extremely low level, even the most advanced modern supercomputers ultimately rely on the state of millions of physical transistors that can be set to either 0 or 1 at any given moment. This results in a physical limitation to the growth in complexity of contemporary devices due to the inability to make these transistors infinitely small. To solve this problem, researchers including Benioff and Deutsch theorized in the 1980s about building computers that exploit the properties of quantum mechanics [15]. Replacing the binary nature of the classical bit, the qubit seeks to represent a superposition of the states 0 and 1, creating a quantum state denoted by $|\psi\rangle$. This has been accomplished using elemental particles such as electrons and photons, where the charge or polarization of the particle represents its current state.

Given the ability of each qubit to represent two states at the same time, a quantum computer can complete $2^n$ computations in one physical step, where $n$ is the number of qubits utilized. Thus, it is possible for a quantum computer to complete tasks in exponentially less steps than a classical computer.

For a detailed history of the development of quantum computing technologies, we suggest an article by Carude and Carude [8]. For a more in-depth introduction to the field of quantum computing, we refer readers to the beginner-friendly paper by Yanofsky [16].

### B. Grover's Algorithm

Grover's algorithm is a quantum search algorithm invented by Grover in 1996 [10]. Given an unsorted list of $N$ elements, Grover's algorithm enables us to find a target element with $O(\sqrt{N})$ operations, whereas a classical algorithm requires $O(N)$ operations. Therefore, it provides a quadratic speedup over its classical counterparts. Also, it has been applied as a subroutine for other quantum algorithms [11].

The search problem we consider is to find the index of the target element among the list of $N = 2^n$ elements, where $n$ is the number of qubits and $N$ is the size of the list. The procedure of Grover's algorithm is as follows:

1) Prepare $|0\rangle^{\otimes n}$ where $\otimes$ means tensor, i.e., $|0\rangle^{\otimes n}$ is equivalent to $|0\rangle \otimes |0\rangle \otimes ... \otimes |0\rangle$ with $n$ terms.
2) Apply $H^{\otimes n}$ to create a superposition.
3) Apply the oracle $O$ to mark the target element by negating its sign, i.e., $O|x\rangle = -|x\rangle$ where $|x\rangle$ is the target.
4) Apply the Grover diffusion operator $D$ to amplify the probability amplitude of the target element.
5) Repeat Steps 3) and 4) for about $\sqrt{N}$ times.
6) Perform measurements.

Fig. 1 shows the schematic circuit for Grover's algorithm. After about $\sqrt{N}$ iterations of Steps 3) and 4), we will find the target element with a high probability.

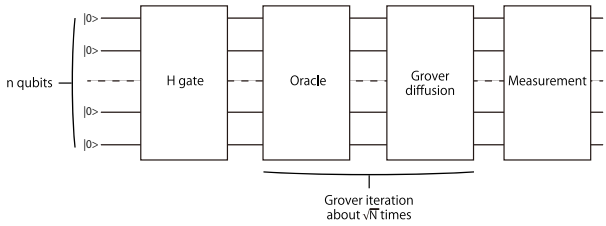


Fig. 1: Schematic circuit for the Grover's algorithm.

As written in Nielsen and Chuang [17], it is useful to note that Grover diffusion operator can be expressed as

$$2|\psi\rangle\langle\psi| - I_N = H^{\otimes n}(2|0\rangle\langle 0| - I_N)H^{\otimes n},$$

where $|\psi\rangle$ is the uniform superposition of states and $I_N$ is the $N$ dimensional identity matrix. As $2|\psi\rangle\langle\psi| - I_N$ operates a reflection about the $|\psi\rangle$, $2|0\rangle\langle 0| - I_N$ operates a reflection about the $|0\rangle$. It turns out that Grover diffusion can be implemented on a quantum circuit with a phase shift operator that negates all the states except for $|0\rangle$ sandwiched between $H^{\otimes n}$ gates (see Section III for the implementation). More detailed explanations can be found in Nielsen and Chuang [17].

### C. IBM Quantum Experience

IBM is one of the major companies that are actively investing large amounts of money into the field of quantum computing [18]. As one of the leaders in the field, at the time of writing IBM currently has three quantum computers (one 5-qubit and two 16-qubit devices) available for public use over the cloud. They also have a publicly usable quantum simulator (32-qubit), which allows users to simulate quantum algorithms without any error. In addition, IBM hosts two 20-qubit devices, which are only available to their partners.

All of IBM's devices can process jobs asynchronously created using their publicly available python library called IBM QISKit [9]. QISKit automatically processes submitted quantum circuits and decomposes them into machine executable gates. Users are allowed to execute a single circuit with up to a maximum of 8192 shots. Once a job is completed, the user is given the results in the form of how long the job took to execute (seconds), and how many times each possible measurement (e.g., (00), (01), (10), (11)) was made. This is the basis for the accuracy and time metrics used in this paper.

## III. Implementation of Grover's Algorithm

In this section, we describe our implementation of Grover's algorithm on the IBM quantum computers. We provide diagrams showing the quantum circuits used for 2-qubit, 3-qubit, and 4-qubit Grover's algorithm. We also give a brief explanation of how these diagrams represent each of the key steps of the algorithm described in Section II-B.

Our implementations of 2-qubit and 3-qubit Grover's algorithm on IBM quantum computers are similar to that of [11],

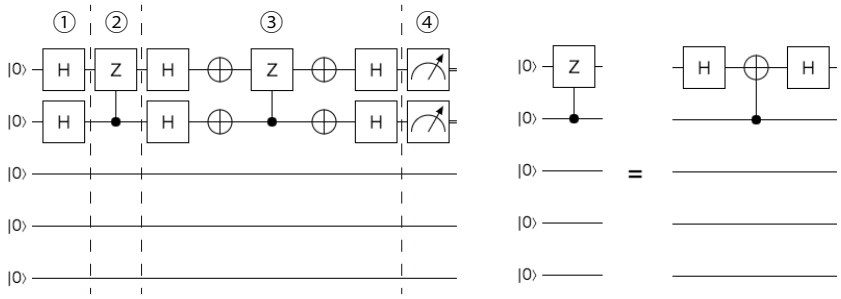


Fig. 2: Circuit for 2-qubit Grover's algorithm to find $|11\rangle$ is shown on the left. The number of Grover's iterations is one. *CZ* gate decomposition is shown on the right [17].

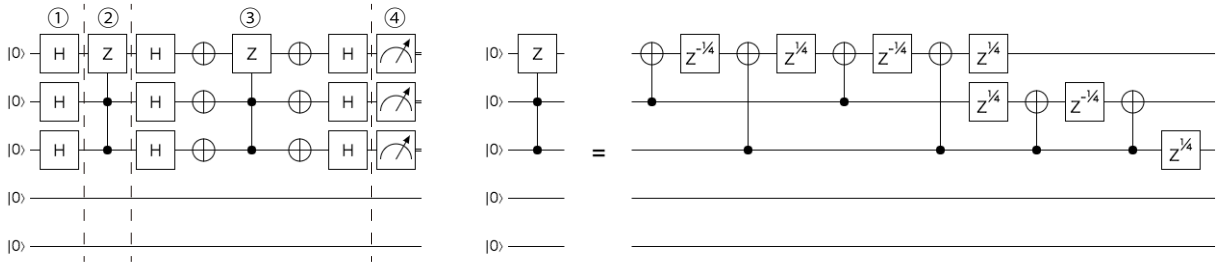


Fig. 3: Circuit for 3-qubit Grover's algorithm to find $|111\rangle$ is shown on the left. We use two Grover's iterations to increase the probability of the measurement. *CCZ* gate decomposition is shown on the right. Note that $Z^{\frac{1}{4}}$ gate is equivalent to $T$ gate and $Z^{\frac{-1}{4}}$ gate is equivalent to $T^{\dagger}$ gate in QISKit [19].

| Algorithm (qubits used) | # of Gates | Accuracy | Execution Time (s) |
|---|---|---|---|
| Grover 2-qubit (0,1) | 18 | 74.05% | 84.56 |
| Grover 3-qubit (0,1,2) | 33 | 59.69% | 84.33 |
| Grover 4-qubit (1,2,3,4) | 632 | 6.56% | 185.13 |

TABLE I: Experiment 1: trials run with 8192 shots for 2-qubit, 3-qubit, and 4-qubit Grover's algorithm on IBM Q 5 Tenerife

[12], [13]. The only implementation of 4-qubit Grover's algorithm that has been demonstrated before on the IBM devices uses just one Grover's iteration and achieves a theoretical accuracy of just under 48% [14]. Additional Grover's iterations amplify the probability of measuring the correct value, and therefore increase the theoretical accuracy of the circuit. Our implementation uses 3 iterations and achieves a theoretical accuracy of approximately 96%.

Figs. 2, 3, and 4 show the base circuits of 2-qubit, 3-qubit, and 4-qubit Grover's algorithm, respectively, along with the necessary gate decomposition for each circuit (*CZ*, *CCZ*, and *CCCZ*, respectively). Each circuit consists of four parts. The first part (denoted by ① in the diagrams) is a set of *H* gates, which creates a uniform superposition. The second part (denoted by ②), consisting of a *CZ*, *CCZ*, or *CCCZ* gate, is the oracle to negate the target. QISKit does not natively support the execution of these gates due to the complexity of compiling them into a circuit of simpler, machine-executable gates. As a result, we must decompose these gates before attempting to run them on the IBM machines. A detailed explanation of the *CCCZ* gate decomposition is included in the Appendix. The third part (denoted by ③) is the Grover diffusion operator to amplify the probability amplitude of the target. Parts ② and ③ together are called a Grover's iteration. Finally, we perform measurements in the fourth part (denoted by ④).

## IV. Experiments and Results

In this section, we provide detailed discussions about five experiments we complete on IBM's quantum devices. Each experiment is accompanied by a brief explanation of the experiment and the motivations for running it. For each experiment, tables are included to show the results, and there is a detailed analysis of the results and their implications.

### A. Experiment 1: Number of Qubits

To observe the effect of increasing the length/complexity of an algorithm, we implement Grover's algorithm using a target of 2 qubits, 3 qubits, and lastly 4 qubits on IBM Q 5 Tenerife (5-qubit machine). Increasing the target size necessarily increases the number of gates used to complete the algorithm. In order to achieve the maximum theoretical accuracy on higher qubit implementations of Grover's algorithm such as the 4-qubit implementation, the Grover's iteration portion of the algorithm must be repeated multiple times. The 4-qubit implementation requires three Grover's iterations to achieve a theoretical accuracy of approximately 96%. Thus, 4-qubit and higher implementations of Grover's

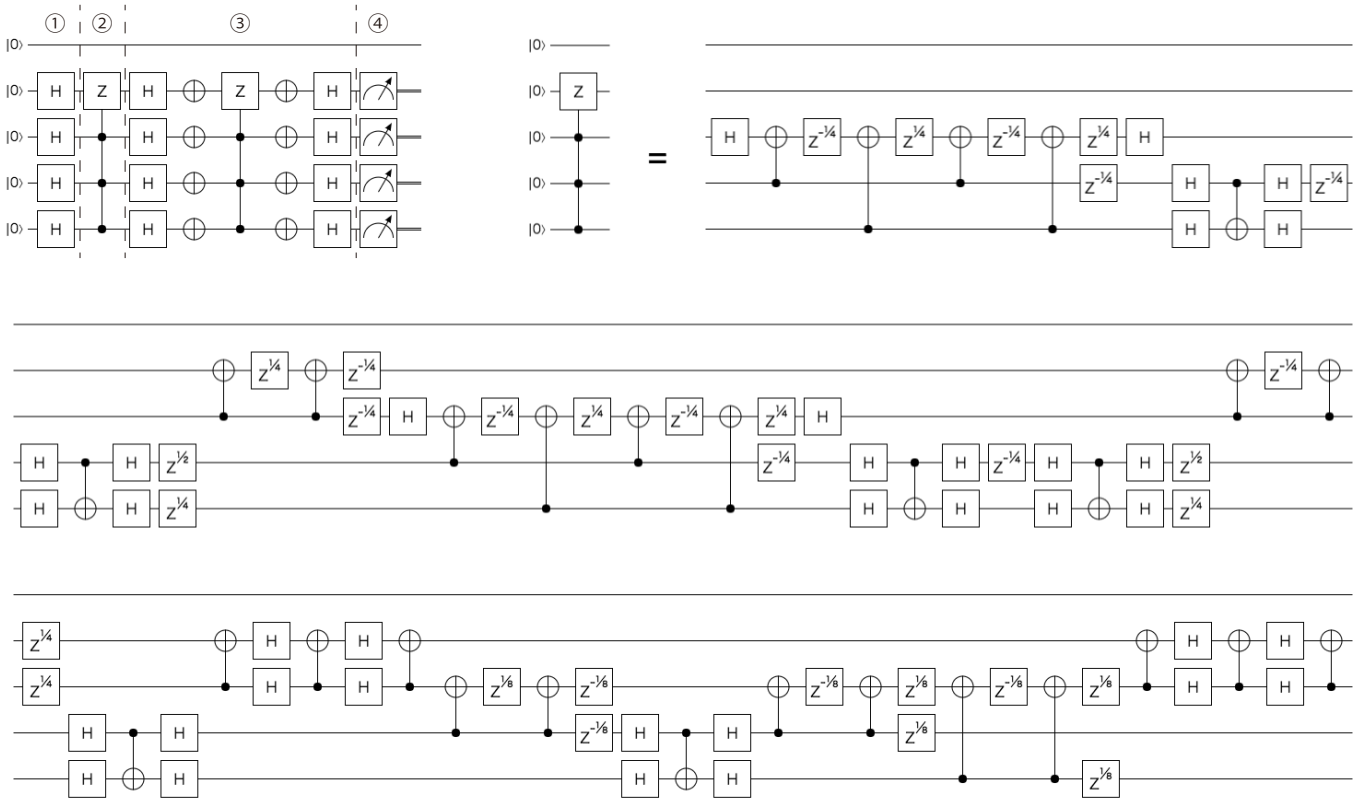

Fig. 4: Circuit for 4-qubit Grover's algorithm to find $|1111\rangle$ is shown in the top-left. We use three Grover's iterations to increase the probability of the measurement. *CCCZ* gate decomposition is shown on the right. Note that $Z^{\frac{1}{2}}$ is equivalent to $S$ gate in QISKit, $Z^{-\frac{1}{2}}$ to $S^{\dagger}$ gate, $Z^{\frac{1}{4}}$ to $T$ gate, and $Z^{-\frac{1}{4}}$ to $T^{\dagger}$ gate, $Z^{\frac{1}{8}} = U1(\frac{\pi}{8})$ gate and $Z^{-\frac{1}{8}} = U1(-\frac{\pi}{8})$ gate respectively. Also note that details of *CCCZ* decomposition are explained in the Appendix.

| Algorithm (qubits used) | Accuracy | Execution Time (s) |
|---|---|---|
| Grover 2-qubit (0,1) | 74.05% | 84.56 |
| Grover 2-qubit (0,2) | 80.85% | 84.61 |
| Grover 2-qubit (1,2) | 78.47% | 84.51 |
| Grover 2-qubit (2,3) | 61.51% | 84.60 |
| Grover 2-qubit (4,3) | 52.89% | 85.47 |
| Grover 2-qubit (2,4) | 55.37% | 84.73 |

TABLE II: Experiments 2 & 3: trials run with 8192 shots for 2-qubit Grover's algorithm with different qubit combinations on IBM Q 5 Tenerife

algorithm require significantly more gates than 2-qubit and 3-qubit implementations.

Our test results (Table I) show a notable decrease in accuracy from 2-qubit Grover's algorithm to 3-qubit and 4-qubit Grover's algorithm. 3-qubit Grover's algorithm is roughly 15% less accurate than 2-qubit Grover's algorithm, and 4-qubit Grover's algorithm has essentially unusable results (less than 7% accuracy). Given the theoretical accuracy of our 4-qubit Grover's implementation is roughly 96%, the actual accuracy being about 7% shows that there is a significant gap between current quantum computers and the theoretical maximum. While the general trend of longer quantum circuits being less accurate is expected, it is surprising to see that 4-qubit Grover's algorithm seemingly cannot be accurately run on current quantum computers. A classical computer could perform the same task as 4-qubit Grover's algorithm (finding a target value in a list of $2^4$ = 16 values) with 100% accuracy in at most 16 steps using linear search. This shows that current quantum computers are nowhere close to matching, let alone surpassing, the speed and accuracy of classical computers.

### *B. Experiment 2: Qubit Choice*

To compare the differences between different qubit choices and to observe the importance of selecting the best qubits when building a quantum circuit, we implement Grover's algorithm using a target of 2 qubits on every pair of qubits natively allowed (*CNOT* gates can only be directly implemented between qubits with a physical link) on both IBM Q 5 Tenerife and IBM Q 16 Rüschlikon. The full list of qubit combinations we consider can be found in Tables II and III.

Our experimental results reveal the importance of choosing the best qubits and qubit pairs when implementing a quantum algorithm. The best qubit pair, (0,2) on IBM Q 5 Tenerife, achieves an accuracy of 80.85%, which is nearly 30% higher than that of the worst qubit pair, (4,3) on IBM Q 5 Tenerife.

| Algorithm (qubits used) | Accuracy | Execution Time (s) |
|---|---|---|
| Grover 2-qubit (0,1) | 72.79% | 54.47 |
| Grover 2-qubit (2,1) | 65.37% | 56.22 |
| Grover 2-qubit (3,2) | 67.91% | 54.49 |
| Grover 2-qubit (4,3) | 77.48% | 54.12 |
| Grover 2-qubit (14,3) | 76.17% | 54.57 |
| Grover 2-qubit (4,5) | 73.88% | 54.88 |
| Grover 2-qubit (5,6) | 73.41% | 54.51 |
| Grover 2-qubit (11,6) | 74.05% | 54.89 |
| Grover 2-qubit (10,7) | 75.90% | 55.24 |
| Grover 2-qubit (7,8) | 76.40% | 56.01 |
| Grover 2-qubit (8,9) | 76.73% | 55.09 |
| Grover 2-qubit (10,9) | 74.38% | 54.04 |
| Grover 2-qubit (10,11) | 76.15% | 55.02 |
| Grover 2-qubit (5,12) | 74.56% | 54.23 |
| Grover 2-qubit (11,12) | 71.88% | 58.68 |
| Grover 2-qubit (13,12) | 77.76% | 55.75 |
| Grover 2-qubit (4,13) | 78.03% | 54.69 |
| Grover 2-qubit (14,13) | 77.92% | 55.50 |
| Grover 2-qubit (0,15) | 78.45% | 55.26 |
| Grover 2-qubit (2,15) | 68.97% | 54.37 |
| Grover 2-qubit (14,15) | 75.15% | 56.66 |

TABLE III: Experiments 2 & 3: trials run with 8192 shots for 2-qubit Grover's algorithm with different qubit combinations on IBM Q 16 Rüschlikon

| Device | Average Accuracy | Standard Deviation of Accuracy | Average Execution Time (s) |
|---|---|---|---|
| IBM Q 16 | 74.44% | 3.41% | 55.17 |
| IBM Q 5 | 67.19% | 11.08% | 84.75 |

TABLE IV: Average and standard deviation of accuracy

This shows that when implementing a longer quantum algorithm for real-world applications, it is important to understand the significant difference in the accuracy of different qubit combinations and to use the best qubit combinations whenever possible.

### C. Experiment 3: Device Comparison

To compare the accuracy and speed of each of the IBM quantum devices, we will compute the averages and standard deviations for these values from the results of Experiment 2.

Comparing the results of Experiment 2 for IBM Q 5 Tenerife (Table II) with that for IBM Q 16 Rüschlikon (Table III) reveals some interesting differences between the machines, besides just their qubit counts. IBM Q 5 Tenerife has an overall lower average accuracy than IBM Q 16 Rüschlikon along with taking much longer to complete the same number of shots of the same quantum circuits. Interestingly enough, IBM Q 5 Tenerife boasts both the highest accuracy qubit pair (0,2) and the lowest accuracy pair (4,3). It is thus unsurprising that the standard deviation of accuracy of its qubit pairs is more than triple that of IBM Q 16 Rüschlikon (see Table IV). This suggests that qubit selection is much more important on IBM Q 5 Tenerife than on IBM Q 16 Rüschlikon. This is likely due to differences in the physical architecture of each machine. IBM Q 5 Tenerife is an older machine, which features a qubit (qubit 2) that is connected to all four of the other qubits in the device, while the other qubits are only physically connected to two qubits each. In comparison, IBM Q 16 Rüschlikon has no qubit physically connected to more than three other qubits.

| Algorithm (qubits used) | Shots | Accuracy | Execution Time (s) |
|---|---|---|---|
| Grover 2-qubit (0,1) | 1 | 100.00% | 15.07 |
| Grover 2-qubit (0,1) | 1024 | 77.25% | 20.57 |
| Grover 2-qubit (0,1) | 2048 | 79.49% | 29.04 |
| Grover 2-qubit (0,1) | 4096 | 78.76% | 48.71 |
| Grover 2-qubit (0,1) | 8192 | 78.36% | 84.31 |

TABLE V: Experiment 4: trials run with varying shot counts for 2-qubit Grover's algorithm on IBM Q 5 Tenerife

| Algorithm (qubits used) | Accuracy | Execution Time (s) |
|---|---|---|
| Grover 2-qubit (0,1) | 69.82% | 54.00 |
| Grover 2-qubit (0,1) | 69.64% | 55.72 |
| Grover 2-qubit (0,1) | 73.43% | 58.06 |
| Grover 2-qubit (0,1) | 69.60% | 55.36 |
| Grover 2-qubit (0,1) | 71.45% | 54.55 |

TABLE VI: Experiment 5: trials run with 8192 shots for 2-qubit Grover's algorithm on IBM Q 16 Rüschlikon

### D. Experiment 4: Shot Count and Execution Time

To observe the effect of loading the quantum computer with varying numbers of executions, we repeatedly execute 2-qubit Grover's algorithm using qubit pair (0,1) on IBM Q 5 Tenerife with varying shot counts from 1 to 8192.

Our test results (Table V) indicate that there does not seem to be an accuracy penalty for running more shots in a single job execution on the machine. This suggests that the machine is able to completely revert back to the ground state in between each shot. The one qubit test shows that there is a significant amount of pre- and post-processing done before and after each shot is being run, which takes approximately 15 seconds.

Using this data (increase in execution time divided by increase in shots) we calculate the average time per shot for IBM Q 5 Tenerife to be about 8.9 milliseconds per shot. This suggests that there is another significant chunk of pre- and post-processing being done before and after each individual shot given that an individual shot should take less than a millisecond to complete [20].

These results are important because they show that quantum computers do not just suffer from an accuracy problem. They reveal that current quantum devices also suffer from a significant amount of overhead work, which prevents algorithms from being run in their theoretical minimum time.

### E. Experiment 5: Variance

To get a sense of the overall variance in results produced by the quantum computers, we execute 2-qubit Grover's algorithm using qubit pair (0,1) repeatedly using the maximum number of shots on IBM Q 16 Rüschlikon (8192 shots per job execution). The goal of this experiment is to determine a reasonable error bar for results obtained from the IBM quantum computers.

With a standard deviation of just 1.49% over 40,960 shots (see Table VI), there is a relatively low amount of variance when re-running the same quantum circuit repeatedly.

## V. Conclusion

Our experimental findings suggest that there appears to be a long way to go before quantum computers are capable of surpassing classical computers in accuracy or speed. The only algorithms that can be run on current publicly available quantum computers with meaningful accuracy are extremely short and serve almost no real-world purpose. That being said, IBM and other quantum computing front-runners are constantly updating their quantum devices and releasing new ones. It will be important in the future to check the progress of these new and updated devices to get a sense for how quickly quantum computing technology is growing. When breakthroughs in quantum computing do occur, large amounts of data can be processed in much shorter times, and many of the most commonly employed cryptographic techniques will become obsolete. It is thus very important to keep track of the progress of quantum computing technologies and how close we are to gaining the promised speedups.

We have also shown that current quantum algorithm implementations can be heavily affected by qubit choice and that researchers will need to take this into account in the future when attempting to accurately implement useful algorithms becomes a real possibility. Those looking to implement their algorithms on real quantum computers need to focus on designing their implementations to use the best qubit combinations possible. IBM measures the multi-qubit gate error between each pair of qubits on each of their devices daily and provides this information on their website [9]. Any *CX* gates that are needed as part of a quantum circuit should be designed to be placed across the qubits with the lowest multi-qubit gate error.

Additionally, our experimental results showed that current quantum computers are running slower than expected. Individual shots of 2-qubit Grover's algorithm should take less than a millisecond for an ideal quantum computer to complete, but we found that they take roughly 8.9 milliseconds each [20]. This means that even if current quantum computers were capable of producing results with reasonable accuracy for longer algorithms, they would still end up being much slower than their classical counterparts. This is a less obvious statistic that must be monitored in the future before quantum computers can be considered viable for real-world tasks.

## Acknowledgement

This work was supported in part by the NSF under Grants CCF-1657162 and CNS-1651947. Aamir Mandviwalla was supported by the NSF under REU Supplement of Grant CCF-1657162 and the Summer Merit Scholarship from Temple University. Keita Ohshiro was supported by the Undergraduate Research Program of Temple University.

## References

[1] T. D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe, and J. L. OBrien, "Quantum computers," *Nature*, vol. 464, no. 7285, p. 45, 2010.

[2] A. S. Szalay, J. Gray, and J. Vandenberg, "Petabyte scale data mining: dream or reality?" in *Survey and Other Telescope Technologies and Discoveries*, vol. 4836. International Society for Optics and Photonics, 2002, pp. 333–339.

[3] R. de Wolf, "The Potential Impact of Quantum Computers on Society," *ArXiv e-prints*, Dec. 2017.

[4] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum Algorithm for Linear Systems of Equations," *Physical Review Letters*, vol. 103, no. 15, p. 150502, Oct. 2009.

[5] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.

[6] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[7] M. Dyakonov, "Prospects for quantum computing: Extremely doubtful," in *International Journal of Modern Physics: Conference Series*, vol. 33. World Scientific, 2014, p. 1460357.

[8] C. S. Calude and E. Calude, "The road to quantum computational supremacy," *arXiv preprint arXiv:1712.01356*, 2017.

[9] I. Corporation, "Ibm quantum experience," https://quantumexperience.ng.bluemix.net/, accessed: 2018-10-06.

[10] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. ACM, 1996, pp. 212–219.

[11] P. J. Coles, S. Eidenbenz, S. Pakin, A. Adedoyin, J. Ambrosiano, P. Anisimov, W. Casper, G. Chennupati, C. Coffrin, H. Djidjev *et al.*, "Quantum algorithm implementations for beginners," *arXiv preprint arXiv:1804.03719*, 2018.

[12] W. Hu, "Empirical analysis of decision making of an ai agent on ibms 5q quantum computer," *Natural Science*, vol. 10, no. 01, p. 45, 2018.

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

[14] P. Strömberg and V. Blomkvist Karlsson, "4-qubit grover's algorithm implemented for the ibmqx5 architecture."

[15] D. Deutsch, "Quantum communication thwarts eavesdroppers," 1989.

[16] N. S. Yanofsky, "An introduction to quantum computing," *arXiv preprint arXiv:0708.0261*, 2007.

[17] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, New York, NY, USA, 2011.

[18] I. Murphy, "Ibm ups the stakes for quantum computing," https://www.enterprisetimes.co.uk/2017/11/13/ibm-ups-stakes-quantum-computing, 2017, accessed: 2018-10-06.

[19] C. Gidney, "Using quantum gates instead of ancilla bits," http://algassert.com/circuits/2015/06/22/Using-Quantum-Gates-instead-of-Ancilla-Bits.html, 2015, accessed: 2018-10-06.

[20] K. Michielsen, M. Nocon, D. Willsch, F. Jin, T. Lippert, and H. D. Raedt, "Benchmarking gate-based quantum computers," *Computer Physics Communications*, vol. 220, pp. 44 – 55, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0010465517301935

## Appendix

In this section, we show how we achieved the *CCCZ* gate decomposition shown in Fig. 4. The first thing to consider is the topology of the IBM Q 5 Tenerife chip. As shown in Fig. 5, *CX* gate can be put in 6 ways on IBM Q 5 Tenerife. Second, *CCCZ* can be decomposed into the circuit shown in Fig. 8. Note that we used two *SWAP* gates (see Fig. 7) because of the topology. Then, *CCX* gate can be decomposed as shown in Fig. 9. Note that we also need to decompose the last two *CX* gates in Fig. 9 with the target on the fourth qubit and control on the fifth qubit because of the topology (see Fig. 6). Lastly, we need to decompose $CZ^{-\frac{1}{2}}(=CS^{\dagger})$, $CZ^{\frac{1}{2}}(=CS)$, $CZ^{\frac{1}{4}}(=CT)$, and $CZ^{\frac{1}{4}}(=CT^{\dagger})$ gate. This can be done as shown in Fig. 10. That's how we achieved the *CCCZ* gate decomposition shown in Fig. 4.

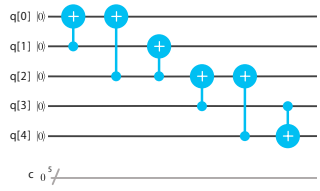


Fig. 5: Topology of IBM Q 5 Tenerife as of October 3rd, 2018. There are six ways to place a *CX* gate as described above.

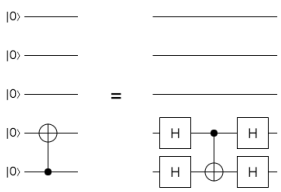


Fig. 6: Decomposition of *CX* gate [17].

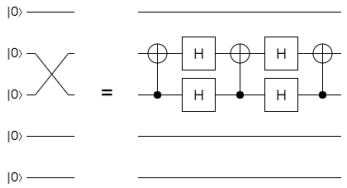


Fig. 7: Decomposition of *SWAP* gate [17].

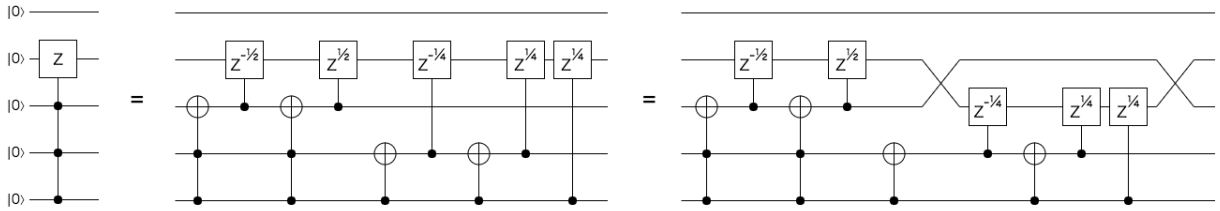


Fig. 8: Decomposition of *CCCZ* gate [19]. We use *SWAP* since *CX* gates with the target on qubits 2 and 3 and control on qubit 5 cannot be placed.

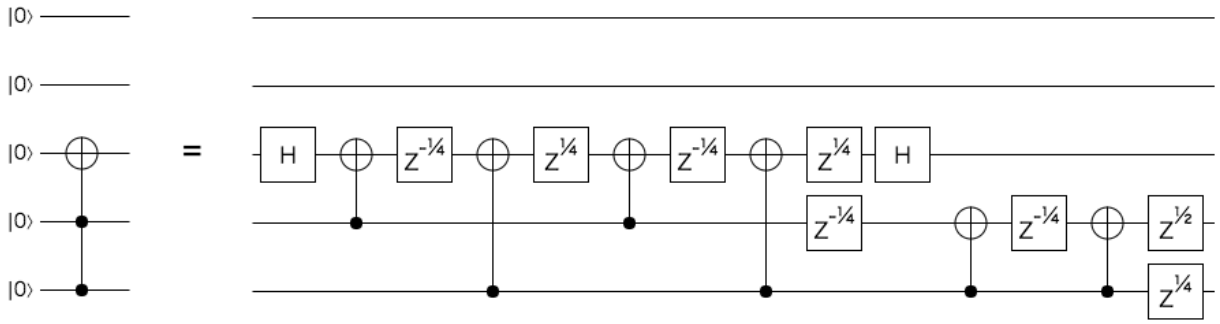


Fig. 9: Decomposition of *CCX* gate [17]. The last two *CX* gates in Fig. 4 with the target on qubit 4 and control on qubit 5 also needed to be decomposed.

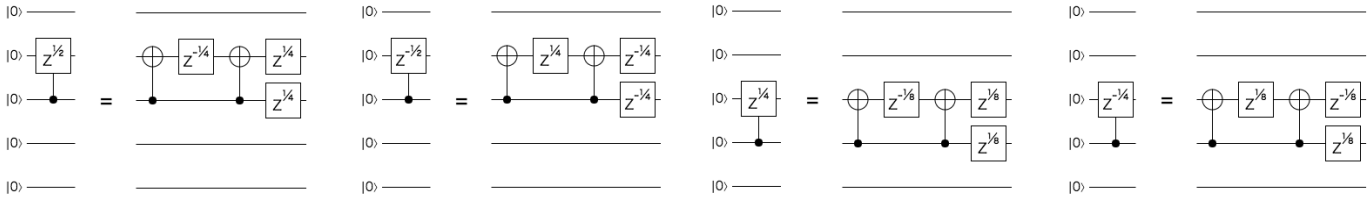


Fig. 10: Decomposition of $CZ^{\frac{1}{2}}(=CS)$, $CZ^{-\frac{1}{2}}(=CS^{\dagger})$, $CZ^{\frac{1}{4}}(=CT)$, and $CZ^{\frac{1}{4}}(=CT\dagger)$ gate [19].