Performance Analysis of the IBM Cloud Quantum Computing Lab against MacBook Pro 2019

1st Alvaro Martin Grande
Dept. of Math and Computer Science
Augustana College
Rock Island, United States
alvaromartingrande20@augustana.edu

2nd Rodrigo Ayala
Dept. of Math and Computer Science
Augustana College
Rock Island, USA
rayalaguerrero18@augustana.edu

3rd Izan Khan
Dept. of Math and Computer Science
Augustana College
Rock Island, United States
izankhan18@augustana.edu

4th Prajwal Sarkar
Dept. of Math and Computer Science
Augustana College
Rock Island, United States
prajwalsarkar18@augustana.edu

5th Tauheed Khan Mohd
Dept. of Math and Computer Science
Augustana College
Rock Island, United States
tauheedkhanmohd@augustana.edu

***Abstract*—This paper covers an introduction to the Quantum Computer World and explains its basic concepts and ideas to understand the principles of this technology. In addition, it addresses and explains some of its most tangible applications nowadays and points out how these ones are going to be developed and implemented in the near future. Moreover, this paper reflects the experimental performance of the IBM's Quantum Computer Cloud Lab, an environment designed to interact with IBM's Quantum Computer using the Jupyter Notebook interface and Python. The results obtained for these experiments are not as expected and therefore, the daily use of this software is not recommend since better run-times can be obtained using regular personal computers.**

***Index Terms*—IBM Quantum Lab, Quantum Computing, Quantum Computing Performance, Introduction to Quantum Computing.**

I. Introduction

Quantum computing opens new realms to technology in ways that we had never imagined before. This new world is based on the foundations of Quantum Mechanics, and our understanding is crucial to develop these new types of machines. Quantum Computation terminates Moore's Law (computer power doubles every eighteen months, valid since 1965) [1] since we cannot use silicon anymore to develop better classical machines [2]. There exists an ultimate limit, dictated by the Laws of Thermodynamics and Quantum Physics which was predicted by Moore. This fundamental limit lies on the size of electronic microprocessors and this is dictated by "the speed of light and atomic nature of matter", Stephen Hawking.

A quantum computer is not an enhanced classical supercomputer which its functioning is based on bits, but a completely new different type of computer which uses the super positioning of bits, called qubits. Using the double nature of the spin of an electron, it allows to perform certain types of computations more efficiently. It is fundamental to understand these computers do not follow the laws of Classical Mechanics because these ones are not able to describe and to understand the quantum world. Richard Feynman, at the MIT Physics of Computation Conference in 1981, stated: "Nature isn't classical... and if you want to make a simulation of Nature, you'd better make it quantum mechanical" [3].

These machines use the principles of the Quantum theory. They can accurately compute reality, and they will eventually let us travel beyond the classical computer limits. They can perform calculations in parallel, in millions of different parallel universes of probability and make them revel a result when it is obtained in one of them. This works through the use of qubits, which besides super positioning each other they interact with each other at the same time. As stated before, they are not classical bits. A bit may either be 0 or 1, although a qubit, or quantum bit, may be 0, 1, both at the same time, or neither of them. These concepts and their functioning will be explained in the next sections.

Quantum Computing has a wide variety of uses and applications such as the creation of enhanced machine learning algorithms, the improvement and training of neuronal networks, the calculation and simulation of cancer propagation, the calculation and replication of molecular reactions, the ability to quantize cybersecurity and blockchain algorithms, the improvement of financial modeling or the rapid creation of solutions for unsolved mathematical models, among others [4].

II. Related Work

Research about the impact of quantum computing technology on future developments addresses the current information about quantum computers and how the used of these ones can impact other types of scientific developments. Nowadays, it is an important field in which many high-technological companies such as IBM, Google or Intel are investing a significant amount of money to develop this technology. According to the authors, at this time one of the main motivation for certain companies to develop quantum computers is to

break public key cryptography with the scheme of RSA which is a type of encryption to protect almost any data such as text messages, connection between cell phones and the Internet. Regarding the plans for the future, this paper mentions the need of researchers to develop quantum computers with lower operating temperatures so one day these machines may become frequently used cloud service computers. Quantum computing is also mostly used for scientific purposes regarding computer science, physics, biology, medicine, and engineering. In addition, it is important to highlight the limitations and the work that still need to be done in order to discover more properties of quantum computing. There is a need of a standardized programming language and a compiler in order to organize and structure the way in which computer scientists write code to communicate with the quantum processor. It could be said that we are the beginning of a new era of computation, and it could be comparable to the beginning of classical computer development at the end of the 20th century [5].

Developing, programming and constructing a quantum computer is an extremely complex task. Quantum Physics is a field of study in science which has been recently developed. In addition, nowadays our understanding about it is short, and its math is complex. The Quantum world does not behave as the world we can see, and Classical Physics is able to explain. This makes quantum computers difficult to build and understand. Nevertheless, the process of developing a quantum computer is similar to a classical one. Processors, memory, disk, compilers, and machine language are needed to develop these machines. The compiler and other high-level elements run on a standard computing system. The machine language instructions are translated via a digital control unit to analog control signals, like the voltage pulses and microwave bursts, which are sent to the qubits via conventional transmission lines. The signals for qubit control are generated using conventional electronics. Although, we define a qubit as a microwave function [**?**], and it can be written as a linear combination of states (0 or 1, ON or OFF), using the next quantum function (Equation 1):

$$|\psi >= \alpha|0 > +\beta|1 >$$

As it can be observed, a qubit is strictly described as a wave. The equation encloses the classical states of a bit, 0 or 1, but they are accompanied by the complex numbers “a” and “b”. This quantum characteristic will allow a qubit to behave as a linear combination of both states of bit. It is able to be either 0 or 1, both at the same time, or maybe neither of them. Note these qubit values are converted to single bit values, for a qubit in 0 or 1. This could make obvious that a classical system approach is required to successfully design and operate a quantum computer, within which the particular qubits are only one part of the entire system, and therefore the bulk of the system is classical. To urge a way of the number of qubits needed to outperform classical technology, we note that storing the state of just 50 quantum bits already requires a memory larger than one of the today’s most powerful (classical) supercomputers. Every additional qubit doubles the memory (and the processing power) that a classical supercomputer requires to calculate the behavior of the qubits. This enables specific computational tasks that are beyond the facility of supercomputers, like simulating quantum chaotic evaluations or cancer propagation, to be solved using 50-qubit systems. The superposition of quantum qubits opens the possibility to make extremely complex probability calculations based on the Quantum Theory.

Moreover, we encounter the concept of Trapped nuclear particles that give one of the main actual stages for carrying out a completely practical quantum computer [6], here a programmable quantum computer model was illustrated [7], and its presentation was contrasted to a superconducting quantum computer [8]. Trapped particles, single molecules with an electron eliminated (consequently, nuclear particles) are utilized to address the qubit, by choosing two interior conditions of the iota. Such particles can be caught, regularly looking like a straight chain, in a design that gives satisfactory electromagnetic fields to bind the particles in an ultrahigh vacuum climate. Current particle traps are made on silicon substrates utilizing micro fabrication innovation [9].

In addition to this, recently quantum reenactments have further been acquiring consideration with regards to significant uses of quantum processing. First noted by Feynman [10] and Manin, a quantum computer is required to be especially appropriate for reenacting different quantum mechanical marvels, similar to how an old style computer is valuable for reenacting different traditional mechanical wonders. A large group of fascinating issues that are not in any case realized how to address proficiently with traditional computers might be effectively tackled with quantum reenactments performed on a quantum computer. A few examples include, in increasing order of difficulty and impact, this follows: 1) more profound comprehension of many body physical science and emphatically corresponded matter, with potential applications in the plan of room-temperature superconductors or materials with good electrical properties; 2) highly-accurate chemistry computations for growing new impetuses and synthetic cycles, with significant applications, for example, finding a swap for the Haber process for nitrogen fixation (NF) utilized in the synthesis of fertilizers; 3) large scale, exceptionally precise sub-atomic elements reproductions to contemplate issues in protein folding and drug design.

III. Theory and Encryption

1) Quantum Computing Performance Theory: The principal goal of this paper is to analyze and compare the performance that a quantum computer can have compared to a classical computer. The idea of quantum computers was introduced by thinking that it could be possible to reduce the time complexity of algorithms and then perform faster operations that could transform several science fields such as biology, chemistry, and material research because of the revolutionary computational power. In order to introduce this power to the world, researchers have been developing

Quantum Processing Units (QPU) that could be implemented in a High Performance Computer to access the quantum computing technology. However some of the features that should be incorporated in those QPU's have very specific settings and more research is needed to come up with a proper configuration [11]. Indeed, the important question and concern is to figure out how much faster a quantum computer can be and if it is worth to spend so many resources in the expensive development process. The way in which it is proven that a quantum computer has a superior processing power and that it can achieve tasks that a classical computer cannot is with quantum supremacy. In order to achieve quantum supremacy, testing is needed with a given circuit to run it in the quantum computer. Afterwards, certain simulations have to be created for the classical computer to emulate and replicate what has been done in the high performance computer. Moreover, the the complexity of the circuit should increase exponentially until the implementation for the classical computer cannot longer be performed by the machine. Consequently, we would reach the state of quantum supremacy where the quantum computer is the only one capable of carrying and executing the given task.

2) Encryption: The question of whether quantum computers are powerful enough to break encryption algorithms is one of the main concerns for researchers, as it was mentioned previously. Cybersecurity is now required for the majority of data that individuals like to share through the internet. Currently, everything can be done in the internet by just by clicking a button, from money transactions, purchases, to access to sensitive private information. In order to keep privacy, and therefore this information safe, there exists encryption. This process consists on encoding information by transforming the data to cipher text. Only authorized parties can decipher this text, to then be able to access the desired information. The authorized parties will need to have a key or gain access to the decipher algorithm in order to obtain the information. That being said, the ability for quantum computers to be able to decipher, and cipher text could breakdown the current algorithms for encryption and the most commonly used methods of encryption to secure our data such as RSA and Elliptic Curve Cryptosystems. In fact these two types of public key algorithms can be easily interrupted with Grover's and Shor's algorithms if they are performed by a fully operative quantum computer [12].

Grover's algorithm is giving an advantage for quantum computers to find the public key in a database. The strength of this algorithm is encountered in the efficiency that this technique has to search anything in a database that could contain classified files and encrypted data. Because of this, sophisticated encryption systems such as AES (Advanced Encryption Standard) could be negatively affected due to the performance of the quantum computer debilitating the security system [13].

Shor's algorithm has been used in quantum computers to perform integer factorization and to find the prime factors of a given integer N. In order to access encrypted data, we should be able to get and compute the factors of N. N is the number by which the data is being replaced when encrypted. On the contrary, the process of finding the prime factors of a large number could take years for a regular computer. However, the power that quantum computers give to the Shor's algorithm is extremely useful since it generates the processing speed that the algorithm needs in order to find the factors very rapidly. Since quantum computers can perform operations in qubits and superposition, it can reduce the time complexity of any algorithm. Some researchers have been trying to discover methods to implement Shor's algorithm properly with designs of a practical quantum computer. Some implementations have been made so far by designing circuits and logic gates such as an architecture with a linear nearest neighbour qubit array [14].

The age of practical quantum computers has been heralded by technological breakthroughs such as the Transmon cryogenic 5-qubit machines [15]. Researchers all over the world are working to perfect the mass production of multi-qubit devices so that large-scale quantum computers with millions or billions of qubits [16] can be built. This would be needed to solve real-world problems. On the other hand, It's also critical to develop a quantum ecosystem that includes a standardized quantum programming language [17], compilers, and debuggers [18], as well as a quantum hardware abstraction layer that enables a single quantum program to be compiled for multiple target quantum hardware platforms, as it is usually done with classical computers. Furthermore, since all qubit technologies available today are very fragile and vulnerable to errors, quantum computers will need extra effort to detect and correct these errors.

IV. Experimental Setup

Throughout our research we have learnt high performance and speed are two of the most important characteristics of a quantum computers. We are aware that there are many factors which condition performance when running certain scripts or algorithms such as available memory, disk, processor or even internet connection speed. Nevertheless, we understand it would be useful to test certain algorithms, techniques and tasks using a quantum computer to actually analyze how it can be operated by regular users. As we are not able to directly access a quantum computer, we will be using an online tool created and developed by IBM which connects to an actual quantum computer. This tool is named "IBM Quantum Lab", and it uses "Jupyter notebooks" with Python. It is easy to access from the internet and it can be used from any personal computer running any operating system. Our code will be implemented in Python and, three different tests using the "timeit" module will be tested. This module is a method which implements software profiling in order to measure the performance of a given piece of code. This performance can measure the resources used by the CPU and the memory, frequency or duration of the function calls or wall clock execution time for a piece of code [19]. Our idea is to utilize this module to compare the performance of a regular personal computer versus the IBM's cloud Quantum Computer. The computer used throughout the experiment was

a MacBook Pro 2019 mounting a 1.4GHz Quad-Core Intel Core i5 processor, and 8 GB 2133 MHz LPDDR3 of RAM Memory. As Jupyter Notebooks is used to test our code in the Quantum Computer, Anaconda and Jupyter were used on our local machine, as well.

It is important to highlight why we use this method. A primitive version of our experiment was tested just using the module "time". This module is extensively known along python users since it allows the programmer to measure how long a computer takes to run a certain script in python. It is implemented as a stopwatch. It is initialized at the beginning of the scrip and then, it is stopped at the end of this one [19]. This method is very convenient to test certain codes or applications in a specific environment. Although, this method is not completely accurate since it will vary on the machine and many different factors such as memory or CPU. This method will not implement any performance calculations but a time measurement. Therefore, "timeit" is used since it tests hypotheses about efficiency of algorithms and Python idioms. In addition, by default, "timeit" will test each function 100,000 times using the "lambda" function defined by this module. This number of times is the default testing procedure, but it can be easily modified by using different methods provided by the module creation which can be found over its documentation website. Throughout our experiment this factor will not be modified, and therefore the experiment will be conducted using this default parameter.

A. Massive Random Numbers Generation

In order to start testing the IBM Cloud Quantum Computer, an script was designed to generate decimal random numbers from 0 to 1, using the "random" module in python. This piece of code was set into a loop, and it will be looped as many times as the user decides it. Then, using a variable which can be modified by the user, it can be set how many times the previous loop and the outter one are going to be run. As an example, if the user enters 100, the loop will execute 100*100. Afterwards, the "timeit" module is added, and the performance of the code is measured an average of 100,000 times (by default using "timeit"). This increases the difficulty of the calculations significantly. The next results were found (Table 1) after performing these measurements different times, and they will be discussed in the next section.

TABLE I
MASSIVE RANDOM NUMBER GENERATOR TESTING COMPARISON.

Rand. Numbers	IBM's Quantum (s)	Mac Pro 2019 (s)
10	1.019	0.943
100	7.473	6.476
1000	81.259	72.277
10000	820.639	744.823

It is important to highlight the difficulty found to perform certain types of computations in the IBM computer. It was impossible to overpass the 100,000 times.

B. Classic Algorithms Performance

Secondly, a similar process is followed to calculate the efficiency of two of the most well-known algorithms in the computer science field: Linear Search and Binary Search. Linear Search is frequently used to locate a target value in certain types of data structures, usually arrays or lists, depending on the programming language used. This method examines each of the elements, from the first one to the last one. This algorithm has a complexity of O(n+1/2) [20]. On the other hand, Binary Search is a frequently used algorithm which locates a target value in an array or a list by successfully eliminating half of the structure from consideration. This algorithm has a complexity of O(logN) [21]. Then, using the "timeit" module, these searching techniques were tested with random data structures. As we can see, the complexity of a Binary Search is lower than a Linear Search and we expect to find faster run-times for this one. Each of the methods were tested ten times in each machine and the average of both was calculated (Figure 2):

TABLE II
CLASSICAL ALGORITHM PERFORMANCE.

Algorithm Name	IBM's Quantum (s)	Mac Pro 2019 (s)
Linear Search	3.87	4.48
Binary Search	2.87	3.50

C. For-loop concatenation

Finally, a script is designed to run 100 times a for-loop. Then, this for-loop is called as many times as the user decides. For our experiment, we went from 10 to 10000 times. Again, the "timeit" module is used to accurately calculate the performance of the code. These are the results obtained (Table 3):

TABLE III
MASSIVE FOR-LOOP CONCATENATION.

Times Looped	IBM's Quantum (s)	Mac Pro 2019 (s)
10	11.675	12.524
100	115.437	119.936
1000	1238.029	1216.072

V. ANALYSIS

It was not expected to find such a wide variety of results. The analysis will be divided into three different sections to discuss about each of the experiments performed.

A. Massive Random Numbers Generation

As it was explained before, the "timeit" module was used to measure the performance of our code when generating a vast amount of random numbers from 0 to 1 using concatenated for-loops. It can clearly be observed on Table 1 that our computer performs better run-times than the IBM Cloud Quantum Computer. This was very interesting since we expected the cloud computer to obtain clearly better results. On the next figure, it can be observed a graph of the time needed to

performed each of the tasks VS the amount of random numbers to be generated (Figure 1):

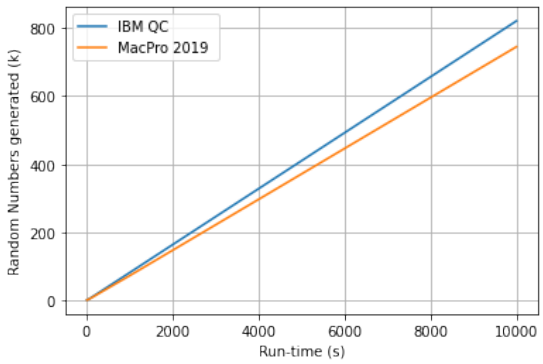

Fig. 1. Massive Random Number Generator Test (0,1) - MB Pro 2019 VS IBM Cloud Quantum Computer

Both computers obtained similar performance when generating random numbers from 10 to 1,000 times. Then, when generating numbers for 10,000 times the IBM computer clearly slowed down and it was beaten by the MacBook Pro. It is important to highlight neither of both computers were able to generate numbers for more than 10,000. In addition, it is important to point out the tendency found for the computer to carry out these tasks was linear. This means we could be able to predict the run-time of certain amount of random numbers calculating the slope of the lines shown on Figure 2.

B. Classic Algorithms Performance

The results obtained for this part of the experiment verifies the complexity of the tested algorithms. We can observe that a Binary Search is performed faster than a Linear Search. This is caused due to the structure of each of the techniques. The Binary Search does not loop through all the structure but just half of it, so it has a complexity of O(logN). The Linear Search algorithm has a complexity of O(n+1/2), and therefore slower run-times were expected. Our experiment concludes, in this case, that the Quantum Computer averagely performs faster searches than our computer. It is able to perform binary searches in less than 2.87s.

C. For-loop concatenation

Some of the results obtained for the for-loop concatenation experiment were expected. We expected to encounter longer run-times than the ones calculated for the random number generator experiment. Nevertheless, we expected to find a non-linear tendency (Figure 2) since we are repeatedly using for-loops which would alter the complexity of the code. It was expected to found quadratic curves, or higher order tendencies. We attribute our error to the low number of trials which could be performed using both computers. At the end of this experiment, we could just compare three different data points, as it can be observed on Table 3:

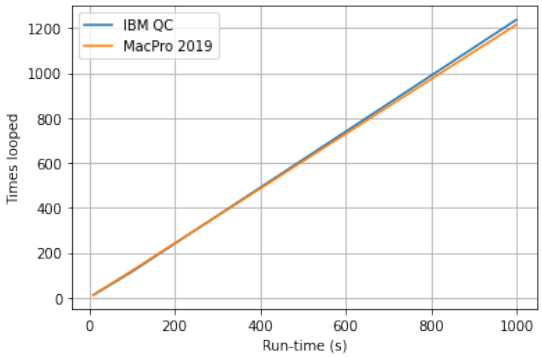

Fig. 2. For-loop Concatenation Test- MB Pro 2019 VS IBM Cloud Quantum Computer

VI. Conclusions

The results obtained throughout the experiment phase did not satisfy our expectations. We expected to find extremely low run-times using the IBM Cloud Quantum Computer. Obtaining more data would dramatically improve the results of our experiment. We propose performance of random number generation vs. run-time, as well as times-looped vs run-time could be predicted by calculating the slope produced by the data gathered. This would proof the linearly behavior of the Quantum Lab to perform the required tasks during our experiment. In addition, this could eventually point out the computational power available for the user by calculating how many tasks can be performed in a specific period of time. Although, we do not have the resources to test if this applies to longer random number generation or loop concatenation processes.

We understand the processing power of a Quantum Computer is extremely higher than a classical one, as it was explained throughout this paper. This is the reason why we cannot attribute the error to the Quantum Computer but to the interface used to interact with it. We believe the IBM's Cloud Service Lab environment does not actually use all the computational power available to run code. This service uses a modified version of Jupyter notebooks which we believe it is stable. IBM is limiting the power a user can utilize. Furthermore, we ignore if the fluctuations on our internet connection or the browser used are elements which would alter the performance of the Quantum Lab environment since it is run online on IBM's cloud. Moreover, we encounter a wide variety of different run-times when performing similar tasks using the Quantum Lab. In addition, while testing our code we found the environment was not working many times, or it had many difficulties to run extremely simple pieces of code. We do not find reliable the IBM environment to use it as a daily coding tool or online compiler. Nevertheless, we extremely recommend it to perform more specific tasks dues to its versatility.

As a conclusion, we doubt the actual usefulness of the IBM Quantum Lab since better run-times can be obtained using regular desktop computer just running Jupyter on Anaconda.

On our future work, we expect to address the issues of the IBM Quantum Lab environment and understand why the processing power is so limited.

REFERENCES

[1] J. R. Powell, "The quantum limit to moore's law," *Proceedings of the IEEE*, vol. 96, no. 8, pp. 1247–1248, 2008.

[2] L. A. S. Laboratory and L. A. N. Laboratory, *Los Alamos Science*. No. 27-28, The Laboratory, 2002.

[3] M. Steffen, D. P. DiVincenzo, J. M. Chow, T. N. Theis, and M. B. Ketchen, "Quantum computing: An ibm perspective," *IBM Journal of Research and Development*, vol. 55, no. 5, pp. 13–1, 2011.

[4] A. Cho, "Google claims quantum computing milestone," 2019.

[5] M. Möller and C. Vuik, "On the impact of quantum computing technology on future developments in high-performance scientific computing," *Ethics and Information Technology*, vol. 19, no. 4, pp. 253–269, 2017.

[6] C. Monroe and J. Kim, "Scaling the ion trap quantum processor," *Science*, vol. 339, no. 6124, pp. 1164–1169, 2013.

[7] S. Debnath, N. M. Linke, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, "Demonstration of a small programmable quantum computer with atomic qubits," *Nature*, vol. 536, no. 7614, pp. 63–66, 2016.

[8] N. M. Linke, D. Maslov, M. Roetteler, S. Debnath, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, "Experimental comparison of two quantum computing architectures," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3305–3310, 2017.

[9] J. T. Merrill, C. Volin, D. Landgren, J. M. Amini, K. Wright, S. C. Doret, C. Pai, H. Hayden, T. Killian, D. Faircloth, *et al.*, "Demonstration of integrated microscale optics in surface-electrode ion traps," *New Journal of Physics*, vol. 13, no. 10, p. 103005, 2011.

[10] R. P. Feynman, "Simulating physics with computers," *Int. J. Theor. Phys*, vol. 21, no. 6/7, 1982.

[11] K. A. Britt and T. S. Humble, "High-performance computing with quantum processing units," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 13, no. 3, pp. 1–13, 2017.

[12] V. Mavroeidis, K. Vishi, M. D. Zych, and A. Jøsang, "The impact of quantum computing on present cryptography," *arXiv preprint arXiv:1804.00200*, 2018.

[13] Z. Sakhi, R. Kabil, A. Tragha, and M. Bennai, "Quantum cryptography based on grover's algorithm," in *Second International Conference on the Innovative Computing Technology (INTECH 2012)*, pp. 33–37, IEEE, 2012.

[14] A. G. Fowler, S. J. Devitt, and L. C. Hollenberg, "Implementation of shor's algorithm on a linear nearest neighbour qubit array," *arXiv preprint quant-ph/0402196*, 2004.

[15] X. Fu, L. Riesebos, L. Lao, C. G. Almudever, F. Sebastiano, R. Versluis, E. Charbon, and K. Bertels, "A heterogeneous quantum computer architecture," in *Proceedings of the ACM International Conference on Computing Frontiers*, pp. 323–330, 2016.

[16] B. Lekitsch, S. Weidt, A. G. Fowler, K. Mølmer, S. J. Devitt, C. Wunderlich, and W. K. Hensinger, "Blueprint for a microwave trapped ion quantum computer," *Science Advances*, vol. 3, no. 2, p. e1601540, 2017.

[17] S. Balensiefer, L. Kregor-Stickles, and M. Oskin, "An evaluation framework and instruction set architecture for ion-trap based quantum micro-architectures," in *32nd International Symposium on Computer Architecture (ISCA'05)*, pp. 186–196, IEEE, 2005.

[18] A. JavadiAbhari, S. Patil, D. Kudrow, J. Heckey, A. Lvov, F. T. Chong, and M. Martonosi, "Scaffcc: Scalable compilation and analysis of quantum programs," *Parallel Computing*, vol. 45, pp. 2–17, 2015.

[19] M. M. Christopher Barker, Joseph Sheedy, "Performance and profiling," 2015.

[20] S. Kamara and T. Moataz, "Boolean searchable symmetric encryption with worst-case sub-linear complexity," in *Advances in Cryptology – EUROCRYPT 2017* (J.-S. Coron and J. B. Nielsen, eds.), (Cham), pp. 94–124, Springer International Publishing, 2017.

[21] R. E. Tarjan, "Amortized computational complexity," vol. 6, pp. 306–318, 1985.