Investigation into the Visualization of Quantum Computing Algorithms

Vaibhav Santurkar
Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, USA
e-mail: vs4503@cs.rit.edu

Hans-Peter Bischof
Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, USA
e-mail: hpb@cs.rit.edu

***Abstract*—The objective of this paper is to design a visualization tool that can efficiently describe the execution of complex quantum algorithms. Through prevailing methods, the execution of many classical algorithms can be easily visualized. For example, sorting algorithms utilize histograms to illustrate the sorting operation in a step-by-step manner. However, a similar technique does not exist for quantum algorithms, as current visualization tools apply intricate quantum principles, which are difficult to comprehend. Therefore, we developed a visualization tool that can demonstrate the execution of said algorithms through simpler methods. Our tool was capable of visualizing the execution of popular quantum algorithms, such as Shor's and Grover's, despite the complexity of the concepts involved.**

***Index Terms*—quantum computing; quantum visualization; Bloch spheres; Shor's algorithm; Grover's algorithm.**

I. Introduction

Quantum computing is a swiftly developing technology, that is now changing the way computer scientists look at classical algorithms. By exploiting the properties of quantum mechanics, this field has begun to yield simple solutions to once-complicated classical problems. Qubits along with superposition, entanglement, and interference are merged to form these solutions. An exponential speed-up in terms of execution is possible for these problems when quantum computing is utilized. While true quantum hardware is still a thing of the future, current computers can simulate their quantum counterparts. Computer scientists can use these simulators to discover solutions that are too complex for conventional hardware to solve in polynomial time.

With the development of IBM's quantum hardware making progress in the world of computer science, many researchers are now developing quantum algorithms and applications [1] [2]. For example, the concept of entanglement has been integrated with classifier algorithms in ML (machine learning), creating a new subset in the field known as quantum machine learning algorithms [3]. Qubits along with superposition, are being used to create evolutionary algorithms for search and optimization methods. These methods operate on the 'survival of the fittest' principle to select the best results before proceeding further [4]. Quantum key distribution protocols for symmetric encryption algorithms is another promising application of quantum computing. Quantum gates are applied sequentially to increase the capacity for transmission of data between a sender and receiver and hence improve communications [5].

The visualization of numerous classical algorithms is done through simple plots, e.g., bar graphs, scatter plots, histograms, etc [19]. As the algorithm executes, the plots can be manipulated to illustrate how the final result is obtained. However, to visualize qubits and quantum states, specific tools which could encapsulate quantum theory had to be built. Researchers have applied these tools, namely, Bloch spheres, quantum circuits, and decision diagrams to visualize popular quantum concepts, i.e., Shor's algorithm and Bell states [6] [7]. Despite these tools being popular amongst experts, users unacquainted with quantum computing required a simpler tool. Therefore, the framework proposed in this paper is designed to allow users to effortlessly recognize and explore the behavior of quantum algorithms.

The paper is organized in the following manner. Section II covers the basic quantum concepts and formulae. Section III describes quantum visualization tools and the proposed visualization framework. Section IV demonstrates the application of the framework on popular quantum algorithms and performs an analysis on the results of the framework. The conclusion and acknowledgment close the paper.

II. Basic Quantum Concepts

The complex quantum algorithms currently being employed today are built using the core quantum principles listed below:

- Superposition
- Qubits
- Entanglement
- Interference

Advanced quantum algorithms and applications are designed using the strong foundation established by these integral concepts.

A. Superposition

Superposition can be defined as an essential component of quantum computing. Consider a system that exists with a certain number of mutually exclusive classical states. A quantum state, in that system, is said to be in superposition, when it is in all classical states at the same time [8]. Each classical

 ISBN: ISBNFILL

state in this system has a specific amplitude associated with it, giving the equation:

$$|\gamma >= \alpha_0|0 > +\alpha_1|1 > +... + \alpha_{n-1}|n-1 > \quad (1)$$

where γ is the pure quantum state, α_n represents the amplitude of the nth state, and n represents the total number of classical states. The quantum state γ can be thought of as a column vector in an n-dimensional vector space with an inner product, i.e., a Hilbert space [8] [9].

The concept of superposition is applied in quantum measurement. By the nature of superposition, a quantum state exists between multiple classical states at any given moment. Therefore, when a quantum state is measured, it collapses into a single classical state, which can then be translated or mapped into classical bits. Additionally, any information related to the collapsed states is lost, and only the resultant state's information is preserved. To accomplish this, a probability distribution formula is applied across all involved states to determine which state the others will collapse into. All of the probability values must sum up to 1, giving the equation:

$$\sum_{j=0}^{N-1} |\alpha_j|^2 = 1 \quad (2)$$

This is also known as the simplest form of Born's rule [8] [10].

The visualization of superposition in certain single-qubit environments is possible. When qubits in a Hilbert space are translated into a three-dimensional space, they are represented as Bloch spheres [9] [14]. Consider a system with the basis states $|0 >$ and $|1 >$. If certain mathematical transformations are applied to a qubit in this system, it is said to be in a superposition of the basis states, which can be seen in Figure 1. In Figure 1, the magenta arrow represents the qubit, the north and south poles of the sphere are the basis states, while x and y are axes labels from the two-dimensional space. As seen in Figure 1 the qubit's orientation is precisely midway between the basis states. This implies that the qubit exists in both states at the same time, and hence illustrates superposition.

B. Entanglement

When 2 qubits are correlated to each other in a quantum sense, it is known as quantum entanglement. To demonstrate this property, the EPR (Einstein, Podolsky, Rosen) pair of qubit states are used [11], which are of the form:

$$\frac{1}{\sqrt{2}}|00 > +\frac{1}{\sqrt{2}}|11 > \quad (3)$$

To begin with, before measurement, both qubit states are said to be in a superposition between the basis states $|0 >$ and $|1 >$. Once the measurement operation is applied to the 2-qubit system, the unique properties of the EPR pair can be perceived. Suppose, after measuring the first qubit state, the result obtained was the classical state '0'. Then by the property of entanglement, both states collapse into $|00 >$. Similarly, if the result was the classical state '1' then both states would collapse into $|11 >$. Therefore, by measuring the first qubit, the second unmeasured qubit immediately collapses into the same classical value.

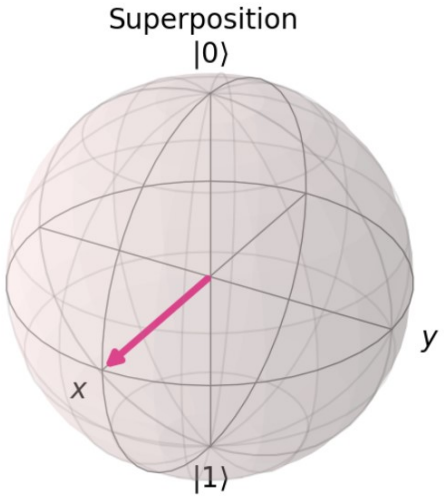

Fig. 1. Bloch sphere depicting superposition. [20]

Visualizing the concept of quantum entanglement is relatively straightforward. Qubits in a Qsphere are represented by multicolored dots, with the following properties: 1) The size of the dot represents the probability of the qubit being in that state 2) The color of the dot represents the phase angle that the qubit vector makes with the axes and 3) The line indicates the qubit states are correlated with each other. As can be seen in Figure 2, the 2-qubit system has been visualized [12]. The dots are of equal size and both share a deep purple color. This implies that there is an equal probability of being in either state with a phase angle of 45 degrees ($\frac{\pi}{4}$), and the line connecting the dots indicates they are entangled.

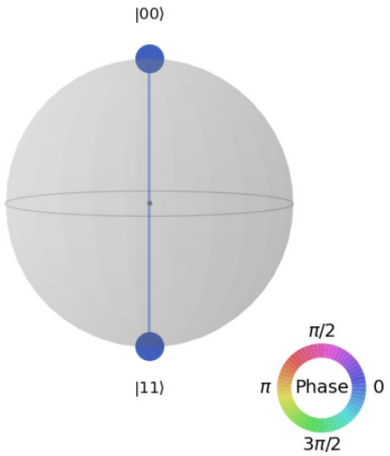

Fig. 2. Qsphere depicting entanglement. [20]

C. Interference

In wave patterns, when 2 waves overlap each other, they can either combine to create a larger wave, or cancel each

other out. This is known as interference and can be extended to quantum mechanics as well. To illustrate this, the Hadamard transformation is used. The Hadamard transformation is a single qubit quantum gate, that sets a qubit in a superposition between the basis states $|0>$ and $|1>$ [8] [13]. The transformation is represented as a matrix of the form:

$$\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix} \tag{4}$$

When the Hadamard transformation is applied to the basis state $|0>$, the following state is generated:

$$\frac{1}{\sqrt{2}}|0> + \frac{1}{\sqrt{2}}|1> \tag{5}$$

And similarly, if applied to the basis state $|1>$, the result obtained is:

$$\frac{1}{\sqrt{2}}|0> - \frac{1}{\sqrt{2}}|1> \tag{6}$$

For interference, the first step is to apply the Hadamard transformation to the basis state $|0>$. Next, the transformation is applied to the result of the previous operation to obtain the basis state $|0>$. This implies that the amplitudes for the basis state $|1>$ had the same value but opposite sign. Therefore, the amplitudes canceled each other out, leaving $|0>$ behind.

Bloch spheres are used to visualize the concept of interference [14]. First, the qubit is set in a superposition, as seen in Figure 1. After the Hadamard transformation is applied to the superposition qubit, the state reverts back to $|0>$, as can be seen in Figure 3. The magenta arrow, which represents the qubit vector, points at the north pole of the sphere, indicating the qubit is now in state $|0>$.

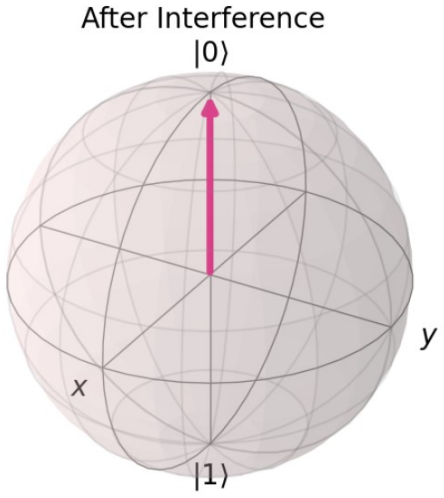

Fig. 3. Bloch sphere depicting interference. [20]

III. Quantum Visualization Tools

Existing visualization tools can be applied to specific quantum environments based on a number of criteria, largely by the number of qubits involved. In the case of single-qubit environments, Bloch spheres and Qspheres can be used to visualize the quantum operation [12] [14]. One tool for multi-qubit environments would be quantum circuits and gates. However, quantum circuit diagrams are an example of a high-level visualization tool, i.e., it is tough to explain a quantum circuit to an observer who is unaware of basic quantum theory.

Therefore, the proposed solution is a visualization framework, that uses Microsoft Quantum functions [15]. Using functions from the diagnostic library, the information related to the active qubits in the program can be dumped onto the console. This information can then be collected into a JSON file, and a bar graph can be constructed from this data. These bar graphs can visually elucidate the execution of any multi-qubit algorithm.

A. Bloch Spheres

A Bloch sphere is a visualization tool, that transforms a two-dimensional Hilbert space into a three-dimensional representation. To accomplish this, the basic representation of a qubit state is used, along with the relative phase variable ($e^{i\phi}$):

$$\alpha_0|0> + e^{i\phi}\alpha_1|1> \tag{7}$$

This state formula is normalized, and the resulting equation is compared with certain trigonometric identities, to give the following form of the qubit state:

$$\cos\frac{\theta}{2}\alpha_0|0> + \sin\frac{\theta}{2}e^{i\phi}\alpha_1|1> \tag{8}$$

Here, θ and ϕ are the variables used as input for the Bloch sphere function [9] [14]. They represent the angle of the qubit state vector with respect to the basis states.

Bloch spheres are powerful tools for visualizing single-qubit operations, as abstract concepts in the two-dimensional environment, become coherent in the three-dimensional environment. Additionally, Qspheres act as a secondary visualization tool for two or three qubit operations [12]. Here, only the relative phase variable between the qubits can be used to illustrate the positions of the vectors in the sphere.

Therefore, the proposed solution of this paper is to design a framework that can visualize quantum algorithms in such a way that is easier to understand.

B. Proposed Solution

The domain-specific, open-source language used to develop quantum algorithms and applications is Q# (Q Sharp). It was developed in 2017, as a part of Microsoft's Quantum Development Kit [15]. The language draws coding components and mechanisms from C# and Python. Q# supports essential modules such as defining data types, constructing methods, and establishing flow control structures. Additionally, quantum-specific data structures such as Hadamard transformations, and Controlled NOT matrices can also be defined. Unlike quantum circuits, the language uses code statements and expressions to illustrate integral quantum concepts. By doing this, classical programs can be easily mapped to their quantum equivalents, with minimal change to the actual code.

Microsoft Quantum's Diagnostic library is a specific function library, that allows the user to detect mistakes and debug

 ISBN: ISBNFILL

errors in programs [15]. This library contains 3 functions that allow the user to dump the information related to the current qubits involved in the program, to the console. These functions are:

- DumpOperation
- DumpMachine
- DumpRegister

The framework proposed in this paper utilizes the DumpMachine function to visualize the execution of the quantum algorithm. To begin with, the user writes a quantum algorithm that solves a certain problem. Next, the DumpMachine function is called at critical points in the algorithm, and the main method of the program is executed. When the DumpMachine function is called, the data related to the active qubits in the program is collected and dumped onto the console. This data can be aggregated into a single JSON file, and then sorted and extracted into Python's list data structures.

Specifically, the qubit amplitude data and the associated qubit states in the JSON file are extracted into the list structures. From this, a bar graph is plotted with the active qubit states on the x-axis and the amplitude data on the y-axis. The advantage of this framework is that the DumpMachine function can be called multiple times, once for each crucial execution step in the algorithm. This implies that multiple entries will be made in the JSON file, allowing multiple bar graphs to be constructed. Hence, the graphs illustrate the execution of the algorithm, through the changes in the amplitudes and states of the qubits. The final result of the algorithm will be in a certain qubit state with a certain amplitude, making it simpler to draw inferences about the algorithm.

For example, suppose the algorithm being visualized is trying to solve the problem of whether a function is constant or balanced. As seen in Figure 4, the bar graph shows the final state of the qubits after all the operations have been applied. The graph shows the amplitude of the qubits as -1 in the state $|00>$. According to the properties of the algorithm, this implies that the input function to the algorithm was balanced. Similarly, if the amplitude had been -1 in the state $|11>$, this would imply that the function is constant. This example demonstrates how the amplitude of the qubits is related to the execution of the given algorithm.

IV. Quantum Algorithms

The most systematic method to test the proposed framework is to pass a quantum algorithm as input and observe the bar graphs generated as output. This output along with the problem statement can be used to draw conclusions on the execution of the algorithm. For example, if a quantum algorithm had the problem statement, "To find if a function is constant and balanced" and an output bar graph of -1 in the state $|11>$, this would imply that the function is constant.

Similarly, this framework has been applied to 2 popular quantum algorithms, namely, Shor's and Grover's algorithms [16] [17]. In this section, the problem statements of the algorithms are explained, along with the conclusions drawn from the bar graphs generated from the framework.

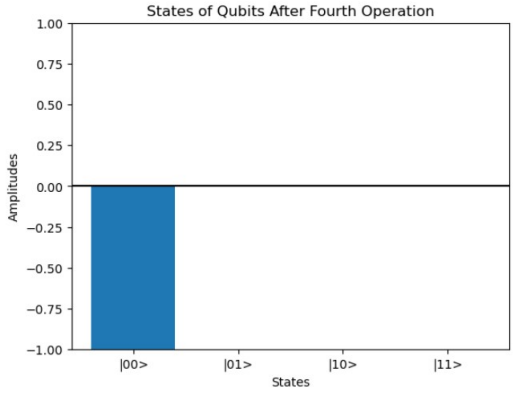

Fig. 4. Final result of proposed framework after simulating quantum algorithm. [20]

A. Shor's Algorithm

Shor's algorithm is a popular quantum algorithm, with applications in cryptography [16]. Classical algorithms take exponential time to solve integer factorization problems, hence Shor's algorithm was proposed to solve the same problems in polynomial time. Since most cryptographic encryption and decryption techniques require integer factorization in some form, Shor's has many powerful applications. As of today, Shor's can break a number of public-key cryptography methods, one of them being the Diffie-Hellman Elliptic Curve key exchange protocol [18]. In the future, with adequate quantum hardware, Shor's algorithm may be able to break highly secure cryptographic schemes, such as AES or SHA-256.

The basic concept behind Shor's algorithm is to find the period of 2 co-prime numbers. In other words, given 2 co-primes, there exists a cycle of remainders when the mod of 1 co-prime is applied to increasing powers of the other co-prime. This equation normally takes the form of: $a^x mod b$, where a and b are co-primes and x is any integer value greater than 0. For example, if the pair of co-primes 2 and 3 are taken, the cycle obtained is 2, i.e., the remainders obtained from $2^n mod 3$ repeat after every 2 values of n.

To achieve this, first, a register of qubits is put into a state of superposition using the Hadamard transformation. Then, the qubit's state is transformed from $|y>$ to $|a^x mod b>$ through a series of quantum adder and multiplier functions, which are built using quantum Fourier transforms. Next, each of these final states is measured so that the value 'x' correlated to that qubit state can be tested against a mathematical cycle-checking function. If that value of 'x' satisfies the cycle parameters, as described before, the algorithm outputs that value, else it continues measuring states and testing other values of 'x'.

The algorithm can test multiple values of 'x' in a single computation using quantum computing and its related quantum concepts. In a classical algorithm, a single computation is required to test every value of 'x'. This leads to significant

 ISBN: ISBNFILL

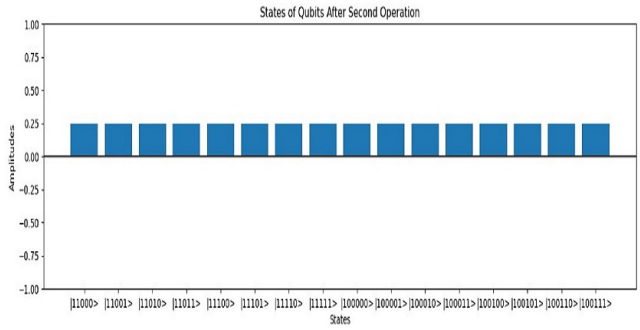

Fig. 5. Result of proposed framework of Shor's algorithm at the second step of execution. [20]

speed-up in terms of execution speed and overall time elapsed. For this algorithm, the framework generated the graph as seen in Figure 5. This bar graph represents all of the states currently in the form $|a^x mod b>$, with a certain value of 'x' associated with each state.

If the state contains the correct value of 'x', the state collapses back into $|0>$ with an amplitude of 1, as can be seen in Figure 6. If not the amplitude of this collapsed state would be 0. Thus, the framework was able to concisely explain the execution of the algorithm.

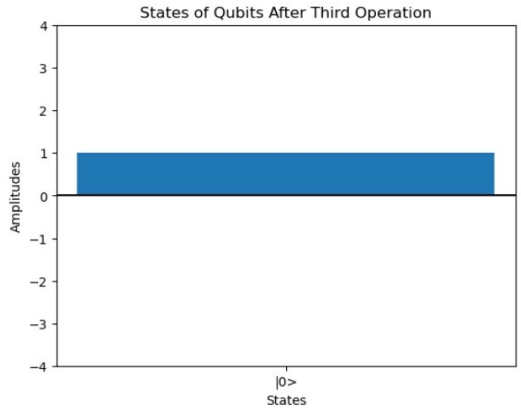

Fig. 6. Result of proposed framework of Shor's algorithm at the final step of execution. [20]

B. Grover's Algorithm

Grover's algorithm serves as a popular quantum search algorithm, that can be used to solve problems such as finding the shortest path in a graph or searching a minimum spanning tree [17]. Classical search algorithms would need to check at least half of the search space to find the solution with a high probability. On the other hand, Grover's algorithm can accept any type of query as input and will find a solution in polynomial time using quantum concepts.

The algorithm operates by using the concept of qubit reflection through template states to find the solution. Just as before, first, a register of qubits is placed in a state of superposition between the basis states. Then, to transform the qubits into the correct state, 2 operations are applied, namely:

1) Grover iterate matrix
2) Quantum reflection through a set of template states

The Grover iterate matrix takes the form of:

$$H^{\otimes n} R_0 H^{\otimes n} \tag{9}$$

Here, R_0 refers to a reversible matrix operation that rotates the qubit vector states, and H represents the Hadamard transformation applied n times [8].

In this example, the algorithm is trying to find a state with 6 qubits in it, with each qubit in the state $|1>$. If t represents the number of possible solutions for the search element x, then the template states for the problem are represented by:

$$|G\rangle = \frac{1}{\sqrt{t}} \sum_{i:x=1} |i\rangle \tag{10}$$

$$|B\rangle = \frac{1}{\sqrt{N-t}} \sum_{i:x=0} |i\rangle \tag{11}$$

Here, $|G>$ and $|B>$ represent the good and bad template states for this specific search problem [8]. These template states instruct the algorithm on which step to take next, based on the qubits' reflection through them. Finally, the algorithm measures the register qubits and outputs the state which contains the search element.

In this case, the states and their amplitudes after the reflection operations are shown in Figure 7. As can be seen, there are numerous states which have a low amplitude value and one state which has a high amplitude value. All of the states represent possible solutions to the search problem. However, the states with low amplitude values represent states where the probability of the search element being present is also low. Similarly, the states with high amplitude values represent states with a greater chance of containing the search element.

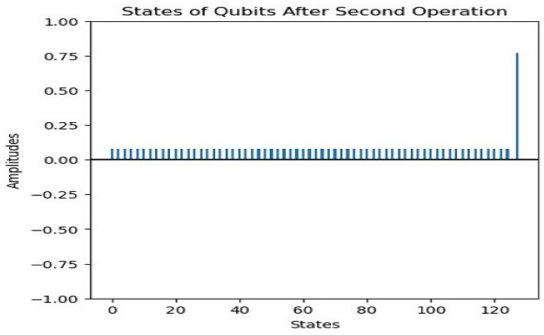

Fig. 7. Result of proposed framework of Grover's algorithm at the second step of execution. [20]

After measuring the possible solution states, the bar graph in Figure 8 is obtained. The state with 6 qubits, with all qubits being in $|1>$, has an amplitude of 1. This implies that the algorithm found the search element, and the framework was able to explain the execution of the algorithm in a systematic manner.

 ISBN: ISBNFILL

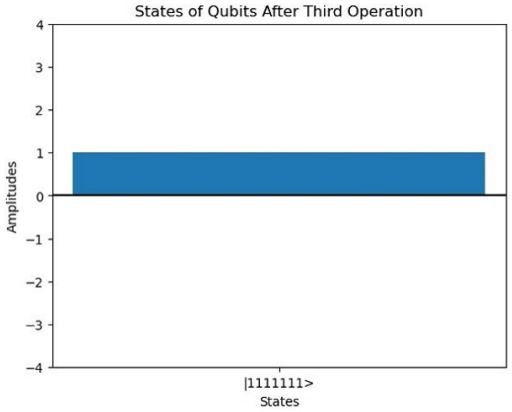

Fig. 8. Result of proposed framework of Grover's algorithm at the final step of execution. [20]

V. Conclusion and Future Work

The execution of various quantum algorithms described in this paper has been adequately visualized. By applying the proposed visualization framework to multi-qubit algorithms, we could effectively explain the execution of said algorithms through the changes in their respective bar graphs. This technique is relatively simple when compared to the existing methods, thus satisfying the objectives we set for the framework. Therefore, a visualization framework has been designed which can elucidate the execution of quantum algorithms in a straightforward manner. In the future, this framework should be extended to accept any quantum algorithm and display the related qubit data and graphs. This will allow multiple users to be able to draw conclusions and provide feedback on the clarity of the framework.

References

[1] L. Mearian, "IBM Touts Quantum Computing Advance," Computerworld, vol. 46, no. 5, p. 4, Mar. 2012, [Online]. Available: https://www.proquest.com/docview/1009070421/citation/E1588ADAB8A34312PQ/1. retrieved: September 2022

[2] "IBM Reveals Major Performance Gain For IBM Q System One," ICT Monitor Worldwide, Mar. 2019, [Online]. Available: http://www.proquest.com/docview/2187918772/citation/D43C5C0046674231PQ/1. retrieved: September 2022

[3] S. Adhikary, "Entanglement assisted training algorithm for supervised quantum classifiers," Quantum Information Processing, vol. 20, no. 8, Aug. 2021, doi: 10.1007/s11128-021-03179-w. retrieved: September 2022

[4] M. Fiasché, "A Quantum-Inspired Evolutionary Algorithm for Optimization Numerical Problems," in Neural Information Processing, 2012, pp. 686–693. doi: 10.1007/978-3-642-34487-9-83. retrieved: September 2022

[5] S. Imre, "Quantum computing and communications – Introduction and challenges," Computers & Electrical Engineering, vol. 40, no. 1, pp. 134–141, Jan. 2014, doi: 10.1016/j.compeleceng.2013.10.008. retrieved: September 2022

[6] Z. Tao, Y. Pan, A. Chen, and L. Wang, "ShorVis: A Comprehensive Case Study of Quantum Computing Visualization," in 2017 International Conference on Virtual Reality and Visualization (ICVRV), Oct. 2017, pp. 360–365. doi: 10.1109/ICVRV.2017.00082. retrieved: November 2022

[7] R. Wille, S. Hillmich, and L. Burgholzer, "Tools for Quantum Computing Based on Decision Diagrams," ACM Transactions on Quantum Computing, vol. 3, no. 3, p. 13:1-13:17, Jun. 2022, doi: 10.1145/3491246. retrieved: November 2022

[8] R. de Wolf, "Quantum Computing: Lecture Notes." arXiv, [Online]. Available: http://arxiv.org/abs/1907.09415. retrieved: November 2022

[9] N. Young, An Introduction to Hilbert Space. Cambridge University Press, 1988. retrieved: November 2022

[10] M. Born, "Quantum Mechanics of Collision Processes," Z. Physics, vol. 38, no. 11, pp. 803–827, Nov. 1926, doi: 10.1007/BF01397184, M. Born, "Quantenmechanik der Stoßvorgänge," Z. Physik, vol. 38, no. 11, pp. 803–827, Nov. 1926, doi: 10.1007/BF01397184. retrieved: November 2022

[11] A. Einstein, B. Podolsky, and N. Rosen, "Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?," Phys. Rev., vol. 47, no. 10, pp. 777–780, May 1935, doi: 10.1103/PhysRev.47.777. retrieved: November 2022

[12] "qiskit.visualization.plot-state-qsphere — Qiskit 0.39.3 documentation." https://qiskit.org/documentation/stubs/qiskit.visualization.plot-state-qsphere.html. retrieved: November 2022

[13] M. A. Nielsen and I. L. Chuang, Quantum computation and quantum information, 10th anniversary ed. Cambridge; New York: Cambridge University Press, 2010. retrieved: November 2022

[14] F. Bloch, W. W. Hansen, and M. Packard, "The Nuclear Induction Experiment," Phys. Rev., vol. 70, no. 7–8, pp. 474–485, Oct. 1946, doi: 10.1103/PhysRev.70.474. retrieved: November 2022

[15] "Microsoft Quantum Development Kit Libraries." Microsoft. "microsoft/QuantumLibraries: Q Sharp libraries for the Quantum Development Kit." Available: https://github.com/microsoft/QuantumLibraries. retrieved: November 2022

[16] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," SIAM J. Comput., vol. 26, no. 5, pp. 1484–1509, Oct. 1997, doi: 10.1137/S0097539795293172. retrieved: November 2022

[17] L. K. Grover, "Quantum Mechanics Helps in Searching for a Needle in a Haystack," Phys. Rev. Lett., vol. 79, no. 2, pp. 325–328, Jul. 1997, doi: 10.1103/PhysRevLett.79.325. retrieved: November 2022

[18] M. Roetteler, M. Naehrig, K. M. Svore, and K. Lauter, "Quantum Resource Estimates for Computing Elliptic Curve Discrete Logarithms," in Advances in Cryptology – ASIACRYPT 2017, Cham, 2017, pp. 241–270. doi: 10.1007/978-3-319-70697-9-9. retrieved: November 2022

[19] S. A. Bhavsar, V. H. Patil, and A. H. Patil, "Graph partitioning and visualization in graph mining: a survey," Multimedia tools and applications, vol. 81, no. 30, pp. 43315–43356, 2022, doi: 10.1007/s11042-022-13017-5. retrieved: January 2023

[20] vs4503, Github Repository, "Capstone-Project." Available: https://github.com/vs4503/Capstone-Project. retrieved: November 2022

 ISBN: ISBNFILL