Solving NP Complete problem using QAOA

Group Name - HM77
Group Members - Pramod Asagodu, Payal
Master of Quantum Science and Technology: Fall Quarter 2022

NP Complete problem of choice - Dominating Set Problem.

I. DOMINATING SET PROBLEM

The dominating set problem is a combinatorial NP Complete problem. In this problem for a graph $G(E, V)$, a subset of vertices $S(V_s)$ is to be found such that all of the vertices V of the graph G either belong to S or are the nearest neighbour of any vertex belonging to the subset, i.e. have an edge with at least one of the vertex in the subset. The question asked by the NP Complete problem is,

What is the smallest possible subset of the graph $G(E, V)$ such that all of the vertices V of the graph G either belong to S or are the nearest neighbour of any vertex belonging to the subset?

A bitstring $z = z_0 z_1 z_2 .....z_{n-1}$ is given as an input to the problem where when the bit = 1, the corresponding node in the graph is present in the subset and if the bit = 0 then the corresponding node is not present in the subset.

The problem statement has 2 clauses.

$$T_k(z) = \begin{cases} 1 & \text{if the kth nodes is connected to some ith node where } z_i = 1 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$D_k(z) = \begin{cases} 1 & \text{if } z_k = 0 \\ 0 & \text{if } z_k = 1 \end{cases} \tag{2}$$

The T clause is used to find the number or vertices that are connected to a vertex that is in the subset S. A node that belongs to the subset S is also considered to be connected to a vertex belonging to the subset.

The D clause gives the number of nodes that are not present in the subset.

The total cost of the problem is calculated by summing the total values of T and D. That is -

$$\text{Cost} = \sum_{\text{k}=0}^{n-1} T_k(z) + D_k(z) \tag{3}$$

The problem is solved when the cost function is maximized. We will use this to optimize the parameters further in the code.

II. SEPERATORS AND MIXERS USED

We have designed the separator and the mixer in a very different way as compared to the mixer and separator matrices described in the class notes and the conventional texts.

The γ parameter and the β parameter of the circuit were further more parameterized than the circuits in conventional texts.

The separator circuit was designed such that when an equal superposition -

$$|\psi\rangle = \tfrac{1}{2^n}(|00...0\rangle + |00...1\rangle + |00...10\rangle ..... + |11...1\rangle) \tag{4}$$

is passed through the seperator gate S, the final state has different phase for each application of the separator gate. This means that the phase of different states of the qubit sequence is not an integer multiple of a particular γ parameter but each individual γ itself are different.

i.e. -

$$S * |\psi\rangle = \tfrac{1}{2^n}(e^{-\mathrm{i}\gamma_0} |00...0\rangle + e^{-\mathrm{i}\gamma_1} |00...1\rangle + e^{-\mathrm{i}\gamma_2} |00...10\rangle ..... + .e^{-\mathrm{i}\gamma_{n-1}} |11..1\rangle) \tag{5}$$

Similarly one of the mixer is defined such that the each of the β parameters is different for diffierent qubits. While, the other mixer used was the conventional mixer which had a single β parameter. That is, one of the mixer is -

$$Mix1(\beta) = R_x(2\beta_0) \otimes R_x(2\beta_1) \otimes .....R_x(2\beta_i) \otimes R_x(2\beta_{i+1})..... \otimes R_x(2\beta_{n-1}) \tag{6}$$

And the other mixer is -

$$Mix2(\beta) = (\otimes n)R_x(2\beta) \tag{7}$$

The parameterization of each of γ and β gives us solution with a very high probability.

III. PARAMTERIZATION OF SOLUTION IN NUMBER OF QUBITS

The input to the code is given in the form of the edges and the number of nodes of the graph for which the solution is required and the iterations of the separator and the mixer gates applied in the quantum circuit. The different functions defined in the code use the input to decide on the number of qubits required to run the quantum circuit to obtain the solution.

The number of qubits is very easy to change as it is not an input to the code nor is the code hardcoded for a specific number of qubits. The only thing that the code requires is the problem graph and the number of iterations of Separator and the Mixer gates in the quantum circuit. The number of qubits is directly proportional to the number of nodes in the graph (N+1, where N is the number of nodes in the graph).

IV. TESTING THE PROGRAM

The three sizes of problems or the three graphs of the problem instances for which the code was run are -

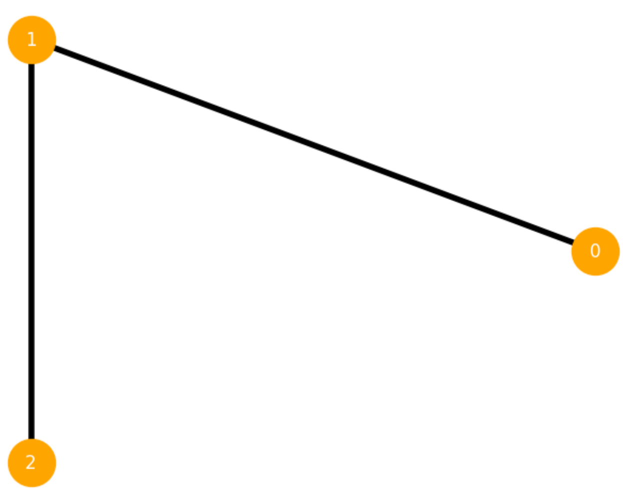

FIG. 1. Problem Instance 1: 3 Node Graph

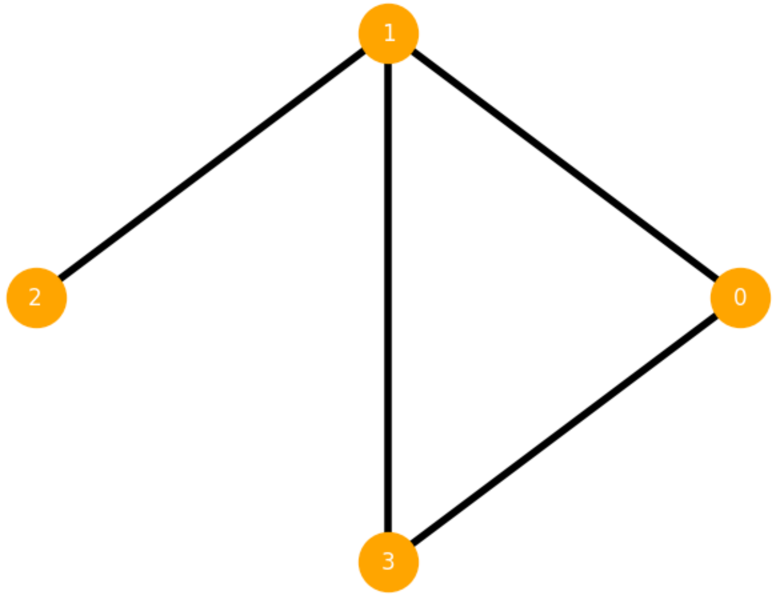

FIG. 2. Problem Instance 2: 4 Node Graph

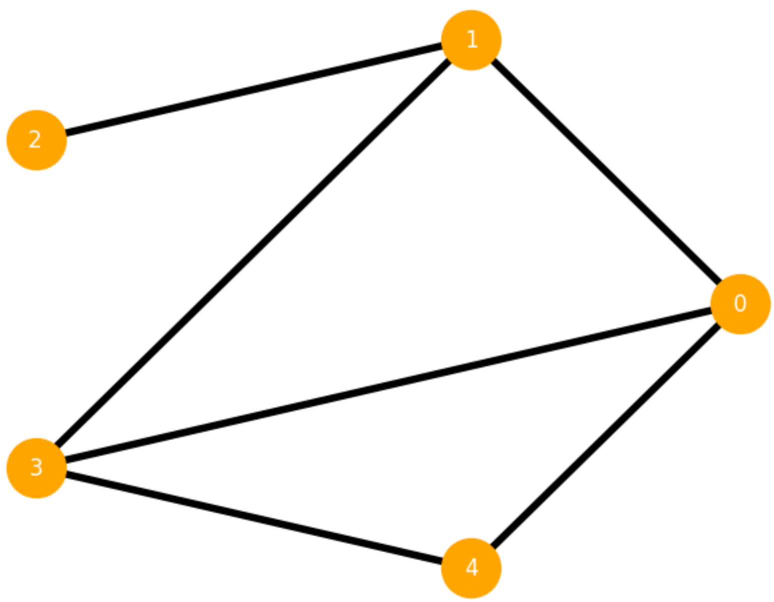

FIG. 3. Problem Instance 3: 5 Node Graph

The problems were tested for different iterations of seperator-mixer implementations in the quantum circuit and on different systems. The parameters were optimized using both noisy and noiseless systems. As evident from the results the solution was best obtained when the whole circuit including the optimization and the final circuit was run on the noiseless circuit.

A. For $Mix1(\beta)$

1. Noiseless Simulation

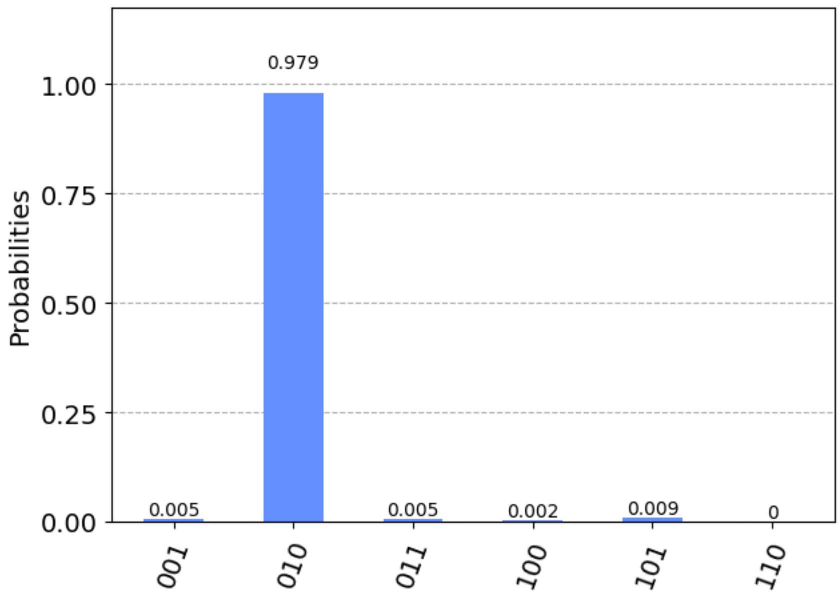

FIG. 4. Solution for 3 Nodes. Expected output = 010, probability = 97.9 %

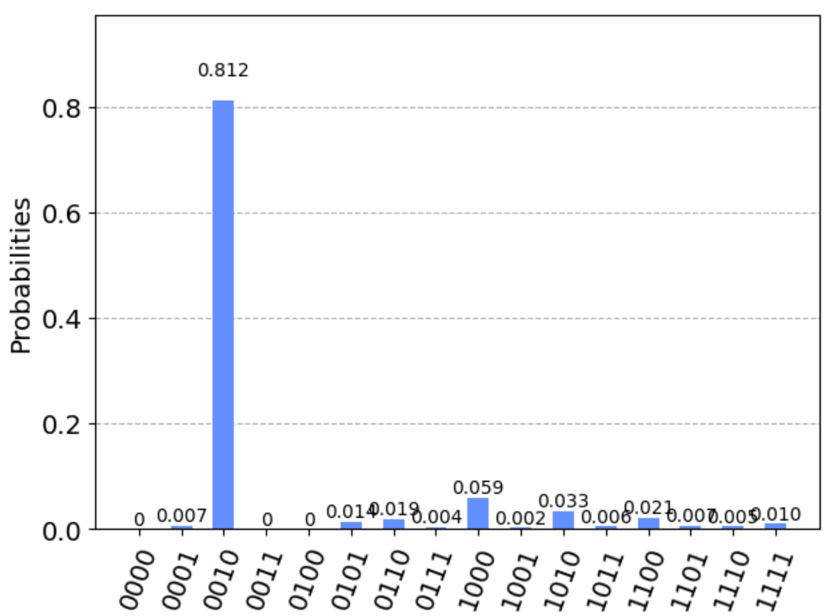

FIG. 5. Solution for 4 Nodes. Expected output = 0100 (the quantum computer gives inverted result), probability = 81.2 %

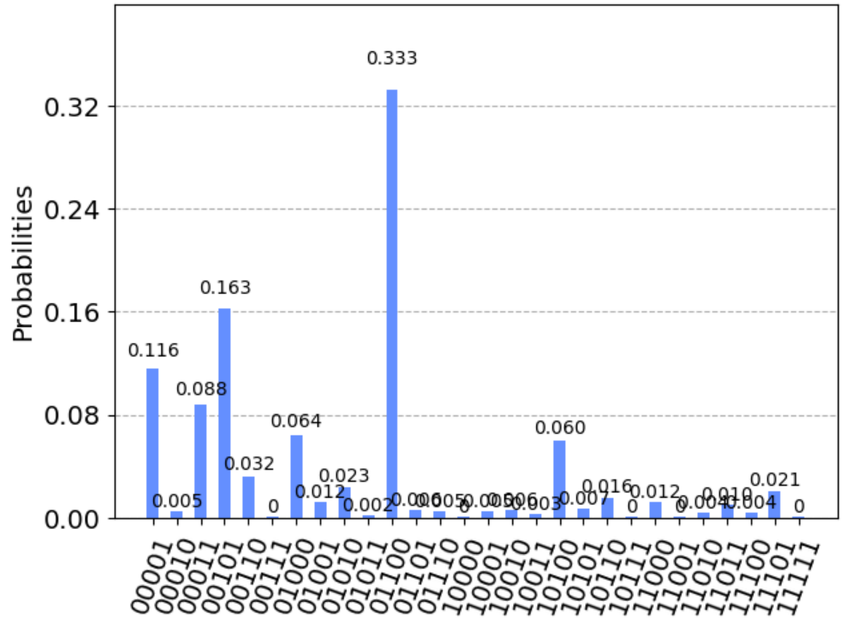

FIG. 6. Solution for 5 Nodes. Expected output = 00110 (the quantum computer gives inverted result), probability = 33.3 %

2. Noisy Simulation (FakeNairobi Backend)

The noise can be added at two places. One where the parameter are optimized and the other where the circuit is run for the final optimized parameters obtained. When the parameters were optimized on a noisy circuit the parameters produced wrong results. for example -

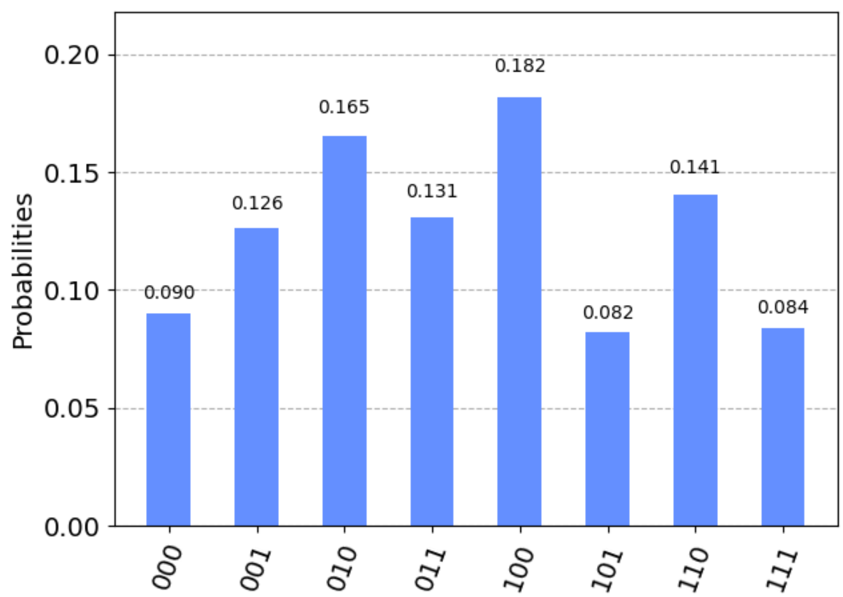

FIG. 7. 3 Node problem: Optimization of parameters on noisy system for 1 iteration

For the same problem and same number of seperator-mixer iterations, when the parameters were optimized on noiseless system and only the final run was made on a noisy system, the graph look like follows-

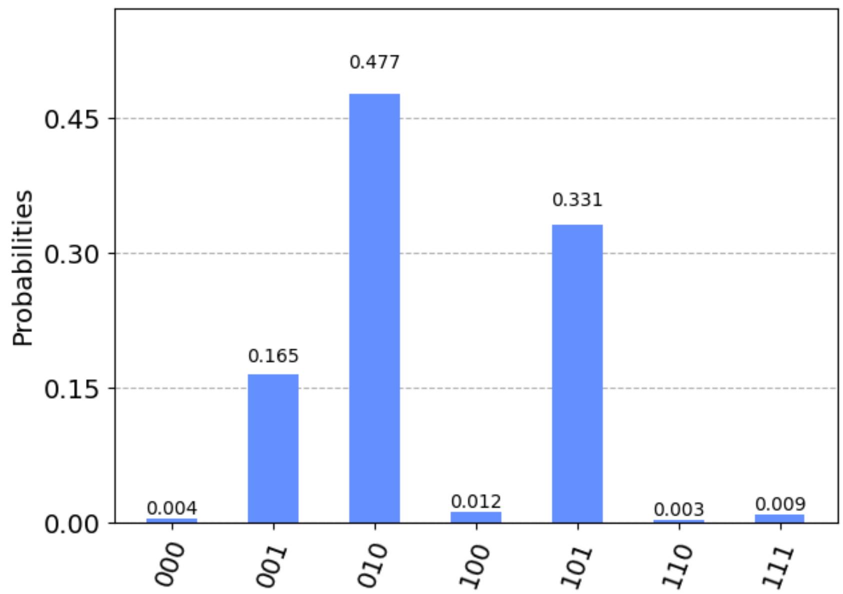

FIG. 8. 3 Node problem: Optimization of parameters on noiseless system for 1 iteration

Therefore the parameters were optimized on noiseless system and then for the optimized parameters obtained, the final simulation, to obtain the solution, was run on a noisy system for both cases; Noisy simulator and the IBM Quantum Computer.

The parameters used to run on the noisy systems were the same as that obtained for running the final circuit on noiseless system. The parameters are obtained for 5 iterations of seperator-mixer circuits.

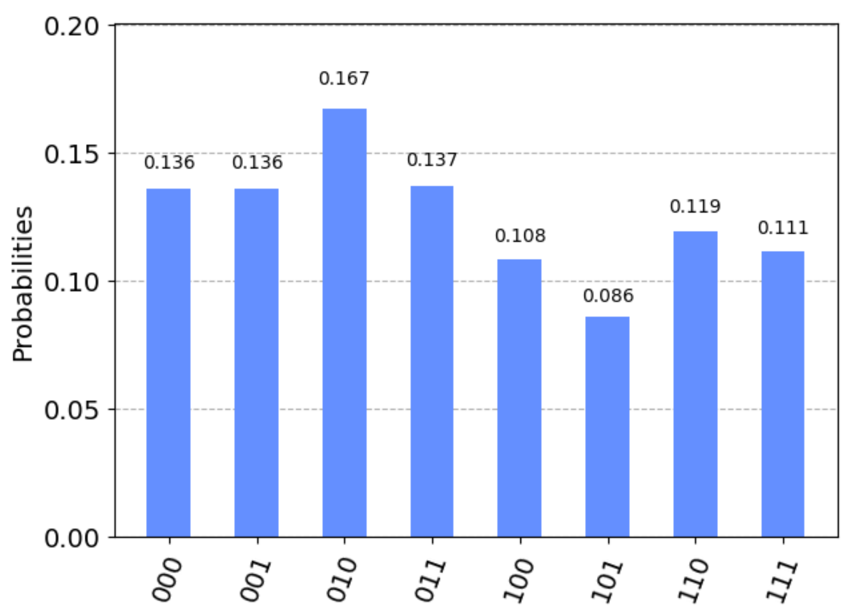

FIG. 9. 3 Node problem: Final circuit run on noisy simulator

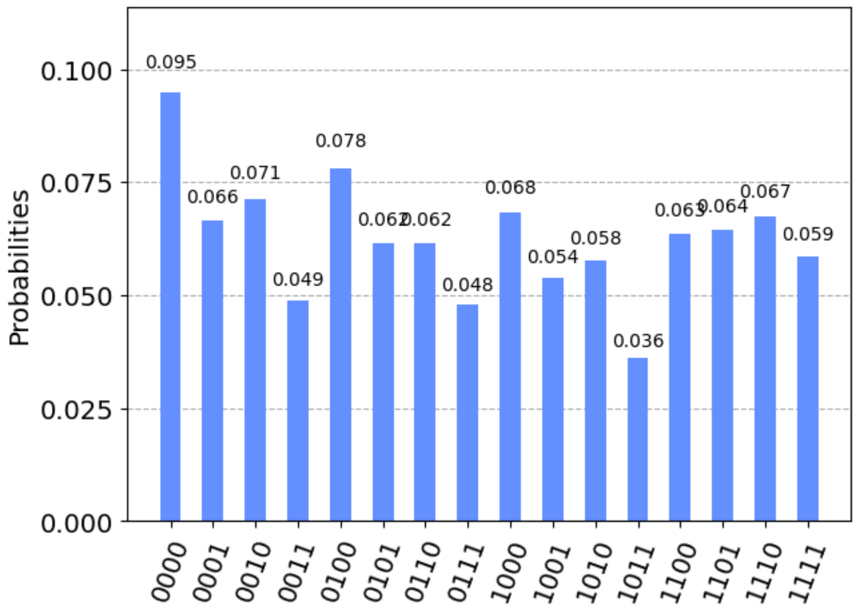

FIG. 10. 4 Node problem: Final circuit run on noisy simulator

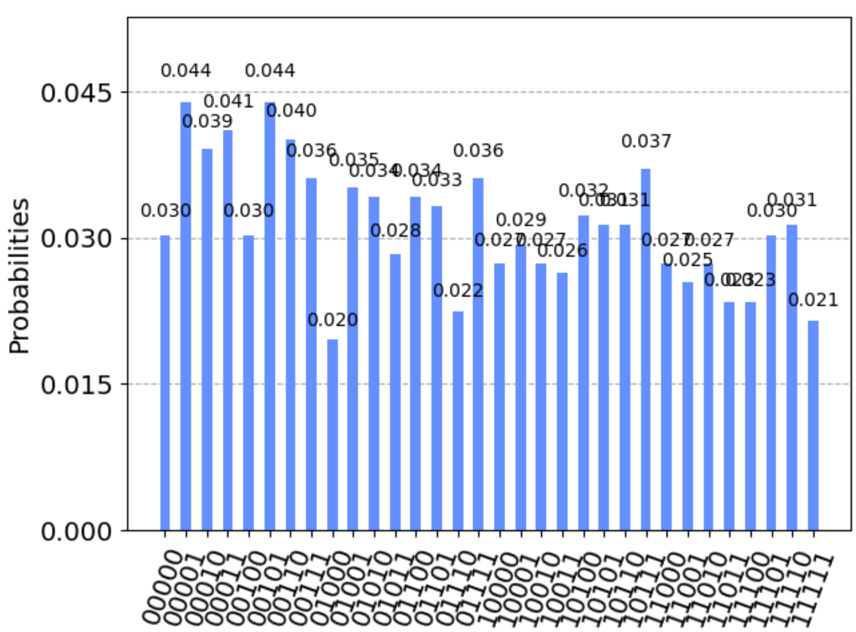

FIG. 11. 5 Node problem: Final circuit run on noisy simulator

3. Implementing on IBM Oslo Quantum Computer

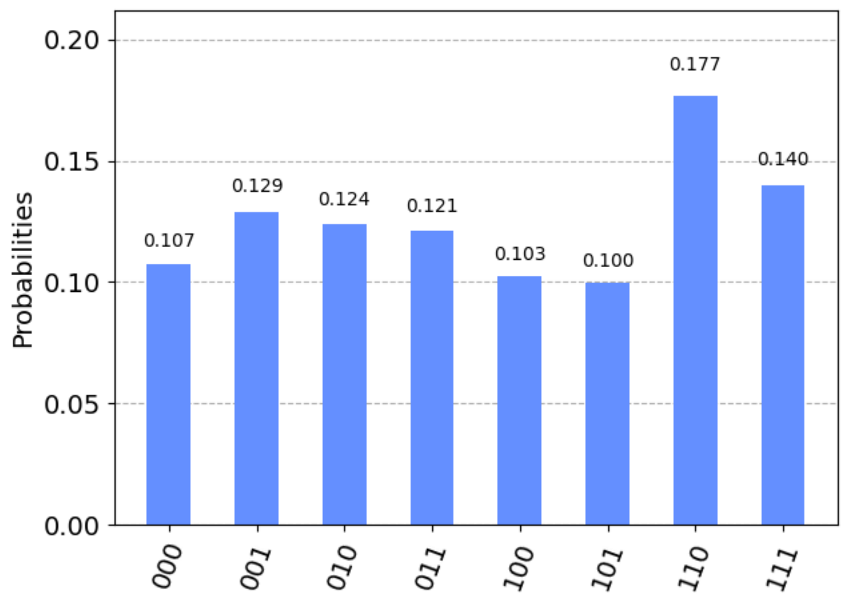

FIG. 12. 3 Node problem: Final circuit run on IBM Quantum computer Oslo

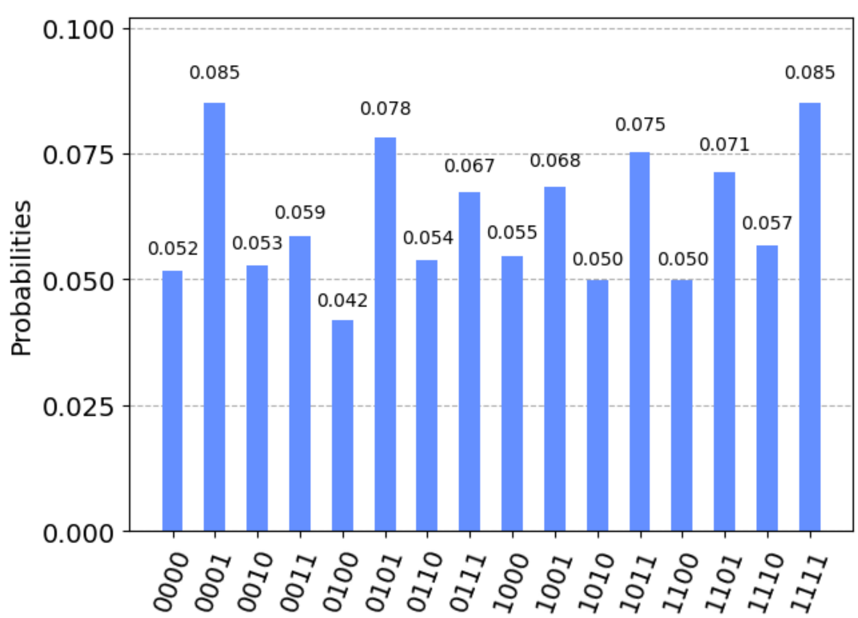

FIG. 13. 4 Node problem: Final circuit run on IBM Quantum computer Oslo

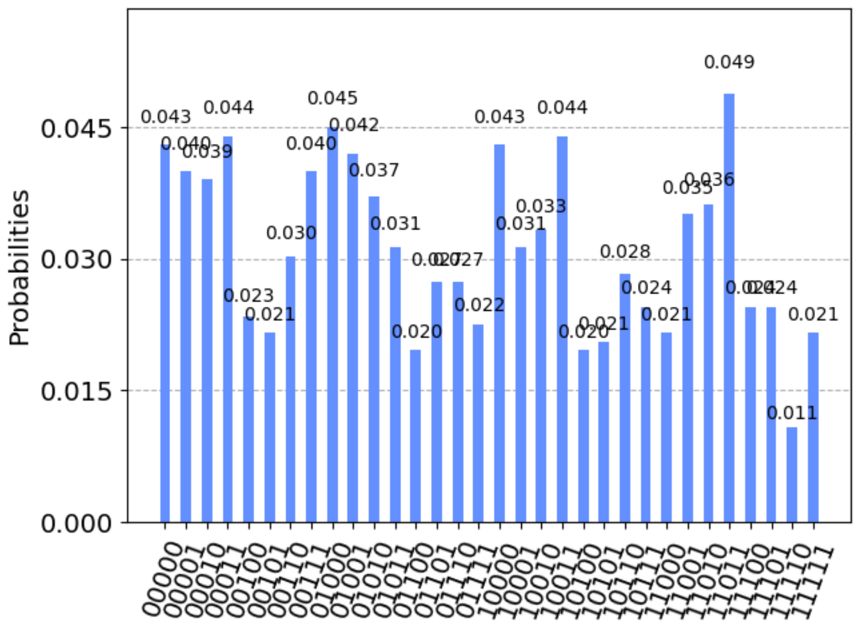

FIG. 14. 5 Node problem: Final circuit run on IBM Quantum computer Oslo

B. For $Mix2(\beta)$

1. Noiseless Simulation

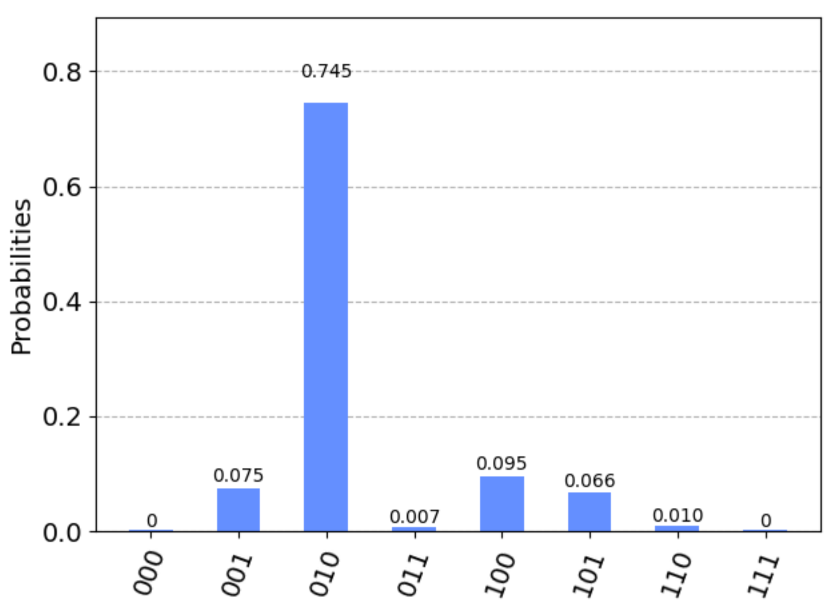

FIG. 15. 3 Node Problem: simulation for $Mix2(\beta)$

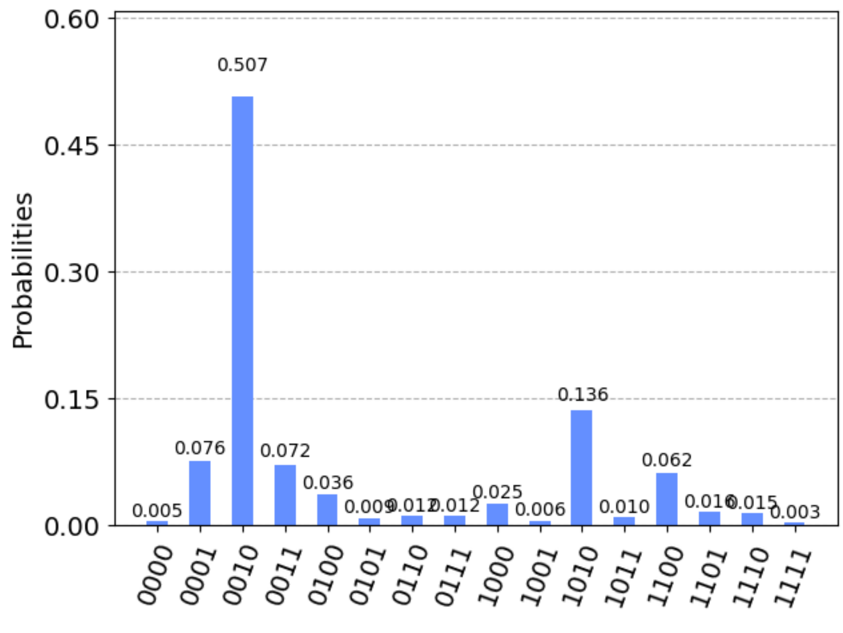

FIG. 16. 4 Node Problem: simulation for $Mix2(\beta)$

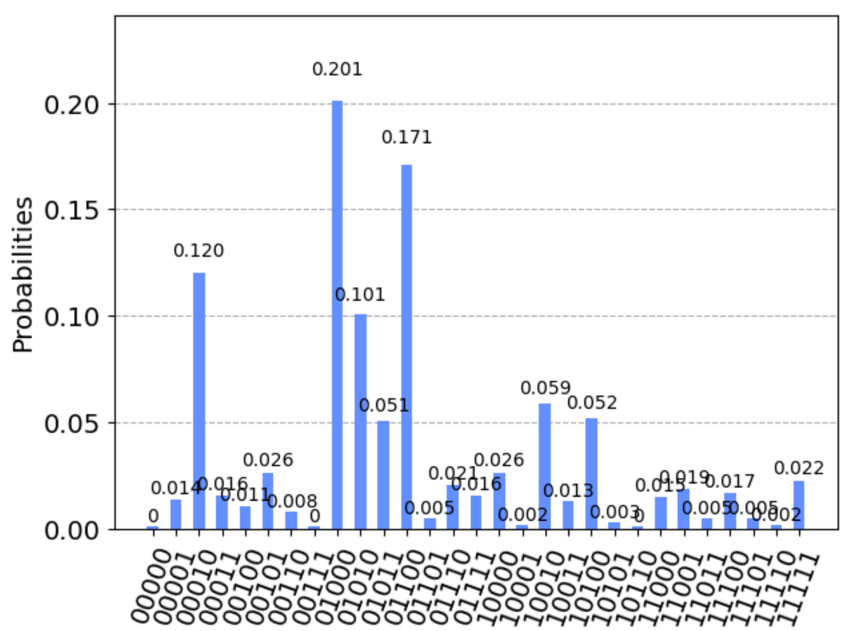

FIG. 17. 5 Node Problem: simulation for $Mix2(\beta)$

2. Noisy Simulation (FakeNairobi)

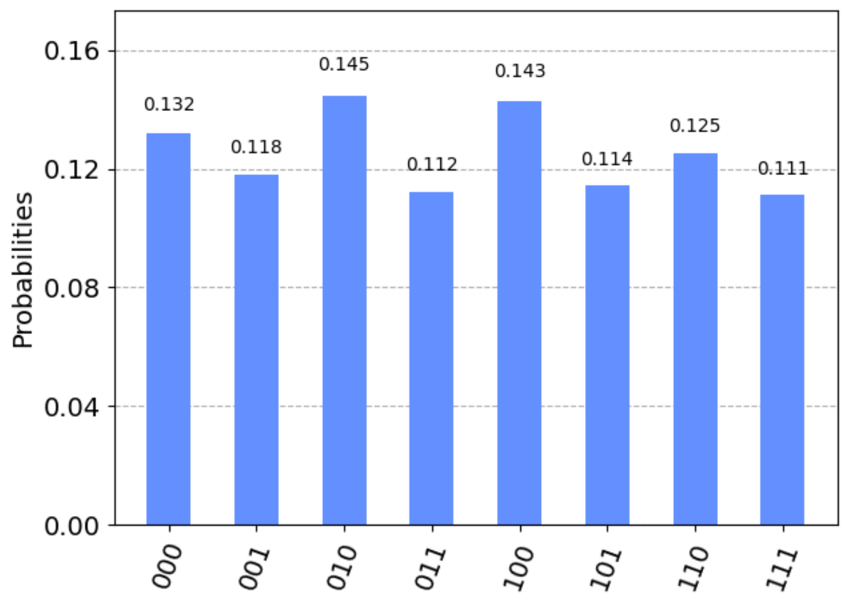

FIG. 18. Noisy simulation of the final circuit using mixer $Mix2(\beta)$ for 3 Node problem

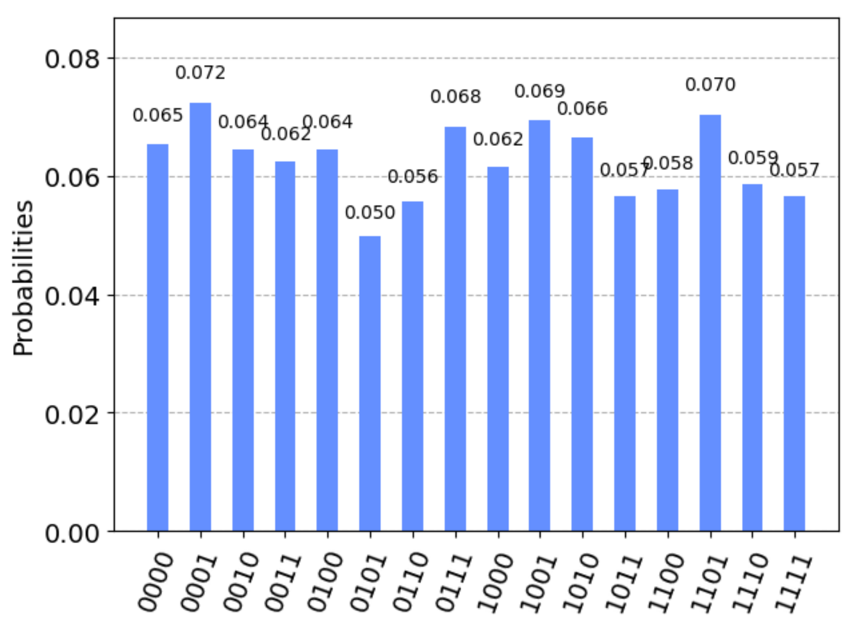

FIG. 19. Noisy simulation of the final circuit using mixer $Mix2(\beta)$ for 4 Node problem

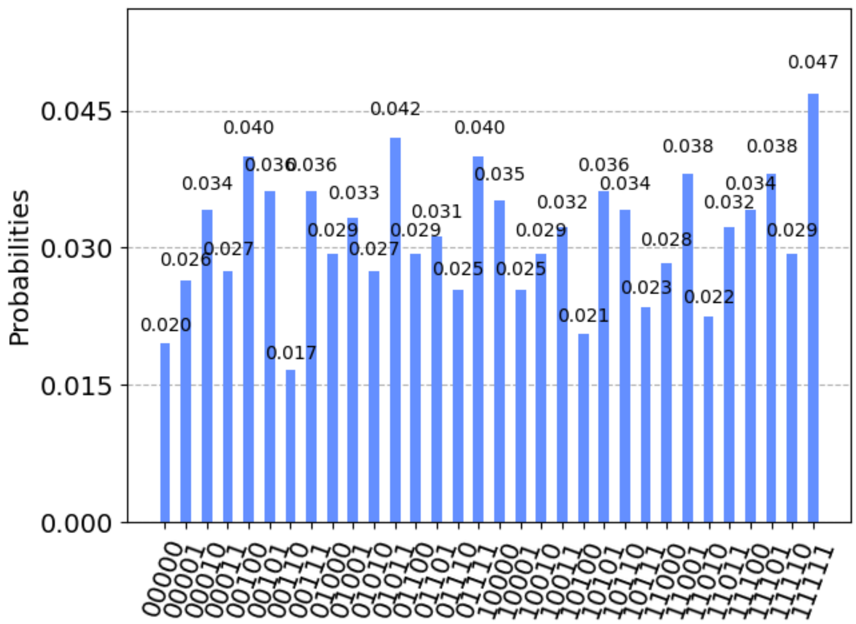

FIG. 20. Noisy simulation of the final circuit using mixer $Mix2(\beta)$ for 5 Node problem

3. Implementing on IBM Oslo Quantum Computer

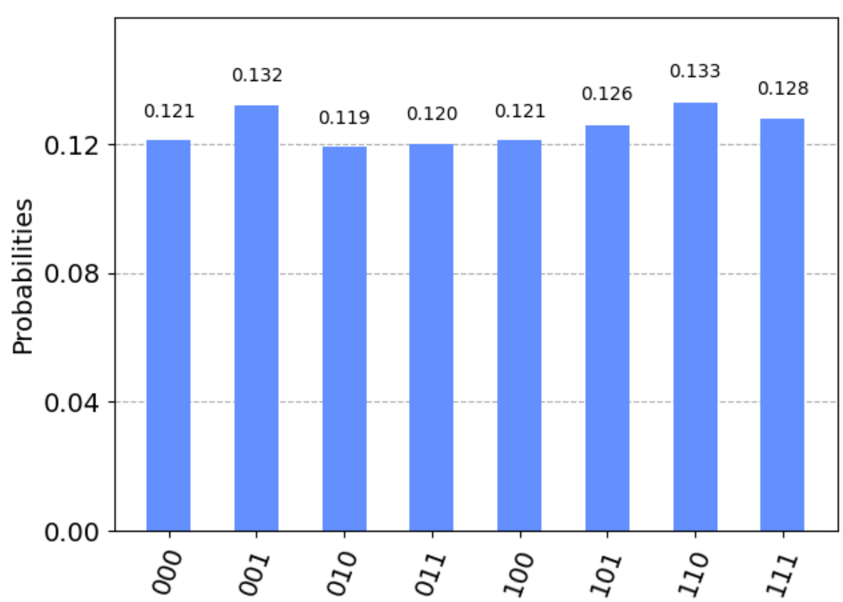

FIG. 21. 3 Node problem: Final circuit run on IBM Quantum computer Oslo for $Mix2(\beta)$

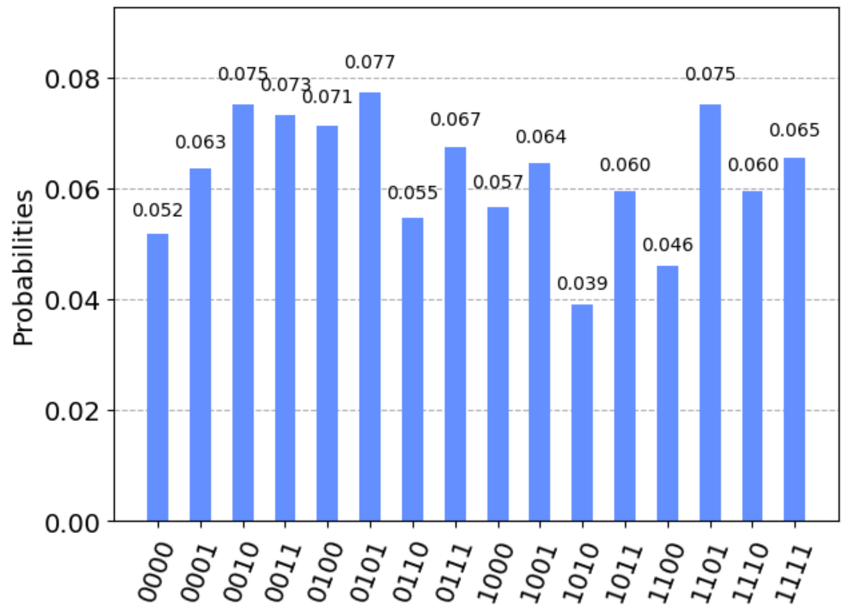

FIG. 22. 4 Node problem: Final circuit run on IBM Quantum computer Oslo for $Mix2(\beta)$

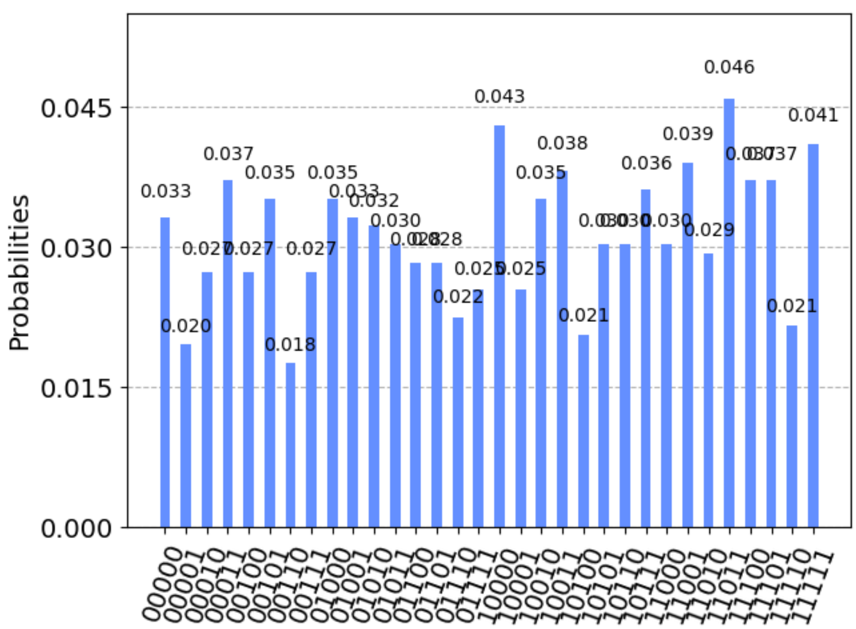

FIG. 23. 5 Node problem: Final circuit run on IBM Quantum computer Oslo for $Mix2(\beta)$

V. SCALABILITY

As the number of qubits grow the time taken for implementing the circuit as a whole increases . Our circuit runs for a many number of times (number unknown as the optimizer runs the circuit according to optimization). The optimization of the parameters is done by running the quantum circuit many number of times until the optimal parameters for finding the solution of the problem is obtained. After the circuit has been run (for around 400 times each for around 1000 shots) the parameters are obtained which are then used to find the final solution by running the circuit again for the obtained parameters and measuring the final state of the qubits. This causes the time of execution to increase for increasing number of qubits. The increase in execution time observed for 4, 5, and 6 qubits respectively was approximately linear suggesting that the code can be scaled to obtain solution for higher number of qubits.

The quality of the solution, on the other hand, decreased as the number of qubits were increased because decoherence starts to kick in due to usage of a lot of quantum gates and long execution time. The other possible decrease in the probability could be existence of mamy possible solutions which decreases the possibility of obtaining each individual solution.

- Execution Time for 3 qubits = 1 min 21.5 seconds
- Execution Time for 4 qubits = 2 min 33 seconds
- Execution Time for 5 qubits = 4 min 6 seconds
- Probability of expected output for 3 qubits = 97.9%
- Probability of expected output for 4 qubits = 81.2%
- Probability of expected output for 5 qubits = 33.3%

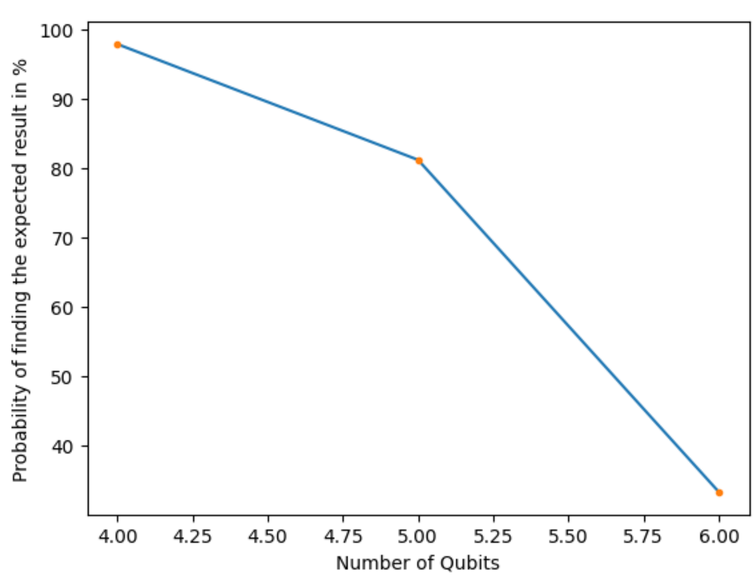

FIG. 24. Probability of finding expected output V/S Number of qubits graph. As you can see the reliability of the output decreases with increase in the number of qubits.

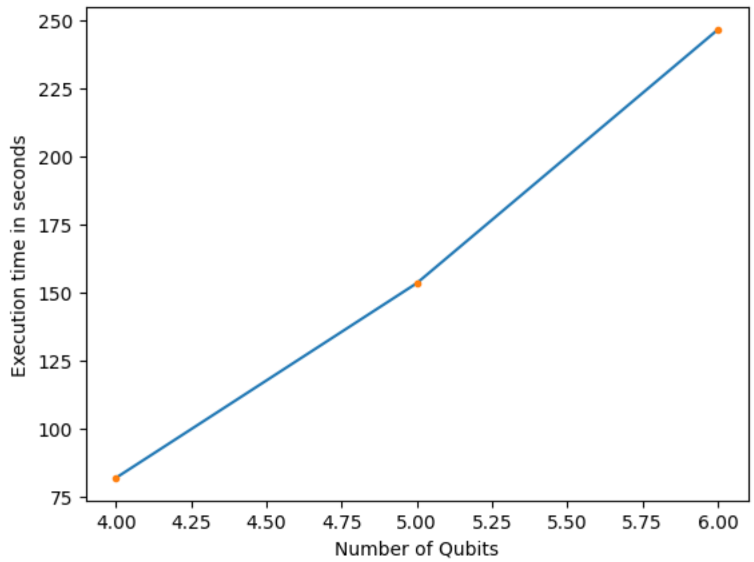

FIG. 25. Execution time V/S Number of qubits graph. As you can see the execution time increases with the increase in the number of qubits. The graph looks linear for number of qubits 4,5,6

VI. COMPARING SIMULATOR AND QUANTUM COMPUTER

A. For 3 Node Problem

- Output of noiseless simulator = 010 with 97.9% probability
- Output of noisy simulator = 010 with 16.7% probability
- Output of IBM Quantum Computer = 110 with 17.7% probability

B. For 4 Node Problem

- Output of noiseless simulator = 0010 with 81.2% probability
- Output of noisy simulator = 0000 with 9.5% probability
- Output of IBM Quantum Computer = 0001 and 1111 with 8.5% probability

C. For 5 Node Problem

- Output of noiseless simulator = 00110 with 33.3% probability
- Output of noisy simulator = 00001 and 00101 with 4.4% probability
- Output of IBM Quantum Computer = 11011 with 4.9% probability

This shows that the QAOA optimization is very sensitive to the noise present in the circuit. The code has an optimization part and the final implementation part that finds the final solution. The optimization part of the QAOA implementation was run on a noiseless backend to make the comparison with for both quantum computer backend and the noisy simulator backend. (The optimizer performs optimization by running the circuit many times and providing the optimized values of the parameters. This cant be done on a quantum computer currently as it is a public hardware and the queue would take days to complete the optimization).

For the 3 Node problem the noiseless simulator gives the best and very accurate result. The noisy simulator gives the same result but with smaller probability and the quantum computer does not give the same output. For the remaining problems, both the noisy and the real quantum computer fail.

This shows that the error in the real quantum computer is very high for the QAOA algorithm to be run to solve Dominating Set Problem of a practical size. The noiseless simulator gives very accurate results for a problem set of small size but as this problem set increases the noiseless simulator itself gives output with smaller probaibilty.

The noise in the noisy simulator is a very near representation of the noise in the quantum computer. The noise model used in the noisy simulation is of "Nairobi" Quantum computer but the Quantum Computer used was "Oslo" (due to a long queue for Nairobi Quantum Computer). Looking at the output it is very difficult to comment about how realistic the noise model is because the outputs of both the system are not accurate. But, since the noisy simulator gave us a correct solution for 3 Node with about 16% it could be said that the Quantum computers are really more noisy than the noisy simulator.

This noise causes decoherence of the qubits and, hence, faulty outputs.

VII. COMPARING THE 2 MIXERS

The mixer $Mix1(\beta)$ is more parameterized than the mixer $Mix2(\beta)$ as you can see from 6 and 7. The two different mixers are used for the same inputs and the same number of iterations of separator-mixer gates implementation in the circuit by running on the noiseless simulator. The output of mixer $Mix1(\beta)$ can be seen in the section IV A 1 and the output of the mixer $Mix2(\beta)$ can be seen in the section IV B 1.

The two mixers give accurate output for the 3 Node and 4 Node problems, but the $Mix1(\beta)$ gives the solution with higher probability. Meanwhile the $Mix2(\beta)$ gives inaccurate result for the 5 Node problem The $Mix1(\beta)$ mixer is more parameterized and gives more accurated result as compared to the conventional $Mix2(\beta)$ mixer that is most commonly used. This parameterization of the individual mixer gates increases the accuracy of the result as compared to having a single parameter of rotation. This is evident in our outputs for the different mixers. We can easily conclude that the $Mix1(\beta)$ is better than the $Mix2(\beta)$ mixer.

VIII. COMPARING 2 INSTANCES

Unfortunately we were not able to find such instances since the noise in the quantum computer limited our ability to check for different values. The noise in the quantum computer and the noisy simulator make these optimizations inaccurate when run for problems of practical sizes. Judging by the different simulations and the real quantum computer runs, it can be said that the problems having multiple correct solutions would be a good choice to run on the quantum computer as the quantum computer would give the correct solutions with equally likely probability but it would be difficult to find such solutions classically. On the other hand such problems sometimes interfere with the optimization part of the QAOA parameters as the optimizer might sometimes get stuck at a particular local minima (or maxima depending upon the optimization being carried out) and not give the right solution.

IX. REFERENCE PAPER

[1]Guerrero, Nicholas J., "Solving Combinatorial Optimization Problems using the Quantum Approximation Optimization Algorithm" (2020). Theses and Dissertations. 3263. https://scholar.afit.edu/etd/3263