AMQIS Project Report

Aditya Mishra, 20PH20002
Anuranjan Sarkar, 20PH20004
Dhatri Raghunathan: 20PH20009
Dhruv Baheti: 20PH20010

Problem statement

Solving vehicle routing problem using Quantum Approximate Optimization Algorithm. The problem grows in complexity with the dependency of the order of $O(N^2)$. Its exact dependance is N(N-1). We will be simulating low depth instances as IBM's simulators are limited to 32 qubits and the available quantum computers only have 7-8 qubits. Thus our simulations will be limited to instances of a maximum of 6.

Explaining the problem

Vehicle Routing Problem is an NP-hard combinatorial optimization problem. Any problem instance (n, k) of VRP involves k vehicles, and n − 1 locations (other than the depot D). Its solution is the set of routes in which all of the k vehicles begin and end in the D, such that each location is visited exactly once. The optimal route is the one in which the total distance travelled by k vehicles is the least.
A quadratic unconstrained binary optimization (QUBO) problem consists of finding a binary vector that minimises something. It is often solved approximately using optimization algorithms.
The Quantum Approximate Optimization Algorithm are a class of quantum-classical hybrid variational algorithms that attempts to solve combinatorial problems.

Thus we attempt to solve the vehicle routing problem using Warm Start QAOA.

Hamiltonian formulation

The first step to solving the VRP using this technique is to map all the constraints to the minimization of an Ising Hamiltonian. For the binary decision variable x_{ij}, the energy functional H_{VRP} of the problem can be written as:

$$H_{VRP} = H_A + H_B + H_C + H_D + H_E$$

$$H_A = \sum_{i \to j} w_{ij} x_{ij}$$

$$H_B = A \sum_{i \in 1,\ldots,n-1} \left(1 - \sum_{j \in source[i]} x_{ij}\right)^2$$

$$H_C = A \sum_{i \in 1,\ldots,n-1} \left(1 - \sum_{j \in target[i]} x_{ji}\right)^2$$

$$H_D = A\left(k - \sum_{j \in source[0]} x_{0j}\right)^2$$

$$H_E = A\left(k - \sum_{j \in target[0]} x_{j0}\right)^2$$

Where the individual terms are indicated as constraints imposed by the VRP. H_A is the main VRP formulation. H_B and H_C impose the node-visiting constraint so that each node is visited exactly once. Also, H_D and H_E pose the constraint to enforce that all the vehicles begin from and return back to depot D, i.e., node 0.

Solution approach

The solution to this problem is inspired by the adiabatic model of quantum computing with the cost and mixer hamiltonian. It is however done in a discretized p step in the circuit model of computation. The details of the above are explained in the paper that we referred to. The goal is to find states that correspond to a lower cost(expectation) given the constraints, which is encoded as a hamiltonian, and to interpret that information in terms of an adjacency matrix(to figure out the actual path that minimises cost).
In this model, to perform any computation we need two Hamiltonians called Hˆmixer and Hˆcost. Amongst them, the ground state of Hˆmixer should be an easily preparable state such as $|+\rangle^{\otimes N}$ and the ground state of Hˆ cost encode the solution to our problem. Both Hamiltonians Hˆmixer and Hˆcost should be local, i.e. they only involve terms for interactions between a constant number of particles.

The algorithm used

In this code, another classical algorithm is used to “warm-start” the program for the quantum computer. Warm starting is the use of an algorithm to use a good initial guess for the solution of the problem. This guarantees the classical performance of the quantum computer which further improves the solution. As we have demonstrated, for low-depth systems, warm starting has a very pronounced effect on the run time and the quality of results for this QUBO problem.

Code

This is the code we implemented using IBM Quantum Lab to run the same. The code along with comparisons and relevant graphics has been made as a separate Jupyter Notebook.

A Peculiar Issue

When the system was solved for the trivial cases like instances (n, n) or (n, n-1) it gave us the correct answers, which if it had not would mean the code is completely wrong. But on testing cases like (5, 2) and (5, 3) we noticed that the optimal path being computed was involving more than 2 or 3 vehicles. This was quite confusing and after manually looking at the adjacency matrix and interpreting the results I realised that all the constraints that we had set on the solution were satisfied yet we had this puzzling result.

So the cause of the issue was that there was a constraint that was missing in the equations of the original paper, from which we had picked up the algebra. When we want to run the above code, the value of k is only used to decide values of parameters of the equations of the system that need to be solved and two of the constraint equations as well. However, it is just entered into the equation as a parameter and not as the number of vehicles available, as we would interpret it in the context of this problem. Thus what ends up happening is the that the condition that k vehicles leave the depot node is satisfied but that does not mean that other vehicles are not available which don't leave from node 0 and still satisfy the entire constraint. Thus we need a way to add the constraint of constant number of vehicles to the hamiltonian as well, the depot node condition does not suffice by itself as a complete set of constraints.

References: We have used majorly 2 papers:

- https://arxiv.org/abs/2002.01351
- https://arxiv.org/abs/2009.10095