COL861: Special Topics in Hardware Systems

Solving Max-Cut problem using QAOA

Mohd. Sayanur Rahman (2021PH10221)[1] Instructor: Prof. Kolin Paul[1]

[1]Indian Institute of Technology, Delhi

Table of contents

Problem

Problem statement

Given a graph, determine a partition of nodes such that the number of edges between the partitions is maximum possible for the graph. The Problem is NP-Hard. Here, we try to approximate the solution in polynomial time complexity using Variational Quantum Algorithm, Quantum Approximate Optimization Algorithm (QAOA)

Methodology

QUBO formulation

The Max-cut problem is first reduced to Quadratic Unconstrained Binary optimization problem (QUBO) as

$$\min_{x \in \{0,1\}^n} \sum_{(i,j) \in Edges} 2x_i x_j - x_i - x_j$$

where binary variable x_i is the partition of i^{th}Node. Based on similar idea, Optimization can be expressed using variables, $z_i = (-1)^{x_i}$, as

$$\min_{x \in \{0,1\}^n} \sum_{(i,j) \in Edges} z_i z_j$$

Cost Hamiltonian

Taking $\sum_{(i,j)\in Edges} z_i z_j$ as the cost function for variational algorithm, the Hamiltonian can be formulated as

$$H_C = \sum_{(i,j)\in Edges} Z_i Z_j$$

where $Z_i = I^{\otimes i-1} \otimes Z \otimes I^{\otimes n-i}$ is the application of Z on i^{th} qubit only.

QAOA

The Variational algorithm here involves application of operator

$$\prod_p e^{-\alpha_p H_c} e^{-\beta_p H_m}$$

where H_m is a mixer Hamiltonian intended for exploring solution space. The Initial state of qubits and the above operator constitutes an ansatz, which transforms the system to a parametric state $|\psi_\theta >$, where the parameters θ are optimized to get minimum energy eigenstate of hamiltonian H_c using classical optimizer.

Examples

Example problems statistics

Three graphs of size $5, 10, 15$ were tried to be solved using QAOA. The problem size is not the best choice for the exploratory analysis, but it is due to the obstacles (as mentioned in the next slide). The results of the exploration are as follows, where it could be seen that to observe the speed up of Quantum algorithm compared to the classical Bruteforce, problems of greater size should be taken. The Jupyter Notebook is submitted with the report for reference

Algorithm	Time (5 nodes)	Time (10 nodes)	Time (15 nodes)	Success Rate
Clasical	34 μs	3.09 ms	225 ms	100%
QAOA	364.02 s	436 s	443 s	97.47%
QAOA (simulator)	96 s	218 s	2158.3 s	98.96%
Optimal	34 μs	3.09 ms	225 ms	100%

Challenges Faced

Challenges

- The memory requirement of a Quantum simulator increases exponentially as the number of qubits increase, since memory needed $= 16 bytes \times 2^n$ where n is the number of qubits.
- The Time taken to simulate a circuit increases exponentially with the number of qubits and is astronomically large for $n > 24$.
- The biggest challenge is that the popular Quantum computers like IBM Quantum, imposes limit on monthly QPU usage ($10 min$ for IBM) and **Session** mode is not there for free-plan user implying that each job has to be individually queued, and so as the problem size increase the optimizer's iteration counts increases, and so number of required jobs increases, increasing the time of whole process of submitting and running jobs for QAOA run.
- Thus, any insightful experiment is practically much time and space consuming.

Thank you!