QAOA for the Number Partitioning Problem

Kaveri Priya Putti

Abstract

Number partitioning - also referred to as the number bi-partitioning problem, or the 2-partition problem - is one of Karp's original NP-hard problems, and can be stated as follows. We have a set S of n numbers, and the goal of the optimisation problem is separate them into two disjoint subsets S1 and S2 such that the difference of the sums of the two subsets is minimised.

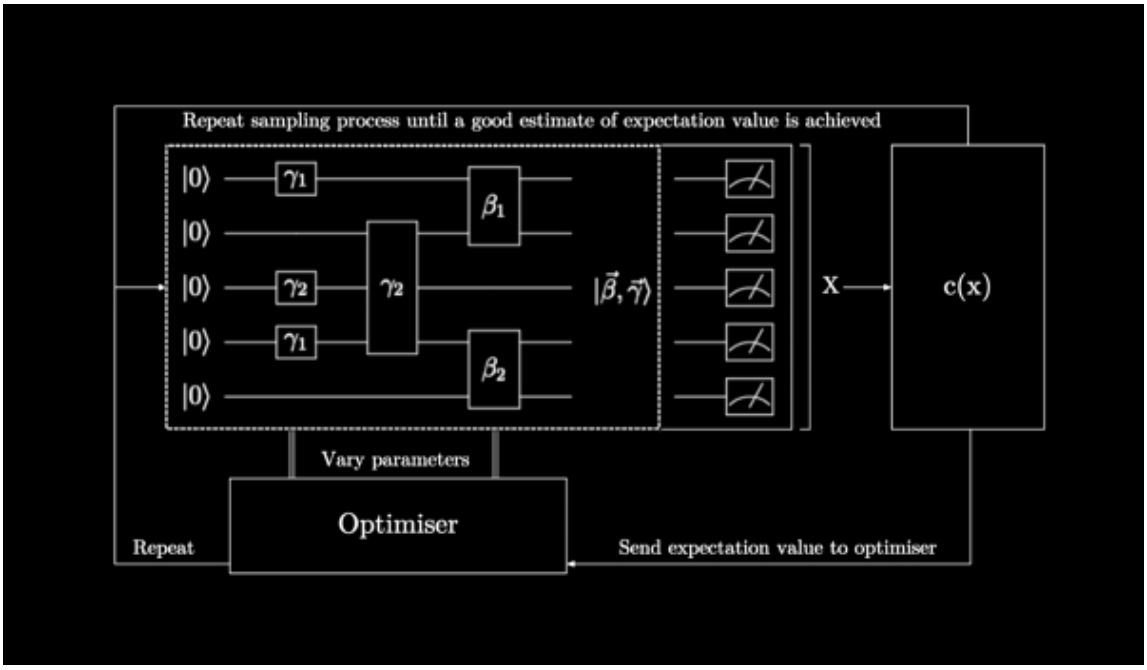
I. Algorithm

The quantum approximate optimization algorithm (QAOA) is a general technique that can be used to find approximate solutions to combinatorial optimization problems, in particular problems that can be cast as searching for an optimal bitstring. Derived the Hamiltonian and unitary operators from source code on this website and theory from here.

1. Define a cost Hamiltonian H_C such that its ground state encodes the solution to the optimization problem.
   $U_C(\boldsymbol{\gamma}) = e^{-i\gamma_p H_C} = e^{-2i\gamma_p \sum_{i<j} s_i s_j \sigma_i^z \sigma_j^z}$
   For qubits (i, j), $U_C(\boldsymbol{\gamma}) = CNOT(i, j)R_z^j(-4\gamma s_i s_j)CNOT(i, j)$
2. Define a mixer Hamiltonian H_M.
   $U_M(\boldsymbol{\alpha}) = e^{-i\alpha_p H_M} = e^{-i\alpha \sum_i \sigma_i^x}$
3. Construct the circuits $e^{-i\gamma H_C}$ and $e^{-i\alpha H_M}$ and call them the cost and mixer layers, respectively.
4. Choose a parameter $p \geq 1$ and build the circuit
   $U(\boldsymbol{\gamma}, \boldsymbol{\alpha}) = e^{-i\alpha_p H_M} e^{-i\gamma_p H_C} \ ... \ e^{-i\alpha_1 H_M} e^{-i\gamma_1 H_C}$,
   consisting of repeated application of the cost and mixer layers.
5. Prepare an initial state, apply $U(\boldsymbol{\gamma}, \boldsymbol{\alpha})$, and use classical techniques to optimize the parameters.
6. After the circuit has been optimized, measurements of the output state reveal approximate solutions to the optimization problem.

The starting point of the QAOA algorithm is the specification of cost and mixer Hamiltonians and then the usage of time evolution and layering to create a variational circuit and optimize its parameters. The algorithm ends by sampling from the circuit to get an approximate solution to the optimization problem.

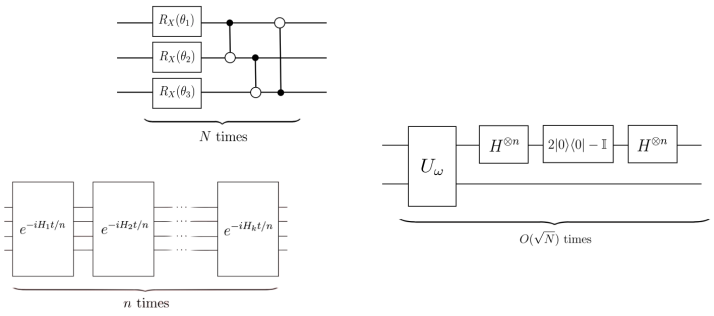
II. Approach



Most of the resources online were QAOA for graph partitioning and there were hardly any resources for number partitioning. Initially, I worked towards implementing QAOA with Qiskit Aqua and reading their QAOA solver's source code to adapt it to a class. After implementing it with Aqua, I worked on building the class with the information from the pseudocodes, articles and flowcharts linked in the references.

III. Analysis of performance with respect to layers

Ideally, the performance of QAOA should increase with increase in the number of layers (reference paper) but as I sampled multiple times with both codes — QAOA class and QAOA Aqua — the results seemed to be a little unstable but overall, the probabilities of correct subset increased from 16.4% (p = 1) to 21.5% (p = 2) to 28.3% (p = 3) for the QAOA class. However, the probabilities for Qiskit Aqua were significantly lower and relatively unstable.



IV. References

1. https://faculty.washington.edu/aragon/pubs/annealing-pt2.pdf
2. https://dl.acm.org/doi/pdf/10.5555/3408352.3408509
3. https://qiskit.org/documentation/_modules/qiskit/optimization/applications/ising/partition.html#get_operator
4. https://supercomputingchallenge.org/18-19/finalreports/29/Using_QAOA_to_Solve_NP_Hard_Problems_on_NISQ_Computers.pdf
5. https://pennylane.ai/qml/demos/tutorial_qaoa_intro.html