FliRe

A way to flight reassignment

Documentation

By Team 22

GitHub Repo Link: https://github.com/alphabet27/mphasis-team22.git
Built for Inter IIT Tech Meet
Hosted by IIT Madras
PS: Quantum Computing Challenge - Mphasis Foundation

Contents

1 Introduction

Introducing the FliRe Documentation, your comprehensive guide to navigating our innovative codebase. FliRe isn't just an application; it's a dynamic flight passenger re-accommodation solution revolutionizing how the most optimized way can be found to use seats on scheduled flights to re-accommodate the passengers of the affected flights. At its core, our solution harnesses the superlative powers of quantum computing and its algorithms to solve one of the most challenging problem of affected flight passenger reallocation that classical computers are unable to solve reasoning to its complexity and limited power of classical computers.

FliRe serves as a robust application, giving the most optimized solution for the affected passenger re-accommodation based on the constraints generally used by airlines. Our solution uses a quantum computing-based approach for calculating the fastest alternate route among scheduled flights and passenger reallocation based on class, cabin, type, loyalty, etc. In this way, the maximum number of passengers will get the best alternative flight against the canceled flight.

A preliminary implementation of our system is provided in the GitHub repo. We have deployed it using the python streamlit library, creating a simple but elegant web interface. Full flexibility of business rules is also provided. On the webpage, one can modify almost every single rule, and the algorithm will take them into account while calculating the PNR best flight for each PNR. It requires the Inventory data as csv, PNR booking details and PNR passenger details. It also requires the cancelled flight(s) as input.

In the backend, we are using the following libraries - Networkx (for graphs in python), Numpy(for linear algebra), Pandas (dataframe management), PennyLane (for quantum computing), matplotlib (for visual representation of graphs) and streamlit for deployment.

2 Project Overview

This project is a quantum computing-based approach solving the challenging problem of the passenger reallocation of the affected flights. It works to find the fastest alternate routes among the available routes and helps to reallocate as much as possible number of passengers. It will find the most optimized solution that will benefit airline companies as well as passengers friendly at the same time.

Our main approach is based on the QAOA algorithm of quantum computing and also involves some of the classical-based pre and post-processing. The main skeleton of our solution can be explained thoroughly as follows. The solution will first involve the pre-processing of the data to remove all the unusable flights from it. These include those who have either departed or cannot be used due to high delays or other issues. This involves several conditional-based filtering of data to remove all the unusable flights. This will make data more precise and more easily processable by the quantum-based algorithm. Next, it will use the QAOA algorithm to find the best alternate route among all the available routes keeping various constraints. Our solution will use a directed graph-based approach consisting of airports with date-time as nodes and flights as edges. The weights of graph edges will be based on the constraints of delay and QAOA will help in finding a route with a minimum weight sum. This will help find the best route in terms of time and satisfying several conditions at the same time. It will allocate passengers using classical post-processing based on the preferences of airline companies. Preference-based allocation will help airlines to first consider more important customers and then lesser ones. This includes specially-abled ones, loyalty-based, class-based, etc

QAOA is the best algorithm found by us while reviewing the scientific literature available on finding the most optimized routes among available ones. The weights of our graphs, also stated before are calculated based on the constraints that the route suggested should follow. Post-processing is a less complex task and classical algorithms are sufficient for them.

3 QAOA algorithm

We have used the QAOA algorithm in our solution. As described earlier we have constructed a directed graph of all the usable scheduled flights and nodes represent airports while edges represent flights. QAOA stands for Quantum Approximate Optimal Algorithm. The graph is solved by modeling the problem as the shortest path problem(SPP). The weights of the graph edges are dependent upon the constraints of various delays and other criteria. It has also been used in the context of graph based problems such as the MaxCut problem [2] and the Travelling Salesman Problem [4].

QAOA is the leading example of a VQA for combinatorial optimization[2]. Variational Quantum Algorithms(VQAs) use a sequence of relatively short quantum circuits with parameters that are iteratively updated by a classical optimizer. VQAs have been designed for a wide range of problems, such as ground state and excited state preparation, quantum state diagonalization, quantum data compression, quantum fidelity estimation, and quantum compiling

QAOA algorithm is mainly used to solve combinatorial optimization problems and provide approximate optimal solutions for them[2]. The execution efficiency of the algorithm mainly depends on the number of iteration steps p. As p increases, the approximation becomes higher and the circuit depth becomes deeper. Recent research results show that QAOA is also expected to achieve quantum computing superiority. The QAOA circuit has a distinct structured character. It starts from a certain initial state and goes through p steps of QAOA subcircuit to reach the final state.

3.1 Shortest Path Problem

We have modeled our alternate route finding problem as the shortest path problem(SPP) in the graph-based approach. SPP consists of three parts: the definition domain, the constraints, and the objective function. The proposed algorithm aims to solve a peer-to-peer shortest path question, which is to find the shortest in a directed path G with n nodes, m edges weighted w_{ij} for node i to node j. The nodes set is V, while the edges set being E and the weights set being W. SPP aims to find the shortest from the source edge to the destination edge.

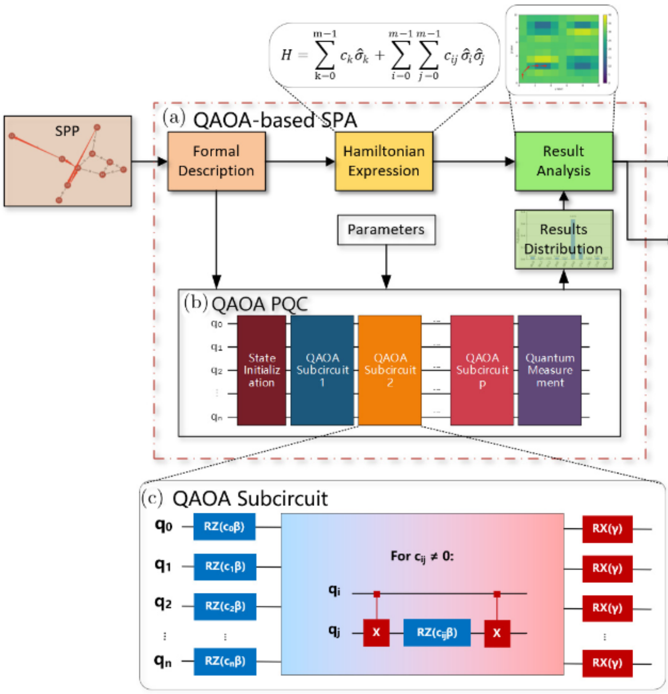
3.2 Problem Formulation

QAOA can handle combinatorial optimization problems, so solving the shortest path of a graph with QAOA requires first transforming the problem into a combinatorial optimization problem. The goal of the combinatorial optimization problem is to find the optimal solution from the feasible solution space of the combinatorial problem, which mainly consists of constraints and objective functions

We have represented the schedule of the alternate flights between different airports as the directed graph. It has nodes representing the source airport and destination airport of the flight at different dates and times and edges representing the flights between corresponding airports.

We have calculated the weight of the directed graph edges using the arrival delays, STD of the proposed flights, and the stopover of the proposed flights. We modeled the graph as the combinatorial optimization problem. It has different constraints and in this way, QAOA helps to find the most optimized alternate flight route between the source and destination airport for the passengers of the affected flights.

In our implementation, inspired by [1] each edge of the graph is assigned a qubit. We have then used the Hadarmard gate, RZ gate and CNOT gates to define a quantum circuit. Our Hamiltonian function consists of two parts, the cost Hamiltonian and the mixer Hamiltonian. The cost Hamiltonian represents which path is being selected (weights consideration) while the mixer Hamiltonian is responsible for the s



In our case, the cost hamiltonian is given by

$$H = \sum_{k=0}^{m-1} c_k \sigma_k + \sum_{i=0}^{m-1} \sum_{j=0}^{m-1} c_{ij} \sigma_i \sigma_j \tag{1}$$

and the optimization function is given by

$$\bar{H} = \sum_{z=|0\rangle^{\otimes m}}^{|1\rangle^{\otimes m}} O(z) \times P(z) \tag{2}$$

A novelty we have added is that in this algorithm, if we know that for a particular edge, the probability of it not being in the final output is greater than 0.5, then it is automatically removed from the graph, during the optimization process itself. This has been achieved by temporarily defining a new Pennylane qnode with a different cost function, one which returns the measured value of each qubit, instead of the actual cost function. Once this is evaluated, the original qnode is called back, this way, we can remove an edge while the parameters are still retained. Therefore, a problem utilizing n-qubits initially eventually reduces to n-1 qubits, n-2... and so on until only the actual shortest path remains. This essentially ensures that the path found by the algorithm is actually the shortest path, and not an approximate solution, as it is much easier to minimize a lower-dimensional variable as compared to a higher one.

4 Pre-Processing of data

Our solution will have the classical pre-processing of the raw data to make it in a more precise form. It will make the data more easily processable by our quantum computing algorithm.

First, we will remove all the flights that don't satisfy the minimum stop-over-time criteria. We will also remove the flights that are not usable. It includes those flights which are arriving at the source airport or which are departing from the destination airport. We will also remove the flights that have larger arrival delays or larger scheduled time departure than the maximum tolerable limit. This all will be taken as input in our program. Taking them as input will give the flexibility to set the maximum tolerable delays.

The inventory file will be input to the program having details of scheduled flight inventory as well as canceled flight inventory. Our program will also take the passenger data in the form of PNR details of passengers, and passenger booking details as input. The index of flights that are canceled from the inventory data is also to be fed. This will let us prepare the data for the passenger list of the affected flights.

We will sort the passenger data concerning their classes, cabins, type of passenger, and other details at the time of booking. This will help us to make it manageable and also will make it more usable to be used during reallocation. During reallocation, passengers will be reallocated concerning various parameters and pre-processing of data accordingly makes task more manageable.

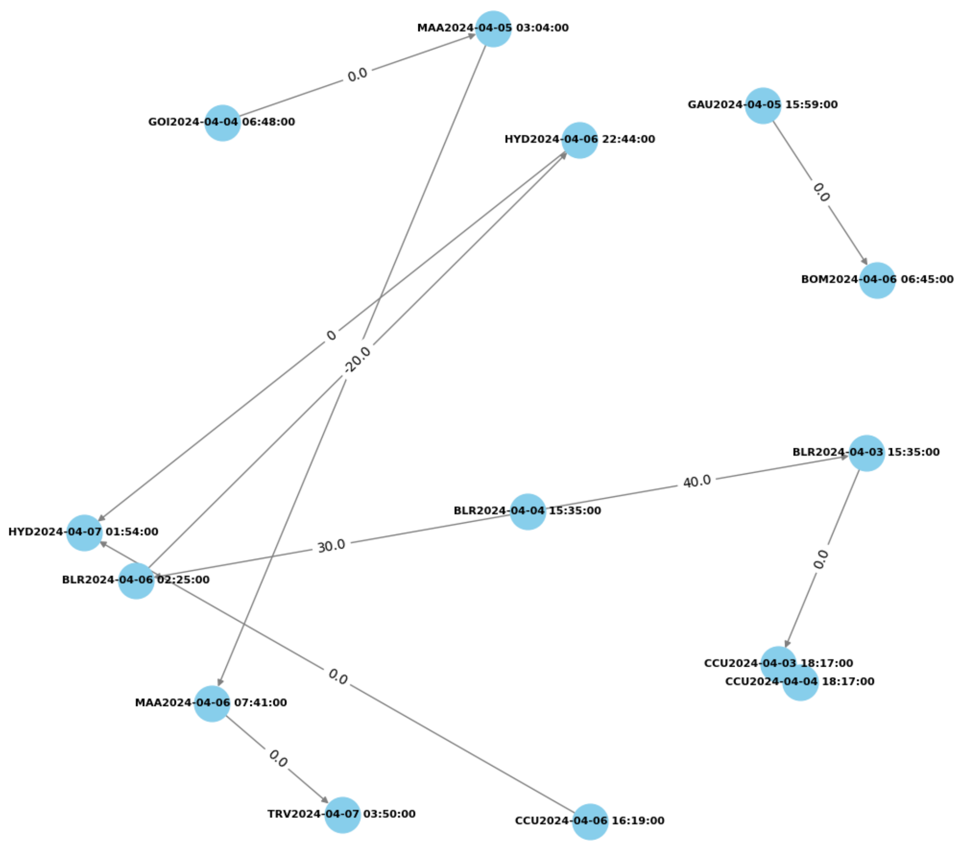
All processing of such sort is done using the pandas python library internally. At the user end, one just has to upload the Inventory file, the PNR Booking file, and the PNR Passenger file. In our streamlit implementation, it's as easy as clicking a button.

5 Graph Description

As stated, we have modeled the schedule of the available alternate flights as a directed graph. Each node of the graph represents a unique (Airport, Time) pair. Edges of the graph represent the scheduled flights between two different airports. Graph nodes have airport names with the date and time of the scheduled flight from the airport.

One airport has multiple flights with different dates and times. We have represented the airport as having flights at different dates and times with different nodes. Figure of the graph below can more easily explain that. Edges are also drawn between the canceled flight airport node and the same airport node but having flights at different dates and times. These edges are used to represent the time delay weights between the canceled flight and the alternate flight.

Graph weights are calculated based on the arrival delay of the alternate flight schedule time delay of the proposed flight and the stopover of the proposed flight. The algorithm will find the route having the minimum weight between the source and the destination airport.



6 Black Box

Running of the QAOA algorithm in the program and the post-processing of the output is taking place in the black box program. It consists of the alternate route-finding algorithm as well as the passenger reallocation program. The black box is a sort of master function, which takes the modeled graph, the source and destination (from canceled flight data), the Minimum Connecting Time between 2 flights, a sorted list of affected PNRs(based on PNR Ranking criteria), and other such constraints as input.

The following is a fundamental process of the black box -

1. Find the best route in the current formulated graph.

2. In this best-selected route, try to fill as many PNRs as possible, with the PNR with higher priority going first.

3. Update the capacity of each flight in the best route (a route may consist of multiple flights), until either the PNR list is empty (all PNRs reallocated) or the flight is full.

4. Remove the edges from the graph which are redundant (have full capacity).

5. If the PNR list is non-empty, repeat this process until either the list is empty or the graph is empty.

The above fundamental process is repeated eight times, in an orderly fashion. In the first 4 iterations, each cabin(Economy, Premium Economy, Business, First) is handled individually. (This is because the capacity is defined for each cabin, and if we handle multiple cabins simultaneously, we may remove an edge because it does not contain the capacity of one particular class, but it may be useful for another PNR for a different cabin). In these initial 4 iterations, we force the algorithm to find only those solutions where the cabin of the PNR remains unchanged, as this should be the first priority.

In the next 4 iterations, we do the same thing, just that this time we allow Upgrade/Downgrade rules, as required by the airline.

This is a rudimentary overview of how the black box is functioning. In this algorithm, calls upon the PNR Sorting function, and the Shortest Path Algorithm function, which are themselves calling multiple functions within them.

7 PNR Sorting and Allocation function

The main job of the PNR sorting function is to fetch the data of the affected PNRs and then sort them according to the various scoring criteria provided by the airline. This function takes in the 'Flight Number' and the 'Departure Date' of a canceled/delayed flight and then the PNR data is fetched from the PNR Booking and PNR Passenger files. Once all the data is fetched, a score value is calculated for each PNR according to the flexible scoring criteria provided at the start. This function also takes into account if a particular PNR has a connecting flight before or after the canceled/delayed flight and ensures that there must be a time buffer between them and the proposed alternate flight.

The fundamental working of this function is explained as follows:

1. Fetch the data of affected PNRs.

2. Calculate the score of each PNR according to the scoring criteria provided.

3. Check if the PNR has any connecting flight adjacent to the affected flight.

4. Define the maximum possible time to reach the destination and the minimum possible time to depart from the source airport, according to the Minimum Connect Time (MCT) provided.

5. Sort the affected PNR data according to the scores.

6. Return the sorted data

In one call of this function, a list of PNRs affected due to 1 cancellation/delay are calculated. Thus, for multiple cancellations, it can simply be called multiple times providing the details of each affected flight.

8 Alternate Approaches of solving problem statement

There are many alternate approaches also possible for solving the problem of passenger reallocation of the affected flight.

Some of them are listed here:

1. Quantum Annealing:

Quantum annealing is a quantum computing approach that leverages quantum fluctuations to find the global minimum of an objective function. This can be useful for optimization problems where we are trying to minimize a cost or maximize an efficiency metric. We can formulate the passenger reallocation problem as an optimization problem with an objective function that represents the cost or discomfort associated with passenger assignments. We can use a quantum annealer, such as D-Wave, to find the optimal assignment of passengers that minimizes the objective function. People have in fact used quantum annealing to solve various kinds of optimization problems such as traffic flow optimization in [3]

2. Variational Quantum Eigensolver (VQE):

VQE is a hybrid quantum-classical algorithm that can be used for optimization problems. It involves a variational quantum circuit that is optimized using classical algorithms. We can map the passenger reallocation problem to a Hamiltonian, and then use VQE to find the ground state of the Hamiltonian, which corresponds to the optimal solution.

3. Quantum Machine Learning:

We can utilize quantum machine learning algorithms to analyze and predict passenger behavior, preferences, and travel patterns. Combine classical machine learning models with quantum-enhanced algorithms to improve the accuracy of passenger predictions, which can then be used in the optimization process. Skolik 2023 [4] have used Quantum Neural Network based approach to solve their Travelling Salesman Problem.

References

[1] Zhiqiang Fan et al. "Solving the Shortest Path Problem with QAOA". In: *SPIN* 13 (1 2023). ISSN: 20103255. DOI: 10.1142/S2010324723500029.

[2] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "A quantum approximate optimization algorithm". In: *arXiv preprint arXiv:1411.4028* (2014).

[3] Florian Neukart et al. *Traffic flow optimization using a quantum annealer*. 2017. arXiv: 1708.01625 [quant-ph].

[4] Andrea Skolik et al. *Equivariant quantum circuits for learning on weighted graphs*. 2023. arXiv: 2205.06109 [quant-ph].