Inter IIT Tech Meet 12.0

Final Evaluation Report For Mphasis
Team ID: 65

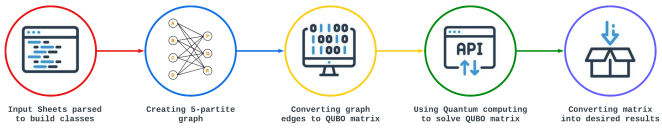

Figure 1: Flow Graph

I. Introduction

In the realm of airline operations, effectively navigating the complexities of planned schedule changes is paramount. This report offers a refined solution to efficiently identify optimal alternate flight options for affected passengers, ensuring strict adherence to established rule sets. The primary objective is disruption minimization and service quality preservation, with a focus on factors such as travel time, ancillary service impact, and prioritization based on passenger attributes. Addressing the dynamic aviation landscape, our report introduces a systematic approach integrating optimization constraints and pre/post-processing calculations to enforce pertinent rule sets. Recognizing the diverse passenger profiles, the report outlines a prioritization mechanism, encompassing factors such as passenger type, employee duty status, loyalty program levels, and class of service. This comprehensive ranking system ensures that the re-accommodation process caters to the specific needs and preferences of individual passengers, bolstering operational resilience. Through this systematic analysis and optimization approach, airlines can streamline re-accommodation processes, ultimately elevating the overall passenger experience. This report aims to contribute valuable insights and methodologies for addressing challenges tied to planned schedule changes in the aviation industry.

II. Problem Statement

In the face of planned schedule changes that have impacted a subset of passengers, the imperative is to determine the optimal and alternative flight solutions. The objective is twofold: firstly, to adhere meticulously to the provided rule sets ensuring the validity of the chosen solution, and secondly, to rank these solutions based on a myriad of factors. These factors include critical parameters such as time to reach the destination and the consequential impact on purchased ancillary services.

The optimization challenge lies in orchestrating a seamless re-accommodation strategy that takes into account the diverse needs of the affected passengers. This necessitates a meticulous consideration of variables such as passenger type, encompassing designations such as unaccompanied minors and on-duty employees, customer loyalty levels, and the paid class of service.

Our approach to this optimization conundrum involves enforcing the rule sets either as integral components of the optimization constraints or as part of a pre or post-process calculation. This dual strategy is crafted to ensure the utmost compliance with the specified rules while maintaining the efficiency and effectiveness of the re-accommodation process.

The resultant solutions will be methodically ranked based on a holistic evaluation encompassing temporal considerations, ancillary service impacts, and a nuanced prioritization of passengers according to specified criteria. This rigorous methodology seeks to deliver not just alternative flight options but a meticulously curated hierarchy of solutions, tailored to the individual needs of the passengers and the overarching objectives of the airline.

III. Literature Study

A. Solving assignment problems via Quantum Computing: Train Seating Arrangement

The paper [1] introduces the use of quantum computing to solve assignment-type combinatorial optimization problems. Specifically, it considers a case study of optimizing seat assignments on high-speed trains under social distancing constraints due to COVID-19.

Figure 2: Train Seating Arrangement Problem

The key motivation is that combinatorial optimization problems are computationally complex but have many real-world applications. Quantum computing is an emerging technology that can potentially solve such problems more efficiently.

The specific contributions are:

- Formulation of the train seat assignment problem as a quadratic unconstrained binary optimization (QUBO) model which is suitable for quantum solvers
- Solution using QBSolv solver on classical hardware and D-Wave's quantum-classical hybrid solver
- Comparison of performance in solution quality and computational times

1) Background: The paper provides background on two key concepts:
   - Quadratic Unconstrained Binary Optimization (QUBO): The standard QUBO formulation is shown in Equation (1) below:

$$\min_{x\in\{0,1\}^{|N|}} x^T Q x \tag{1}$$

     where x is a binary decision vector and Q is a symmetric coefficient matrix. The constraints and objective function of an optimization problem can be encoded into the Q matrix.
   - Quantum Annealing: The key idea is that quantum fluctuations can help tunnel through barriers in an optimization landscape to reach low energy states quickly. D-Wave systems provide quantum annealers specialized for QUBO problems.
2) Problem Description: The specific problem considered is assigning passengers to train seats under social distancing constraints. The sets, parameters, and decision variables are defined. The optimization objectives and constraints are modeled as a quadratic program in Equations (3) to (8).

$$\max \quad \sum_k \sum_{(r,c)} x_{(r,c),k} \cdot x_{(r+1,c),k} \tag{2}$$

$$+ \sum_k \sum_{(r,c)} x_{(r,c),k} \cdot x_{(r,c+1),k} \tag{3}$$

$$\text{s.t.} \quad \sum_{(r,c)} x_{(r,c),k} = n_k, \quad \forall k \in K \tag{4}$$

$$\sum_k x_{(r,c),k} \leq 1, \quad \forall r \in R, c \in C \tag{5}$$

$$x_{(r,c),k} \cdot x_{(r+1,c),k'} = 0, \quad \text{if } k \neq k' \tag{6}$$

$$x_{(r,c),k} \cdot x_{(r,c+1),k'} = 0, \quad \text{if } k \neq k' \tag{7}$$

$$x_{(r,c),k} \in \{0,1\}, \quad \forall r, c, k \tag{8}$$

Lastly, in most cases both solvers find equally good solutions. For larger instances, the hybrid solver finds better solutions in lesser time. The quantum advantage is more significant as problem size increases. The paper demonstrates the applicability of quantum computing to assignment problems, using train seat allocation as a case study. Formulating as QUBO models allows the use of quantum tools like QBSolv and D-Wave hybrid solvers. Further research can apply this approach to other large-scale assignment problems.

B. Airline Gate-Scheduling Optimization using Quantum Computers

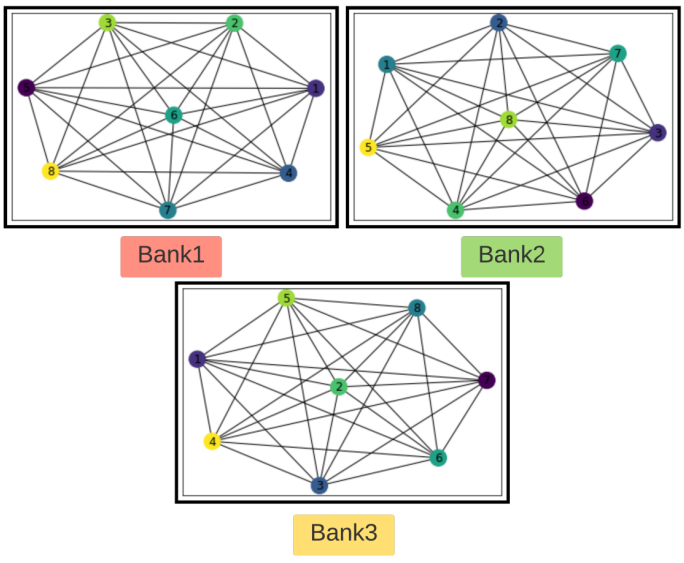

Figure 3: Airline Gate Scheduling Problem

The paper explores using quantum computing to solve airline gate assignment problems formulated as quadratic assignment problems (QAPs). Gate assignment is computationally complex with many real-world constraints. The paper [2] implements gate scheduling on quantum simulators and hardware to demonstrate potential speedups.

The specific contributions are:

- Formulation of gate scheduling as a QAP optimization problem.
- Implementation using the variational quantum eigensolver hybrid algorithm.
- Addition of graph coloring constraints for efficient embedding.
- Experimental demonstration on IBM quantum systems.

1) Gate Scheduling QAP Formulation: The gate scheduling problem is formulated as a QAP in Equation 9:

$$\min_{x \in 0,1^{|P|\times|G|}} \sum_{i,j \in P} \sum_{k,l \in G} x_{ik} x_{jl} v_{ij} d_{kl} \tag{9}$$

Where $x_{ik} = 1$ means plane p_i is assigned to gate g_k. The objectives are to minimize passenger walking distances $v_{ij}d_{kl}$ between gates. Constraints ensure valid allocations.

2) Quantum Optimization: The paper utilizes the variational quantum eigensolver (VQE) hybrid algorithm for optimization. The key idea is to iteratively prepare a parameterized trial state $|\psi(\theta)\rangle$ and minimize the energy expectation value:

$$E(\theta) = \langle \psi(\theta)| H |\psi(\theta)\rangle \tag{10}$$

where H is the problem Hamiltonian. A classical optimizer updates θ until convergence.

C. Quantum Metropolis Sampling

The Metropolis algorithm is a seminal method for sampling configurations of classical systems according to their Boltzmann distribution. However, direct simulations of quantum systems suffer from the sign problem. Quantum computing promises efficient simulations of quantum systems. The key open problem is preparing physically relevant states like thermal states and ground states.

The paper makes the following contributions:

- A quantum version of the Metropolis algorithm suitable for quantum computers
- A proof that the algorithm samples from the Gibbs ensemble
- Analysis of convergence rates
- Potential implementation with small scale quantum devices

The thermal Gibbs state is:

$$\rho_G = \frac{e^{-\beta H}}{Z} \tag{11}$$

Where H is the system Hamiltonian, β is the inverse temperature and Z is the partition function.

The paper [3] presents a quantum algorithm that performs a random walk over energy eigenstates of the Hamiltonian to sample the Gibbs distribution. The key insight is using quantum phase estimation coherently and tailored measurements to implement the accept/reject step without collapsing the wavefunction.

The thermal state samples produced replicate averages over the actual energy eigenstates rather than a classical mapping. This allows simulation of fermionic systems as well. Convergence rates and implementation challenges on near-term quantum hardware are analyzed.

The algorithm serves as a potential route for preparing arbitrary thermal and ground states on quantum computers to enable efficient quantum simulations.

IV. Goals of the Problem Statement

A. Minimization of Unresolved PNRs

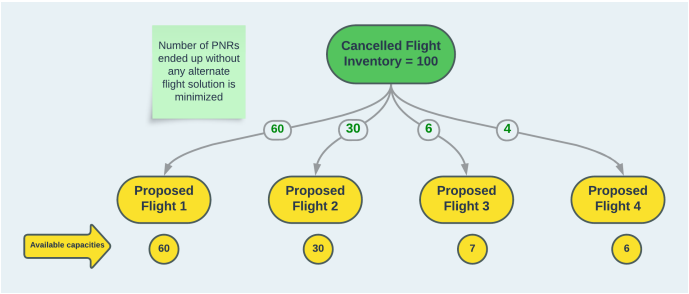

Figure 4

- **Objective:** The primary objective of this optimization effort is to minimize the occurrence of unresolved PNRs, ensuring that each passenger affected by schedule changes is provided with a viable alternate flight solution. By implementing strategic algorithms and prioritization mechanisms, the system aims to efficiently

assign optimal flight alternatives, thereby enhancing passenger satisfaction and operational resilience.

- **Mathematical Formulation:**

Decision Variables:
Let x_{ij} be a binary variable representing whether Passenger i is assigned an alternate flight solution j. The decision variable takes the value 1 if the assignment is made, and 0 otherwise.

Objective Function:
Minimize the total number of unresolved PNRs:

$$\max \left(\sum_i \sum_j x_{ij} \right) \tag{12}$$

Constraints:
Each passenger is assigned to at most one alternate flight:

$$\sum_j x_{ij} \leq 1, \text{ for all } i$$

A passenger is assigned only if there is a valid alternate flight solution:

$$x_{ij} = 1, \text{ if flight solution } j \text{ is valid for } i$$

- ***Rationale:***
PNR reallocation, ensures a smooth transition for passengers impacted by schedule changes. This sophisticated operation reflects the commitment to minimizing disruptions and maintaining the highest standards of service excellence.

B. Adherence to Default Solution with Capacity Flexibility:

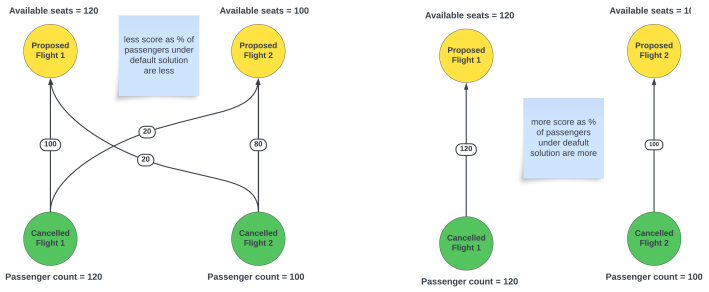

Figure 5

- **Objective:** Prioritize strict adherence to the flight-level default solution for PNR reallocation, with the flexibility to accommodate deviations in cases where capacity constraints demand nuanced adjustments.

- **Mathematical Formulation:**

Decision Variables:
Let y_{kj} be a binary variable representing that k flight is rescheduled on flight j and taking x_{ij} from previous goal.

Objective Function:
Maximize strict adherence to the flight-level default solution while allowing for flexibility:

$$\max \sum_i \sum_k y_{kj} * x_{ij} \tag{13}$$

Constraints:
Capacity constraint adjustments:

Flexibility Constraint: $\sum_i y_{ij} \leq$ Available Capacity

- **Rationale:** Default solutions provide a reliable baseline, and any deviations must be meticulously managed to uphold consistency while navigating the complexities introduced by varying operational capacities.

C. Minimization of Delays:

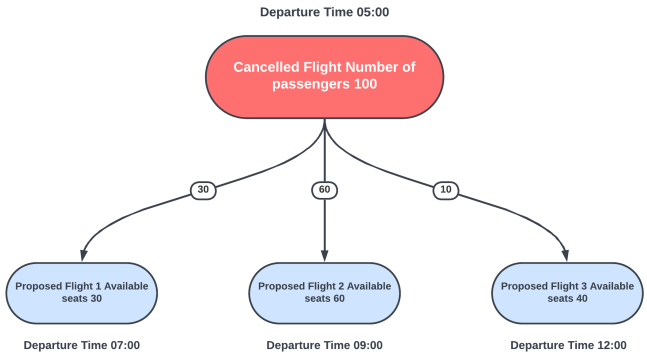

Figure 6

- **Objective:** Systematically minimize the overall delay attributable to the re-accommodation process, underscoring the significance of

timeliness in passenger satisfaction.

- **Mathematical Formulation:**

Decision Variables:
Let d_{ij} represent delay for PNR i on flight j.
Objective Function:
Minimize the overall delay attributable to the re-accommodation process:

$$\min \sum_i \sum_j d_{ij} \tag{14}$$

Constraints:

Capacity constraint adjustments:

$$\text{Flexibility: } d_{ij} \leq 72 \text{ hours}$$

- **Rationale:** Timely re-accommodations are paramount in fostering positive passenger experiences, necessitating a meticulous approach that minimizes disruptions and prioritizes efficiency in the face of schedule changes.

D. Customer Class Priority Consideration:

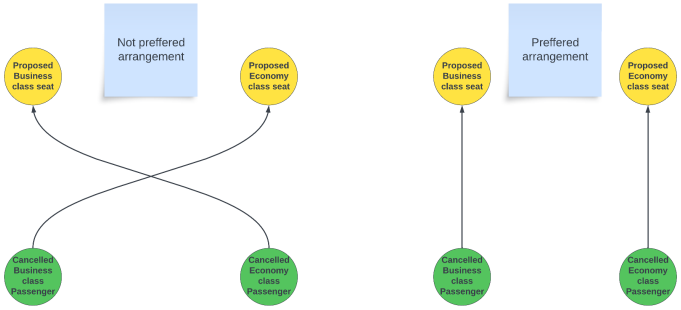

Figure 7

- **Objective:** Integrate nuanced considerations of customer priority into the reallocation process, affording higher precedence to specific passenger categories, such as business class over economy class.

- **Mathematical Formulation:**

Decision Variables:
Let p_{ij} be a binary variable representing whether PNR i is assigned to flight j considering customer class priority. The decision variable takes the value 1 if the assignment is made, and 0 otherwise.
Objective Function:
Maximize the overall customer class priority in the reallocation process:

$$\max \sum_i \sum_j \text{Priority Weight}_{ij} \cdot p_{ij} \tag{15}$$

Constraints:
Capacity constraint adjustments:

$$\sum_i p_{ij} \leq \text{Available Capacity}$$

- **Rationale:** The differentiation of priority levels allows for personalized re-accommodations, acknowledging the diverse requirements and preferences of passengers occupying distinct service classes.

E. Preference for Direct Flights:

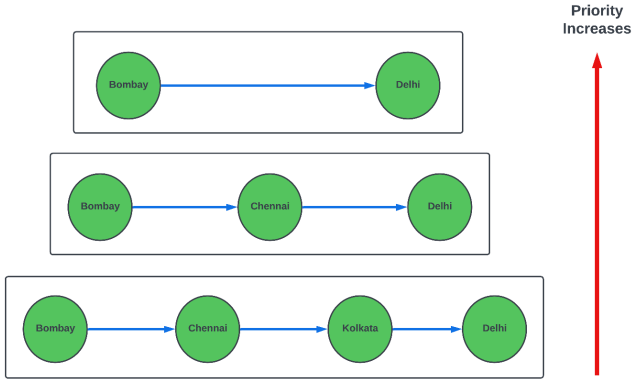

Figure 8

- **Objective:** Establish a pronounced preference for direct flights over connecting flights in the re-accommodation process, accentuating the advantages of reduced travel time and heightened passenger convenience.

- **Mathematical Formulation:**

Decision Variables:
Let d_{ij} be a binary variable representing whether PNR i is assigned to a direct flight j. The decision variable takes the value 1 if the assignment is made, and 0 otherwise.
Objective Function:
Maximize the preference for direct flights over

connecting flights in the re-accommodation process:

$$\max \sum_i \sum_j \text{Direct Flight Weight}_{ij} \cdot d_{ij} \quad (16)$$

- **Rationale:** The favoring of direct flights addresses the paramount concerns of passengers, emphasizing the importance of streamlined travel and reduced complexities inherent in connecting flights.

V. Approach

A. *Problem Reduced to Graph Representation*

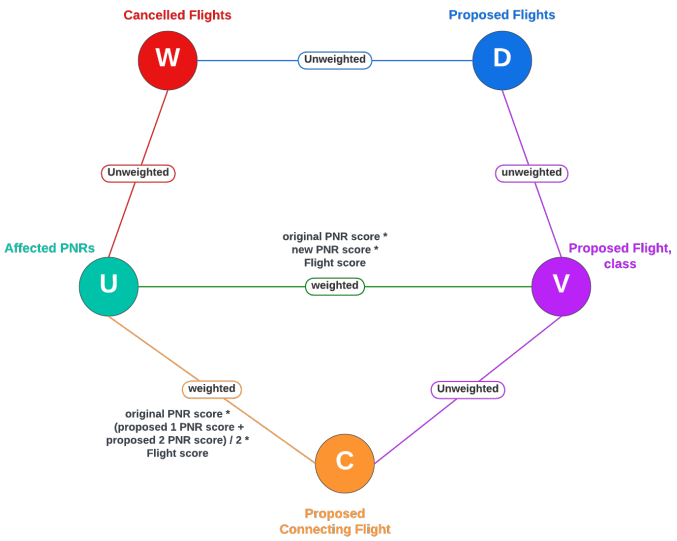

Figure 9: Graphical Representation

Here,
W represents set of nodes denoting canceled flights.
U represents set of nodes denoting affecting PNRs.
D represents set of nodes denoting proposed flights.
V represents set of nodes denoting direct proposed flight, class.
C represents set of nodes denoting Connecting proposed flights,class.

B. *Final Constraints Deduced:*

1) At most one alternative flight solution is selected for each passenger in U, representing all impacted PNR(s) which ensures an effective reallocation of alternate flights for every affected passenger.Here $x_{u,v}$ and $x_{u,c}$ are boolean representations of existence of edge in (U, V) and (U, C) respectively.

$$\forall u \left(\sum_{v \in V} x_{u,v} + \sum_{c \in C} x_{u,c} \leq 1 \right) \quad (17)$$

2) It is imperative to ensure that seat allocations do not exceed the available number of seats, maintaining a meticulous alignment between the number of seats allocated and the actual capacity.

$$\forall v \left(\sum_u (w_u \cdot x_{u,v} + \sum_{(c,v) \in E} \sum_c w_u \cdot x_{u,c}) \right) \leq w_v \quad (18)$$

3) Let D denote the set comprising all proposed flights for rescheduling, and W represent the set of cancelled flights. A critical constraint demands that each cancelled flight, residing within set W, be exclusively assigned to a single new rescheduled flight from the set D.

$$\forall w \left(\sum_d x_{w,d} \leq 1 \right) \quad (19)$$

C. *Proposed Scoring Function*

The proposed scoring function for Optimisation is as follows:

$$f(N, M, \lambda) = N \cdot n_{PNR} + M \cdot n_{Passenger} + \sum s_v + \sum \frac{s_c}{\lambda_c}$$

where,
n_{PNR} = Number of PNR reallocated;
$n_{Passengers}$ = Number of Passengers allocated through default solution;
s_v = Score of Edges selected in (U, V);
s_c = Score of Edges selected in (U, C).
λ_c = Number of Flights in C.

Here, N, M are scaling factors for each individual property which are used to quantify its priority. The Pseudo Code for finding out above variables is given below.

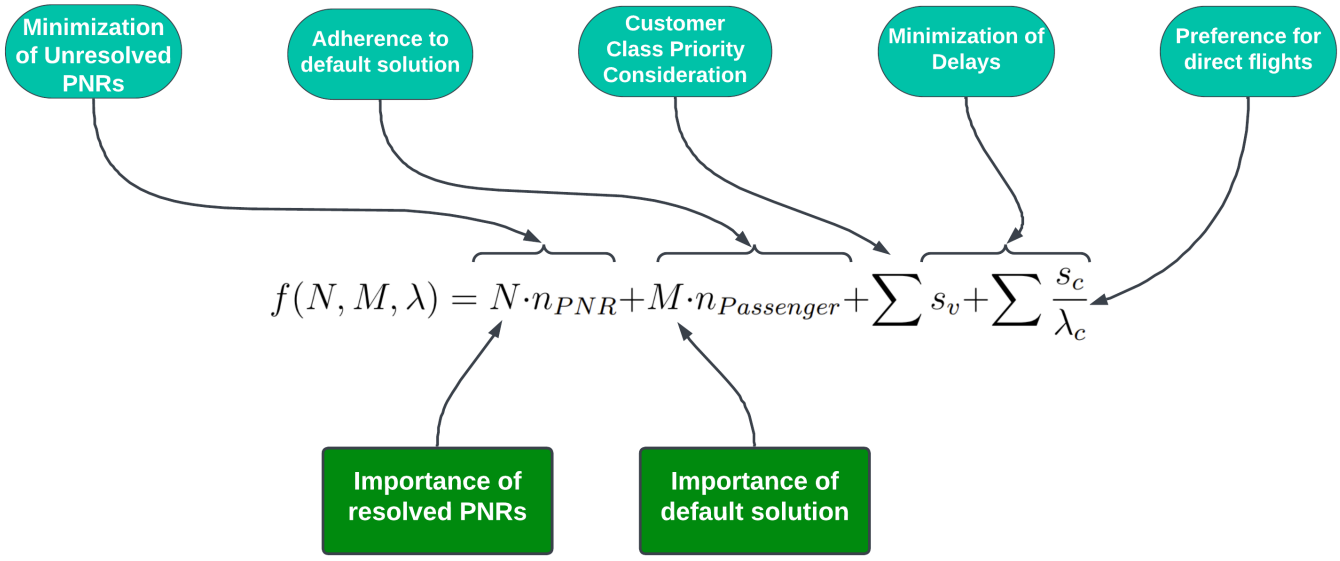

Figure 10: Here's how goals are included in Scoring function.

Algorithm 1 Flight Evaluation Algorithm

procedure ARRDEPTIMEDIFF($flight_1, flight_2$)
    return $arrivalTime(flight_2) - departureTime(flight_1)$
end procedure
procedure
    $AffectedPNRs =$ empty list
    for all (p_id, P) in Pnr_Map **do**
        if P has only one flight and P's flight is in $CancelledFlights$ **then**
            $ImpactedPnr$.append(p_id)
        else
            for all $flight$ in P's flights **do**
                if $flight$ is in $DelayedFlights$ **then**
                    $cond1 = DelayedFlights[flight] >$ Time$(72, 0)$
                    $cond2 =$ ArrDepTimeDiff($flight$, next flight) $<$ Min.ConnectingTime
                    if $cond1$ or $cond2$ **then**
                        $AffectedPNRs$.append(p_id)
                        break
                    end if
                end if
            end for
        end if
    end for
end procedure

Algorithm 2 PNR Score Function

procedure PNRSCORE($journeyId, proposed =$ CD::NIL)
  if $proposed =$ CLASS CD::NIL **then**
    $proposed =$ journeyMap$[journeyId] \rightarrow$ classCD
  end if
  $pnr\_score = 0$
  for all $passenger\_id$ in JourneyToPnrMap$[journeyId] \rightarrow$ passengers **do**
    $pnr\_score+ =$ len(ssrCodes) $\times$ SSR_SCORE + PAX_SCORE
  end for
  return $pnr\_score +$ classScoresMap$[proposed]$
end procedure

Algorithm 3 Flight Score Function

function FLIGHTSCORE(OId, PId)
  $score = 0$
  $score+ =$ ArrDelay LT Score $\times$ ArrTimeDiff
  $score+ =$ DepDelay LT Score $\times$ DepTimeDiff
  if $OriginalEqpNo = Proposed1EqpNo$ **then**
    $score+ =$ Eqp_Score
  end if
  return $score +$ CityPair_Score
end function

Algorithm 4 Connecting Flight Score Function

function CONNECTINGFLIGHTSCORE(OId, PId)
  $score = 0$
  $score+ =$ ArrDelay LT Score $\times$ ArrTimeDiff
  $score+ =$ DepDelay LT Score $\times$ DepTimeDiff
  $t1 =$ Proposed1ArrTime $-$ Proposed1DepTime
  $t2 =$ Proposed2ArrTime $-$ Proposed2DepTime
  if $OriginalEqpNo = Proposed1EqpNo$ **then**
    $score1 =$ Eqp_Score
  else
    $score1 = 0$
  end if
  if $OriginalEqpNo = Proposed2EqpNo$ **then**
    $score2 =$ Eqp_Score
  else
    $score2 = 0$
  end if
  $score+ = \frac{(t1 \times score1 + t2 \times score2)}{t1+t2}$
  return $score +$ CityPair_Score
end function

D. *Connecting Flights*

1) Using Flight Genration : We are finding a list of all valid connecting flight combinations for each affected flight. This is done via the following 3 step process.

label=1)

1) `FlightsfromSrc()` – This function creates a list of all flights that have the same initial source city as the affected flight, within a 72-hour time duration.
2) `FlightsfromDest()` – This function creates a list of all flights that have the same final destination city as the affected flight, within a 72-hour time duration.
3) `ConnectingFlights()` – This function creates a list of all pairs of flight combinations between 1) and 2) which: label=α)
   a) Do not exceed the 72-hour time constraint for alternate flight solutions.
   b) The first flight lands at least 1 hour (`MINIMUM_CONNECTING_TIME`) before the departure of the second flight.

Now that we have a list of all connecting flight combinations, we have to prune the list down and only keep at most 5 alternative flights per affected flight (`MAXIMUM_ALLOWED_CONNECTING_FLIGHTS_PER_JOURNEY`).

This is done to reduce the number of connecting flight nodes in our graph. We will now apply some heuristics to optimally choose the best possible connecting flight combinations out of our list.

This is done by the following process:

`getConnectingFlightScore()` – We will create a function that will heuristically help us compare between alternate connecting flight solutions, by generating a flight score value for each alternate connecting flight.

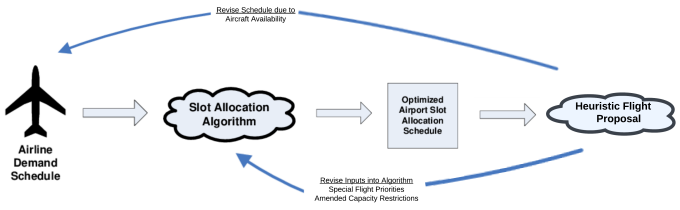

Figure 11: Connecting Flight heuristics

2) Using Hubs: To address the complex challenges inherent in generating connecting flight solutions, our team decided to integrate a heuristic algorithm to optimize the allocation of resources and enhance the overall reliability of our connecting flight solution. This heuristic algorithm is based on the hub and spoke model , which is used commercially by various airlines to route their plane traffic.

Hubs refer to large central airports through which flights are routed , and spokes are the routes that planes take out of the hub airport. Our algorithm first identifies airport destinations which have high incoming and outgoing flight traffic as hubs , and then heuristically generates connecting flight solutions by rerouting the affected passengers source and destination city via a hub . For example , let's say we are in Jaipur and we wish to go to Patna. Since there will be relatively few direct flights between Jaipur and Patna, our algorithm identifies flights from Jaipur to a hub like New Delhi , and a corresponding connecting flight from New Delhi to Patna.

3) Using Multi-Source BFS: This method Multi-Source Breadth-First Search (BFS) approach to efficiently find optimal paths from the set of Passenger Name Records (PNRs), denoted as U, to the final proposed flights with their corresponding classes, represented by set V. The intermediate set of nodes, C, signifies the proposed connecting flights along with their associated classes. The algorithm strategically explores various paths, starting from multiple sources in U and navigating through the potential connecting flights in C before reaching the final proposed flights in V.

The primary objective is to identify the shortest path lengths, ensuring an optimal re-accommodation solution for passengers impacted by schedule changes. This method enhances the algorithm's ability to navigate through complex flight connections and efficiently determine the most suitable alternatives for affected passengers.

We can further optimize this by using heuristic algorithms like A* Algorithm, as well as Contraction hierarchies which are used to find out the shortest paths. We can also use Eppstein's Algorithm to find out multiple shorter paths.

E. *Reduction to QUBO*

PNR scores are calculated based on different business rules. Scores indicate importance of a PNR.

$$S_i : \text{Score of a PNR from flight } i \qquad (20)$$

$$S_{ij} : \text{Score of PNR in flight } j \text{ which was reallocated as a substitute to delayed/cancelled flight } i \tag{21}$$

$$F_{u,v} : \text{Flight Score} \tag{22}$$

$$\text{ES} = S_u \cdot S_{uv} \cdot F_{u,v} \quad \text{(Edge Score)} \tag{23}$$

$$\begin{aligned} S_u = &(\text{class\_score}(u) + \text{SSR\_score} \cdot \#\text{SSR}(u) \\ &+ \text{special\_score} \cdot \#\text{special\_request}(u) \end{aligned} \tag{24}$$

1) Analysing the Scoring Function:

$$\begin{aligned} func = &\left(N \sum_u \sum_v x_{u,v} \right) \\ &+ \left(M \sum_{w \in W} \sum_{d \in D} x_{w,d} \sum_{(d,v) \in E} \sum_{(w,u) \in E} x_{u,v} w_u \right) \\ &+ \left(\sum_{u,v} ES_{u,v} \cdot x_{u,v} \right) \\ &+ \left(\sum_{u,c} ES_{u,c} \cdot x_{u,c} \right) \end{aligned}$$

Table-1 illuminates a salient characteristic of the scoring function, denoting its degree as 2, thereby classifying it as a second-degree polynomial. This classification places it within the realm of a 2D Optimization problem subject to 1D Constraints.

2) Handling the Constraints: The main goal is to reduce [4] a linear inequality constraint $Ax \leq b$ in to an equality constraint and then to a 2D unconstrained QUBO matrix. The key steps are:

1) **Transform the Inequality:**
Transform the inequality $Ax \leq b$ by dividing both sides by a user-defined integer $\rho > 0$ to get:
$$\frac{Ax}{\rho} \leq \frac{b}{\rho}$$
2) **Approximate the Right Hand Side:**
Approximate the right-hand side $\frac{b}{\rho}$ using a single slack variable s_1. Formulate as equality constraint: $A_{\text{mod}} x = s_1$.This reduces the number of slack variables from $O(\log(s_1 + 1))$ to just 1.
3) **Binary Expand Slack Variable**
Apply binary expansion to represent integer slack variables as binaries..This increases the number of slack variables to $O(\log(b+1))$ but note that these are binary variables.
4) **Adjust the Penalty Coefficient:**
Adjust the penalty coefficient M to $M' = \rho^2 \cdot M$ to account for the transformation. This maintains a similar penalty scale.
5) **Construct QUBO**
Construct QUBO: $Q_{\text{qubo}} = Q + penalty(A_{\text{mod}} x - b)^2$

Algorithm 5 Handling Inequality Constraints

Input: Quadratic matrix Q, linear matrix A, vector b, penalty coefficient M
Output: Optimized binary decision vector x
Parameters: User-defined parameter $\rho > 0$
procedure TRANSFORMINEQUALITY
    Transform inequality: $Ax/\rho \leq b/\rho$
end procedure
procedure APPROXIMATERHS
    Approximate RHS with slack variable s_1: $Ax/\rho = s_1$
end procedure
procedure SETPENALTYCOEFFICIENT
    Set penalty coefficient: $M' = \rho^2 \cdot M$
end procedure
procedure SOLVETRANSFORMEDQUBO
    Solve transformed QUBO: $\min\ x^T Q x + M' \cdot (Ax/\rho - s_1)^2$
end procedure

3) Applying QUBO Techniques to a Constrained Quadratic Model: The Constrained Quadratic Model (CQM) formulates optimization problems as:

Minimize objective:

$$\sum_i a_i x_i + \sum_{i<j} b_{ij} x_i x_j + c,$$

Subject to constraints:

$$\sum_i a_i^{(m)} x_i + \sum_{i<j} b_{ij}^{(m)} x_i x_j + c^{(m)} \circ 0, \quad m = 1, \ldots, M \tag{25}$$

Table I: Order analysis of Scoring fucntion

Term	Degree	Remarks
$\left(N \sum_u \sum_v x_{u,v}\right)$	1D	Only variable is x
$\left(M \sum_{w \in W} \sum_{d \in D} x_{w,d} \sum_{(d,v) \in E} \sum_{(w,u) \in E} x_{u,v} w_u\right)$	2D	w is constant, so x^2 is variable
$\left(\sum_{u,v} ES_{u,v} \cdot x_{u,v}\right)$	1D	ES is constant, so only variable is x
$\left(\sum_{u,c} ES_{u,c} \cdot x_{u,c}\right)$	1D	ES is constant, so only variable is x

Here, x_i are binary decision variables, a_i, b_{ij}, c are coefficient values, $\circ \in \geq, \leq, =$, and M is the number of constraints.

The quadratic nature of the objective function presents an opportunity to leverage quadratic unconstrained binary optimization (QUBO) techniques. QUBO is designed to handle problems with a quadratic objective over binary variables. By transforming the constrained CQM into an unconstrained QUBO formulation, we enable the application of specialized QUBO solvers. These solvers can exploit the structure of a QUBO problem to effectively search the solution space.

The transformation process requires encoding the constraints into penalty terms added to the objective function. The magnitude of the penalty coefficients must be set sufficiently high to discourage constraint violations. Once encoded as a QUBO, the problem can be submitted to both classical QUBO solvers as well as quantum annealing technology. The latter leverages quantum effects to transverse the search space, converging towards optimal or near-optimal solutions.

By judiciously applying QUBO techniques, certain CQM problem classes become amenable to novel quantum and classical algorithms tailored to QUBOs. This enables the scalable resolution of problems exhibiting quadratic complexity.

VI. Using **Quantum Computing** to Solve QUBO

A. Quadratic Unconstrained Binary Optimization (QUBO)

- **Definition:**
  - QUBO: Quadratic Unconstrained Binary Optimization.
  - Mathematical framework for optimization with binary variables (0 or 1).
  - Objective: Minimize or maximize a quadratic objective function.
- **Mathematical Formulation:**
  - General QUBO equation:

$$\text{minimize } f(x) = \sum_{i=1}^{N} q_i x_i + \sum_{i=1}^{N} \sum_{j=i+1}^{N} q_{ij} x_i x_j \tag{26}$$

  - where x_i is a binary variable, q_i is the linear coefficient, and q_{ij} is the quadratic coefficient.

B. Using Quantum Computing to Solve QUBO

- **Quantum Annealing:**
  - Approach in quantum computing for optimization problems, including QUBO.
  - Quantum system evolves from higher-energy state to find optimal solution.
  - Process governed by a quantum Hamiltonian, gradually changing over time.
- **Mapping to Ising Model:**
  - QUBO problems mapped to Ising model.
  - Transformation involves mapping binary variables to spins and QUBO to Ising Hamiltonian.

C. D-Wave Systems and D-Wave Leap

- **D-Wave Systems:**
  - Company specializing in quantum computing.
  - Quantum processors use quantum annealing for optimization problems.
- **D-Wave Leap:**
  - D-Wave's cloud-based service for remote access to quantum processors.
  - Users submit problems in QUBO or Ising models for solving using quantum annealing.
- **Leap Hybrid Solver:**
  - D-Wave Leap's Hybrid Solver is a cloud-based service that seamlessly blends classical and quantum processing.

– It harnesses the power of D-Wave's quantum processing unit (QPU) in conjunction with classical computing, providing an effective and accessible solution for addressing optimization problems in diverse real-world applications.

D. Annealing and Adiabatic Quantum Computing

- **Quantum Annealing Process:**
  - Quantum annealing starts in a higher-energy state and gradually evolves to a state with the lowest energy.
  - Quantum fluctuations allow exploration of different possibilities simultaneously.
- **Adiabatic Quantum Computing:**
  - Broader approach encompassing quantum annealing.
  - Relies on the adiabatic theorem; the system remains in the ground state during slow evolution between initial and final Hamiltonians.
- **Mathematical Expression (Adiabatic Theorem):**
  - Adiabatic theorem:

$$E_0(t) \leq E_1(t) \leq E_2(t) \leq \dots \tag{27}$$

  - If a quantum system starts in its ground state, a slow enough evolution ensures it remains in the ground state if the evolution is slow compared to the energy gap.

VII. Optimizations

A. Subgraph-Centric Optimization Strategy

In the realm of optimization, a discerning approach is imperative when confronted with a multitude of variables, as their sheer abundance augments computational intricacies. A pragmatic strategy, guided by efficiency considerations, entails a deliberate focus on subgraphs. This meticulous subgraph dissection is rooted in the understanding that the impact of a perturbation—such as delays or cancellations—on one subset, or subgraph, is largely insulated from others.

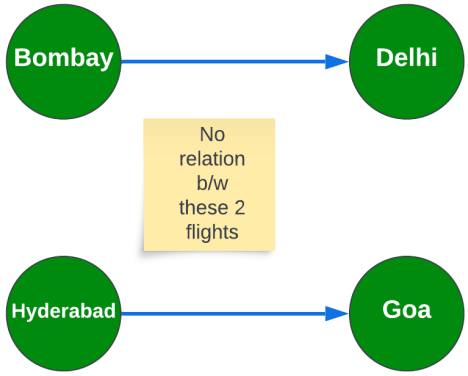

Figure 12: Different Subgraphs

Consider the example of air travel logistics: a delay or cancellation affecting a flight from Bombay to Delhi bears minimal relevance to flights connecting Hyderabad to Goa, and vice versa. In light of this, a judicious decomposition of the overarching optimization problem into discrete subgraphs emerges as an intellectually sound and computationally expedient course of action.

By resolving optimization challenges on a subgraph basis, we can harness the advantage of localized problem-solving. Consequently, the proposed strategy not only streamlines the computational load but also strategically mitigates the complexities inherent in managing an extensive array of variables.

B. Using Multithreading to solve subgraphs simultaneously

In the pursuit of enhanced computational efficiency, a sophisticated optimization framework has been devised, leveraging a concurrent and parallelized approach. The optimization problem at hand, characterized by multiple subgraphs, is addressed through the implementation of multithreading, where each distinct subgraph is assigned to a dedicated computational thread.

The optimization process involves the formulation of Quantum Unconstrained Binary Optimization (QUBO) matrices for individual subgraphs, encapsulating the specific constraints and objectives unique to each subset. Subsequently, simultaneous API requests are dispatched to the D-Wave sampler, orchestrating a concurrent and parallelized exploration of the solution space. This multithreaded execution not only expedites the overall optimization procedure but also capitalizes on the parallel

processing capabilities, allowing for a more expedient resolution of complex problem instances. The seamless integration of multithreading with quantum optimization techniques constitutes a robust and scalable methodology, poised to significantly enhance the computational efficacy of the overall optimization framework. This concurrent approach underscores a commitment to harnessing cutting-edge technologies to address intricate problem domains with unprecedented speed and efficiency.

VIII. Flexibility of Scoring Function

The presented scoring function, $f(N, M, \lambda)$, embodies a flexible and adaptive framework for the optimization of resource allocation in the context of Passenger Name Records (PNRs) and passenger assignments. The incorporation of scaling factors, N and M, allows for the dynamic prioritization of key properties, enabling stakeholders to tailor the optimization process according to their specific requirements. The utilization of N and M as scaling factors provides a nuanced control mechanism, allowing users to emphasize the significance of PNR reallocation (n_{PNR}) or passenger allocation through default solutions ($n_{Passenger}$), based on the particular objectives and constraints of the allocation scenario.

Additionally, the inclusion of edge scores (s_v and s_c) and the reciprocal of the number of flights (λ_c) in clusters (C) further augments the adaptability of the solution, enabling the system to account for nuanced variations in edge importance and the size of flight clusters. This inherent flexibility positions the optimization model as a versatile tool, capable of accommodating diverse scenarios and evolving requirements within the realm of resource allocation and optimization.

IX. Runtime Analysis

A. Defining Variables

1) $n \leftarrow$ # cancelled flights
2) $\hat{m} \leftarrow$ Avg. PNR per flight
3) $m \approx n \cdot \hat{m}$ (# PNRs affected)
4) $\hat{a}_p \leftarrow \#$ proposed alternate flight per PNR
5) $\hat{a}_f \leftarrow \#$ flight-to-flight proposition per cancelled flight
6) $\hat{a}_c \leftarrow \#$ possible proposed connecting flight
7) $\frac{\hat{w}}{\hat{c}} \leftarrow \#$ seats per flight
8) $\hat{c} \approx \#$ avg. proposed class (According to Business Rules)

B. Approximate Values

In adherence to the temporal constraint of 72 hours, the ensuing delineation encapsulates the potential ranges for distinct variables. The values attributed to the parameter $\hat{c}$ have been judiciously selected through a heuristic approach.

SNo	Variables	Max Possible Values
1.	$\hat{m}$	50
2.	$\hat{a}_p$	5
3.	$\hat{a}_f$	5
4.	$\hat{a}_c$	2
5.	$\hat{c}$	4

Table II: Constraints

$$
\begin{aligned}
\text{var} &= m \cdot \hat{a}_p \cdot \hat{c} \\
&\quad + m \cdot \hat{a}_c \cdot \hat{c} \\
&\quad + n \cdot \hat{a}_f \cdot \hat{c} \cdot \log\left(\frac{\hat{m} \cdot \hat{w}}{\hat{c}}\right) \qquad (\because m = n \cdot \hat{m}) \\
\text{var} &= n(\hat{m} \cdot \hat{a}_p \cdot \hat{c} \\
&\quad + \hat{m} \cdot \hat{a}_c \cdot \hat{c} \\
&\quad + \hat{a}_f \cdot \hat{c} \cdot \log\left(\frac{\hat{m} \cdot \hat{w}}{\hat{c}}\right))
\end{aligned}
$$

C. Value Calculation

$$
\begin{aligned}
\text{var} &= n \cdot (50 \times 5 \times 4 + 50 \times 2 \times 4 \\
&\quad + 5 \times 4 \times \log\left(\frac{50 \times 200}{4}\right)\Big) \qquad (28) \\
\text{var} &= n \cdot (1000 + 400 + 20 \times 3.4) \\
\text{var} &= n \cdot (1468)
\end{aligned}
$$

X. Experiments

To conduct our experiments and validate our solution, we examined a percentage of flights that could be impacted. In this context, "affected" refers to flights experiencing either delays (resulting from time changes) or cancellations (involving potential alterations to the schedule, flight deletion, or frequency adjustments). Assuming a general scenario where 1% of flights are affected, a portion of these flights will be randomly selected for cancellation, while the remaining affected flights will experience delays.Assume:

A 1% Flights to be affected

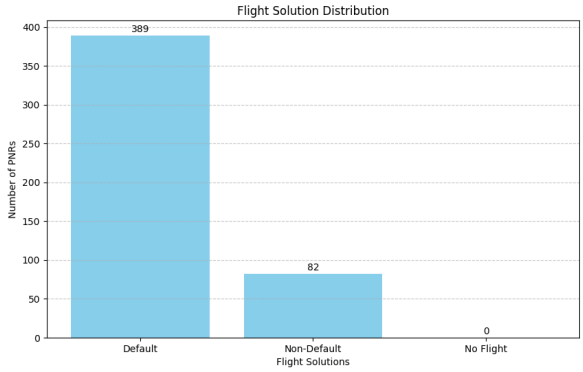

Figure 13: Represents the number of PNRs allotted to default flight, alternate flight, and none

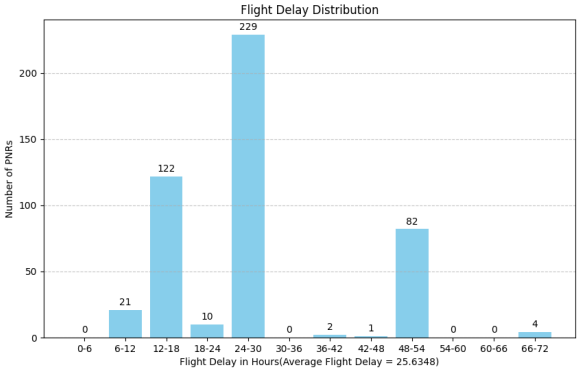

Figure 14: Represents delay distributions

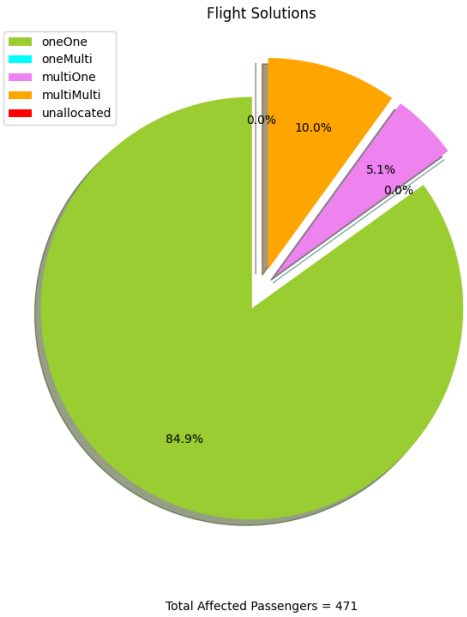

Figure 15: Represents Solutions

Experiment Number		1	2	3	4	5
Cancelled Flights		13	11	13	15	13
Delayed Flights		4	12	7	3	5
PNR Reallocation %	One to One	73.9	84.9	92.1	82.2	87.2
	One to Many	4.9	0	0	0	0
	Many to One	9.9	5.1	4.7	15.9	5.9
	Many to Many	9.1	10	2.6	5.5	6.1
	Unallocated	2.3	0	0.6	0.5	0.7
#Default Flight	Default Flight	348	389	480	497	452
	Alternate Flight	127	82	51	118	99
	No Flight	11	0	3	3	4
Average Delay		30.3441	25.63	32.86	26.46	37.47

Figure 16: Observations

B 2% Flights to be affected

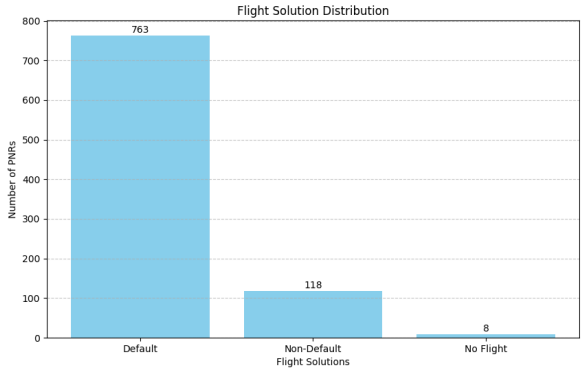

Figure 17: Represents the number of PNRs allotted to default flight, alternate flight, and none

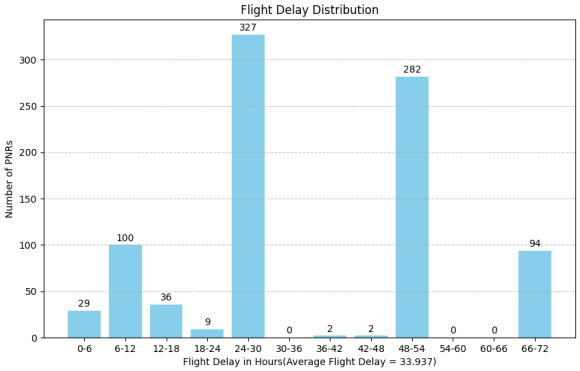

Figure 18: Represents delay distributions

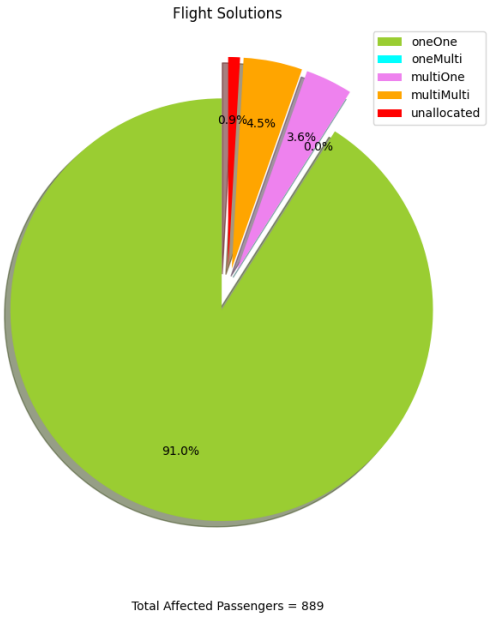

Figure 19: Represents Solutions

Experiment Number		1	2	3	4	5
Cancelled Flights		34	26	20	25	28
Delayed Flights		13	15	17	16	25
PNR Reallocation %	One to One	89.6	87.8	87.9	91	89.6
	One to Many	0	0	0	0	0.4
	Many to One	4.1	3.9	6.6	3.6	4.8
	Many to Many	6	7.5	5.5	4.5	4.8
	Unallocated	0.3	0.8	0	0.9	0.3
#Default Flight	Default Flight	1105	860	605	763	1069
	Alternate Flight	198	197	170	118	166
	No Flight	4	8	0	8	4
Average Delay		27.05	31.88	32.04	33.93	35.8

Figure 20: Observations

C 5% Flights to be affected

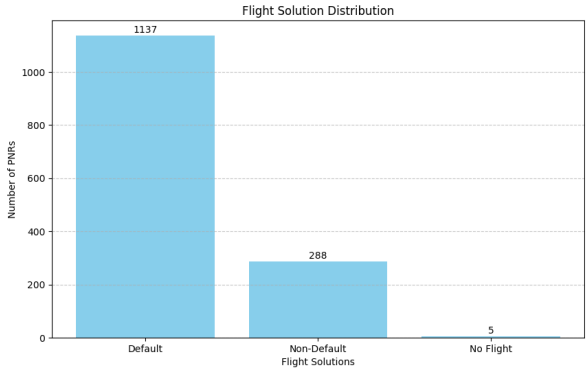

Figure 21: Represents the number of PNRs allotted to default flight, alternate flight, and none

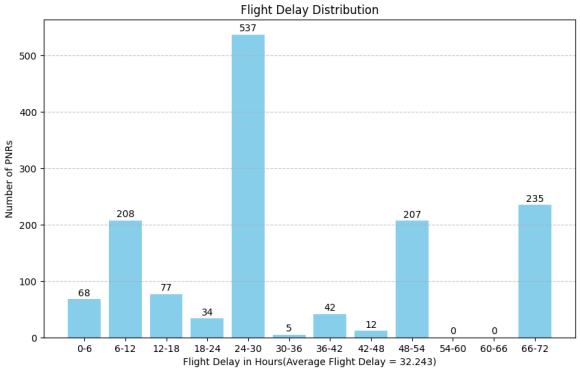

Figure 22: Represents delay distributions

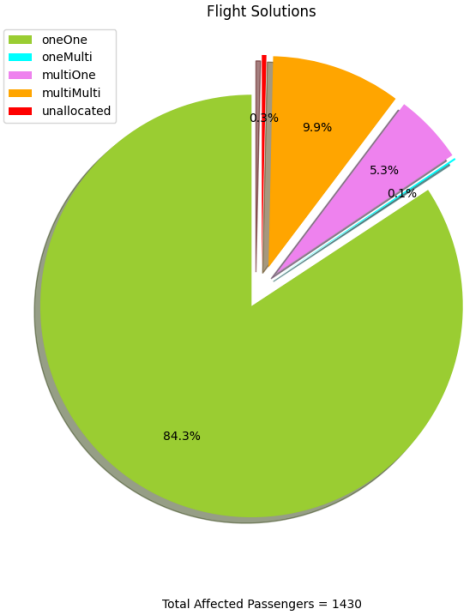

Figure 23: Represents Solutions

Experiment Number		1	2	3	4	5
Cancelled Flights		64	61	56	41	71
Delayed Flights		29	41	40	29	36
PNR Reallocation %	One to One	88.9	87.1	87.7	84.3	86
	One to Many	0	1.5	0.6	0.1	0
	Many to One	4.8	5.7	4.8	5.3	7.1
	Many to Many	5.7	4.7	6.4	9.9	6.6
	Unallocated	0.6	0.9	0.5	0.3	0.2
#Default Flight	Default Flight	1829	2025	1824	1137	2576
	Alternate Flight	373	279	349	288	412
	No Flight	13	21	10	5	6
Average Delay		26.14	22.96	28.35	32.24	28.98

Figure 24: Observations

D 10% Flights to be affected

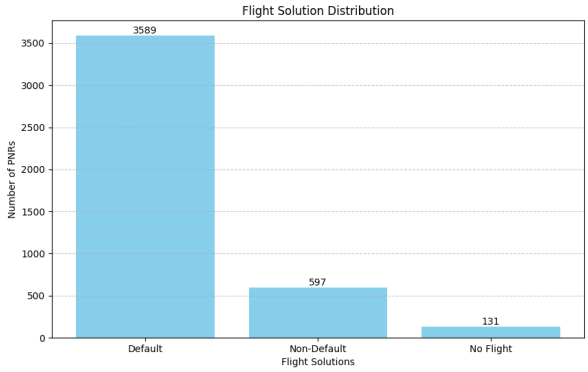

Figure 25: Represents the number of PNRs allotted to default flight, alternate flight, and none

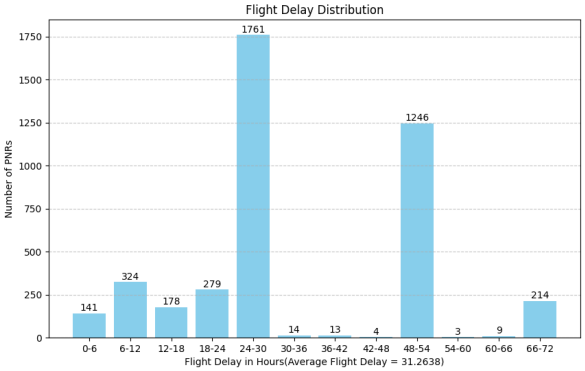

Figure 26: Represents delay distributions

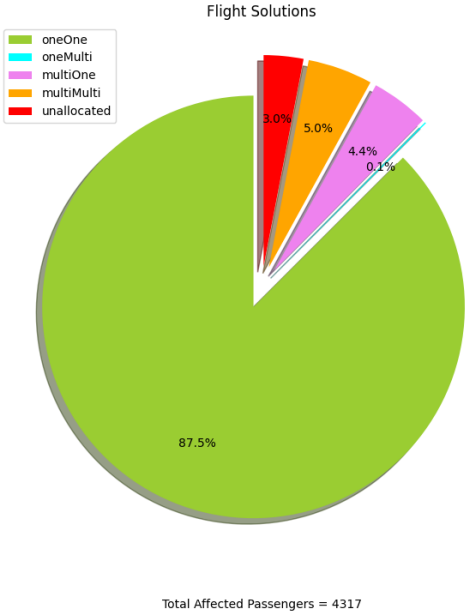

Figure 27: Represents Solutions

Experiment Number		1	2	3	4	5
Cancelled Flights		126	89	106	106	97
Delayed Flights		88	66	81	90	81
PNR Reallocation %	One to One	87.4	88.9	87.8	87.5	84.8
	One to Many	0.2	0.6	0	0.1	0.6
	Many to One	4.1	4.1	4.3	4.4	5.6
	Many to Many	5.9	5.7	5.1	5	7.8
	Unallocated	2.5	0.7	2.8	3	1.1
#Default Flight	Default Flight	4181	2832	3277	3589	3043
	Alternate Flight	730	486	665	597	714
	No Flight	125	24	112	131	43
Average Delay		32.24	31.03	29.04	31.26	31.42

Figure 28: Represents delay distributions

XI. Analysis of Results & Conclusion

Most of our research efforts are centered around the crucial aspect of PNR reallocation, and the findings are highly promising. The approach we utilised continuously attains a remarkable degree of PNR reallocation, guaranteeing that just a small percentage of passengers are left without a feasible alternative solution. The analysis of default flight results underscores the fact that a majority of passengers benefit from reallocation through the default solution. This result enhances the overall consistency in the suggested alternative flight options.

In addition, our algorithm strongly prioritises one-to-one solutions, favouring them over multi-stop alternatives. This enhances the overall effectiveness of the proposed solution. The efficiency of our approach is underscored by the fact that all our experiments finish within a duration of less than 10 minutes of quantum computing time on a DWave quantum machine. This not only marks our approach as highly efficient but also as significantly cost-effective.

Throughout our experiments, we observed a remarkable degree of parallelism in the solution process. The original problem is divided into smaller subgraphs, which greatly enhances the scalability of our method. The significant degree of parallelism in our technique promotes its adaptability and robustness in many scenarios related to PNR reallocation in the airline industry.

References

[1] I. Gioda, D. Caputo, E. Fadda, D. Manerba, B. Silva Fernández, and R. Tadei, "Solving assignment problems via quantum computing: a case-study in train seating arrangement," in *2021 16th Conference on Computer Science and Intelligence Systems (FedCSIS)*, 2021, pp. 217–220.

[2] H. Mohammadbagherpoor, P. Dreher, M. Ibrahim, Y.-H. Oh, J. Hall, R. E. Stone, and M. Stojkovic, "Exploring airline gate-scheduling optimization using quantum computers," *arXiv:2111.09472 [quant-ph]*, 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2111.09472

[3] K. Temme, T. J. Osborne, K. G. Vollbrecht, D. Poulin, and F. Verstraete, "Quantum metropolis sampling," *Nature*, vol. 471, no. 7336, pp. 87–90, Mar 2011. [Online]. Available: https://doi.org/10.1038/nature09770

[4] A. Verma and M. Lewis, "Variable reduction for quadratic unconstrained binary optimization," *arXiv preprint arXiv:2105.07032*, 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2105.07032