Applications of Operations Research to Capacity Planning and Revenue Management

Moses Addai, Hardik Gupta, Rob Mailley

March 9, 2024

Source code at: `https://github.com/guptahardik/tailassignment`

1 Capacity Planning

The objective of the "Aircraft Route Scheduling" model is to assign aircraft to routes in a way that maximizes total profit while adhering to operational constraints.

1.1 Notation

i, j - Indices representing origin and destination airports, respectively.

k - Index representing an aircraft.

Profit_{ijk} - Profit from assigning aircraft k to route from i to j.

Duration_{ij} - Flight duration from i to j.

Seats_k - Number of seats in aircraft k.

Fare_{ij} - Fare for the route from i to j.

Distance_{ij} - Distance of the route from i to j.

x_{ijk} - Binary decision variable; 1 if aircraft k is assigned to route i to j, 0 otherwise.

1.2 Objective Function

$$\text{Maximize} \sum_{i,j,k} (\text{Fare}_{ij} \times \text{Seats}_k - \text{OpCost} \times \text{Distance}_{ij}) \times x_{ijk} \tag{1}$$

1.3 Constraints

1. **Total Flying Time**: For each aircraft, the total flying time must not exceed 24 hours.

$$\sum_{i,j} \left(\frac{\text{Distance}_{ij}}{\text{Speed}} \right) x_{ijk} \leq 24, \quad \forall k \tag{2}$$

2. **Unique Aircraft per Route**: Each route can be assigned to at most one aircraft.

$$\sum_{k} x_{ijk} \leq 1, \quad \forall i, j \tag{3}$$

3. **Minimum Operation**: Each aircraft must operate a minimum of X hours.

$$\sum_{i,j} \left(\frac{\text{Distance}_{ij}}{\text{Speed}} \right) x_{ijk} \geq \text{MinDuration}, \quad \forall k \tag{4}$$

4. **Continuity**: For non-base airports, the number of departures equals the number of arrivals.

$$\sum_{i} x_{ijk} = \sum_{j} x_{ijk}, \quad \forall k, \text{ for non-base airports} \tag{5}$$

1.4 Assumptions

The model incorporates several key assumptions:

- Total operational time for each aircraft does not exceed 24 hours within a scheduling period.
- Each route is assigned to at most one aircraft.
- Operational continuity is maintained by ensuring the number of arrivals equals the number of departures at non-base airports.
- The primary goal is to maximize total profit based on fixed operating costs and fares.
- Operating costs and fares are assumed to be constant.
- The model simplifies route and aircraft selection without considering code-sharing, aircraft range, or regulatory constraints.

1.5 Conclusion

This mathematical model provides a framework for optimizing aircraft route assignments to maximize profit, given a set of operational constraints. The model's assumptions aim to simplify complex airline scheduling challenges into a tractable linear programming problem.

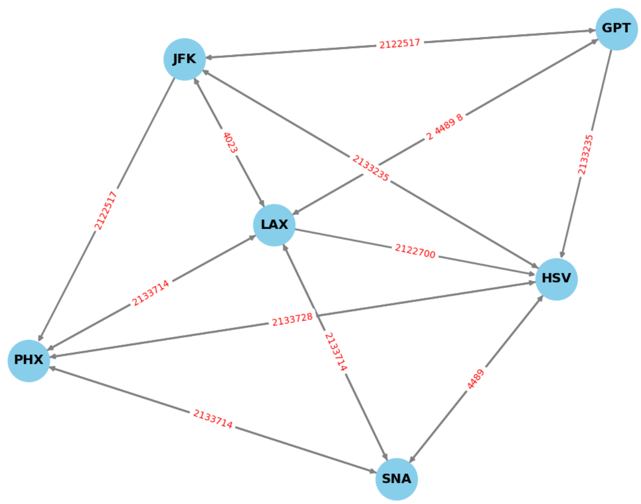

Figure 1: Example result graph on a small dataset, JFK, PHX, HSV are base airports

2 Route Finding

For a successful airline, building a route is only half the battle. After deregulation in 1979, the US airline sector became a highly competitive business, and different airlines began to try to differentiate their services in different ways. But differentiation between classes of service and between other airlines only will get an airline so far. At the end of the day, they are selling transportation from point A to point B, and customers want to get there as quickly and as cheaply as possible, with emphasis on cheaply. So if a customer wants to purchase an itinerary between two destinations, they search their origin, destination, and many other parameters, and expect to see a multitude of options to get to their destination.

In reality, these queries often have hundreds of parameters. SABRE, the pioneering revenue management system, has a schedule search API endpoint which includes as parameters date, time, class of service, and about a hundred other options.

Our implementation is much simpler, and extends Djikstra's algorithm. We use Yan's algorithm to find the top k shortest paths through our network.

We give our interpretation of the pseudocode below. Our code is given in the Github Appendix.

We can alter the distance labels on the graph to account for customer preferences.

For example, if we have a customer that is booking on business class, they are likely less price sensitive.

$$\text{label} = a * \text{duration} + b * \text{price}$$

Algorithm 1 Yen's Algorithm. Assume DJIKSTRA returns a list of nodes.

```
procedure YEN(G, s, t, k)
    A = [ ]
    B = [ ]

    A.append(DJIKSTRA'S(G, s, t))

    for 1 ≤ kth ≤ k do
        for 1 ≤ i ≤ (length of last path in A) −1 do
            spur_node = A[−1][i]
            root_path = A[−1][: i + 1]

            G.remove_edge(root_path[−1], A[−1][i + 1])

            spur_path_length = length of DJIKSTRA(G, spur_node, t)
            total_path = DJIKSTRA(G, s, spur_node) \ spur_node+DJIKSTRA(G, spur_node, t)
            total_path_len =length of DJIKSTRA(G, s, t)

            B.heap_push((total_path, total_path_len))                ▷ on length
            G.add_edge(root_path[−1], A[−1][i + 1])
        if B is Null then
            Break
        A.heap_push(B.heap_pop[path])

    return A
```

So, we would alter these parameters based on input from the revenue management team to show each customer a unique output.

An example is as follows: say we have a frequent flyer who is not at all price sensitive. Then, we can show them data based on only duration that is $a = 1, b = 0$ for each distance label. Between XNA and BDL, we have:

```
(base) robmailley@Robs-MacBook-Pro ~ % /opt/homebrew/bin/python3
/Users/robmailley/Documents/103Final/yen.py
K-shortest paths from XNA to BDL:
Path 1: ['XNA', 'BDL']
Path 2: ['XNA', 'LAS', 'PVU', 'BDL']
```

A customer who has never booked with us before, booking on economy class is likely heavily price sensitive. therefore, we would have $a = 0, b = 1$. Running the model with k = 2, we have

```
(base) robmailley@Robs-MacBook-Pro ~ % /opt/homebrew/bin/python3
/Users/robmailley/Documents/103Final/yen.py
K-shortest paths from PVU to LAS:
Path 1: ['PVU', 'LAS']
Path 2: ['PVU', 'SBD', 'LAS']
```

Finally, weighting the two at $a = .5$ and $b = .5$, searching for the top 2 routes between PVD and PVU, we have

```
(base) robmailley@Robs-MacBook-Pro ~ % /opt/homebrew/bin/python3
/Users/robmailley/Documents/103Final/yen.py
K-shortest paths from PVD to PVU:
Path 1: ['PVD', 'CHS', 'PVU']
Path 2: ['PVD', 'CHS', 'BDL', 'PVU']
```

Not very helpful, unless one was trying to make a mileage run. With more input parameters and weights, this would be an even more useful search, and testing these new weights over different passengers would yield interesting statistical results.

3 Game Theory

In the previous models described above we assume the routes in the network are operated by a single airline with no external rivalry. However, this is not representative of the real airline industry where there is fierce competition for consumers. Consider our network with two airlines flying multiple routes across the nodes. Depending on the density and vastness of the network each airline can have an infinite number of pure strategies with regards to connecting passengers from one node to another. In such a case, we can use the Cournot Competition model to calculate the Pure Strategy Nash Equilibrium(PSNE) of flights each airline would offer for a given route.

Case Study

Two airlines Breeze Airways and Southern Air fly offer daily flights between Los Angeles International Airport (LAX) and Jacksonville Airport(JAX) which are two nodes in our network. Given linear price function of P = -1.89Q + 148.89 and linear marginal cost of constant C =100, we can determine the PSNE quantity of flights for each airline using Cournot's Model.

The price p is a decreasing function of the total quantity produced such that q = q1 + q2

Production costs $ci(qi) = 100qi$ where $qi \in [0, \infty)$
p(q) = max(0, -1.89q + 148.89)

Hence the profit function $u_i(q1, q2)$ is given by
$q_i * max(0, -1.89q + 148.89) - 100q_1$
$u1 = -1.89q_2^2 - 1.89q1q2 + 148.89q1 - 100q1$

Taking the first order differentials gives us $\frac{\sigma u1}{\sigma q1} = -3.78q_1 - 1.89q_2 + 148.89 - 100 = 0$
Therefore $q1 = \frac{48.89 - 1.89q_2}{3.78}$ and $q2 = \frac{48.89 - 1.89q_1}{3.78}$
We then solve the two equations simultaneously to obtain PSNE at (q1,q2) = (8.62, 8.62) which represent number of flights for Breeze and Southern Air respectively per day.

This is a straightforward case whereby we are able to solve for the PSNE given linear functions. What if the price functions is non-linear? This is possible in a Cournot oligopoly model whereby more than two airlines are involved. Such a context involves implicit function problems for which the functional equation is described as below:

$$p + \frac{S(p)}{D'(p)} - MC(S(p)) = 0$$

where S(p)=supply or quantity, MC = marginal cost, D(p)= market demand curve

In simple cases, the function can be solved explicitly but in complicated cases, there may be no explicit solutions.