A Game-Theoretic Approach to the Daily Aircraft Routing Problem

Akshi Jain, 21025 Harshal Badge, 21069
Sushrut Jog, 21131 Ishan Thoke, 21328

DSE/ECS 311 : Project Review

Motivation

- Flight delays bring misery to the airlines, the passengers, and the airports.
- The average cost of aircraft block time for US passenger airlines was estimated to be $100.80 per minute (Airlines for America).
- Given the complexity of aviation networks, optimal flight scheduling, which accounts for such delay costs reasonably, is crucial.

What we have done – We formally review...

- We formally review Deng et al. (2022), which solves a daily aircraft route problem (DARP), incorporating route planning and route assignment in a novel game theory framework.
- The paper combines a column generation algorithm and variable neighborhood search, a meta-heuristic algorithm.
- A model of integrated aircraft routing and scheduling that considers fleet assignment.
- Uncertain factors such as weather, air traffic control that cause flight delays are incorporated.

Flight Plans

1. Advanced strategy
   - It refers to the prepared flight plan before the scheduled flight takes place.
   - Improves the robustness of flight plans by adjusting the transit time appropriately for the actual take-off.
2. After the fact strategy
   - Used to undertake certain measures in time after flight delays, such as aircraft transfer, aircraft exchange, etc., to avoid causing larger delays

Literature

- Levin (1971) proposed DARP as a 0-1 programming model using branch and bound methods from Land-and-Doig techniques.
- These methods later evolved to column generation and Dantzig-Wolfe decomposition used by Desaulniers et al. (1997).
- Xu et al. (2021) implemented column generation as well as VNS for a solution to a mixed integer programming model of the airline integrated robust scheduling problem, including flight retiming decisions and delay considerations.
- Game-theoretic methods have tackled robustness in power systems and rail networks by framing worst-case disruptions as adversarial players in Lima et al. (2008); Laporte et al. (2010).

Definitions

- $F(g)$ = the utility function of the DARP problem for a schedule $g \in G$
- $F(g, v)$ = the utility function where the flight $v \in V$ is delayed
- t = the minimum connection time between two legs
- *Leg*: a leg is a flight segment.
- *Optimal robust schedule*: if there exists a schedule $g^* \in G$ such that

  $$g^* = \min_{g \in G} \max_{v \in V} F(g, v),$$

  then g^* is called an *optimal robust schedule*.

Network Graph Construction

We represent the problem as a directed graph $G(V, E)$, where V is the set of all legs represented as vertices, and E is the set of all permissible directed edges. Each leg v has four attributes:

- DA_v = departure airport,
- AA_v = arrival airport,
- DT_v = departure time,
- AT_v = arrival time.

The edge set E is constructed, given the set V and a constant t as: there exists a directed edge $[v_1 \rightarrow v_2]$ *iff* the following two conditions are satisfied:

1. $AA_{v_1} = DA_{v_2}$, that is, the arrival airport of v_1 is the same as the departure airport of v_2, and
2. $DT_{v_2} - AT_{v_1} > t$, that is, the departure time of v_2 is no more than t away from the arrival time of v_1.

Example of a *valid* network graph

Consider the leg set $V := \{a, b, c, d, e, f\}$ and minimum connection time $t := 8$ (in hours, kept unitless for now). Let the attributes for the legs be given as 4-tupels (DA_i, AA_i, DT_i, AT_i) for $i \in \{a, b, c, d, e, f\}$ as:

$$a = (\text{BHO}, \text{BLR}, 0830, 1000) \qquad d = (\text{BOM}, \text{DEL}, 0400, 0530)$$

$$b = (\text{BLR}, \text{DEL}, 2330, 0150) \qquad e = (\text{BLR}, \text{MAA}, 1510, 1600)$$

$$c = (\text{BOM}, \text{DEL}, 1200, 1330) \qquad f = (\text{MAA}, \text{BHO}, 2000, 2115)$$

Parameters

A = a set of all matching routes in the plane and flight connection network,

V = the set of all legs that need to be scheduled,

K = the set of all aircrafts,

c_a = the cost of the selected matching $a \in A$,

x_a = a binary variable which is 1 if the matching route $a \in A$ is selected, and 0 otherwise,

$\alpha_{a,v}$ = a binary variable which is 1 if the selected matching $a \in A$ contains the leg v, and 0 otherwise,

$\beta_{a,k}$ = a binary variable which is 1 if the selected matching $a \in A$ contains aircraft $k \in K$, and 0 otherwise.

Formulating the Problem

With these variables, we model the problem as:

$$\min \sum_{a \in A} c_a x_a \tag{1}$$

Subject to:

$$\sum_{a \in A} \alpha_{a,v} x_a = 1, \quad \forall v \in V, \tag{2}$$

$$\sum_{a \in A} \beta_{a,k} x_a \leq 1, \quad \forall k \in K, \tag{3}$$

$$x_a \in \{0, 1\}, \quad \forall a \in A. \tag{4}$$

The objective function is to minimize the total cost when all flights are not delayed.

Min-max Two-player Zero-sum Game

- A **min-max**, or minimax game is a game where each player adopts a strategy to minimise their maximum possible loss. An alternate way to model this is via a **max-min** or a maximin game, where players maximise their minimum possible gain.
- A **zero sum game** is when there is no net change in advantage. This means that player one's gain results in player two's loss. For example, chess.
- A state where no player can improve their payoff by changing just their own strategy is called as the **Nash equilibrium**.

DARP as a Min-Max Two-player Zero-sum Game

- To deal with the uncertainty of flight delays, a min-max game framework (a robust optimization model based on game theory) is introduced.
- It treats delays as a “game” between two players

The min-max model under uncertainty is specified as:

$$\min_{g \in G} \max_{v \in V} F(g, v), \tag{5}$$

subject to $G(x, g, v) \leq 0$

The goal is to find a schedule where the airline’s plan works well even in the worst-case scenario of delays.

The Three Elements of the Game

1. Player set:
   - Player 1 (Airline): Wants to assign flights to planes as cheaply as possible.
   - Player 2 (Uncertainty): Acts like an "attacker" trying to cause delays (e.g., bad weather) to maximize costs.
2. Strategy Set:
   - Player 1: All feasible scheduling networks, i.e., $g \in G$,
   - Player 2: All fight segments, i.e., $v \in V$
3. Payoff function: Here, the gain of one player causes other player's loss. Player 1 finds the schedule with the lowest cost so it minimizes the cost. Player 2 picks which flight to delay to cause the most financial harm, so it maximizes cost.

Calculating Cascading Delays

- When player 2 chooses to attack a certain flight v, the subsequent flight may also be delayed due to the spread.
- This delay probability can be obtained through experience

Using Cui et al. approach, the probability of flight delay is determined by the transit time between two flights (time between the arrival of flight F_i and departure of F_{i+1} on the same aircraft.)

For an aircraft route $[F_1, F_2, \ldots, F_n]$, if F_1 is delayed, the probability of subsequent delays is:

$$p_{F_k} = \prod_{r=1}^{k-1} q_r \quad \text{for } k = 2, 3, \ldots, n$$

where q_r is the probability of the rth delay time after fight i.

Example: Calculating the total expected cost

Calculating the total expected cost

- **Route**: $[F_1, F_2, F_3, F_4]$.
- **Initial Delay**: F_1 is delayed (e.g., due to weather).
- **Delay Costs**: $c_{F_1} = 1000$, $c_{F_2} = 800$, $c_{F_3} = 600$, $c_{F_4} = 400$
- **Delay Probabilities**: $p_{F_2} = 0.8$, $p_{F_3} = 0.5$, $p_{F_4} = 0.3$

$$
\begin{aligned}
c &= c_{F_1} + c_{F_2} p_{F_2} + c_{F_3} p_{F_2} p_{F_3} + c_{F_4} p_{F_2} p_{F_3} p_{F_4} \\
&= 1000 + (800 \times 0.8) + (600 \times 0.8 \times 0.5) + (400 \times 0.8 \times 0.5 \times 0.3) \\
&= 1000 + 640 + 240 + 48 \\
&= 1928
\end{aligned}
$$

The airline expects an **additional cost of 1928 units** if F_1 is delayed, accounting for cascading delays in F_2, F_3, and F_4.

LP Model Under Uncertainity

- c_a: Cost of assignment $a \in \mathcal{A}$.
- c_v: Additional cost if leg $v \in \mathcal{V}$ is delayed.
- p_v: Probability of delay propagation for leg v (depends on connection time).

$$\min_x \max_v \left[\sum_{a \in A} c_a\, x_a + \sum_{v \in V} p_v\, c_v\right], \tag{6}$$

$$\sum_{a \in \mathcal{A}} \alpha_{av} x_a = 1, \forall v \in \mathcal{V} \tag{7}$$

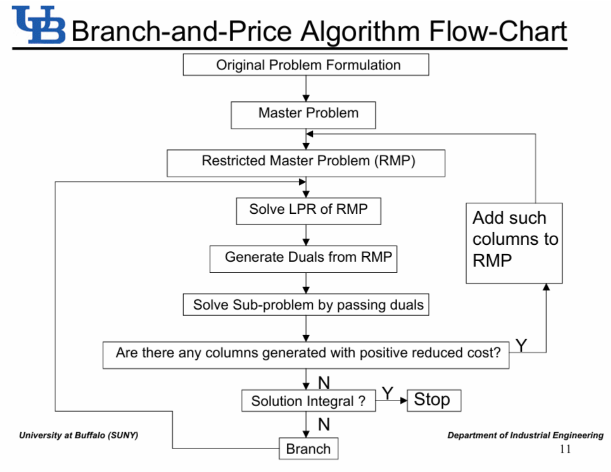
$$\sum_{a \in \mathcal{A}} \beta_{ak} x_a \leq 1, \forall k \in \mathcal{K} \tag{8}$$

$$x_a \in \{0, 1\}, \forall a \in \mathcal{A} \tag{9}$$

Branch Pricing Algorithm

- Branch pricing algorithm $\equiv$ column generation algorithm + branch and bound methods
- Solves integer linear programming and MILP problems with (too) many variables.
- Idea: To reduce the number of feasible schedules considered $\Longrightarrow$ scheduling cost is limited to a certain proportion of the optimal scheduling.

Branch Pricing Algorithm



- Instead of branching, VNS is employed for efficiency..

Variable Neighborhood Search (VNS)

A metaheuristic opimization algorithm that systematically explores neigh- borhood structures to escape local minima. We use the following facts:

Fact 1. A local minimum with respect to one neighborhood structure is not necessarily a local minimum for another structure.
Fact 2. A global minimum is a local minimum with respect to all possible neighborhood structures.
Fact 3. (Empirical) For many problems local minima with respect to one or several neighborhoods are relatively close to each other.

Model 1: Deterministic Aircraft Routing

Model 1: Pseudocode

We formulate the problem as a mixed-integer linear program (MILP). The objective is to select a subset of feasible aircraft routes that covers all flight legs exactly once while minimizing the total operating cost.

Algorithm 1: MILP for Deterministic Aircraft Routing (Model 1)

Input: Flight legs V, route set A, route costs c_a, aircraft limit $|K|$

Output: Selected routes $S \subseteq A$, minimizing total cost

```
Initialize MILP model
Define binary variables x_a for all a ∈ A
Set objective: minimize ∑_{a∈A} c_a · x_a
foreach flight leg v ∈ V do
    Add constraint: ∑_{a∈A:v∈a} x_a = 1
end
Add constraint: ∑_{a∈A} x_a ≤ |K|
Solve the MILP using a solver (e.g., CBC, Gurobi)
return S = {a ∈ A | x_a = 1} and total cost
```

Model 2: Game-Theoretic Robust Aircraft Routing

Model 2 extends the deterministic MILP by incorporating a worst-case delay cost using a game-theoretic min–max approach.

Algorithm 2: VNS Heuristic for Robust Aircraft Routing

Input: Initial solution x^0 from Model 1, max iterations, $k_{\max}$
Output: Best robust solution found

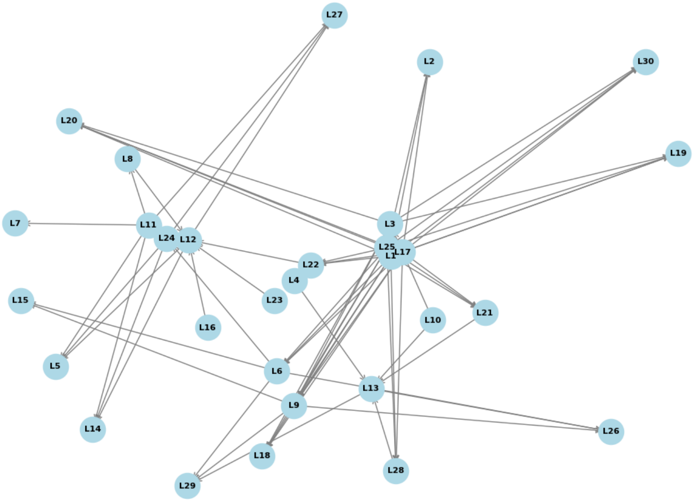
Initialize:
  best_solution ← x^0
  best_cost ← WorstCaseCost(x^0)
for *t = 1* **to** *max_iter* **do**   `// Outer iterations`
  $k \leftarrow 1$
  while $k \leq k_{\max}$ **do**`// Explore neighborhood`
    Shake: Drop k random routes from current_solution
    sol' ← DropKAndRecover(x, k)
    Optionally: sol" ← LocalSearch(sol')
    cost" ← WorstCaseCost(sol")
    if *cost" ¡ best_cost* **then**
      best_solution ← sol"
      best_cost ← cost"
      $k \leftarrow 1$
    end
    else
      $k \leftarrow k + 1$
    end
  end
end
return best_solution, best_cost

Dataset and Setup

- **Synthetic Dataset:**
  - 30 flight legs, randomly generated over a 1-day schedule
  - 5 synthetic airports (AP1–AP5)
  - Flight durations between 60–180 minutes
  - 30% of flights include extra buffer (60–120 mins)
- **Flight-Connection Graph:**
  - Nodes: Flight legs with departure/arrival times
  - Directed edges for feasible same-aircraft transitions (45 min turnaround)
  - Routes generated as simple paths (max 4–5 legs)
- **Cost Model:**
  - Fixed cost per route: $1000
  - Variable cost: $5 per minute of flight time
  - Delay cost (Model 2): $30 per minute ($\times$6 penalty)
- **Tools:** Python (pandas, NetworkX, PuLP), CBC solver

Visualization of the Graph G

Flight Connection Graph (Nodes = Flight Legs, Edges = Feasible Connections)



Model 1: Deterministic DARP Implementation

- Solved using **PuLP** with the **CBC solver**
- Full MILP includes route selection, coverage, and fleet size constraints
- Feasible routes pre-generated from connection graph

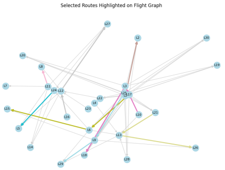
Model 1 Output:

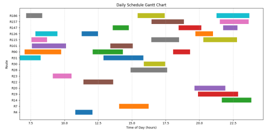
Status: Optimal | Routes selected: 17/30 | Total cost: $34,305

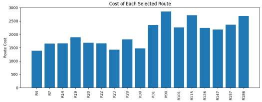
Route_ID	Legs	Duration (min)	Cost ($)
$R4$	$[L4]$	77	1,385
$R7$	$[L7]$	131	1,655
$R14$	$[L14]$	132	1,660
...	...	...	...
$R186$	$[L25, L9, L29]$	338	2,690

Model 1: Visualizations

- **Flight Connection Graph:** Nodes are flight legs; edges represent feasible connections with 45 min turnaround.
- **Gantt Chart:** Aircraft-wise leg schedules showing time sequences.
- **Cost per Route:** Bar chart of operating cost per selected route.







Model 2: Robust DARP Output

- Solved using **PuLP + CBC solver** with an auxiliary variable z capturing worst-case delay cost
- Objective includes both operating and delay propagation costs

Model 2 Output:

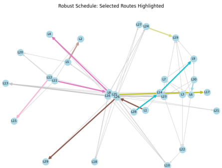
- **Selected Routes:** 17 / 30 aircraft
- **Operating Cost:** \$30,515
- **Worst-Case Delay:** \$6,261
- **Total Robust Objective:** \$43,037
- **Robustness Premium:** 41.03%

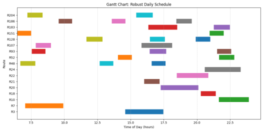
Selected Robust Routes:

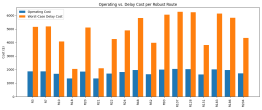
- R3: legs = [L3], cost = \$1,860, delay = \$5,160
- R7: legs = [L7], cost = \$1,865, delay = \$5,190
- R10: legs = [L10], cost = \$1,680, delay = \$4,080
- ...
- R204: legs = [L30, L6], cost = \$1,720, delay = \$4,331

Model 2: Visualizations

- **Robust Flight Graph:** Highlights routes selected in the robust solution.
- **Robust Gantt Chart:** Displays time distribution across aircraft with robustness buffers.
- **Cost vs Delay per Route:** Bar chart comparing operating and delay cost per selected route.







Model Comparison & Trade-off Summary

- Model 2 reduces worst-case delay by **85%**
- Operating cost increases by **23%**
- Overall robustness premium of **+41%**

Metric	Model 1	Model 2	Change	Pct.
Operating Cost	\$24 890	\$30 515	+\$5 625	+22.6%
Worst-Case Delay Cost	\$41 126	\$ 6 261	\$34 865	84.8%
Total Objective	—	\$43 037	—	—
Robustness Premium	—	+41.0%	—	—

Algorithm Runtime & Performance

Method	Solve Time (s)	Objective
Model 1: Deterministic MILP	0.45	\$24 890
Model 2: Robust MILP (CBC)	0.62	\$43 037
VNS Heuristic (Model 2 initial)	3.2 (200 iters)	\$41 126
VNS Heuristic (Model 2, deep search)	4.8 (200 iters, $k_{\max} = 5$)	\$41 126

- MILPs solve quickly (under 1 second)
- Heuristic (VNS) finds good solutions, but didn't outperform MILP on this instance
- Model 2 offers robust schedules with no significant runtime penalty

Conclusion

- Developed two MILP models for daily aircraft routing:
  - **Model 1 (Deterministic):** minimizes total operating cost
  - **Model 2 (Robust):** guards against worst-case delay propagation
- Robust MILP (Model 2) achieved:
  - **85% reduction** in worst-case delay cost
  - **23% increase** in operating cost
  - **41% robustness premium**
- All models solved rapidly (under 1s); heuristics validated but not superior
- Clear trade-off: *small increase in cost leads to major robustness gains*