Chapter 2: Passenger Recovery Algorithm for Airline Disruptions

In this chapter, we present our work on passenger scheduling algorithms for the airline recovery problem. This work is in collaboration with Clifford Stein, and based on my internship experience at GE Global Research in 2019.

The organization of this chapter is as follows- we provide the background behind airline disruptions and recovery in Section 2.1 and summarize the prior work in the field in Section 2.2. We outline our contributions in 2.3. In Section 2.4, we introduce the basic concepts of the problem space. Section 2.5 details our proposed algorithms for airline recovery, with experiments based on a real-world data set in Section 2.6. Finally, we conclude and discuss ideas for future work in Section 2.7.

2.1 Background on airline disruptions

Airline disruptions occur due to a resource needed to operate a flight unavailable before departure, and may affect the departure of one or more flights [49]. Disruptions occur at short notice and causes flights delays and cancellations, ferried flights, crew and passenger misconnects, and have a domino effect on the whole network. The top challenges for airlines in 2018 are network disruptions, unplanned maintenance and fuel overspend [9]. According to industry statistics, in 2018, 5.6 million departing flights were delayed for an average of 57 minutes, $436,000$ flights were canceled, and the travel plans of over 655 million passengers were disrupted [9]. Irrespective of any gradual improvements, such statistics reflect poor performance. Moreover, disruptions are now commonplace [9], affecting airline branding, airline costs, and passenger loyalty.

Delays and cancellations cost $\$33.4$ billion in 2018. Figure 2.1 shows the breakdown of disruption costs to airlines. The major costs are passenger welfare (reimbursing cancelled itineraries,

providing hotels or refreshment, passenger dissatisfaction due to delays etc.), followed by the costs of changing aircraft and crew schedules which are often tightly regulated. Figure 2.2 highlights the reason for airline delays and disruptions. While there are many reasons shown for network disruptions, for example, infrastructure constraints such as airport and airspace (congestion) are important in Figure 2.2. The reality is that two-thirds of all delays and cancellations are within an airline's control. Indeed, almost half of all delays (43%) are reactionary in nature and caused by the primary delay (e.g., bad weather or airport congestion). Thus, the significance of reacting well to the uncertainty of a primary disruption by re-planning schedules is illustrated in Figure 2.2.

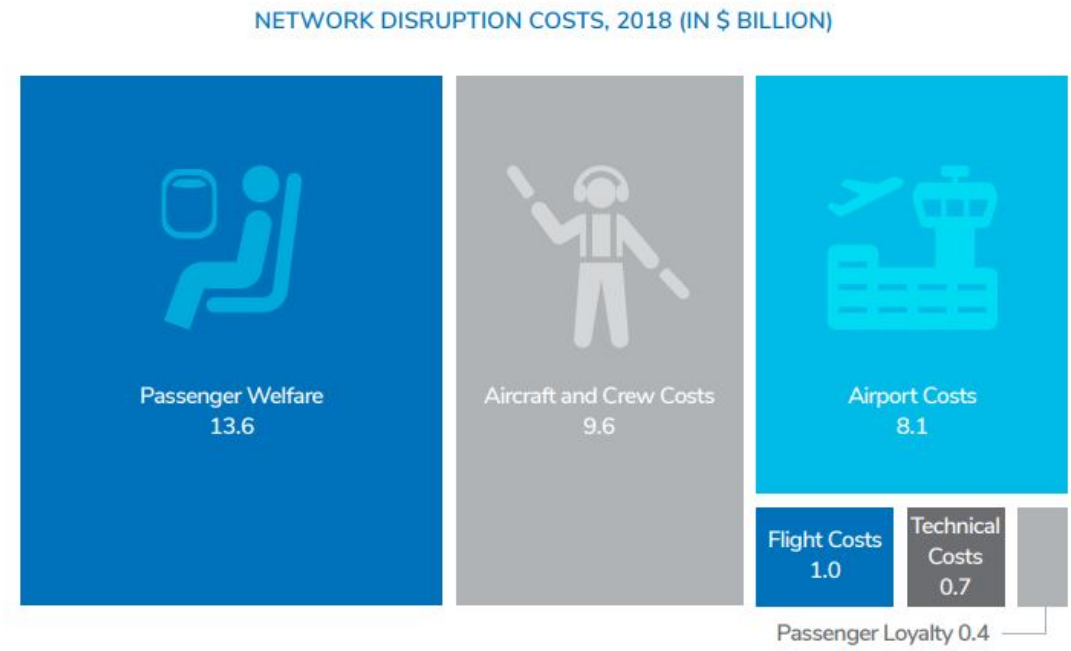

Figure 2.1: Breakdown of costs due to disruptions. From Sullivan [9].

Airlines make their schedules well in advance [8]. A flight schedule created using demand predictions and resource constraints, followed by a fleet assignment which allocates aircraft types while maximizing revenue. Next, aircraft routing determines which aircraft (*tails)* fly which routes, while taking maintenance into account. A month in advance, crew pairings (sequence of flights and duties) are assigned, often solved with a crew bidding algorithm. However, a disruption may occur due to weather, congestion, unplanned maintenance (repairs), employee strikes etc.

Figure 2.3 from Kasirzadeh *et al.* [50] shows how airlines think about their decision making

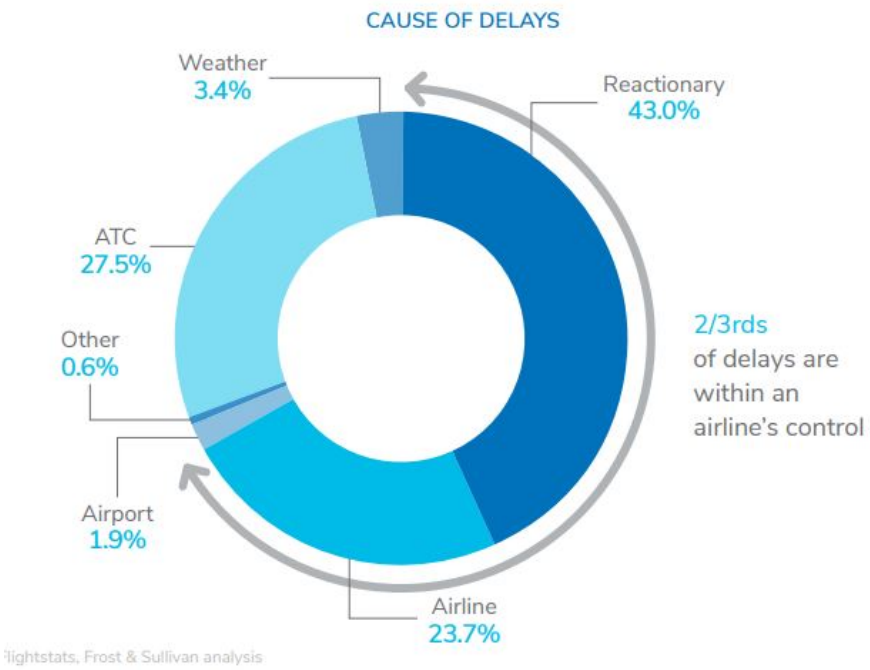

Figure 2.2: Reasons for disruption. From Sullivan [9].

in scheduling. In the planning procedure, airlines decide the routes they are going to fly (flight scheduling) and assign specific aircraft in their fleet to each flight number (fleet assignment). Tight operational and regulatory constraints like aircraft maintenance and routing are considered months in advance, after which crew schedules are handed out typically a month before [50]. Crew schedules are governed by a raft of regulations and preferences for home bases by the crew. Passengers book flight tickets subject to capacity of each flight and cabin class, and expect airlines to stick to the posted schedule. Thus, a carefully pre-planned schedule for flights, crew and passengers are all affected by random events that cause disruption, and a domino effect of the primary disruptions propagate throughout the network. Figure 2.3 shows the problem of dealing with disruptions, called *airline recovery* for flights, crew members and passengers. There is a need for a quick, user independent, consistent and near optimal recovery decision.

The solution of the airline recovery problem by definition requires the schedules to match the original schedules at the end of the recovery window. A host of decisions can be made- adjusting flight plans, aircraft assignments, crew assignments and passenger itineraries within the period of time called recovery period [49]. The length of recovery period depends on the airline and

disruption [51].

There are three major recovery problems- recovering flight schedules (*operations recovery*), crew schedules/pairings (*crew recovery*) and passenger schedules (*passenger recovery*). Traditionally, these problems are solved sequentially: operations, crew followed by passenger recovery. Instead of a global optimization approach, decision making at one level can affect other problems, leading to huge inefficiencies [13]. Integrating different levels of recovery (*holistic recovery*) is a significant goal to achieve.

Though airlines have been using OR models in solving complex planning and operational problems [52], the expanding size of airlines and their networks have led to increased complexity. Notable advances in column generation, which can handle large scale problems with lower computational burdens were popular in this field. Since 2001, there has been focus on robustness [53, 54, 55, 56] where instead of optimality, suboptimal and fast solutions that fare well under uncertainty have been preferred. With advances in processing and clusters, solving large linear or mixed integer programs are becoming easier. Recent work has been focused on the challenge of holistic/integrated recovery [13] with not just integer programs, but heuristics like neighborhood search [49], as well as graph based algorithms [8]. Optimization in the presence of stochastic external disturbances present many challenges. In disruption of networks, an operations recovery solution may be very unfriendly to crew recovery feasibility. The bigger picture here is that we have planned schedules and resource allocations but there could be major drivers of anomaly- unusual events, key performance indicators (e.g. on-time performance) and confidence bounds for robustness. The flight network for an airline is a massive temporal graph with often tight constraints (regulatory or operational). Right now, there is no reliable way to score solutions in the solution space of these problems, giving us no strong measure of quality of a solution, or looking at neighborhoods of a particular point. Therefore a major question is to think of robustness of a solution to unseen disturbances.

As explained in Clausen *et al.* [12], recovering schedules is a complex task, since many re-

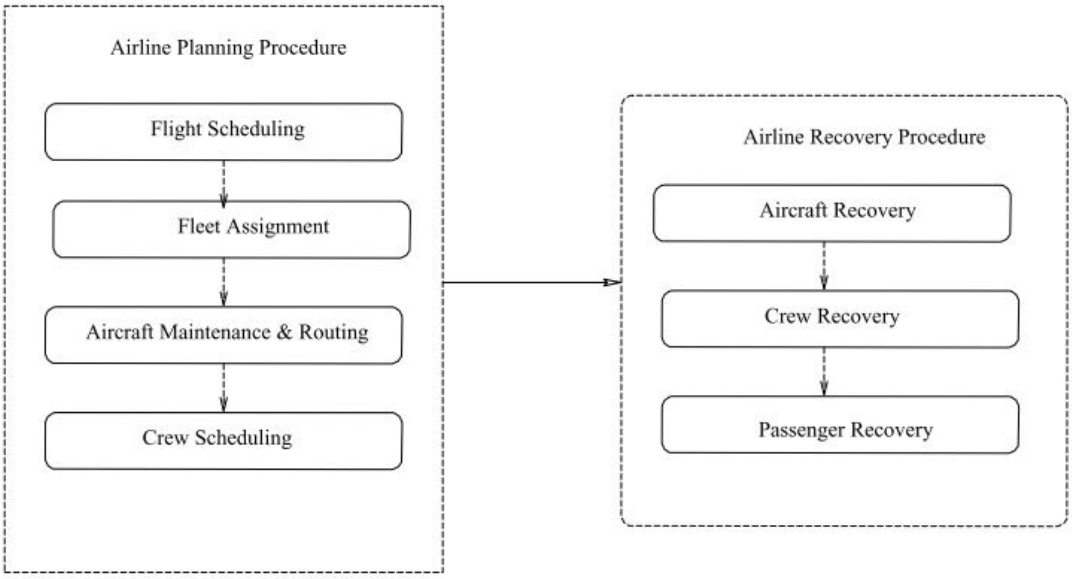

Figure 2.3: Airline decision making. From Kasirzadeh *et al.* [50].

sources (crew, aircraft, passengers, slots, catering, cargo etc.) have to be re-planned. Large airlines focus on the ground problems first, necessitating a sequential process of solving flight infeasibilities first, followed by crew problems. Finally, the impact on passengers is evaluated. The objective function can be composed of several conflicting and sometimes non-quantifiable goals [12]. In most airlines, controllers performing the recovery have only limited decision support to help them construct recovery options or evaluate the quality of the recovery action they are about to implement. Often, controllers are content with only producing one viable recovery plan since there is no time to consider alternatives [12]. **Therefore, instead of designing robust schedules, we prefer to obtain fast (in order of minutes) recovery solutions.**

2.2 Prior Work and Approaches

Airline recovery is a scheduling problem in the presence of a flavor of robustness, where the presence of randomness disrupts pre-planned schedules, and requiring a reactive decision. Of course, one can think of constructing robust schedules at the planning stage that can adapt to disruptions. However, robust planning is a difficult task, particularly due to the unpredictability of disruptions. There have been efforts to set up a theoretical framework to evaluate and quantify

robustness (see a summary in Clausen *et al.* [12]). The situation, however, resembles that of approximation algorithms: if mathematical proofs are to be given, the results turn out to be weak, and the methods and strategies are far from those known to be most efficient in practice [12]. Hence, we focus on managing the robustness by re-planning schedules within the computational requirement, given a pre-planned flight schedule that we do not stray too far away from in the solution.

Recall the definition of integrated recovery- to solve a global optimization problem in the space (at least two among flight, crew and passenger recovery together). In operations research, the overall approach to airline recovery problems can be classified according to the following categories (see surveys in Filar *et al.* [10], Ball *et al.* [11], Clausen *et al.* [12], and Castro *et al.* [14]):

- Non-integrated recovery - only one of the three networks - or dimensions - is considered.
- Non-simultaneous integrated recovery - airline recovery considers the three dimensions– - aircraft, crews and passengers - but separately, not simultaneously.
- Simultaneous integrated recovery - the three dimensions - aircraft, crews and passengers are treated simultaneously, without imposing degrees of importance to them.
- Partially integrated recovery - airline recovery considers two dimensions, simultaneously or not

Right now, we are solving for non-simultaneous integrated recovery, for the passenger recovery problem, and moving towards partially integrated recovery, incorporating flight into the passenger problem.

A summary of important work related to non-simultaneous integrated recovery, simultaneous integrated recovery and partially integrated recovery is presented in Figure 2.4.

2.2.1 Non-simultaneous integrated recovery

In [58], the authors present a sequential modeling with a distinct approach for each of the problem dimensions, aiming to minimize operating costs and passenger costs. For the aircraft dimen-

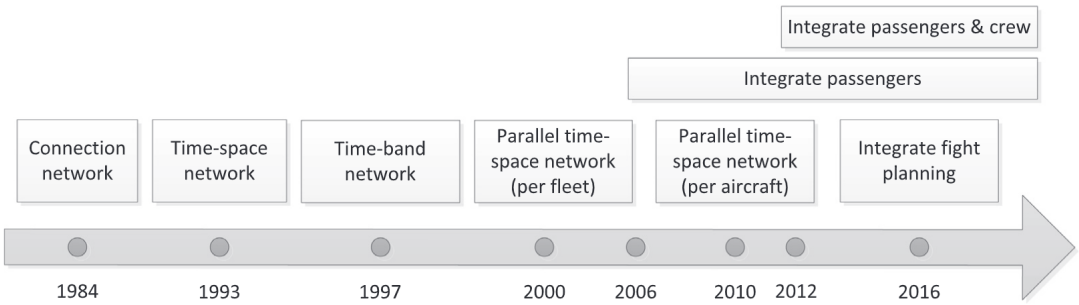

Fig. 1. Timeline of developments in modeling the aircraft recovery problem.

Figure 2.4: Timeline of developments in modeling the aircraft recovery problem , from Vink *et al.* [57]

sion, a local search heuristic was used; for the crew dimension, a whole programming approach was used with resolution per column generation; and for the passenger dimension, multi-commodity network programming was used. The model considers the cost of delay at the final destination for the relocated passenger, direct costs for rescheduled passengers (hotel, food, etc.), impact on customer perception and costs related to class upgrade and downgrade (in this case strongly impacting the customers experience and their willingness to fly again with the company).

[59, 60] present a sequential approach using programming and resolution with a column generation method; passenger recovery occurs after the resolution for aircraft and crew dimensions. The passenger cost depends on the number of passengers relocated due to flight cancellation and also on the delay costs for passengers relocated to other flights.

[61] use mixed mathematical programming modeling, applying benders' decomposition method. Passenger costs depend on the number of passengers impacted by cancellations and on delays to passengers related to their final destination.

[62] present a large search in the vicinity heuristic, considering, for passengers, the costs of delay, flight cancellation and cabin downgrade

2.2.2 Passenger Recovery- motivation and prior work

Arguably, passenger recovery is the most relevant problem for airline disruption management since high passenger delay cost and continuous flight disruptions will lead to a potential loss of

goodwill and long-term reputation damage. Passenger recovery can be formulated as follows: given a recovered flight and crew schedule and a set of disrupted passenger itineraries, re-assign to each disrupted itinerary the (recovered) flights necessary (given seat availability) to accommodate passengers from their current position to their destination while minimizing cost. These passenger recovery costs can include both hard and soft costs. Hard costs are directly incurred when a passenger cannot complete its scheduled itinerary (e.g., compensation for delay and cancellation as stipulated by government regulations). Soft costs are the potential losses of future revenue as a result of passenger inconvenience, possibly causing the passenger to switch to a different airline in the future. These costs are approximations made by the airline and can differ per passenger class or frequent flyer status. Alternatively, these passenger disruption costs are minimized by minimizing the total number of passenger delay minutes.

For the soft cost, nearly all papers that focus on Passenger Recovery (either stand-alone or in combination with Aircraft and/or Crew Recovery) assume linear delay costs – i.e., a 2-h delay is twice the cost of a 1-h delay. The authors in Cook *et al.* [63] studied the inconvenience experienced by passengers as a function of delay. duration. The study has shown that the delay cost as a function of delay duration can be represented as a sigmoid function. Studies that incorporate such a relation generally use a piece-wise linear relation for delay costs, if they seek to prevent a nonlinear recovery model.

As shown in Figure 2.5 below, in the period 2009–2020, there has been one single publication simply addressing the passenger recovery problem as a stand-alone recovery problem. In that work, McCarty and Cohn (2018) presented a two-stage stochastic to deal with the rerouting of passengers, re-accommodating passengers as soon as a delay is known and before the length of the delay is realized. In the first stage, passengers are preemptively assigned to new itineraries as soon as it is known that a flight will be delayed and in anticipation of the delay's impact. The second stage further modifies itineraries for passengers who miss connections after the delay has been realized. Benders decomposition is used to solve the problem within reasonable computation times. The presented method is tested on a case study using a real-life flight schedule with 15

generated delay variations of a single flight. The case study consists of 1144 flights and in the different test instances, there are 50, 100, or 200 passengers on the delayed flight. For the 15 test instances, the final destination of each passenger on the delayed flight is randomly selected. On average, all test instances were solved within 115 s.

Overview and classification for literature focusing on Passenger Recovery.

Paper	Network	Type	Solution Approach	Disruption types				Recovery actions				Problem characteristics		Data	Largest Case dimensions			CPU
				Flight Delay	Flight Canx.	AC U/A	Airport Disruption	Flight Delay	Flight Canx.	AC Swap	Pax Itinerary change	Multi-Fleet	Maint.		Aircraft	Fleets	Flights	[sec]
McCarty and Cohn (2018)	-	EX	2-level stochastic problem with Benders decomposition	Y	N	N	N	N			Y			RL			1144	93,9

Abbreviations used in table: U/A: Unavailable, AC: Aircraft, Canx: Cancellation, Maint: Maintenance constraints, Pax: Passengers, CPU: Computation time in seconds,
EX: Exact method, MHL: (Meta-) heuristic, HH: Hybrid heuristic, O: Others, MA: Multi-Agent System,
Y: Included or mentioned, N: Not included nor considered, ' ': Not mentioned or not relevant, F: Airport Flow restriction, C: Airport Closure, G: Generated Data, RL: Real-life Data.

Figure 2.5: Overview of Literature on Passenger Recovery since 2010. From Hassan *et al.* [64]

2.2.3 From passenger recovery to integrating flight recovery

As mentioned earlier, there has been a trend towards integrating more than one resource in recovery models. Sequential optimization approaches do not fully capture the interdependencies between aircraft, crew, and passengers and therefore usually result in sub-optimal recovery solutions. The papers in this section attempt to overcome these downsides by simultaneously solving the aircraft and passenger recovery. The overview of the papers addressing both aircraft and passenger disruptions is presented in Figure 2.6 below.

Of the studies addressing aircraft and passenger recovery the majority considers aircraft unavailabilities (81%) and flight delays (75%) as disruption types, less than half consider airport disruptions (44%). From the 16 papers reviewed, all considered flight delays as a recovery action and the majority of papers considered flight cancellations (88%), aircraft swaps (94%), and/or passenger itinerary changes (88%) as well. This means that two studies ([65]; [66]) do not explicitly model passengers and their itinerary recovery.

2.3 Our contributions

Thus in this work, we focus on new approaches to the passenger recovery problem. When there is a disruption and we are able to obtain the operations and crew solutions, we follow a sequential

Overview and classification for literature focusing on Aircraft and Passenger Recovery.

Paper	Network	Type	Solution Approach	Disruption types						Recovery actions								Problem characteristics		Data	Largest Case dimensions			CPU
				Flight Delay	Flight Canx	AC U/A	Airport Disruption	Crew U/A	Other	Flight Delay	Flight Canx.	Create flight	AC Swap	Reserve AC	Ferry AC	Cruise speed control	Pax Itineray change	Other	Multi-Fleet	Maint.		Aircraft	Fleets	Flights	[sec]
Jafari and Niloofarafari and Niloofar (2010)	Connection	EX	Rolling horizon time framework with MILP	Y		Y	N			Y	Y		Y	Y	Y		Y	Over-flying	Y	N	RL	13	2	100	
Zegordi and Hessameddin (2010)		MH	Ant Colony Optimization	Y		Y	N			Y	Y		Y			Y			Y	Y	RL	13	2	100	26
Jafari and Niloofar (2011)	Connection	EX	Rolling horizon time framework with MILP	Y		Y	N			Y	Y		Y	Y	Y		Y	Over-flying	Y	Y	RL	13	2	100	
Bisaillon et al. (2011)	Time Space	MH	Large Neighboorhood Search	Y		Y	C			Y	Y	Y	Y				Y		Y	Y	RL	256	1	1423	<600
Mansi et al. (2012)	Time Space	HH	Math heuristics	Y	Y	Y	F		Personnel strikes	Y	Y		Y				Y		Y	Y	RL	618	1	2178	<600
Jozefowiez et al. (2013)	Connection	MH	Heuristic based on shortest path	Y	Y	Y	F			Y	Y	Y	Y				Y		Y	Y	RL	618	1	2178	230
Sinclair et al. (2014)	Time Space	MH	Large Neighboorhood Search	Y	Y	Y	F			Y	Y	Y	Y				Y		Y	Y	RL	256	1	1423	<600
Hu et al. (2015)	Time Band	EX	Integer programming model	N	N	Y	N			Y	Y		Y				Y		Y	N	RL	188	13	628	172
Sinclair et al. (2016)	Time Space	HH	Large Neighboorhood Search with Col Generation	Y	Y	Y	F			Y	Y		Y						Y	Y	RL	618	1	2178	1315
Arikan et al. (2017)	Time Space	EX	Conic quadratic MILP	Y						Y	N		Y		Y	Y			Y	N	RL		6	1429	<142
Zhang et al. (2016)	Time Space	MH	Sequential three stage heuristic	Y	Y	Y	F			Y	Y	Y	Y				Y		Y	Y	RL	618	1	2178	<420
Hu et al. (2016)	Time Space	MH	GRASP			Y				Y	Y		Y				Y		Y	N	RL	87	3	340	<100
Marla et al. (2017)	Time Space	EX	Rolling horizon time framework with MILP	Y						Y	Y		Y		Y	Y			Y	Y	RL			250	<120
Santos et al. (2017)		EX	Rolling horizon time framework with MILP	Y						Y	N		N				Pax reallocation		Y		RL			250	< 3600
Yang and Tianshun (2019)	Time Space	MH	Multi-objective Genetic Algorithm			Y				Y	Y		Y				Y		Y	N	RL	59	1	209	<11
Vink et al. (2020)	Time Space	HH	MILP combined with a fleet selection method	Y	Y	Y	C			Y	Y		Y			Y			Y	Y	RL	100	2	600	<44

Abbreviations used in table: U/A: Unavailable, AC: Aircraft, Canx: Cancellation, Maint: Maintenance constraints, Pax: Passengers, CPU: Computation time in seconds,
EX: Exact method, MHL: (Meta-) heuristic, HH: Hybrid heuristic, O: Others, MA: Multi-Agent System,
Y: Included or mentioned, N: Not included nor considered, ' ': Not mentioned or not relevant, F: Airport Flow restriction, C: Airport Closure, G: Generated Data, RL: Real-life Data.

Figure 2.6: Overview of Literature on Aircraft + Passenger Recovery. From Hassan *et al.* [64]

recovery process. The goal is to come up with an efficient and close to optimal passenger recovery solution.

We construct and tested an improved Mixed Integer Linear Program (MILP) compared to existing literature, that can solve passenger recovery. We also have post optimization approaches to improve the solution.

More importantly, we are interested in developing new algorithms without integer programs. Using efficient preprocessing and multi label shortest paths with batches and rounding, we have a solution which is closer to optimal and has demonstrably faster running time on a test dataset.

2.4 Basic concepts of our problem space

We describe the problem space with the setup prescribed in Palpant *et al.* [13], our primary data set to illustrate our algorithms' performance.

2.4.1 Flights and aircrafts

A *flight schedule* is a set of all flights operated by an airline in a given time period [13]. Each flight in the schedule is defined by:

- a flight number, e.g. Flight 261.
- origin and destination airport which define the length of the flight, e.g. JFK to LAX.

- flight type (domestic, continental or intercontinental) which are abbreviated D/C/I.
- departure time and date, arrival time and date (e.g. 1/1/2018 08:00 AM).

Given an aircraft a (e.g. Boeing 747 with tail number 1234), a *rotation* is the sequence of flights assigned to this aircraft. This also comes with constraints on continuity, origin airport of a flight is the same as destination of the previous flight flown by that aircraft as well as time continuity. Figure 2.7 is an example of a rotation of an aircraft.

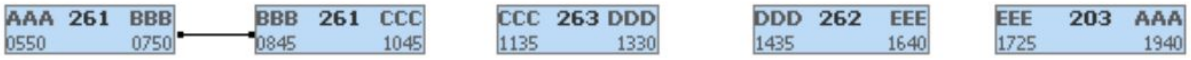

Figure 2.7: Example of a rotation. From Palpant *et al.* [13].

2.4.2 Crews

Crew can be of two types- cockpit (pilot and copilot) and cabin (flight attendants). The size of the crew for each flight depends on the model and configuration of the aircraft, as well as length of the flight. Computing crew pairings (a sequence of flights that a crew member is assigned to) is computationally challenging [52]. There are regulations and contractual rules on a minimum sit and rest time, the elapsed time and the flying time of a duty is upper bounded, and there is the complicated 8-in-24 rule imposed by the FAA. Depending on if it is an international flight, the EU has their own regulations on crew pairings as well. Even on a medium size problem, the combinatorial nature of the problem means the number of pairings is enormous (e.g. Fleets with 200 flights can have 1 billion pairings).

In the planning stage, an optimization team within an airline solves the crew pairing problem to generate daily pairings for each crew member. When disruptions occur, some pairings are *broken*, meaning some flights are delayed/cancelled and the crew cannot complete all duties in their pairing. [67]. There have been many large scale disruption incidents (e.g. March 1993 eastern seaboard super storm [56]) that have motivated the need for effective solutions for crew recovery.

If the crew recovered solution has flights which can be scheduled but do not have enough crew to cover them, then these are *open flights*. Open flights are often cancelled by airlines, a waste of resource which could have been avoided.

2.4.3 Passengers

Passengers have reservations on flights. We group passengers into a set K of itineraries- each itinerary has the following features:

- a unique itinerary number
- passenger count, number of passengers in the itinerary
- average cost paid by a passenger in this itinerary
- nature of itinerary- inbound or outbound (I/O)- outbound is a one way trip or outbound portion of a round trip whereas Inbound is the return portion of a roundtrip whose outbound portion was finished before the planning horizon. **Inbound is preferred over outbound** as passengers should not be left stranded in an intermediate airport (which might require hotel costs to be paid by the airline).
- description of the itinerary- one or several flight legs, with one cabin class for each leg (E- Economy, B- Business, F-First).

An example of an itinerary is

```
2 Out $1752.5 27 F243 20/01/08 B F245 21/01/08 B
```

This is itinerary 2 with 27 passengers booked from Singapore to London (Flight 243) and London to Paris-Charles de Gaulle (Flight 245), on the outbound portion of their trip, travelling in business class on both flights, and having paid $\$1,752.5$ on average.

2.4.4 Disruption and recovery period

The goal in each of the problems- operations recovery (for flights), crew recovery and passenger recovery is to resume normal operations as quickly as possible, after disruptions to the planned schedule. We model the problem as a specific time at which a disruption occurs, followed by the *recovery period* for which we must determine new schedules and at the end of the recovery period, the original schedules must resume. Recovery period depends on the needs and philosophy of each airline (some do 24 hours, some do up to 96 hours) [51].

One example how airlines perceive disruptions is from Air Traffic Control or Airport Control- they are told to reduce their in and out flight volume by say 50% between $3:00-8:00$ PM at JFK airport. In this case, the disruption occurs at $3:00$ PM and the airline can take say 24 hours to recover until the normal schedule resumes at $3:00$ PM the next day. Recall in Figure 3.1, that half of all delays (43%) are reactionary in nature and caused by the primary delay. Disruptions

Recovery problem	**Possible decisions airlines could make**
Operations	- Intentional flight cancellations and delay - Possible additions of flights - Aircraft changes
Crew	- Crew put in delayed flights - Cancelled duties and overnight stays - Calling up reserve crew
Passenger	- Re-accommodate on a new itinerary - Downgrade cabin class on original itinerary - Cancellation of trip

Table 2.1: Decisions for different recovery problems

propagate- one flight delayed for 3 hours in JFK means that aircraft is unavailable at the destination at the required time thus delaying another flight, and the crew members are also not available for the next flight though another aircraft and its passengers could be ready.

2.4.5 Objective function

The objective function in all recovery problems is the cost to the airline and depending on the type of recovery problem, it ranges from flight and aircraft operating costs, crew salaries and

overtime, passenger disutility (to model inconvenience) as well as refunds for any cancellations. We will take an expanded look in following sections.

2.4.6 Variables

K : Set of all original itineraries

K_{dis} : Those itineraries in K that are disrupted

M : Cabin class set $\{F, B, E\}$

For an itinerary $k \in K$,

y_{fmk} : Number of passengers moved through flight f in itinerary k and class m

y_{ijmk} : Number of passengers moved between airports $i \rightarrow j$ in itinerary k and class m

n_k : number of passengers in the itinerary

V_k : Set of all feasible airports in the itinerary with source s and sink t

A_k : Set of all feasible arcs in the itinerary, including dummy arc from s to t

F_k : Set of all feasible flights in the itinerary including dummy flight g from s to t

Other parameters,

F : Set of all flights $\cup_{k \in K_{dis}} F_k$

$cap(f, m)$: Capacity of flight f in class type

2.4.7 Regulation costs

We can model **mandatory** delay and cancellation costs for the passengers, an example from the ROADEF challenge in Palpant *et al.* [13] inspired by the European Union regulations is shown below (we use the same costs to compare performance):

- The airline must provide drinks and a meal in case of a delay longer than: two hours on a trip with an initially planned duration strictly less than two hours; three hours on a trip with a duration greater than or equal to two hours and strictly less than four and a half hours;

four hours on a trip with a duration greater than or equal to four and a half hours. For this challenge, the cost associated with this item is assumed to be 15 euros per passenger.

- In addition to the previous item, the airline must provide lodging (if necessary) in the case of a delay longer than five hours. The cost of a hotel night is assumed to be 60 euros per passenger for this problem.

- In the case of a cancellation, the airline must reimburse the ticket price regardless of the length of the trip, as well as provide financial compensation. The financial compensation per passenger is: 250 euros for a trip with an initially planned duration strictly less than two hours; 400 euros for a trip with a duration greater than or equal to two hours and strictly less than four and a half hours; 600 euros for a trip with a duration greater than or equal to four and a half hours.

2.4.8 Passenger disutility costs

Apart from regulation costs, we need to measure disutility perceived by a passenger when we delay, downgrade or cancel their flights. The precise objective function is too complex to be completely expressed and not trivial to compute [68]. We penalize the objective of the recovery schedule if any of the following are not fulfilled as much as possible, as explained in Jozefowiez *et al.* [62]:

- Passengers should not be delayed or cancelled

- Maximum delay in the recovered itinerary should not exceed a threshold (e.g., 18 hours for domestic flights or 36 hours for international flights, depending on the airline's philosophy).

- Passengers should not be downgraded from their reference class (e.g., a business class passenger being assigned an economy seat).

- By the end of recovery period, passenger schedules should mirror the originally planned schedules.

Given an itinerary k, if an itinerary is composed of several legs, the itinerary's reference cabin class is assumed to be the highest of the booking cabin classes on those legs. All costs will be calculated based on this cabin class. The itinerary type is defined as the type of its longest leg (I = intercontinental > C = continental > D = domestic).

Downgrading cost: Costs associated with downgrading are applied only in the case of re-accommodation of a passenger. They are calculated on an individual leg basis, for all legs of the recovered itinerary. For each leg, these costs depend upon the type of the leg and the level of downgrading (difference between the itinerary's reference cabin class and the cabin in which the passenger actually travels on that leg).

Delay cost: These are only applied if there is a net delay at the destination between original and recovered schedule, irrespective of changes in intermediate airports. This is linear in the delay length, with the slope determined by flight type (I/C/D) and reference cabin class of the passenger.

Cancellation cost: This is obviously much larger than the maximum delay cost for that trip; the cancellation cost is much larger in case of a return (inbound) portion of a trip, hence we would prefer to cancel an outbound over an inbound portion of a trip. In practice, we assign a dummy flight from source to destination with the cancellation cost, any passenger assigned to this itinerary is meant to have their trip cancelled and refunded.

Thus, the passenger disutilities in the objective of the recovery are modeled as a linear combination of downgrading, delay and cancellation costs.

2.4.9 Inputs to the problem

1. **Configurations:** Start and end of recovery window, cost parameters for delay, downgrading and cancellation.

2. **Original and Recovered Flights:** List of original flights and solution to Ops recovery prob-

lem, i.e. recovered flights after disruption, both in the format:

```
Flight Orig Dest DepTime ArrTime PrevFlight
```

3. **Aircraft information:** For each aircraft type, e.g. A320, we have their info in the format

```
Aircraft Model Seat Capacities
```

4. **Itineraries:** Original planned itineraries.

2.4.10 Output to be computed

We want the **recovered itineraries**, i.e. a solution to the passenger recovery problem given original itineraries, original flight schedules, recovered flight schedules and recovery window.

Various approaches to solve passenger recovery are discussed in the next section.

2.5 Algorithms for Passenger Recovery

In this section, we present a Mixed Integer Linear Program (MILP) to solve for recovered passenger itineraries from scratch, and a new network based approach based on multi-label shortest paths.

2.5.1 Mixed Integer Program formulation

We write the MILP formulation using the variables defined in Section 2.4.6.

$$\min_{y} \quad \sum_{k\in K_{\text{dis}}} \sum_{m\in M} \left[\sum_{(i,j)\in A_k} \sum_{f\in F_k(i,j)} c^{\text{down}}_{fmk}\, y_{fmk} + \sum_{(i,t)\in A_k} \sum_{f\in F_k(i,t)} c^{\text{del}}_{fmk}\, y_{fmk} + c^{\text{can}}_{gmk}\, y_{gmk} \right] \tag{2.1}$$

$$\text{s.t.} \quad \sum_{i:(i,j)\in A_k} \sum_{m\in M} y_{ijmk} - \sum_{i:(j,i)\in A_k} \sum_{m\in M} y_{jimk} = 0, \;\; [\forall\, k \in K_{\text{dis}}, j \in V_k \setminus \{s,t\}] \tag{2.2}$$

$$\sum_{j:(s,j)\in A_k} \sum_{m\in M} y_{sjmk} = n_k, \;\; [\forall\, k \in K_{\text{dis}} \text{ where } s : \text{source}(k)] \tag{2.3}$$

$$y_{ijmk} = \sum_{f\in F_k(i,j)} y_{fmk}, \;\; [\forall\, k \in K_{\text{dis}}, (i,j) \in A_k] \tag{2.4}$$

$$\sum_{k\in K_{\text{dis}}} \mathbb{I}\{(i,j) \in A_k, f \in F_{ij}\} y_{fmk} \leq \text{cap}(f,m), \;\; [\forall\, f : (i \to j) \in F] \tag{2.5}$$

$$y_{fmk} \leq \text{cap}(f,m), \;\; [\forall\, f \in F, m \in M, k \in K_{\text{dis}}] \tag{2.6}$$

$$y_{fmk} \in \mathbb{Z}_{\geq 0}, \;\; [\forall\, f \in F, m \in M, k \in K_{\text{dis}}] \tag{2.7}$$

In the objective function (2.1), $\sum_{(i,j)\in A_k} \sum_{f\in F_k(i,j)} c^{\text{down}}_{fmk}$ is the downgrading cost for all passengers assigned to flight f in cabin class m in the itinerary k. Similarly we have $\sum_{(i,t)\in A_k} \sum_{f\in F_k(i,t)} c^{\text{del}}_{fmk}\, y_{fmk}$ for the delay costs. The cancellation cost $c^{\text{can}}_{gmk}\, y_{gmk}$ is for those passengers assigned on a dummy flight g which represents a flight that is cancelled in the itinerary k. This is summed over all disrupted itineraries and all cabin classes.

The first constraint (2.2) is a flow constraint- for every itinerary and every intermediate airport j which is not the origin s or destination t of the itinerary, the number of passengers of that itinerary coming in to that airport is the same as number of passengers of that itinerary going out of that airport.

The second constraint (2.3) makes sure that for a given itinerary k, the number of passengers leaving the source airport s is n_k, the passenger count of that itinerary.

The third constraint (2.4) splits the decision variables y_{fmk} which is the number of passengers in an itinerary k and cabin class m into multiple variables y_{ijmk} splitting them into passengers

moved between airports $i \rightarrow j$.

The fourth constraint (2.5) is the interesting one- across all itineraries k, the number of passengers utilizing a flight f in cabin class m should be at most the remaining capacity available in f in cabin class m. This constraint is the one preventing parallelization since it uses all itineraries k to write out. Else all constraints and the objective function could be handled for each itinerary separately.

The fifth constraint (2.6) upper bounds the decision variables with the capacity of the flight and cabin class, and the sixth constraint (2.7) ensures the variables are nonnegative integers.

To solve for recovered passenger itineraries, we provide Algorithm 4 below.

If any $k \in K_{dis}$ is disrupted, we cancel the whole itinerary. For each such itinerary, we want to create new schedules for those passengers using the MIP (2.1) to get the recovered passenger itineraries. For example, even if only a passenger's second leg is disrupted, we cancel the whole itinerary and re-plan for a better solution. A simple post processing step can construct the recovered passenger itineraries from the output variables y^*_{fmk}.

Algorithm 4: Compute recovered passenger itineraries using MIP (2.1)

Input: Configurations, original and recovered flights, airline information and itineraries `// as given in Section 2.4.9`
Output: $K_{recov} = \{K_{dis}$ sorted in decreasing order of cancellation costs$\}$ `// set of recovered itineraries`

1: Solve the MIP (2.1) and obtain the outputs y^*_{fmk}.
2: **for** $k \in K_{dis}$ **do**
3: Construct recovered itinerary for a passenger in k by following the paths in $\{(f, m) : y^*_{fmk} > 0\}$ from source to sink of that itinerary
4: **end for**
5: Output K_{recov}, the set of all recovered itineraries.

2.5.2 Solution Approach using multi label shortest paths

Beyond a MILP, we have developed a graph based algorithm (using multi label shortest paths) which could be parallelized. In Bisaillon *et al.* [8] for examples, the authors use a step where they solve a series of shortest path problems with arc labels. On flight networks, each arc corresponds

to a flight and is orders of magnitude higher than the number of nodes (airports). We present an algorithm based on computing multiple labels for each node, and solving a shortest path problem based on these labels.

2.5.3 Preprocessing

From the original schedules, and knowledge of disruptions, a preprocessing step can be performed, without impacting the running time of algorithms that can compute passenger recovery. There are two major steps in preprocessing,

1. Computing flight/crew recovery solutions, that become input to the passenger recovery approach under the sequential model. We can use MIP based algorithms to compute flight recovery from prior literature [56].

2. Constructing graphs of the time-space flight network that can be used by passenger recovery algorithms. Given an itinerary $k \in K_{dis}$, we create the graph $G_k = (V_k, A_k)$ with source s_k and sink t_k. This represents all the feasible flights using the recovered flights data, i.e. we want to know all possible combinations a passenger can take from s_k to t_k. We prune this graph so that there are no flights that reach beyond the end of the recovery window, as well as upper bound the number of legs a passenger can take to reach their destination.

We present our preprocessing Algorithm 5. We also build on Algorithm 5 by rounding flight departure and arrival times in the flight network. While Algorithm **??** increases the complexity of preprocessing the flight network, we aim to construct faster algorithms at the expense of suboptimality.

2.5.4 Shortest Path approach

With the graphs obtained from preprocessing, we present our algorithm for computing recovered passenger itineraries. The key constraint in the passenger recovery problem are the remaining capacities of the flights (recall we are only solving for itineraries that are disrupted, and do not

Algorithm 5: Preproccesing method PP1

Input: Configurations, original and recovered flights, airline information and itineraries as given in Section 2.4.9. Let there be M airports in the airline network.

Output: For each itinerary k, graphs G_k representing all feasible flights using recovered flight data.

Given an itinerary k, store the source s and start time $t(s)$ of the graph G_k.
$\Phi \leftarrow \emptyset$ `// set of visited nodes`
$u \leftarrow s$ `// current visited node`
while $|\Phi| < M$ **do** `// until we hit the sink of the itinerary`
  Let $N(u)$ be the neighboring nodes of v in the network.
  for each v in $N(u)$ **do**
    Compute the list of all possible flights that start from u after time $t(u)$ and reach v within end of the recovery window.
    If there is at least one such flight, add v to the graph G_k as a node, and each such flight between u to v as an arc subject to pruning rules: Ignore if distance of v from s exceeds the upper bound on number of legs in the passenger itineraries. We also drop flights that arrive after the original end time of itinerary k plus a threshold (typically $24 - 96$ hours depending on airline), so we would rather cancel itineraries than make the passengers very late.
  end for
  $\Phi \leftarrow \Phi + \{u\}$
  if $N(u)$ is nonempty and $N(u) \not\subseteq \Phi$ **then** `// if unvisited nodes in neighbor`
    $u \leftarrow$ some unvisited vertex in $N(u)$
  end if
end while

touch the undisrupted itineraries). Among the three major costs in the passenger recovery problem, cancellation costs are the highest by an order of magnitude (an example is provided in Section 2.4.7).

When solving in a MIP based approach, we solve for all the disrupted itineraries together, whereas an iterative graph based approach requires prioritizing the commodity in low supply, i.e., remaining flight capacities for high value flights. We therefore sort all disrupted itineraries in decreasing order of cancellation costs, and aim to compute recovered flight schedules for each itinerary in this order. During a real world disruption, there is typically an end of recovery window shorter than what is needed to replan all itineraries, and a certain fraction of itineraries end up getting cancelled, hence solving first for highest cancellation cost itineraries is justified.

The key idea is to computing and storing multiple labels for each node. Each label has three components, and corresponding to a choice of flights that construct a path from origin to destination of the itinerary (third component), along with costs (delay and downgrading) in the second

component, and number of legs in the itinerary as the first component (we prune labels that have too many legs).

We parse the graph G_k in a BFS order, and compute many labels while pruning labels that are dominated in the first two components. For each flight f in the recovered flight schedule, we compute downgrading cost for choosing that flight and each cabin class. This cost differs depending on f being in the original flight schedule, in which case we can directly compare if a passenger is being downgraded in that flight. Else, we can make a reference cabin class for the passengers in the itinerary based on the highest cabin class in their original schedule, and compare to the cabin class we are choosing in the current flight f.

The challenge in a shortest path graph approach is that delay costs cannot be computed until we reach the sink labels. After labels are computed for the sink using only downgrading costs, we can finally check the third components of each label that show us the flights chosen in the path, and compare the arrival time of the destination flight to the original destination time for the passengers to compute delay costs (which are often linear in the number of minutes of delay).

Finally, we prune all the labels, and identify the top N shortest path labels at the destination, and allocate the passengers according to the flight paths in these labels to create recovered passenger itineraries. We pay the delay and destination costs according to the second component of the chosen labels. We cancel the itineraries of the passengers who are unable to be accommodated, and pay the cancellation costs. Algorithm 6 shows the pseudocode.

Because of the iterative solving for itineraries based on cancellation costs, suboptimality is possible due to exhausting precious flight capacities in earlier iterations. Algorithm 6 aims to address this by solving for *batches* of itineraries together. We can expect a solution whose costs are lower in Algorithm 7 than Algorithm 6, since batching removes some suboptimality in solving iteratively. Batch sizes have to chosen according to the available data, as we illustrate experimentally.

Another improvement could be to simplify the flight network using the preprocessing step Algorithm **??**, and implementing the batched multi label shortest paths. We consider this case as

Algorithm 6: Multi Label Shortest Paths approach

Input: Configurations, original and recovered flights, airline information and itineraries as given in Section 2.4.9, and graphs G_k from PP1
Output: K_{recov}, the set of all recovered itineraries
Sort the disrupted itineraries K_{dis} by decreasing order of cancellation costs
for $k \in K_{dis}$ **do**
    Initialize source s and all nodes of G_k with the label 3-tuple $(0, 0, \emptyset)$
    Update labels in a BFS manner as follows
    Start with $v \leftarrow s$
    while *There are nodes not visited in* G_k **do**
        for *each airport* v_i *adjacent to* v *and each flight* f *between them* **do**
            Create a new label for v_i for each cabin class m as follows
            if *Flight* f *is in the original itinerary* k **then**
                Check if there is a strict downgrading by comparing the original cabin class and m, and update the second component of v_i's label
            else
                Use the reference cabin class to calculate downgrading cost and update the second component.
            Update the first component of the label by 1 to capture the number of legs so far.
            Store the flight f to the third component that maintains list of flights in the path
            If for any node, one label dominates another strictly in the first two components, drop the latter.
            Set $v \leftarrow v_i$.
        For each label of the itinerary sink t, compute delay costs by comparing arrival time in the constructed itinerary and the original destination arrival time, add delay costs to the second component of each label.
        Prune dominated labels and identify the top N shortest paths, sort by second component of label
        For each of the selected labels, consider the smallest remaining capacity among all flights in its path, and allocate as many passengers as possible among the n_k passengers in the itinerary.
        Add this solution to K_{recov} and update the remaining flight capacities.

well in the experiments.

2.6 Experiments

In this section, we give a comprehensive experimental treatment to show the performance of our proposed algorithms.

Algorithm 7: Batched Multi Label Shortest Paths

Input: Configurations, original and recovered flights, airline information and itineraries as given in Section 2.4.9, and graphs G_k from PP1

Output: K_{recov}, the set of all recovered itineraries

Sort the disrupted itineraries K_{dis} by decreasing order of cancellation costs into batches of equal size β.

for *each batch* $B_k \in K_{dis}$ **do**

- Combine the graphs G_k for all itineraries within the batch to graph G_{B_k}.
- Initialize source s and all nodes of B_k with the label 3-tuple $(0, 0, \emptyset)$
- Update labels in a BFS manner as follows
- Start with $v \leftarrow s$
- **while** *There are nodes not visited in* B_k **do**
  - **for** *each airport* v_i *adjacent to* v *and each flight* f *between them* **do**
    - Create a new label for v_i for each cabin class m as follows
    - **if** *Flight* f *is in the original itinerary* k **then**
      - Check if there is a strict downgrading by comparing the original cabin class and m, and update the second component of v_i's label
    - **else**
      - Use the reference cabin class to calculate downgrading cost and update the second component.
    - Update the first component of the label by 1 to capture the number of legs so far.
    - Store the flight f to the third component that maintains list of flights in the path
    - If for any node, one label dominates another strictly in the first two components, drop the latter.
    - Set $v \leftarrow v_i$.
  - For each label of the itinerary sink t in the batch, compute delay costs by comparing arrival time in the constructed itinerary and the original destination arrival time, add delay costs to the second component of each label.
  - Prune dominated labels and identify the top N shortest paths, sort by second component of label
  - For each of the selected labels, consider the smallest remaining capacity among all flights in its path, and allocate as many passengers as possible among the n_k passengers in the itinerary.
  - Add this solution to K_{recov} and update the remaining flight capacities, and move to the next batch.

2.6.1 Data

The French Operational Research (OR) and Decision Support Society (ROADEF) organizes an OR challenge dedicated to industrial applications, in collaboration with an industrial partner every other year. The theme in 2009 was to integrate operations and passenger recovery [13]. Multiple teams of researchers participated with the top finalists publishing their work [8, 69, 70, 71]. We use the public dataset from the challenge to test out our algorithms. The size of our dataset is shown

below:

Table 2 - Test instances description - Source: Palpant et al., 2009

Instances	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
Number of flights to be analyzed per day of the recovery window	608	608	608	608	608	608	608	608	608	608
Number of aircraft	85	85	85	85	85	85	85	85	85	85
Number of airports	35	35	35	35	35	35	35	35	35	35
Number of itineraries	1943	1943	1943	1943	3959	1872	1872	1872	1872	3773
Number of airports affected with \reduced capacity	0	0	0	2	35	0	0	0	2	35
Number of delayed and cancelled flights	63	107	83	41	0	63	107	83	41	0
Recovery window period of time	7/Jan 12:00 8/Jan 04:00	7/Jan 16:00 8/Jan 04:00	7/Jan 14:00 8/Jan 04:00	7/Jan 10:00 8/Jan 04:00	7/Jan 00:00 9/Jan 04:00	7/Jan 12:00 8/Jan 04:00	7/Jan 16:00 8/Jan 04:00	7/Jan 14:00 8/Jan 04:00	7/Jan 10:00 8/Jan 04:00	7/Jan 00:00 9/Jan 04:00

Table 3 - "B" test instances description - Source: Palpant et al., 2009

Instances	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10
Number of flights to be analyzed per day of the recovery window	1422	1422	1422	1422	1422	1422	1422	1422	1422	1422
Number of aircraft	255	255	255	255	255	255	255	255	255	255
Number of airports	44	44	44	44	44	44	44	44	44	44
Number of itineraries	11214	11214	11214	11214	11214	11565	11565	11565	11565	11565
Number of airports affected with reduced capacity	0	0	0	1	33	0	0	1	0	33
Number of delayed and cancelled flights	229	254	228	229	0	229	254	228	229	0
Recovery window period of time	1/Mar 16:00 3/Mar 04:00	1/Mar 16:00 3/Mar 04:00	1/Mar 16:00 3/Mar 04:00	1/Mar 16:00 3/Mar 04:00	1/Mar 00:00 3/Mar 04:00	1/Mar 16:00 3/Mar 04:00	1/Mar 16:00 3/Mar 04:00	1/Mar 16:00 3/Mar 04:00	1/Mar 16:00 3/Mar 04:00	1/Mar 00:00 3/Mar 04:00

Figure 2.8: Size of datasets used in this work.

2.6.2 Computational Goal

Most airline operations controllers demand operational disruption models to provide good solutions at the fleet level in one or two minutes [57].

In the ROADEF challenge, [8] developed a large neighborhood search heuristic to solve the Aircraft Recovery Problem (ARP) while including passenger itineraries. Solutions were found within 10 min time limit, and the model won the ROADEF 2009 Challenge. his model was improved in [70, 71]. The focus of these papers was on improving the solution quality, assuming fixed running times of five or ten minutes. A heuristic method was also developed by [62] as part of the ROADEF 2009 Challenge. The authors proposed a three-phase heuristic to solve the schedule, aircraft and passenger rescheduling problems. By comparing their results with the results previously

published, the authors showed that their heuristic method was capable of obtaining the best-found solution on more than half of the large instances, while not taking more than four minutes.

As an example, [57] worked on the Aircraft Recovery problem, providing good solutions within one minute. The size of the problem was a fleet of over 100 aircraft, serving 70 destinations on 600 flights.

In this work, the recovery algorithms were implemented in Python using CPLEX 12.7 to solve the MIP problem. Computational tests were conducted on a computer using a 64 bit Intel $i5-3470$ 3.20 GHz processor and 8 GB RAM.

2.6.3 Performance of the MIP

We subsequently relaxed some variables using some intelligent approximations taking capacity constraints into account and used CPLEX to compute the performance. In the size of the dataset we were working with, this was a few thousand variables and a few hundred thousand constraints. we also created a bijective mapping algorithm from the variables y_{ijmk} to assigning passengers to flights and cabin classes. There are also post-optimization ideas that helped in lowering the itinerary costs even further- e.g. using non-disrupted itineraries (notice our formulation only used disrupted itineraries) to intentionally delay non-disrupted passengers to make way for cleverly chosen disrupted passengers. The holy grail will be to integrate all three problems together and avoid the pitfalls of sequential solutions.

We tried batch sizes $5, 10, 25, 100$ and $\beta = 10, 25$ worked fast (the problem with $\beta = 100$ was that we needed to store compare labels of 100 graphs at the same time, so it seemed to be slower). We got a running time of around $3 - 4$ minutes on the above dataset.

2.7 Conclusions and Future Work