Robust Tail Assignment

Incorporating Delay Predictions into a Tail Assignment Model to Decrease Flight Operation Costs

Joep Bom

Delft University of Technology

TU Delft

Robust Tail Assignment

Incorporating Delay Predictions into a Tail Assignment Model to Decrease Flight Operation Costs

by

Joep Bom

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday, October 7, 2022 at 10:00 AM.

Student number:	4311116	
Project duration:	January 11, 2021 – October 7, 2022	
Thesis committee:	Dr. K.S. Postek,	TU Delft, supervisor
	Dr. C. Kraaikamp,	TU Delft
	Dr. K. Kumar,	KLM

Cover:	KLM Cargo Boeing 747-400F, ERF PH-CKC by Maarten-sr
Style:	TU Delft Report Style, with modifications by Daan Zwaneveld

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

Dear reader,

This master thesis marks the end of my journey as a student of Applied Mathematics at the Delft University of Technology. It has been a great journey, where I learned a lot about the world of science, and about myself. The process of writing this thesis was not always easy, partly because I had to work from home due to the pandemic. The problem provided to me by KLM Royal Dutch Airlines for this thesis was quite challenging. I really enjoyed coming up with ideas and directions for solving the problem, trying several model approaches, and iteratively improving the model formulation.

I would like to thank my academic supervisor Krzysztof Postek for his support and feedback during this thesis. He always had a positive view on my work, which kept me excited, even in more difficult times. The meetings we had were very constructive and supportive, making the process of writing the thesis much easier.

On the KLM side, I would like to thank my supervisor Kunal Kumar for his support. He helped me with all KLM related parts of the thesis, helping me understand the problem and terminology, as well as providing data and a framework to use for experimenting. Throughout the time I was employed at KLM, we had weekly meetings and even won an internal hackathon together! I would also like to thank the ODS department of KLM for inviting me in the online stand-up and stand-down meetings, and including me in the team as much as possible.

Joep Bom
Delft, October 2022

Summary

In this thesis a novel model is proposed to solve the Robust Tail Assignment problem. The Robust Tail Assignment problem aims to assign aircraft to flights, while minimize expected costs of operating a flight schedule, including expected delay costs. This problem is difficult, because delays can propagate between successive flights in the schedule, creating dependencies between flights assigned to the same aircraft.

Using probability distributions of delay for every individual flight, as well as expected costs associated with delaying flights, the expected delay costs of a full flight schedule can be estimated. The workings of a simulator are described, which can be used to evaluate the total expected costs of solution schedules for the Robust Tail Assignment problem.

To be able to incorporate expected delay costs in a mathematical model, the construction of a multi-commodity flow network is described, which uses departure and arrival states for flight rotations, corresponding to discrete amounts of delay. The amount of flow through edges of this network represents the probability of these states transitioning into other states. By activating and deactivating edges, based on the assignment of aircraft to rotations, this network can be used in a model to approximate the total expected delay costs of a model solution.

The proposed robust flow model uses such a state network in a MIP model, that can be solved using an iterative solver to find good solutions to the Robust Tail Assignment problem. Delay costs are imposed on edges in the network, to quantify the expected delay costs. In the model, the network size is reduced by only considering connections between rotations that have high probabilities of propagating delay. This reduces the accuracy of the model, but shortens the run-time of the optimization process significantly.

Several experiments are done to test the run-time and performance of the robust flow model. The model proved hard to solve to optimality, but is able to find good solutions, if the model parameters are well tuned. Recommendations are given for using the model, as well as future research directions.

Contents

1 Introduction

1.1. Background

Air travel is a popular means of transport, that has been steadily growing in recent decades, doubling in use between 2006 and 2019, transporting over 4.3 billion passengers in 2019 [3]. Air travel is provided by airlines, big companies that sell aircraft tickets and provide cargo transport for revenue. These airlines take care of all operations necessary for air travel, such as scheduling, airport handling, IT services, aircraft maintenance or catering. The size of these operations is substantial. In 2016, 65.5 million jobs were supported by air transport, causing $2.7 trillion of economic activity, amounting to 3.6% of the global GDP [15].

Since air travel is such a large business, there is much to gain in terms of reducing costs and work by optimizing airline operations. This thesis, assigned by KLM, will focus on optimizing a part of the airline planning process, to reduce the costs of operating flights. KLM Royal Dutch Airlines is the flag carrier airline of the Netherlands and the oldest airline in the world, operating flights to 145 destinations.

1.1.1. Flight planning process

This section will contain an overview of the flight planning process, from long term decisions to short term decisions. This process is a generalisation of the processes of all airlines, meaning the specifics can differ per airline.

Figure 1.1 shows the decision-making steps of the airline planning process. The steps are divided into four categories, and four time horizon intervals.

The outer layer represents 12 to 3 months before the day of operation, where long-term decisions are made, such as the fleet composition and route planning. Fleet composition deals with the purchase of new aircraft and maintenance or retirement of existing aircraft. Route planning deals with creating the initial flight schedule, to satisfy the expected demand. This schedule needs to be implementable with the current fleet. KLM and many other big airlines use a hub-and-spoke system for their flights. In this system the airline has a central hub, that all flights are connected to. For KLM this hub is Amsterdam Airport Schiphol, meaning all KLM flights fly either from Schiphol, or towards Schiphol, see Figure 1.2. In Figure 1.3 you will find an example of a flight schedule, where the blocks correspond to an outgoing flights followed by a return flight. After creating the initial schedule, ticket sale will start.

The inner layers represent the time period from 3 months to the day of operations, where the short-term decisions are made. At this time, more information is available about the required maintenance of the aircraft, as well as an expectation of the final amount of bookings for every flight. Using this information, an assignment of aircraft to flights in created, with the goal to create an executable schedule, while minimizing the expected operational costs. In the time period of 2 weeks to 1 day before day of operations, this schedule will be re-assessed to account for the most recent changes and information, to further reduce the expected operational costs of the schedule.

The middle circle represents the day of operations, where disruptions and last-minute changes are handled, and flights are delayed if necessary.

Figure 1.1: Integrated decision-making across organisational silos and changing planning horizons.[1]

1.1.2. Robust Tail Assignment

The focus of this thesis in the Robust Tail Assignment problem. The Tail Assignment problem is the problem that is often solved in the time period of 2 weeks to 1 day before the day of operations, denoted by “Tail assign" and “Tail swap" in Figure 1.1. The word *tail* is often used in airline literature, representing an individual aircraft.

At this time, an initial assignment of aircraft to flights is already created, and the bookings of every flight are mostly known. This new information means the schedule can be re-optimized. If a flight is expected to be under-booked, this flight could be assigned a smaller aircraft (different subtype), saving fuel and operational costs. This process is called down-gauging. Note that this is only possible if a smaller aircraft is available, or if another flight is up-gauged to a larger aircraft. Up-gauging a flight is likely to cause extra costs in fuel and operational costs, so these aircraft “swaps" are only implemented if the net gain is positive.

Besides this, 3 days to 1 day before the day of operations, some airlines have information regarding expected delays of flights, which depend on factors such as the destination, time and weather forecast. Using this information, a flight that has a high likelihood to be delayed can be scheduled in such a way, that the assigned aircraft has longer buffer times between this flight and the next scheduled flight. This way, the next flight is less likely to be delayed as a result of the aircraft arriving late. Taking this factor of delay propagation and its related costs into account, the tail assignment can again be re-optimized. This problem is called the Robust Tail Assignment problem.

So, in general, the goal of the Robust Tail Assignment problem is to find an assignment of aircraft to all flights, such that the final schedule is executable, while minimizing the total expected costs. These expected costs consist of many different terms, such as the expected fuel costs, landing fees, parking charges and expected delay costs. It is usually preferred to keep the assignments as close as possible to the original assignments, as to not change the schedule too much soon before the day of operations. For this purpose, a reassignment penalty can be used.

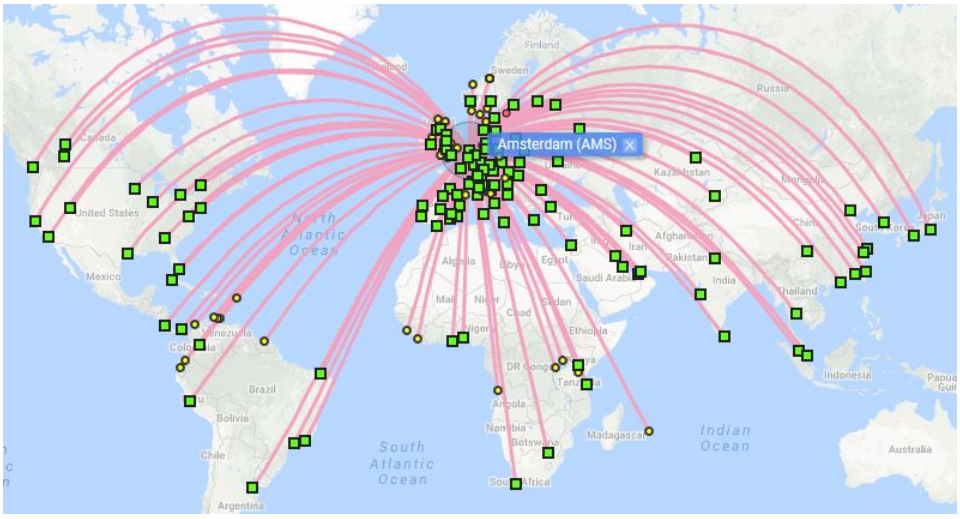

Figure 1.2: Map of KLM flights, a hub-and-spoke system.

Figure 1.3: Example flight schedule over three days by KLM. The dates are randomized.

1.2. Complexity

The Robust Tail Assignment problem aims to find a feasible assignment of aircraft to all flights with the lowest expected costs. A feasible flight assignment means the resulting flight schedule is executable. All relevant costs of a flight are only dependent on the assigned aircraft and the corresponding flight route. Note that by propagation of delay from one flight to the next, the expected delay of costs of a flight in a flight route is dependent on all previous flights in that route. Because of this dependency, the expected delay costs of a flight cannot be calculated locally, i.e. by only considering the predecessor of every flight.

One thing to note is that the costs of the flights in a specific aircraft's flight route are independent of other aircraft's flight routes. Using this observation and the fact that every flight has to be assigned precisely one aircraft, the problem reduces to a set partitioning problem, where the sets are possible flight routes and the elements of the sets are flights.

1.2.1. Set partitioning problem

Let us define the problem as a set partitioning problem. Define A to be the set of available aircraft, and F to be the set of all flights in the schedule. Define $R_a \subset F$ to be the set of feasible flight routes for an aircraft $a \in A$. Let c_{ra} be the expected cost of operating flight route $r \in R_a$ on aircraft a. Lastly, define

b_{fr} to equal 1 if flight $f \in F$ is included in route $r \in R_a$, and 0 if not. Now we can formulate the problem:

$$\underset{x_{ra}}{\text{minimize}} \quad \sum_{a \in A} \sum_{r \in R_a} c_{ra} x_{ra} \tag{1.1}$$

$$\text{subject to} \quad \sum_{a \in \mathcal{A}_r} \sum_{r \in R_a} b_{fr} x_{ra} = 1 \qquad \forall f \in F \tag{1.2}$$

$$\sum_{r \in R_a} x_{ra} = 1 \qquad \forall a \in A \tag{1.3}$$

$$x_{ra} \in \{0, 1\} \qquad \forall a \in A, r \in R_a \tag{1.4}$$

In this formulation, the variables x_{ra} equal 1 if the flight route r is assigned to aircraft a. The objective function, Equation (1.1), calculates the sum of the expected costs of all selected flight routes. Constraint (1.2) ensures every flight is assigned to exactly one aircraft. Constraint (1.3) ensures that an aircraft operates exactly one flight route.

This formulation is a constrained set partition problem, where all flights have to be included in exactly one set, and the sets are chosen from the possible sets in $\bigcup_{a \in A} R_a$. The problem is constrained, since exactly one set from every R_a has to be chosen.

The set partitioning problem is known to be NP-hard [13]. Also, note that the sets R_a are very big, since they contain all possible feasible flight routes for an aircraft a. If we only consider one specific flight route in a set R_a containing k flights, all subsets of this flight route are also feasible flight routes, meaning there are 2^k such flight routes in R_a. Thus, the sets R_a are exponentially big, making the Robust Tail Assignment problem very difficult.

1.3. Previous work

The Tail Assignment problem is a well studied problem. Grönkvist [14] contains an overview of the Tail Assignment problem and approaches for solving the problem. In the literature many different problem formulations related to tail assignment have been studied, as the process of airline planning differs among airlines. Airlines with regular schedules and large fleets of aircraft can often decide on tail assignment closer to the day of operations, in which case more exact data and expected costs of operation are available.

Flight delays are a large cost factor for airlines. Eurocontrol [11] estimates the cost of delaying a flight to be around 59 to 85 euros per minute. Solving the Tail Assignment problem while minimizing the operational costs, as well as delay costs is called the Robust Tail assignment problem.

In the literature, the Tail Assignment and Robust Tail Assignment are often solved separately. First aircraft are assigned to flights, to minimize operational costs, without accounting for delay costs. The assignments are then reoptimized for robustness, while not allowing aircraft of different subtype from before to be assigned to flights. Since the operational costs (e.g. fuel costs, landing fees, parking charges) are very similar for aircraft of the same subtype, this optimization for robustness often does not consider costs in its objective, but rather a notion of the amount of delay propagation.

To solve the non-robust Tail Assignment problem, two main formulation types are used. Firstly, there is the set partitioning formulation, as described in Section 1.2.1, which is often solved using Lagrangian relataxion (e.g. Clarke et al. [8]), or a column generating algorithm (e.g. Kabbani et al. [17]). The second formulation is a multi-commodity flow network, used by Feo et al. [12]. This formulation defines a network with flights as nodes and feasible connections between flights as arcs. The different commodities correspond to different aircraft. Some papers, such as Desaulniers et al. [9] even propose an algorithm that uses both formulations, to obtain optimal branching strategies for the column generating method.

Since this flow formulation considers local connections between flight, it is not trivial to incorporate robustness into such a model, because of the dependency throughout the whole flight route.

Approaches to solve the Robust Tail Assignment problem can be split into three main categories that we shortly address here.

1.3.1. Key performance indicators (KPI)

A simple way to introduce a sense of robustness into the Tail Assignment problem is to change or replace the objective of the problem using terms that measure robustness in a heuristic way. These terms are called "key performance indicators". For example, one could optimize for having an even distribution of buffer time between flights. Examples of KPI approaches for solving the Robust Tail Assignment problem can be found in Burke et al. [7], Reoenberger et al. [18] and Ahmadbeygi et al. [2]. These appoaches respectively use swap opportunities, cancellation cycles, and locally propagated delays. Note that these approaches do not optimize directly for the expected total costs, but rather for measures that decrease the expected total costs heuristically. These approaches are often applied in situations when not much information about expected delays is available.

1.3.2. Robust optimization

The field of robust optimization considers problems where some input parameters are not fixed, but are uncertain. Soyster [19] first introduced robustness to linear programming. The idea is that a solution needs to be feasible for all possible values of its input parameters, i.e. the worst case values. This approach is highly conservative, and not applicable to the Robust Tail Assignment problem because of its many possibilities of delay. Later work, such as Ben-Tal & Nemirovski [4] and Bert-simas & Sim [5] propose less conservative approaches. For example, Bert-simas & Sim [5] only allow a limited amount of parameters for which to account uncertainty to consider a solution feasible. But, because of the conservative nature of such worst case concepts, these approaches are expected to high costs and have not yet been applied to the Robust Tail Assignment problem.

1.3.3. Stochastic optimization

In stochastic optimization, the uncertainty of the input parameters of the problem is considered by regarding these parameters as random variables. As an objective, the expectation of some objective function is minimized. For the Robust Tail Assignment problem, a stochastic model can be used to consider possible delays and delay propagation. Borndörfer [6] and Dovica [10] use the sum of the probability of delay propagation for all flights as the objective to minimize. The problem is formulated as a set partitioning problem, and a column generation algorithm is used to find the optimal solution to the linear relaxation of the problem. The solution to this linear relaxation is then rounded to an integer solution.

1.4. Thesis outline

In this thesis we will propose a model to solve the Robust Tail Assignment problem. In Chapter 2 the problem will be mathematically formulated, a way to evaluate a solution of the problem will be given, and both a non-robust and a benchmark robust model provided by KLM will be described.

Chapter 3 will mathematically describe delay propagation through a flight schedule, explain its difficulties and give a way to calculate the expected delay costs of a flight route.

In Chapter 4, a Robust Flow Model will be described and explained, which can be used to solve the Robust Tail Assignment problem.

In Chapter 5 the experiments performed on the model will be described and justified, after which the results of the experiments will be given and analysed.

Finally, Chapter 6 will contain the conclusion of this thesis, and give directions the usage of the model, as well as for further research.

2

Robust Tail Assignment Problem

2.1. Problem description

The Robust Tail Assignment problem deals with the assignment of aircraft (or tails) to scheduled flights, shortly before the day of operation. The goal is to construct a feasible flight route for every aircraft, such that every flight is operated by some aircraft, while considering business rules (such as restrictions on aircraft size at certain airports), necessary maintenance, turn-around times at the airport and minimizing the total expected costs. This problem is solved every day, to adjust the previously constructed routes based on the most recent information about expected disruptions and changes in the schedule.

This thesis will focus on flight schedules that use a hub-and-spoke system, meaning one aircraft has to both operate the flight to another airport and the flight back from this airport. The central airport is called the *hub* station, while the other airports are called *outstations*. When using a hub-and-spoke system, the scheduled flights can be combined into *rotations*. A rotation consists of a journey of two flights, where the first flight departs from the hub station and the second flight arrives back at the hub station [1]. The schedule also contains maintenance blocks, which we will consider to be rotations, consisting of one "flight". Maintenance blocks are bound to a certain aircraft, that requires some maintenance at a given time slot.

Using this description of rotations, the problem reduces to an aircraft-rotation assignment problem.

2.1.1. Business rules

Business rules are instructions or constraints on the business activities. These rules must be satisfied in the final flight schedule and assignments of the aircraft to flights.

Several business rules apply to the assignment of aircraft to flights. For example, some outstations are unable to operate a certain subtype of aircraft. This means that rotations containing flights to and from this outstation are not allowed to be assigned to an aircraft of this subtype.

These assignment business rules affect the input of the problem and have a restricting effect, meaning they reduce the solution space. For the sake of this thesis we will omit these business rules, since they vary greatly between airlines, often change over time, and have little impact on the difficulty of the problem. So, every aircraft-rotation combination is considered to be valid.

A business rule that will be incorporated into this thesis is the use of reserve aircraft. If an aircraft arrives much later than scheduled, causing the next flight to be delayed by more than 120 minutes, a reserve aircraft will be used for the next flight. This rule has some exceptions, as there is not an unlimited supply of reserve aircraft, and other business rules may prevent a reserve aircraft to be used. For the purpose of this thesis, this notion is simplified. We will assume an unlimited supply of reserve aircraft, and set a fixed cost to the use of a reserve aircraft.

[1] Note that some airlines also schedule rotations consisting of more than two flights. For the purpose of this thesis we shall restrict ourselves to rotations of two flights. All methods and theory studied in this thesis can be modified to also work for rotations with more than two flights.

2.1.2. Costs

The costs dependent on the assignment of aircraft to rotations considered for the Robust Tail Assignment problem are the following:

Assignment costs

Assigning an aircraft to a rotation has many associated costs. There are operational costs, which consist of many types of fees, that are dependent on the outstation and the subtype of the aircraft. These include airport landing fees, taxes, parking charges at outstation and night stop fees (including hotel costs for crew).

The fuel cost of a rotation is dependent on the aircraft, since some aircraft are more fuel efficient than others. The average fuel costs per minute for every aircraft are given as input, so the total fuel price for every rotation-aircraft pair can be estimated, using the average flight duration of the flights in the rotation.

Using the information on bookings for a flight, an estimation can be made for the costs of swapping to an aircraft of a different subtype. For example, if not enough business class seats are available, but the aircraft has free economy class seats, a cost will be imposed. This will imply a denied boarding cost on the corresponding rotation-aircraft combination. This cost consists of refunds and future value loss. Future value loss is an estimation of the missed revenue of passengers that may be less likely to choose the same airline for their next journeys due to the negative experience.

The assignment cost for every aircraft-rotation combination consists of these costs, which can all be independently calculated.

Delay costs

If a flight arrives later than scheduled, there are several fees that have to be paid, depending on the flight and the amount of delay. For example, a delay may cause refunds for passengers that miss their connecting flight and passengers can request EU claims for delayed flights. Also, future value loss needs to be considered again. All these factors induce a type of expected costs or reduced future income. Based on the flight and the amount of delay minutes, an estimation can be made for the delay costs.

2.2. Robustness

Planned schedules are never exactly executable in real life. Due to unforeseeable events, every scheduled rotation has a possibility of taking longer than planned. If two tightly connected rotations are scheduled to be flown on the same aircraft, longer flight times of the first rotation can cause delays to the second rotation, which might in turn cause delays for the next scheduled rotation. To reduce the total amount of disruption in the real life flight schedule, we need to account for this propagation.

Dealing with such situations can be classified in two approaches: reactive or proactive. Reactive approaches require you to find a good solution close to the original schedule quickly, when disruptions occur. In the case of a flight schedule, these disruptions are often only identified at the moment they occur. The decision of aircraft assignment to rotations are decisions that need to be made in advance, because of the size and dependencies of the operations required to fly aircraft. Therefore, it is not preferred to change the assignments at the last moment, meaning a solely reactive approach is not ideal.

Proactive approaches try to find a schedule that can handle disruptions better, in advance. To accomplish this, the schedule needs to be *robust*, meaning that the impact of delays on the schedule is as small as possible. To achieve this robustness, the schedule needs to have buffer time between rotations that are more likely to be delayed, or cost a lot to be delayed. To be able to quantify the robustness of a schedule, weather forecasts and data about the previous delays of certain flights can be used to create expected delay distributions for all flights[2]. These delay distributions can then be used to find an expectation of the delay costs for all flights, based on the connected rotations in the schedule, the buffer time between these rotations, and the delay distributions of the flights.

Calculating the expected total delay cost of a schedule based on these delay predictions is not a trivial task. Section 3.1 will explain exactly why this is the case.

[2]The process of predicting delay distributions for a flight is not covered in this thesis. This prediction can be achieved through Machine Learning algorithms, see e.g. [20]. For the purpose of this thesis, we assume independent delay distributions are available for all flights in the schedule.

2.3. Problem Formulation

In this section some mathematical notation will be introduced and the input data, solution space and objective function will be described.

2.3.1. Input data

The input of the problem consists of:

i) **Decision scope**: typically the three consecutive days before the day of departure.

ii) **Schedule**: A schedule of all rotations that are to arrive or depart within the decision scope. The rotations in the schedule have an initially assigned aircraft. The schedule contains information about the rotations, such as the scheduled departure time, scheduled arrival time and probability distribution of expected delay of its flights.

iii) **Aircraft**: A list of all available aircraft, containing information about the aircraft, such as subtype, fuel usage, required turnaround time at all stations, etc.)

iv) **Operational restrictions**: A mapping of stations to compatible aircraft subtypes, along with restrictions active on aircraft.

v) **Delay predictions**: For every flight, a probability distribution of the amount of delay is given as a random variable. These random variables are considered to be independent.

vi) **Costs**: Assignment costs for each pairing of destination and aircraft. Also, for every flight, the expected costs associated with arrival delay of the flight are given.

2.3.2. Solution space

A feasible solution is an assignment of aircraft to rotations, such that:

- Every rotation is assigned one aircraft, which is compatible according to the business rules of the rotation.
- If no flights are delayed, there is enough turnaround time between rotations assigned to the same aircraft, i.e. every aircraft is assigned a feasible flight route if no flight is disrupted.

2.3.3. Objective

The goal of this optimization tool is to minimize the sum of the total expected costs of all flights in the time scope.

2.3.4. Mathematical notation

Sets

T	Set of time-values in scope
$\mathcal{R}$	Set of rotations scheduled to arrive or depart within T, indexed by r
$\mathcal{F}$	Set of flights and maintenance blocks, scheduled to arrive or depart within T, indexed by f
$\mathcal{F}_r$	Set of flights and maintenance blocks in rotation $r \in \mathcal{R}$, scheduled to arrive or depart within T, indexed by f
$\mathcal{S}$	Set of all aircraft subtypes available within T, indexed by s
$\mathcal{S}_r$	Set of all aircraft subtypes that are allowed to operate rotation $r \in \mathcal{R}$
$\mathcal{A}$	Set of all aircraft available within T, indexed by a
$\mathcal{A}_s$	Set of all aircraft of subtype $s \in \mathcal{S}$ available within T, indexed by a
$\mathcal{A}_r$	Set of all aircraft $a \in \mathcal{A}$ allowed to operate rotation $r \in \mathcal{R}$
$\mathcal{D}_f$	Random variable, containing the probability distribution of delay of a flight $f \in \mathcal{F}$

Parameters

std_f	Scheduled departure time of a flight $f \in \mathcal{F}$
sta_f	Scheduled arrival time of a flight $f \in \mathcal{F}$

std_r	Scheduled departure time of the first flight of a rotation $r \in \mathcal{R}$
sta_r	Scheduled arrival time of the last flight of a rotation $r \in \mathcal{R}$
a_r^{init}	Initially assigned aircraft ($\in \mathcal{A}$) for rotation $r \in \mathcal{R}$
s_a	Subtype ($\in \mathcal{S}$) of aircraft $a \in \mathcal{A}$
δ_s^0	Minimum ground time needed between flights at the hub station for aircraft of subtype $s \in \mathcal{S}$
δ_s^r	Minimum ground time needed between the two flights of rotation r at the outstation for aircraft of subtype $s \in \mathcal{S}$

Costs

c_{ra}^{assign}	Expected assignment costs of assigning a rotation $r \in \mathcal{R}$ to an aircraft $a \in \mathcal{A}$
$c_f^{delay}(t)$	Function mapping the amount of arrival delay of a flight $f \in \mathcal{F}$ in minutes to the expected delay costs
$c^{reserve}$	Expected cost of using reserve aircraft

Example 1.
Let us consider a small example of the Robust Tail Assignment problem. This example will be used to elaborate the workings of the models explained in this thesis.

We consider a schedule containing seven rotations, to be scheduled using a fleet of three aircraft with two subtypes. The rotations contain thirteen flights in total, where one flight is a maintenance block. We have the following information about the schedule:

$$
\begin{aligned}
T &:= [0, 500] \\
\mathcal{R} &:= \{r_1, r_2, r_3, r_4, r_5, r_6, r_7\} \\
\mathcal{F} &:= \{f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{13}\} \\
\mathcal{A} &:= \{a_1, a_2, a_3\} \\
\mathcal{S} &:= \{s_1, s_2\}
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{F}_{r_1} &= \{f_1, f_2\} & \mathcal{F}_{r_5} &= \{f_8, f_9\} \\
\mathcal{F}_{r_2} &= \{f_3, f_4\} & \mathcal{F}_{r_6} &= \{f_{10}, f_{11}\} \\
\mathcal{F}_{r_3} &= \{f_5, f_6\} & \mathcal{F}_{r_7} &= \{f_{12}, f_{13}\} \\
\mathcal{F}_{r_4} &= \{f_7\} \\
\mathcal{A}_{s_1} &= \{a_1, a_2\} \\
\mathcal{A}_{s_2} &= \{a_3\} \\
\mathcal{A}_{r_4} &= \{a_2\} \\
\mathcal{A}_r &= \mathcal{A} \qquad \forall r \in \mathcal{R} \backslash \{r_4\} \\
\mathcal{A}_{r_4} &= \{a_2\}
\end{aligned}
$$

f_1	std	0
	sta	45
	maint	False
	$\mathcal{D}_{f_1}$	Gamma(6, 5) - 30
f_2	std	80
	sta	125
	maint	False
	$\mathcal{D}_{f_2}$	Gamma(5, 6) - 30
f_3	std	5
	sta	70
	maint	False
	$\mathcal{D}_{f_3}$	Gamma(9, 8) - 55
f_4	std	110
	sta	175
	maint	False
	$\mathcal{D}_{f_4}$	Gamma(8, 9) - 55
f_5	std	10
	sta	105
	maint	False
	$\mathcal{D}_{f_5}$	Gamma(9, 7) - 70
f_6	std	145
	sta	240
	maint	False
	$\mathcal{D}_{f_5}$	Gamma(11, 6) - 70
f_7	std	205
	sta	250
	maint	a_2
	$\mathcal{D}_{f_5}$	0
f_8	std	210
	sta	295
	maint	False
	$\mathcal{D}_{f_8}$	Gamma(7, 8) - 55
f_9	std	330
	sta	415
	maint	False
	$\mathcal{D}_{f_9}$	Gamma(9, 6) - 55
f_{10}	std	290
	sta	365
	maint	False
	$\mathcal{D}_{f_{10}}$	Gamma(6, 7) - 50
f_{11}	std	405
	sta	480
	maint	False
	$\mathcal{D}_{f_{11}}$	Gamma(5, 8) - 50
f_{12}	std	350
	sta	405
	maint	False
	$\mathcal{D}_{f_{12}}$	Gamma(8, 4) - 35
f_{13}	std	440
	sta	495
	maint	False
	$\mathcal{D}_{f_{13}}$	Gamma(6, 5) - 35

Table 2.2: Flight information of example problem. The std and sta values represent the amount of minutes after the start of the schedule. Flight f_7 is a maintenance block for aircraft a_2, which has no delay predictions. All other flights have delay predictions in the form Gamma(k, θ) + s, representing a shifted Gamma distribution, with shape parameter k, scale parameter θ, shifted by s minutes.

δ_s^r	s_1	s_2
0	25	30
r_1	30	35
r_2	35	40
r_3	35	40
r_5	30	35
r_6	35	40
r_7	30	30

Table 2.3: Turnaround times in minutes of aircraft subtypes at the various stations in the schedule of the example problem. The value $r = 0$ represents the hub-station, while the r_i values represent the outstations on the corresponding rotations.

c_{ra}^{assign}	a_1	a_2	a_3
r_1	1210	1290	1330
r_2	2150	2110	2240
r_3	3410	3420	3510
r_5	2930	2900	3120
r_6	2350	2270	2550
r_7	1990	2090	2120

flight	$c_f^{delay}(d)$
f_1	$20 * d$
f_2	$18 * d$
f_3	$10 * d$
f_4	$12 * d$
f_5	$15 * d$
f_6	$13 * d$
f_7	$5 * d$
f_8	$8 * d$
f_9	$9 * d$
f_{10}	$21 * d$
f_{11}	$20 * d$
f_{12}	$19 * d$
f_{13}	$18 * d$

$c^{reserve}$
2500

Table 2.4: Costs in euros for example problem. The delay costs are linear functions of the amount of delay in minutes d.

In the example, shifted Gamma distributions are picked as probability distributions of delay, since these distributions have a minimum, but no maximum. This makes sense in practice, since flights have a physical limitation on the amount of minutes they can arrive early (e.g. minimum flight time), but no limitation on the amount of extra delay that may be realised due to unforeseen events.

Gamma distributions have a shape parameter k and a scale parameter θ, a mean equal to $k\theta$ and variance equal to $k\theta^2$, and start at 0. So, for example, the shifted gamma distribution used for f_1 starts at -30 minutes, has mean $\mathbb{E}[\mathcal{D}_{f_1}] = 5 \cdot 5 - 30 = -5$ minutes and variance $Var[\mathcal{D}_{f_1}] = 5 \cdot 5^2 = 125$. This distribution is plotted in Figure 2.1. Note that these distributions can take negative values of delay. This means a flight has a positive probability to negate propagated delays.

Note that the cost functions of arrival delay are linear functions, depending on the amount of delay minutes.

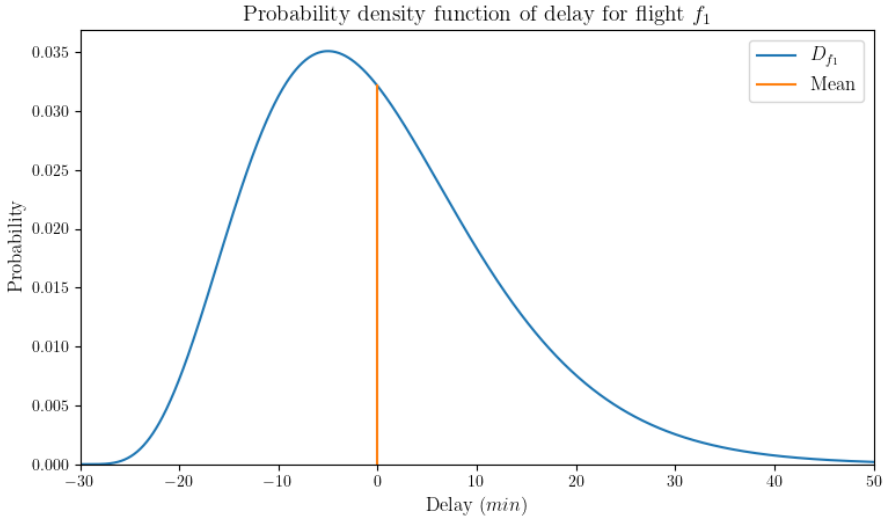

Figure 2.1: Delay prediction for flight f_1, a shifted Gamma distribution Gamma$(6, 5) - 30$, with mean $\mathbb{E}[\mathcal{D}_{f_1}] = 6 \cdot 5 - 30 = 0$ and variance $Var[\mathcal{D}_{f_1}] = 6 \cdot 5^2 = 150$.

An example of a solution to the problem can be found in Figure 2.2. In this solution rotations r_1 and r_5 are assigned to aircraft a_1, rotations r_2, r_4 and r_6 to aircraft a_2 and rotations r_3 and r_7 to aircraft a_3. Note that the flights in rotation r_6 and r_7 have higher delay costs than r_5, and the flights in rotation r_2 have a high probability of delay, compared to the other rotations. This means that operating this schedule will likely result in a high delay cost for the flights in rotation r_6.

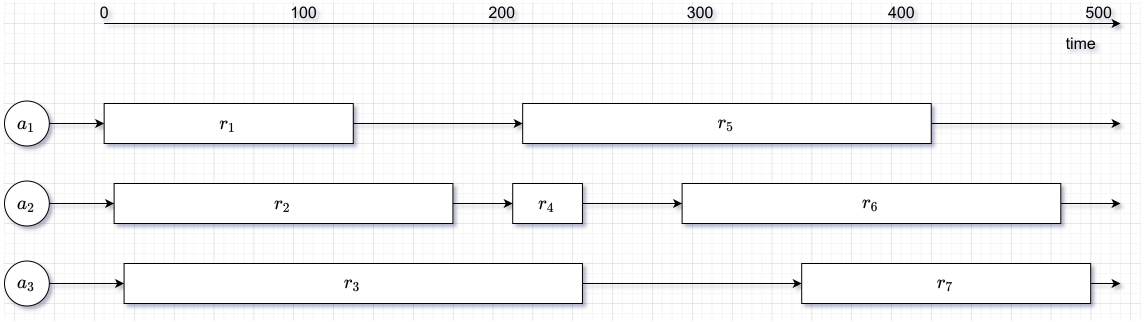

Figure 2.2: Example solution schedule for this example problem. The arrows indicate the flight path of the different aircraft. Note that the rotations are represented by blocks, where the length of the block corresponds to the operating time of the block, when no flights are delayed.

After some quick deductions, it can easily be seen that this problem has 8 feasible solutions, given in Table 2.5. The total assignment costs of these solution is also given in this table. As we can see, sol1 has the lowest assignment cost, but the question remains which schedule has the lowest total expected costs.

solution	r_1	r_2	r_3	r_4	r_5	r_6	r_7	c^{assign}
sol1	a_1	a_2	a_3	a_2	a_1	a_2	a_3	14150
sol2	a_1	a_2	a_3	a_2	a_1	a_3	a_2	14400
sol3	a_2	a_1	a_3	a_2	a_1	a_2	a_3	14270
sol4	a_2	a_1	a_3	a_2	a_1	a_3	a_2	14520
sol5	a_2	a_3	a_1	a_2	a_3	a_1	a_2	14500
sol6	a_2	a_3	a_1	a_2	a_3	a_2	a_1	14320
sol7	a_3	a_2	a_1	a_2	a_3	a_1	a_2	14410
sol8	a_3	a_2	a_1	a_2	a_3	a_2	a_1	14230

Table 2.5: All feasible solutions to the example problem, and their total expected assignment costs.

2.4. Evaluation

To compare the quality of different models of the Robust Tail Assignment problem, we need to be able to evaluate the model solutions. In a real life scenario, there is only one schedule that can be executed, so we cannot easily compare different solutions. The only way to compare solutions is by their expected costs, as given in the problem input.

Given a solution schedule, most of the expected costs can easily be calculated. The assignment costs, consisting of operational costs, fuel costs, and swap costs are solely based on the assignment of aircraft to rotations in the solution. But, the delay costs are more difficult to calculate, since we need to consider random variables. Moreover, because a delayed flight can cause the next flight operated by the same aircraft to also be delayed, the expected amount of delay costs of a flight is dependent on the delay of previous flights. In Section 3.1 this notion will be described in more detail.

To avoid the need to calculate the distributions of (propagated) delay for every flight, a simulation approach can be used. This section will explain the workings of a simple simulation engine, that simulates the flight schedule many times, using random draws from the delay distributions. This simulation engine can be used to evaluate and compare different solutions of the Robust Tail Assignment Problem. The next section will describe the workings of the simulation engine.

2.4.1. Simulation Engine

We describe the workings of a simulation engine, that estimates the delay costs of a schedule using simulations. In one simulation, for every flight a delay value is drawn from the given distribution of delay. In this way, by performing many simulations, the delay distributions for every flight are approximated. In every simulation we use these delay values to calculate the delay costs of every flight. In this calculation propagation of delay is considered. The simulations are performed in the following way:

Every fleetline can be simulated separately by following the timeline, using a time variable. Initially, this time variable is set to the departing time of the first flight in the fleetline. The flights in the fleetline are evaluated in order, starting them at the maximum of the time variable and the scheduled departure time of the flight. If the departure delay is more than 120 minutes, a reserve aircraft is utilised, and the flight will depart on the scheduled time. Then, the time variable is updated by adding the duration of the flight and the drawn amount of delay of the flight, after which the resulting arrival delay of the flight is saved. Finally, the turnaround time is added to the time variable, before evaluating the next flight in the fleetline. The saved arrival delay values are used for estimating the delay costs of the solution schedule. The pseudocode for the simulation engine is given in Algorithm 1.

Algorithm 1 Simulation engine

$simulationAmount \leftarrow$ amount of simulations to perform
$fleetlines \leftarrow$ dictionary holding the solution fleetlines
    (aircraft $\rightarrow$ sorted list of rotations)
$simDur_f \leftarrow$ list containing $simulationAmount$ simulated duration
    times of flight f (scheduled duration plus randomly drawn
    delay from distribution) $\forall f \in \mathcal{F}$
$\delta_s^f \leftarrow$ turnaround time after flight f for aircraft subtype s, $\forall f \in \mathcal{F}, s \in \mathcal{S}$
$delays \leftarrow$ dictionary holding the simulated arrival delays
    (flight $\rightarrow$ list of length $simulationAmount$)
$reserveAircraft \leftarrow$ list of zeros of length $simulationAmount$
for $a \in \mathcal{A}$ **do**
    $time \leftarrow [-\infty, -\infty, \ldots, -\infty]$ of length $simulationAmount$
    for $r \in fleetlines[a]$ **do**
        for $f \in \mathcal{F}_r$ **do**
            for $k \in 0, 1, \ldots simulationAmount - 1$ **do**
                $time[k] \leftarrow \max(time[k], std_f)$
                if $time[k] \geq std_f + 120$ and f first flight of rotation f **then**
                    $time[k] \leftarrow std_f$
                    $reserveAircraft[k] \leftarrow reserveAircraft[k] + 1$
                end if
                $delays[f][k] \leftarrow \max(time[k] + simDur_f[k] - sta_f, 0)$
                $time[k] \leftarrow time[k] + simDur_f[k] + \delta_{s_a}^f$
            end for
        end for
    end for
end for
return $delays, reserveAircraft$

Using the delay and reserve aircraft values found using Algorithm 1, the total expected delay and reserve aircraft costs can be calculated using the delay costs and reserve aircraft as defined in the problem description. We get:

$$\mathbb{E}[\text{delay costs}] = \sum_{k \in \{0,1,\ldots,simulationAmount-1\}} \sum_{f \in \mathcal{F}} \frac{c_f^{delay}(delays[f][k])}{simulationAmount} \tag{2.1}$$

$$\mathbb{E}[\text{reserve aircraft costs}] = \frac{c^{reserve}}{simulationAmount} \cdot \sum_{k \in \{0,1,\ldots,simulationAmount-1\}} reserveAircraft[k] \tag{2.2}$$

By performing many simulations, these equation approximate the total expected delay and reserve aircraft costs. Since the delay costs are not the same in every simulation, we can show the total costs of a solution using a cumulative distribution function (cdf) of the simulated costs. See Example 2 for an example of the outcome of the simulations.

Example 2.
Let us evaluate the different solutions given in Table 2.5, to the example problem defined in Example 1. Using the simulation engine to perform 10000 simulations for every solution, we get the results given in Figure 2.3 and Table 2.6.

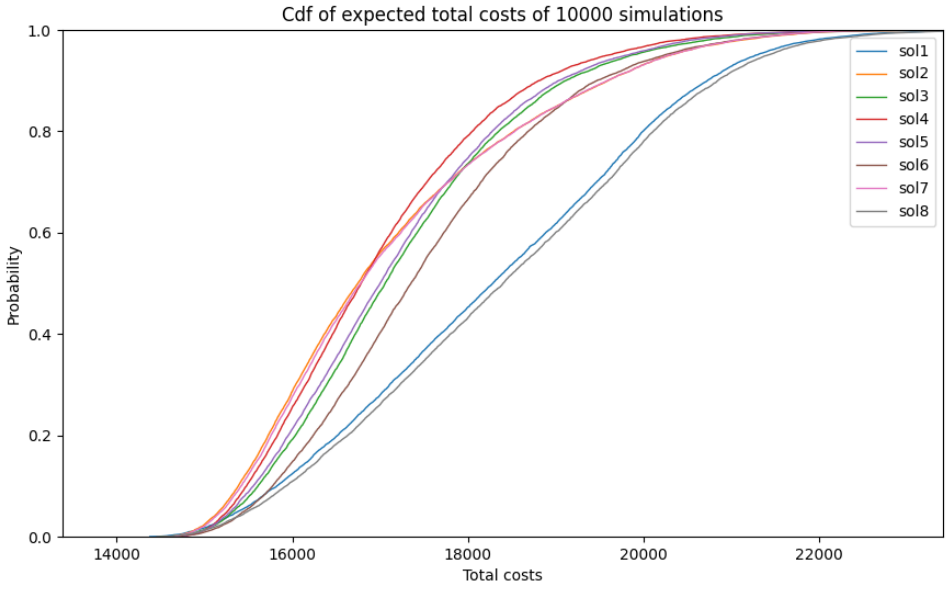

Figure 2.3: Result a cumulative distribution function of expected total costs for all 8 solutions to the example problem.

	assignment costs	delay costs	reserve aircraft costs	total costs	delay minutes
sol1	14150	3900.00	253.5	18303.50	304.3
sol8	14230	3934.17	253.5	18417.67	307.1
sol3	14270	2923.03	57.0	17250.03	255.4
sol6	14320	3116.78	84.75	17521.53	276.5
sol2	14400	2673.56	70.0	17143.56	246.5
sol7	14410	2683.36	70.0	17163.36	248.1
sol5	14500	2596.89	84.75	17181.64	250.9
sol4	14520	2427.52	57.0	17004.52	231.0

Table 2.6: Expected costs for all feasible solutions to the example problem, averaged over 10000 simulations. The solutions are sorted by the expected assignment costs.

By simulating the solutions, we found that sol4 has the lowest expected total costs, averaged over all 10000 simulations. This does not mean the solution the cheapest in every simulation. For example, if a single simulation draws negative delays for all flights, the delay costs will be zero for all flights, and the solution with the lowest assignment costs will be the cheapest. In fact, sol4 has the highest assignment costs of all solutions, so it will be the most expensive in this simulation! But, sol4 is more robust, and has a smaller probability of propagating delays, so higher delays drawn in simulations often cause smaller delay costs for sol4. In Figure 2.3, a steeper cdf corresponds to a more robust solution. Solutions sol1 and sol8 have the least robustness, which results in high expected delay costs for these solutions.

2.5. Non-Robust Tail Assignment Model

The Robust Tail Assignment problem is a stochastic problem, since we are considering random variables to calculate the expected delay costs of a solution. If we disregard the delay costs of the Robust Tail Assignment problem, this problem is reduced to the Non-Robust Tail Assignment problem, a purely

deterministic problem. This decreases the difficulty tremendously. In this section an Integer Linear Programming model formulation of the model is proposed, to illustrate the basic workings of a Tail Assignment problem.

For every allowed rotation-aircraft combination (r, a), a binary decision variable $X_{r,a}$ is created. These variables will indicate the solution, where $X_{r,a} = 1$ if aircraft a is assigned to rotation r, and $X_{r,a} = 0$ if not. Using these variables, we can express the total costs in an objective function, which we will try to minimize, as

$$\sum_{(r,a):r\in\mathcal{R},a\in\mathcal{A}_r} c_{ra}^{assign} X_{r,a}.$$

What remains is to assure feasibility of the problem. Firstly, exactly one aircraft has to be assigned to a rotation. This can be enforced through the following constraints:

$$\sum_{a\in\mathcal{A}_r} X_{r,a} = 1 \quad \forall r \in \mathcal{R}$$

Secondly, flight routes need to be feasible. This can also be achieved through constraints on the decision variables $X_{r,a}$. Let RP_s^{ovlp} denote the set of rotation pairs (r_1, r_2), where $r_1, r_2 \in \mathcal{R}$, that *overlap* if assigned to the same aircraft of subtype $s \in \mathcal{S}$. This means that either r_1 and r_2 are overlapping blocks, or that if r_1 and r_2 are assigned to such an aircraft, there is not enough turn-around time between the rotations. We can describe this set mathematically as:

$$RP_s^{ovlp} := \{(r_1, r_2) : r_1, r_2 \in \mathcal{R}, std_{r_1} \leq std_{r_2} \text{ and } sta_{r_1} + \delta_s^0 \geq std_{r_2}\} \tag{2.3}$$

The sets RP_s^{ovlp} can be used to enforce feasibility of the flight routes, through the constraints:

$$X_{r_1,a} + X_{r_2,a} \leq 1 \quad \forall (r_1, r_2) \in RP_s^{ovlp}, \forall a \in \mathcal{A}_s \cup \mathcal{A}_{r_1} \cup \mathcal{A}_{r_2}$$

With this objective function and these two sets of constraints, the model is complete. A full description of the model can be found in Equation 2.4. The model can be optimized using an iterative solver, such as Gurobi [16]. Section 4.6 will explain the workings of such an iterative solver.

$$\begin{aligned}
\underset{X_{r,a}}{\text{minimize}} \quad & \sum_{(r,a):r\in\mathcal{R},a\in\mathcal{A}_r} c_{ra}^{assign} X_{r,a} & \\
\text{subject to} \quad & \sum_{a\in\mathcal{A}_r} X_{r,a} = 1 & r \in \mathcal{R}. \\
& X_{r_1,a} + X_{r_2,a} \leq 1 & (r_1, r_2) \in RP_s^{ovlp}, a \in \mathcal{A}_s \cup \mathcal{A}_{r_1} \cup \mathcal{A}_{r_2} \\
& X_{r,a} \in \{0, 1\} & r \in \mathcal{R}, a \in \mathcal{A}_r
\end{aligned} \tag{2.4}$$

Example 3.
We build the Non-Robust Tail Assignment Model for our example problem, as defined in Example 1. We have $\mathcal{A}_{r_4} = \{a_2\}$ and $\mathcal{A}_r = \mathcal{A}$ for all $r \in \mathcal{R}\backslash\{r_4\}$. This means we have the following decision variables:

X_{r_1,a_1}	X_{r_1,a_2}	X_{r_1,a_3}
X_{r_2,a_1}	X_{r_2,a_2}	X_{r_2,a_3}
X_{r_3,a_1}	X_{r_3,a_2}	X_{r_3,a_3}
	X_{r_4,a_2}	
X_{r_5,a_1}	X_{r_5,a_2}	X_{r_5,a_3}
X_{r_6,a_1}	X_{r_6,a_2}	X_{r_6,a_3}
X_{r_7,a_1}	X_{r_7,a_2}	X_{r_7,a_3}

Using Definition (2.3), we get:

$$RP_{s_1}^{ovlp} := \{(r_1, r_2), (r_1, r_3), (r_3, r_4), (r_3, r_5), (r_4, r_5), (r_5, r_6), (r_5, r_7), (r_6, r_7)\}$$
$$RP_{s_2}^{ovlp} := \{(r_1, r_2), (r_1, r_3), (r_3, r_5), (r_5, r_6), (r_5, r_7), (r_6, r_7)\}$$

Using these sets, we can create the model. The full description of the model can be found in Appendix A.1.

2.6. Benchmark Robust Model

For the purpose of evaluating the quality of the robust model given in this thesis, KLM provided a Benchmark Robust Model to compare the model to. This model uses two separate sub-models to solve this problem. The result of the first model is used as input for the second model. In this way, some of the complexity of the problem can be split into two models, making both problems significantly faster to optimize, at the cost of optimality. For the sake of confidentiality, the full model description is omitted from this thesis.

Subtype assignment model
This model determines which aircraft subtype is to be used on which rotations, to minimize the total operational costs, without considering the delay costs. To ensure feasibility of the subtype assignment in the final solution schedule, the model actually assigns an aircraft to every rotation, creating a fully feasible solution schedule. The subtypes of the assigned aircraft in this model are used as input for the next model, the robust model.

Robust model
This model is used for aircraft assignment. It determines which aircraft is to be used on which rotation, given the assigned subtype by the subtype assignment model solution. It aims to minimize operational costs, while also improving the overall robustness of the schedule in a heuristic way. The schedule is considered robust if the probability of delay propagation from one rotation to the next is low. This model uses the delay predictions of the flights to model this robustness.

3

Towards Incorporating Delay Propagation

In this section we will give a mathematical description of delay propagation through a schedule, show why incorporating this propagated delay in an optimization is not an easy task, and give an approach for approximating the expected costs of delay, including propagated delays, for a solution of the Robust Tail Assignment problem.

3.1. Mathematical description of delay propagation

If for every flight a probability distribution for the delay of the flight is given, we can mathematically propagate these distributions through the flight schedule. In this section we will describe the workings of this propagation.

Assume we know the probability distribution of the delay of any scheduled flight, and that these distributions are independent. For a flight $f \in \mathcal{F}$, we denote the delay as a random variable $\mathcal{D}_f$, and the propagated delay from previous flights by PD_f. Let us denote the resulting total delay (arrival delay) as a random variable TD_f for every flight $f \in \mathcal{F}$. For these random variables the following equation holds:

$$TD_f = PD_f + \mathcal{D}_f \tag{3.1}$$

Note that the probability distribution of TD_f is equal to the convolution of the probability distributions of PD_f and $\mathcal{D}_f$.

To mathematically describe the propagation of the delay, we define the random variables $PD_{g\to f,s}$ for every flight pair $(g, f) \in \mathcal{F} \times \mathcal{F}$ and aircraft subtype $s \in \mathcal{S}$. This variable holds the propagated delay of flight $f \in \mathcal{F}$, if flight g and f are scheduled to be operated in succession by the same aircraft of subtype $s \in \mathcal{S}$. Let δ_s^f denote the required turnaround time at the airport after a flight f for an aircraft of subtype $s \in \mathcal{S}$. Let std_f and sta_f denote the scheduled departure and arrival time of flight f. The following equation relates the variables $PD_{g\to f,s}$ to the arrival delay of flight g:

$$PD_{g\to f,s} = \max(sta_g + TD_g + \delta_s^g - std_f, 0) \qquad \forall (g, f) \in \mathcal{F} \times \mathcal{F}, s \in \mathcal{S} \tag{3.2}$$

Note that a flight $f \in \mathcal{F}$ has multiple flights $g \in \mathcal{F}$ that could possibly propagate delay to flight f, but only one of these connections is realised in a schedule. For every pair of flights (f, g) and subtype $s \in \mathcal{S}$, define $x_{fgs} \in \{0, 1\}$ to equal 1 if flight g follows flight f using the same aircraft with subtype $s \in \mathcal{S}$ in the schedule, and 0 if not. Then the following equation selects the correct connection in the schedule:

$$PD_f = \sum_{g\in\mathcal{F}, s\in\mathcal{S}} x_{gfs} \cdot PD_{g\to f,s} \qquad \forall f \in \mathcal{F} \tag{3.3}$$

In Equation (3.3), the variables x_{gfs} select correct $PD_{g\to f,s}$, that corresponds to the preceding flight of f, with the correct subtype. Note that in this summation, only one x will equal 1. Also note that for a long fleetline, the PD of the last flight will be dependent on the distributions of all preceding flights.

By Equation (3.2), the propagated delay of a flight is a summation of two random variables with some translating. Note that if we get a negative propagated delay, the flight will depart at its scheduled departing time. Therefore, all the probability of negative propagated delay is assigned to 0 delay using a max-operation. Note that, if we disregard the max-operation, adding and translating random variables are operations that can be performed exactly. But, because of the max-operation, calculating these propagated delay probabilities is not a trivial task. If we consider continuous random variables, the max-operation sends all probability of negative delay to the exact point 0, giving a mixed distribution that is partly discrete in the point $d = 0$, and partly continuous for $d > 0$. Since performing exact operations on these mixed random variables is very difficult, it is convenient to approximate these operations by discretizing the continuous random variables.

3.2. Discretization

To avoid dealing with mixed random variables, we will discretize our continuous random variables. To discretize the distributions, we define a step size $h^{prob} \in \mathbb{R}_{>0}$ (e.g. $h^{prob} = 1$ minute).. Now we can limit our distributions to only consider values that are multiples of this h^{prob}. We get the following set of delay options:

$$T^{delay} := \{t \in \mathbb{R} \text{ s.t. } \exists n \in \mathbb{Z} : t = n \cdot h^{prob}\} \tag{3.4}$$

Note that the random variables of delay may have an infinite number of these points $t \in T^{delay}$ that have a positive probability of occurring. To reduce this amount to a finite number, we can limit the amount of delay options by defining a distribution threshold, the *distribution cutoff point* p^{dcp}. Using this threshold, the set of discrete delay options to consider for the random variable of delay of a flight $f \in \mathcal{F}$ can be limited by defining:

$$t_f^{min} := \max\left(t \in T^{delay} \quad \text{s.t. } \mathbb{P}\left(\mathcal{D}_f \le t\right) \le p^{dcp}\right) \tag{3.5}$$

$$t_f^{max} := \min\left(t \in T_f^{delay} \quad \text{s.t. } \mathbb{P}\left(\mathcal{D}_f \ge t\right) \le p^{dcp}\right) \tag{3.6}$$

$$T_f^{delay} := \{t \in T^{delay} \text{ s.t. } t_f^{min} \le t \le t_f^{max}\} \tag{3.7}$$

Note that if a random variable already has a lower or upper bound, we can refrain from using t^{min} or t^{max}, respectively. The continuous probability distributions of delay for the flights $f \in \mathcal{F}$ can now be discretized by defining the probability mass functions:

$$p_{D_f}(t) := \mathbb{P}\left(t - \frac{h^{prob}}{2} < \mathcal{D}_f \le t + \frac{h^{prob}}{2}\right) \quad \forall t \in T_f^{delay} \backslash \{t_f^{min}, t_f^{max}\} \tag{3.8}$$

$$p_{D_f}(t_f^{min}) := \mathbb{P}\left(\mathcal{D}_f \le t_f^{min} + \frac{h^{prob}}{2}\right) \tag{3.9}$$

$$p_{D_f}(t_f^{max}) := \mathbb{P}\left(t_f^{max} - \frac{h^{prob}}{2} < \mathcal{D}_f\right) \tag{3.10}$$

Note that the discretized random variables are denoted by D_f for all $f \in \mathcal{F}$, as opposed to the continuous random variables $\mathcal{D}_f$.

Example 4.
Let us discretize the probability distribution of expected delay of flight f_1 from the example problem defined in Example 1, using time-steps $h^{prob} = 1$ minute, and a distribution cutoff point $p^{dcp} = 0.001$. The continuous distribution given is a shifted Gamma distribution, which has a lower bound and no upper bound, so we calculate

$$t_{f_1}^{max} = \min\left(t \in \mathbb{Z} \, s.t. \, \mathbb{P}(\mathcal{D}_{f_1} \ge t\right) \le 0.001 = 54$$

Using Definition (3.8) and (3.10), we get the discrete distribution plotted in Figure 3.1. In all the future examples regarding the problem defined in Example 1, the values $h^{prob} = 1$ and $p^{dcp} = 0.001$ are used to

discretize the probability distributions.

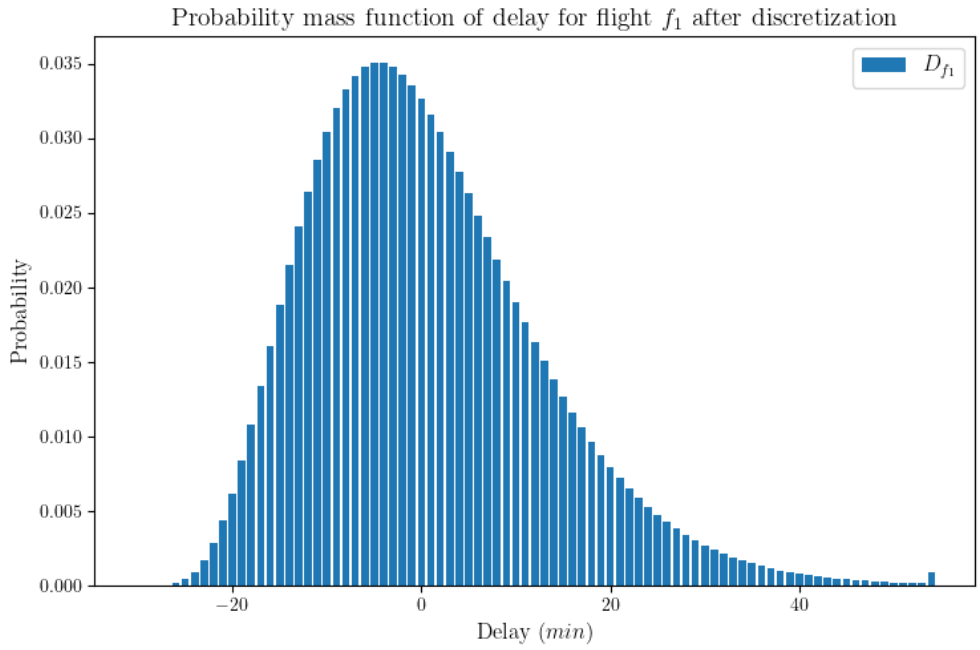

Figure 3.1: Discretized delay prediction for flight f_1, using $h^{prob} = 1$ and $p^{dcp} = 0.001$. The original delay distribution for f_1 is a shifted Gamma distribution Gamma(6,5)-30, see Figure 2.1.

3.2.1. Delay propagation of discrete delay distributions

Let us revisit the derived equations for propagated delay of continuous distributions as defined in Section 3.1. Equation (3.1) and (3.3) still hold when we consider discrete random variables D_f for all $f \in \mathcal{F}$. Equation (3.1) can be used to find the delay distribution of total delay for a flight. Note that if both the random variables PD_f and D_f only take delay values in T^{delay}, the random variable TD_f is also restricted to delay values in T^{delay}.

Equation (3.3) selects the correct $PD_{g\to f,s}$, that corresponds to the preceding flight of f, with the correct subtype.

Equation (3.2) calculates the distribution $PD_{g\to f,s}$, for flights $f, g \in \mathcal{F}$ and $s \in \mathcal{S}$. Note that if $sta_g + \delta_s^g - std_f$ is not a multiple of h^{prob}, using this equation will cause $PD_{g\to f,s}$ to contain delay values t not in T^{delay}. In this case, we need to round the values to values in T^{delay}, giving the following equations for the probability mass function of $PD_{g\to f,s}$:

$$p_{PD_{f\to g,s}}(0) = \sum_{k:\ \mathbb{P}(PD_f=k)>0} \mathbb{P}\left(D_f \leq std_g - std_f - \delta_s - k + \frac{h^{prob}}{2}\right) \cdot \mathbb{P}(PD_f = k) \tag{3.11}$$

$$p_{PD_{f\to g,s}}(t) = \sum_{k:\ \mathbb{P}(PD_f=k)>0} \mathbb{P}\left(\frac{h^{prob}}{2} < D_f - (std_g - sta_f - \delta_s + t - k) \leq \frac{h^{prob}}{2}\right) \cdot \mathbb{P}(PD_f = k) \quad \forall t \in T^{delay} \tag{3.12}$$

Note that if $sta_g + \delta_s^g - std_f$ is a multiple of h^{prob}, Equation (3.11) and (3.12) do not require any rounding, and reduce to:

$$p_{PD_{f\to g,s}}(0) = \sum_{k:\ \mathbb{P}(PD_f=k)>0} \mathbb{P}\left(D_f \leq std_g - std_f - \delta_s - k\right) \cdot \mathbb{P}(PD_f = k) \tag{3.13}$$

$$p_{PD_{f\to g,s}}(t) = \sum_{k:\ \mathbb{P}(PD_f=k)>0} \mathbb{P}\left(D_f = std_g - sta_f - \delta_s + t - k\right) \cdot \mathbb{P}(PD_f = k) \quad \forall t \in T^{delay} \tag{3.14}$$

Note that if $sta_g + \delta_s^g - std_f$ is a multiple of h^{prob}, Equation (3.13) and (3.14) are equivalent to Equation (3.2).

Example 5.
To provide some insight into this propagation of delay using Equations (3.1), (3.3), (3.2), (3.13) and (3.14), let us go over a simple example. We consider two flights f and g scheduled to be operated on aircraft a, where g is operated right after f. For simplicity sake, we refrain from using actual timestamps, and just use generic "time-steps". Let $sta_f = 0$, $std_g = 30$, $\delta_{s_a} = 28$. The discrete distributions of PD_f and D_f, with $h^{prob} = 1$ can be found in Table 3.1 and 3.2. Note that the values of sta_f, std_g and δ_{s_a} are all multiples of h^{prob}, so there is no need for rounding.

Delay	Prob
0	0.7
1	0.2
2	0.1

Table 3.1: PD_f

Delay	Prob
-1	0.5
0	0.2
1	0.15
2	0.1
3	0.05

Table 3.2: D_f

Delay	Prob
-1	0.35
0	0.24
1	0.195
2	0.12
3	0.07
4	0.02
5	0.005

Table 3.3: $(PD_f + D_f) = TD_f$

Delay	Prob
0	0.905
1	0.07
2	0.02
3	0.005

Table 3.4: $\max(sta_f + PD_f + D_f + \delta_{s_a} - std_g, 0) = PD_g$

Firstly, we use Equation (3.2) to calculate $PD_{f\to g,s}$. Using the probabilities of delays of PD_f and D_f, we can find the distribution of $(PD_f + D_f)$. We get:

$$\begin{aligned}
\mathbb{P}(PD_f + D_f = -1) &= \mathbb{P}(PD_f = 0) \cdot \mathbb{P}(D_f = -1) = 0.7 \cdot 0.5 = 0.35 \\
\mathbb{P}(PD_f + D_f = 0) &= \mathbb{P}(PD_f = 0) \cdot \mathbb{P}(D_f = 0) + \mathbb{P}(PD_f = 1) \cdot \mathbb{P}(D_f = -1) \\
&= 0.7 \cdot 0.2 + 0.2 \cdot 0.5 = 0.24 \\
\mathbb{P}(PD_f + D_f = 1) &= \mathbb{P}(PD_f = 0) \cdot \mathbb{P}(D_f = 1) + \mathbb{P}(PD_f = 1) \cdot \mathbb{P}(D_f = 0) \\
&\quad + \mathbb{P}(PD_f = 2) \cdot \mathbb{P}(D_f = -1) \\
&= 0.7 \cdot 0.15 + 0.2 \cdot 0.1 + 0.1 \cdot 0.5 = 0.195 \\
&\vdots
\end{aligned}$$

In this way we can calculate all the values and probabilities for $(PD_f + D_f)$, see Table 3.3. Note that by Equation (3.1), this is equal to TD_f. To find the values for $PD_g = PD_{f\to g} = \max(sta_f + T_f + \delta_{s_a} - std_g, 0)$, we first translate this distribution. We have $sta_f + \delta_{s_a} - std_g = -2$, so we translate the values of T_f by two time-units. After taking the max*, we get the distribution found in Table 3.4. Note that we could have also*

used Equation (3.13) and (3.14), giving:

$$
\begin{aligned}
p_{PD_{f\to g,s}}(0) &= \sum_{k\in\{0,1,2\}} \mathbb{P}\left(D_f \leq 30-0-28-k\right)\cdot PD_f(k)\\
&= \mathbb{P}\left(D_f \leq 2\right)\cdot PD_f(0) + \mathbb{P}\left(D_f \leq 1\right)\cdot PD_f(1) + \mathbb{P}\left(D_f \leq 0\right)\cdot PD_f(2)\\
&= 0.95\cdot 0.7 + 0.85\cdot 0.2 + 0.7\cdot 0.1 = 0.905\\
p_{PD_{f\to g,s}}(1) &= \sum_{k\in\{0,1,2\}} \mathbb{P}\left(D_f = 30-0-28+1-k\right)\cdot PD_f(k)\\
&= \mathbb{P}\left(D_f = 3\right)\cdot PD_f(0) + \mathbb{P}\left(D_f = 2\right)\cdot PD_f(1) + \mathbb{P}\left(D_f = 1\right)\cdot PD_f(2)\\
&= 0.05\cdot 0.7 + 0.1\cdot 0.2 + 0.15\cdot 0.1 = 0.07\\
p_{PD_{f\to g,s}}(2) &= \sum_{k\in\{0,1,2\}} \mathbb{P}\left(D_f = 30-0-28+2-k\right)\cdot PD_f(k)\\
&= \mathbb{P}\left(D_f = 4\right)\cdot PD_f(0) + \mathbb{P}\left(D_f = 3\right)\cdot PD_f(1) + \mathbb{P}\left(D_f = 2\right)\cdot PD_f(2)\\
&= 0\cdot 0.7 + 0.05\cdot 0.2 + 0.1\cdot 0.1 = 0.02\\
p_{PD_{f\to g,s}}(3) &= \sum_{k\in\{0,1,2\}} \mathbb{P}\left(D_f = 30-0-28+3-k\right)\cdot PD_f(k)\\
&= \mathbb{P}\left(D_f = 5\right)\cdot PD_f(0) + \mathbb{P}\left(D_f = 4\right)\cdot PD_f(1) + \mathbb{P}\left(D_f = 3\right)\cdot PD_f(2)\\
&= 0\cdot 0.7 + 0\cdot 0.2 + 0.05\cdot 0.1 = 0.005
\end{aligned}
$$

All distributions in this example are also plotted in Figure 3.2.

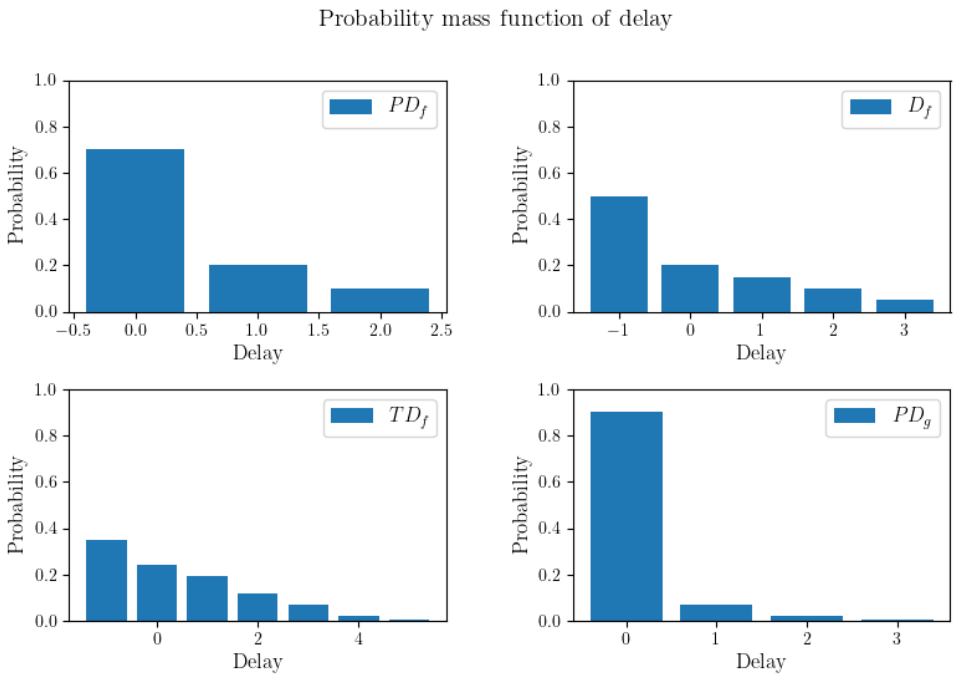

Figure 3.2: Probability mass functions of the random variables evaluated in the example.

Now that we have seen the workings of Equations (3.1), (3.3), (3.2), (3.13) and (3.14), we can apply these equations to a fleetline of our example problem.

Example 6.
Let us consider the fleetline of aircraft a_1 for our example problem defined in Example 1, in the solution schedule from Figure 2.2. The fleetline contains two rotations, r_1 and r_5. Both rotations contain two flights,

where delay may propagate between the flights. Similarly to Example 4, we discretize the delay distributions of the flights in the fleetline using $h^{prob} = 1$ *and* $p^{dcp} = 0.001$*, see Figure 3.3. Note that all values defined in Example 1 are multiples of* h^{prob}*, so we can use Equations (3.13) and (3.14).*

Since rotation r_1 *is the first rotation in the fleetline of* a_1 *we have no propagated delay for* f_1*, i.e.* $PD_{f_1} = 0$*. Using Equation (3.13) and (3.14), we can calculate the distribution of propagated delays* PD_f *for flights* $f \in \{f_2, f_8, f_9\}$*, see Figure 3.4. We see that for this fleetline, there is only a very small probability of propagating delay from flight* f_2 *to flight* f_8*.*

The distributions of the random variables $TD_f = PD_f + D_f$ *for* $f \in \{f_1, f_2, f_8, f_9\}$ *are given in Figure 3.5.*

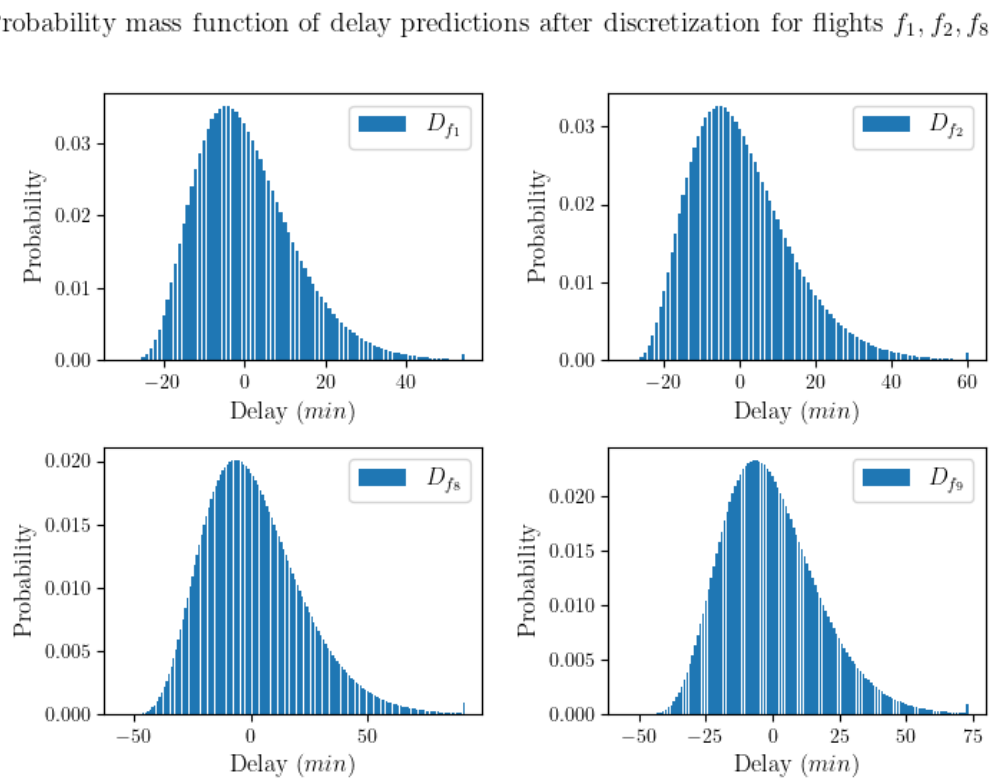

Figure 3.3: Discretized delay predictions for flights f_1, f_2, f_8 and f_9, using $h^{prob} = 1$ and $p^{dcp} = 0.001$.

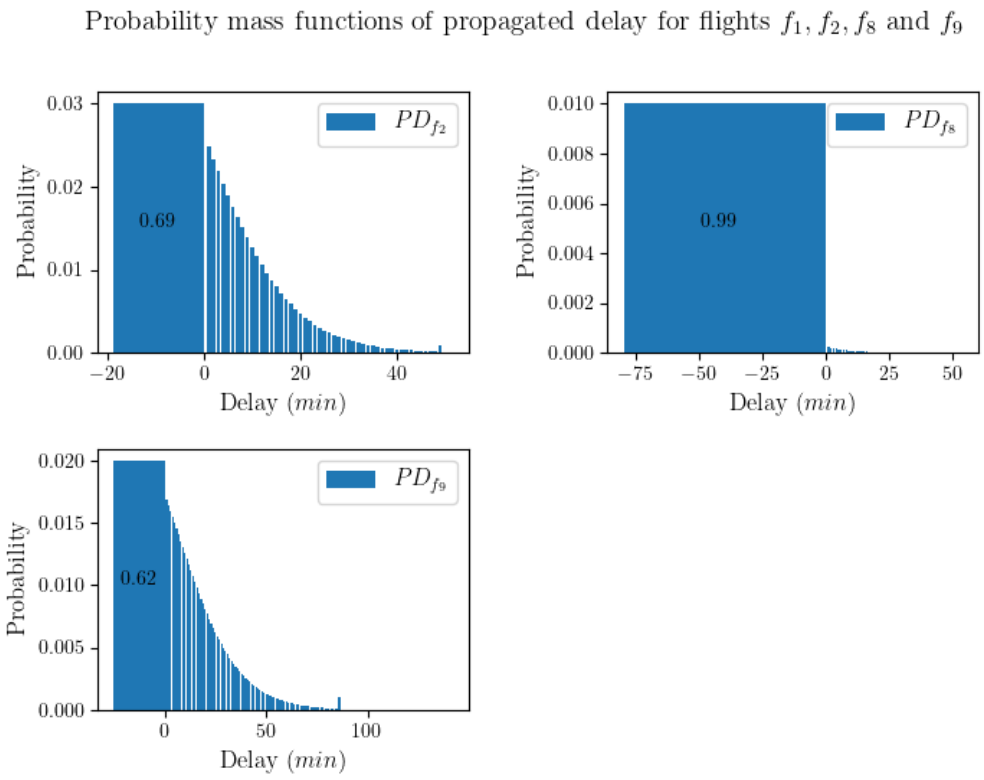

Figure 3.4: Propagated delay distributions for flights f_2, f_8 and f_9. The leftmost bars represent the probability of no delay. The area of the bars are consistent with this probability, which is written inside the bar.

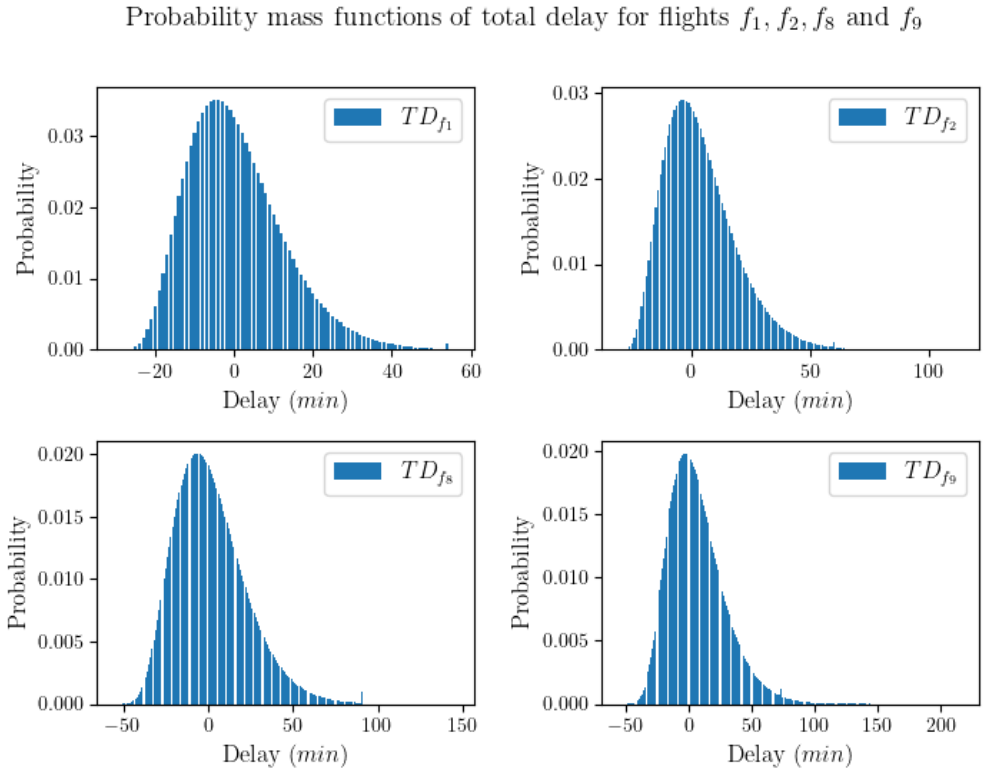

Figure 3.5: Total delay distribution of arrival, for flights f_1, f_2, f_8 and f_9.

3.3. Cost of delay

If a flight arrives later than scheduled, the airline has to pay several costs, depending on multiple factors. Among these factors are the amount of delay, amount of passengers, missed connections of passengers and expected future value loss. Based on these costs, a cost function based on amount of delay minutes can be defined for every flight. So we define:

$$c_f^{delay}(t) := \text{ Cost of flight } f \in \mathcal{F} \text{ arriving } t \text{ minutes late.}$$

Using this definition, the total expected delay cost of a flight $f \in \mathcal{F}$ is equal to:

$$\mathbb{E}\left[c_f^{delay}\left(\max(TD_f, 0)\right)\right] = \sum_{d \in TD_f} \mathbb{P}(TD_f = d) \cdot c_f^{delay}(\max(d, 0)) \tag{3.15}$$

Note that if a flight arrives earlier than scheduled, it has no delay, so we round negative delays to 0 in the cost function. The total expected delay cost of a schedule can now be expressed as:

$$\sum_{f \in \mathcal{F}} \mathbb{E}\left[c_f^{delay}\left(\max(TD_f, 0)\right)\right] = \sum_{f \in \mathcal{F}} \sum_{d \in TD_f} \mathbb{P}(TD_f = d) \cdot c_f^{delay}(\max(d, 0)) \tag{3.16}$$

Example 7.
Let us again consider the fleetline of a_1 for our example problem defined in Example 1, in the solution schedule from Figure 2.2. The expected delay cost of this fleetline is equal to the sum of the expected delay costs of its flights. Using the distributions found in Figure 3.5 we get:

$$\mathbb{E}\left[c_{f_1}^{delay}\left(\max(TD_{f_1}, 0)\right)\right] = \sum_{d \in TD_{f_1}} \mathbb{P}\left(TD_{f_1} = d\right) \cdot c_{f_1}^{delay}\left(\max(d, 0)\right) = 100.77$$

$$\mathbb{E}\left[c_{f_2}^{delay}\left(\max(TD_{f_2}, 0)\right)\right] = \sum_{d \in TD_{f_2}} \mathbb{P}\left(TD_{f_2} = d\right) \cdot c_{f_2}^{delay}\left(\max(d, 0)\right) = 137.68$$

$$\mathbb{E}\left[c_{f_3}^{delay}\left(\max(TD_{f_3}, 0)\right)\right] = \sum_{d \in TD_{f_8}} \mathbb{P}\left(TD_{f_8} = d\right) \cdot c_{f_8}^{delay}\left(\max(d, 0)\right) = 72.31$$

$$\mathbb{E}\left[c_{f_4}^{delay}\left(\max(TD_{f_4}, 0)\right)\right] = \sum_{d \in TD_{f_9}} \mathbb{P}\left(TD_{f_9} = d\right) \cdot c_{f_9}^{delay}\left(\max(d, 0)\right) = 106.85$$

So the total expected delay cost of the fleetline of a_1 is equal to

$$\sum_{f \in \{f_1, f_2, f_8, f_9\}} \mathbb{E}\left[c_f^{delay}\left(\max(TD_f, 0)\right)\right] = 417.61$$

4

Robust Flow Model

In this chapter a new proposed model for the Robust Tail Assignment problem will be motivated and described. In the first section, various model approaches are discussed. In the later sections, the model will be explained, built and formulated.

4.1. Model approaches

To find a solution that is close to optimal, the delay costs of a solution need to be accurately approximated. Therefore, it is necessary to incorporate the expected propagated delay distributions for all flights in the model. The delay costs of flights in the schedule are dependent on all earlier scheduled flights by propagation of delay. This dependency is the main difficulty of the problem.

One way to solve the problem is by brute force, i.e. trying all possible solutions. For a full schedule with n rotations and m aircraft, we have $\mathbb{O}((m!)^{n/m})$ possible solutions [1]. For all these solutions, the total expected costs are to be compared. To find the delay costs of a solution, the propagated delay distributions for all flights need to be calculated, using Equation (3.2) for every flight. This means we need to perform these heavy operations on random variables $\mathbb{O}(n \cdot (m!)^{n/m})$ times, which will take a very long time. Therefore, brute force is not feasible for solving the Robust Tail Assignment problem.

Another approach to solve the problem would be to use dynamic programming. This method requires the problem to be dividable into nested sub-problems, where the optimal solution of a sub-problem is also a part of the optimal solution of the whole problem. But, because of the dependencies in the problem, this is not the case, meaning we cannot use dynamic programming.

As deduced by exploring the brute force approach, calculating the propagated delay distributions for all possible fleetlines in a solution is not feasible. An approach often used in literature is a column generating algorithm, using the set partitioning formulation as defined in Section 1.2.1. In this formulation, possible fleetlines correspond to columns in a Mixed-Integer Linear Programming (MIP). Since it is infeasible to consider all possible fleetlines, columns that can improve the solution are dynamically added to the linear relaxation of the problem, until no more improving columns exist. At this point the linear relaxation is optimally solved. A rounding heuristic can be used to round the solution to an integer solution. Note that, because of this final rounding stage, the optimality is lost.

To find optimal integer solutions, delay propagation and delay costs needs to be incorporated in a model of the problem. In a solution, the propagation of delay distributions can locally be described as a dependency of a rotation to the previous and the next rotation. This local relation can be depicted using a graph of nodes and edges, where nodes correspond to rotations, which are connected by edges

[1] A full schedule means a schedule where all aircraft are operational most of the time. In the worst case, this schedule consists of n/m blocks of time (with enough turnaround time for the aircraft in between), where in each of these timeblocks m rotations are operated by the m aircraft. In that case, each block has a total of $m!$ possible assignments. Since we have n/m independent blocks, we have a total of $(m!)^{n/m}$ possible solutions.

that represent the operation of delay propagation. The model described in this thesis is a MIP model. The model uses a depiction of local dependencies, to accurately approximate the delay propagation through the entire fleetline. The operations of delay propagation are performed by considering various states of delay a rotation can reside in, where a state is connected to all states it can possibly transition into in any solution by directional edges. Probabilities are calculated using flow over these edges. The probability of a rotation being in a certain state is determined by the flow over its connected edges. Depending on the assignments of a solution, edges in this network are activated or deactivated, to be able to approximate the correct delay distributions for that solution. The values of the active edges will be used to approximate the total expected delay costs of a solution.

The creation of this model will be explained in detail and in small steps in the next sections.

4.2. Aircraft assignment

The Robust Tail Assignment problem is an assignment problem, with the same degree of freedom as the Non-Robust Tail Assignment problem. Therefore, the base of the model is similar to the Non-Robust Tail Assignment model, as described in 2.5. For every feasible rotation-aircraft combination, a decision variable X is created, which can take the binary values 0 and 1:

$$X_{r,a} \in \{0,1\} \qquad \forall r \in \mathcal{R}, a \in \mathcal{A}_r. \tag{4.1}$$

If such a variable $X_{r,a}$ has value 1 in a model solution, rotation r will be assigned to aircraft a is the schedule. These values form the solution of the model. In a solution, exactly one aircraft needs to be assigned to every rotation, which can be enforced using the constraints:

$$\sum_{a \in \mathcal{A}_r} X_{r,a} = 1 \qquad \forall r \in \mathcal{R}. \tag{4.2}$$

To ensure the assignment costs are optimized for in the model, the sum over these costs for every assignment variable is added to the objective value:

$$\sum_{r \in \mathcal{R}} \sum_{a \in \mathcal{A}_r} c_{r,a}^{assign} \cdot X_{r,a}. \tag{4.3}$$

Finally, to have a solution schedule that is feasible, rotations that will overlap if assigned to the same aircraft must be forced to be assigned to different aircraft. In the Non-Robust Tail Assignment model from Section 2.5, this is achieved by finding pairs of rotations that overlap, and defining constraints for these pairs, see Equation (2.3). Note that if we have k rotations that all overlap with each other, this requires $k \cdot (k-1)/2 = \mathbb{O}(k^2)$ constraints per aircraft, while the same can be achieved using one constraint per aircraft, forcing the sum of the values of X corresponding to this aircraft and the k rotations to be less than or equal to 1.

To find these sets of overlapping rotations, let us define a small preprocessing algorithm, shown in Algorithm 2. Since the turnaround time depends on the aircraft subtype, we define the variables RS_s^{ovlp} to hold rotation sets, that overlap if two of these rotations are assigned to the same aircraft of subtype s. The idea of this algorithm is create a running set, adding rotations to this set in chronological order, based on their std_r. If this std_r is higher than the sta_q plus turnaround time of any of the rotations q currently in the running set, these rotations are deleted from the set. In this way the running set always contains rotations that all overlap with each other, for which we can create a single constraint per aircraft.

What remains is to define the corresponding feasibility constraints:

$$\sum_{r \in rs, \text{ s.t. } a \in \mathcal{A}_r} X_{r,a} \leq 1 \qquad \forall a \in \mathcal{A}, rs \in RS_{s_a}^{ovlp}. \tag{4.4}$$

Algorithm 2 Determine overlapping rotations

1: $RS_s^{ovlp} \leftarrow$ empty set that will hold rotation sets $\quad \forall s \in \mathcal{S}$
2: $sortedList \leftarrow$ list of all rotations $r \in \mathcal{R}$, sorted by std_r
3: $runningSet_s \leftarrow$ empty set that will hold the running sets $\quad \forall s \in \mathcal{S}$
4: **for** $r \in sortedList$ **do**
5: **for** $s \in \mathcal{S}_r$ **do**
6: $toDelete \leftarrow$ empty set that will hold rotations to be deleted
7: **for** $q \in runningSet_s$ **do**
8: **if** $sta_q + \delta_s^0 \leq std_r$ **then**
9: add q to $toDelete$
10: **end if**
11: **end for**
12: **if** $|toDelete| > 0$ **then**
13: add $runningSet_s$ to RS_s^{ovlp}
14: **end if**
15: **for** $q \in toDelete$ **do**
16: delete q from $runningSet_s$
17: **end for**
18: add r to $runningSet_s$
19: **end for**
20: **end for**
21: **for** $s \in \mathcal{S}$ **do**
22: **if** $runningSet_s \notin RS_s^{ovlp}$ **then**
23: add $runningSet_s$ to RS_s^{ovlp}
24: **end if**
25: **end for**
26: **return** $RS_s^{ovlp} \quad \forall s \in \mathcal{S}$

Example 8.
We apply Algorithm 2 to create feasibility constraints for our example problem, as defined in Example 1.

We loop over the rotations in order of std, so we start with r_1. We have $S_{r_1} = \{s_1, s_2\}$. The sets $runningSet_{s_1}$ and $runningSet_{s_2}$ are intially empty, so after evaluating r_1 we have:

$$runningSet_{s_1} = runningSet_{s_2} = \{r_1\}$$

Now r_2 is evaluated. We have $sta_{r_1} + \delta_{s_1}^0 = 125 + 25 > 5 = std_{r_2}$ and $sta_{r_1} + \delta_{s_2}^0 = 125 + 30 > 5 = std_{r_2}$, so no rotations are to be deleted from the running sets. We get:

$$runningSet_{s_1} = runningSet_{s_2} = \{r_1, r_2\}$$

After evaluating r_3, we have:

$$runningSet_{s_1} = runningSet_{s_2} = \{r_1, r_2, r_3\}$$

Evaluating r_4, we only loop over $S_{r_4} = \{s_1\}$. We have:

$$sta_{r_1} + \delta_{s_1}^0 = 125 + 25 \leq 205 = std_{r_4}$$
$$sta_{r_2} + \delta_{s_1}^0 = 175 + 25 \leq 205 = std_{r_4}$$
$$sta_{r_3} + \delta_{s_1}^0 = 240 + 25 > 205 = std_{r_4}$$

We now have $toDelete = \{r_1, r_2\}$, so by line 12-18 of the algorithm, we get $RS_{s_1}^{ovlp} = \big\{\{r_1, r_2, r_3\}\big\}$, and

$runningSet_{s_1} = \{r_3, r_4\}$. *Looping over the last three rotations* r_5, r_6 *and* r_7*, we end up with the sets:*

$$\begin{aligned} RS_{s_1}^{ovlp} &= \{\{r_1, r_2, r_3\}, \{r_3, r_4, r_5\}\} \\ RS_{s_2}^{ovlp} &= \{\{r_1, r_2, r_3\}, \{r_3, r_5\}\} \\ runningSet_{s_1} &= \{r_5, r_6, r_7\} \\ runningSet_{s_2} &= \{r_5, r_6, r_7\} \end{aligned}$$

In lines 21-25, these final running sets are added to the overlapping sets, and we get:

$$\begin{aligned} RS_{s_1}^{ovlp} &= \{\{r_1, r_2, r_3\}, \{r_3, r_4, r_5\}, \{r_5, r_6, r_7\}\} \\ RS_{s_2}^{ovlp} &= \{\{r_1, r_2, r_3\}, \{r_3, r_5\}, \{r_5, r_6, r_7\}\} \end{aligned}$$

Now, we can define the feasibility constraints for the example problem, using Equation (4.4):

$$\begin{aligned} X_{r_1,a_1} + X_{r_2,a_1} + X_{r_3,a_1} &\leq 1 \\ X_{r_1,a_2} + X_{r_2,a_2} + X_{r_3,a_2} &\leq 1 \\ X_{r_1,a_3} + X_{r_2,a_3} + X_{r_3,a_3} &\leq 1 \\ X_{r_3,a_1} + X_{r_5,a_1} &\leq 1 \\ X_{r_3,a_2} + X_{r_4,a_2} + X_{r_5,a_2} &\leq 1 \\ X_{r_3,a_3} + X_{r_5,a_3} &\leq 1 \\ X_{r_5,a_1} + X_{r_6,a_1} + X_{r_7,a_1} &\leq 1 \\ X_{r_5,a_2} + X_{r_6,a_2} + X_{r_7,a_2} &\leq 1 \\ X_{r_5,a_3} + X_{r_6,a_3} + X_{r_7,a_3} &\leq 1 \end{aligned}$$

Note that only 9 constraints are required, as opposed to 23 constraints if we create overlapping constraint per pair of rotations.

4.3. Delay propagation by flow

As described in Section 3, subsequent flights assigned to the same aircraft may propagate delay. This section will describe a way to estimate the expected delay costs of a schedule, taking into account this delay propagation, using a flow network. Firstly, Section 4.3.1 describes what a flow network is. Section 4.3.2 describes how a flow network can be used to model propagation of probabilities. Then, in Section 4.3.3, an algorithm for determining the pairs of rotations that may propagate delay is given. Finally, in Section 4.4, the necessary steps for creating the network are described. Section 4.5.2 will contain the mathematical formulation of the model.

4.3.1. Flow network

In graph theory, a flow network is a directed graph containing nodes and edges. Every edge receives a flow value, which cannot exceed the capacity of the edge. This flow must satisfy the flow propagation restriction, which means that in every node that is not a source or sink, the sum of flows of incoming edges is equal to the sum of flows of outgoing edges. A source node has a predefined total outgoing flow, while a sink node has a predefined total incoming flow.

4.3.2. Network description

The general idea of this network is to create a network of connected nodes, corresponding to departure and arrival timestamps for every rotation, with directional edges between nodes, where flow represents probability. The network will contain states as nodes, where the flow though these nodes corresponds to the probability that this state occurs. These state nodes will be subtype dependent, since the subtype of the assigned has an impact on the turnaround time of the aircraft between flights, and therefore has an impact on the delay propagation. Edges in this network will be activated based on the assignments created by the assignment decision variables of the base model, using constraints. To ensure conservation

of probability, for every node the outgoing probability should be equal to the incoming probability, which can also be achieved by constraints.

For this model, instead of the actual timestamps, let us consider the relative time compared to the start of the schedule, in minutes. For example, for a schedule that starts on January 1st at 00:00, the timestamp 01/01 01:10 will have a relative time of $60 + 10 = 70$ minutes. This means that we will refer to this timestep as $t = 70$.

The nodes in this network correspond to timestep states where a rotation can possibly start or end, and are subtype dependent. We will denote these nodes by $(t, r, s), t \in T, r \in \mathcal{R}, s \in \mathcal{S}_r$. There are two types of nodes, departure state nodes and arrival state nodes, for every rotation. We denote the departure and arrival state nodes for a rotation $r \in \mathcal{R}$ and subtype $s \in \mathcal{S}$ by N_{rs}^{dep} and N_{rs}^{arr}, respectively. The nodes in N_{rs}^{dep} represent the states of departure for rotation r, so a node $(t, r, s) \in N_{rs}^{dep}$ corresponds to the state that rotation r departs at timestep t on an aircraft of subtype s. Similarly, the nodes in N_r^{arr} represent the states of arrival for rotation r, so a node $(t, r, s) \in N_{rs}^{arr}$ corresponds to the state that rotation r arrives at timestep t on an aircraft of subtype s. For this model, we incorporate the turnaround time at the hub airport into the arrival states. This means a state $(t, r, s) \in N_{rs}^{arr}$ indicates the assigned aircraft is ready for the next rotation at timestep t.

These nodes are connected by edges, that can take any value between 0 and 1, corresponding to probabilities. There are two types of edges, edges between rotations and edges within rotations. Edges between rotations connect arrival state nodes of one rotation to departure state nodes of another rotation. Edges within rotations connect departure state nodes to arrival state nodes. Nodes are only connected by edges if the transition of states from the outgoing state node to the incoming state node is feasible and has a positive probability of happening, based on the delay predictions. Precisely how this is determined, will be described in the following sections.

The total incoming flow into a node $(t, r, s) \in N_{rs}^{dep}$ represents the probability that rotation r departs at timestep t on an aircraft of subtype s, i.e. $\mathbb{P}\left(t_r^{dep} = t \wedge \text{ assigned subtype } = s\right)$. Note that if, in a solution, an aircraft with another subtype is assigned, this probability will be equal to 0, thus the node should have no incoming flow. Similarly, the total incoming flow into a node $(t, r, s) \in N_{rs}^{out}$ represents the probability that rotation r is turned around at the hub station at timestep t on an aircraft of subtype s:

$$\mathbb{P}\left(t_r^{arr} = t \wedge \text{ assigned subtype } = s\right).$$

An edge between two nodes represents the probability that the first and the second event happen. As we have seen in Section 2.2, within a single rotation, the delay probabilities of the arrival of the last flight in the rotation are solely dependent on the probabilities of the departure of the first flight in the rotation (propagated delay from previous rotations), and the subtype of the assigned aircraft. Therefore, we create edges that are subtype specific. That way, we can preprocess all delay propagation within every rotation. For example, an edge from a node $(t_1, r, s) \in N_{rs}^{dep}$ to a node $(t_2, r, s) \in N_{rs}^{arr}$ represents the probability:

$$\begin{aligned}
&\mathbb{P}\left(t_r^{dep} = t_1 \wedge t_r^{arr} = t_2 \wedge \text{ assigned subtype } = s\right) \\
&= \mathbb{P}\left(t_r^{arr} = t_2 | t_r^{dep} = t_1, \text{ assigned subtype } = s\right) \cdot \mathbb{P}\left(t_r^{dep} = t_1 \wedge \text{ assigned subtype } = s\right). \qquad (4.5)
\end{aligned}$$

The probability $\mathbb{P}\left(t_r^{arr} = t_2 | t_r^{dep} = t_1, \text{ assigned subtype } = s\right)$ can be calculated in preprocessing, using Equations (3.13) and (3.14) from Section 2.2, while the probability $\mathbb{P}\left(t_r^{dep} = t_1 | \text{ assigned subtype } = s\right)$ is equal to the incoming flow from edges corresponding to subtype s in the node (t_1, r, s). Therefore, we can enforce the right amount of flow into the corresponding edge $((t_1, r), (t_2, r), s)$, using a linear constraint. These constraints, and this preprocessing step will be further explained in Section 4.4.2

Edges between arrival states of a rotation $N_{r_1 s}^{arr}$ and departure states of another rotation $N_{r_2 s}^{dep}$ are quite straightforward. For every rotation pair (r_1, r_2) and subtype s for which $r_1 \rightarrow r_2$ is a feasible connection on an aircraft of subtype s, we create edges. Every arrival state $(t_1, r_1, s) \in N_{r_1 s}^{arr}$ is connected to the first feasible departure state $(t_2, r_2, s) \in N_{r_2 s}^{dep}$. This means we take the minimum t_2, such that

$(t_2, r_2, s) \in N^{dep}_{r_2 s}$ and $t_2 \geq t_1$, meaning it is a feasible connection. If rotations r_1 and r_2 are assigned to be operated in succession on the same aircraft of subtype s by the decision variables, these edges are activated. If this is not the case, these edges are set to 0.

4.3.3. Propagation pairs

If we consider this network for delay propagation, we need to connect the arrival state nodes of a rotation scheduled early in the day to departure state nodes of all feasible rotations that can be scheduled directly after this rotation. This means we need to create a great number of edges between rotations, creating many variables. Having many variables will cause the model to be more difficult to solve, impacting the optimization speed. Since for a lot of these connections, the probability of propagating delay is very small or even negligible, we might be able to disregard these connections, and reduce the number of edges in the network, speeding up the optimization. To accomplish this, let us set a threshold of probability, where if a connection in the worst-case scenario has a probability of delay propagation lower than this threshold, we disregard the connection. We will call this threshold the *propagation cutoff point* or p^{pcp}. So, for a connection between two rotations r_1 and r_2, we consider the worst-case scenario of propagated delay for rotation r_1. If we assign r_2 to be operated right after r_1, and the probability of r_2 having any propagated delay from r_1 is smaller than p^{pcp}, we do not connect arrival state nodes of r_1 to departure state nodes of r_2. If we do assign these rotations to the same aircraft, we send all probability from arrival state nodes of r_1 to a sink and let the departure state node of r_2 that corresponds to no delay be a source of 1 flow, disregarding the propagated delay probability. In this way, the number of edges between rotations in the network can be greatly reduced.

To determine which pairs of rotation do need to be connected by edges, and to determine the possible departure and arrival states options that need to be considered for every rotation, we need to know exactly how much delay can be propagated from previous rotations. Let us define a small pre-processing algorithm to determine the maximum timestep where these rotations can arrive over all possible assignments, as well as the maximum timestep where the threshold p^{pcp} is passed, using the given probability distributions of delay.

For every feasible rotation-aircraft combination $(r, a), r \in \mathcal{R}, a \in \mathcal{A}_r$, we define the variables $tmax^{dep}_{ra}, tmax^{arr}_{ra}, tmax^{pcp}_{ra}$ as:

$tmax^{dep}_{ra}$	The maximum timestep where rotation r can depart if assigned to aircraft a, considering possible propagated delays.
$tmax^{arr}_{ra}$	The maximum timestep where rotation r can be turned around at the hub airport if assigned to aircraft a, considering possible propagated delays.
$tmax^{pcp}_{ra}$	The maximum timestep where the p^{pcp} threshold is passed, considering possible propagated delays.

Since for a rotation $r \in \mathcal{R}$, the time-steps $tmax^{dep}_{ra}$ and $tmax^{pcp}_{ra}$ only depend on rotations q that are scheduled to arrive before r departs in the schedule, we can determine these time-steps by evaluating the rotations in order of increasing std. Note that the time-steps $tmax^{pcp}_{ra}$ depend on the full worst-case probability distribution of propagated delay of r, it is dependent all the distributions of previous rotations. To simplify this, we will only consider the worst-case mean value ($meanProp_{ra}$) of the propagated delay from previous rotations to calculate this value.

Using these variables, we can define an algorithm that can be used to determine all values of $tmax^{dep}_{ra}$, $tmax^{arr}_{ra}$ and $tmax^{pcp}_{ra}$, see Algorithm 3.

First, for every rotation $r \in \mathcal{R}$, the maximum duration of the flights (dur^{max}_f) is determined, based on the delay predictions of these flights. Also, a list of all rotations ($sortedList$) is created, sorted by std. For every rotation-aircraft combination $(r, a), r \in \mathcal{R}, a \in \mathcal{A}$, we initially save the worst-case mean of propagated delay as $meanProp_{ra} = 0$. Also, an empty set $propPairs$ is created, which will hold pairs of

rotations with aircraft subtypes, for which we need to create edges.

Algorithm 3 Determine propagation pairs

$dur_f^{max} \leftarrow sta_f - std_f + max(D_f) \qquad \forall f \in \mathcal{F}$
$sortedList \leftarrow$ list of all rotations $r \in \mathcal{R}$, sorted by std_r
$propPairs \leftarrow$ empty set that will hold tuples (r_1, r_2, s) to create edges for
$tmax_{ra}^{dep} \leftarrow std_r \qquad \forall r \in \mathcal{R}, a \in \mathcal{A}_r$
$tmax_{ra}^{arr} \leftarrow sta_r \qquad \forall r \in \mathcal{R}, a \in \mathcal{A}_r$
$tmax_{ra}^{pcp} \leftarrow 0 \qquad \forall r \in \mathcal{R}, a \in \mathcal{A}_r$
$meanProp_{ra} \leftarrow 0 \qquad \forall r \in \mathcal{R}, a \in \mathcal{A}_r$
for $r \in sortedList$ **do**
  for $a \in \mathcal{A}_r$ **do**
    $maxMean_a \leftarrow std_r$
    for $q \in sortedList_{<r}$ **do**
      if $q \rightarrow r$ is feasible & $tmax_{qa}^{pcp} > std_r$ **then**
        add (q, r, s_a) to $propPairs$
        $maxMean_a \leftarrow \max(maxMean_a, meanProp_{qa})$
        if r is a maintenance block **then**
          $tmax_{ra}^{dep} \leftarrow \max(tmax_{ra}^{dep}, tmax_{qa}^{arr})$
        else
          $tmax_{ra}^{dep} \leftarrow \max(tmax_{ra}^{dep}, \min(tmax_{qa}^{arr}, std_r + 120))$
        end if
      end if
    end for
    if r contains two flights, f_1, f_2. **then**
      $tmax_{ra}^{arr} \leftarrow \max(std_{f_2}, tmax_{ra}^{dep} + dur_{f_1}^{max} + \delta_{s_a}^{r}) + dur_{f_2}^{max} + \delta_{s_a}^{0}$
      $Dmax_{ra}^{arr} = \max(maxMean_a + D_{f_1} + sta_{f_1} - std_{f_1} + \delta_{s_a}^{r}, std_{f_2}) +$
        $D_{f_2} + sta_{f_2} - std_{f_2} + \delta_{s_a}^{0}$
    else ▷ r is a maintenance block
      $tmax_{ra}^{arr} \leftarrow tmax_{ra}^{dep} + dur_{f_1}^{max} + \delta_{s_a}^{0}$
      $Dmax_{ra}^{arr} \leftarrow maxMean_a + D_{f_1} + sta_r - std_r + \delta_{s_a}^{0}$
    end if
    $meanProp_{ra} \leftarrow \mathbb{E}[Dmax_{ra}^{arr}]$
    $tmax_{ra}^{pcp} = \min(t \in T \text{ s.t. } \mathbb{P}(Dmax_{ra}^{arr} \geq t) \leq p^{pcp})$
  end for
end for
return $tmax^{dep}, tmax^{arr}, tmax^{pcp}, propPairs$

Let us again consider a simple example, to illustrate the workings of Algorithm 3.

Example 9.
Let us again consider the example problem defined in Example 1, to illustrate the workings of Algorithm 3. The probability distributions of delay are discretized using $h^{prob} = 1$ and $p^{dcp} = 0.001$, as in Example 4. For the algorithm, we take propagation cutoff point $p^{pcp} = 0.05$. This means that only connections with a probability of propagated delay higher than 5% are considered in the model. First we sort the rotations by std, and get:

$$sortedList = [r_1, r_2, r_3, r_4, r_5, r_6]$$

We loop over $sortedList$, starting with r_1. Rotation r_1 can be operated by all aircraft, so we loop over $[a_1, a_2, a_3]$. By line 10 we get:

$$maxMean_{a_1} \leftarrow std_{r_1} = 0$$

There are no rotations in $sortedList$ before r_1, lines 11 to 21 are skipped. This means that we have $tmax^{dep}_{r_1a_1} = std_{r_1} = 0$.
Rotation r_1 contains two flights, f_1 and f_2, so in line 22 to 31 we calculate:

$$\begin{aligned}
tmax^{arr}_{r_1a_1} &\leftarrow \max(std_{f_2}, tmax^{dep}_{r_1a_1} + dur^{max}_{f_1} + \delta^{r}_{s_{a_1}}) + dur^{max}_{f_2} + \delta^{0}_{s_{a_1}} \\
&= \max(80, 0 + 99 + 30) + 105 + 25 = 259 \\
Dmax^{arr}_{r_1a_1} &\leftarrow \max(maxMean_{r_1a_1} + D_{f_1} + sta_{f_1} - std_{f_1} + \delta^{r}_{s_a}, std_{f_2}) + D_{f_2} + sta_{f_2} - std_{f_2} + \delta^{0}_{s_a} \\
&= \max(0 + D_{f_1} + 45 - 0 + 30, 80) + D_{f_2} + 125 - 80 + 25 \\
&= \max(D_{f_1} + 75, 80) + D_{f_2} + 70 \\
meanProp_{r_1,a_1} &\leftarrow \mathbb{E}[Dmax^{arr}_{ra}] \\
&= \mathbb{E}[\max(D_{f_1} + 75, 80) + D_{f_2} + 70] = 158 \\
tmax^{pcp}_{r_1a_1} &\leftarrow \min(t \in T \; s.t \; \mathbb{P}(Dmax^{arr}_{ra} \geq t) \leq p^{pcp}) \\
&= \min(t \in T \; s.t \; \mathbb{P}(\max(D_{f_1} + 75, 80) + D_{f_2} + 70 \geq t) \leq 0.05) = 182
\end{aligned}$$

Similarly, we find the values for the pairs $(r_1, a_2), (r_1, a_3)$, as well as all the pairs with r_2 and r_3, since they have no feasible preceding rotations. For these values, see Table 4.1.

Next, rotation r_4 is considered. This rotation is a maintenance block for aircraft a_2, so we loop over $[a_2]$. By line 10 we get:

$$maxMean_{a_2} \leftarrow std_{r_4} = 205$$

Rotations r_1 and r_2 are feasible to be operated before r_4, and we have $tmax^{pcp}_{r_1a_2} = 182 \leq 205$ and $tmax^{pcp}_{r_2a_2} = 313 > 205$, so by line 12, we only operate lines 13 to 19 for rotation r_2. We get:

$$(r_2, r_4, s_1) \text{ is added to } propPairs$$

$$maxMean_{a_2} \leftarrow \max(205, 250) = 250$$

Rotation r_4 is a maintenance block, so we get:

$$tmax^{dep}_{r_4a_2} \leftarrow \max(205, 460) = 460$$

Then, lines 22 to 31 are operated in the same way as before. Repeating this process for all rotations and aircraft, we get the values as found in Table 4.1. The propagation pairs that are saved by the algorithm are:

$$\begin{aligned}
propPairs = \{&(r_2, r_4, s_1), (r_2, r_5, s_1), (r_2, r_5, s_2), (r_2, r_6, s_1), (r_2, r_6, s_2), \\
&(r_3, r_6, s_1), (r_3, r_6, s_2), (r_4, r_6, s_1), (r_4, r_7, s_1)\}
\end{aligned}$$

Figure 4.1 shows the propagation pairs as arrows between rotation blocks.

Rotation r	Aircraft a	$tmax_{ra}^{dep}$	$tmax_{ra}^{arr}$	$tmax_{ra}^{pcp}$	$meanProp_{ra}$
r_1	a_1	0	259	182	158
r_1	a_2	0	259	182	158
r_1	a_3	0	269	190	164
r_2	a_1	5	460	313	250
r_2	a_2	5	460	313	250
r_2	a_3	5	470	322	259
r_3	a_1	10	416	306	274
r_3	a_2	10	416	306	274
r_3	a_3	10	426	314	280
r_4	a_2	460	530	383	320
r_5	a_1	330	709	515	468
r_5	a_2	330	709	515	468
r_5	a_3	330	724	539	490
r_6	a_1	410	757	534	510
r_6	a_2	410	757	558	519
r_6	a_3	410	762	539	515
r_7	a_1	350	609	542	524
r_7	a_2	470	729	542	524
r_7	a_3	350	614	547	529

Table 4.1: Values of $tmax_{ra}^{dep}$, $tmax_{ra}^{arr}$ and $tmax_{ra}^{pcp}$ in minutes for the example problem, found using Algorithm 3.

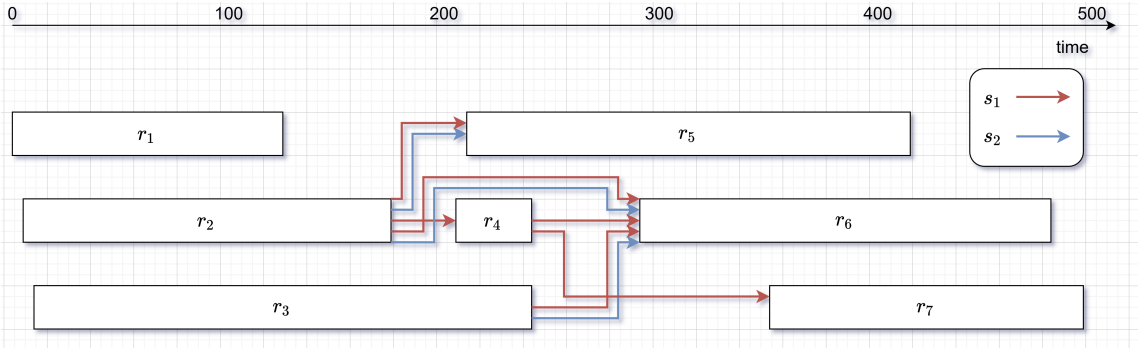

Figure 4.1: Visualisation of the propagation pairs $propPairs$, found using Algorithm 3. The arrows correspond to feasible connections that have a significant probability of propagating delay, if assigned to the same aircraft. The color of the arrows corresponds to the aircraft subtype.

Using the values found using Algorithm 3, we can easily define all the possible departure and arrival state options for every rotation, which will be described in the next section.

4.4. Network creation

This section describes the creation of the delay propagation network, using the results of Algorithm 3.

First, we define a constant step size h^{step} in minutes, which we use for the discrete departure and arrival state nodes of the rotations.

4.4.1. Nodes

Using the values found using pre-processing Algorithm 3, we can create the nodes of the network. For every rotation $r \in \mathcal{R}$ and subtype $s \in \mathcal{S}_r$, we define:

$$N_{rs}^{dep} := \{(std_r, r, s)\} \cup \left\{(t, r, s) \quad \text{s.t. } t \equiv 0 \mod h^{step}, std_r < t \leq \max_{a \in \mathcal{A}_s} tmax_{ra}^{dep} + h^{step}/2\right\} \tag{4.6}$$

$$N_{rs}^{arr} := \left\{(sta_r + \delta_{s_a}^0, r, s)\right\} \cup \left\{(t, r, s) \quad \text{s.t. } t \equiv 0 \mod h^{step}, \right.$$
$$\left. sta_r + \delta_{s_a}^0 < t \leq \min\left(\max_{a \in \mathcal{A}_s} tmax_{ra}^{arr}, \max_{\substack{q \text{ s.t.} \\ (r,q,s) \in propPairs}} \max_{a \in \mathcal{A}_s} t_{qa}^{dep} + h^{step}\right) + h^{step}/2\right\} \tag{4.7}$$

$$N^{dep} := \bigcup_{r \in \mathcal{R}, s \in \mathcal{S}_r} N_{rs}^{dep} \tag{4.8}$$

$$N^{arr} := \bigcup_{r \in \mathcal{R}, s \in \mathcal{S}_r} N_{rs}^{arr} \tag{4.9}$$

$$N := N^{dep} \cup N^{arr} \tag{4.10}$$

Firstly, regardless of the step size h^{step}, for every rotation $r \in \mathcal{R}$ and subtype $s \in \mathcal{S}_r$, a node (std_r, r, s) is created, corresponding to the scheduled time of departure, which is added to N_{rs}^{dep}. Similarly the nodes $(sta_r + \delta_{s_a}^0, r, s)$ are created and added corresponding N_{rs}^{arr}. These nodes are the earliest possible departure and arrival states to consider, since a rotation is not allowed to depart before its scheduled time, and two rotations are only assigned to the same aircraft if the connection is feasible, meaning it is not necessary to consider earlier arrival times. We can simply round earlier arrival time states up to the $sta_r + \delta_s^0$ states.

Note that for the other state nodes, the timesteps are multiples of h^{step}, so they are globally rounded, which will make it easier to define edges between rotations. Departure state nodes are created for all globally rounded possible time steps, as found by Algorithm 3. For the arrival states, we take a minimum, to reduce unnecessary states. If for a rotation the maximum timestep of the departure states of its propagation pairs is later than the maximum arrival state timestep, this means all states between these timesteps will result in a reserve aircraft being used, if this pair is connected. Therefore, we only need to create a single state that represents these reserve aircraft states. Also note that if a rotation has no outgoing propagation pairs, only one arrival state node is created, corresponding to the sta of the rotation.

Example 10.
Using Definitions (4.6) and (4.7), we can determine the state nodes for our example problem, defined in Example 1. For this example, let us take a stepsize of $h^{step} = 15$ minutes. First, let us consider rotation r_1. We have $tmax_{r_1a_1}^{dep} = tmax_{r_1a_2}^{dep} = tmax_{r_1a_3}^{dep} = 0$, so we get:

$$N_{r_1,s_1}^{dep} = \{(0, r_1, s_1)\}$$
$$N_{r_1,s_2}^{dep} = \{(0, r_1, s_2)\}$$

Rotation r_1 has no outgoing propagation pair for s_1 and s_2, so we get:

$$N_{r_1,s_1}^{arr} = \{(150, r_1, s_1)\}$$
$$N_{r_1,s_2}^{arr} = \{(155, r_1, s_2)\}$$

For rotation r_2, we have $tmax^{dep}_{r_2a_1} = tmax^{dep}_{r_2a_2} = tmax^{dep}_{r_2a_3} = 5$, so we get:

$$N^{dep}_{r_2,s_1} = \{(5, r_2, s_1)\}$$
$$N^{dep}_{r_2,s_2} = \{(5, r_2, s_2)\}$$

Rotations r_2 has three outgoing pairs for s_1, and two outgoing pairs for s_2, where:

$$\max_{\substack{q \text{ s.t.} \\ (r_2,q,s_1)\in propPairs}} \max_{a\in\mathcal{A}_{s_1}} t^{dep}_{qa} = \max(t^{dep}_{r_4a_2}, t^{dep}_{r_5a_1}, t^{dep}_{r_5a_2}, t^{dep}_{r_6a_1}, t^{dep}_{r_6a_2}) = \max(460, 330, 330, 410, 410) = 460$$

$$\max_{\substack{q \text{ s.t.} \\ (r_2,q,s_2)\in propPairs}} \max_{a\in\mathcal{A}_{s_2}} t^{dep}_{qa} = \max(t^{dep}_{r_5a_3}, t^{dep}_{r_6a_3}) = \max(330, 410) = 410$$

$tmax^{arr}_{r_2a_1} = tmax^{arr}_{r_2a_2} = 460$ *and* $tmax^{arr}_{r_2a_3} = 470$, *so we get:*

$$\begin{aligned} N^{arr}_{r_2,s_1} = \{&(200, r_2, s_1), (210, r_2, s_1), (225, r_2, s_1), (240, r_2, s_1), (255, r_2, s_1), (270, r_2, s_1), (285, r_2, s_1),\\ &(300, r_2, s_1), (315, r_2, s_1), (330, r_2, s_1), (345, r_2, s_1), (360, r_2, s_1), (375, r_2, s_1), (390, r_2, s_1),\\ &(405, r_2, s_1), (420, r_2, s_1), (435, r_2, s_1), (450, r_2, s_1), (465, r_2, s_1), \}\\ N^{arr}_{r_2,s_2} = \{&(205, r_2, s_2), (210, r_2, s_2), (225, r_2, s_2), (240, r_2, s_2), (255, r_2, s_2), (270, r_2, s_2), (285, r_2, s_2),\\ &(300, r_2, s_2), (315, r_2, s_2), (330, r_2, s_2), (345, r_2, s_2), (360, r_2, s_2), (375, r_2, s_2), (390, r_2, s_2),\\ &(405, r_2, s_2), (420, r_2, s_2)\} \end{aligned}$$

Similarly, nodes are created for the other rotations. All nodes created for this problem can be found in Tables 4.2 and 4.3.

Rotation r	Subtype s	Timesteps t, s.t. $(t, r, s) \in N^{dep}_{r,s}$
r_1	s_1	0
r_1	s_2	0
r_2	s_1	5
r_2	s_2	5
r_3	s_1	10
r_3	s_2	10
r_4	s_1	205, 210, 225, 240, 255, 270, 285, 300, 315, 330, 345, 360, 375, 390, 405, 420, 435, 450, 465
r_5	s_1	210, 225, 240, 255, 270, 285, 300, 315, 330
r_5	s_2	210, 225, 240, 255, 270, 285, 300, 315, 330
r_6	s_1	290, 300, 315, 330, 345, 360, 375, 390, 405
r_6	s_2	290, 300, 315, 330, 345, 360, 375, 390, 405
r_7	s_1	350, 360, 375, 390, 405, 420, 435, 450, 465
r_7	s_2	350

Table 4.2: All departure state nodes in the Robust Flow Model network for the example problem.

Rotation r	Subtype s	Timesteps t, s.t. $(t, r, s) \in N^{arr}_{r,s}$
r_1	s_1	150
r_1	s_2	155
r_2	s_1	200, 210, 225, 240, 255, 270, 285, 300,315,330, 345, 360, 375, 390, 405, 420, 435, 450, 465
r_2	s_2	205, 210, 225, 240, 255, 270, 285, 300, 315, 330, 345, 360, 375, 390, 405, 420
r_3	s_1	265, 270, 285, 300, 315, 330, 345, 360, 375, 390, 405, 420
r_3	s_2	270, 285, 300, 315, 330, 345, 360, 375, 390, 405, 420
r_4	s_1	275, 285, 300, 315, 330, 345, 360, 375, 390, 405 420, 435, 450, 465, 480
r_5	s_1	440
r_5	s_2	445
r_6	s_1	505
r_6	s_2	510
r_7	s_1	520
r_7	s_2	525

Table 4.3: All arrival state nodes in the Robust Flow Model network for the example problem.

4.4.2. Edges

We need to define two sets of edges, between rotations and within rotations. First, for every rotation $r \in \mathcal{R}$ and subtype $s \in \mathcal{S}_r$, let us define:

$$T^{dep}_{rs} := \left\{ t \quad \text{s.t. } (t, r, s) \in N^{dep}_{rs} \right\} \tag{4.11}$$

$$T^{arr}_{rs} := \{ t \quad \text{s.t. } (t, r, s) \in N^{arr}_{rs} \} \tag{4.12}$$

Now, for every rotation pair with subtype $(r_1, r_2, s) \in propPairs$, let us define the edges between rotations as:

$$E^{prop}_{r_1 r_2 s} := \left\{ ((t, r_1, s), (t, r_2, s)) \quad \forall t \in T^{arr}_{r_1,s} \cap T^{dep}_{r_2 s} \right\} \tag{4.13}$$

$$\cup \left\{ ((t, r_1, s), (std_{r_2}, r_2, s)) \quad \forall t \leq std_{r_2} \in T^{arr}_{r_1,s} \backslash T^{dep}_{r_2 s} \right\} \tag{4.14}$$

$$\cup \left\{ ((t, r_1, s), (std_{r_2}, r_2, s)) \quad \forall t \geq \max(T^{dep}_{r_2 s}), t \in T^{arr}_{r_1 s} \right\} \tag{4.15}$$

The first set of edges (4.13) connects arrival states and departure states of the two rotations in a propagation pair with the same subtype, corresponding to the exact same time steps. The second set of edges (4.14) connects arrival states and departure states of the two rotations in a propagation pair with the same subtype, where the second rotation departs on time. The last set of edges (4.15) corresponds to reserve aircraft edges, where the first rotations causes more than 120 minutes of delay for the second rotation, so a reserve aircraft is used. To be able to use this reserve aircraft set in our objective function, let us define

$$E^{reserve}_{r_1,r_2,s} = \left\{ ((t, r_1, s), (std_{r_2}, r_2, s)) \quad \forall t \geq \max(T^{dep}_{r_2 s}), t \in T^{arr}_{r_1 s} \right\} \qquad \forall (r_1, r_2, s) \in propPairs \tag{4.16}$$

$$E^{reserve} = \bigcup_{(r_1,r_2,s) \in propPairs} E^{reserve}_{r_1,r_2,s} \tag{4.17}$$

For every rotation $r \in \mathcal{R}$, we define the edges within this rotation as:

$$E^{rot}_{rs} := \{ ((t_1, r, s), (t_2, r, s)) \quad \forall t_1 \in T^{dep}_{rs}, t_2 \in T^{arr}_{rs} \\ \text{if } \mathbb{P}\left(t_2 - h^{step} < t^{arr}_r \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } = s \right) > 0 \} \tag{4.18}$$

Note that the probability $\mathbb{P}\left(t_2 - h^{step} < t^{arr}_r \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } = s \right)$ is dependent on the distribution of all flights in the rotation. The calculations that need to be done to find this probability use Equations (3.13) and (3.14), as defined in Section 2.2.

Creating edges within a maintenance rotation is treated as a special case, since a maintenance block has no available delay prediction. This means these block are assumed to have the trivial delay distribution $D_f = 0$, i.e. $\mathbb{P}\left(D_f = 0\right) = 1$. If the duration of this maintenance (including turnaround time) is not an exact multiple of the defined step size h^{step}, using Definition (4.22) will cause departure states corresponding to d minutes of delay to either be rounded up or down to an arrival state corresponding to $d' \neq d$ minutes. Because of the trivial delay distribution, this rounding will be the same for all departure states corresponding to delay. This will cause the delay propagation in these blocks to be inaccurate.

To avoid this inaccuracy, in this case we will create two edges for every departure state node, to both the arrival state nodes corresponding to delay minutes closest to d minutes (rounded up and down). The probability of this departure state will be linearly split over these nodes, based on the distance to the exact amount of delay minutes d of the departure state, using a constraint which we will formulate later. In this way, delay propagation is maintenance blocks will be more accurate.

Besides these edges, we also create edges from and to the sink and source respectively. If a rotation has no propagated delay coming from previously assigned rotation, its departure state corresponding to no delay needs to get a probability of 1 from the source. To achieve this, we define the edges:

$$E_{rs}^{source} := \{(source, (std_r, r, s)) \quad \forall r \in \mathcal{R}, s \in \mathcal{S}_r\} \tag{4.19}$$

If a rotation is not connected to any of its outgoing propagation pairs, we need to let all the probability flow away into the sink, since no delay will be propagated. The delay costs used in the objective value are imposed on edges towards departure state nodes, which will be explained in the next section. For this reason, in the case of no outgoing propagation pairs, probability can flow from these departure states directly to the sink. So we can define the edges to the sink as:

$$E_{rs}^{sink}\{((t, r, s), sink) \quad \forall r \in \mathcal{R}, s \in \mathcal{S}_r, (t, r, s)^{arr} \in N_{rs}^{arr}\} \tag{4.20}$$

Let us define the sets

$$E^{prop} := \bigcup_{r_1, r_2, s \in propPairs} E_{r_1, r_2, s}^{prop} \tag{4.21}$$

$$E^{rot} := \bigcup_{r \in \mathcal{R}, s \in \mathcal{S}_r} E_{r,s}^{rot} \tag{4.22}$$

$$E^{source} := \bigcup_{r \in \mathcal{R}, s \in \mathcal{S}_r} E_{r,s}^{rot} \tag{4.23}$$

$$E^{sink} := \bigcup_{r \in \mathcal{R}, s \in \mathcal{S}_r} E_{r,s}^{rot} \tag{4.24}$$

$$E := E^{prop} \cup E^{rot} \cup E^{source} \cup E^{sink} \tag{4.25}$$

$$E_v^+ := \{(v_1, v_2) \in E^{prop} \cup E^{source} : v_2 = v\} \quad \forall v \in N^{dep} \tag{4.26}$$

$$E_v^+ := \{(v_1, v_2) \in E^{rot} : v_2 = v\} \quad \forall v \in N^{arr} \tag{4.27}$$

$$E_v^- := \{(v_1, v_2) \in E^{rot} \cup E^{sink} : v_1 = v\} \quad \forall v \in N^{dep} \tag{4.28}$$

$$E_v^- := \{(v_1, v_2) \in E^{prop} : v_1 = v\} \quad \forall v \in N^{arr} \tag{4.29}$$

Example 11.
Let us again consider the example defined in Example 1, and use Definitions (4.13), (4.14), (4.15), (4.19) and (4.20) to determine the edges in the model. For the definition of the nodes in the model, see Example 10. First, let us consider the edges between rotations. We create edges for all rotation pairs with subtypes in $propPairs$,

that we found using Algorithm 3. One of the pairs with subtype to consider is (r_4, r_6, s_1)*. We have*

$$\begin{aligned}
T^{dep}_{6,1} &= \{290, 300, 315, 330, 345, 360, 375, 390, 405\} \\
T^{arr}_{4,1} &= \{275, 285, 300, 315, 330, 345, 360, 375, 390, 405, 420, 435, 450, 465, 480\} \\
T^{arr}_{4,1} \cap T^{dep}_{6,1} &= \{300, 315, 330, 345, 360, 375, 390, 405\} \\
\left(T^{arr}_{4,1} \backslash T^{dep}_{6,1}\right)_{\leq std_{r_2}} &= \{275, 285\} \\
\left(T^{arr}_{4,1} \backslash T^{dep}_{6,1}\right)_{\geq std_{r_2}+120} &= \{420, 435, 450, 465, 480\}
\end{aligned}$$

So, we get

$$\begin{aligned}
E^{prop}_{4,6,1} := \{&((300, r_4, s_1), (300, r_6, s_1)), \\
&((315, r_4, s_1), (315, r_6, s_1)), \\
&((330, r_4, s_1), (330, r_6, s_1)), \\
&((345, r_4, s_1), (345, r_6, s_1)), \\
&((360, r_4, s_1), (360, r_6, s_1)), \\
&((375, r_4, s_1), (375, r_6, s_1)), \\
&((390, r_4, s_1), (390, r_6, s_1)), \\
&((405, r_4, s_1), (405, r_6, s_1))\} \\
\cup\ &\{((275, r_4, s_1), (290, r_6, s_1)), \\
&((285, r_4, s_1), (290, r_6, s_1)), \\
\cup\ &\{((420, r_4, s_1), (195, r_6, s_1)), \\
&((435, r_4, s_1), (290, r_6, s_1)), \\
&((450, r_4, s_1), (290, r_6, s_1)), \\
&((465, r_4, s_1), (290, r_6, s_1)), \\
&((480, r_4, s_1), (290, r_6, s_1))\}
\end{aligned}$$

Next, we consider the edges within rotations. For example, rotation 2 *has edges from* $(5, r_2, s_1)$ *to all nodes* $(t, r_2, s_1) \in N^{arr}_{r_2,s_1}$*, as all the corresponding state transitions have a positive probability.*

Rotation 4 *is a maintenance block, which has no delay prediction. This maintenance is scheduled to take 65 minutes plus a turnaround time of 30 minutes, while the state nodes correspond to times globally rounded to 15 minutes. Since 105 is not exactly divisible by 15, this block is treated as a special case, and we will create two edges for every departure state node. For example, the node* $(210, r_4, s_1)^{dep}$ *has a probability of 1 to have arrival state* $(305, r_4, s_1)^{arr}$*, but this node does not exist. The closest arrival state nodes are the nodes* $(300, r_4, s_1)^{arr}$ *and* $(315, r_4, s_1)^{arr}$*, so we will connect the departure node to both these nodes.*

Finally edges from the source and to the sink are created. All edges can be found in Appendix A.3.

The created edges are decision variables in the flow network, which can take any value in the interval $[0, 1]$, corresponding to probabilities. Let these decision variables be defined as:

$$e_{v_1,v_2} \in [0, 1] \qquad \forall (v_1, v_2) \in E \tag{4.30}$$

To achieve conservation of probability, we need to to have flow propagation in every node, which can be achieved through the following constraints:

$$\sum_{(v_1,v_2)\in E^+_v} e_{v_1,v_2} = \sum_{(v_1,v_2)\in E^-_v} e_{v_1,v_2} \qquad \forall v \in N \tag{4.31}$$

The edges within rotations, E^{rot} (4.18), send the probability flowing into the departure state nodes to possible arrival state nodes, according to the delay distributions of the flights within the rotation. By the previously defined Constraints (4.31), there is flow propagation in the departure state nodes. In order to

assign the correct amount of flow to outgoing edges of these nodes, this incoming flow needs to be split over these edges in the correct amounts. To achieve this, constraints can be used that enforce the ratio of flow values between these edges to be consistent with the correct probabilities.

Example 12.
For example, suppose we have a departure state $(t,r,s)^{dep}$ that can transition into three states $(t_1,r,s)^{arr}$, $(t_2,r,s)^{arr}$ and $(t_3,r,s)^{arr}$ with probabilities:

$$\mathbb{P}((t_1,r,s)^{arr}|(t,r,s)^{dep}) = 0.5$$
$$\mathbb{P}((t_2,r,s)^{arr}|(t,r,s)^{dep}) = 0.3$$
$$\mathbb{P}((t_3,r,s)^{arr}|(t,r,s)^{dep}) = 0.2$$

To ensure the incoming flow in node $(t_1,r,s)^{arr}$ is split correctly over the three edges, the ratios need to be consistent with these probabilities. We can use the following constraints:

$$e_{(t,r,s)^{dep},(t_1,r,s)^{arr}} \cdot 0.3 = e_{(t,r,s)^{dep},(t_2,r,s)^{arr}} \cdot 0.5$$
$$e_{(t,r,s)^{dep},(t_2,r,s)^{arr}} \cdot 0.2 = e_{(t,r,s)^{dep},(t_3,r,s)^{arr}} \cdot 0.3$$
$$e_{(t,r,s)^{dep},(t_1,r,s)^{arr}} \cdot 0.2 = e_{(t,r,s)^{dep},(t_3,r,s)^{arr}} \cdot 0.5$$

Note that the third constraint in the above example is implied by the first two constraints, meaning we can omit this constraint. In fact, if we have n possible arrival states a departure state can transition into, we can arbitrarily choose one such state, and relate the edge towards this state node to all other edges. So, in general, let us define the following probability splitting constraints:

$$\begin{aligned}
&\forall r \in \mathcal{R}, s \in \mathcal{S}_r, t^{dep} \in T^{dep}_{rs} : \\
&\text{Pick an arbitrary } t_0 \in T^{arr}_{rs}, \text{ such that } \mathbb{P}((t_0,r,s)^{arr}|(t^{dep},r,s)^{dep}) > 0 \\
&\forall t_i \in T^{dep}_{rs} \backslash \{t_0\} : \\
&e_{(t^{dep},r,s)^{dep},(t_i,r,s)^{arr}} \cdot \mathbb{P}\big((t_0,r,s)^{arr}|(t^{dep},r,s)^{dep}\big) \\
&\qquad = e_{(t^{dep},r,s)^{dep},(t_0,r,s)^{arr}} \cdot \mathbb{P}\big((t_i,r,s)^{arr}|(t^{dep},r,s)^{dep}\big)
\end{aligned} \tag{4.32}$$

The values of the probabilities $\mathbb{P}\big((t^{arr},r,s)^{arr}|(t^{dep},r,s)^{dep}\big)$ can again be calculated using Equations 3.13 and 3.14, in preprocessing. By these equations, the following holds for a rotation containing two flights, f_1 and f_2, if assigned to an aircraft of subtype $s_1 \in \mathcal{S}_r$:

$$\begin{aligned}
TD_{f_2} &= PD_{f_2} + D_{f_2} \\
&= \max(sta_{f_1} + PD_{f_1} + D_{f_1} + \delta^{f_1}_{s_1} - std_{f_2}, 0) + D_{f_2} \qquad (4.33)\\
& \qquad (4.34)
\end{aligned}$$

Note that if we want to calculate the values of $\mathbb{P}\big((t^{arr},r,s)^{arr}|(t^{dep},r,s)^{dep}\big)$, we are given $PD_{f_1} = t^{dep} - std_{f_1}$ and $s_1 = s$, so we get the following:

$$\Big(TD_{f_2}|(t^{dep},r,s)^{dep}\Big) = \max(sta_{f_1} + t^{dep} - std_{f_1} + D_{f_1} + \delta^{f_1}_{s} - std_{f_2}, 0) + D_{f_2} \tag{4.35}$$

Since we created distinct arrival state nodes using Definition (4.7), with a step size of h^{step}, the following holds for most of the nodes $(t^{arr},r,s)^{arr}$:

$$\begin{aligned}
&\mathbb{P}\big((t^{arr},r,s)^{arr}|(t^{dep},r,s)^{dep}\big) \\
&\quad = \mathbb{P}\big(t^{arr} - h^{step} < sta_{f_2} + \delta^0_s + \Big(TD_{f_2}|(t^{dep},r,s)^{dep}\Big) \leq t^{arr} + h^{step}\big)
\end{aligned} \tag{4.36}$$

In Definition (4.7), we took the minimum of two values as an upper bound for t, which may cause higher values of the distribution of $\big(TD_{f_2}|(t^{dep},r,s)^{dep}\big)$ to not be considered in these probabilities. Therefore,

for the latest arrival state node, we omit the upper bound in the calculation of this probability. So for the latest arrival state node $(t^{max}, r, s)^{arr} \in N^{arr}_{r,s}$, we get:

$$\begin{aligned}&\mathbb{P}\big((t^{max}, r, s)^{arr} | (t^{dep}, r, s)^{dep}\big)\\ &\quad = \mathbb{P}\big(t^{max} - h^{step} < sta_{f_2} + \delta^0_s + \Big(TD_{f_2} | (t^{dep}, r, s)^{dep}\Big)\big)\end{aligned} \tag{4.37}$$

Similarly, we need to round all probability of a negative arrival delay to the first arrival state node. Additionally, since the difference in time between the first and second arrival state node is not necessarily equal to h^{step}, we need to adjust the bounds for these nodes, so they do not overlap. For the first arrival state node $(t^0, r, s)^{arr}$ and the second arrival state nodes $(t^1, r, s)^{arr}$, we get:

$$\begin{aligned}&\mathbb{P}\big((t^0, r, s)^{arr} | (t^{dep}, r, s)^{dep}\big)\\ &\quad = \mathbb{P}\big(sta_{f_2} + \delta^0_s + \Big(TD_{f_2} | (t^{dep}, r, s)^{dep} \leq \min(t^0 + h^{step}, t^1 - h^{step})\Big)\big)\end{aligned} \tag{4.38}$$

$$\begin{aligned}&\mathbb{P}\big((t^1, r, s)^{arr} | (t^{dep}, r, s)^{dep}\big)\\ &\quad = \mathbb{P}\big(\min(t^0 + h^{step}, t^1 - h^{step}) < sta_{f_2} + \delta^0_s + \Big(TD_{f_2} | (t^{dep}, r, s)^{dep} \leq t^1 + h^{step}\Big)\big)\end{aligned} \tag{4.39}$$

There are two special cases to these equations. If a departure state node is connected to only one arrival state node, naturally we send all probability from the departure state node to this arrival state node. If a departure state node is connected to two arrival state nodes, Equation (4.38) holds, but for the second arrival state node we get a combination of Equation (4.39) and (4.37):

$$\begin{aligned}&\mathbb{P}\big((t^1, r, s)^{arr} | (t^{dep}, r, s)^{dep}\big)\\ &\quad = \mathbb{P}\big(\min(t^0 + h^{step}, t^1 - h^{step}) < sta_{f_2} + \delta^0_s + \Big(TD_{f_2} | (t^{dep}, r, s)^{dep}\Big)\big)\end{aligned} \tag{4.40}$$

The values of $\mathbb{P}\big((t^{arr}, r, s)^{arr} | (t^{dep}, r, s)^{dep}\big)$ are calculated for every edge $\big((t^{arr}, r, s)^{arr}, (t^{dep}, r, s)^{dep}\big) \in E^{rot}$, in preprocessing. These values are then used to define the probability splitting constraints (4.32).

4.4.3. Delay costs

To impose delay costs in the objective value of the model, we need to add costs to some of the created edges. We are given a cost function $c^{delay}_f(t)$ for every flight, which returns the costs of delaying flight f by t minutes. Since the nodes of our network represent departure and arrival states of rotations, and not of flights, we can not use this cost function directly. The easiest way to assign delay costs is to calculate the expected delay costs for both flights in a rotation, given an departure state, and assign this cost to all edges flowing into this departure state node.

To calculate this cost, Equations (3.2), (3.1) and (3.15) can be used. For an departure state node $(t, r, s) \in N^{dep}$, where r contains flights f_1 and f_2, we take $PD_{f_1} = t - std_{f_1}$, such that $\mathbb{P}(PD_{f_1} = t - std_{f_1}) = 1$. Then Equations (3.2) and (3.1) are used to calculate the probability distributions of TD_{f_1} and TD_{f_2}. Using these distributions, the expected delay costs of flights f_1 and f_2 can be determined, using Equation (3.15). The sum of these two values will be imposed as expected delay cost of departure state node (t, r, s), by adding this cost to all edges flowing into this state node.

For all departure nodes $(t, r, s) \in N^{dep}$, this process is performed in preprocessing, and the costs are saved. To access these values in the model, define a function $c^{delaynode}(t, r, s)$, which returns this value for every node $(t, r, s) \in N^{dep}$.

Example 13.
We calculate the cost of the node $(0, r_1, s_1) \in N^{dep}$ *in our example problem, as defined in Example 1. Rotation* r_1 *contains two flights,* f_1 *and* f_2*. We have* $std_{f_1} = 0$*, so we take* $PD_{f_1} = 0 - std_{f_1} = 0$*. The calculations of* TD_{f_1} *and* TD_{f_2} *if* PD_{f_1} *and rotation* r_1 *is assigned an aircraft of subtype* s_1*, and the calculations of the expected delay costs are already performed in Example 6 and Example 7. So, the value that will be imposed to the node* $(0, r_1, s_1)$ *will be equal to* $c^{delaynode}(0, r_1, s_1) = 100.77 + 137.68 = 238.45$*. All incoming edges for this node in the model network will have a cost of* 238.45*.*

4.4.4. Deactivating edges

In order to use the defined network to calculate the delay cost in an optimization model, we need activate the right edges of the network, based on the selected assignments of aircraft to rotations in the model. To achieve this, let us define some more variables:

$$noIncPair_{r,s} \in \{0,1\} \qquad \forall r \in \mathcal{R}, s \in \mathcal{S}_r \tag{4.41}$$

$$noOutPair_{r,s} \in \{0,1\} \qquad \forall r \in \mathcal{R}, s \in \mathcal{S}_r \tag{4.42}$$

$$pair_{r_1,r_2,s} \in \{0,1\} \qquad \forall (r_1,r_2,s) \in propPairs \tag{4.43}$$

The variables $noIncPair_{r,s}$ and $noOutPair_{r,s}$ equal 1 if rotation r is assigned to an aircraft of subtype s, and respectively no incoming or no outgoing propagation pair is assigned to the same aircraft, else they equal 0. This means that if $noIncPair_{r,s} = 1$ for some $r \in \mathcal{R}, s \in \mathcal{S}_r$, rotation r has no probability of propagated delay in the model. Similarly, if $noOutPair_{r,s} = 1$, rotation r has no chance of propagating delay to the next rotation in the schedule, in the model. Note that if rotation r is not assigned an aircraft of subtype s, these variables always equal 0.

If two rotations $r_1, r_2 \in \mathcal{R}$ are assigned to be consecutively operated on the same aircraft of subtype s, and $(r_1, r_2, s) \in propPairs$, the variable $pair_{r_1,r_2,s}$ equals 1, else it equals 0.

Note that these rotations need to be consecutively connected, meaning that if for a propagation pair $(r_1, r_2, s) \in propPairs$, there is no rotation assigned to be operated on the same aircraft between rotations r_1 and r_2. For example, if $propPairs$ contains the propagation pairs (r_1, r_2, s), (r_2, r_3, s) and (r_1, r_3, s), and all three rotations r_1, r_2 and r_3 are assigned to the same aircraft of subtype s, only the edges considering the pairs (r_1, r_2, s) and (r_2, r_3, s) should be activated. To achieve this, let us define the sets:

$$propPairs^{betw}_{r_1,r_2,s} := \{(r_1, r_3, s) \in propPairs \text{ s.t. } sta_{r_3} + \delta^0_s \leq std_{r_2}\} \tag{4.44}$$

$$propPairs^{out}_{r,s} := \{(r_1, r_2, s') \in propPairs \text{ s.t. } r_1 = r, s' = s\} \tag{4.45}$$

$$propPairs^{inc}_{r,s} := \{(r_1, r_2, s) \in propPairs \text{ s.t. } r_2 = r, s' = s\} \tag{4.46}$$

Now we can enforce these variables $noIncPair_{r,s}$, $noOutPair_{r,s}$ and $pair_{r_1,r_2,s}$ to have the correct value using the following constraints:

$$pair_{r_1,r_2,s} + \sum_{(r_1,r_3,s) \in propPairs^{betw}_{r_1,r_2,s}} pair_{r_1,r_3,s} \geq X_{r_1,a} + X_{r_2,a} - 1$$

$$\forall (r_1, r_2, s) \in propPairs, a \in \mathcal{A}_{r_1} \cap \mathcal{A}_{r_2} \cap \mathcal{A}_s \tag{4.47}$$

$$noOutPair_{r,s} + \sum_{(r_1,r_2,s') \in propPairs^{out}_{r,s}} pair_{r_1,r_2,s')} = \sum_{a \in \mathcal{A}_r \cap \mathcal{A}_s} X_{r,a}$$

$$\forall r \in \mathcal{R}, s \in \mathcal{S}_s \tag{4.48}$$

$$noIncPair_{r,s} + \sum_{(r_1,r_2,s') \in propPairs^{inc}_{r,s}} pair_{r_1,r_2,s')} = \sum_{a \in \mathcal{A}_r \cap \mathcal{A}_s} X_{r,a}$$

$$\forall r \in \mathcal{R}, s \in \mathcal{S}_s \tag{4.49}$$

Constraint (4.47) sets the $pair$ variables to 1 if assigned consecutively on the same aircraft of the right subtype. Constraint (4.48) ensures that precisely one of the variables $noOutPair$ and $pair$ is selected to equal 1 for the assigned aircraft subtype, and none for the other subtypes. Constraint (4.49) does the same thing as Constraint (4.48), for the variables $noIncPair$ and $pair$.

Example 14.
Let us consider our example problem, defined in Example 1. As found in Example 9, we have propagation pairs:

$$\begin{aligned} propPairs = \{&(r_2, r_4, s_1), (r_2, r_5, s_1), (r_2, r_5, s_2), (r_2, r_6, s_1), (r_2, r_6, s_2), \\ &(r_3, r_6, s_1), (r_3, r_6, s_2), (r_4, r_6, s_1), (r_4, r_7, s_1)\} \end{aligned}$$

Note that we have propagation pairs $(r_2, r_4, s_1), (r_4, r_6, s_1)$ and (r_2, r_6, s_1). This means, using Definition

(4.44), we have $propPairs^{between}_{r_2,r_6,s_1} = \{(r_2, r_4, s_1)\}$. *For all other propagation pairs* (r_i, r_j, s), *there is no feasible propagation pair in between the pair, so we get an empty set* $propPair^{between}_{r_i,r_j,s}$. *We get the following set of constaints:*

$$
\begin{aligned}
pair_{r_2,r_4,s_1} &\geq X_{r_2,a_2} + X_{r_4,a_2} - 1 \\
pair_{r_2,r_5,s_1} &\geq X_{r_2,a_1} + X_{r_5,a_1} - 1 \\
pair_{r_2,r_5,s_1} &\geq X_{r_2,a_2} + X_{r_5,a_2} - 1 \\
pair_{r_2,r_5,s_2} &\geq X_{r_2,a_3} + X_{r_5,a_3} - 1 \\
pair_{r_2,r_6,s_1} + pair_{r_2,r_4,s_1} &\geq X_{r_2,a_1} + X_{r_6,a_1} - 1 \\
pair_{r_2,r_6,s_1} + pair_{r_2,r_4,s_1} &\geq X_{r_2,a_2} + X_{r_6,a_2} - 1 \\
pair_{r_2,r_6,s_2} &\geq X_{r_2,a_3} + X_{r_6,a_3} - 1 \\
pair_{r_3,r_6,s_1} &\geq X_{r_3,a_1} + X_{r_6,a_1} - 1 \\
pair_{r_3,r_6,s_1} &\geq X_{r_3,a_2} + X_{r_6,a_2} - 1 \\
pair_{r_3,r_6,s_2} &\geq X_{r_3,a_3} + X_{r_6,a_3} - 1 \\
pair_{r_4,r_6,s_1} &\geq X_{r_4,a_2} + X_{r_6,a_2} - 1 \\
pair_{r_4,r_7,s_1} &\geq X_{r_4,a_2} + X_{r_7,a_2} - 1 \\
noOutPair_{r_1,s_1} &= X_{r_1,a_1} + X_{r_1,a_2} \\
noOutPair_{r_1,s_2} &= X_{r_1,a_3} \\
noOutPair_{r_2,s_1} + pair_{r_2,r_4,s_1} + pair_{r_2,r_5,s_1} + pair_{r_2,r_6,s_1} &= X_{r_2,a_1} + X_{r_2,a_2} \\
noOutPair_{r_2,s_2} + pair_{r_2,r_5,s_2} + pair_{r_2,r_6,s_2} &= X_{r_2,a_3} \\
noOutPair_{r_3,s_1} + pair_{r_2,r_6,s_1} &= X_{r_3,a_1} + X_{r_3,a_2} \\
noOutPair_{r_3,s_2} + pair_{r_2,r_6,s_2} &= X_{r_3,a_3} \\
noOutPair_{r_4,s_1} + pair_{r_4,r_6,s_1} + pair_{r_4,r_7,s_1} &= X_{r_4,a_2} \\
noOutPair_{r_5,s_1} &= X_{r_5,a_1} + X_{r_5,a_2} \\
noOutPair_{r_5,s_2} &= X_{r_5,a_3} \\
noOutPair_{r_6,s_1} &= X_{r_6,a_1} + X_{r_6,a_2} \\
noOutPair_{r_6,s_2} &= X_{r_6,a_3} \\
noOutPair_{r_7,s_1} &= X_{r_7,a_1} + X_{r_7,a_2} \\
noOutPair_{r_7,s_2} &= X_{r_7,a_3} \\
noIncPair_{r_1,s_1} &= X_{r_1,a_1} + X_{r_1,a_2} \\
noIncPair_{r_1,s_2} &= X_{r_1,a_3} \\
noIncPair_{r_2,s_1} &= X_{r_2,a_1} + X_{r_2,a_2} \\
noIncPair_{r_2,s_2} &= X_{r_2,a_3} \\
noIncPair_{r_3,s_1} &= X_{r_3,a_1} + X_{r_3,a_2} \\
noIncPair_{r_3,s_2} &= X_{r_3,a_3} \\
noIncPair_{r_4,s_1} + pair_{r_2,r_4,s_1} &= X_{r_4,a_2} \\
noIncPair_{r_5,s_1} + pair_{r_2,r_5,s_1} &= X_{r_5,a_1} + X_{r_5,a_2} \\
noIncPair_{r_5,s_2} + pair_{r_2,r_5,s_2} &= X_{r_5,a_3} \\
noIncPair_{r_6,s_1} + pair_{r_2,r_6,s_1} + pair_{r_4,r_6,s_1} + pair_{r_3,r_6,s_1} &= X_{r_6,a_1} + X_{r_6,a_2} \\
noIncPair_{r_6,s_2} + pair_{r_2,r_6,s_2} + pair_{r_3,r_6,s_2} &= X_{r_6,a_3} \\
noIncPair_{r_7,s_1} + pair_{r_4,r_7,s_1} &= X_{r_7,a_1} + X_{r_7,a_2} \\
noIncPair_{r_7,s_2} &= X_{r_7,a_3}
\end{aligned}
$$

These variables can now be used to deactivate edges that do not correspond to the assignment selected by the decision variables of the model, using the constraints:

$$e_{source,(std_r,r,s)} = noIncPair_{r,s} \qquad \forall r \in \mathcal{R}, s \in \mathcal{S}_r \tag{4.50}$$

$$\sum_{(v,sink)\in E_{rs}^{sink}} e_{v,sink} \leq noOutPair_{r,s} \qquad \forall r \in \mathcal{R}, s \in \mathcal{S}_r \tag{4.51}$$

$$\sum_{(v^{arr},v^{dep})\in E^{p}rop_{r_1,r_2,s}} e_{v^{arr},v^{dep}} \leq pair_{r_1,r_2,s} \qquad \forall r_1, r_2, s \in propPairs \tag{4.52}$$

By Constraints (4.50), edges from the source are set to 1 if the corresponding rotation is assigned an aircraft of the corresponding subtype, and this rotation-subtype combination has no incoming propagation pair. Constraints (4.51) set edges to the sink to 0 if $noOutPair_{r,s}$ is 0, meaning no probability should flow away into the sink if a propagation pair is selected. Constraints (4.52) ensure that probability can only flow to rotation states corresponding to a selected rotation pair.

4.5. Model formulation

4.5.1. Mathematical notation

This model uses the same notation used in the problem formulation in Section 2.3.4. Also, the notation used in the definitions in Sections 4.2, 4.3 and 4.4 will be used, which is summarized here:

Sets

N^{dep}	Set of nodes $(t,r,s)^{dep} \quad \in \bigcup_{r\in\mathcal{R},s\in\mathcal{S}_r} N^{dep}_{rs}$
N^{arr}	Set of nodes $(t,r,s)^{arr} \quad \in \bigcup_{r\in\mathcal{R},s\in\mathcal{S}_r} N^{arr}_{rs}$
N	Set of nodes $(t,r,s) \quad \in N^{dep} \cup N^{arr}$
T^{dep}_{rs}	Set of timesteps for which departure state nodes are created
T^{arr}_{rs}	Set of timesteps for which arrival state nodes are created
E^{prop}	Set of edges $(v^{arr}, v^{dep}) \quad \in \bigcup_{(r_1,r_2,s)\in propPairs} E^{prop}_{r_1r_2s}$
E^{rot}	Set of edges $(v^{dep}, v^{arr}) \quad \in \bigcup_{r\in\mathcal{R},s\in\mathcal{S}_r} E^{rot}_{rs}$
E^{source}	Set of edges $(source, (std_r, r, s)) \quad \forall r \in \mathcal{R}, s \in \mathcal{S}_r$
E^{sink}	Set of edges $(v, sink) \quad \forall v \in N^{arr}$
E	$E^{prop} \cup E^{rot} \cup E^{source} \cup E^{sink}$
E^+_v	$\subseteq E$, set of all incoming edges for any node $v \in N$
E^-_v	$\subseteq E$, set of all outgoing edges for any node $v \in N$
$E^+_{v,s}$	$\subseteq E$, set of all incoming edges for any node $v \in N$, corresponding to subtype $s \in \mathcal{S}$
$E^-_{v,s}$	$\subseteq E$, set of all outgoing edges for any node $v \in N$, corresponding to subtype $s \in \mathcal{S}$
$propPairs$	Set of all propagation pairs r_1, r_2, s
$propPairs^{betw}_{r_1,r_2,s}$	$\subseteq propPairs$, set of propagation pairs r_1, r_3, s that may be assigned between rotations r_1 and r_2 on an aircraft of subtype s
$propPairs^{out}_{r,s}$	$\subseteq propPairs$, set of outgoing propagation pairs for rotation r on an aircraft of subtype s
$propPairs^{inc}_{r,s}$	$\subseteq propPairs$, set of incoming propagation pairs for rotation r on an aircraft of subtype s
RS^{ovlp}_s	Set of sets of rotations $r \in \mathcal{R}$ that overlap if assigned to an aircraft of subtype $s \in \mathcal{S}$

Decision Variables

e_{v_1,v_2}	$\in [0,1] \quad \forall (v_1,v_2) \in E$
$X_{r,a}$	$\in \{0,1\} \quad \forall r \in \mathcal{R}, a \in \mathcal{A}_r$
$pair_{r_1,r_2,s}$	$\in \{0,1\} \quad \forall (r_1,r_2,s) \in propPairs$
$noIncPair_{r,s}$	$\in \{0,1\} \quad \forall r \in \mathcal{R}, s \in \mathcal{S}_r$
$noOutPair_{r,s}$	$\in \{0,1\} \quad \forall r \in \mathcal{R}, s \in \mathcal{S}_r$

Cost functions

$c^{assign}(r,a)$	Assignment cost of assigning rotation $r \in \mathcal{R}$ to aircraft $a \in \mathcal{A}_r$
$c^{delaynode}(t,r,s)$	Expected delay costs of rotation $r \in \mathcal{R}$, if its first flight departs at timestep $t \in T^{dep}_{rs}$, on an aircraft of subtype $s \in \mathcal{S}_r$

4.5.2. Formulation

$$\underset{e,\ X}{\text{minimize}} \quad \sum_{r\in\mathcal{R},a\in\mathcal{A}} c^{assign}(r,a)X_{r,a} \tag{4.53}$$

$$+ \sum_{v_{dep}\in N^{dep}} \sum_{(v,v^{dep})\in E^{+}_{v^{dep}}} c^{delaynode}(v^{dep})\cdot e_{v,v^{dep}} \tag{4.54}$$

subject to

$$\sum_{a\in\mathcal{A}} X_{ra} = 1 \qquad \forall r\in\mathcal{R} \tag{4.55}$$

$$\sum_{r\in rs,\ \text{s.t. } a\in\mathcal{A}_r} X_{r,a} \le 1 \qquad \forall a\in\mathcal{A}, rs\in RS^{ovlp}_{s_a} \tag{4.56}$$

$$\sum_{(v',v)\in E^{+}_{(t,r,s),s}} e_{v',v} - \sum_{(v,v')\in E^{-}_{(t,r,s),s}} e_{v,v'} = 0 \qquad \forall (t,r,s)\in N \tag{4.57}$$

$$pair_{r_1,r_2,s} + \sum_{(r_1,r_3,s)\in propPairs^{betw}_{r_1,r_2,s}} pair_{r_1,r_3,s} \ge X_{r_1,a} + X_{r_2,a} - 1 \qquad \forall (r_1,r_2,s)\in propPairs,$$

$$a\in\mathcal{A}_s\cap\mathcal{A}_{r_1}\cap\mathcal{A}_{r_2} \tag{4.58}$$

$$noOutPair_{r,s} + \sum_{(r_1,r_2,s)\in propPairs^{out}_{r,s}} pair_{r_1,r_2,s} = \sum_{a\in\mathcal{A}_s\cap\mathcal{A}_r} X_{r,a} \qquad \forall r\in\mathcal{R}, s\in\mathcal{S}_r \tag{4.59}$$

$$noIncPair_{r,s} + \sum_{(r_1,r_2,s)\in propPairs^{in}_{r,s}} pair_{r_1,r_2,s} = \sum_{a\in\mathcal{A}_s\cap\mathcal{A}_r} X_{r,a} \qquad \forall r\in\mathcal{R}, s\in\mathcal{S}_r \tag{4.60}$$

$$\sum_{(v^{arr},v^{dep})\in E^{prop}_{r_1r_2^s}} e_{v^{arr},v^{dep}} \le pair_{r_1,r_2,s} \qquad \forall (r_1,r_2,s)\in propPairs \tag{4.61}$$

$$e_{source,(std_r,r,s)} = noIncPair_{r,s} \qquad \forall r\in\mathcal{R}, s\in\mathcal{S}_r \tag{4.62}$$

$$\sum_{(v,sink)\in E^{sink}_{rs}} e_{v,sink} \le noOutPair_{r,s} \qquad \forall r\in\mathcal{R}, s\in\mathcal{S}_r \tag{4.63}$$

$$\mathbb{P}((t_1,r,s)^{arr}|(t_0,r,s)^{dep})\cdot e_{(t_0,r,s)^{dep},(t_2,r,s)^{arr}} = \mathbb{P}((t_2,r,s)^{arr}|(t_0,r,s)^{dep})\cdot e_{(t_0,r,s)^{dep},(t_1,r,s)^{arr}} \qquad \forall r\in\mathcal{R}, (t_0,r,t)^{in}\in D^{in}_f, (f,t_1)^{out}\in D^{out}_f, \text{ where } (f,t_2)^{out}\in D^{out}_f \text{ arbitrary} \tag{4.64}$$

$$e_{v_1,v_2}\in[0,1] \qquad \forall (v_1,v_2)\in E \tag{4.65}$$

$$X_{r,a}\in\{0,1\} \qquad \forall r\in\mathcal{R}, a\in\mathcal{A}_r \tag{4.66}$$

$$pair_{r_1,r_2,s}\in\{0,1\} \qquad \forall (r_1,r_2,s)\in propPairs \tag{4.67}$$

$$noIncPair_{r,s}\in\{0,1\} \qquad \forall r\in\mathcal{R}, s\in\mathcal{S}_r \tag{4.68}$$

$$noOutPair_{r,s}\in\{0,1\} \qquad \forall r\in\mathcal{R}, s\in\mathcal{S}_r \tag{4.69}$$

4.5.3. Parameter influence

The size and shape of the network, as well as the accuracy of the solutions are dependent on the parameters used to create the network. In total, we defined four parameters: h^{prob}, p^{dcp}, p^{pcp} and h^{step}. In this section, the influence of these four parameters on the network is evaluated. The decision variables corresponding to the network are the edges E, as well as the variables $noIncPair$, $noOutPair$ and $pair$. Since the number of variables $|noIncPair|$ and $|noOutPair|$ are not dependent on the size and shape of the network, an the number of variables $|pair|$ is simply equal to $|pair| = |propPairs|$, we will omit these variables in the analysis, and only consider the number of edges. First the number of edges is given as a function of the number of state nodes N^{dep} and N^{arr}, as well as the number of propagation pairs $propPairs$. Then, conclusions are drawn about the influence of the four parameters. Finally, the accuracy of the solutions of the network is evaluated.

Number of Edges
The network contains four types of edges; edges between rotations E^{prop}, edges within rotations E^{rot}, edges from the source to departure state nodes E^{source} and edges from arrival state nodes to the sink E^{sink}. Let us consider the influence of the number of state nodes and number of propagation pairs on these types of edges separately.

- Edges between rotations E^{prop} connect all arrival state nodes to exactly one departure state nodes of another rotation, if these rotations are a propagation pair. So the total number of edges between rotations is equal to:
$$|E^{prop}| = \sum_{(r,q,s)\in propPairs} |N^{arr}_{r,s}| \tag{4.70}$$

- Edges within rotations E^{rot} connect departure state nodes to arrival state nodes of the same rotation. One departure state node is connected to many arrival state nodes, corresponding to possible state transitions, based on the delay distributions of the flights in the rotation. By Definition (4.18), a departure state node (t_1, r, s) is connected to an arrival state node (t_2, r, s) if $\mathbb{P}\left(t_2 - h^{step} < t^{arr}_r \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } = s\right) > 0$. So the number of edges within rotations is equal to:
$$\begin{aligned} |E^{rot}| = \sum_{r\in\mathcal{R}} \sum_{s\in\mathcal{S}_r} \sum_{(t_1,r,s)\in N^{dep}_{rs}} |\{(t_2, r, s) \in N^{arr}_{rs} \\ \text{s.t. } \mathbb{P}\left(t_2 - h^{step} < t^{arr}_r \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } s\right) > 0\}| \end{aligned} \tag{4.71}$$

  We can further evaluate this equation, if we assume the distribution to satisfy the following (natural) relation for $t^{arr}_{r,0} < t^{arr}_{r,1}$:

$$\begin{aligned} &\mathbb{P}\left(t_2 - h^{step} < t^{arr}_{r,0} \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } s\right) > 0 \\ &\wedge \mathbb{P}\left(t_2 - h^{step} < t^{arr}_{r,1} \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } s\right) > 0 \Longrightarrow \\ &\mathbb{P}\left(t_2 - h^{step} < t^{arr}_{r,2} \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } s\right) > 0 \qquad \forall t^{arr}_{r,0} \leq t^{arr}_{r,2} \leq t^{arr}_{r,1} \end{aligned}$$

  Let $t^{min}_{(t_1,r,s)}$ be equal to the minimum value, such that
  $\mathbb{P}\left(t_2 - h^{step} < t^{min}_{(t_1,r,s)} \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } s\right) > 0$.
  Similarly, let $t^{max}_{(t_1,r,s)}$ be equal to the maximum value, such that
  $\mathbb{P}\left(t_2 - h^{step} < t^{max}_{(t_1,r,s)} \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } s\right) > 0$. Then, we get:

$$\begin{aligned} |E^{rot}| &= \sum_{r\in\mathcal{R}} \sum_{s\in\mathcal{S}_r} \sum_{(t_1,r,s)\in N^{dep}_{rs}} |\{(t_2, r, s) \in N^{arr}_{rs} \\ &\qquad \text{s.t. } \mathbb{P}\left(t_2 - h^{step} < t^{arr}_r \leq t_2 + h^{step} | t^{dep}_r = t_1, \text{ subtype } s\right) > 0\}| && (4.72) \\ &\approx \sum_{r\in\mathcal{R}} \sum_{s\in\mathcal{S}_r} \sum_{(t_1,r,s)\in N^{dep}_{rs}} \frac{\min(t^{max}_{(t_1,r,s)}, \max(T^{arr}_{rs})) - \max(t^{min}_{(t_1,r,s)}, \min(T^{arr}_{rs}))}{h^{step}} && (4.73) \\ &= \sum_{r\in\mathcal{R}} \sum_{s\in\mathcal{S}_r} \sum_{(t_1,r,s)\in N^{dep}_{rs}} \frac{\min(t^{max}_{(t_1,r,s)}, \max(T^{arr}_{rs})) - \max(t^{min}_{(t_1,r,s)}, sta_{rs} + \delta^0_s)}{h^{step}} && (4.74) \end{aligned}$$

  For real-life instances of the Robust Tail Assignment problem, considering sensible parameters, the terms of the sum in this equation would roughly be equal the some constant $0 < c < 1$ times the amount of arrival state nodes of the rotation-subtype pair (r, s).

- Every feasible rotation-subtype combination has exactly one edge from the source to the corresponding no-delay departure state, see Equation (4.19). Therefore, the number of edges from source in the network is equal to:

$$|E^{source}| = \sum_{r \in \mathcal{R}} |\mathcal{S}_r| \tag{4.75}$$

- For every arrival state node in the network, exactly one edge to the sink is created, see Equation (4.20). Therefore, the total number of edges to the sink is equal to:

$$|E^{sink}| = \sum_{r \in \mathcal{R}} \sum_{s \in \mathcal{S}_r} |N^{arr}_{rs}| \tag{4.76}$$

Distribution discretizing parameters

The probability step size h^{prob} and distribution cutoff point p^{dcp}, as defined in Section 3.2, influence the discretization and rounding of the delay distributions for every flight in the schedule. The discretization of the delay distribution approximates the actual distribution, where higher values of step size h^{prob} cause a rougher approximation of the distribution. The value p^{dcp} influences the tail of the delay distributions of flights, causing high delay options to be rounded down to a maximum value. A higher value of p^{dcp} causes a lower cutoff point in the tail of the distribution. If we have $h^{step} > h^{prob}$ and $p^{pcp} >> p^{dcp}$ [2], the values h^{prob} and p^{dcp} do not influence the state creation of the model. But these values do influence the accuracy of the probabilities concerning the various possible state transitions, i.e. edges in the model. Also, the accuracy of the delay costs assigned to states in the model are dependent on h^{prob} and p^{dcp}. Lower values of h^{prob} and p^{dcp} cause these probabilities and delay costs to be more accurate, at the cost of more preprocessing time.

Propagation cutoff point

The propagation cutoff point p^{pcp}, as defined in Section 4.3.3, is used in Algorithm 3 and influences the number of edges in the network, and therefore the number of decision variables in the model, in the following way:

By line 31 in Algorithm 3, the values of $tmax^{pcp}_{ra}$ negatively correlate to p^{pcp}, i.e. a lower value of p^{pcp} causes a higher values of $tmax^{pcp}_{ra}$. By line 12, higher values of $tmax^{pcp}_{ra}$ cause more pairs of rotations with subtype to be added to $propPairs$. A higher number of propagation pairs directly causes more edges between rotations E^{prop} and edges to the sink E^{sink} to be created in the network,see Equation (4.70) and (4.76). But, $propPairs$ also influences the number of edges in a different way. By Equation (4.7), the arrival state nodes created for a rotation r and subtype s are dependent on the maximum departure timesteps $tmax^{dep}_{qs}$ of rotations q such that $(r, q, s) \in propPairs$. Also, the values of these maximum departure timesteps $tmax^{dep}_{qs}$ are dependent on the $propPairs$, by line 15-19 in Algorithm 3. This means that the number of departure state nodes are also dependent on the number of $propPairs$. Therefore, an increase of these propagation pairs causes an increase in departure and arrival state nodes. As discovered in Equation (4.70), (4.74) and (4.75), more state nodes means more edges between rotations, within rotations and from the source.

Note that a higher cutoff point causes more propagation pairs to be omitted from the model, decreasing the accuracy of the model.

So, in general, the propagation cutoff point has a negative correlation with the number of edges in the network, as well as the accuracy of the model.

State step size

For the state creation of the model, the step size h^{step} is used. State nodes corresponding to delayed states all correspond to timesteps that are multiples of h^{step}, which directly influences the number of

[2] In theory, the parameter p^{pcp} can have an impact on the number of nodes created in the network. By line 1, 23 and 27 of Algorithm 3, the values of $tmax^{arr}_{ra}$ are dependent on p^{pcp}, meaning these values $tmax^{arr}_{ra}$ increase for decreasing p^{pcp}. But by Definition (4.7), the arrival states nodes are bounded by the departure state nodes of its outgoing propagation pairs, which are bounded in line 18 of Algorithm 3.

nodes in the network. For a rotation r and subtype s, by Definition (4.6) and (4.7), we have:

$$c_{rs}^{dep} := \max_{a \in \mathcal{A}_s} \left(tmax_{ra}^{dep} - std_r \right) \tag{4.77}$$

$$c_{rs}^{arr} := \min \left(\max_{a \in \mathcal{A}_s} tmax_{ra}^{arr}, \max_{\substack{q \text{ s.t.} \\ (r,q,s) \in propPairs}} \max_{a \in \mathcal{A}_s} t_{qa}^{dep} \right) - sta_r - \delta_{s_a}^0 \tag{4.78}$$

$$|N_{rs}^{dep}| \approx 1 + \frac{c_{rs}^{dep} + h^{step}/2}{h^{step}} = \frac{3}{2} + \frac{c_{rs}^{dep}}{h^{step}} \tag{4.79}$$

$$|N_{rs}^{arr}| \approx 1 + \frac{c_{rs}^{arr} + h^{step}/2}{h^{step}} = \frac{3}{2} + \frac{c_{rs}^{arr}}{h^{step}} \tag{4.80}$$

Note that the values of c_{rs}^{dep} and c_{rs}^{arr} do not depend on h^{step}. The exact values for $|N_{rs}^{dep}|$ and $|N_{rs}^{arr}|$ are equal to the derived approximation rounded up or down, depending on the values of std_r, c_{rs}^{dep} and $sta_r, \delta_{s_a}^0, c_{rs}^{arr}$. Using this approximation we can conclude that the number of nodes in the network decreases linearly with the parameter h^{step}. This means doubling h^{step} will approximately half the number of arrival and departure state nodes.

The number of nodes in the network impacts the number of edges in the network. Combining Equation (4.70) and (4.80), we find:

$$|E^{prop}| = \sum_{(r,q,s) \in propPairs} |N_{r,s}^{arr}| \approx \sum_{(r,q,s) \in propPairs} \frac{3}{2} + \frac{c_{rs}^{arr}}{h^{step}} \tag{4.81}$$

Similarly, using Equation (4.76) and (4.80), we get:

$$|E^{sink}| = \sum_{r \in \mathcal{R}} \sum_{s \in \mathcal{S}} |N_{r,s}^{arr}| \approx \sum_{r \in \mathcal{R}} \sum_{s \in \mathcal{S}} \frac{3}{2} + \frac{c_{rs}^{arr}}{h^{step}} \tag{4.82}$$

So, the number of edges between rotations and edges to the sink are decreases linearly with the parameter h^{step}.

Using Equation (4.74) and (4.79), we find that the number of edges within rotations decreases quadratically with the parameter h^{step}. This means halving h^{step} would increase the number of edges within rotations by a factor of 4.

The number of edges from the source is not influenced by the parameter h^{step}.

So, increasing the parameter h^{step} will decrease the number of edges between rotations and to sink linearly, and the number of edges within rotations quadratically.

4.6. Solution method

The model formulation of the Robust Tail Assignment problem, as given in Section 4.5.2 is solved using the Gurobi solver package [16], using a Python environment. This main part of this solver is a linear-programming based branch-and-bound algorithm. This section will describe the workings of this algorithm, and other operations performed by the solver.

4.6.1. Branch-and-Bound

The main algorithm of the solver uses a search tree to find the optimal solution. See Figure 4.2 for an example of a search tree. The algorithm will evaluate nodes of this tree in turn, starting at a root node (the top node), and working its way down. Every node corresponds to a slightly different MIP model. The root node corresponds to the original MIP model. For every other node, the MIP is has the same formulation as the MIP of the parent node, with one added constraint.

Throughout the algorithm, the best solution found thus far is saved, as well as its objective value, called the "incumbent". Initially, the value of incumbent is set to None.

Evaluating nodes

When evaluating a node, the corresponding MIP model is reduced to its linear-programming relaxation, a Linear Programming (LP) model, by lifting all integrality restrictions. In the MIP formulated in

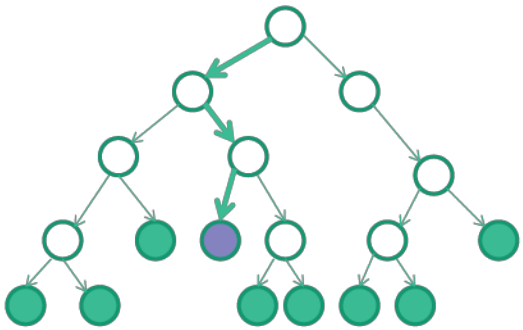

Figure 4.2: Example of a search tree.

Section 4.5.2, this means the binary variables X, $pair$, $noIncPair$ and $noOutPair$ are no longer binary variables, but can take any value in the interval $[0, 1]$. Note that every feasible solution of the MIP model is also a solution of the corresponding LP model. Solving such an LP model is much easier than the corresponding MIP model, the Gurobi solver can solve this LP model very quickly using efficient implementations of the simplex and interior-point methods.

If the LP model has no feasible solutions, the MIP model also contains no feasible solution. In this case the node can be omitted from the search tree. If an optimal solution to the LP model is found, this solution is checked. Three possibilities are considered:

- The solution satisfies all integrality restrictions of the MIP model.
  This solution is also the optimal solution to the corresponding MIP model. In this case the node is considered fathomed, meaning the node does not need further branching. The solution is also a solution to the original MIP in the root node by the definition of the search tree. Therefore, we can update the incumbent value to the minimum of the current incumbent value and the objective value of this solution.

- The solution does not satisfy all integrality restrictions of the MIP model and the objective value is not lower than the current incumbent value. In this case the objective value of the solution is a lower bound on solutions of this MIP model. Since this lower bound is higher than the current value of incumbent, there is no need to further examine this branch, since no better incumbent solutions can be found. The node is fathomed and needs no further branching.

- The solution does not satisfy all integrality restrictions of the MIP model and the objective value is lower than the current incumbent value.
  Since the integrality constraints are not satisfied, the solution is not feasible for the MIP model of this node. In this case the objective value of the solution is a lower bound on solutions of this MIP model. Since the lower bound is lower than the current incumbent value, this node is worth exploring. To further explore the MIP model, a single non-integral variable $Var = k$, $k \notin \mathbb{Z}$ from the solution is picked. The node is branched into two nodes, with the added constraints $Var \geq \overline{k}$ and $Var \leq \underline{k}$, respectively. Here $\overline{k}$ denotes the value of k rounded up, and $\underline{k}$ rounded down. In this way, the search space of the MIP model is cut into two distinct parts.

After evaluating a node and creating branches if necessary, another node is picked to be evaluated, until all nodes are fathomed. At that moment, the best solution found is the optimal solution to the original MIP model. This is the end of the branch-and-bound algorithm.

Note that order in which the nodes are evaluated, as well as the variable selected for branching, have big impact on the run-time of the algorithm. If better choices are made, incumbent solutions are found quicker, and nodes are more often fathomed. The Gurobi package uses many sophisticated techniques to select nodes and variables, as well as several other operations to speed up the algorithm.

4.6.2. Other operations

In this section some of the most important algorithm improving operations are described. More information about these operations can be found on the Gurobi website [16].

Presolve
Before initiating the branch-and-bound algorithm, the model formulation is reduced and tightened by presolve operations. Such operations include deleting constraints implied by other constraints, tightening constraints using integrality if variables, and deleting variables that have implied values. Note that these operations change the MIP model, meaning some of these operations need to be reversed to be able to represent the final solution in terms of the original MIP model.

Cutting planes
When evaluating nodes, after a solution to the LP-relaxation is found, which contains some non-integer variable values that ought to be integers in the MIP model, clever constraints can be added to the model to cut this solution from the LP search space. Now the LP model can be resolved, and the LP solver will find a different optimal solution.

Note that this operation adds constraints to the model, making the LP-relaxations harder to solve. Therefore, such constraints are only added by the Gurobi solver, if they benefit the solving process.

Heuristics
In order to find good incumbent solutions early in the algorithm, heuristics can be used. Heuristics are quick methods to find a feasible solution of a model, with no guarantee of optimality. For example, non-integer variables in the solution of the LP-relaxation can be rounded to integer values and fixed, after which the LP-relaxation can be resolved with the hope of finding an incumbent solution. A good incumbent solution will cause the branch-and-bound algorithm to be able to fathom nodes quicker, speeding up the process. Gurobi employs several heuristics on the nodes of the branch-and-bound algorithm, to find incumbent solutions.

Parallelism
The operations performed at the nodes of the branch-and-bound algorithm are independent of each other, and can be parallelized. A computer contains multiple cores, that Gurobi can control to evaluate separate nodes at the same time, speeding up the solving process.

5
Experiments

The Robust Flow Model, described in Section 4, will be tested using two types of experiments. Firstly, the model is tested using a variety of parameter values, to determine the influence of these parameters on the run-time of the model. Secondly, the model is compared to robust model currently used by KLM, described in Section 2.6, as well as the Non-Robust model, described in Section 2.5, to see which model finds better solutions within a given time-frame. In this section, the setup of these experiments will be explained, the results of the experiments are discussed, and conclusions about the performance of the model are drawn.

5.1. Experimental setup

This section will describe the setup of the experiments. First the setup of the problem schedule and the delay distributions will be described. Then, the setups of the experiments will be given. All experiments are run on a 64-bit laptop PC with an Intel(R) Core(TM) i7-8750H processor with 2.20GHz and 16 GB of RAM memory, running Windows 11. The code is written in Python.

5.1.1. Problem flight schedule

Since this thesis is written on behalf of KLM, the problem used in the testing environment is a flight schedule provided by the KLM. This problem contains flights and maintenance blocks, with all corresponding expected cost functions, as well as an original assignment of aircraft to flights. Since the flight schedule and expected costs are confidential, this information will be omitted from this thesis. In order to be able to show the performance of the different models, the expected costs will be scaled to the expected costs of this original assignment solution. This way, the percentage difference in expected costs between the models can be shown.

The given problem scenario is a flight schedule of three days long, where information about the delay distributions is only provided for the first day. The goal is to consider the expected delay costs for the flights operated on this first day. For the last two days robustness can be ignored, as in a real life scenario the assignments for these days can be re-evaluated the next day, when more information about the expected delays is available. By including these two days in the problem in a non-robust way, a feasible connection between the days is ensured. Also, since the schedule often contains bigger overnight gaps between flights, the chance of delay propagation between different days is usually very small.

Some general information about the problem can be found in Table 5.1, and a figure of the flight schedule can be found in Figure 5.1. Note that the aircraft names and the dates are randomized.

Flights	1630
Rotations	879
Flights on first day	618
Rotations on first day	341
Maintenance blocks	23
Maintenance blocks on first day	14
Aircraft	101
Aircraft Subtypes	5
Average Flight Duration	97.9 min
Minimum Flight Duration	34 min
Maximum Flight Duration	306 min

Table 5.1: General information about the problem used for the experiments.

Figure 5.1: Full schedule used for the experiments, with original assignments. The dates are randomized.

To be able to find the influence of problem size on the run-time of the Robust Flow Model, need to test the model using various problem sizes. The problem is determined by the problem size, i.e. the amount of flights/rotations/aircraft, and the delay distributions of the flights in the schedule. In this analysis, two ways are used to change the problem size.

Firstly, a parameter $sparsity$ is used to determine the sparsity of the schedule. If we use $sparsity = 1$, the full schedule is used. If we take $sparsity = 0.5$, half the rotations in the schedule are deleted from the problem. An example of a schedule with $sparsity = 0.5$ is given in Figure 5.2.

Secondly, a parameter $size$ is used to determine the amount of aircraft in the schedule. If we have $size = 0.5$, half the aircraft are omitted from the problem. To preserve feasibility of the problem, all rotations that were originally assigned to these aircraft are also omitted. An example of a schedule with $size = 0.5$ is given in Figure 5.3.

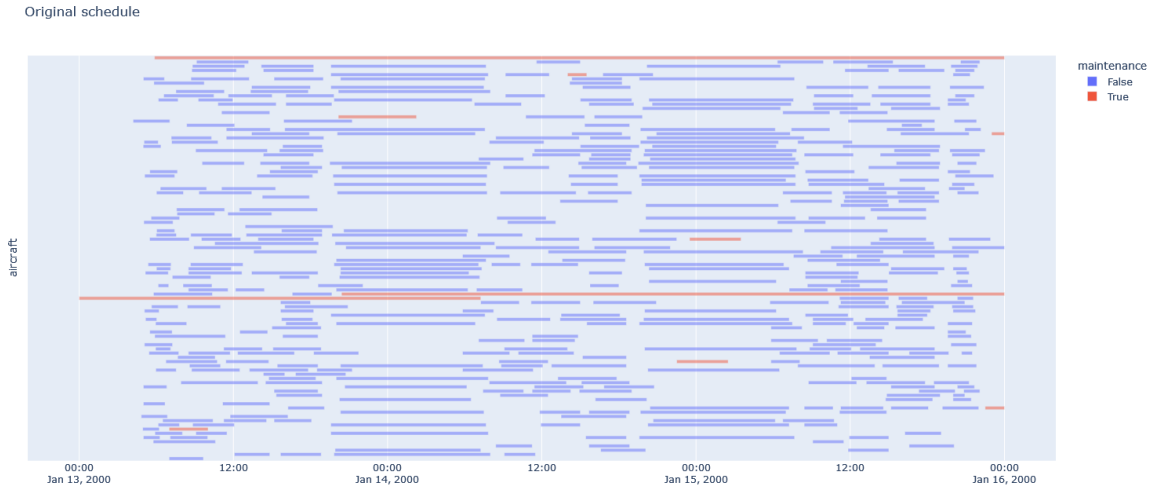

Figure 5.2: Schedule with $sparsity = 0.5$, with original assignments. Half of the rotations are deleted.

Figure 5.3: Schedule with $size = 0.5$, with original assignments. Half the aircraft with corresponding rotations are deleted.

5.1.2. Delay distributions

Besides the flight schedule, the problem also depends on the probability distributions of delay for the flights in the schedule. In practice, these distributions are based on several factors, such as weather and previous data on delays. In this section, some research will be done to figure out what realistic probability distributions of delay look like, after which an way to generate such distributions for the experiments is given.

Research

In order to use realistic probability distributions of delay in our experiments, we do some research into real delay predictions, as given by a KLM delay prediction model. This model gives delay predictions using a percentile description, with a step size of 5%, from 5% to 95%. Also, the 1% and 99% percentiles are given. We denote by $perc_{i\%}$ the value of the $i\%$ percentile. An example plot of delay predictions of all flights in a one day schedule, is given in Figure 5.4.

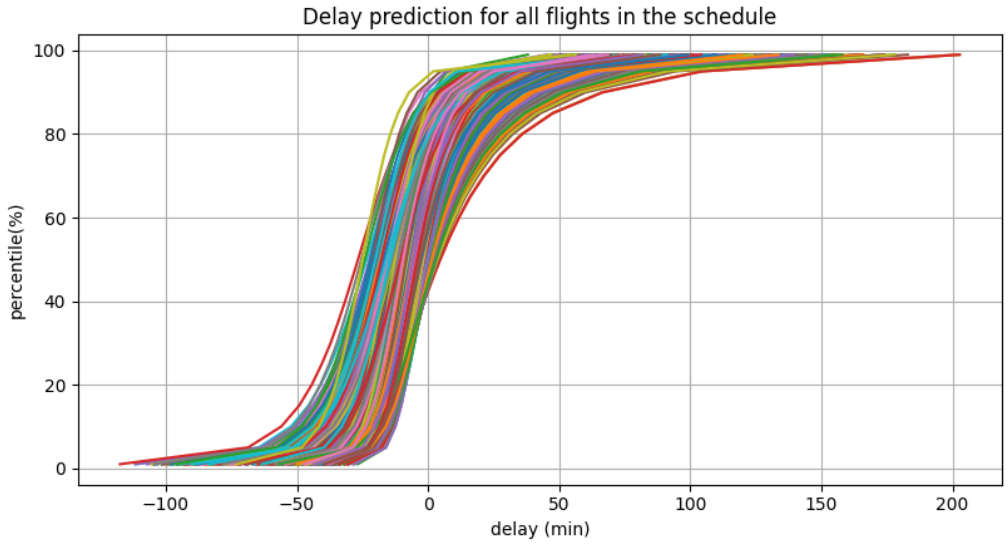

Figure 5.4: Cumulative distribution function of delay predictions by percentile for all flights in the example schedule.

Note that in this figure, we assume a linear relation between the different percentile prediction points, as an approximation of the cumulative distribution function (cdf). Under this assumption, we can find the probability density function (pdf), corresponding to this cdf. Using the piecewise linearity of the cdf, we have that a linear line segment $[(a_x, a_y), (b_x, b_y)]$ in the cdf corresponds to a horizontal line segment $[(a_x, (b_y - a_y)/(b_x - a_x)), (b_x, (b_y - a_y)/(b_x - a_x))]$ in the pdf. In Figure 5.5 the pdf and cdf of the delay prediction for one flight are given.

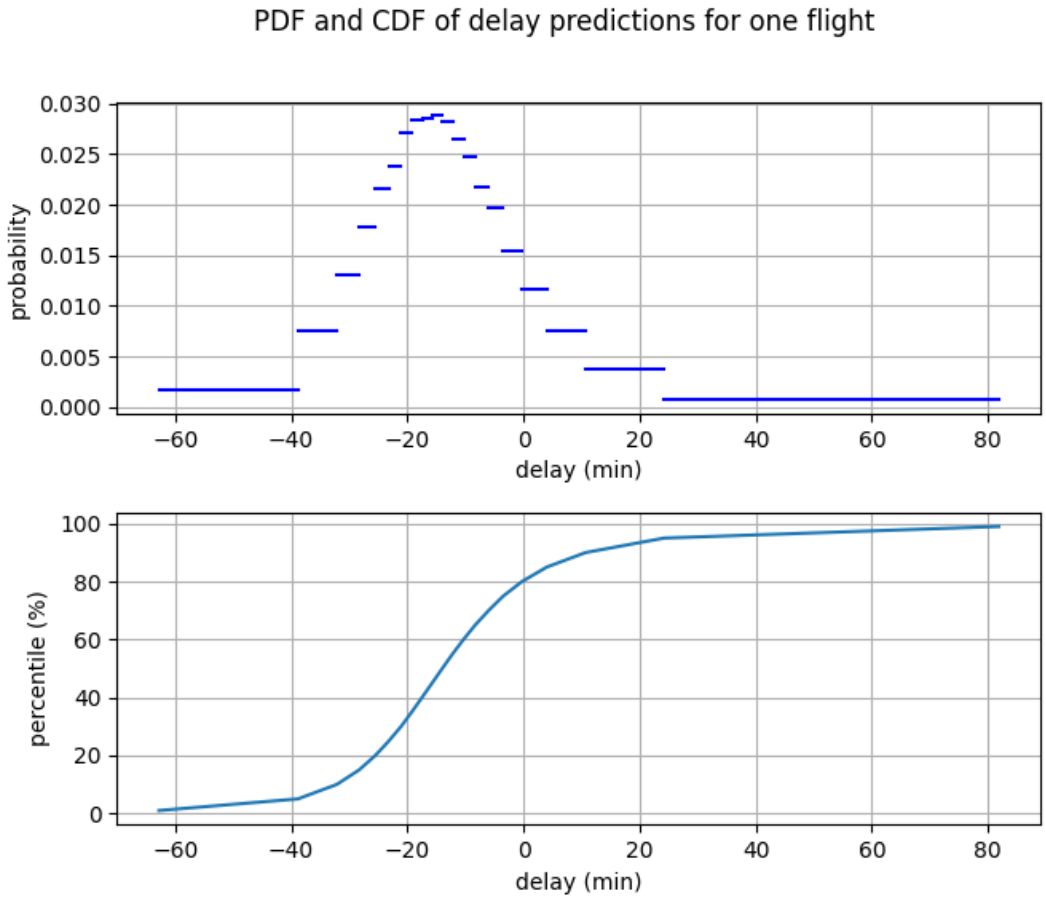

Figure 5.5: Example probability density function (pdf) and cumulative distribution function (cdf) of the delay predictions for a single flight.

Note that the shape of the approximation of the PDF has one mode and is bell-shaped. This is the case for all delay predictions in the schedule. Let us find the mean and variance of the delay distributions for these flights, using the definitions:

$$\mu = \mathbb{E}(X) = \sum_{i=1}^{n} x_i p_i, \tag{5.1}$$

$$\sigma^2 = \mathbb{V}(X) = \sum_{i=1}^{n} (x_i - E(X))^2 p_i, \tag{5.2}$$

where x_i are discrete events of the sample space X, where $|X| = n$, that occur with a probability of p_i. As an estimation, we take the x_i's to be the midpoints between two percentiles, and take as p_i the difference between the percentiles. For the remaining 1% blocks at the ends of the predictions, we take

as x_i the value of the percentile. So, we get:

$$\begin{aligned}\mu &= 0.01perc_{1\%} + 0.04(perc_{5\%} + perc_{1\%})/2 + \sum_{i=1}^{18} 0.05(perc_{5(i+1)\%} + perc_{5i\%})/2 \\ &\quad + 0.04(perc_{99\%} + perc_{95\%})/2 + 0.01perc_{99\%} \\ \sigma^2 &= 0.01(perc_{1\%} - \mu)^2 + 0.04((perc_{5\%} + perc_{1\%})/2 - \mu)^2 \\ &\quad + \sum_{i=1}^{18} 0.05((perc_{5(i+1)\%} + perc_{5i\%})/2 - \mu)^2 \\ &\quad + 0.04((perc_{99\%} + perc_{95\%})/2 - \mu)^2 + 0.01(perc_{99\%} - \mu)^2\end{aligned}$$

If we plot the mean and standard deviation of the delay distributions for every flight in the schedule on both a stormy and calm day, we find the results as shown in Figure 5.6 and 5.7.

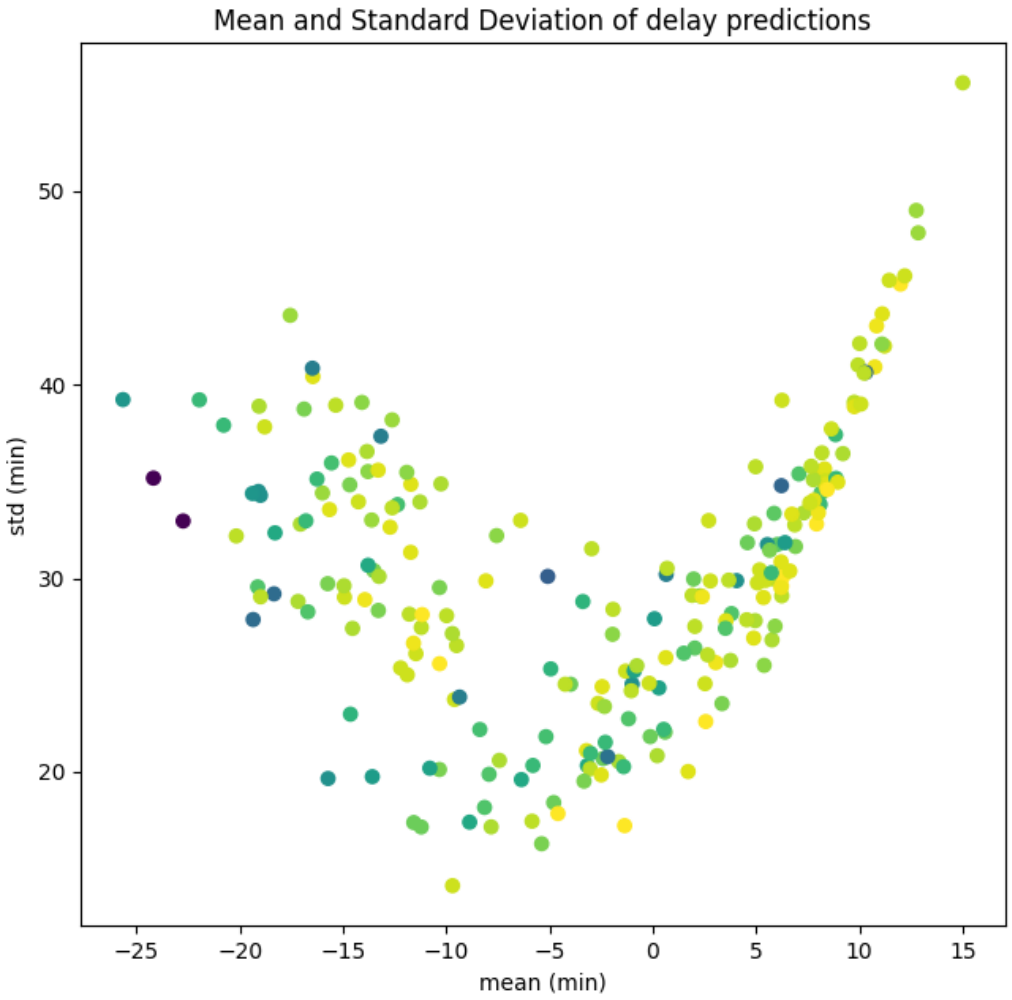

Figure 5.6: Mean and standard deviation of delay predictions for every flight on the first day of the schedule, on a stormy day. The colour of the points corresponds to the length of the flight, where a longer flight has a darker colour.

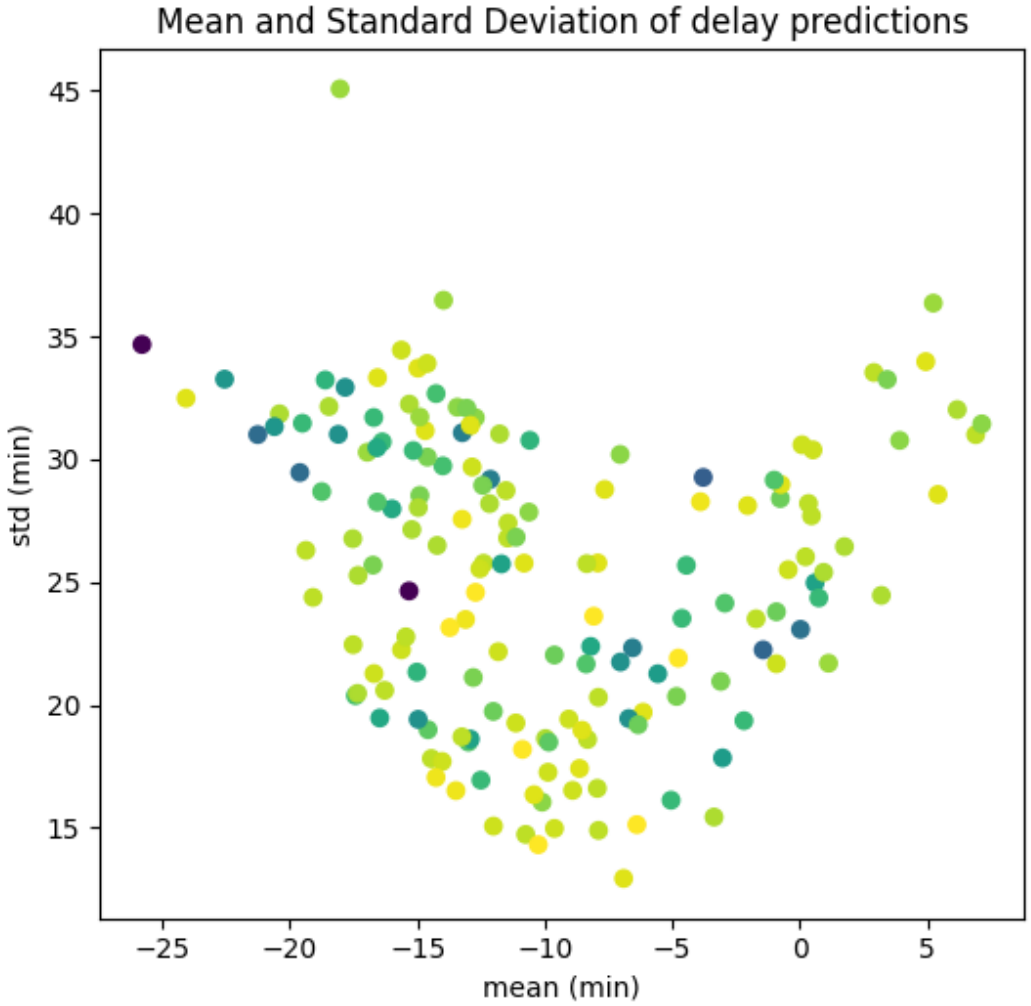

Figure 5.7: Mean and standard deviation of delay predictions for every flight on the first day of the schedule, on a non-stormy day. The colour of the points corresponds to the length of the flight, where a longer flight has a darker colour.

From figure 5.7 we conclude that the delay distributions of non-disrupted flights have a mean between -25 and 10, with most distributions between -20 and 0 minutes of delay. We also see that the standard deviation mostly takes values between 13 and 40, where the standard deviation takes lower values when the mean is close to -8, and and higher values when the mean is further away from this -8.

The means of the delay distributions of this non-disrupted day, have a mean of -10 and a standard deviation of 7.

In Figure 5.6 we see that some flights have a higher expected delay (i.e. higher mean) than in Figure 5.7. We see many means between 0 and 15. These are flights that are expected to be disturbed by a storm. The delay distributions of these disturbed flights seem to follow the relation $\sigma = \frac{25\mu+120}{9}$.

Delay distribution generation

Using this information, we will generate realistic delay distributions to use for the experiments. First, let us address the type of probability distribution to use. The Robust Flow Model is not restricted to any distribution. Based on the analysis performed in the previous section, a continuous unimodal distribution is expected. Also, since there is a physical limitation to the amount of time a flight can be faster than scheduled (minimum flight time), we can assume that there is a lower bound to the

distributions. For these reasons, a shifted Gamma distribution will be used in the experiments. This distribution has a lower bound and no upper bound in its domain, is unimodal and is continuous.

The Gamma distribution has a shape parameter $k > 0$ and a scale parameter $\theta > 0$, and is defined on the domain $(0, \infty)$. It has a mean equal to $\mu = k\theta$ and a variance equal to $\sigma^2 = k\theta^2$. If we shift the Gamma distribution by a factor s, the domain changes to (s, ∞) and the mean will equal $\mu = k\theta + s$, while the variance remains $\sigma^2 = k\theta^2$. Note that the shifted Gamma distribution can be uniquely described by its mean μ, standard deviation σ^2 and lower bound s. Given these values, the shape and scale parameters are equal to:

$$k = \frac{(\mu - s)^2}{\sigma^2} \tag{5.3}$$

$$\theta = \frac{\sigma^2}{\mu} \tag{5.4}$$

What remains is to generate shifted Gamma distributions to use in the experiments. In the experiments, we will test the models using various degrees of disturbances. Therefore, we need to be able to create multiple scenarios of delay predictions, ranging from an undisturbed schedule to a very disturbed schedule. To achieve this, we will use two different pools of distributions to draw samples from, one pool corresponding to non-disrupted flights and one corresponding to disrupted flights. In the experiments, we use a parameter $disruption \in [0, 1]$, which determines the fraction of flights that draw from the disrupted pool. The remainder of flights will draw from the non-disrupted pool. The distributions in the pools will be described using their mean μ, standard deviation σ^2 and lower bound s.

By our research in the previous section, on a non-disrupted day we found that the means of the distributions are distributed with a mean of -10 and a standard deviation of 7. Therefore, for a delay distribution of a non-disrupted flight, we will draw a mean μ using a truncated normal distribution with mean -10 and standard deviation 7. The distribution is truncated at -30 and 10, in order to discard extreme outliers. The standard deviation σ^2 of the non-disrupted delay distribution is drawn uniformly from a interval within two bounds, based on Figure 5.7.

For means between -30 and -12, we draw standard deviations using a uniform distribution, such that

$$-\frac{11\mu + 41}{7} \leq \sigma \leq \frac{9 - 7\mu}{3}.$$

For means between -12 and -5, we draw standard deviations using a uniform distribution, such that

$$13 \leq \sigma \leq 31.$$

For means between -5 and 15, we draw standard deviations using a uniform distribution, such that

$$\frac{5\mu + 64}{3} \leq \sigma \leq \frac{\mu + 67}{2}.$$

By our research in the previous section, we also found that delay distributions of disrupted flights have means and standard deviations that follow the relation $\sigma = \frac{25\mu+120}{9}$. Since these predictions have been evaluated by KLM to underestimate the impact of disturbances, we will draw means that are somewhat higher than we found in this research. So, for the delay distributions of disrupted flights, we will draw means μ using a truncated normal distribution with mean 10 and standard deviation 10. The distribution is truncated at -20 and 40, in order to discard extreme outliers. As standard deviations of the delay distributions for these flights we will use the relation $\sigma = \frac{25\mu+120}{9}$, but we take $\sigma = 20$ as a minimum.

Now that we have a way of generating realistic values μ, σ for non-disrupted and disrupted flights, we can use the corresponding gamma distributions as delay distributions for the flights in the schedule. As lower bound s for these distributions, we take -0.5 times the expected flight time, which seems like a reasonable physical limitation.

5.1.3. Run-time analysis

Let us define the experiments to run, in order to determine the influence of the problem size and shape, as well as the model parameters on the run-time of the Robust Flow Model. In total we defined 7 parameters, which influence the problem and the model, these parameters are

$$sparsity, size, disruption, p^{dcp}, h^{prob}, p^{pcp}, h^{step}.$$

The problem itself is defined by parameters $sparsity, size$ and $disruption$, while the model depends on the parameters $p^{dcp}, h^{prob}, p^{pcp}$ and h^{step}. As concluded in Section 4.5.3, the parameters p^{dcp} and h^{prob} only influence the accuracy of the approximation of the delay costs, at the cost of some preprocessing time. As the main bottleneck of the model is the solving process, we will omit these parameters from our run-time analysis. To be able to see the influence of the 5 remaining parameters separately, three values for every parameter will be taken. For every combination of these values, an experiment will be run. In total, there are $3^5 = 243$ such combinations. Note that we have 3 problem-defining parameters and 2 model-defining parameters. In order to maintain consistency in the experiments, experiments using the same problem-defining parameters will be performed on the exact same problem. This means a total of $3^3 = 27$ problems will be generated, where each such problem will be solved using $2^3 = 9$ different models.

The parameter values used in the experiments are the following:

$$\begin{aligned} sparsity &\in \{0.25, 0.5, 1\} \\ size &\in \{0.25, 0.5, 1\} \\ disruption &\in \{0, 0.5, 1\} \\ p^{dcp} &= 0.001 \\ h^{prob} &= 1 \\ p^{pcp} &\in \{0.025, 0.05, 0.1\} \\ h^{step} &\in \{5, 10, 20\} \end{aligned}$$

In every experiment, the problem is solved using the Robust Flow Model and evaluated using the evaluator. This process is divided into multiple parts, for which the run-time is saved seperately. These parts are:

- **Preprocessing.** In this step, all the relevant sets for the model are created and the probabilities corresponding to the edges are calculated, using Algorithm 2, Algorithm 3 and the equations formulated in Section 4.4.
- **Model creation.** In this step, the model variables are initiated, and the objective function and all constraints are defined using these variables.
- **Optimization.** In this step, the MIP solver Gurobi [16] is used to solve the model. For the optimization part, a time limit of one hour is used.
- **Simulation.** In this step, the solution of the model is simulated using the simulator.

5.1.4. Performance analysis

To analyse the performance of the Robust Flow Model (RFM), the final solution of the model will to be compared to the solutions found by the Non-Robust Tail Assignment model (NRM) and the Benchmark Robust Model (BRM), as well as the original schedule.

To resemble the real-life use case as much a possible, these experiments will be performed using on the full schedule, and using a time limit of 15 minutes. This means we use problems with $sparsity = 1$ and $size = 1$. To determine the effectiveness of RFM on differing levels of expected disruptions, a wider range of $disruption$ values will be used. To be able to find the best parameters settings to use for RFM, a wider range of p^{pcp} and h^{step} values will be used. The parameter values used in these experiments are the following:

$$
\begin{aligned}
sparsity &= 1 \\
size &= 1 \\
disruption &\in \{0, 0.2, 0.4, 0.6, 0.8, 1\} \\
p^{dcp} &= 0.001 \\
h^{prob} &= 1 \\
p^{pcp} &\in \{0.05, 0.1, 0.2, 0.35, 0.5, 0.65\} \\
h^{step} &\in \{10, 20, 40, 60, 90\}
\end{aligned}
$$

For every problem (defined by $disruption$), the problem is solved using the RFM with all possible settings p^{pcp} and h^{step}, as well as using one instance of BRM and NRM. The solutions found by all these models are evaluated using the evaluator, and compared to the originally given schedule.

5.1.5. Expectations

Run-time analysis

In the preprocessing step, Algorithm 3 is used to define the set of propagation pairs $propPairs$. The most time-consuming operation of this algorithm is expected to be the calculation of the propagated delay distributions (line 24/28). This operation is performed once for every feasible rotation-aircraft combination considered for robustness.

Also, to find the probabilities associated with the propagation of probability over edges within rotations E^{rot}, see Equation (4.35) in Section 4.4.2, a similar calculation needs to be performed for every departure state node N^{dep}. After this operation, propagation probabilities for these edges can be calculated. This process is performed for every departure state node, making this process expected to be more time-consuming than Algorithm 3. Since the amount of departure state nodes increases linearly with the amount of rotations considered for robustness, and decreases linearly with the step size h^{step}, we expect the run-time of the preprocessing step to show the same behaviour.

In the model creating step, the variable objects and constraints are created, using the results from the preprocessing step. The most time consuming part is expected to be the creation of the flow propagating constraints (see Constraint 4.57 in the formulation of the Robust Flow Model, Section 4.5.2). To create these constraints, we need to loop over the incoming and outgoing edges for every node in the network. This means every edge in the network needs to be incorporated twice in such constraints. Therefore, the model creating time is expected to increase linearly with the total amount of edges in the network.

Besides this, for a lower value of the step size h^{step}, a departure state node is connected to more arrival state nodes. This means that more edges are captured in a single constraint. So, besides the total amount of edges in the network, the value of h^{step} is expected to affect the model creating time as well. A lower value of h^{step} with the same amount of edges is expected to have shorter model creating time than a higher value of h^{step}.

The run-time of the optimization step is not easy to predict, since it depends on the "difficulty" of the model to solve. The main decision variables are the assignment variables X_{ra}; these variables determine the solution, that all other variables in the model depend on. Therefore, the amount of variables X_{ra} directly impacts the size of the search space, and therefore the difficulty of the problem. But, the real difficulty of the problem lies in the dependency in the solutions. By only considering relevant propagation pairs $propPairs$, this dependency is simplified in the Robust Flow Model. If more $propPairs$ are considered in the model, the model is harder to solve. Another factor of the difficulty of the problem may be the $sparsity$ of the problem. In a more dense schedule, there are more feasibility constraints acting on the assignment variables, which should make it more difficult for the optimizer to navigate through the search space. Therefore, the run-time of the optimization step of the model is expected to be mostly influenced by these factors. Note that the amount of $propPairs$ considered in the model for a specific problem setup is directly impacted by the value of p^{pcp}.

Besides these factors, the step size h^{step} will impact the run-time as well, since its value will directly change the total amount of variables and constraints in the model.

Performance analysis
The performance of RFM depends on several factors. Lower values of h^{step} cause the model to be more accurate in the approximation of the delay costs, but will cause the model to contain more variables as well. Similarly, a lower value of p^{pcp} will cause the model to be more accurate and contain more propagation pairs $|propPairs|$, causing more edges as well. More variables will generally cause longer optimizing times. But, since the main difficulty of the problem lies in the propagation, more propagation pairs will likely have the biggest impact on the optimizing time.

For the full schedule, the optimizer will likely not find optimized results within the given timespan. This means the performance of the model will be a tradeoff between accuracy of the model, and simplicity of the model. A higher accuracy will cause the model to be more difficult to solve, which will decrease the quality of the best solution found by the model in the given timespan. What parameter values will cause the best performance of the model likely depends on the problem size, as well as the expected disruption in the schedule. A higher expected disruption will cause more propagation pairs to be considered in the model, with the same value of p^{pcp}.

In general, the quality of the solution found by the model is expected to mostly depend on the value of p^{pcp}.

5.2. Results

In this section, the general results of the experiments will be given. The results and information for every specific experiment can be found in Appendix A.4.

5.2.1. Run-time

The run-time of the model consists of three parts; preprocessing time, model creation time and optimizing time. In this section the run-time of these parts will be evaluated separately.

Preprocessing time
In the preprocessing step, the propagation pairs $propPairs$ are determined using Algorithm 3, and the sets N^{dep}, N^{arr} are created, as well as all edges E, with their corresponding probabilities (see Equation (4.32)).

Figure 5.8 and 5.9 show the preprocessing time in the plots of the experiments, plotted against the amount of rotations considered for robustness. From these figures, we can conclude that the preprocessing time linearly depends on this amount of robust rotations in the problem, where the slope depends (mostly) on the values of h^{step} and $disruption$.

Model creation time
In the model creating step, the sets from defined in the preprocessing step are used to create all variables and constraints in the model.

Figure 5.10 shows the model creating time in the experiments plotted against the total amount of edges in the model network. In this figure, we can clearly see that the model creating time depends linearly on the amount of edges in the network, where the steepness depends on the step size h^{step}. Do note that the amount of edges also depends on the value of h^{step}.

Optimization process
The optimizing time is the biggest bottleneck in solving the model. For 91 out of the 243 experiments, the optimal value was not found after 60 minutes of optimization. For 6 experiments, not a single feasible solution was found in 60 minutes. But, the optimizing time can still be analysed. To analyse the run-time of the optimization process, we will look at two different values; The first timestep when a feasible solution is found and the timestep when the final solution is proven to be optimal. If in an experiment such a timestep is not reached within the timeframe of 60 minutes, the experiment will be plotted using a cross symbol on the timestep value of 3700 seconds.

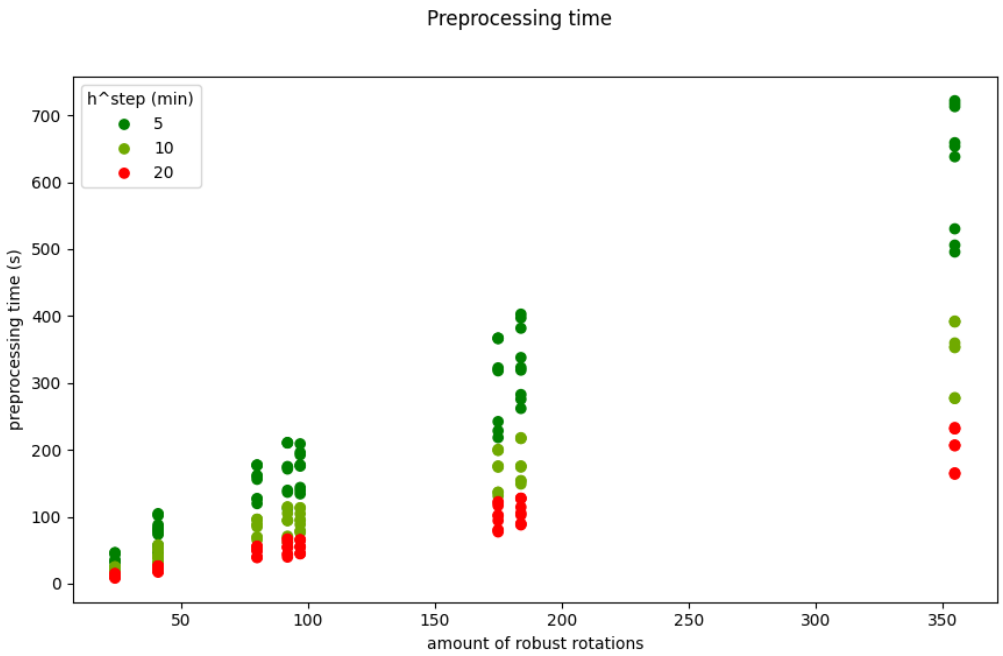

Figure 5.8: Processing time of the experiments, plotted against the amount of rotations in the problem to consider for robustness. The results are grouped by the values of h^{step} used in the model.

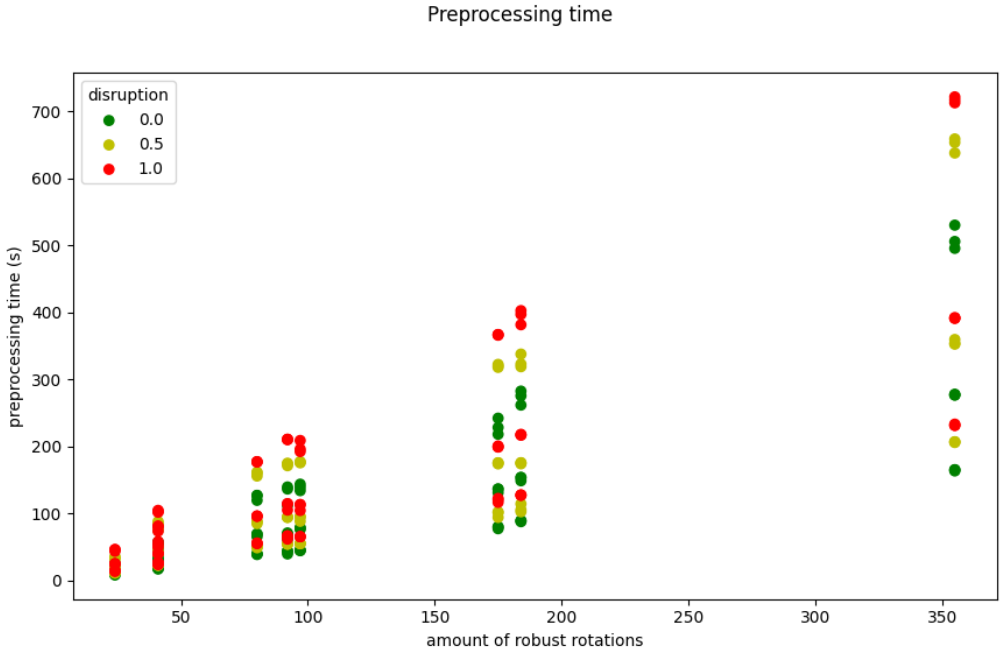

Figure 5.9: Processing time of the experiments, plotted against the amount of rotations in the problem to consider for robustness. The results are grouped by the values of $disruption$ used in the model.

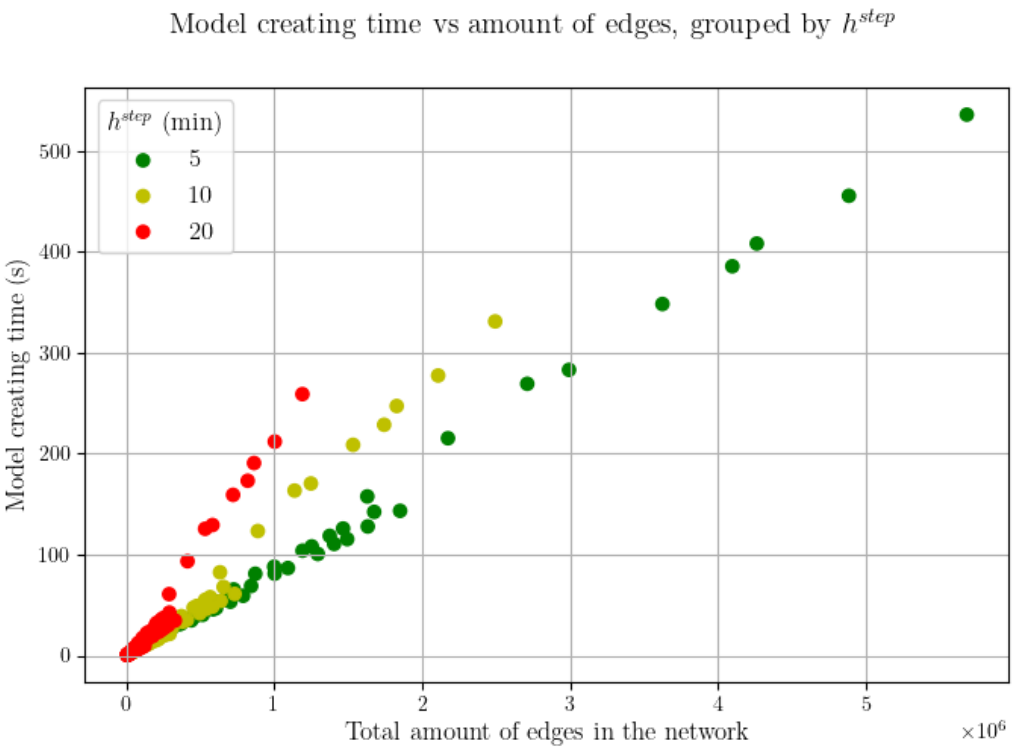

Figure 5.10: Model creating time in the experiments plotted against the amount of edges in the model network. The results are grouped by the value of h^{step}

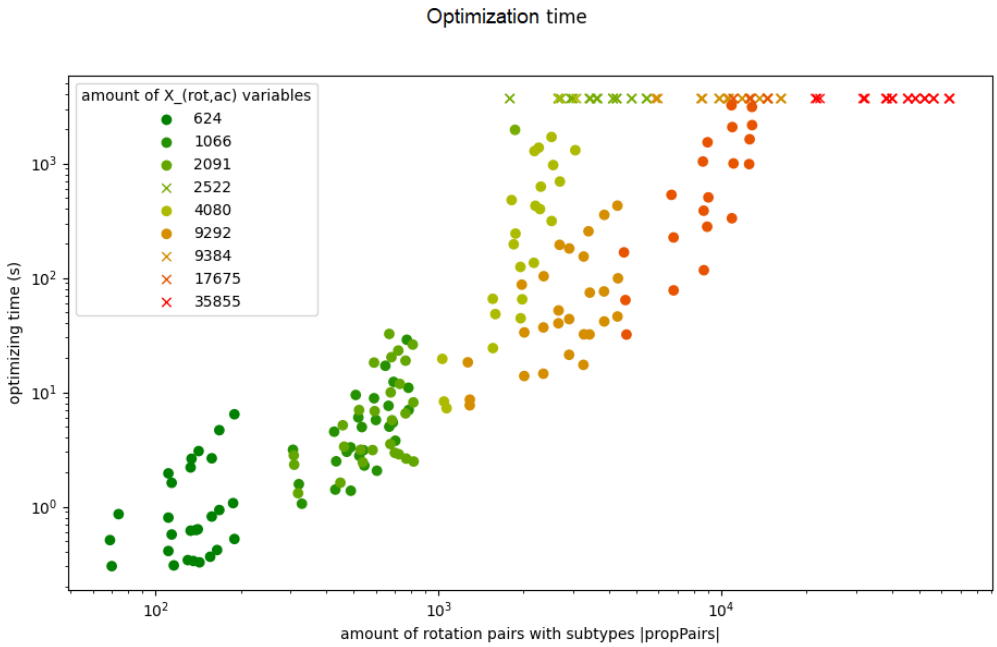

Figure 5.11: Log-log plot of the optimization time in the experiments against the amount of propagation pairs $|propPairs|$. The results are grouped by the amount of assignment variables $X_{rot,ac}$.

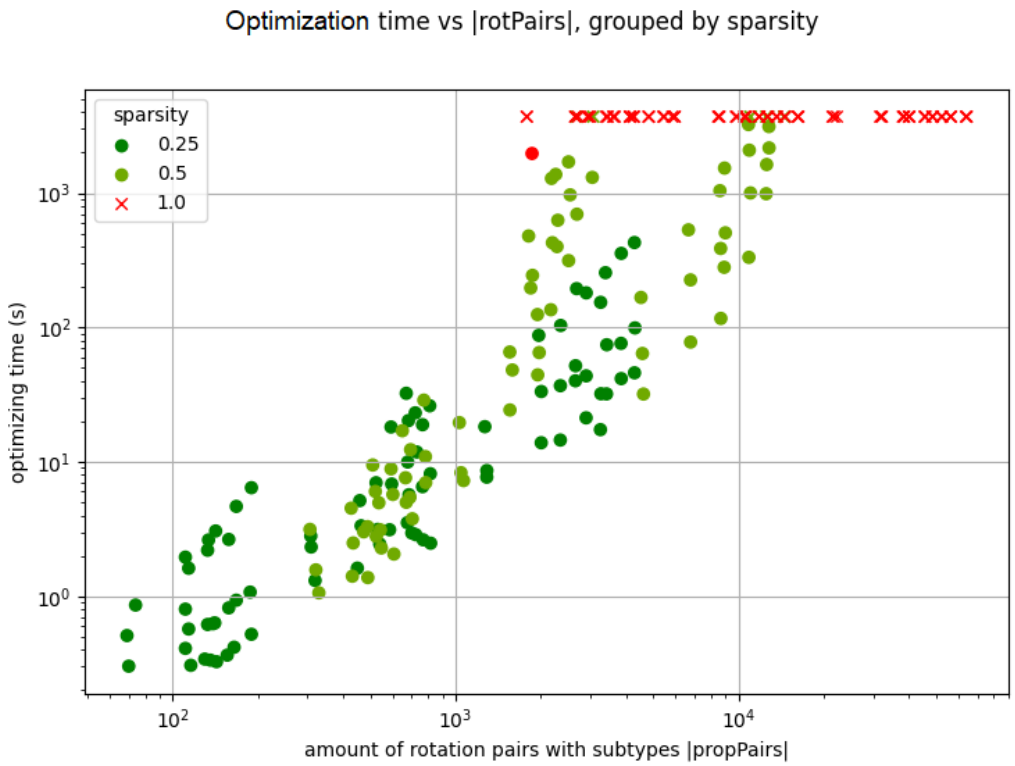

Figure 5.12: Log-log plot of the optimization time in the experiments against the amount of propagation pairs $|propPairs|$. The results are grouped by the sparsity of the problem used in in the experiments.

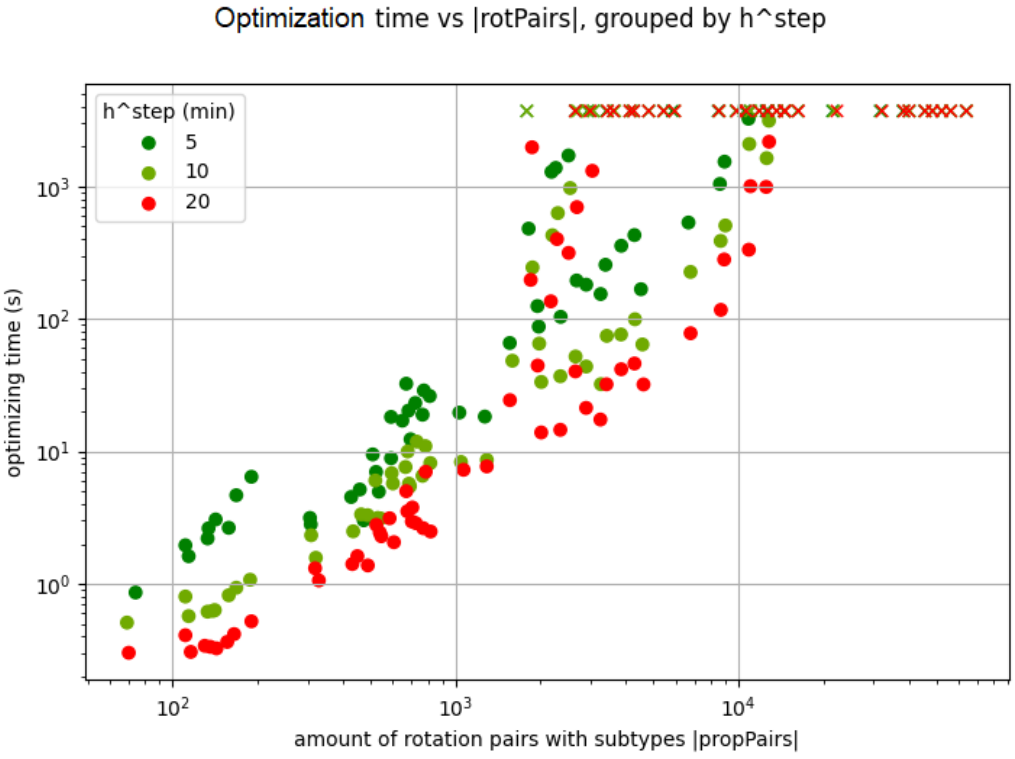

Figure 5.13: Log-log plot of the optimization time in the experiments against the amount of propagation pairs $|propPairs|$. The results are grouped by the value of h^{step} used in the model.

Optimization time

First, let us consider the optimization time, i.e. the timestep when the final solution is proven to be the optimal solution. In section 5.1.5 we expected this time to mostly depend on the amount of propagation pairs $|propPairs|$, the amount of main decision variables $X_{rot,ac}$, the sparsity of the problem, and the step size h^{step}. Figures 5.11, 5.12 and 5.13 show the optimization time in the experiments plotted against the amount of propagation pairs.

From Figure 5.11 and 5.13, we can conclude that for every distinct schedule setup used in the experiments, and for every distinct value of h^{step}, a linear relation in the log-log plot is present between the optimization time and the amount of propagation pairs $|propPairs|$. Using Figure 5.12, we see that the steepness of this linear relation in the log-log plot seems to depend on the sparsity of the problem.

A linear relation in a log-log plot corresponds to a polynomial relation. Thus, for every distinct problem schedule and value of h^{step}, we have:

$$optimizationtime \approx c * |propPairs|^k$$

The steepness of the linear relation in the log-log plot is equal to the power k in this polynomial relation. If we look at specific problem setups $(size, sparsity, disruption)$, we see that for different values of h^{step}, the time-to-optimal-solution shows linear relations with the amount of propagation pairs $propPairs$ in the log-log plot, with similar steepness. As an example, see Figure 5.14. Therefore, we assume the value of the power k is not dependent on the value of h^{step}. Note that for different problem setups, the steepness of this relation in the log-log plot can be different. This means that k is a function of the problem setup. Thus, when solving a specific problem, the value of $|propPairs|$ has a polynomial influence on the optimization time of the problem.

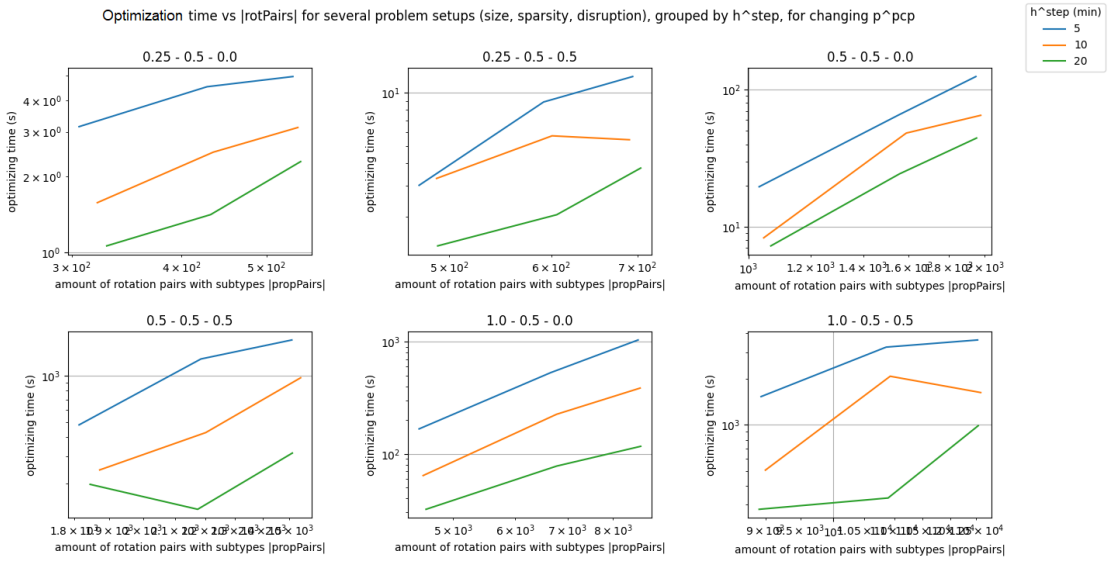

Figure 5.14: Log-log plots of the optimization time in the experiments against the amount of propagation pairs $|propPairs|$. The lines correspond to same problem setup and value of h^{step}, for changing p^{pcp} (and thus changing $|propPairs|$. The titles of the sub-figures correspond to the problem setups $(size, sparsity, disruption)$. Only a subset of the experiments are shown.

The value of h^{step} had a translating effect on the relation between the optimization time and the amount of propagation pairs $|propPairs|$, see Figure 5.13. To investigate the precise influence of the value of h^{step} on the optimization time, let us plot the optimization time against the value of h^{step}, while keeping all other parameters constant, see Figure 5.15.

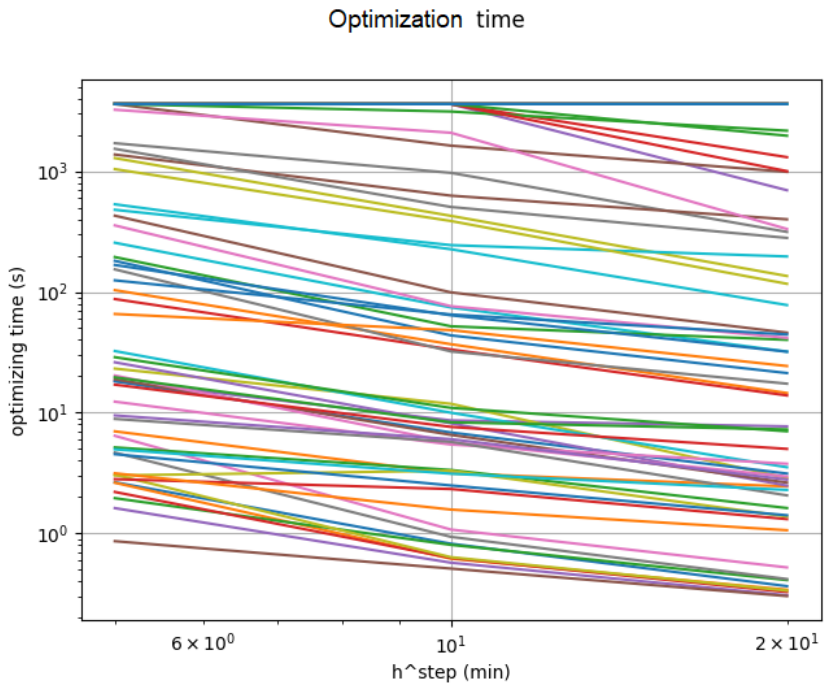

Figure 5.15: Log-log plot of the optimization time in the experiments against the value of h^{step} used in the model. The lines correspond to experiments on the same problem setup, as well as the same value of p^{pcp} for the model, for changing values of h^{step}.

In general, the lines in Figure 5.13 show a linear relation between the optimization time and the value of h^{step} in the log-log plot, when all other parameters are constant. Also, we see that the steepness of these linear relations in this plot is quite constant. Note that the lines reaching an optimization time of 3700 seconds are experiments that did not reach optimality, and were cut off at 3600 seconds of optimization. If consider the steepnesses of the linearized relations in this plot, disregarding any experiments that were cut off, we find an average steepness of approximately -1.18. Therefore, we update the our equation for the optimization time to:

$$optimizationtime \approx c * (h^{step})^{-1.18} * |propPairs|^k$$

Note that both c and k are solely dependent on the specific problem setup.

In conclusion, given a problem, both the amount of propagation pairs $|propPairs|$, and the step size h^{step} have polynomial influence on the optimization time of the Robust Flow Model.

First solution time

Next, let us consider the first solution time, i.e. the timestep in the optimization process when the first feasible solution is found. We perform an similar analysis to the optimization time. Figure 5.16 and 5.17 show the first solution time in the experiments versus the amount of propagation pairs $|propPairs|$.

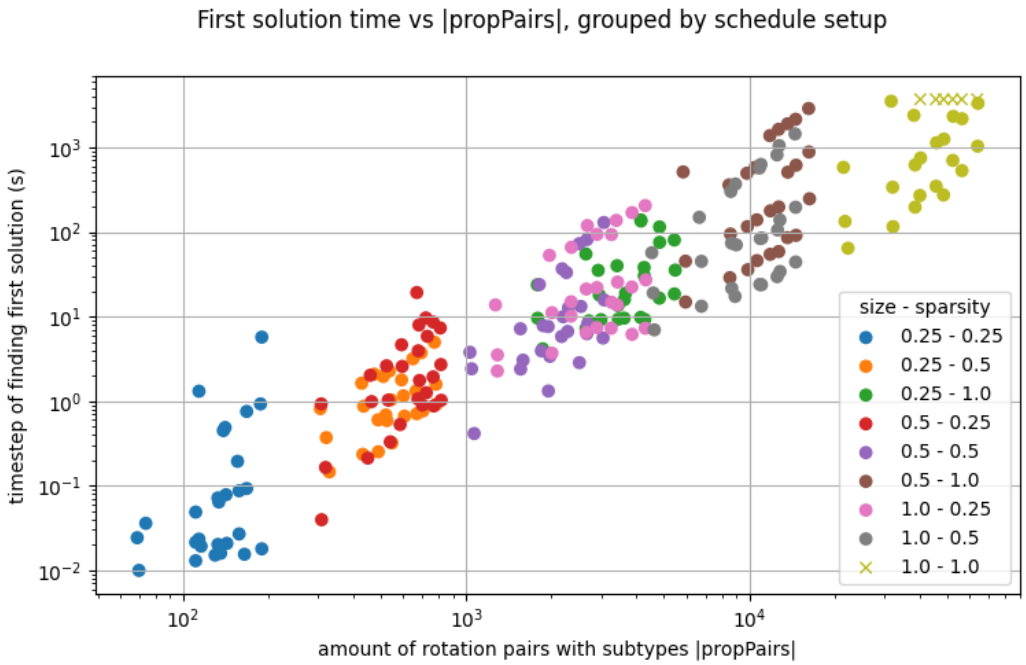

Figure 5.16: Log-log plot of the first solution time in the experiments against the amount of propagation pairs $|propPairs|$. The results are grouped by the problem setup.

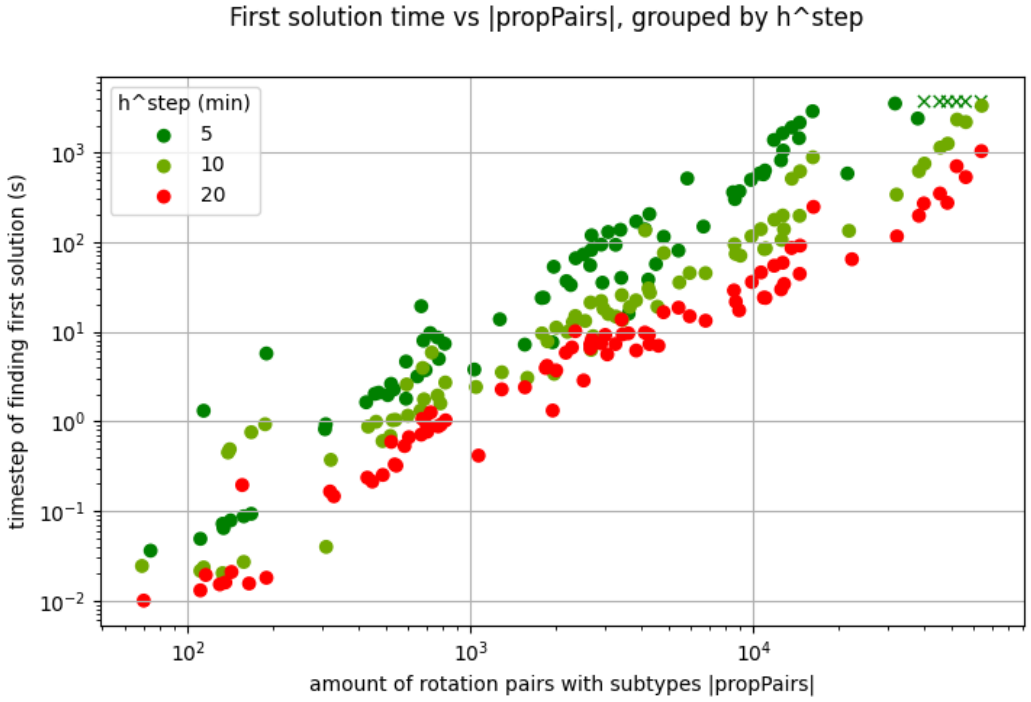

Figure 5.17: Log-log plot of the first solution time in the experiments against the amount of propagation pairs $|propPairs|$. The results are grouped by the value of h^{step} used in for the model.

Using Figure 5.16 and 5.17, we see that for every distinct value of h^{step}, a linear relation in the log-log plot is present between $|propPairs|$ and the first solution time, with the same steepness. The linear relation approximation of this relation is given in Figure 5.18.

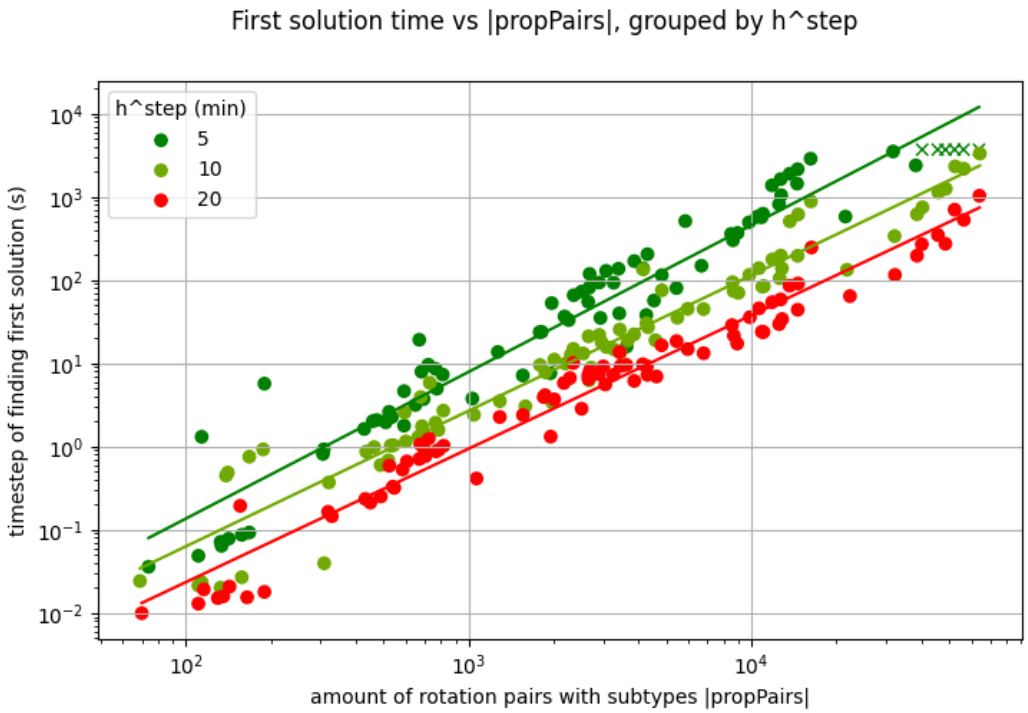

Figure 5.18: Log-log plot of the first solution time in the experiments against the amount of propagation pairs $|propPairs|$. The results are grouped by the value of h^{step} used in for the model. For every value of h^{step}, a linear approximation of the values is also shown.

A linear relation in a log-log plot corresponds to a polynomial relation. Thus, we have the following time complexity:

$$firstsolutiontime \approx c * |propPairs|^k$$

Note that c depends on the value of h^{step}. The steepness of the linear relation in the log-log plot is equal to the power k in this polynomial relation. The linear approximations in Figure 5.18 have a steepness of 1.66, which is independent of the value of h^{step}. So, we get the time complexity:

$$firstsolutiontime \approx c * |propPairs|^{1.66}$$

The value of h^{step} has a translating effect in the on the linear relation in the log-log plot between the first solution time and the amount of propagation pairs $|propPairs|$, see Figure 5.18. To investigate the precise influence of the value of h^{step} on the first solution time, let us plot the first solution time against the value of h^{step}, while keeping all other parameters constant, see Figure 5.19.

The lines in Figure 5.19, disregarding the experiments corresponding to very short times (that are prone to fluctuations, see the lines with timesteps under one second), show a linear relation between the first solution time and the value of h^{step} in the log-log plot. We see that the steepness of these linear relations in this plot is quite constant. If consider the steepnesses of the linearized relations in this plot, we find an average steepness of approximately -1.64. Therefore, we update the assumed time complexity of the first solution time to:

$$firstsolutiontime \approx c * (h^{step})^{-1.64} * |propPairs|^{1.66}$$

If we assume c to be constant, we have:

$$firstsolutiontime = O\left((h^{step})^{-1.64} * |propPairs|^{1.66}\right)$$

In conclusion, both the amount of propagation pairs $|propPairs|$, and the step size h^{step} have polynomial influence on the first solution time of the Robust Flow Model.

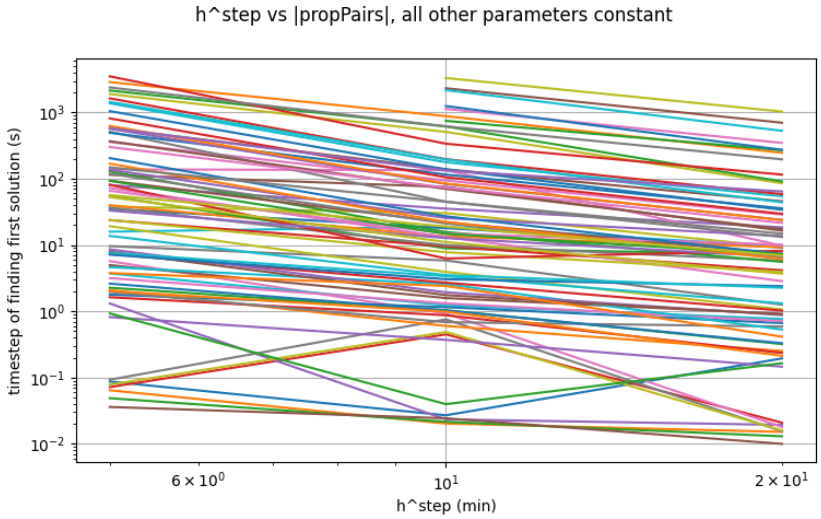

Figure 5.19: Log-log plot of the first solution time in the experiments against the value of h^{step} used in the model. The lines correspond to experiments on the same problem setup, as well as the same value of p^{pcp} for the model, for changing values of h^{step}.

5.2.2. Performance

In this section the results of the performance analysis for RFM will be evaluated. First, the approximation ratios of the RFM solutions from these experiments are evaluated. Next, we analyse the achieved MIP gap within the timespan of the experiments. Finally, the solutions found by the RFM models are compared to each other, to the BRM solution, and the NRM solution.

For the purpose of this analysis, a total of 180 experiments are performed, one experiment for each combination of the parameter values specified in Section 5.1.4. For these experiments, a time limit of 15 minute is given. Note that this time limit includes the preprocessing, model creating, and optimizing time. Out of the 180 experiments, three models were solved to optimality, and 32 experiments did not find any feasible solutions within the given time-span. For every problem setup, BRM and NRM are also optimized.

Approximation ratio

RFM approximates the expected costs of operating a solution schedule in real life. In the creating process of RFM, several approximation are made, causing the model to be solved more easily, at the cost of optimality. The delay distributions are discretized, the states are discretized with steps of length h^{step}, and propagation of delay is only considered for pairs of rotations with higher probabilities of propagation. Note that these approximations only apply to the calculations concerning expected delay costs (including reserve aircraft costs). The expected assignment costs are not approximated in the model.

Exactly how much the approximated expected cost of a solution differs from its actual expected cost is called the *approximation ratio*. For a solution S, let us define this approximation ratio R^{approx} as:

$$R^{approx}(S) := \frac{C^{delay}_{model}(S)}{C^{delay}_{simulation}(S)}$$

The values $C^{delay}_{model}(S)$ and $C^{delay}_{simulation}(S)$ correspond to the delay costs of a solution S, respectively calculated by the model and by the simulation engine. If this ratio is close to 1 for any feasible solution, the model is considered accurate. Note that we only consider delay costs for this ratio, since these are the only costs that are approximated in the model.

To evaluate the accuracy of RFM, approximation ratios $R^{approx}(S)$ are calculated for the final solution S of every experiment. Note that these solutions are often not proven to be the optimal solutions. Figure 5.20, 5.21 and 5.22 show these ratios plotted against the amount of propagation pairs $|propPairs|$ in the corresponding models.

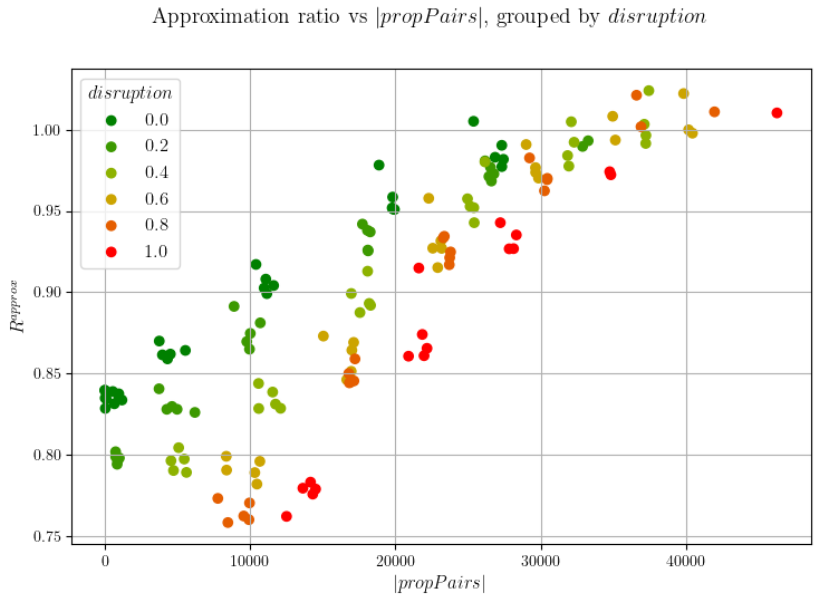

Figure 5.20: Plot of the approximation ratios R^{approx} in the experiments against the amount of propagation pairs $|propPairs|$ used in the model network. The results are grouped by the value of $disruption$, which corresponds to the different problem setups.

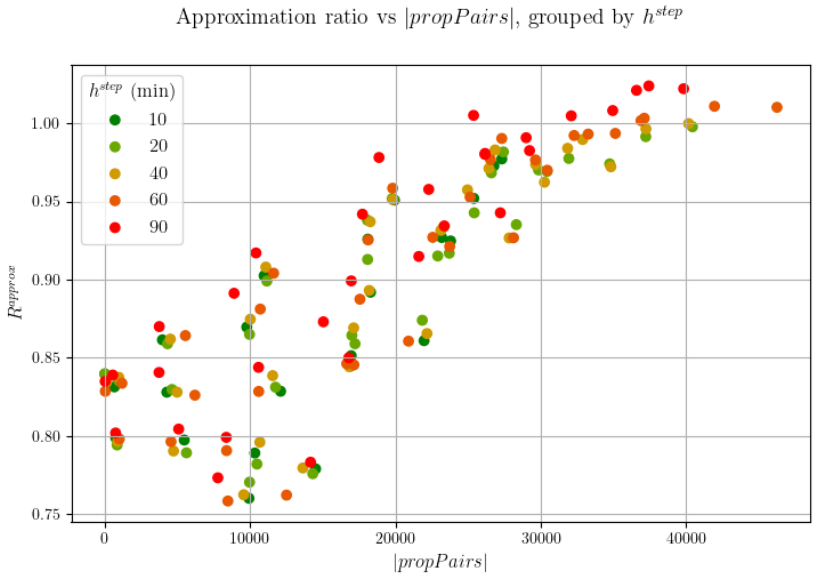

Figure 5.21: Plot of the approximation ratios R^{approx} in the experiments against the amount of propagation pairs $|propPairs|$ used in the model network. The results are grouped by the value of h^{step} used in the models.

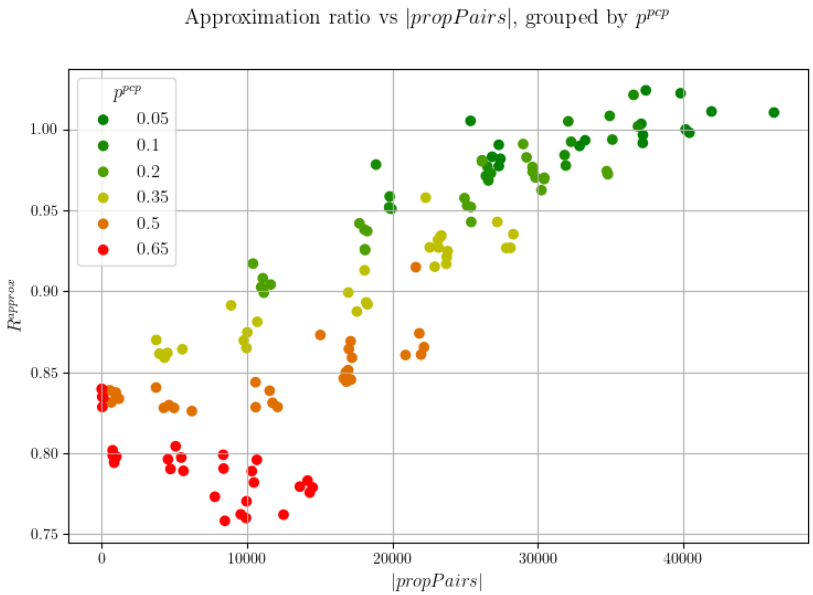

Figure 5.22: Plot of the approximation ratios R^{approx} in the experiments against the amount of propagation pairs $|propPairs|$ used in the model network. The results are grouped by the value of p^{pcp} used in the models.

From these figures, we can conclude that there is a clear relation between the approximation ratio R^{approx} and the amount of propagation pairs $|propPairs|$. This makes sense, since a lower amount of propagation pairs means that more omitted propagation pairs in the model. Also, for schedules with higher $disruption$, the accuracy drops faster when lowering the amount of propagation pairs $|propPairs|$, since more heavily disrupted schedules naturally have more delay propagation. This means more propagation pairs are necessary to capture most of this propagation into the model.

Besides this, also note that the value of h^{step} has some influence on the approximation ratio. On average higher values of h^{step} causes the approximation ratio to be slightly higher. Note that for lower p^{pcp} (and thus lower $|propPairs|$), a bigger step size h^{step} causes the approximation ratio to increase often above 1. This is the a result of the more extreme rounding that occurs with higher values of h^{step}. Since the probability density functions of expected propagated delay are usually decreasing functions throughout the fleetline (see Example 6 in Section 3.2), a bigger step size h^{step} causes the delay to be rounded up.

MIP-gap

Most of the performed experiments do not finish the optimization process, and are unable to find the optimal solution or prove that a found solution is optimal. As we have seen in the run-time analysis, even if we extent the run-time to 3600 seconds, schedules with full sparsity prove hard to optimize. Therefore, we shall analyse the final MIP-gaps achieved in the experiments.

As described in section 4.6, the Gurobi solver used to solve the model uses a branch and bound algorithm to search for feasible solutions. At any stage in this process, the MIP-gap is defined as:

$$MIP := |z_P - z_D|/|z_P|,$$

where z_P denotes the objective value of the best feasible solution found thus far, and z_D denotes the best proven bound on feasible solutions. If the $z_P = z_D$ in any stage of the solving process, it is proven that the found solution is optimal. In this case we have $MIP = 0$.

In the experiments, the value of the MIP-gap after 15 minutes of run-time is saved. Figure 5.23 shows the MIP-gap plotted against the amount of propagation pairs $|propPairs|$. In this figure we see that if we consider relatively little propagation pairs ($|propPairs| \leq 20000$), the MIP-gap often drops below 1%. Also, we see that the MIP-gap depends on the amount of disruption in the schedule. This can be explained by considering the fraction of expected costs that are delay costs in model, for the final solution found by the optimizer. Figure 5.24 shows this fraction of expected delay costs for all experiments. In this figure we see that the delay costs incorporate about 8% of the expected costs in

schedules with minimal expected disruption, and up to 23 % of the expected costs in very disrupted schedules. The delay costs are most difficult to optimize over, which is why the MIP-gap will be higher if a higher fraction of the expected costs are delay costs.

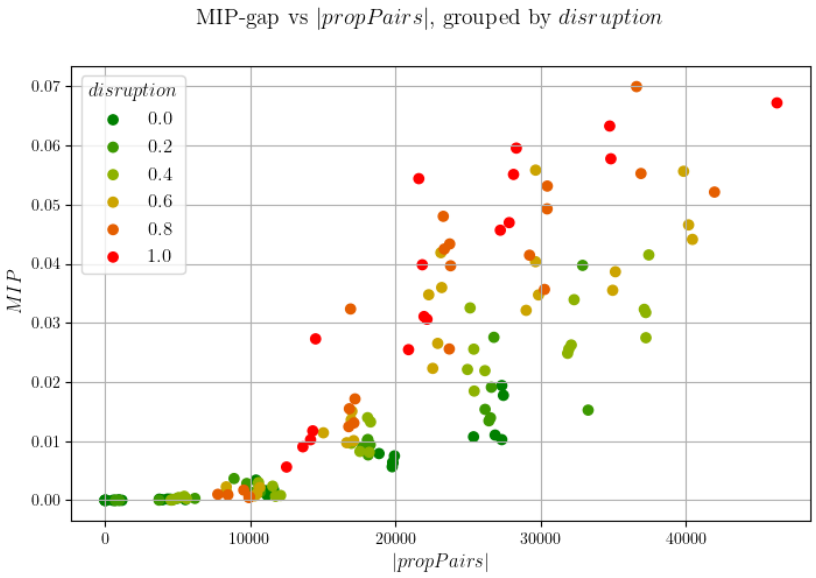

Figure 5.23: Plot of the the MIP-gap plotted against the amount of propagation pairs $|propPairs|$. The results are grouped by the value of $disruption$, which corresponds to the different problem setups.

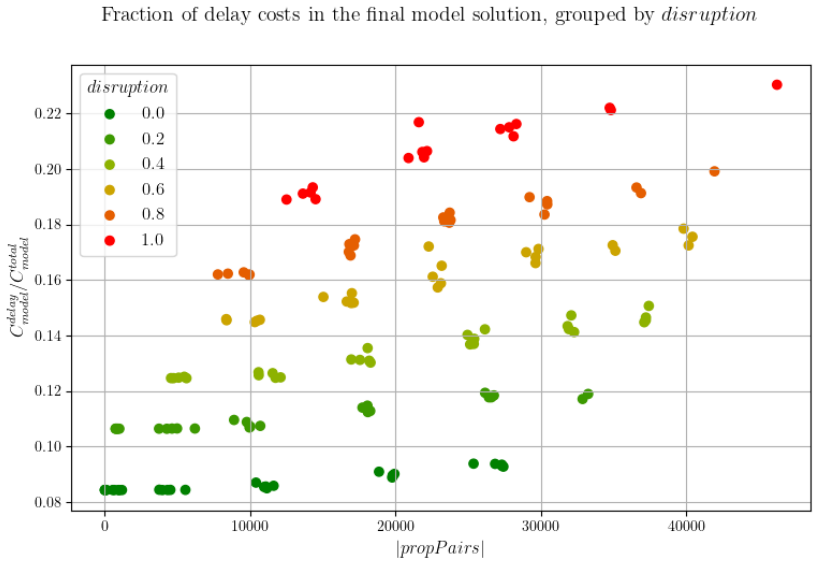

Figure 5.24: Plot of the the fraction of delay costs in the final model solution, plotted against the amount of propagation pairs $|propPairs|$. The results are grouped by the value of $disruption$, which corresponds to the different problem setups.

Simulation results

The approximation ratio combined with MIP-gap, after 900 seconds, give an indication of the expected performance of the model. If more propagation pairs $propPairs$ are considered in the model, the approximation ratio gets better, but the MIP-gap gets worse. To get the best model performance, we need to find the sweet spot for this trade-off.

To evaluate the quality of the final solutions found by the models in the experiments, Figure 5.25 shows the average total costs after simulating these solution schedules 500 times each. The average costs are given as a ratio of the average costs of the given original schedule. Note that for schedules with

more expected disruption, there are more delay costs to be reduced by the models, which will cause lower ratios of costs. Therefore, we also plot these average simulated costs separately for the different values of $disruption$, see Figure 5.26. For comparison, this figure also includes the solutions found by BRM and NRM.

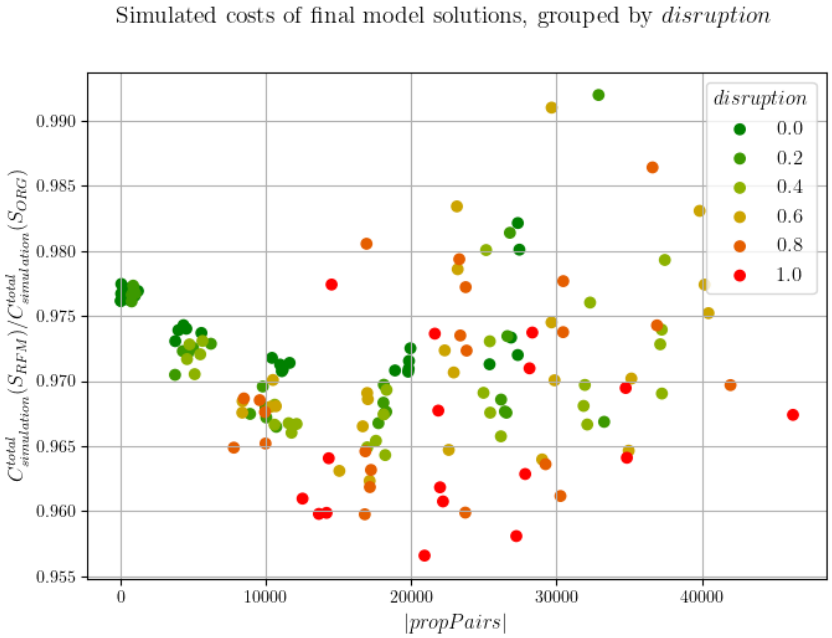

Figure 5.25: Plot of the simulated costs of final model solutions, scaled by the simulated costs of the original schedule, against the amount of propagation pairs $|propPairs|$ in the model. The results are grouped by the value of $disruption$, which corresponds to the different problem setups.

From these figures, we can conclude that the optimal amount of propagation pairs $|propPairs|$ for the tested problem schedule and time limit seems to be around 20000 propagation pairs, independently of the amount of $disruption$ in the schedule. A direct correlation between the value of h^{step} and the solution quality is not clearly deducible. For schedules with small amounts of $disruption$ (i.e. $disruption \in \{0.0, 0.2\}$), models with higher step size h^{step} seem to perform better. In schedules with higher amount of $disruption$, there is a lot of variation in the quality of the solution of the models.

Figure 5.27 shows the same figure as before, but this time colored based on the final MIP-gap reached by the optimizer. From this figure we can deduce that the MIP-gap is a clear indicator of the quality of a solution. For models with the same value of p^{pcp} (and therefore similar $|propPairs|$), a lower MIP-gap implies a better solution, irregardless of the value of h^{step}. Therefore, h^{step} does not seem to influence the quality of the solution much (within the set of values used for h^{step}), but may still have an impact on the speed of solving the model and thus on the MIP-gap.

We do see that models with very little propagation pairs $|propPairs|$ do not produce very good solutions while having a very low MIP-gap. This can be explained by the low approximation ratios, as discussed before.

An explanation for the large variation in MIP-gaps for similar experiments can be due to luck when branching in the branch and bound solving method. If a relatively good solution is found early in the process, the algorithm can cut branches faster, resulting in faster optimization.

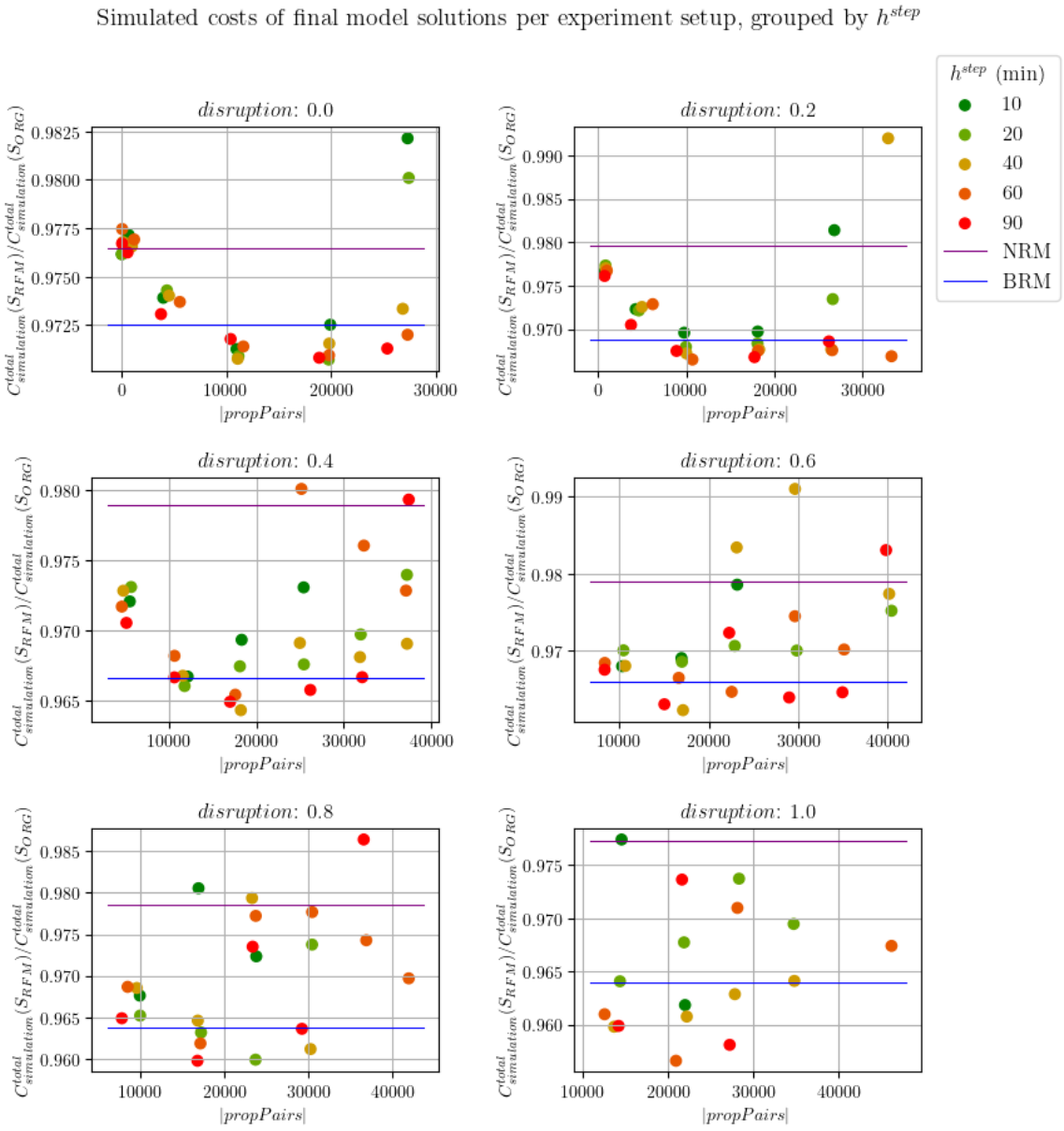

Figure 5.26: Plots of the simulated costs of final model solutions, scaled by the simulated costs of the original schedule, against the amount of propagation pairs $|propPairs|$ in the model. The results are split by the value of $disruption$, which corresponds to the different problem setups, and grouped by the value of h^{step} used in the model. Also, the scaled simulated costs of the final solutions of the Benchmark Robust Model (BRM) and the Non-Robust Model (NRM) for each problem setup are shown.

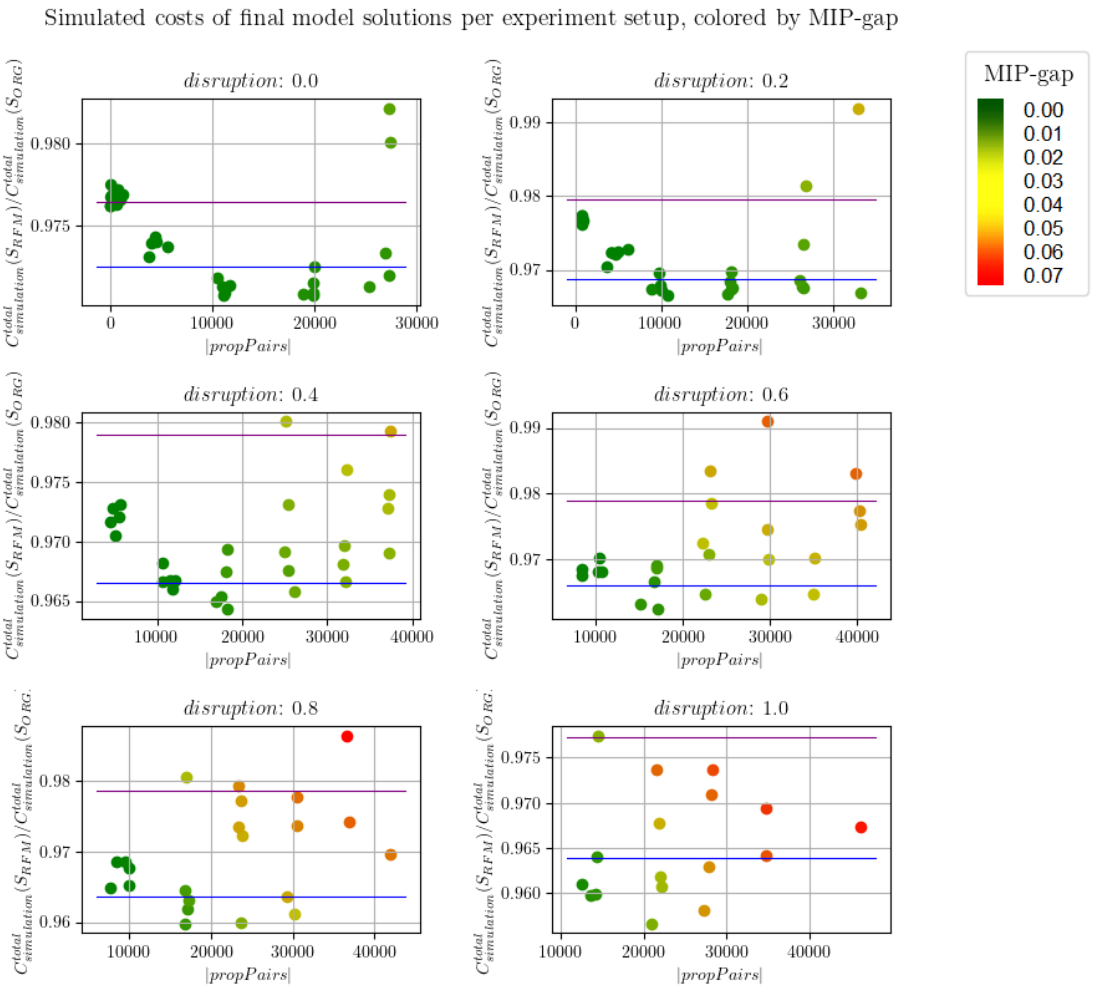

Figure 5.27: Plots of the simulated costs of final model solutions, scaled by the simulated costs of the original schedule, against the amount of propagation pairs $|propPairs|$ in the model. The results are split by the value of $disruption$, which corresponds to the different problem setups, and coloured by the MIP-gap achieved in the experiments. Also, the scaled simulated costs of the final solutions of the Benchmark Robust Model (BRM) and the Non-Robust Model (NRM) for each problem setup are shown.

From these experiments, we can conclude that BRM, as provided by KLM, finds better robust solutions than NRM. But the experiments also show that even better solutions are possible, and that RFM is capable of finding such solutions.

In order to find good solutions using RFM, a sufficient amount of propagation pairs need to be included in the delay network, to be able to capture the propagated delay. This amount of propagation pairs $|propPairs|$ can be controlled by changing the value of the parameter p^{pcp} in the network creating process. If a solver is able to reduce the MIP-gap enough, the solution will be of great quality. For big problems sizes, such as the problem used in this analysis, the model proves hard to optimize using the Gurobi solver on a regular computer. But, if good solutions are found quickly in the solving process, RFM can outperform BRM.

To show that the RFM indeed outperforms BRM if low enough MIP-gaps are reached in the solving process, let us consider a smaller problem size, i.e. 20% of the full schedule. As before, Figure 5.28 shows the evaluated results of solving this problem for several instances of RFM, compared to BRM and NRM. All models used in this experiment were fully optimized. From this figure we can conclude that considering more propagation pairs $|propPairs|$ included in the network increases the accuracy of RFM, providing better solutions if solved to optimality. If enough propagation pairs are included, RFM outperforms BRM. The value of h^{step} used for RFM does not seem to influence the quality of the solution much.

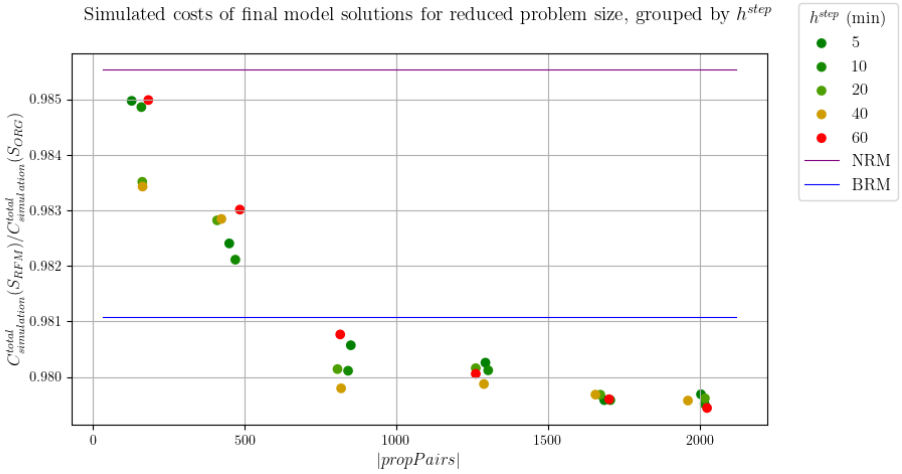

Figure 5.28: Plot of the simulated costs of final model solutions, scaled by the simulated costs of the original schedule, against the amount of propagation pairs $|propPairs|$ in the model. The problem solved is a problem with reduced size ($size = 0.2$). The results are grouped by the value of h^{step} used in the model. Also, the scaled simulated costs of the final solutions of the Benchmark Robust Model (BRM) and the Non-Robust Model (NRM) for each problem setup are shown.

6

Conclusion

In this thesis a novel model for the Robust Tail Assignment problem is described and analysed. The proposed Robust Flow Model uses a multi-commodity flow network to approximate the propagation of delay through solution flight schedules. The delay options that are considered in the network are discretized, using a constant step size. To reduce the amount of connections in the network, only connections between flight rotations are considered if the probability of delay propagation exceeds a threshold. This network is utilized in a Mixed Integer Programming model, which can be solved using an iterative solver. The model uses the network to consider expected delay costs for every flight, including propagated delay, using assignment variables to activate and deactivate edges in the network, creating the correct flows.

With the right parameters, the model is capable of finding very good robust solutions. A good robust solution will cause a decrease in expected total costs of operating the schedule. But for big problem instances, the model proves difficult to solve to optimality. Using a iterative solver, based on a branch-and-bound algorithm, the quality of solutions found during the solving process depends on the branching decisions. Therefore, if a time limit is used in the solving process, causing the model to not be solved to optimality, the best solution found in this time period can vary in quality. Table 6.1 contains a summary of the pros and cons of the three different models that were tested in this thesis.

Robust Flow Model	
Pros	Capable find great robust solutions Optimizes accurately for expected costs
Cons	Parameters need to be well tuned Difficult to solve to optimality for big problem sizes

Benchmark Robust Model	
Pros	Finds good robust solutions Solvable in reasonable time for big problem sizes
Cons	Does not optimize for expected costs Optimality is lost by limiting the assignments to one subtype based on non-robust optimization

Non-Robust Model	
Pros	Fast optimization
Cons	Not robust

Table 6.1: Pros and cons of the three models described in this thesis.

The Robust Flow Model can be extended to include other airline related aspects. Two common extensions to the Tail Assignment Problem are Schengen/Non-Schengen connections and crew connections. Schengen/Non-Schengen connections influence the required turnaround time of the aircraft at the hub station based on the destinations of the rotations, which can directly be included while creating the delay propagation network, since this network is connection based. Crew connections can cause delays in the fleetline of one aircraft to propagate to the fleetline of another aircraft. This means delays in the different fleetlines are no longer independent. Therefore, considering delay propagation caused by crew connections increases the difficulty of the problem significantly. But, the state nodes in the Robust Flow Model and the corresponding probabilities can be used to account for this propagation of delay by crew connections, using constraints that allow crew connections to update the probabilities of departure delay states accordingly.

In order to speed up the solving process for the Robust Flow Model, further research can be done. The solver could benefit from advanced branching methods or rounding heuristics, which will cause the branch-and-bound algorithm to find good solutions faster, and therefore reduce the optimization time. Another way of speeding up the algorithm, is by supplying a good initial solution. A simple heuristic or heuristic model can be used to find a reasonably good robust solution, which can be supplied as an initial solution. The Robust Flow Model can then be initialised using this solution, to kick-start the branch-and-bound algorithm with a low incumbent value. This way, nodes in the search tree can be fathomed more often, decreasing the optimization time.

To use the Robust Flow Model in practice, a simple option to get better performance from the model is by using a better computer. The computational capabilities of a laptop are limited, using hardware specifically designed for solving computational problems can speed up the solving process significantly.

Besides this, the Robust Flow Model can be simplified significantly by using a similar approach as the Benchmark Robust Model. One could first solve the non-robust tail assignment problem, without considering delay costs, and use the resulting assignment as input for the Robust Flow Model. By restricting the Robust Flow Model to only assign aircraft of the same subtype as the aircraft assigned by this non-robust model to the rotations, the problem is split into sub-problems. This will decrease the search space of the model and speed up the model, at the cost of optimality. But, since the Robust Flow Model accurately optimizes over the total expected costs, it will still outperform the Benchmark Robust Model, which uses a heuristic method to quantify robustness.

Finally, it is worth noting that the simulator, as defined in Section 2.4.1 can be used to the advantage of the user. Several solutions found in the process of solving the Robust Flow Model for varying model parameters can be saved and evaluated, after which the best solution can be chosen to use in practise. Note that this is a general tool that can be used to evaluate solutions of any robust tail assignment model, and is not restricted to the Robust Flow Model.

Appendices

A

Example problem formulations

A.1. Example Non-Robust Tail Assignment Model

$$
\begin{aligned}
\underset{X_{r,a}}{\text{minimize}} \quad & 1210 \cdot X_{r_1,a_1} + 1290 \cdot X_{r_1,a_2} + 1330 \cdot X_{r_1,a_3} \\
& + 2150 \cdot X_{r_2,a_1} + 2110 \cdot X_{r_2,a_2} + 2240 \cdot X_{r_2,a_3} \\
& + 3410 \cdot X_{r_3,a_1} + 3420 \cdot X_{r_3,a_2} + 3510 \cdot X_{r_3,a_3} \\
& + 0 \cdot X_{r_4,a_2} \\
& + 2930 \cdot X_{r_5,a_1} + 2900 \cdot X_{r_5,a_2} + 3120 \cdot X_{r_5,a_3} \\
& + 2350 \cdot X_{r_6,a_1} + 2270 \cdot X_{r_6,a_2} + 2550 \cdot X_{r_6,a_3} \\
& + 1990 \cdot X_{r_7,a_1} + 2090 \cdot X_{r_7,a_2} + 2120 \cdot X_{r_7,a_3}
\end{aligned}
$$

$$
\begin{aligned}
\text{subject to} \quad & X_{r_1,a_1} + X_{r_1,a_2} + X_{r_1,a_3} && = 1 \\
& X_{r_2,a_1} + X_{r_2,a_2} + X_{r_2,a_3} && = 1 \\
& X_{r_3,a_1} + X_{r_3,a_2} + X_{r_3,a_3} && = 1 \\
& X_{r_4,a_2} && = 1 \\
& X_{r_5,a_1} + X_{r_5,a_2} + X_{r_5,a_3} && = 1 \\
& X_{r_6,a_1} + X_{r_6,a_2} + X_{r_6,a_3} && = 1 \\
& X_{r_7,a_1} + X_{r_7,a_2} + X_{r_7,a_3} && = 1 \\
& X_{r_1,a_1} + X_{r_2,a_1} && \leq 1 \\
& X_{r_1,a_2} + X_{r_2,a_2} && \leq 1 \\
& X_{r_1,a_1} + X_{r_3,a_1} && \leq 1 \\
& X_{r_1,a_2} + X_{r_3,a_2} && \leq 1 \\
& X_{r_2,a_1} + X_{r_3,a_1} && \leq 1 \\
& X_{r_2,a_2} + X_{r_3,a_2} && \leq 1 \\
& X_{r_3,a_2} + X_{r_4,a_2} && \leq 1 \\
& X_{r_3,a_1} + X_{r_5,a_1} && \leq 1 \\
& X_{r_3,a_2} + X_{r_5,a_2} && \leq 1 \\
& X_{r_4,a_2} + X_{r_5,a_2} && \leq 1 \\
& X_{r_5,a_1} + X_{r_6,a_1} && \leq 1 \\
& X_{r_5,a_2} + X_{r_6,a_2} && \leq 1 \\
& X_{r_5,a_1} + X_{r_7,a_1} && \leq 1 \\
& X_{r_5,a_2} + X_{r_7,a_2} && \leq 1 \\
& X_{r_6,a_1} + X_{r_7,a_1} && \leq 1 \\
& X_{r_6,a_2} + X_{r_7,a_2} && \leq 1 \\
& X_{r_1,a_3} + X_{r_2,a_3} && \leq 1 \\
& X_{r_1,a_3} + X_{r_3,a_3} && \leq 1 \\
& X_{r_2,a_3} + X_{r_3,a_3} && \leq 1 \\
& X_{r_3,a_3} + X_{r_5,a_3} && \leq 1 \\
& X_{r_5,a_3} + X_{r_6,a_3} && \leq 1 \\
& X_{r_5,a_3} + X_{r_7,a_3} && \leq 1 \\
& X_{r_6,a_3} + X_{r_7,a_3} && \leq 1 \\
& X_{r,a} \in \{0,1\} && \quad r \in \mathcal{R}, a \in \mathcal{A}_r
\end{aligned}
$$

A.2. Base of example Robust Flow Model

$$
\begin{aligned}
\underset{X_{r,a}}{\text{minimize}} \quad & 1210 \cdot X_{r_1,a_1} + 1290 \cdot X_{r_1,a_2} + 1330 \cdot X_{r_1,a_3} \\
& + 2150 \cdot X_{r_2,a_1} + 2110 \cdot X_{r_2,a_2} + 2240 \cdot X_{r_2,a_3} \\
& + 3410 \cdot X_{r_3,a_1} + 3420 \cdot X_{r_3,a_2} + 3510 \cdot X_{r_3,a_3} \\
& + 0 \cdot X_{r_4,a_2} \\
& + 2930 \cdot X_{r_5,a_1} + 2900 \cdot X_{r_5,a_2} + 3120 \cdot X_{r_5,a_3} \\
& + 2350 \cdot X_{r_6,a_1} + 2270 \cdot X_{r_6,a_2} + 2550 \cdot X_{r_6,a_3} \\
& + 1990 \cdot X_{r_7,a_1} + 2090 \cdot X_{r_7,a_2} + 2120 \cdot X_{r_7,a_3}
\end{aligned}
$$

subject to

$$
\begin{aligned}
X_{r_1,a_1} + X_{r_2,a_1} + X_{r_3,a_1} &\leq 1 \\
X_{r_1,a_2} + X_{r_2,a_2} + X_{r_3,a_2} &\leq 1 \\
X_{r_1,a_3} + X_{r_2,a_3} + X_{r_3,a_3} &\leq 1 \\
X_{r_3,a_1} + X_{r_5,a_1} &\leq 1 \\
X_{r_3,a_2} + X_{r_4,a_2} + X_{r_5,a_2} &\leq 1 \\
X_{r_3,a_3} + X_{r_5,a_3} &\leq 1 \\
X_{r_5,a_1} + X_{r_6,a_1} + X_{r_7,a_1} &\leq 1 \\
X_{r_5,a_2} + X_{r_6,a_2} + X_{r_7,a_2} &\leq 1 \\
X_{r_5,a_3} + X_{r_6,a_3} + X_{r_7,a_3} &\leq 1 \\
pair_{r_2,r_4,s_1} &\geq X_{r_2,a_2} + X_{r_4,a_2} - 1 \\
pair_{r_2,r_5,s_1} &\geq X_{r_2,a_1} + X_{r_5,a_1} - 1 \\
pair_{r_2,r_5,s_1} &\geq X_{r_2,a_2} + X_{r_5,a_2} - 1 \\
pair_{r_2,r_5,s_2} &\geq X_{r_2,a_3} + X_{r_5,a_3} - 1 \\
pair_{r_2,r_6,s_1} + pair_{r_2,r_4,s_1} &\geq X_{r_2,a_1} + X_{r_6,a_1} - 1 \\
pair_{r_2,r_6,s_1} + pair_{r_2,r_4,s_1} &\geq X_{r_2,a_2} + X_{r_6,a_2} - 1 \\
pair_{r_2,r_6,s_2} &\geq X_{r_2,a_3} + X_{r_6,a_3} - 1 \\
pair_{r_3,r_6,s_1} &\geq X_{r_3,a_1} + X_{r_6,a_1} - 1 \\
pair_{r_3,r_6,s_1} &\geq X_{r_3,a_2} + X_{r_6,a_2} - 1 \\
pair_{r_3,r_6,s_2} &\geq X_{r_3,a_3} + X_{r_6,a_3} - 1 \\
pair_{r_4,r_6,s_1} &\geq X_{r_4,a_2} + X_{r_6,a_2} - 1 \\
pair_{r_4,r_7,s_1} &\geq X_{r_4,a_2} + X_{r_7,a_2} - 1
\end{aligned}
$$

$$
\begin{aligned}
noOutPair_{r_1,s_1} &= X_{r_1,a_1} + X_{r_1,a_2} \\
noOutPair_{r_1,s_2} &= X_{r_1,a_3} \\
noOutPair_{r_2,s_1} + pair_{r_2,r_4,s_1} + pair_{r_2,r_5,s_1} + pair_{r_2,r_6,s_1} &= X_{r_2,a_1} + X_{r_2,a_2} \\
noOutPair_{r_2,s_2} + pair_{r_2,r_5,s_2} + pair_{r_2,r_6,s_2} &= X_{r_2,a_3} \\
noOutPair_{r_3,s_1} + pair_{r_2,r_6,s_1} &= X_{r_3,a_1} + X_{r_3,a_2} \\
noOutPair_{r_3,s_2} + pair_{r_2,r_6,s_2} &= X_{r_3,a_3} \\
noOutPair_{r_4,s_1} + pair_{r_4,r_6,s_1} + pair_{r_4,r_7,s_1} &= X_{r_4,a_2} \\
noOutPair_{r_5,s_1} &= X_{r_5,a_1} + X_{r_5,a_2} \\
noOutPair_{r_5,s_2} &= X_{r_5,a_3} \\
noOutPair_{r_6,s_1} &= X_{r_6,a_1} + X_{r_6,a_2} \\
noOutPair_{r_6,s_2} &= X_{r_6,a_3} \\
noOutPair_{r_7,s_1} &= X_{r_7,a_1} + X_{r_7,a_2} \\
noOutPair_{r_7,s_2} &= X_{r_7,a_3} \\
noIncPair_{r_1,s_1} &= X_{r_1,a_1} + X_{r_1,a_2} \\
noIncPair_{r_1,s_2} &= X_{r_1,a_3} \\
noIncPair_{r_2,s_1} &= X_{r_2,a_1} + X_{r_2,a_2} \\
noIncPair_{r_2,s_2} &= X_{r_2,a_3} \\
noIncPair_{r_3,s_1} &= X_{r_3,a_1} + X_{r_3,a_2} \\
noIncPair_{r_3,s_2} &= X_{r_3,a_3} \\
noIncPair_{r_4,s_1} + pair_{r_2,r_4,s_1} &= X_{r_4,a_2} \\
noIncPair_{r_5,s_1} + pair_{r_2,r_5,s_1} &= X_{r_5,a_1} + X_{r_5,a_2} \\
noIncPair_{r_5,s_2} + pair_{r_2,r_5,s_2} &= X_{r_5,a_3} \\
noIncPair_{r_6,s_1} + pair_{r_2,r_6,s_1} + pair_{r_4,r_6,s_1} + pair_{r_3,r_6,s_1} &= X_{r_6,a_1} + X_{r_6,a_2} \\
noIncPair_{r_6,s_2} + pair_{r_2,r_6,s_2} + pair_{r_3,r_6,s_2} &= X_{r_6,a_3} \\
noIncPair_{r_7,s_1} + pair_{r_4,r_7,s_1} &= X_{r_7,a_1} + X_{r_7,a_2} \\
noIncPair_{r_7,s_2} &= X_{r_7,a_3} \\
X_{r,a} \in \{0,1\} \qquad & \forall r \in \mathcal{R}, a \in \mathcal{A}_r
\end{aligned}
$$

A.3. Edges in example Robust Flow Model

A.3.1. Edges between rotations E^{prop}

$((200, r_2, s_1), (205, r_4, s_1))$
$((210, r_2, s_1), (210, r_4, s_1))$
$((225, r_2, s_1), (225, r_4, s_1))$
$((240, r_2, s_1), (240, r_4, s_1))$
$((255, r_2, s_1), (255, r_4, s_1))$
$((270, r_2, s_1), (270, r_4, s_1))$
$((285, r_2, s_1), (285, r_4, s_1))$
$((300, r_2, s_1), (300, r_4, s_1))$
$((315, r_2, s_1), (315, r_4, s_1))$
$((330, r_2, s_1), (330, r_4, s_1))$
$((345, r_2, s_1), (345, r_4, s_1))$
$((360, r_2, s_1), (360, r_4, s_1))$
$((375, r_2, s_1), (375, r_4, s_1))$
$((390, r_2, s_1), (390, r_4, s_1))$
$((405, r_2, s_1), (405, r_4, s_1))$
$((420, r_2, s_1), (420, r_4, s_1))$
$((435, r_2, s_1), (435, r_4, s_1))$
$((450, r_2, s_1), (450, r_4, s_1))$
$((465, r_2, s_1), (465, r_4, s_1))$

$((205, r_2, s_2), (210, r_5, s_2))$
$((210, r_2, s_2), (210, r_5, s_2))$
$((225, r_2, s_2), (225, r_5, s_2))$
$((240, r_2, s_2), (240, r_5, s_2))$
$((255, r_2, s_2), (255, r_5, s_2))$
$((270, r_2, s_2), (270, r_5, s_2))$
$((285, r_2, s_2), (285, r_5, s_2))$
$((300, r_2, s_2), (300, r_5, s_2))$
$((315, r_2, s_2), (315, r_5, s_2))$
$((330, r_2, s_2), (330, r_5, s_2))$
$((345, r_2, s_2), (210, r_5, s_2))$
$((360, r_2, s_2), (210, r_5, s_2))$
$((375, r_2, s_2), (210, r_5, s_2))$
$((390, r_2, s_2), (210, r_5, s_2))$
$((405, r_2, s_2), (210, r_5, s_2))$
$((420, r_2, s_2), (210, r_5, s_2))$

$((205, r_2, s_2), (290, r_6, s_2))$
$((210, r_2, s_2), (290, r_6, s_2))$
$((225, r_2, s_2), (290, r_6, s_2))$
$((240, r_2, s_2), (290, r_6, s_2))$
$((255, r_2, s_2), (290, r_6, s_2))$
$((270, r_2, s_2), (290, r_6, s_2))$
$((285, r_2, s_2), (290, r_6, s_2))$
$((300, r_2, s_2), (300, r_6, s_2))$
$((315, r_2, s_2), (315, r_6, s_2))$
$((330, r_2, s_2), (330, r_6, s_2))$
$((345, r_2, s_2), (345, r_6, s_2))$
$((360, r_2, s_2), (360, r_6, s_2))$
$((375, r_2, s_2), (375, r_6, s_2))$
$((390, r_2, s_2), (390, r_6, s_2))$
$((405, r_2, s_2), (405, r_6, s_2))$
$((420, r_2, s_2), (290, r_6, s_2))$

$((200, r_2, s_1), (210, r_5, s_1))$
$((210, r_2, s_1), (210, r_5, s_1))$
$((225, r_2, s_1), (225, r_5, s_1))$
$((240, r_2, s_1), (240, r_5, s_1))$
$((255, r_2, s_1), (255, r_5, s_1))$
$((270, r_2, s_1), (270, r_5, s_1))$
$((285, r_2, s_1), (285, r_5, s_1))$
$((300, r_2, s_1), (300, r_5, s_1))$
$((315, r_2, s_1), (315, r_5, s_1))$
$((330, r_2, s_1), (330, r_5, s_1))$
$((345, r_2, s_1), (210, r_5, s_1))$
$((360, r_2, s_1), (210, r_5, s_1))$
$((375, r_2, s_1), (210, r_5, s_1))$
$((390, r_2, s_1), (210, r_5, s_1))$
$((405, r_2, s_1), (210, r_5, s_1))$
$((420, r_2, s_1), (210, r_5, s_1))$
$((435, r_2, s_1), (210, r_5, s_1))$
$((450, r_2, s_1), (210, r_5, s_1))$
$((465, r_2, s_1), (210, r_5, s_1))$

$((200, r_2, s_1), (290, r_6, s_1))$
$((210, r_2, s_1), (290, r_6, s_1))$
$((225, r_2, s_1), (290, r_6, s_1))$
$((240, r_2, s_1), (290, r_6, s_1))$
$((255, r_2, s_1), (290, r_6, s_1))$
$((270, r_2, s_1), (290, r_6, s_1))$
$((285, r_2, s_1), (290, r_6, s_1))$
$((300, r_2, s_1), (300, r_6, s_1))$
$((315, r_2, s_1), (315, r_6, s_1))$
$((330, r_2, s_1), (330, r_6, s_1))$
$((345, r_2, s_1), (345, r_6, s_1))$
$((360, r_2, s_1), (360, r_6, s_1))$
$((375, r_2, s_1), (375, r_6, s_1))$
$((390, r_2, s_1), (390, r_6, s_1))$
$((405, r_2, s_1), (405, r_6, s_1))$
$((420, r_2, s_1), (290, r_6, s_1))$
$((435, r_2, s_1), (290, r_6, s_1))$
$((450, r_2, s_1), (290, r_6, s_1))$
$((465, r_2, s_1), (290, r_6, s_1))$

$((265, r_3, s_1), (290, r_6, s_1))$
$((270, r_3, s_1), (290, r_6, s_1))$
$((285, r_3, s_1), (290, r_6, s_1))$
$((300, r_3, s_1), (300, r_6, s_1))$
$((315, r_3, s_1), (315, r_6, s_1))$
$((330, r_3, s_1), (330, r_6, s_1))$
$((345, r_3, s_1), (345, r_6, s_1))$
$((360, r_3, s_1), (360, r_6, s_1))$
$((375, r_3, s_1), (375, r_6, s_1))$
$((390, r_3, s_1), (390, r_6, s_1))$
$((405, r_3, s_1), (405, r_6, s_1))$
$((420, r_3, s_1), (290, r_6, s_1))$

$((270, r_3, s_2), (290, r_6, s_2))$
$((285, r_3, s_2), (290, r_6, s_2))$
$((300, r_3, s_2), (300, r_6, s_2))$
$((315, r_3, s_2), (315, r_6, s_2))$
$((330, r_3, s_2), (330, r_6, s_2))$
$((345, r_3, s_2), (345, r_6, s_2))$
$((360, r_3, s_2), (360, r_6, s_2))$
$((375, r_3, s_2), (375, r_6, s_2))$
$((390, r_3, s_2), (390, r_6, s_2))$
$((405, r_3, s_2), (405, r_6, s_2))$
$((420, r_3, s_2), (290, r_6, s_2))$

$((275, r_4, s_1), (290, r_6, s_1))$
$((285, r_4, s_1), (290, r_6, s_1))$
$((300, r_4, s_1), (300, r_6, s_1))$
$((315, r_4, s_1), (315, r_6, s_1))$
$((330, r_4, s_1), (330, r_6, s_1))$
$((345, r_4, s_1), (345, r_6, s_1))$
$((360, r_4, s_1), (360, r_6, s_1))$
$((375, r_4, s_1), (375, r_6, s_1))$
$((390, r_4, s_1), (390, r_6, s_1))$
$((405, r_4, s_1), (405, r_6, s_1))$
$((420, r_4, s_1), (290, r_6, s_1))$
$((435, r_4, s_1), (290, r_6, s_1))$
$((450, r_4, s_1), (290, r_6, s_1))$
$((465, r_4, s_1), (290, r_6, s_1))$
$((480, r_4, s_1), (290, r_6, s_1))$

$((275, r_4, s_1), (350, r_7, s_1))$
$((285, r_4, s_1), (350, r_7, s_1))$
$((300, r_4, s_1), (350, r_7, s_1))$
$((315, r_4, s_1), (350, r_7, s_1))$
$((330, r_4, s_1), (350, r_7, s_1))$
$((345, r_4, s_1), (350, r_7, s_1))$
$((360, r_4, s_1), (360, r_7, s_1))$
$((375, r_4, s_1), (375, r_7, s_1))$
$((390, r_4, s_1), (390, r_7, s_1))$
$((405, r_4, s_1), (405, r_7, s_1))$
$((420, r_4, s_1), (420, r_7, s_1))$
$((435, r_4, s_1), (435, r_7, s_1))$
$((450, r_4, s_1), (450, r_7, s_1))$
$((465, r_4, s_1), (465, r_7, s_1))$
$((480, r_4, s_1), (350, r_7, s_1))$

A.3.2. Edges within rotations E^{rot}

$((0, r_1, s_1), (150, r_1, s_1))$
$((0, r_1, s_2), (155, r_1, s_2))$

$((5, r_2, s_1), (200, r_2, s_1))$
$((5, r_2, s_1), (210, r_2, s_1))$
$((5, r_2, s_1), (225, r_2, s_1))$
$((5, r_2, s_1), (240, r_2, s_1))$
$((5, r_2, s_1), (255, r_2, s_1))$
$((5, r_2, s_1), (270, r_2, s_1))$
$((5, r_2, s_1), (285, r_2, s_1))$
$((5, r_2, s_1), (300, r_2, s_1))$
$((5, r_2, s_1), (315, r_2, s_1))$
$((5, r_2, s_1), (330, r_2, s_1))$
$((5, r_2, s_1), (345, r_2, s_1))$
$((5, r_2, s_1), (360, r_2, s_1))$
$((5, r_2, s_1), (375, r_2, s_1))$
$((5, r_2, s_1), (390, r_2, s_1))$

$((5, r_2, s_2), (205, r_2, s_2))$
$((5, r_2, s_2), (210, r_2, s_2))$
$((5, r_2, s_2), (225, r_2, s_2))$
$((5, r_2, s_2), (240, r_2, s_2))$
$((5, r_2, s_2), (255, r_2, s_2))$
$((5, r_2, s_2), (270, r_2, s_2))$
$((5, r_2, s_2), (285, r_2, s_2))$
$((5, r_2, s_2), (300, r_2, s_2))$
$((5, r_2, s_2), (315, r_2, s_2))$
$((5, r_2, s_2), (330, r_2, s_2))$
$((5, r_2, s_2), (345, r_2, s_2))$
$((5, r_2, s_2), (360, r_2, s_2))$
$((5, r_2, s_2), (375, r_2, s_2))$
$((5, r_2, s_2), (390, r_2, s_2))$
$((5, r_2, s_2), (405, r_2, s_2))$

$((10, r_3, s_1), (265, r_3, s_1))$
$((10, r_3, s_1), (270, r_3, s_1))$
$((10, r_3, s_1), (285, r_3, s_1))$
$((10, r_3, s_1), (300, r_3, s_1))$
$((10, r_3, s_1), (315, r_3, s_1))$
$((10, r_3, s_1), (330, r_3, s_1))$
$((10, r_3, s_1), (345, r_3, s_1))$

$((10, r_3, s_2), (270, r_3, s_2))$
$((10, r_3, s_2), (285, r_3, s_2))$
$((10, r_3, s_2), (300, r_3, s_2))$
$((10, r_3, s_2), (315, r_3, s_2))$
$((10, r_3, s_2), (330, r_3, s_2))$
$((10, r_3, s_2), (345, r_3, s_2))$

$((205, r_4, s_1), (275, r_4, s_1))$
$((210, r_4, s_1), (285, r_4, s_1))$
$((225, r_4, s_1), (285, r_4, s_1))$
$((225, r_4, s_1), (300, r_4, s_1))$
$((240, r_4, s_1), (315, r_4, s_1))$
$((240, r_4, s_1), (300, r_4, s_1))$
$((255, r_4, s_1), (330, r_4, s_1))$
$((255, r_4, s_1), (315, r_4, s_1))$
$((270, r_4, s_1), (330, r_4, s_1))$
$((270, r_4, s_1), (345, r_4, s_1))$
$((285, r_4, s_1), (345, r_4, s_1))$
$((285, r_4, s_1), (360, r_4, s_1))$
$((300, r_4, s_1), (360, r_4, s_1))$
$((300, r_4, s_1), (375, r_4, s_1))$
$((315, r_4, s_1), (375, r_4, s_1))$
$((315, r_4, s_1), (390, r_4, s_1))$
$((330, r_4, s_1), (405, r_4, s_1))$
$((330, r_4, s_1), (390, r_4, s_1))$
$((345, r_4, s_1), (420, r_4, s_1))$
$((345, r_4, s_1), (405, r_4, s_1))$
$((360, r_4, s_1), (420, r_4, s_1))$
$((360, r_4, s_1), (435, r_4, s_1))$
$((375, r_4, s_1), (450, r_4, s_1))$
$((375, r_4, s_1), (435, r_4, s_1))$
$((390, r_4, s_1), (465, r_4, s_1))$
$((390, r_4, s_1), (450, r_4, s_1))$
$((405, r_4, s_1), (465, r_4, s_1))$
$((405, r_4, s_1), (480, r_4, s_1))$
$((420, r_4, s_1), (480, r_4, s_1))$
$((435, r_4, s_1), (480, r_4, s_1))$
$((450, r_4, s_1), (480, r_4, s_1))$
$((465, r_4, s_1), (480, r_4, s_1))$

$((210, r_5, s_1), (440, r_5, s_1))$
$((225, r_5, s_1), (440, r_5, s_1))$
$((240, r_5, s_1), (440, r_5, s_1))$
$((255, r_5, s_1), (440, r_5, s_1))$
$((270, r_5, s_1), (440, r_5, s_1))$
$((285, r_5, s_1), (440, r_5, s_1))$
$((300, r_5, s_1), (440, r_5, s_1))$
$((315, r_5, s_1), (440, r_5, s_1))$
$((330, r_5, s_1), (440, r_5, s_1))$

$((210, r_5, s_2), (445, r_5, s_2))$
$((225, r_5, s_2), (445, r_5, s_2))$
$((240, r_5, s_2), (445, r_5, s_2))$
$((255, r_5, s_2), (445, r_5, s_2))$
$((270, r_5, s_2), (445, r_5, s_2))$
$((285, r_5, s_2), (445, r_5, s_2))$
$((300, r_5, s_2), (445, r_5, s_2))$
$((315, r_5, s_2), (445, r_5, s_2))$
$((330, r_5, s_2), (445, r_5, s_2))$

$((290, r_6, s_1), (505, r_6, s_1))$
$((300, r_6, s_1), (505, r_6, s_1))$
$((315, r_6, s_1), (505, r_6, s_1))$
$((330, r_6, s_1), (505, r_6, s_1))$
$((345, r_6, s_1), (505, r_6, s_1))$
$((360, r_6, s_1), (505, r_6, s_1))$
$((375, r_6, s_1), (505, r_6, s_1))$
$((390, r_6, s_1), (505, r_6, s_1))$
$((405, r_6, s_1), (505, r_6, s_1))$

$((290, r_6, s_2), (510, r_6, s_2))$
$((300, r_6, s_2), (510, r_6, s_2))$
$((315, r_6, s_2), (510, r_6, s_2))$
$((330, r_6, s_2), (510, r_6, s_2))$
$((345, r_6, s_2), (510, r_6, s_2))$
$((360, r_6, s_2), (510, r_6, s_2))$
$((375, r_6, s_2), (510, r_6, s_2))$
$((390, r_6, s_2), (510, r_6, s_2))$
$((405, r_6, s_2), (510, r_6, s_2))$

$((350, r_7, s_1), (520, r_7, s_1))$
$((360, r_7, s_1), (520, r_7, s_1))$
$((375, r_7, s_1), (520, r_7, s_1))$
$((390, r_7, s_1), (520, r_7, s_1))$
$((405, r_7, s_1), (520, r_7, s_1))$
$((420, r_7, s_1), (520, r_7, s_1))$
$((435, r_7, s_1), (520, r_7, s_1))$
$((450, r_7, s_1), (520, r_7, s_1))$
$((465, r_7, s_1), (520, r_7, s_1))$
$((350, r_7, s_2), (525, r_7, s_2))$

A.4. Experiment results

A.4.1. Run-time experiments

$sparsity$	$size$	$disruption$	h^{step}	p^{pcp}	$\|\mathcal{R}\|$	$\|\mathcal{R}^{rob}\|$	$\|\mathcal{A}\|$	$\|propPairs\|$	preprocessing time (s)	model creating time (s)	optimizing time (s)	first solution time (s)
0.250	0.250	0	5	0.025	57	24	26	133	24.946	5.121	2.202	0.072
0.250	0.250	0	5	0.050	57	24	26	114	24.441	4.841	1.617	1.311
0.250	0.250	0	5	0.100	57	24	26	74	20.395	3.421	0.860	0.036
0.250	0.250	0	10	0.025	57	24	26	139	13.844	1.576	0.625	0.449
0.250	0.250	0	10	0.050	57	24	26	114	14.819	1.505	0.571	0.023
0.250	0.250	0	10	0.100	57	24	26	69	12.075	1.023	0.510	0.024
0.250	0.250	0	20	0.025	57	24	26	143	9.203	0.607	0.326	0.021
0.250	0.250	0	20	0.050	57	24	26	116	10.150	0.567	0.307	0.019
0.250	0.250	0	20	0.100	57	24	26	70	8.350	0.475	0.302	0.010
0.250	0.250	0.500	5	0.025	57	24	26	158	35.847	7.150	2.648	0.087
0.250	0.250	0.500	5	0.050	57	24	26	134	33.239	8.193	2.624	0.064
0.250	0.250	0.500	5	0.100	57	24	26	111	32.271	5.696	1.955	0.049
0.250	0.250	0.500	10	0.025	57	24	26	158	20.836	2.186	0.820	0.027
0.250	0.250	0.500	10	0.050	57	24	26	133	18.868	1.958	0.617	0.020
0.250	0.250	0.500	10	0.100	57	24	26	111	17.826	1.578	0.802	0.021
0.250	0.250	0.500	20	0.025	57	24	26	156	11.440	0.739	0.365	0.194
0.250	0.250	0.500	20	0.050	57	24	26	130	11.319	0.665	0.341	0.015
0.250	0.250	0.500	20	0.100	57	24	26	111	11.117	0.617	0.410	0.013
0.250	0.250	1	5	0.025	57	24	26	190	46.256	9.553	6.425	5.708
0.250	0.250	1	5	0.050	57	24	26	168	46.340	8.457	4.664	0.093
0.250	0.250	1	5	0.100	57	24	26	142	43.818	7.007	3.060	0.078
0.250	0.250	1	10	0.025	57	24	26	188	25.194	2.531	1.074	0.926
0.250	0.250	1	10	0.050	57	24	26	168	25.237	2.431	0.935	0.755
0.250	0.250	1	10	0.100	57	24	26	141	24.651	2.036	0.634	0.487
0.250	0.250	1	20	0.025	57	24	26	190	15.331	0.904	0.522	0.018
0.250	0.250	1	20	0.050	57	24	26	165	14.666	0.858	0.418	0.015
0.250	0.250	1	20	0.100	57	24	26	136	13.594	0.705	0.335	0.016
0.250	0.500	0	5	0.025	119	41	51	524	47.836	10.612	6.994	2.616
0.250	0.500	0	5	0.050	119	41	51	459	56.895	10.048	5.150	2.022
0.250	0.500	0	5	0.100	119	41	51	308	47.004	7.848	2.810	0.928
0.250	0.500	0	10	0.025	119	41	51	531	28.437	3.572	3.144	1.028
0.250	0.500	0	10	0.050	119	41	51	464	36.704	3.619	3.345	0.987
0.250	0.500	0	10	0.100	119	41	51	309	30.748	3.148	2.327	0.040
0.250	0.500	0	20	0.025	119	41	51	540	21.610	1.706	2.445	0.330
0.250	0.500	0	20	0.050	119	41	51	450	18.709	1.451	1.619	0.212
0.250	0.500	0	20	0.100	119	41	51	319	18.921	1.456	1.311	0.164
0.250	0.500	0.500	5	0.025	119	41	51	721	82.778	18.390	23.156	9.637
0.250	0.500	0.500	5	0.050	119	41	51	670	85.701	17.747	32.370	19.228
0.250	0.500	0.500	5	0.100	119	41	51	592	88.038	16.989	18.177	4.639
0.250	0.500	0.500	10	0.025	119	41	51	730	49.613	6.631	11.839	5.846
0.250	0.500	0.500	10	0.050	119	41	51	678	41.782	5.211	9.977	3.935
0.250	0.500	0.500	10	0.100	119	41	51	595	40.370	4.570	6.845	2.576
0.250	0.500	0.500	20	0.025	119	41	51	724	23.043	2.217	2.873	1.256
0.250	0.500	0.500	20	0.050	119	41	51	676	22.984	2.069	3.527	1.065
0.250	0.500	0.500	20	0.100	119	41	51	585	22.774	1.844	3.126	0.529
0.250	0.500	1	5	0.025	119	41	51	811	104.707	24.021	26.129	7.332
0.250	0.500	1	5	0.050	119	41	51	765	102.008	21.330	18.900	8.603
0.250	0.500	1	5	0.100	119	41	51	682	103.193	17.256	20.275	7.944
0.250	0.500	1	10	0.025	119	41	51	815	54.085	7.372	8.160	2.706
0.250	0.500	1	10	0.050	119	41	51	765	58.120	6.642	6.538	1.936
0.250	0.500	1	10	0.100	119	41	51	685	49.922	5.124	5.697	1.756

$sparsity$	$size$	$disruption$	h^{step}	p^{pcp}	$\|\mathcal{R}\|$	$\|\mathcal{R}^{rob}\|$	$\|\mathcal{A}\|$	$\|propPairs\|$	preprocessing time (s)	model creating time (s)	optimizing time (s)	first solution time (s)
0.250	0.500	1	20	0.025	119	41	51	817	26.165	2.433	2.483	1.024
0.250	0.500	1	20	0.050	119	41	51	769	26.405	2.257	2.631	0.877
0.250	0.500	1	20	0.100	119	41	51	701	26.471	2.051	2.956	0.898
0.250	1	0	5	0.025	226	92	101	2681	139.636	35.569	194.503	118.839
0.250	1	0	5	0.050	226	92	101	1970	136.638	30.308	87.308	52.954
0.250	1	0	5	0.100	226	92	101	1270	112.247	23.497	18.284	13.728
0.250	1	0	10	0.025	226	92	101	2658	71.037	13.602	51.923	21.141
0.250	1	0	10	0.050	226	92	101	2011	69.045	11.158	33.372	11.099
0.250	1	0	10	0.100	226	92	101	1291	61.942	8.423	8.620	3.519
0.250	1	0	20	0.025	226	92	101	2655	44.447	7.127	40.103	6.505
0.250	1	0	20	0.050	226	92	101	2010	43.720	5.725	13.875	3.695
0.250	1	0	20	0.100	226	92	101	1290	39.733	4.238	7.695	2.269
0.250	1	0.500	5	0.025	226	92	101	3391	175.090	46.220	255.560	136.984
0.250	1	0.500	5	0.050	226	92	101	2899	174.060	41.579	180.909	93.622
0.250	1	0.500	5	0.100	226	92	101	2355	171.385	35.823	103.448	65.914
0.250	1	0.500	10	0.025	226	92	101	3427	94.568	17.827	74.356	25.544
0.250	1	0.500	10	0.050	226	92	101	2898	94.759	15.755	43.640	21.970
0.250	1	0.500	10	0.100	226	92	101	2349	94.180	13.212	36.884	14.952
0.250	1	0.500	20	0.025	226	92	101	3420	54.415	9.464	32.050	13.662
0.250	1	0.500	20	0.050	226	92	101	2896	54.688	8.357	21.269	7.412
0.250	1	0.500	20	0.100	226	92	101	2348	54.402	6.815	14.532	10.117
0.250	1	1	5	0.025	226	92	101	4292	210.936	58.451	428.244	205.077
0.250	1	1	5	0.050	226	92	101	3856	210.267	53.325	355.699	169.254
0.250	1	1	5	0.100	226	92	101	3260	210.480	47.401	153.776	93.134
0.250	1	1	10	0.025	226	92	101	4311	105.151	23.353	99.208	27.150
0.250	1	1	10	0.050	226	92	101	3848	114.719	20.493	76.155	22.332
0.250	1	1	10	0.100	226	92	101	3267	114.521	18.082	32.119	14.747
0.250	1	1	20	0.025	226	92	101	4294	66.610	12.434	45.975	7.273
0.250	1	1	20	0.050	226	92	101	3855	66.742	11.018	41.628	6.160
0.250	1	1	20	0.100	226	92	101	3254	61.849	9.524	17.368	7.277
0.500	0.250	0	5	0.025	114	41	26	536	53.058	11.363	4.963	2.268
0.500	0.250	0	5	0.050	114	41	26	428	57.284	10.075	4.524	1.630
0.500	0.250	0	5	0.100	114	41	26	306	47.501	7.176	3.144	0.817
0.500	0.250	0	10	0.025	114	41	26	543	31.699	3.560	3.120	1.038
0.500	0.250	0	10	0.050	114	41	26	435	31.064	3.345	2.492	0.872
0.500	0.250	0	10	0.100	114	41	26	321	31.111	2.955	1.573	0.371
0.500	0.250	0	20	0.025	114	41	26	547	18.737	1.429	2.287	0.320
0.500	0.250	0	20	0.050	114	41	26	432	18.468	1.182	1.410	0.235
0.500	0.250	0	20	0.100	114	41	26	329	17.156	1.027	1.061	0.146
0.500	0.250	0.500	5	0.025	114	41	26	694	78.675	15.397	12.327	3.728
0.500	0.250	0.500	5	0.050	114	41	26	592	78.617	13.889	8.870	1.785
0.500	0.250	0.500	5	0.100	114	41	26	474	75.024	11.778	3.018	2.085
0.500	0.250	0.500	10	0.025	114	41	26	690	42.385	4.697	5.441	1.239
0.500	0.250	0.500	10	0.050	114	41	26	601	42.156	4.343	5.724	1.155
0.500	0.250	0.500	10	0.100	114	41	26	489	40.261	3.587	3.298	0.601
0.500	0.250	0.500	20	0.025	114	41	26	704	24.996	1.771	3.771	0.765
0.500	0.250	0.500	20	0.050	114	41	26	606	24.795	1.581	2.062	0.664
0.500	0.250	0.500	20	0.100	114	41	26	490	24.227	1.374	1.378	0.252
0.500	0.250	1	5	0.025	114	41	26	773	81.479	17.755	28.785	4.969
0.500	0.250	1	5	0.050	114	41	26	649	80.384	16.022	17.049	3.165
0.500	0.250	1	5	0.100	114	41	26	510	73.678	14.173	9.477	1.958

$sparsity$	$size$	$disruption$	h^{step}	p^{pcp}	$\|\mathcal{R}\|$	$\|\mathcal{R}^{rob}\|$	$\|\mathcal{A}\|$	$\|propPairs\|$	preprocessing time (s)	model creating time (s)	optimizing time (s)	first solution time (s)
0.500	0.250	1	10	0.025	114	41	26	783	40.477	5.531	10.948	1.584
0.500	0.250	1	10	0.050	114	41	26	666	40.093	4.865	7.614	1.317
0.500	0.250	1	10	0.100	114	41	26	521	40.454	4.282	6.026	0.685
0.500	0.250	1	20	0.025	114	41	26	785	23.740	2.151	7.005	0.915
0.500	0.250	1	20	0.050	114	41	26	670	25.959	1.833	5.004	0.710
0.500	0.250	1	20	0.100	114	41	26	525	25.868	1.600	2.787	0.589
0.500	0.500	0	5	0.025	230	80	51	1949	127.100	31.121	124.720	7.610
0.500	0.500	0	5	0.050	230	80	51	1556	126.921	26.750	65.678	7.169
0.500	0.500	0	5	0.100	230	80	51	1032	119.681	22.252	19.595	3.777
0.500	0.500	0	10	0.025	230	80	51	1977	69.436	10.088	65.028	3.387
0.500	0.500	0	10	0.050	230	80	51	1589	69.550	8.878	48.255	3.048
0.500	0.500	0	10	0.100	230	80	51	1046	65.553	7.021	8.310	2.413
0.500	0.500	0	20	0.025	230	80	51	1953	39.918	4.288	44.388	1.318
0.500	0.500	0	20	0.050	230	80	51	1558	39.722	3.691	24.329	2.394
0.500	0.500	0	20	0.100	230	80	51	1068	38.884	3.005	7.256	0.414
0.500	0.500	0.500	5	0.025	230	80	51	2510	161.798	39.499	1706.891	72.721
0.500	0.500	0.500	5	0.050	230	80	51	2185	160.475	35.198	1284.845	36.624
0.500	0.500	0.500	5	0.100	230	80	51	1814	155.976	31.250	479.195	23.932
0.500	0.500	0.500	10	0.025	230	80	51	2544	87.675	13.811	969.364	13.166
0.500	0.500	0.500	10	0.050	230	80	51	2201	88.620	12.076	426.966	9.946
0.500	0.500	0.500	10	0.100	230	80	51	1872	84.970	10.436	244.186	7.805
0.500	0.500	0.500	20	0.025	230	80	51	2512	50.287	5.811	314.306	2.852
0.500	0.500	0.500	20	0.050	230	80	51	2174	50.219	5.321	135.490	5.816
0.500	0.500	0.500	20	0.100	230	80	51	1845	48.947	4.353	196.712	3.933
0.500	0.500	1	5	0.025	230	80	51	3067	176.952	47.334	3600.508	129.335
0.500	0.500	1	5	0.050	230	80	51	2673	177.428	42.657	3600.388	81.423
0.500	0.500	1	5	0.100	230	80	51	2262	177.049	37.853	1376.143	33.155
0.500	0.500	1	10	0.025	230	80	51	3074	95.900	16.268	3600.183	15.715
0.500	0.500	1	10	0.050	230	80	51	2670	96.141	14.307	3600.182	6.263
0.500	0.500	1	10	0.100	230	80	51	2301	96.352	12.860	627.688	12.749
0.500	0.500	1	20	0.025	230	80	51	3046	56.050	7.096	1307.878	5.579
0.500	0.500	1	20	0.050	230	80	51	2687	55.363	6.478	695.974	8.188
0.500	0.500	1	20	0.100	230	80	51	2285	55.293	5.415	399.683	6.667
0.500	1	0	5	0.025	451	175	101	8607	242.263	80.985	1041.284	301.534
0.500	1	0	5	0.050	451	175	101	6663	228.424	65.463	532.427	148.864
0.500	1	0	5	0.100	451	175	101	4527	218.287	52.124	167.188	56.842
0.500	1	0	10	0.025	451	175	101	8659	136.611	35.070	387.015	73.812
0.500	1	0	10	0.050	451	175	101	6777	136.681	28.580	225.794	44.946
0.500	1	0	10	0.100	451	175	101	4581	131.251	21.276	64.020	18.978
0.500	1	0	20	0.025	451	175	101	8673	80.370	22.181	116.752	21.563
0.500	1	0	20	0.050	451	175	101	6780	79.831	17.345	77.821	13.242
0.500	1	0	20	0.100	451	175	101	4620	77.378	12.275	31.965	6.961
0.500	1	0.500	5	0.025	451	175	101	12528	322.513	118.525	3601.105	815.600
0.500	1	0.500	5	0.050	451	175	101	10862	320.041	103.856	3232.693	569.751
0.500	1	0.500	5	0.100	451	175	101	8937	318.093	87.953	1533.478	368.890
0.500	1	0.500	10	0.025	451	175	101	12589	175.660	55.164	1629.842	105.437
0.500	1	0.500	10	0.050	451	175	101	10931	175.678	47.293	2083.873	83.514
0.500	1	0.500	10	0.100	451	175	101	8997	174.078	39.101	505.813	70.443
0.500	1	0.500	20	0.025	451	175	101	12540	102.319	35.259	988.595	29.635
0.500	1	0.500	20	0.050	451	175	101	10895	94.220	29.815	332.309	23.979
0.500	1	0.500	20	0.100	451	175	101	8910	101.497	23.781	280.253	17.298

$sparsity$	$size$	$disruption$	h^{step}	p^{pcp}	$\|\mathcal{R}\|$	$\|\mathcal{R}^{rob}\|$	$\|\mathcal{A}\|$	$\|propPairs\|$	preprocessing time (s)	model creating time (s)	optimizing time (s)	first solution time (s)
0.500	1	1	5	0.025	451	175	101	14507	367.036	142.298	3601.497	1444.277
0.500	1	1	5	0.050	451	175	101	12749	366.982	125.830	3601.232	1049.609
0.500	1	1	5	0.100	451	175	101	10993	366.087	108.100	3600.956	628.300
0.500	1	1	10	0.025	451	175	101	14572	199.586	67.927	3600.979	196.112
0.500	1	1	10	0.050	451	175	101	12828	200.769	57.872	3125.061	138.755
0.500	1	1	10	0.100	451	175	101	11069	199.156	49.158	3600.668	84.892
0.500	1	1	20	0.025	451	175	101	14585	116.676	42.559	3602.123	44.128
0.500	1	1	20	0.050	451	175	101	12842	120.593	37.047	2166.988	34.082
0.500	1	1	20	0.100	451	175	101	11038	122.543	32.309	1001.672	23.801
1	0.250	0	5	0.025	258	97	26	3412	134.180	35.764	3602.005	39.883
1	0.250	0	5	0.050	258	97	26	2650	143.806	31.585	3600.345	54.961
1	0.250	0	5	0.100	258	97	26	1785	139.309	25.890	3600.298	23.758
1	0.250	0	10	0.025	258	97	26	3441	79.046	12.032	3600.195	13.478
1	0.250	0	10	0.050	258	97	26	2714	78.104	10.234	3600.191	8.902
1	0.250	0	10	0.100	258	97	26	1790	76.291	8.159	3600.179	9.560
1	0.250	0	20	0.025	258	97	26	3450	45.364	5.346	3600.162	9.320
1	0.250	0	20	0.050	258	97	26	2652	45.224	4.325	3600.099	7.226
1	0.250	0	20	0.100	258	97	26	1865	44.603	3.524	1969.120	4.154
1	0.250	0.500	5	0.025	258	97	26	4253	177.272	45.704	3600.448	38.065
1	0.250	0.500	5	0.050	258	97	26	3623	176.830	40.294	3600.418	15.906
1	0.250	0.500	5	0.100	258	97	26	2928	175.665	34.800	3600.479	35.181
1	0.250	0.500	10	0.025	258	97	26	4258	88.449	15.814	3600.256	30.340
1	0.250	0.500	10	0.050	258	97	26	3658	95.296	14.026	3600.247	18.854
1	0.250	0.500	10	0.100	258	97	26	2953	94.978	11.695	3600.209	17.991
1	0.250	0.500	20	0.025	258	97	26	4275	55.169	7.091	3600.179	9.234
1	0.250	0.500	20	0.050	258	97	26	3615	55.216	5.949	3600.133	9.575
1	0.250	0.500	20	0.100	258	97	26	2995	55.064	5.060	3600.151	9.192
1	0.250	1	5	0.025	258	97	26	5437	209.013	59.172	3600.396	80.364
1	0.250	1	5	0.050	258	97	26	4822	192.498	53.293	3600.447	114.589
1	0.250	1	5	0.100	258	97	26	4140	196.116	47.070	3600.421	138.756
1	0.250	1	10	0.025	258	97	26	5476	104.552	21.398	3600.254	35.402
1	0.250	1	10	0.050	258	97	26	4826	113.149	19.775	3600.277	75.410
1	0.250	1	10	0.100	258	97	26	4147	113.507	16.423	3600.234	134.858
1	0.250	1	20	0.025	258	97	26	5441	65.401	9.586	3600.218	18.508
1	0.250	1	20	0.050	258	97	26	4814	65.556	8.254	3600.200	16.492
1	0.250	1	20	0.100	258	97	26	4135	65.529	7.026	3600.135	9.767
1	0.500	0	5	0.025	470	184	51	10487	261.848	81.146	3601.013	580.381
1	0.500	0	5	0.050	470	184	51	8479	282.683	68.887	3600.686	359.563
1	0.500	0	5	0.100	470	184	51	5841	275.662	55.321	3600.588	512.314
1	0.500	0	10	0.025	470	184	51	10630	153.558	33.419	3600.269	139.161
1	0.500	0	10	0.050	470	184	51	8581	154.057	27.763	3600.361	94.451
1	0.500	0	10	0.100	470	184	51	5954	148.889	21.004	3600.285	45.133
1	0.500	0	20	0.025	470	184	51	10645	89.228	18.902	3600.361	45.825
1	0.500	0	20	0.050	470	184	51	8541	89.459	14.879	3600.312	28.820
1	0.500	0	20	0.100	470	184	51	5957	87.863	10.895	3600.192	14.857
1	0.500	0.500	5	0.025	470	184	51	13666	337.915	115.303	3600.725	1900.757
1	0.500	0.500	5	0.050	470	184	51	11823	323.073	100.590	3600.625	1376.361
1	0.500	0.500	5	0.100	470	184	51	9808	319.097	86.648	3600.597	494.167
1	0.500	0.500	10	0.025	470	184	51	13678	175.604	48.339	3600.360	509.010
1	0.500	0.500	10	0.050	470	184	51	11874	175.825	41.851	3600.285	177.697
1	0.500	0.500	10	0.100	470	184	51	9845	174.215	35.296	3600.623	115.733

$sparsity$	$size$	$disruption$	h^{step}	p^{pcp}	$\|\mathcal{R}\|$	$\|\mathcal{R}^{rob}\|$	$\|\mathcal{A}\|$	$\|propPairs\|$	preprocessing time (s)	model creating time (s)	optimizing time (s)	first solution time (s)
1	0.500	0.500	20	0.025	470	184	51	13645	105.536	27.545	3600.185	85.709
1	0.500	0.500	20	0.050	470	184	51	11848	102.637	23.310	3600.487	54.419
1	0.500	0.500	20	0.100	470	184	51	9883	114.460	19.306	3600.302	35.795
1	0.500	1	5	0.025	470	184	51	16201	381.735	143.361	3600.876	2881.946
1	0.500	1	5	0.050	470	184	51	14579	402.769	127.847	3600.756	2150.661
1	0.500	1	5	0.100	470	184	51	12693	397.189	110.536	3600.678	1635.734
1	0.500	1	10	0.025	470	184	51	16239	217.061	61.028	3600.400	882.221
1	0.500	1	10	0.050	470	184	51	14629	217.144	54.081	3600.393	615.937
1	0.500	1	10	0.100	470	184	51	12705	218.168	46.084	3600.308	197.584
1	0.500	1	20	0.025	470	184	51	16316	127.778	34.701	3600.222	246.764
1	0.500	1	20	0.050	470	184	51	14624	127.458	30.133	3600.232	91.208
1	0.500	1	20	0.100	470	184	51	12702	127.282	25.298	3600.179	58.688
1	1	0	5	0.025	902	355	101	40152	530.286	269.173	3602.028	None
1	1	0	5	0.050	902	355	101	31703	505.926	215.259	3601.225	3514.667
1	1	0	5	0.100	902	355	101	21508	495.680	157.608	3601.772	581.162
1	1	0	10	0.025	902	355	101	40259	276.782	163.410	3600.801	749.811
1	1	0	10	0.050	902	355	101	32080	276.890	123.339	3600.733	338.219
1	1	0	10	0.100	902	355	101	21787	278.045	82.471	3600.490	133.372
1	1	0	20	0.025	902	355	101	40053	165.198	125.535	3600.551	269.684
1	1	0	20	0.050	902	355	101	32185	165.439	93.431	3600.483	115.592
1	1	0	20	0.100	902	355	101	22316	163.584	60.816	3601.237	64.312
1	1	0.500	5	0.025	902	355	101	52271	658.974	408.085	3606.961	None
1	1	0.500	5	0.050	902	355	101	45571	653.635	348.207	3601.602	None
1	1	0.500	5	0.100	902	355	101	38133	637.966	282.982	3601.606	2392.121
1	1	0.500	10	0.025	902	355	101	52517	352.722	247.166	3607.400	2324.515
1	1	0.500	10	0.050	902	355	101	45827	354.084	208.695	3600.951	1133.626
1	1	0.500	10	0.100	902	355	101	38497	359.672	170.414	3601.151	619.397
1	1	0.500	20	0.025	902	355	101	52311	207.123	190.681	3600.816	701.954
1	1	0.500	20	0.050	902	355	101	45685	206.932	159.154	3600.656	347.843
1	1	0.500	20	0.100	902	355	101	38498	205.994	129.282	3600.576	196.509
1	1	1	5	0.025	902	355	101	63768	721.717	535.770	3609.536	None
1	1	1	5	0.050	902	355	101	56228	712.856	455.418	3605.596	None
1	1	1	5	0.100	902	355	101	48548	717.290	385.588	3603.948	None
1	1	1	10	0.025	902	355	101	64197	391.477	330.976	3601.590	3318.057
1	1	1	10	0.050	902	355	101	56380	392.411	277.367	3602.242	2181.179
1	1	1	10	0.100	902	355	101	48661	391.167	228.595	3601.055	1253.526
1	1	1	20	0.025	902	355	101	64004	233.382	258.936	3600.992	1033.265
1	1	1	20	0.050	902	355	101	56360	231.545	211.919	3600.890	530.310
1	1	1	20	0.100	902	355	101	48513	231.501	173.151	3601.228	273.765

A.4.2. Performance experiments

$sparsity$	$size$	$disruption$	h^{step}	p^{pcp}	$\|propPairs\|$	MIP-gap	Simulated costs	Approximation ratio
1	1	0	10	0.050	27329	0.019	0.982	0.977
1	1	0	10	0.100	19936	0.008	0.973	0.951
1	1	0	10	0.200	10966	0.002	0.971	0.903
1	1	0	10	0.350	3963	0.000	0.974	0.861
1	1	0	10	0.500	654	0.000	0.977	0.831
1	1	0	10	0.650	0	0.000	0.976	0.840
1	1	0	20	0.050	27438	0.018	0.980	0.982
1	1	0	20	0.100	19781	0.006	0.971	0.952
1	1	0	20	0.200	11154	0.001	0.971	0.899
1	1	0	20	0.350	4313	0.000	0.974	0.859
1	1	0	20	0.500	963	0.000	0.977	0.835
1	1	0	20	0.650	0	0.000	0.976	0.840
1	1	0	40	0.050	26866	0.011	0.973	0.983
1	1	0	40	0.100	19837	0.006	0.972	0.951
1	1	0	40	0.200	11081	0.002	0.971	0.908
1	1	0	40	0.350	4510	0.000	0.974	0.862
1	1	0	40	0.500	961	0.000	0.977	0.838
1	1	0	40	0.650	106	0.000	0.977	0.834
1	1	0	60	0.050	27327	0.010	0.972	0.990
1	1	0	60	0.100	19815	0.006	0.971	0.959
1	1	0	60	0.200	11619	0.002	0.971	0.904
1	1	0	60	0.350	5545	0.000	0.974	0.864
1	1	0	60	0.500	1177	0.000	0.977	0.834
1	1	0	60	0.650	32	0.000	0.977	0.829
1	1	0	90	0.050	25384	0.011	0.971	1.005
1	1	0	90	0.100	18875	0.008	0.971	0.978
1	1	0	90	0.200	10405	0.003	0.972	0.917
1	1	0	90	0.350	3746	0.000	0.973	0.870
1	1	0	90	0.500	545	0.000	0.976	0.839
1	1	0	90	0.650	22	0.000	0.977	0.835
1	1	0.200	10	0.100	26785	0.028	0.981	0.973
1	1	0.200	10	0.200	18105	0.010	0.970	0.926
1	1	0.200	10	0.350	9770	0.003	0.970	0.870
1	1	0.200	10	0.500	4266	0	0.972	0.828
1	1	0.200	10	0.650	746	0.000	0.977	0.799
1	1	0.200	20	0.100	26611	0.019	0.973	0.968
1	1	0.200	20	0.200	18078	0.010	0.968	0.938
1	1	0.200	20	0.350	9959	0.001	0.968	0.865
1	1	0.200	20	0.500	4621	0.000	0.972	0.830
1	1	0.200	20	0.650	846	0	0.977	0.794
1	1	0.200	40	0.050	32890	0.040	0.992	0.990
1	1	0.200	40	0.100	26442	0.013	0.968	0.971
1	1	0.200	40	0.200	18269	0.009	0.968	0.937
1	1	0.200	40	0.350	10014	0.001	0.967	0.875
1	1	0.200	40	0.500	4973	0.000	0.973	0.828
1	1	0.200	40	0.650	880	0	0.977	0.796
1	1	0.200	60	0.050	33273	0.015	0.967	0.993
1	1	0.200	60	0.100	26547	0.014	0.968	0.977
1	1	0.200	60	0.200	18127	0.008	0.967	0.925
1	1	0.200	60	0.350	10700	0.001	0.966	0.881
1	1	0.200	60	0.500	6199	0.000	0.973	0.826
1	1	0.200	60	0.650	997	0.000	0.977	0.798

$sparsity$	$size$	$disruption$	h^{step}	p^{pcp}	$\lvert propPairs \rvert$	MIP-gap	Simulated costs	Approximation ratio
1	1	0.200	90	0.100	26177	0.015	0.969	0.981
1	1	0.200	90	0.200	17734	0.009	0.967	0.942
1	1	0.200	90	0.350	8895	0.004	0.967	0.891
1	1	0.200	90	0.500	3731	0.000	0.970	0.841
1	1	0.200	90	0.650	741	0.000	0.976	0.802
1	1	0.400	10	0.200	25400	0.026	0.973	0.952
1	1	0.400	10	0.350	18292	0.013	0.969	0.892
1	1	0.400	10	0.500	12099	0.001	0.967	0.829
1	1	0.400	10	0.650	5466	0.001	0.972	0.797
1	1	0.400	20	0.050	37239	0.032	0.974	0.992
1	1	0.400	20	0.100	31944	0.026	0.970	0.978
1	1	0.400	20	0.200	25429	0.018	0.968	0.943
1	1	0.400	20	0.350	18091	0.014	0.967	0.913
1	1	0.400	20	0.500	11750	0.001	0.966	0.831
1	1	0.400	20	0.650	5619	0.000	0.973	0.789
1	1	0.400	40	0.050	37249	0.027	0.969	0.997
1	1	0.400	40	0.100	31859	0.025	0.968	0.984
1	1	0.400	40	0.200	24967	0.022	0.969	0.958
1	1	0.400	40	0.350	18201	0.008	0.964	0.893
1	1	0.400	40	0.500	11552	0.002	0.967	0.839
1	1	0.400	40	0.650	4727	0.000	0.973	0.790
1	1	0.400	60	0.050	37135	0.032	0.973	1.003
1	1	0.400	60	0.100	32305	0.034	0.976	0.992
1	1	0.400	60	0.200	25147	0.033	0.980	0.953
1	1	0.400	60	0.350	17562	0.008	0.965	0.888
1	1	0.400	60	0.500	10594	0.002	0.968	0.828
1	1	0.400	60	0.650	4555	0.000	0.972	0.796
1	1	0.400	90	0.050	37450	0.042	0.979	1.024
1	1	0.400	90	0.100	32111	0.026	0.967	1.005
1	1	0.400	90	0.200	26166	0.022	0.966	0.980
1	1	0.400	90	0.350	16966	0.010	0.965	0.899
1	1	0.400	90	0.500	10581	0.003	0.967	0.844
1	1	0.400	90	0.650	5082	0.000	0.971	0.804
1	1	0.600	10	0.350	23189	0.036	0.979	0.927
1	1	0.600	10	0.500	16960	0.014	0.969	0.851
1	1	0.600	10	0.650	10324	0.001	0.968	0.789
1	1	0.600	20	0.050	40454	0.044	0.975	0.998
1	1	0.600	20	0.200	29856	0.035	0.970	0.970
1	1	0.600	20	0.350	22918	0.027	0.971	0.915
1	1	0.600	20	0.500	17005	0.015	0.969	0.864
1	1	0.600	20	0.650	10470	0.001	0.970	0.782
1	1	0.600	40	0.050	40188	0.047	0.977	1.000
1	1	0.600	40	0.200	29658	0.056	0.991	0.974
1	1	0.600	40	0.350	23135	0.042	0.983	0.932
1	1	0.600	40	0.500	17129	0.010	0.962	0.869
1	1	0.600	40	0.650	10671	0.002	0.968	0.796
1	1	0.600	60	0.100	35146	0.039	0.970	0.994
1	1	0.600	60	0.200	29649	0.040	0.975	0.977
1	1	0.600	60	0.350	22575	0.022	0.965	0.927
1	1	0.600	60	0.500	16654	0.010	0.967	0.846
1	1	0.600	60	0.650	8381	0.001	0.968	0.791
1	1	0.600	90	0.050	39846	0.056	0.983	1.022
1	1	0.600	90	0.100	34963	0.036	0.965	1.008
1	1	0.600	90	0.200	29005	0.032	0.964	0.991
1	1	0.600	90	0.350	22297	0.035	0.972	0.958
1	1	0.600	90	0.500	15045	0.011	0.963	0.873
1	1	0.600	90	0.650	8360	0.002	0.968	0.799

$sparsity$	$size$	$disruption$	h^{step}	pp^{cp}	$\|propPairs\|$	MIP-gap	Simulated costs	Approximation ratio
1	1	0.800	10	0.350	23803	0.040	0.972	0.925
1	1	0.800	10	0.500	16915	0.032	0.981	0.845
1	1	0.800	10	0.650	9922	0.000	0.968	0.760
1	1	0.800	20	0.200	30448	0.049	0.974	0.969
1	1	0.800	20	0.350	23713	0.026	0.960	0.917
1	1	0.800	20	0.500	17221	0.017	0.963	0.859
1	1	0.800	20	0.650	9961	0.001	0.965	0.770
1	1	0.800	40	0.200	30265	0.036	0.961	0.962
1	1	0.800	40	0.350	23300	0.048	0.979	0.933
1	1	0.800	40	0.500	16834	0.015	0.965	0.844
1	1	0.800	40	0.650	9562	0.002	0.969	0.762
1	1	0.800	60	0.050	41971	0.052	0.970	1.011
1	1	0.800	60	0.100	36911	0.055	0.974	1.002
1	1	0.800	60	0.200	30459	0.053	0.978	0.970
1	1	0.800	60	0.350	23741	0.043	0.977	0.921
1	1	0.800	60	0.500	17146	0.013	0.962	0.845
1	1	0.800	60	0.650	8471	0.001	0.969	0.758
1	1	0.800	90	0.100	36603	0.070	0.986	1.021
1	1	0.800	90	0.200	29239	0.041	0.964	0.983
1	1	0.800	90	0.350	23370	0.043	0.974	0.934
1	1	0.800	90	0.500	16794	0.012	0.960	0.850
1	1	0.800	90	0.650	7779	0.001	0.965	0.773
1	1	1	10	0.500	21975	0.031	0.962	0.861
1	1	1	10	0.650	14508	0.027	0.977	0.779
1	1	1	20	0.200	34757	0.063	0.969	0.974
1	1	1	20	0.350	28329	0.060	0.974	0.935
1	1	1	20	0.500	21855	0.040	0.968	0.874
1	1	1	20	0.650	14314	0.012	0.964	0.776
1	1	1	40	0.200	34837	0.058	0.964	0.972
1	1	1	40	0.350	27837	0.047	0.963	0.927
1	1	1	40	0.500	22178	0.031	0.961	0.866
1	1	1	40	0.650	13634	0.009	0.960	0.779
1	1	1	60	0.050	46273	0.067	0.967	1.010
1	1	1	60	0.350	28136	0.055	0.971	0.927
1	1	1	60	0.500	20908	0.025	0.957	0.861
1	1	1	60	0.650	12508	0.006	0.961	0.762
1	1	1	90	0.350	27227	0.046	0.958	0.943
1	1	1	90	0.500	21616	0.054	0.974	0.915
1	1	1	90	0.650	14162	0.010	0.960	0.783

$sparsity$	$size$	$disruption$	h^{step}	p^{pcp}	$\|propPairs\|$	MIP-gap	Simulated costs	Approximation ratio
1	0.200	0	5	0.0125	2005	0	0.980	0.972
1	0.200	0	5	0.025	1686	0	0.980	0.970
1	0.200	0	5	0.050	1294	0	0.980	0.954
1	0.200	0	5	0.100	850	0	0.981	0.925
1	0.200	0	5	0.200	449	0	0.982	0.877
1	0.200	0	5	0.350	127	0	0.985	0.845
1	0.200	0	10	0.0125	2017	0	0.980	0.974
1	0.200	0	10	0.025	1706	0	0.980	0.972
1	0.200	0	10	0.050	1303	0	0.980	0.958
1	0.200	0	10	0.100	841	0	0.980	0.940
1	0.200	0	10	0.200	469	0	0.982	0.880
1	0.200	0	10	0.350	159	0	0.985	0.846
1	0.200	0	20	0.0125	2018	0	0.980	0.975
1	0.200	0	20	0.025	1673	0	0.980	0.973
1	0.200	0	20	0.050	1262	0	0.980	0.956
1	0.200	0	20	0.100	806	0	0.980	0.926
1	0.200	0	20	0.200	409	0	0.983	0.886
1	0.200	0	20	0.350	162	0	0.984	0.859
1	0.200	0	40	0.0125	1962	0	0.980	0.982
1	0.200	0	40	0.025	1657	0	0.980	0.978
1	0.200	0	40	0.050	1289	0	0.980	0.964
1	0.200	0	40	0.100	818	0	0.980	0.940
1	0.200	0	40	0.200	423	0	0.983	0.882
1	0.200	0	40	0.350	163	0	0.983	0.868
1	0.200	0	60	0.0125	2025	0	0.979	0.991
1	0.200	0	60	0.025	1702	0	0.980	0.985
1	0.200	0	60	0.050	1262	0	0.980	0.968
1	0.200	0	60	0.100	815	0	0.981	0.918
1	0.200	0	60	0.200	484	0	0.983	0.879
1	0.200	0	60	0.350	182	0	0.985	0.847

References

[1] URL: https://airlineoperations.ai/digital-transformation/.

[2] Shervin Ahmadbeygi, Amy Cohn, and Marcial Lapp. "Decreasing airline delay propagation by re-allocating scheduled slack". In: *IIE transactions* 42.7 (2010), pp. 478–489.

[3] The World Bank. *Air transport, passengers carried*. URL: https://data.worldbank.org/indicator/IS.AIR.PSGR.

[4] Aharon Ben-Tal and Arkadi Nemirovski. "Robust convex optimization". In: *Mathematics of operations research* 23.4 (1998), pp. 769–805.

[5] Dimitris Bertsimas and Melvyn Sim. "The price of robustness". In: *Operations research* 52.1 (2004), pp. 35–53.

[6] Ralf Borndörfer et al. "Robust tail assignment". In: (2010).

[7] Edmund K Burke et al. "A multi-objective approach for robust airline scheduling". In: *Computers & Operations Research* 37.5 (2010), pp. 822–832.

[8] Lloyd Clarke et al. "The aircraft rotation problem". In: *Annals of Operations Research* 69 (1997), pp. 33–46.

[9] Guy Desaulniers et al. "Daily aircraft routing and scheduling". In: *Management Science* 43.6 (1997), pp. 841–855.

[10] Ivan Dovica. "Robust tail assignment". In: (2014).

[11] Eurocontrol. "Costs of air transport delay in europe". In: *Technical report, EUROCONTROL* (2000).

[12] Thomas A Feo and Jonathan F Bard. "Flight scheduling and maintenance base planning". In: *Management Science* 35.12 (1989), pp. 1415–1432.

[13] Michael R Garey and David S Johnson. "Computers and intractability". In: *A Guide to the Theory of NP-completeness* (1979).

[14] Mattias Grönkvist. *The tail assignment problem*. Chalmers tekniska högskola, 2005.

[15] Air Transport Action Group. *Aviation: Benefits Beyond Borders*. Oct. 2018. URL: https://www.aviationbenefits.org/media/166344/abbb18_full-report_web.pdf.

[16] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2021. URL: https://www.gurobi.com.

[17] Nader M Kabbani and Bruce W Patty. "Aircraft routing at American airlines." In: *proceedings of the agifors symposium*. 1992.

[18] Jay M Rosenberger, Ellis L Johnson, and George L Nemhauser. "A robust fleet-assignment model with hub isolation and short cycles". In: *Transportation science* 38.3 (2004), pp. 357–368.

[19] Allen L Soyster. "Convex programming with set-inclusive constraints and applications to inexact linear programming". In: *Operations research* 21.5 (1973), pp. 1154–1157.

[20] Maryam Farshchian Yazdi et al. "Flight delay prediction based on deep learning and Levenberg-Marquart algorithm". In: *Journal of Big Data* 7.1 (Nov. 2020), p. 106. ISSN: 2196-1115. DOI: 10.1186/s40537-020-00380-z. URL: https://doi.org/10.1186/s40537-020-00380-z.

Index