Robust Tail Assignment

vorgelegt von

Mgr Ivan Dovica

aus Rožňava Slowakei

Von der Fakultät II – Mathematik und Naturwissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
– Dr. rer. nat. –

genehmigte Dissertation

Promotionsausschuss
Vorsitzender: Prof. Dr. Peter Bank
Berichter: Prof. Dr. Dr. h.c. mult. Martin Grötschel
Prof. Dr. Ralf Borndörfer

Tag der wissenschaftlichen Aussprache: 28.7.2014

Berlin 2014
D 83

Zusammenfassung

Der erste Teil dieser Arbeit behandelt das allgemeine stochastische „Kürzeste-Wege-Problem“. Gesucht ist der kürzeste Weg in einem Graphen, in dem die Bogenlängen ungewiss und als kontinuierliche Zufallsvariablen definiert sind. Dieses Problem steht im Zentrum verschiedener Anwendungen, vor allem in der robusten Verkehrsplanung, bei der die jeweiligen Wege den Flugzeug-, Zug- oder Busumläufen oder den Dienstplänen der Mitarbeiter entsprechen. Wir schlagen eine neue Lösungsmethode vor, die auf der Diskretisierung von kontinuierlichen Zufallsvariablen basiert. Die Methode ist anwendbar auf jede Klasse von kontinuierlichen Zufallsvariablen. Wir liefern Abschätzungen für den Approximationsfehler der diskretisierten Weglängen im Vergleich mit den kontinuierlichen Weglängen. Des Weiteren präsentieren wir theoretische Ergebnisse zum asymptotischen Laufzeitverhalten der Methode.

Im zweiten Abschnitt wenden wir diese Methode für ein reales Praxisproblem im Flugverkehr an: auf das sogenannte „Tail-Assignment-Problem“. Das Ziel des „Tail-Assignment-Problems“ ist es, die Flugzeugumläufe, d. h. die Routen, bestehend aus Flugsegmenten, für eine Menge von individuellen Flugzeugen so zu erzeugen, dass eine Menge von gegebenen Flügen unter Berücksichtigung der Betriebsbedingungen von jedem einzelnen Flugzeug und unter Beachtung der kurz- bis langfristigen individuellen Wartungserfordernisse überdeckt werden.

Wir stellen eine stochastische Formulierung dieses Problems vor und zeigen, wie unsere Methode dieses Problem innerhalb eines Spaltenerzeugungs - Frameworks effizient löst. Wir zeigen den Vorteil unseres stochastischen Ansatzes gegenüber dem KPI-Ansatz durch den Vergleich der Propagation der Verspätungen. Es zeigt sich, dass unser Ansatz zu weniger Betriebskosten ohne Erhöhung der Berechnungskomplexität führt.

Eine Kernidee unserer Vorgehensweise zur Lösung des Problems der Robusten Optimierung ist das Anpassen des zugrundeliegenden stochastischen Modells auf komplexe Problemstellungen aus der Realität. Wir schlagen ein Modell von Verspätungspropagation vor, das realistisch und gleichzeitig einfach anwendbar, und deswegen ideal für Vorhersagen einsetzbar ist.

Wir bewerten unsere Ergebnisse durch umfassende Simulationen. Wir zeigen eine erhebliche Verringerung von Ankunftsverspätungen und damit auch von Kosten für den Durchschnittsfall sowie für eine Vielzahl der Szenarien. Wir evaluieren diese Verbesserung in anderen realistischen Benchmarks durch

Simulation unter Einbeziehung von Gegenmaßnahmen (Rettungsmaßnahmen) und in Szenarien, basierend auf historischen Verspätungsdaten anstelle des stochastischen Modells.

Abstract

The first part of this thesis is devoted to the general problem of stochastic shortest path problem. It is about searching for the shortest path in a graph in which arc lengths are uncertain and specified by continuous random variables. This problem is at the core of various applications, especially in robust transportation planning where paths correspond to aircraft, train, or bus rotations, crew duties or rosters, etc. We propose a novel solution method based on a discretisation of random variables which is applicable to any class of continuous random variables.

We also give bounds on the approximation error of the discretised path lengths compared to the continuous path lengths. In addition, we provide theoretical results for the computational complexity of this method.

In the second part we apply this method to a real world airline transportation problem: the so-called tail assignment problem. The goal of the tail assignment problem is to construct aircraft rotations, routes consisting of flight segments, for a set of individual aircraft in order to cover a set of flight segments (legs) while considering operational constraints of each individual aircraft as well as short- to long-term individual maintenance requirements.

We state a stochastic programming formulation of this problem and we show how to solve it efficiently by using our method within a column generation framework. We show the gain of our stochastic approach in comparison to standard KPI in terms of less propagated delay and thus less operational costs without growth of computational complexity.

A key point of our complex approach to robust optimisation problem is the fit of the underlying stochastic model with reality. We propose a delay propagation model that is realistic, not overfitted, and can therefore be used for forecasting purposes.

We benchmark our results using extensive simulation. We show a significant decrease of arrival delays and thus monetary savings on average as well as in the majority of our disruption scenarios. We confirm these benefits in even more life-like benchmarks as simulation where recovery actions are taken and in scenarios which use historical delays directly instead of the stochastic model.

Acknowledgements

I would like to thank Prof. Dr. Dr. h.c. mult. Martin Grötschel for giving me the opportunity to work at ZIB. I am very thankful to Prof. Ralf Borndörfer for his supervision, support, suggestions, and all discussions regarding research and non-research topics. I am also very grateful to Lufthansa Systems team, especially Ivo Nowak and Thomas Schickinger, for their cooperation on the Robust Tail Assignment project and fruitful discussions on meetings.

I would like to thank Carlos and Dung for proof-reading parts of my thesis; Miška and Miloš for grammar corrections; and Benjamin, Thomas, Marcus, and Ambros for help with translations to German during my time at ZIB. Special thanks goes to Bettina for helping me with many day-to-day problems that often seemed to be harder than research itself. I would like to thank also all colleagues at ZIB for creating great working atmosphere that made my time at Berlin so great.

Last but not least I want to thank my wife Tereza and my parents for their endless patience and support.

Contents

Introduction

Eurocontrol STATFOR (2010) [52] reports that in the most-likely future scenario European airports will serve about 80% more flight legs in 2030 than in 2009. This corresponds to an average annual growth of 2.8% while many European airports are nowadays already reaching their capacity limits. Traffic growth will increase congestion on airports and thus increase delay generation.

Highly congested airports and airspaces are more likely to cause delays and are less flexible in accommodating unexpected traffic (caused by delayed legs). This is even more true as slack times in aircraft and staff utilization have been reduced by continuous advances in the optimisation techniques. All these factors contribute to delay increase, annoy, and shoot up operational costs of airlines.

Figure 1 demonstrates this fact on data derived from annual reports of Central Office for Data Analysis (CODA) for the years 2003 to 2012, see CODA [35]. We observe correlation between air traffic volume and the share of delayed legs on arrival. After years of traffic and delay growth, there is a steep decrease during the economical crisis in 2008 and 2009. The discrepancy in the 2010 and 2011 data development is caused by an extraordinary growth of delays in 2010 due to the ash cloud caused by the volcano eruption on Iceland and poor weather.

Clearly, delays are causing huge financial losses to airlines. Therefore, they are subject of massive investigation with the goal to get delays and financial impacts under control. Ball et al. (2010) [11] estimate the total cost of all US air transportation delays in 2007 at $32.9 billion, of which direct airline costs are estimated at $8.3 billion and the value of passengers' lost time at $16.7 billion. Based on European data, Cook, Tanner & Anderson (2004)

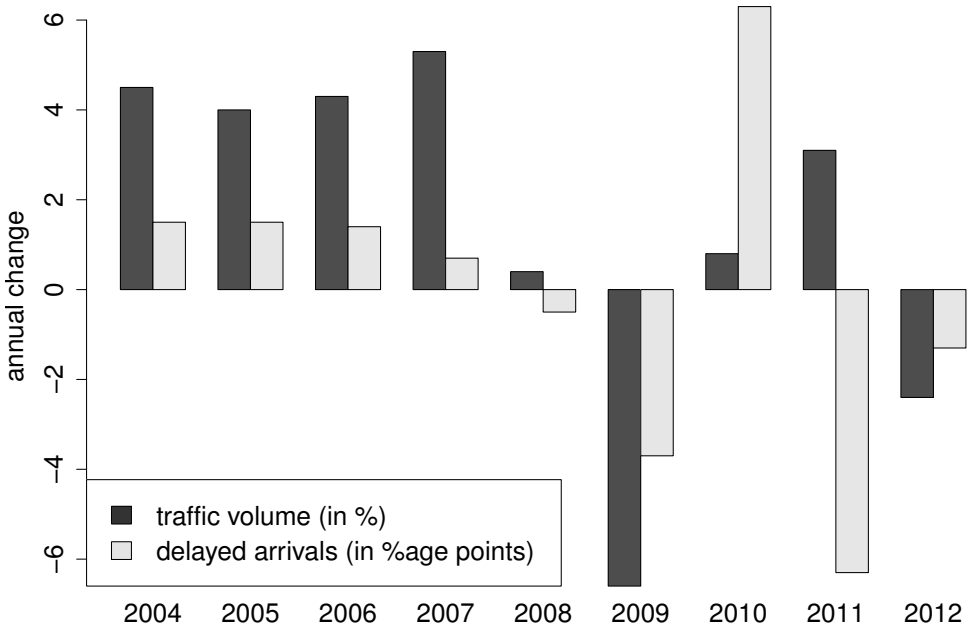


Figure 1: Comparison of the annual growth of air traffic volume (measured by number of operated legs) on European airports and the increase of legs delayed at least 15 minutes on arrival (measured in absolute changes in percent).

[38] estimate the average cost of one minute of delay longer than 15 minutes at 72 €.

These figures have consequences for airline scheduling. The growth of operational costs in comparison to the decrease in scheduled costs led airlines to rethink their perception of the "ideal schedule". They now ask for schedules that are cost effective in the real operation, under delays, and not only "on the paper". Such schedules are called robust schedules.

The last decade brought various approaches to robust scheduling in all planning stages from the fleet assignment to tail assignment. The easiest way, from a mathematical point of view, is to improve robustness by incorporating some additional robustness measure, called key performance indicator (KPI), into the current optimisation objectives. KPIs assure favourable properties of

the schedule that are supposed to improve its performance under disruptions.

Clearly, different scheduling problems desire different KPIs. Typical KPIs are for example, minimising the total buffer for each rotation, stipulating some threshold ground buffer before each leg, avoiding visits of single aircraft on certain airports too often during one day, minimising the number of aircraft types serving each hub, avoiding crews changing the aircraft etc.

KPI approaches are typically simple to implement. Their disadvantage is that there is no clear connection between the KPI and increased leg punctuality, decreased propagated delay, or additional expenses. Hence, the benefit of a KPI approach is disputable or easily be negative.

In order to achieve better results on a solid scientific basis, one has to come up with more accurate and rigorous models which have to cope with disruptions directly and which asses their impact directly.

It is very important to properly model disruptions and their impact on the whole network. Unfortunately, the computation of delay propagation is difficult even for small sized problems. Approaches from the literature are too slow for practical applications, have disputable accuracy of results, or apply only under special conditions that are not fulfilled in practice.

Contribution

In this thesis, we propose a novel method to compute the length of a path in a graph whose arc lengths are subject to uncertainty and specified by random variables. This stochastic shortest path problem is widely applicable in transportation planning problems in which uncertainties of transfer durations. We focus on a concrete application in airline planning, namely, the tail assignment problem. We consider the optimisation of natural objectives such as the expected arrival delay, the probability of delay propagation, and the expected additional airline costs.

The basis of this method is an approximation of continuous functions, representing probability density functions of continuous random variables, by step functions. This is kind of discretisation.

We provide an effective and accurate implementation of operations that are required for computing of delay propagations in an airline network. Furthermore, the discretisation is applicable to any class of continuous distributions.

Therefore, the method can be applied to any suitable problem independently of the choice of the stochastic model. It also ensures a high accuracy of the underlying stochastic model.

Our discretisation method is particularly suited for problems which are solved by the column generation framework. We demonstrate its application at the example of the tail assignment problem, which deals with the generation of aircraft rotations a few days before the day of operation. We formulate a stochastic programming model for the robust tail assignment problem in which we minimise the total probability of delay propagation. Due to the discretisation method we are able to solve this model effectively by a deterministic algorithm.

Furthermore, we develop a simple but precise stochastic model of airline operations. The model has only very few parameters. It therefore can be calibrated using rather small data sets.

We pay particular attention to an assessment of the benefits of our approach under as realistic conditions as possible.

To this purpose, we develop a framework for simulation and evaluation of aircraft schedules. Our simulator allows an evaluation of schedules under randomly generated disruptions sampled using our stochastic model. The simulator is able to perform recovery actions; this brings simulation even closer to reality. Finally, we use the simulator for the visualisation of schedules and simulation results.

Our results were awarded with the Anna Valicek Award 2010 of the Airline Group of the International Federation of Operations Research Societies (AGIFORS) which recognises original and innovative research in the application of operations research to airline and/or airline related business problems.

Thesis Outline

In Chapter 1 we investigate a classical problem of the graph theory, the shortest path problem. It appears in many applications in transportation. We formulate a stochastic version of this problem in which arcs have stochastic lengths. We call this problem the stochastic shortest path problem.

In order to solve the stochastic shortest path problem, we propose a discretisation method for continuous random variables. We study algorithmic

computation of operations such as convolution, expectation, and stochastic dominance over discretised random variables. Furthermore, we provide a theoretical bound on the approximation error of the method.

Chapters 2-7 are dedicated to the application of the discretisation method to robust airline planning. Chapter 2, we survey current planning practices of airlines and give a survey of the field of robust airline planning.

In Chapter 3, we study how delay propagates in airline networks. We formalise this process in a mathematical model and we show how to apply the discretisation method to the computation of the delay propagation in this model. We extend the formal definition of the method from Chapter 1, which applies to general graphs, in order to make it applicable to airline networks. We also provide implementations of all stochastic operations required. The efficiency of the method is demonstrated in a computational study which demonstrates low computation times and high accuracy when applied to the computation of delay propagation in realistic aircraft rotations.

In Chapter 4 we formulate a robust version of the tail assignment problem. We give a stochastic programming formulation which minimises the probability of delay propagation across the network. In order to solve the model, we use results from the previous chapter. Namely, the probability of delay propagation, and thus the computation of the propagated delay, along aircraft rotation is required in the pricing problem of a column generation approach.

Furthermore, we state a two-stage stochastic programming model in order to solve the robust tail assignment problem with consideration of non-rotational delay propagation. Such delays are caused, for example, by crew changing the aircraft, transferring passengers, or cargo. We propose an approximate column generation algorithm for solving this problem. The algorithm, again, relies on the discretisation method in solving the pricing problem.

In Chapter 5, we present details of our stochastic model of primary delay generation. The model is derived from historical operational data. It focuses simplicity and plausibility. The model is used in our optimisation approach as well as in our simulator.

Chapter 6 documents the simulation and visualisation tool Ops Simulator. The simulator is an important part of this work since it allows to estimate the practical gain of our approach over KPI methods in an lifelike environment.

In Chapter 7, we present computational results for real world instances of the robust tail assignment problem. Using the Ops Simulator, we document a genuine decrease in operational costs.

Chapter 1

The Stochastic Shortest Path Problem

The goal of this chapter is to present an algorithm for solving the stochastic shortest path problem. The stochastic shortest path problem can be seen as a generalisation of the shortest path problem where arc lengths are random variables.

First in Section 1.1, we give an overview of known formulations of the stochastic shortest path problem from the literature and we discuss their algorithmic complexity. We state a general formulation of the problem. That is, we do not take any assumptions on the arc length distributions and we assume the cost function may be any non-decreasing function. In Section 1.2, we propose a labelling algorithm for solving this formulation.

An implementation of the algorithm relies on a novel discretisation technique for representation of continuous random variables which is presented in Section 1.3. In Section 1.4, we define all essential operations over discretised random variables required for implementation of labelling algorithm. Moreover in Section 1.5, we estimate an approximation error of distribution of the path length introduced through employment of the discretisation technique.

Discretisation method plays central role in this work. In Chapter 3, we provide an extension of our discretisation method for computation of the propagated delay along aircraft rotations and we provide details about its application. Such extension is then applied for solving the robust tail assignment, which is introduced in Chapter 4.

1.1 Problem Definition

The shortest path problem is one of the most important problems in Graph Theory. Many practical problems can be formulated as the shortest path problem or contain this problem as a sub-problem. Several efficient algorithms have been proposed to solve this problem.

However, recent advances in applied mathematics ask for more realistic modelling of practical problems which lead to generalisations of this problem towards reflecting uncertainty of the parameters. Input data for the mathematical models are often noisy, unknown, or have stochastic nature. Such data, thus, have to be predicted by random variables.

For example, travel time between two cities depends on congestion of the road, which may be predicted by a random variable. In the context of the shortest path problem this idea leads to the definition of the stochastic shortest path problem where arc lengths, representing distances between cities, are random variables.

Formally, the stochastic shortest path problem can be stated as follows. Let $G = (V, E)$ be a graph with set of vertices V, arcs E, and two distinguished vertices $s, t \in V$. Arc lengths $X_e, e \in E$ are real-valued random variables. We are looking for the shortest st-path in the graph. So defined stochastic shortest path problem finds application in many practical problems. For all we mention application in public transportation planning, see for instance Borndörfer et al. (2010) [26].

In context of the stochastic shortest path problem, definition of the optimal (or "'shortest"') path is disputable and depends on specific application. Moreover, the objective function has impact on solvability of the problem. The literature proposes various formulations of the problem and complexity results for the formulations.

The most natural seems to be to minimise expected length of the path. Although let us have, in the aforementioned example, given some deadline for arriving to destination city (or vertex t). Minimisation of expected travel duration (or path length) does not make sense since we rather want to find a path which gets us to destination city (or vertex t) most likely before the deadline. Frank (1969) [56] proposes this optimality criterion, that is, the objective function is to identify the path with the highest probability to have weight smaller than some given threshold. Another optimality criterion

propose Sigal, Pritsker & Solberg (1980) [98]. They search for the path that is the shortest of all paths with the highest probability.

Most optimality criteria can be formally defined by some cost function w defined on the path length. Then the goal is to minimise expected value of the cost function. Whether the problem can be solved efficiently or not depends on the cost function. Nikolova, Brand & Karger (2006) [85] prove that it is NP-hard to find the path with the lowest expected cost for general cost functions with a global minimum.

On the other hand, there are known cases when the problem is efficiently solvable. For example, minimisation of expectation of linear cost function is efficiently solvable since it holds

$$\mathbb{E}\left(w\left(X_{e_1}+X_{e_2}\right)\right)=\mathbb{E}\left(w\left(X_{e_1}\right)\right)+\mathbb{E}\left(w\left(X_{e_2}\right)\right).$$

Thus, one can replace arc lengths X_e by their expected costs $\mathbb{E}\left(w\left(X_e\right)\right)$ and solve the corresponding deterministic shortest path problem by standard algorithms.

Efficient dynamic programming algorithm is also applicable for the exponential cost function, see Eiger, Mirchandani & Soroush (1985) [49] and Loui (1983) [80]. Eiger, Mirchandani & Soroush (1985) [49] prove validity of Bellman's Principle of Optimality for the exponential cost function. They show that if expected cost of some path p^1_{sv} from s to v is smaller than of path p^2_{sv}, then the same holds for any extensions of these paths. This follows from non-negativity of $\mathbb{E}\left(w\left(X_e\right)\right)$ and from the fact that for exponential w

$$\mathbb{E}\left(w\left(X_{p^1_{sv}}+X_e\right)\right)=\mathbb{E}\left(w\left(X_{p^1_{sv}}\right)\right)\mathbb{E}\left(w\left(X_e\right)\right).$$

We consider the problem in very general setting where the expected value of some non-decreasing cost function is minimised. Such formulation includes many of aforementioned formulations as a special case. In order to avoid negative cycles in the graph, we restrict ourselves to directed acyclic graphs. This formulation is also satisfactory for the application of the problem in following chapters. Another possibility how to avoid negative cycles is to require non-negative edge lengths and positive cost function instead. Definition 1.1.4 states studied problem.

Definition 1.1.1. *Given a probability space $(\Omega,\mathcal{F},P)$ where Ω denotes a sample space, $\mathcal{F}\subset 2^\Omega$ is a set of events, and $P:\mathcal{F}\to[0,1]$ is a probability measure; a random variable X is a measurable function $X:\Omega\to\mathbb{R}^+$*

v

Definition 1.1.2. *The probability density function of a random variable X is the function $f_X : \mathbb{R} \to [0, 1]$ where*

$$P[a \leq X \leq b] = \int_a^b f_X(x)dx.$$

Definition 1.1.3. *The cumulative density function of a random variable X is the function $F_X : \mathbb{R} \to [0, 1]$ where*

$$F_X(x) = P[X \leq x].$$

Definition 1.1.4 (Stochastic Shortest Path Problem)**.** *Let $G = (V, E)$ be an acyclic directed graph with set of vertices V, set of arcs E, and two distinguished vertices $s, t \in V$. Given real-valued independent continuous random variables $X_e, e \in E$, of arc lengths with known probability density functions, and non-decreasing cost function w, we want to find the st-path p^*_{st} with the smallest expected cost*

$$p^*_{st} = \min_{p \in P_{st}} \mathbb{E}\left(w\left(X_p\right)\right)$$

where $X_p = \sum_{e \in p} X_e$ and P_{st} is the set of all directed paths p from s to t.

1.2 Labelling Algorithm

Loui (1983) [80] proves that the formulation of the stochastic shortest path problem, where expectation of non-decreasing cost function is minimised, cannot be efficiently solved since Bellman's Principle of Optimality does not hold. This principle is inevitable assumption for development of an efficient dynamic programming algorithm.

Loui (1983) [80], therefore, proposes the labelling algorithm for solving two special cases of the stochastic shortest path problem. The first one with the quadratic cost function, and the second with arc length distributions closed under convolution and uniquely determined by the first two moments. In both cases the cost of every path length can be computed easily from arc length distributions and is determined by the first two moments. Thus, the first and the second moment of the path length are used as the labels in the labelling algorithm. Murthy & Sarkar (1996) [84] enhance the performance

Algorithm 1: Labelling algorithm for solving the stochastic shortest path problem

Input : Digraph $G = (V, E)$ with random variables for arc lengths $X_e, e \in E$, non-decreasing cost function w, and two vertices $s, t \in V$

Output: st-path with the smallest expected cost

```
1  L_s := {(0)};
2  L_v := ∅ for all v ∈ V \ {s};
3  sort V topologically as v_1, ..., v_j = s, ..., v_k = t, ..., v_n;
4  for i := j to k − 1 do
5      u := v_i;
6      remove all dominated labels from L_u by Dominance Rule 1.2.1;
7      for (u, v) ∈ E do
8          for (X_{p_su}) ∈ L_u do
9              X_{p_sv} := X_{p_su} + X_uv;
10             L_v := L_v ∪ {X_{p_sv}};
11 return arg min_{p_st : X_{p_st} ∈ L_t} E(w(X_{p_st}));
```

of the labelling algorithm for the concave quadratic cost function by the pruning technique what decreases number of enumerated labels.

In our problem formulation, we do not take any assumptions on the arcs length distributions and on the cost function. Thus, we have to introduce more general labels and dominance rule for the labelling algorithm. Moreover, we have to tackle a problem of summing general continuous random variables in order to derive distributions of path lengths.

Our approach is described by Algorithm 1. We propose the labelling algorithm that uses full description of the path distribution in the labels. We assign sets L_v of labels $X_{p_{sv}}$ to all vertices. Each label set L_v stores probability distribution functions of the length of paths p_{sv} from vertex s to vertex v. We guarantee finiteness of the algorithm by examination of the vertices in topological order. We recall that topological order is a linear order of vertices with no backward arcs. In order to save some running time over pure enumeration, we prune labels of all dominated paths in Step (6) by Dominance Rule 1.2.1. Correctness of the dominance rule proves Theorem 1.2.2.

Definition 1.2.1 (Stochastic Dominance Rule)**.** *Let p_1 and p_2 be two sv-paths with lengths $X_{sv}^{p_1}$ and $X_{sv}^{p_2}$, respectively. Denote $F_{X_{sv}^{p_1}}$ and $F_{X_{sv}^{p_2}}$ cumulative density function of random variable $X_{sv}^{p_1}$ and $X_{sv}^{p_2}$, respectively. We say, that path p_2 dominates path p_1 if for all $x \in \mathbb{R}$*

$$F_{X_{sv}^{p_1}}(x) \leq F_{X_{sv}^{p_2}}(x).$$

Theorem 1.2.2 (Correctness of Dominance Rule)**.** *Let w be non-decreasing cost function, and let us have two sv-paths p_1 and p_2 with lengths represented by random variables $X_{sv}^{p_1}$ and $X_{sv}^{p_2}$, respectively. Denote $F_{X_{sv}^{p_1}}$ and $F_{X_{sv}^{p_2}}$ cumulative density functions of $X_{sv}^{p_1}$ and $X_{sv}^{p_2}$, respectively. If for all $x \in \mathbb{R}$*

$$F_{X_{sv}^{p_1}}(x) \leq F_{X_{sv}^{p_2}}(x),$$

then path p_1 is suboptimal, that is, it is not part of any st-path with the smallest expected cost.

Proof. Let us denote p_3 any vt-path. First, we show that if p_2 dominates p_1, then for extension of paths p_1 and p_2 along path p_3 to vertex t holds $F_{X_{st}^{p_1+p_3}}(x) \leq F_{X_{st}^{p_2+p_3}}(x)$, for all $x \in \mathbb{R}$. In other words, any extension of path p_1 is dominated by the same extension of path p_2.

Then we show that every dominated path is suboptimal, that is,

$$F_{X_{sv}^{p_1}}(x) \leq F_{X_{sv}^{p_2}}(x), \forall x \in \mathbb{R} \quad \Rightarrow \quad \mathbb{E}\left(w\left(X_{sv}^{p_1}\right)\right) \geq \mathbb{E}\left(w\left(X_{sv}^{p_2}\right)\right).$$

These two facts prove that there is always extension of p_2 that has lower costs than corresponding extension of p_1 and therefore path p_1 can be pruned.

Let us pick some vt-path and denote $X_{vt}^{p_3}$ random variable of the length of this path, that is,

$$X_{vt}^{p_3} = \sum_{e \in p_3} X_e.$$

Then $X_{st}^{p_1+p_3} = X_{sv}^{p_1} + X_{vt}^{p_3}$ and $X_{st}^{p_2+p_3} = X_{sv}^{p_2} + X_{vt}^{p_3}$. From Definition 1.1.4, we know that X_e, for all $e \in E$ are independent random variables and since all paths in the graph are acyclic, $X_{vt}^{p_3}$ is independent of $X_{sv}^{p_1}$ and $X_{sv}^{p_2}$. The

cumulative density function of the sum of two independent random variables can be computed by the following convolution integral

$$F_{X+Y}(t) = \int_{-\infty}^{\infty} F_X(x) f_Y(t-x) dx.$$

Therefore, for all $t \in \mathbb{R}$ holds

$$\begin{aligned} F_{X_{sv}^{p_2}+X_{vt}^{p_3}}(t) - F_{X_{sv}^{p_1}+X_{vt}^{p_3}}(t) &= \int_{-\infty}^{\infty} F_{X_{sv}^{p_2}}(x) f_{X_{vt}^{p_3}}(t-x) dx \\ &\quad - \int_{-\infty}^{\infty} F_{X_{sv}^{p_1}}(x) f_{X_{vt}^{p_3}}(t-x) dx = \\ &= \int_{-\infty}^{\infty} \left(F_{X_{sv}^{p_2}}(x) - F_{X_{sv}^{p_1}}(x) \right) f_{X_{vt}^{p_3}}(t-x) dx \geq 0. \end{aligned}$$

where the last inequality follows from non-negativity of $f_{X_{vt}^{p_3}}$ and dominance of p_1 by p_2. This concludes the first part of the proof.

Let us define function $u : \mathbb{R} \to \mathbb{R}$ such that

$$F_{X_{sv}^{p_1}}\left(u\left(x\right)\right) = F_{X_{sv}^{p_2}}\left(x\right).$$

Without loss of generality we assume that $F_{X_{sv}^{p_1}}$ and $F_{X_{sv}^{p_2}}$ are increasing and continuous functions. Together with the fact that p_2 dominates p_1 it is easy to see that u is non-decreasing function. Expectation of some random variable can be defined as

$$\mathbb{E}(w(X)) = \int_{-\infty}^{\infty} w(x) f(x) dx = \int_0^1 w(x) dF(x).$$

Hence,

$$\begin{aligned} \mathbb{E}\left(w\left(X_{sv}^{p_1}\right)\right) &= \int_0^1 w(x) dF_{X_{sv}^{p_1}}(x) = \int_0^1 w(u(x)) dF_{X_{sv}^{p_1}}(u(x)) \\ &= \int_0^1 w(u(x)) dF_{X_{sv}^{p_2}}(x) \geq \int_0^1 w(x) dF_{X_{sv}^{p_2}}(x) = \mathbb{E}\left(w\left(X_{sv}^{p_2}\right)\right). \end{aligned}$$

Notice that the inequality follows from the fact that functions u and w are both non-decreasing. □

Unfortunately, Algorithm 1, as stated, is only of theoretical relevance. In order to make it implementable in practise, we have to introduce

- compact representation of continuous random variables,
- computation of the distribution of the path length (Step 9),
- evaluation of dominance rule (Step 6),
- computation of expected cost of the path (Step 11).

In particular, computation of sum of the random variables, which leads to evaluation of convolution integral, is difficult and widely studied problem. To solve this problem, we propose a novel approach to computing convolution integral by discretising random variables in the next section.

1.3 Discretisation of Continuous Random Variables

The core of our approach lies in the approximation of probability density functions of continuous random variables by the step functions, which are piecewise constant functions, with finitely many pieces. This method is applicable to every distribution class of random variables with known cumulative density function.

The step functions can be represented by finite number of real values which simplifies representation of random variable in the computer memory. In Section 1.4, we show that such representation allows very efficient way to compute convolution integral as well. Moreover, the technique can be repeatedly used to arbitrarily many summands since result of the convolution can be easily approximated by the step function and thus used in next convolution. The method allows analysis of the approximation error, which is addressed in Section 1.5.

We approximate the probability density function f_X of continuous random variable X by step function f_{X^k} with constant step size k. We choose step values in a way that the cumulated probability of each step equals cumulated probability of f_X on this interval. This also assures that f_{X^k} integrates to one on the domain. Therefore function f_{X^k} can be seen as probability density function of a continuous random variable and we denote the random variable with probability density function f_{X^k} as X^k.

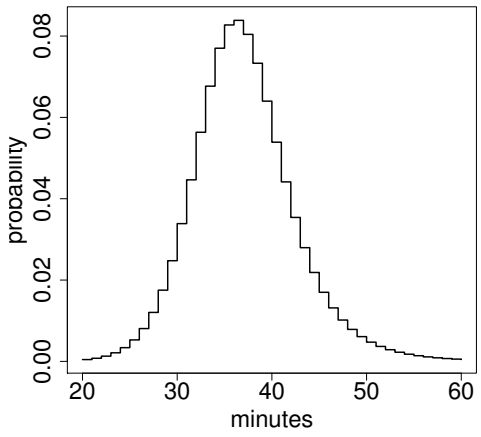


Figure 1.1: Discretised probability density function of a continuous random variable.

For every step end point $ik, i \in \mathbb{Z}$ also holds

$$F_X(ik) = F_{X^k}(ik)$$

where F_X and F_{X^k} are cumulative density functions. For illustrative example of a discretised probability density function see Figure 1.1.

Most of distributions have unbounded support
$mathbbR$ or $\mathbb{R}^+$. In order to be able to represent the probability density function by a finite number of steps, we restrict the support, on which the function gains positive values, to a bounded interval. Clearly, $f_X(x) > \varepsilon$ for each $\varepsilon > 0$ holds on some bounded support. Denote $x_l(\varepsilon)$ lower bound and $x_u(\varepsilon)$ upper bound of this support

$$\begin{aligned} x_l(\varepsilon) &= \max\{x \in \mathbb{R} \mid f_X(y) < \varepsilon, \forall y < x\}, \\ x_u(\varepsilon) &= \min\{x \in \mathbb{R} \mid f_X(y) < \varepsilon, \forall y > x\}. \end{aligned} \tag{1.1}$$

Note that we pick ε very small and fixed.

Such bounding of the support is not harming the performance of the method because of limited accuracy of the representation of real numbers in the computer where the non-zero values smaller than some very small threshold values are rounded to zero.

We assume that tails of the distribution, which are cut off after we round values smaller than ε to zero, have cumulated probability less than $\delta(\varepsilon) > 0$

where $\delta(\varepsilon)$ is still very small. Therefore, we assume that

$$\int_{x_l(\varepsilon)}^{x_u(\varepsilon)} f(x)dx = 1.$$

The following definition formalises described discretisation process.

Definition 1.3.1 (Discretised Random Variable)**.** *Let X be a continuous random variable with probability density function f_X and cumulative density function F_X. We denote by X_k the continuous random variable with probability density function f_{X^k} which is the step function with step size k and f_{X^k} is defined as*

$$f_{X^k}(x) = \begin{cases} \alpha_i^{X^k}, & \text{for } x \in (k\,(i-1)\,, ki]\,, \quad i = l_{X^k}, \ldots, u_{X^k}, \\ 0, & \text{else}, \end{cases} \tag{1.2}$$

where

$$\alpha_i^{X^k} = \frac{\int_{k(i-1)}^{ki} f_X(x)dx}{k} = \frac{F_X(ki) - F_X(k(i-1))}{k}, \quad i = l_{X^k}, \ldots, u_{X^k}. \tag{1.3}$$

Lower and upper bound on non-zero steps are defined as

$$l_{X^k} = \lfloor x_l(\varepsilon)/k \rfloor \text{ and } u_{X^k} = \lceil x_u(\varepsilon)/k \rceil.$$

We refer to X_k as discretised continuous random variable.

Notice that l_{X^k} and u_{X^k} also depend on ε thus, notion $l_{X^k}(\varepsilon)$ and $u_{X^k}(\varepsilon)$ would be more appropriate. Nevertheless, we prefer the simpler notion when no confusion can arise.

1.4 Discretised Labelling Algorithm

Now we show how to implement Algorithm 1 with employment of discretised random variables. We refer to such implementation as Discretised Labelling Algorithm.

We start with showing how to compute expected cost of the path for Step (11), and evaluation of dominance rule in Step (6). In the end of this section, we show the implementation of the convolution of discretised random variables which is necessary operation for implementation of Step (9).

1.4.1 Expectation

Expected cost of X is for some real valued integrable cost function w defined

$$\mathbb{E}(w(X)) = \int_{-\infty}^{\infty} w(x) f_X(x) dx.$$

When using X^k instead of X, computation of expectation is straightforward because it reduces to the sum

$$\mathbb{E}(w(X^k)) = \sum_{i=l_{X^k}}^{u_{X^k}} \alpha_i^{X^k} \int_{ki-k}^{ki} w(x) dx.$$

In case there is no analytical description of $\int w(x)dx$ given, one can approximate the value of the function by some quadrature rule for numerical integration as trapezoidal rule

$$\int_{ki-k}^{ki} w(x)dx \approx k\frac{w(ki-k) + w(ki)}{2}$$

or Simpson's rule

$$\int_{ki-k}^{ki} w(x)dx \approx k\frac{w(ki-k) + 4w(ki-\frac{k}{2}) + w(ki)}{6}.$$

For short survey on this topic see Smyth (1998) [102].

1.4.2 Dominance

The cumulative density function of the discretised continuous random variable is a piecewise linear function with equally spaced segment end-points. Hence, to implement Dominance Rule 1.2.1 for X_1^k and X_2^k it suffices to check dominance at segment end-points what is done by the evaluation of sums

$$\sum_{i=\min\{l_{X_1^k}, l_{X_2^k}\}}^{j} k f_{X_1^k}(ik) \leq \sum_{i=\min\{l_{X_1^k}, l_{X_2^k}\}}^{j} k f_{X_2^k}(ik),$$

for each $j = \min\{l_{X_1^k}, l_{X_2^k}\}, \min\{l_{X_1^k}, l_{X_2^k}\} + 1, \ldots, \max\{u_{X_1^k}, u_{X_2^k}\}$.

1.4.3 Convolution

Let X and Y be two independent random variables with integrable probability density functions f_X and f_Y, and let Z by their sum $Z = X + Y$. Probability density function f_Z of Z can be computed as the following convolution integral

$$f_Z(t) = \int_{-\infty}^{\infty} f_X(x) f_Y(t - x) dx. \tag{1.4}$$

Since we do not take any assumptions on distribution classes and parameters of random variables we can compute this integral numerically; analytical evaluation is unfortunately possible only in special cases. Various approaches to numerical computation of integrals can be found in the literature. See Krommer & Ueberhuber (1994) [73], Krommer & Ueberhuber (1987) [72], and Srivastava & Buschman (1992) [105] for an overview of various methods for numerical integration.

The primal focus of these methods is to compute a single integral. Our application, however, requires method for finding function f_Z rather than a value in single point $f_Z(t)$. Besides, effective repeated employment of the method is one of the key properties which we require from the integration method. Therefore, resulting function f_Z has to fulfil assumptions taken by the method on initially integrated functions f_X and f_Y so that f_Z can be used for further computation. Moreover, some methods are computationally intensive, which makes their application thousands of times during a single optimisation prohibitive.

Fan, Kalaba & Moore II (2005) [53] tackle the problem of evaluation of convolution along a path by application of Laplace transforms. Badinelli (1996) [9] provides computational study of various methods using approximation by orthogonal polynomials. He compares the accuracy of the methods with respect to the number of approximation points. Another approach presents Frank (1969) [56], where he proposes a Monte Carlo simulation method to estimate the probability density function of the path length.

Clearly, convolution of discrete random variables is much easier to compute since it reduces to evaluation of sums. This idea was used by Dodin (1985) [43] where he discretises probability density functions to finite number of values, which are computed by solving the system of non-linear equations. Obtained discrete distributions are then used for further computations. In

other words, he approximates continuous random variables by discrete random variables.

We also use the advantage of working rather with sums instead of integrals. Unlike Dodin (1985) [43], we still work with continuous random variables. Discretisation introduced in the previous section allows us to compute convolution as easily as for discrete random variables while still working with continuous random variables.

Let X and Y be continuous random variables and X^k and Y^k be their respective discretisations with identical step size k. Notice that the variables does not necessarily have identical upper and lower step bounds l_{X^k}, l_{Y^k}, and u_{X^k}, u_{Y^k} but the points separating the steps in each function are the same equal, namely, integers divisible by k.

In the first step, instead of computing probability density function f_Z of $Z = X + Y$, by evaluating Integral (1.4), we compute $\bar{Z}^k = X^k + Y^k$. As we show, probability density function of $\bar{Z}^k$ is not a step function. Therefore in the second step, we approximate $\bar{Z}^k$ by Z^k with f_{Z^k} as step function.

In order to compute $f_{\bar{Z}^k}$, we have to evaluate integral

$$f_{\bar{Z}^k}(t) = \int_{-\infty}^{\infty} f_{X^k}(x) f_{Y^k}(t-x)dx. \tag{1.5}$$

Observation 1.4.1. *Function $f_{\bar{Z}^k}$ is piecewise linear function with line segments end-points ik where*

$$i \in \{l_{\bar{Z}^k}, l_{\bar{Z}^k}+1, \ldots, u_{\bar{Z}^k}\} = \{l_{X^k} + l_{Y^k} - 2, l_{X^k} + l_{Y^k} - 1, \ldots, u_{X^k} + u_{Y^k}\}.$$

and $f_{\bar{Z}^k}(x) = 0$ on intervals $(-\infty, k\,(l_{X^k} + l_{Y^k} - 2))$ and $(k\,(u_{X^k} + u_{Y^k}), \infty)$. Moreover, function values in the line segment end-points can be computed as

$$f_{\bar{Z}^k}(ik) = \sum_{j=-\infty}^{\infty} k\alpha_j^{X^k}\alpha_{i-j+1}^{Y^k} = \sum_{j=\max\left\{l_{X^k},\, i+1-u_{Y^k}\right\}}^{\min\left\{u_{X^k},\, i+1-l_{Y^k}\right\}} k\alpha_j^{X^k}\alpha_{i-j+1}^{Y^k}. \tag{1.6}$$

Let us explain this fact. Since both density functions have equal step size, we can rewrite Integral (1.5) for any ik as the sum of integrals over individual

steps

$$\begin{aligned} f_{\bar{Z}^k}(ik) &= \int_{-\infty}^{\infty} f_{X^k}(x) f_{Y^k}(ik - x) dx \\ &= \sum_{j=-\infty}^{\infty} \int_0^k f_{X^k}(jk - y) f_{Y^k}(ik - jk + y) dy \\ &= \sum_{j=-\infty}^{\infty} \int_0^k \alpha_j^{X^k} \alpha_{i-j+1}^{Y^k} dy = \sum_{j=-\infty}^{\infty} k \alpha_j^{X^k} \alpha_{i-j+1}^{Y^k}. \end{aligned} \tag{1.7}$$

where the second equation is only reformulation following from the definition of discretised random variables, see Definition 1.3.1. Notice that, for simplification, we consider $\alpha_i^{X^k}$ to be defined also for $i < l_{X^k}$ and $i > u_{X^k}$ and clearly, $\alpha_i^{X^k} = 0$ in such situation.

Clearly, summand j is non-zero only if $\alpha_j^{X^k}$ is non-zero and $\alpha_{i-j+1}^{Y^k}$ is non-zero which holds only for j such that

$$\begin{aligned} l_{X^k} &\leq j \leq u_{X^k}, \\ l_{Y^k} &\leq i - j + 1 \leq u_{Y^k}. \end{aligned}$$

It is also easy to see that for $i \leq l_{X^k} + l_{Y^k} - 2$ and for $i \geq u_{X^k} + u_{Y^k}$ there is no such j and therefore all summands equal zero as well as $f_{\bar{Z}^k}(ik)$.

Now, we approximate resulting probability density function by the step function. We approximate $f_{\bar{Z}^k}$ by step function f_{Z^k} so that the mass of the function on each line segment is preserved. Therefore,

$$f_{Z^k}(x) = \begin{cases} \alpha_i^{Z^k}, & \text{for } x \in (k(i-1), ki], \quad i = l_{Z^k}, \ldots, u_{Z^k}, \\ 0, & \text{else,} \end{cases}$$

where

$$\alpha_i^{Z^k} = \frac{f_{\bar{Z}^k}(k(i-1)) + f_{\bar{Z}^k}(ki)}{2}, \quad i = l_{Z^k}, \ldots, u_{Z^k}, \tag{1.8}$$

and $l_{Z^k} = l_{\bar{Z}^k} + 1$ and $u_{Z^k} = u_{\bar{Z}^k}$. Notice that this approach preserves properties of probability density functions for f_{Z^k} because f_{Z^k} is non-negative function and

$$\int_{kl_{Z^k}}^{ku_{Z^k}} f_{Z^k}(x) dx = 1$$

under assumption that f_{X^k} and f_{Y^k} are also probability density functions.

1.5 Approximation Error

In the previous section we showed how to implement Algorithm 1 by using discretised random variables. Discretisation itself introduces an approximation error in the representation of probability density function of the random variables. By the approximation error, we understand absolute maximal discrepancy between true value of the function and the represented value.

Along the path search, as the sum of random variables is evaluated, the initial approximation error is amplified. It is, therefore, important to asses the growth of approximation error with increasing amount of summed random variables.

Now, let us prove that the approximation error of the implementation is very small. Actually, we show that the error is linear in terms of number of convoluted random variables and discretisation step size.

Theorem 1.5.1. *Let X_i for $i = 1, \ldots, n$ be continuous random variables and $Z_n = \sum_{i=1}^{n} X_i$. Discretised Labelling Algorithm with discretisation step size k approximates Z_n by $Z_n^k = \sum_{i=1}^{n} X_i^k$ with approximation error of probability density function $f_{Z_n}^k$*

$$e = O(k \sum_{i=1}^{n} f'_{X_i}(\xi_i))$$

where f'_{X_i} denotes first derivative of f_{X_i} and $\xi_i \in \mathbb{R}$.

Proof. We prove the theorem by induction in number of summed random variables.

First, let us start by looking at the approximation error of discretising single probability density function f_{X_i} of random variable X_i by step function $f_{X_i^k}$ with step size k. See Definition 1.3.1. We look on approximation error on interval $\left(kl_{X_i^k}, ku_{X_i^k}\right)$ where steps have non-zero values. We analyse the approximation error for every step separately. Clearly, the approximation error on $\left(kl_{X_i^k}, ku_{X_i^k}\right)$ is the maximum of approximation errors on individual

intervals.

$$\begin{aligned} e_{X_i^k} &= \max_{x\in(kl_{X_i^k},ku_{X_i^k})} \left|f_{X_i}(x) - f_{X_i^k}(x)\right| \\ &= \max_{m\in l_{X_i^k},\ldots,u_{X_i^k}} \max_{x\in(k(m-1),km)} \left|f_{X_i}(x) - f_{X_i^k}(x)\right| \qquad (1.9) \\ &\leq \max_{m\in l_{X_i^k},\ldots,u_{X_i^k}} kf'_{X_i}(\xi_i^m) = kf'_{X_i}(\xi_i) \end{aligned}$$

where $\xi_i^m \in (k(m-1), km]$ and $\xi_i \in \{\xi_i^{l_{X_i^k}}, \ldots, \xi_i^{u_{X_i^k}}\} \in \left(kl_{X_i^k}, ku_{X_i^k}\right]$.

Now, we look on approximation error on intervals where f_{X_i} is approximated by zero, namely on intervals $\left(-\infty, kl_{X_i^k}\right]$ and $\left(ku_{X_i^k}, \infty\right)$. Notice that $l_{X_i^k}$ and $u_{X_i^k}$ depend on the choice of ε and we know that $f_{X_i}(x) \leq \varepsilon$, see (1.1). So the approximation error on these intervals is at most ε. Random variables X_i are given in advance so the discretisation step size k; thus, we may assume that ε has been chosen in a way that

$$\varepsilon \leq e_{X_i^k}, \quad \forall i \in 1, \ldots, n.$$

We conclude that $f_{X_i^k}$ approximates f_{X_i} with error at most $e_{X_i^k}$ on $\mathbb{R}$ what proves the first step of the induction for X_1.

Let $Z_s = \sum_{i=1}^{s} X_i$ for some $s < n$, and let Z_s^k be random variable computed as the sum of $X_1^k, \ldots, X_s^k$ by repeated application of Formulae (1.6) and (1.8). From the induction step, we assume that probability density function $f_{Z_s^k}$ approximates f_{Z_s} with error

$$e_{Z_s^k} = O(k \sum_{i=1}^{s} f'_{X_i}(\xi_i))$$

for $\xi_i \in \mathbb{R}$.

Now, let us analyse computation of $Z_{s+1}^k = Z_s^k + X_{s+1}^k$ in two steps. First we compute $\bar{Z}_{s+1}^k = Z_s^k + X_{s+1}^k$ by Formula (1.6) and then we compute Z_{s+1}^k from $\bar{Z}_{s+1}^k$ by Formula (1.8).

Recall that resulting probability density function $f_{\bar{Z}_{s+1}^k}$ of $\bar{Z}_{s+1}^k$ is a piecewise linear function. We bound the approximation error of the probability density

function $f_{\bar{Z}^k_{s+1}}$ with respect to $f_{Z_{s+1}}$ as

$$\begin{aligned}
\bar{e}_{Z^k_s + X^k_{s+1}} &= \max_{t\in\mathbb{R}} \left| f_{\bar{Z}^k_{s+1}}(t) - f_{Z^k_{s+1}}(t) \right| \\
&= \max_{t\in\mathbb{R}} \left| \int_{-\infty}^{\infty} f_{Z^k_s}(t-x) f_{X^k_{s+1}}(x)dx - \int_{-\infty}^{\infty} f_{Z_s}(t-x) f_{X_{s+1}}(x)dx \right| \\
&\leq \max_{t\in\mathbb{R}} \left| \int_{-\infty}^{\infty} f_{Z^k_s}(t-x) \left(f_{X^k_{s+1}}(x) - f_{X_{s+1}}(x) \right) dx \right| + \\
&+ \left| \int_{-\infty}^{\infty} f_{X_{s+1}}(x) \left(f_{Z^k_s}(t-x) - f_{Z_s}(t-x) \right) dx \right| \\
&= e_{X^k_{s+1}} + e_{Z^k_s}.
\end{aligned}$$

The last equality follows from the fact that $|f_{X^k_{s+1}}(x) - f_{X_{s+1}}(x)| \leq e_{X^k_{s+1}}$ and from the induction step $|f_{Z^k_s}(t-x) - f_{Z_s}(t-x)| \leq e_{Z^k_s}$, as well as from the fact that $f_{X_{s+1}}$ and $f_{Z^k_s}$ integrate to one on $\mathbb{R}$.

Now, we transform $f^k_{\bar{Z}_{s+1}}$ to step function by (1.8). This operation introduces the additional approximation error on each segment i by at most

$$e^i_{\bar{Z}^k_{s+1}} = \frac{|f_{\bar{Z}^k_{s+1}}(ik) - f_{\bar{Z}^k_{s+1}}(ik-k)|}{2}.$$

This follows from the fact that we approximate linear function by the constant function thus, the biggest approximation error occurs in extreme points of the approximation interval. From Formula (1.6) we equivalently get

$$\begin{aligned}
e^i_{\bar{Z}^k_{s+1}} &= \frac{\sum_{j=-\infty}^{\infty} k f_{Z^k_s}(jk) \left(f_{X^k_{s+1}}((i-j+1)k) - f_{X^k_{s+1}}((i-j)k) \right)}{2} \leq \\
&\leq \frac{\sum_{j=-\infty}^{\infty} k f_{Z^k_s}(jk) \left(k f'_{X_{s+1}}(\xi^i_{X_{s+1}}(i-j+1)) \right)}{2} \leq k f'_{X_{s+1}}(\xi^i_{X_{s+1}})
\end{aligned}$$

where $\xi^i_{X_{s+1}}(j) \in (k(j-1), kj]$ and $\xi^i_{X_{s+1}} \in \left(k l_{X^k_{s+1}}, k u_{X^k_{s+1}} \right]$. Last inequality follows from the fact that $\sum_{j=-\infty}^{\infty} k f_{Z^k_s}(jk) = 1$.

Therefore, the transformation of $f_{\bar{Z}^k_{s+1}}$ to step function $f_{Z^k_{s+1}}$ introduces additional approximation error

$$e_{\bar{Z}^k_{s+1}} \leq \max_{i \in l_{X^k_{s+1}}, \ldots, u_{X^k_{s+1}}} k f'_{X_{s+1}}(\xi^i_{X_{s+1}}) = k f'_{X_{s+1}}(\xi_{X_{s+1}})$$

where $\xi_{X_{s+1}} \in \left(l_{X^k_{s+1}}k, u_{X^k_{s+1}}k\right]$.

Finally, we proved that the discretisation method approximates Z_{s+1} with approximation error

$$e_{Z^k_{s+1}} \leq e_{Z^k_s} + e_{X^k_{s+1}} + e_{\bar{Z}^k_{s+1}} \leq e_{Z^k_s} + 2e_{X^k_{s+1}}$$

what concludes the proof. □

Besides theoretical results on the accuracy of the approximation, we study performance of the discretisation method in practical application. We apply the discretisation method for computation of the distribution of propagated delay along aircraft rotations. This problem is alike to the problem of finding the stochastic shortest path. We refer reader to Chapter 3 for more details about aircraft delay propagation. The computational results on accuracy of the discretisation method are given in Section 3.6.

Chapter 2

Introduction To Airline Planning

Mathematical optimisation has wide range of applications in the air transportation industry. In this chapter, we provide a brief overview on some of these applications connected to aircraft and crew planning.

Ideally, one would formulate and solve a single mathematical model coping with all aspects of the planning at once while maximising airlines profit. This problem would be, unfortunately, too complex to be solved. Therefore, planning is decomposed into several problems which are solved sequentially. In Section 2.1-2.6, we go through these problems. We start with schedule generation which is solved few months before the day of operations and we end with schedule recovery problem which is solved on the day of operations.

However, recent development shows that integration of some of aforementioned problems into one problem became tractable. Topic of integration of individual problems is reviewed in Section 2.7. Robustness in the planning is a separate topic of Section 2.8.

Detailed overview on the airline planning problems and models can be found in Yu (1997) [110], Barnhart, Belobaba & Odoni (2003) [15], Klabjan (2005) [65], Belobaba, Odoni & Barnhart (2009) [17], and Weide (2009) [108].

2.1 Schedule Generation

Prior to aircraft and crew planning, a flight schedule has to be created. Construction of the schedule depends on many strategic decisions including choice of markets and airports to serve, time and frequency of the connections. All these decisions highly influence profitability of the airline and depend on various factors as airlines customer base, expected future development of the market, the airlines network type, and overall strategy of the airline. These aspects are difficult to quantify and even harder to mathematically model. Due to all aforementioned reasons, the schedule generation is mostly done manually with less mathematical optimisation involved.

The set of legs, that could be part of the schedule, is enormous. However, airlines maintain their schedules as stable as possible between the seasons. They do not want to construct the entire schedule from the scratch every time but they apply only incremental changes to the current schedule. Lohatepanont & Barnhart (2004) [79] therefore integrate the schedule generation problem with fleet assignment where the goal is to choose legs to be included to the schedule from given list of candidate legs.

2.2 Fleet Assignment

After the schedule is generated, each leg has to be assigned to specific aircraft type. This is achieved by solving so called fleet assignment problem. At this stage of the planning, the goal is to maximise profit by minimising aircraft operational costs and maximising passenger revenue.

Feasible solution of the fleet assignment has to fulfil aircraft balance and count constraints. The balance constraint ensures that from every airport in any moment departs not more aircraft of each type than it has previously landed at this airport. The count constraint ensures that the solution does not use more aircraft, for each fleet, than it is available. We refer to Abara (1989) [2], Hane et al. (1995) [62], and Barnhart et al. (1998) [12] for more information about the fleet assignment problem, models, and algorithms.

Literature also proposes extensions of the fleet assignment model towards more complex formulations. Rexing et al. (2000) [90] propose a model which allows to shift leg departures in a given time window which may lead to decrease operational costs. Barnhart, Kniker & Lohatepanont (2002) [14]

propose itinerary-based model which improves accuracy of the forecasting of passenger demand; Barnhart, Farahat & Lohatepanont (2009) [16] improve tractability of this model.

2.3 Maintenance Routing

Once the legs are split into various fleets, feasibility of the schedule within each fleet has to be ensured. Every aircraft has to undergo regularly maintenance checks which vary in frequency and duration. The most regular maintenance check is done every few days and is carried out during the night; the most complex check is done every three to five years and takes few weeks; see Lan (2003) [75] for more details about aircraft checks.

The goal of the maintenance routing problem is to construct a rotation for every aircraft in order to fulfil aircraft maintenance requirements and other type specific constraints. The rotations are constructed few months before the day of operation hence, actual maintenance requirements of the aircraft at the day of operation may differ and construction of different routings may be necessary. Nevertheless, these rotations are important for construction of crew pairings and for proving the maintenance feasibility of fleet assignment.

Besides construction of maintenance feasible rotations there is no clear objective to be minimised. Clarke et al. (1997) [33] solve aircraft maintenance routing where through value is maximised. Through value is revenue gained by attracting passengers by providing them a multi-leg itinerary on the same aircraft. Furthermore, they require the same route for all aircraft what ensures equal utilisation of all aircraft. Another objective propose Sarac, Batta & Rump (2006) [93] where they minimise too high utilisation of the aircraft between maintenance checks.

2.4 Crew Planning

After the construction of aircraft routings, the planning process continues with crew members. Due to tractability reasons, the crew planning is divided into two steps. In the first step, crew pairings are constructed by solving so called crew pairing problem. The pairing is a sequence of flight legs starting and ending at the same crew base. In the second step, called the crew

rostering, crew pairings are combined to monthly work schedules which are assigned to individual crew members. The crew pairing problem is of a high significance because the crew costs are the second highest operational costs of the airlines.

Typically, one pairing consists of few duty periods separated by the rest time. The cost function of the crew pairings is complex and non-linear depending on duration of the pairing, total flight time, and total work time of the crew within the pairing. Furthermore, the crew pairings have to satisfy various work rules given by national law and labour unions like: restriction of maximum duty period, maximum flying time during a duty, maximum number of duties within a pairing, and requirement of minimum rest time between duties. We refer to Hoffman & Padberg (1993) [63], Borndörfer et al. (2006) [25], AhmadBeygi, Cohn & Weir (2009) [5], and Klabjan et al. (2001) [67] for an overview on models and solution methods for solving the crew pairing problem.

In the crew rostering problem, also called the crew assignment problem, pairing are assigned to individual crew members. Here, besides minimising costs, quality of life ,or crew satisfaction, is an optimisation criterion.

European and North American airlines approach this problem differently. North American airlines mostly use bidline approach where first, anonymous rosters are constructed and then, the rosters are assigned to individual crew members in bidding process in seniority order.

Most European airlines prefer personalised rostering. The crew members submit their preferences beforehand and the roster is constructed for each crew member according to these preferences. For more details about this topic we refer reader to Gamache et al. (1999) [59], Day & Ryan (1997) [40], and Kohl & Karisch (2004) [70].

2.5 Tail Assignment

The tail assignment problem is alike to the maintenance routing problem with the difference that this problem is solved shortly before the day of operation. The goal is to construct final rotations for a set of individual aircraft in order to cover a set of legs. One can take into consideration individual operational constraints of each aircraft as well as short- to long-term individual maintenance requirements. The problem is repeatedly solved on a daily basis

in order to adjust previously constructed routings to current situation which is influenced by recent disruptions and changes in the schedule. More details can be found in the thesis of Grönkvist (2005) [61] as well as in Chapter 4 which is dedicated to this problem.

2.6 Recovery

Delays, crew sickness, malfunction of technical equipment or bad weather conditions on the day of operation may require changes in the schedule. Traditional models used in the planning are not suitable for repairing the schedules because of high running time requirements. Due to time pressure, a new solution has to be available within few minutes.

Every change in the planning is time consuming and risky thus, deviations from the current schedule need to be minimal. what restricts the scope of the optimisation in comparison to standard settings. On top of that, aircraft, crew, and passengers have to be recovered at the same time. Often, good solution for one of these resources leads to very bad for another resource. Recovery of disrupted schedule is, therefore, very complex task and requires to find some compromise solution for all three resources within short time.

As first, recovery models for individual resources were formulated. Argüello, Bard & Yu (1997) [7] solve the aircraft recovery problem where aircraft are rerouted or legs are cancelled in order to make the schedule again operationally feasible and to minimise cost of leg delays. Lettovsky, Johnson & Nemhauser (2000) [77], Yu et al. (2003) [111] study the crew recovery problem where disrupted crew pairings are repaired in order to fulfil all work rules and to minimise impact on the schedule.

As we mentioned, recovery algorithm should consider all resources, namely, aircraft, crew, and passengers, at once and to find the optimal solution for such joint formulation. This motivation is pursued in some recent works. Bratu & Barnhart (2006) [28] propose model for the aircraft recovery problem where they consider disrupted passengers and crews as well. Bisaillon et al. (2010) [24] propose a large neighbourhood search heuristic for integrated aircraft and passenger recovery problem.

Kohl et al. (2007) [71] outline current practise of airlines in dealing with disruptions and describe disruption management system for solving recovery problem for aircraft, cockpit and cabin crew, and passengers. The system

is based on integration of recovery solutions for each independent resource. Petersen et al. (2010) [88] present integrated recovery model comprised of the aircraft recovery sub-problem for ensuring aircraft maintenance requirements; the crew recovery problem for recovering cockpit crew; and the passenger recovery problem for re-assigning disrupted passengers to a new itinerary. These sub-problems are linked together by the schedule recovery problem deciding which flight legs to cancel, delay or divert. We suggest Ball et al. (2006) [10] as a reference for detailed literature review on recovery and on description of recovery models.

2.7 Integration

Effective algorithms and growing computational power of computers allow us to solve larger optimisation problems than ever before. Hence, more complex models are being developed. Primal interest is the integration of two or more previously introduced individual problems into one integrated mathematical model. Integration increases total cost efficiency of the solutions. Moreover, the individual problems are not totally independent and their sequential solving may lead to infeasibility which can be overcome by integrated approach.

In the fleet assignment problem, legs are assigned to fleets however, maintenance feasibility of so created fleet schedules is not examined. Thus the maintenance routing problem may be infeasible for some fleet. If this is the case, the fleeting has to be updated. Barnhart et al. (1998) [12] overcome this complication by introducing the model integrating the fleet assignment with the maintenance routing. A model integrating fleet assignment with crew pairing introduce Gao, Johnson & Smith (2009) [60]. Such model offers more degrees of freedom to crew pairing construction and thus, allows lower costs of the pairings.

The aircraft routings and the crew pairings have even stronger dependency in between. If the crew continues the duty on the different aircraft, it needs typically more time to prepare for the next flight as if the crew stays on the same aircraft. Hence, initial aircraft rotations, produced by the maintenance routing problem, strongly predefine solution space for the crew pairing problem. Furthermore, frequent changes between aircraft are very uncomfortable and even restricted by labour unions. Every change of the aircraft is non-robust since it causes spreading of a single delay to more than one

consecutive leg. For all these reasons, crew pairings typically follow aircraft rotations. Weide (2009) [108], Dück (2010) [45], and Dunbar, Froyland & Wu (2010) [46] propose to solve maintenance routing and crew pairing iteratively in order to increase robustness of the planning.

Integrated model for crew pairing and aircraft routing is huge and difficult to solve hence, Cohn & Barnhart (2003) [37] propose smaller model consisting of a crew pairing model extended with maintenance routing decisions about short connections. Cordeau et al. (2001) [39]; Mercier, Cordeau & Soumis (2004) [83] proposes to solve an integrated aircraft routing and crew pairing model through Benders decomposition. Papadakos (2009) [87] goes even further and proposes the integrated model for fleet assignment, aircraft routing and crew pairing.

2.8 Robustness in Airline Planning

Due to growth of disruptions in the last years, robustness became an important topic in the planning. In context of airline planning, we see robustness as stability of the schedules under disruptions. This property increases chances of realisation of cost efficient schedules. There are two mathematical approaches tackling this problem: robust optimisation and stochastic programming. The approaches consider input data with some uncertainty, which represent delays, in a different way.

Robust Optimization

Soyster (1973) [104], as first, introduced robustness into linear programming. In his approach, some input parameters are not fixed but they belong to some uncertainty set. Any feasible solution of the problem then has to be feasible for all combinations of input parameters from this set. This worst-case approach produces solutions that are too conservative to be applicable for most practical applications.

Ben-Tal & Nemirovski (1998) [18], Ben-Tal & Nemirovski (1999) [19], and El Ghaoui & Lebret (1997) [50] propose less conservative approaches where the most extreme value combinations of input parameters do not have to be covered by feasible solutions.

Another approach to bound the conservatism of the solutions propose Bertsimas & Sim (2004) [21]; Sim (2004) [99]. They introduce a parameter which specifies maximum number of uncertain coefficients that may be changed while the solution stays feasible. For an application of this method on discrete optimisation problems see Bertsimas & Sim (2003) [20].

Sometimes, robustness of the solution is rather "nice to have" property than "a must". Then one wants to improve robustness as much as possible while keeping the cost of the solution close to optimal. Fischetti & Monaci (2008) [54] propose such approach where non-robustness is minimised while increase of the objective value is limited by some parameter.

In context of airline planning, uncertainty lies in delays occurring during the day of operations. For example, hedging against a single delay of thirty minutes is already too expensive or even impossible to implement. Moreover, there is no reason to require so strong protection since a recovery action can be taken to resolve the problem later during the operations. For these reasons, as far as we know, such worst case concepts have not been applied to actual airline scheduling problems.

A concept more matching to our requirements provides, so-called recoverable robustness, introduced by Stiller (2008) [106]. Here, some recovery algorithm to fix infeasibility is the part of the input. The solution of the problem may be itself infeasible for some scenarios, but it has to be possible to recover the solution by predefined recovery algorithm. Caprara et al. (2008) [31] apply this approach on train platforming.

Stochastic Programming

In stochastic programming, uncertainty is approached in a different way. While robust optimisation requires solution resistant to all considered realisation of the input parameters, stochastic programming considers uncertainty in a "soft" way.

Realisations of input data are considered with some probabilities and expected value of an objective function is minimised. Typically, violation of some constraints is penalised in the objective function. Another option to enforce fulfilment of the constraint is by chance constraints that ensure the given constraint is satisfied with at least some given probability.

Multi-stage stochastic programming, furthermore, gives an opportunity to model response to realisation of uncertain parameters from the previous stage. In context of airline planning problems, this is an important possibility how to asses propagated delay and recovery actions performed during the day of operation.

Stochastic programming approach is, from a mathematical point of view, the most satisfying solution approach for modelling robust airline planning problems; unfortunately, stochastic programming formulations are very often intractable. See Birge & Louveaux (2000) [23] as a general reference to stochastic programming.

Applications

Currently, it is intractable to implement complex models with incorporation of recovery operations that minimise overall delays and additional costs. So far, simplified approaches has been developed with focus on partial goals contributing to overall robustness. We recognise two basic directions pursued by robustness objective functions.

First, **stability**, or sometimes called reliability, reflects immunity of the schedule to disruptions. Aim is to construct schedule which is more likely to be operated without any ad-hoc changes and thus probability of realisation of planned costs of the schedules is higher. However, changes are sometimes inevitable. In such case, **flexibility** of the schedule, also called recoverability, is crucial. It ensures that the schedule can be recovered from disruption faster and for lower costs and effort.

Burke et al. (2010) [29] perform a simulation study in order to find a trade-off between stability and flexibility of aircraft schedules leading to highest increases of overall robustness. They show that stability has higher impact than flexibility.

Key performance indicators (KPIs) provide technically easiest way to improve the robustness. KPIs are usually simple to model and are based on experience. Clearly, there are many various KPIs with varying effectiveness AhmadBeygi et al. (2007) [3] and Bian et al. (2003) [22] analyse historical aircraft schedules in order to find features of schedules which enhance robustness; based on that, they propose KPIs for improvement of robustness. The biggest weakness of KPI approach is impossibility of assessing their real impact before putting into operations. This can be overcome by simulation of

schedules. Eggenberg & Salani (2009) [47], Shebalov & Klabjan (2006) [96], Dück (2010) [45], and others use simulation to prove effectiveness of their methods. See Section 6.1 for overview of complex simulation frameworks.

Let us recall some recent KPIs proposed in the literature. Grönkvist (2005) [61] proposes to minimise medium length ground times, that is, ground times which are too long to effectively absorb delays while at the same time too short to make the aircraft available as a back up for another delayed aircraft. In addition, he proposes to avoid multiple visits of critical airports or, similarly as Weide (2009) [108], to avoid tight crew connections with short buffer time. Eggenberg & Salani (2009) [47] suggest to improve robustness in the maintenance routing problem, with allowed retiming of legs, by maximisation of minimal idle time in every rotation, the total idle time of all aircraft, or by maximisation of number of plane crossings.

Stability of schedules is also topic of Sohoni, Lee & Klabjan (2008) [103] where they propose to adapt flight block times in order to increase punctuality of the airline or to decrease probability of disrupting passenger itineraries. In general, many approaches targeting to increase stability of the schedule are focusing on minimisation of delay propagation or some alike objective.

AhmadBeygi, Cohn & Lapp (2008) [4] minimise estimate of propagated delay which is computed as the sum of delay propagated to all consecutive legs. In aircraft maintenance routing, Lan (2003) [75]; Lan, Clarke & Barnhart (2006) [76] minimise total expected propagated delay; furthermore they allow retiming of flight legs for decreasing average number of passengers missing connection. In work of Dück (2010) [45] and Dunbar, Froyland & Wu (2010) [46] propagated delay is also chosen as a robustness measure. Tam, Ehrgott & Zakeri (2009) [107] compares bi-criteria KPI approach to crew pairing problem of Ehrgott & Ryan (2002) [48] with stochastic programming approach of Yen & Birge (2006) [109]. In both, the goal is to minimise delay propagation due to crew changing aircraft. In the crew pairing problem, Schaefer et al. (2005) [94] improves robustness by minimisation of expected crew pairing costs. Aforementioned works use different techniques to estimate or compute propagated delay. Overview of used techniques is given in Section 3.2.

Improvement of flexibility is the main focus of work of Smith & Johnson (2006) [101], Gao, Johnson & Smith (2009) [60] and Rosenberger, Johnson & Nemhauser (2004) [92]. They investigate fleet assignment where Smith & Johnson (2006) [101] increase robustness of airline schedules by restricting

number of different aircraft types visiting airports. Gao, Johnson & Smith (2009) [60] extend this approach by restrictions on crew members. Rosenberger, Johnson & Nemhauser (2004) [92] propose to increase robustness by minimising number of aircraft connecting different hubs and by shortening sequences of the legs without visiting the hub airport. Shebalov & Klabjan (2006) [96] maximise opportunities to swap crew members by increasing frequency of repeated crossings of crews.

Topic of robust scheduling is studied also in context of other problems transportation problems. For example, we refer to application in railway timetabling see Liebchen et al. (2007) [78], Fischetti, Salvagnin & Zanette (2009) [55] and Kroon et al. (2007) [74].

Chapter 3

Delay Propagation

This chapter is devoted to study of delay propagation in airline networks. In Section 3.1, we point out the importance of reduction of delay propagation in aviation. We show a motivational example demonstrating incapability of minimising delay propagation by ad hoc decisions and key performance indicators (KPIs). In Section 3.2, we give an overview on approaches for computation of delay propagation in airline networks known from the literature.

In this thesis, we propose a stochastic approach that relies on knowledge of distributions of primary delays. We model delay propagation directly. This allows an accurate minimisation of robustness objectives based on propagated delay.

In Section 3.3, we state the model of airline operations under which we study delays and delay propagation. Then, in section 3.4, we provide a formal description of computation of propagated delay along aircraft rotations.

We propose a practical implementation of all required operations in Section 3.4.2. This implementation is an extension of the discretisation method presented in Chapter 1, which approximates continuous probability density functions by step functions.

In Section 3.5, we enhance the method by including effects of non-rotational delay propagation. Such delay propagation is, in reality, caused by changing crew members, passengers, or awaiting cargo from another flight leg.

Implementation of additional operations is given in Section 3.5.2. We further discuss algorithmic difficulties of non-rotational delay propagation in Section 3.5.3 and Section 3.5.4.

In Section 3.6, we demonstrate that both methods introduced in this chapter are suitable for application in column generation framework in order to solve large-scale real world problems. We study accuracy and speed of the method on instances of the tail assignment problem for various parameters of discretisation method.

3.1 Motivation

On the day of operation, aircraft get disturbed by random events like bad weather, equipment malfunction, late passengers, baggage handling problems, etc. Most of these delays are inevitable and planners have to cope with them. Such delays are called primary delays. Deeper discussion on delay classification in given in Section 5.2.

Once a longer delay occurs it is difficult to eliminate it. After the landing, some of the delay can be absorbed by the ground buffer, if there is any, but the remaining delay propagates to the succeeding leg. Such a delay is called propagated delay (in the literature also called knock-on or reactionary delay).

According to Central Office for Delay Analysis CODA (2009) [36], 44% of all delay minutes in Europe in 2009 were due to propagated delays. The statistics show continuous growth of delays together with growth of traffic volume, see Figure 1. Statistics on various years and up to date statistics can be found on homepage of CODA [35].

There is nothing we can do about primary delays. However, propagated delays can be reduced by constructing aircraft rotations in a way that buffer times are allocated at the "right" connections. Figure 3.1 shows an example of effect of the aircraft rotations on delay propagation. We consider an airport with two incoming legs A and B and two outgoing legs C and D. The left side of the picture shows two rotations that connect legs AC and BD in a first-in, first-out manner, resulting in a nearly even distribution of the buffer times. However, suppose that leg A tends to be delayed more often than leg B and typically suffers from longer delays. One such situation is visualised on the figure by the thin lines while thick lines express planned operation of the legs. With this information, it makes sense to allocate more buffer

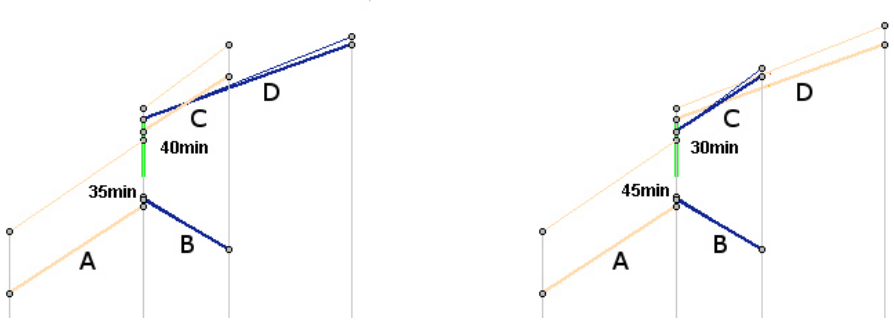


Figure 3.1: Effects of an aircraft routing on delay propagation (thick: plan, thin: day of operation).

after leg A by connecting legs AD and BC. Such a routing is shown on the right of Figure 3.1. As you can see, the solution on the right suffers less from propagated delays as the solution on the left, which is counter-intuitive.

Thinking this example through, it becomes apparent that the delay propagation from one leg to another does not only depend on the two legs involved, but on the entire rotation as well. The arrival delay of a leg depends on preceding legs, and the effect of a delay depends on the succeeding legs.

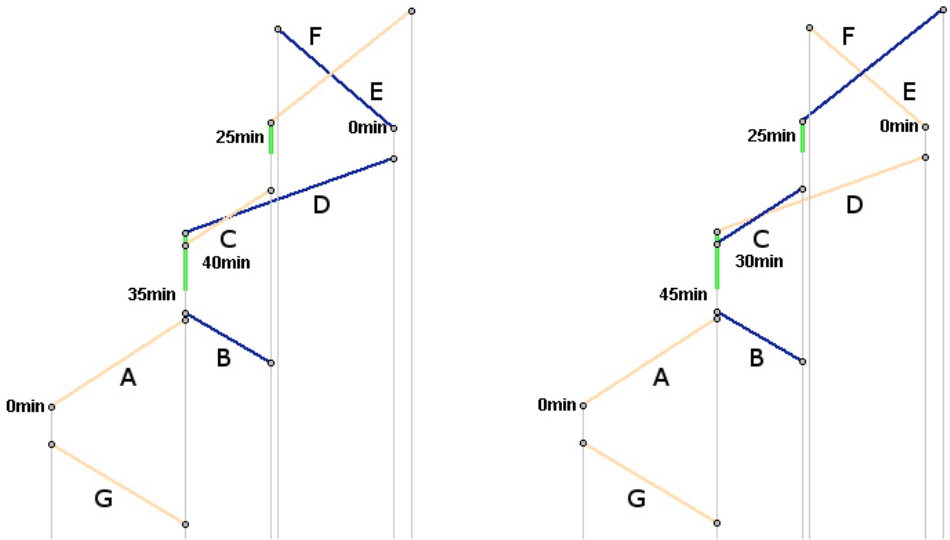


Figure 3.2: The effect of an aircraft routing on delay propagation along an entire rotation.

Figure 3.2 illustrates these facts by showing the rotations of Figure 3.1 in

whole. We see that leg A is preceded by leg G, and there is no ground buffer between leg G and A. Arrival delays of leg G are therefore fully propagated to leg A and thus the arrival delay of leg A in the rotation may well be much higher than would be assumed by considering leg A alone. It may be also higher than the arrival delay of leg B although leg A itself may suffer from delays less than leg B. Now, the solution on the right side of Figure 3.2 seems to be more robust than the solution on the left. Similarly, leg C is followed by leg F with 25 minutes of ground buffer and leg D is followed by leg E with no buffer. It may, therefore, be appropriate to cover legs D and E in a rotation that is unlikely to be delayed, because there is no further possibility to reduce propagated or additional primary delay. Aware of all these information, it becomes unclear which of the two studied routings is more robust.

The example shows that delay propagation has an impact on the whole rotation and therefore cannot be decided locally. On the other hand, minimisation of propagated delay in a network requires some quantitative measure of likelihood and magnitude of primary delays on the legs.

This motivation example explains importance of stochastic formulation and representation of delays and propagated delay. While the stochastic model is introduced in Chapter 5, computation of propagated delay is examined in the rest of this chapter.

3.2 Delay Propagation in the Literature

Topic of delay propagation is extensively studied in the literature. In general, it is a difficult problem to solve. The basic complication originates from computation of sum of two random variables which corresponds to convolution of their probability density functions. This problematic is already studied in Chapter 1; especially Section 1.4.3 is dedicated to overview of common approaches to computation of convolution.

On top of that, an application in context of airline planning requires some other transformations of the probability density functions, as we address further in this section. These transformations make the computation of the propagated delay even more complicated. Therefore, various ways how to compute distribution of propagated delay approximately, or under some special assumptions, were proposed.

An analytical approach to computation of convolution chooses Fuhr (2007) [57]. She uses a special class of Erlang-Exp/Exp-Erlang distributions to model delays. This class is, up to an approximation error, closed under convolution. The first three moments of the resulting distribution can be analytically computed, which allows fast computation of delay propagation along rotations. Similar approach developed Arikan et al. (2010) [8]. They use log-Laplace distributions which allow analytical approximated estimation of the first two moments of the distribution of propagated delay.

Many other methods avoid direct computation of convolution. An approximation method proposed by Dunbar, Froyland & Wu (2010) [46] approximates distributions of the primary delay by their expected values. The expected value of the propagated delay is then computed as a sum of the expected values of primary delays. It is easy to see that this method underestimates true value of the expected propagated delay.

Kleywegt, Shapiro & Homem-de Mello (2001) [69] propose a general Monte Carlo approach, which approximates desired expected values by the average of the samples. They propagate many individual delay samples whose average approximates the expected propagated delay. Similar approach is used by Schaefer et al. (2005) [94] to estimate expected costs of the crew pairings.

The idea of propagation of individual samples is also used by Lan, Clarke & Barnhart (2006) [76]. They propagate historical primary delays. Resulting histogram of the propagated delay is then fitted with a continuous distribution in order to obtain the distribution of the arrival delay.

3.3 The Model of Airline Operations

The structure of the model of airline operations has important consequences in algorithmic computation of the propagated delay. Therefore, before the actual computation of the propagated delay, we clarify the structure of the model which we further assume throughout whole thesis.

We separate aircraft operations into two phases called a gate phase and a block phases. The gate phase represents the time period an aircraft spends at a gate, that is, the time between arrival of an incoming leg at a gate and departure of the succeeding outgoing leg from this gate. The block phase represents the time from departure of the aircraft from the gate at departure airport until arrival to the gate at the arrival airport. This can be also

seen as a simplification of the models described and used in the simulation frameworks, which we discuss in Chapter 6. Clearly, duration of each such phase can deviate from the scheduled duration. We express these deviations by random variables.

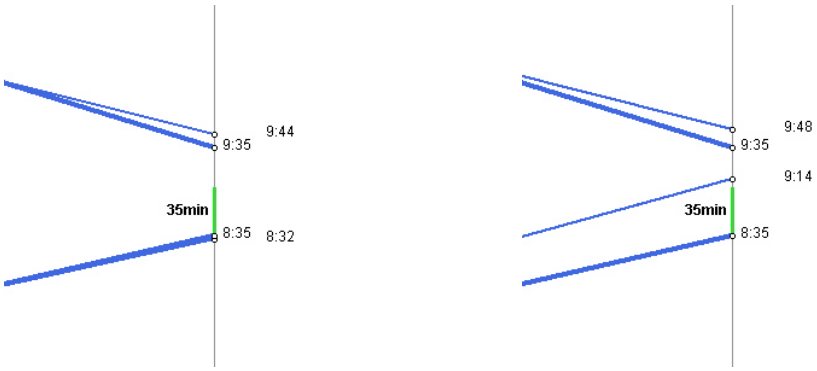


Figure 3.3: Absorption of the arrival delay at the airport by the ground buffer.

After landing, passengers have to leave the aircraft, the aircraft has to be prepared for the next flight and new passengers have to board the aircraft. Time necessary for all aforementioned operations is accounted in so called minimum ground time, which we consider to be a constant time period dependent on the aircraft type and airport. It means, an aircraft is not able to depart earlier than minimum ground time after the arrival to the gate.

Time between the scheduled arrival time and the scheduled departure time of the next leg in the rotation is called scheduled ground time. The time difference between the scheduled ground time and the minimum ground time is called scheduled buffer time, or just buffer. The buffer is an idle time period of the aircraft. Clearly, the buffer may absorb eventual arrival delays and avoid their propagation to the next leg.

Preparations for the next leg are mostly started some time before the departure, therefore buffers may not influence their completion time and possible delays. The same holds also for the congestion of the airport and weather conditions which may cause some delay too. It means that buffers are unable to avoid new primary delays at the gate.

Simple example demonstrating absorption of the delay shows Figure 3.3. There are two legs with scheduled ground time 60 minutes from which 25 minutes form minimum ground time and 35 minutes are buffer. On the left,

the aircraft has the arrival delay of 39 minutes; on the right, the aircraft arrives 3 minutes ahead of scheduled arrival time. Additionally, in both cases aircraft suffers from 9 minutes of primary gate delay. This results in delayed departure by 9 minutes and 13 minutes on the left and right side of the figure, respectively.

Translating described behaviour to stochastic model, gate phase can be split to ground time and primary gate delay generated afterwards. Ground time does not have stochastic representation and absorbs possible propagated delays from previous leg, depending on buffer length. Primary gate delay at departure is modelled by random variable.

Representation of the block phase is rather straightforward. By random variable, we model deviation of the block duration from the scheduled block duration. We remark that block time deviation can be also shorter than scheduled if an early arrival occurs. It is in contrast with the gate phase where random variable models only delays.

3.3.1 Crew Operations

Ground movement of the crew members may cause delay propagation as well. If the crew stays on board of the aircraft, time necessary for preparation of the crew is accounted in the minimum ground time. However, if the crew's duty continues on a different aircraft, there is some time necessary for transferring the crew to this aircraft. We refer to this time as minimum sit time. Often, the minimum sit time is greater than the minimum ground time. The minimum sit time is, similarly to minimum ground time, considered to be a constant time period dependent on the airport.

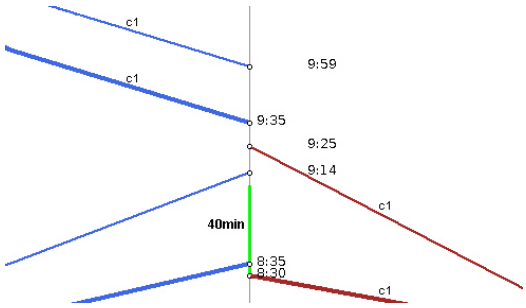


Figure 3.4: Delay propagation due to crew change.

For demonstration, let us consider situation from the right side of Figure 3.3 where in addition, we assume changing crew $c1$ from another aircraft. Crew $c1$ has 65 minutes to change the aircraft from which 25 minutes are minimum sit time and 40 minutes are buffer. As discussed earlier, there is propagated delay to departing leg due to delayed arrival of aircraft by 4 minutes. Since crew arrives with delay of 55 minutes, 15 minutes of delay propagate to departing leg. Therefore, next leg departs with delay of 24 minutes thereof 9 are primary gate delay and 15 minutes are propagated delay.

3.4 Rotational Delay Propagation

Now, we show how to compute distribution of the propagated delay for presented model of the airline operations. First, we offer the formal process of the computation; and afterwards, we give algorithmic details of this computation based on discretisation method introduced in Section 1.3.

3.4.1 Formal Computation of Propagated Delays

Let us demonstrate computation of a propagated delay along a rotation $r = (1, 2, \ldots, k)$ with k legs. We consider random variables

G_i the primary delay at the gate before leg i, $i = 1, 2, \ldots, k$,

B_i the deviation from the scheduled block time of leg i, $i = 1, 2, \ldots, k$.

We derive random variables of propagated and arrival delays along a rotation from the random variables of primary delays. Since both depend also on legs prior to i in rotation r we denote them

AD_i^r the arrival delay of leg i in rotation r, $i = 1, 2, \ldots, k$,

PD_i^r the delay propagated to leg i in rotation r, $i = 1, 2, \ldots, k$.

Let us further denote

$b_{i,i+1}$ the scheduled buffer time between legs i and $i+1$, $i = 1, 2, \ldots, k - 1$.

Notice that $b_{i,i+1}$ are not random variables, but constant values. We also recall that AD_i^r actually represent the deviations from the scheduled arrival times since we allow also early arrivals, not only delayed. Furthermore, we

assume that all random variables G_i and B_i are independent. We will now argue that the delay propagated to leg i in rotation r is given by the recursion

$$PD_1^r = 0, \tag{3.1}$$
$$PD_i^r = \max\left\{PD_{i-1}^r + G_{i-1} + B_{i-1} - b_{i-1,i}, 0\right\}, \ i = 2, 3, \ldots, k. \tag{3.2}$$

To this purpose, we show by induction that the arrival delay and the propagated delay to leg i in rotation r are

$$AD_i^r = PD_i^r + G_i + B_i, \qquad i = 1, 2, \ldots, k \tag{3.3}$$
$$PD_1^r = 0, \quad PD_i^r = \max\left\{AD_{i-1}^r - b_{i-1,i}, 0\right\}, \qquad i = 2, 3, \ldots, k. \tag{3.4}$$

Substitution for AD_i^r gives Equation (3.2).

In fact, since leg 1 is the first leg in rotation r, there is no propagated delay from previous legs, that is, $PD_1^r = 0$. Hence, the arrival delay of leg 1 in rotation r is the sum of gate delay G_1 and block deviation B_1

$$AD_1^r = G_1 + B_1 = PD_1^r + G_1 + B_1.$$

Now, consider leg $i + 1$, for $i \geq 1$. If arrival delay AD_i^r is greater than buffer time $b_{i,i+1}$, then $b_{i,i+1}$ minutes of the delay are absorbed by the ground buffer; only the remaining delay propagates to leg $i + 1$. If arrival delay is smaller than buffer time, no delay propagates to leg $i + 1$. Therefore, the delay propagated to leg $i + 1$ in rotation r is

$$PD_{i+1}^r = \max\left\{AD_i^r - b_{i,i+1}, 0\right\}.$$

The arrival delay of leg $i+1$ in rotation r is the sum of the propagated delay from leg i and primary gate delay G_{i+1} and block deviation B_{i+1}

$$AD_{i+1}^r = PD_i^r + G_{i+1} + B_{i+1}.$$

3.4.2 Implementation

To implement Operation (3.1), respectively Operations (3.3) and (3.4), we engage the discretisation method for numerical computation of convolution introduced in Chapter 1. Since our method is independent of specific distribution classes of primary delays, we omit details on delay distributions which can be found in Chapter 5.

It is apparent that the application of discretisation technique in the computation of propagated delay along aircraft rotation requires some extensions of the method. First, we introduce mixed random variables, which are more appropriate for representation of propagated delay than continuous random variables. Application of mixed random variables, instead of continuous random variables, requires some modifications in implementation of convolution from Section 1.3. Besides, we show the implementation of additional operations over discretised random variables; in particular, shifting the distribution, taking the positive part of the distribution and computing maximum of two distributions.

Mixed Random Variables

Main difference between the computation of the propagated delay and the path's length, discussed in Chapter 1, lies in "'forgetting"' the negative part of the distribution of the propagated delay performed in Operation (3.4). Thinking this through, it becomes clear that resulting distribution of propagated delay PD_i^r is partially discrete since it has positive probability of no propagated delay, and partially continuous when looking at distribution of positive delay length. Natural support of such situation provide mixed random variables. See their formal characterisation in Definition 3.4.1. Notice that random variables of primary gate delay G_i are mixed random variables as well; this is explained in Section 5.4.

Definition 3.4.1. *Random variable X is a mixed random variable, if the sample space of X can be divided into countable set Ω_d and uncountable set Ω_c where all $\omega \in \Omega_d$ have non-negative probability. Moreover, $P[X \in \Omega_d] = p$ and $P[X \in \Omega_c] = 1 - p$.*

In our application, both, propagated delay PD_i^r and gate primary delay G_i are mixed random variables with the same structure. There is only one point in Ω_d, which represents the probability that the length of the delay is zero. The rest of the sample space represents the non-negative delay length and belongs to Ω_c. Hence, we restrict ourselves to the mixed random variables with special structure which can be denoted $X = \left(p_0^X, X_c\right)$ where X_c is a continuous random variable with probability density function f_{X_c} and cumulative density function F_{X_c}; and p_0^X is a probability that outcome of X is

zero. Previous can be formally summarised as follows

$$\begin{aligned}
\Omega_d &= \{0\}\,, \\
\Omega_c &= (0,\infty), \\
P[X=0] &= p_0^X, \\
P[a \le X \le b] &= (1-p_0^X)\int_a^b f_{X_c}(x)dx \qquad \text{for } a,b \in (0,\infty)\,, \\
P[X \le y] &= p_0^X + (1-p_0^X)\int_0^y f_{X_c}(x)dx \qquad \text{for } y \in [0,\infty)\,.
\end{aligned}$$

Discretisation

Let us recall the discretisation of continuous random variables from Section 1.3. Let X be a continuous random variable with probability density function f_X and cumulative density function F_X. We approximate f_X by step function f_{X^k} as follows

$$f_{X^k}(x) = \begin{cases} \alpha_i^{X^k} & \text{for } x \in (k\,(i-1)\,, ki]\,, \quad i = l_{X^k}, l_{X^k}+1, \ldots, u_{X^k}, \\ 0 & \text{else,} \end{cases} \tag{3.5}$$

where

$$\alpha_i^{X^k} = \frac{\int_{k(i-1)}^{ki} f_X\,(x)\,dx}{k} = \frac{F_X\,(ki) - F_X\,(k\,(i-1))}{k}, \quad i = l_{X^k}, l_{X^k}+1, \ldots, u_{X^k}.$$

Finiteness of the representation is guaranteed by the choice of bounds l_{X^k} and u_{X^k} which we define as

$$\begin{aligned}
l_{X^k} &= \max\{z \in \mathbb{Z} \mid f_X(x) < \varepsilon, \forall x < kz\}\,, \\
u_{X^k} &= \min\{z \in \mathbb{Z} \mid f_X(x) < \varepsilon, \forall x > kz\}\,,
\end{aligned} \tag{3.6}$$

for some very small $\varepsilon > 0$. For simplicity, we may use notion $\alpha_i^{X^k}$ also for $i < l_{X^k}$ and $i > u_{X^k}$. In such situation, clearly, $\alpha_i^{X^k} = 0$.

Despite the truncation of the probability density function in the tails, we assume that

$$\int_{-\infty}^{\infty} f_{X^k}(x)dx = \int_{kl_{X^k}}^{ku_{X^k}} f_{X^k}(x)dx = 1. \tag{3.7}$$

This assumption is based on the fact that the mass in the tails is very small, which can be controlled by a proper choice of ε. From the numerical point of view, this inaccuracy can be seen as a rounding error. Deeper discussion on this topic, supported by experimental results, follows in Section 3.6.2.

Under this assumption, f_{X^k} fulfils properties of probability density functions and we denote by X^k the random variable corresponding to f_{X^k}.

Discretisation of mixed random variables is then straightforward. Let $Y = (p_0^Y, Y_c)$ be a mixed random variable. It is necessary to discretise only Y_c, which is done by (3.5). Accordingly, we denote $Y^k = (p_0^{Y^k}, Y_c^k)$ discretised mixed random variable Y.

Convolution

The algorithm computing convolution of two discretised continuous random variables is presented in Section 1.4.3. Here, we state an algorithm for computation of the convolution of a discretised continuous and a discretised mixed random variable as well as of two discretised mixed random variables.

Let $X = (p_0^X, X_c)$ be a mixed random variable and Y be a continuous random variable. Assuming X and Y are independent, their sum $V = X + Y$ is the continuous random variable and holds

$$V = p_0^X Y \oplus \left(1 - p_0^X\right)(X_c + Y). \tag{3.8}$$

It means V is the mixture of random variable Y and $X_c + Y$ with mixture weights p_0^X and $1 - p_0^X$. Mixture of random variables can be defined as shows Definition 3.4.2. In order to avoid possible confusion, we use $+$ operator to denote sum of random variables and $\oplus$ operator to denote mixture of random variables.

Definition 3.4.2 (Mixture). *Let us have random variables $X_1, X_2, \ldots, X_n$ with probability density functions $f_{X_1}, f_{X_2}, \ldots, f_{X_n}$, and mixture weights $\omega_1, \omega_2, \ldots, \omega_n \geq 0$ for which holds $\sum_{i=1}^n w_i = 1$. Random variable $Z = \bigoplus_{i=1}^n w_i X_i$ is mixture of $X_1, X_2, \ldots, X_n$ with probability density function*

$$f_Z(x) = \sum_{i=1}^{n} w_i f_{X_i}(x).$$

We implement (3.8) by approximating X and Y by discretised random variables Y^k and X^k. First, we have to compute the sum of X_c^k and Y^k. Since both are discretised continuous random variables, we already know how to compute their sum (see Section 1.4.3 for details). Then we have to derive weighted sum of probability density functions f_{Y^k} and $f_{X_c^k+Y^k}$. Since f_{Y^k} and $f_{X_c^k+Y^k}$ are step function with equal segment end points, their weighted sum f_{V^k} is also the step function defined as

$$f_{V^k}(x) = \begin{cases} p_0^{X^k}\alpha_i^{Y^k} + \left(1-p_0^{X^k}\right)\alpha_i^{X_c^k+Y^k} & \text{for } x \in (k\,(i-1)\,, ki]\,, \\ & \quad i = l_V, l_V+1, \ldots, u_V, \\ 0 & \text{else,} \end{cases}$$

where $l_{V^k} = \min(l_{X^k}, l_{X_c^k+Y^k})$ and $u_{V^k} = \max(u_{X^k}, u_{X_c^k+Y^k})$.

Now, let us show how to compute the convolution of two mixed random variables. Let $Z = (p_0^Z, Z_c)$ be a mixed random variable independent of X and denote $W = X + Z$. W is mixed random variable $W = \left(p_0^W, W_c\right)$ and can be computed as

$$\begin{aligned} p_0^W &= p_0^X p_0^Z \\ W_c &= \frac{p_0^Z\left(1-p_0^X\right)}{1-p_0^X p_0^Z} X_c \oplus \frac{p_0^X\left(1-p_0^Z\right)}{1-p_0^X p_0^Z} Z_c \oplus \frac{\left(1-p_0^X\right)\left(1-p_0^Z\right)}{1-p_0^X p_0^Z}\left(X_c + Z_c\right). \end{aligned}$$

The implementation of this operation over discretised mixed random variables is similar to previous case since W_c^k is a mixture of random variables X_c^k, Z_c^k, and $X_c^k + Z_c^k$ which we already know how to compute. Formally,

$$f_{W_c^k}(x) = \begin{cases} w_1\alpha_i^{X_c^k} + w_2\alpha_i^{Z_c^k} + w_3\alpha_i^{X_c^k+Z_c^k} & \text{for } x \in (k\,(i-1)\,, ki]\,, \\ & \quad i = l_{W_c^k}, l_{W_c^k}+1, \ldots, u_{W_c^k}, \\ 0 & \text{else,} \end{cases}$$

where $w_1 = \frac{p_0^{Z^k}\left(1-p_0^{X^k}\right)}{1-p_0^{X^k}p_0^{Z^k}}$, $w_2 = \frac{p_0^{X^k}\left(1-p_0^{Z^k}\right)}{1-p_0^{X^k}p_0^{Z^k}}$, $w_3 = \frac{\left(1-p_0^{X^k}\right)\left(1-p_0^{Z^k}\right)}{1-p_0^{X^k}p_0^{Z^k}}$, $l_{W_c^k} = 1$, and $u_{W_c^k} = \max(u_{X_c^k}, u_{Z_c^k}, u_{X_c^k+Z_c^k})$.

Shifting

Operation (3.4) involves the subtraction of a non-negative constant from the continuous random variable what corresponds to delay absorption by ground

buffer of length b. Let us denote $Y = X - b$ and let b be a non-negative constant. Clearly, probability density function f_Y of Y is defined as

$$f_Y(x) = f_X(x + b) \quad x \in \mathbb{R}.$$

As for all other operations, the implementation of this one is carried out over discretised random variables and the result of the operation has to be a step function with fixed segment end points ik for $i \in \mathbb{Z}$. In case $b = kj$ for some $j \in \mathbb{Z}$ the implementation is straightforward and we just shift the indices of the segments.

$$f_{Y^k}(x) = \begin{cases} \alpha_{i+j}^{X^k} & \text{for } x \in (k(i-1), ki], \quad i = l_{X^k} - j, l_{X^k} - j + 1, \ldots, u_{X^k} - j, \\ 0 & \text{else.} \end{cases}$$

If $b \neq jk$, for all $j \in \mathbb{Z}$, we transform f_{X^k} in a way that segment end points as well as total mass of the function are preserved. In general, let $b = jk + h$ for $0 \leq h < k$, $j \in \mathbb{Z}$. We compute the probability density function of Y^k as

$$f_{Y^k}(x) = \begin{cases} \frac{h}{k}\alpha_{i+1}^{X^k} & \text{for } x \in (k(i-1), ki], \quad i = l_{X^k} - j - 1, \\ \frac{h}{k}\alpha_{i+1}^{X^k} + \frac{k-h}{k}\alpha_i^{X^k} & \text{for } x \in (k(i-1), ki], \quad i = l_{X^k} - j, l_{X^k} - j + 1, \\ & \qquad \ldots, u_{X^k} - j - 1, \\ \alpha_i^{X^k}\frac{k-h}{k} & \text{for } x \in (k(i-1), ki], \quad i = u_{X^k} - j, \\ 0 & \text{else.} \end{cases}$$

Maximum

Furthermore, in Operation (3.4) we take non-negative part of the delay which formally corresponds to taking maximum of the delay value and zero. After giving an implementation of this operation, we state the implementation of more general operation which takes the maximum of two mixed random variables. Thoughtful reader may have noticed that, so far, we did not utilise such operation. This operation is required in Section 3.5.

Let $Y = \max\{X, 0\}$ where X is a continuous random variable. We see that Y is the mixed random variable for which

$$\begin{aligned} P[Y = 0] &= P[X \leq 0], \\ P[Y = x] &= P[X = x] \quad \text{for } x > 0. \end{aligned}$$

Therefore, we define $Y = \left(p_0^Y, Y_c\right)$ as

$$\begin{aligned} p_0^Y &= \int_{-\infty}^{0} f_X(x)dx, \\ f_{Y_c}(x) &= \frac{f_X(x)}{1-p_0^Y} \qquad x \in (0,\infty). \end{aligned}$$

Now, let us have a look at implementation of this transformation over discretised random variables. We denote $Y^k = \max\left\{X^k, 0\right\}$ which is computed as

$$\begin{aligned} p_0^{Y^k} &= \sum_{i=l_{X^k}}^{0} k\alpha_i^{X^k}, \\ f_{Y_c^k}(x) &= \begin{cases} \frac{1}{1-p_0^{Y^k}}\alpha_i^{X^k} & \text{for } x \in (k(i-1), ki], \quad i = 1, 2, \ldots, u_{X^k}, \\ 0 & \text{else.} \end{cases} \end{aligned} \tag{3.9}$$

Computation of the maximum of two mixed random variables requires slightly different approach. Let $W = \max\{X, V\}$ and X, V be independent mixed random variables with cumulative density functions F_X and F_V, respectively. Instead of concentrating on the probability density function of W we look at the cumulative density function. We know that

$$F_W(x) = F_X(x)F_V(x) \quad x \in \mathbb{R}. \tag{3.10}$$

As (3.5) shows, knowledge of F_W allows us to compute discretisation of W. Actually, it is enough to know values of F_W in the segment end-points since

$$\alpha_i^{W^k} = \frac{\int_{k(i-1)}^{ki} f_W(x)dx}{k} = \frac{F_W(ki) - F_W(k(i-1))}{k}, \quad i = l_{W^k}, l_{W^k}+1, \ldots, u_{W^k}. \tag{3.11}$$

Then (3.10) and (3.11) are satisfactory for implementation. Let us recall that $X^k = \left(p_0^{X^k}, X_c^k\right)$, $Y^k = \left(p_0^{Y^k}, Y_c^k\right)$. Cumulative density functions of X^k and

V^k are defined

$$\begin{aligned} F_{X^k}(ik) &= p_0^{X^k} + \left(1 - p_0^{X^k}\right) F_{X_c^k}(ik) \\ &= p_0^{X^k} + \left(1 - p_0^{X^k}\right) k \sum_{j=0}^{i} \alpha_j^{X_c^k}, \\ F_{V^k}(ik) &= p_0^{V^k} + \left(1 - p_0^{V^k}\right) F_{V_c^k}(ik) \\ &= p_0^{V^k} + \left(1 - p_0^{V^k}\right) k \sum_{j=0}^{i} \alpha_j^{V_c^k}. \end{aligned}$$

We derive $W^k = \left(p_0^{W^k}, W_c^k\right)$ as

$$\begin{aligned} p_0^{W^k} &= p_0^{X^k} p_0^{V^k} \\ f_{W_c^k}(x) &= \begin{cases} \frac{1}{1-p_0^{W^k}} \alpha_i^{W^k} & \text{for } x \in (k(i-1), ki], \\ & \quad i = 1, 2, \ldots, \max\{u_{X_c^k}, u_{V_c^k}\}, \\ 0 & \text{else} \end{cases} \end{aligned}$$

and

$$\begin{aligned} \alpha_i^{W^k} &= \frac{F_{X^k}(ki) F_{V^k}(ki) - F_{X^k}(k(i-1)) F_{V^k}(k(i-1))}{k} \\ &= k \left(1 - p_0^{V^k}\right) \left(1 - p_0^{X^k}\right) \alpha_i^{X_c^k} \alpha_i^{V_c^k} + \left(1 - p_0^{X^k}\right) \alpha_i^{X_c^k} F_{V^k}((i-1)k) + \\ &+ \left(1 - p_0^{V^k}\right) \alpha_i^{V_c^k} F_{X^k}((i-1)k). \end{aligned}$$

As you may have noticed we assume that X and Y are mixed random variables, which is the case in our application in Section 3.5. Please notice that the operation can be defined, in a similar way, for two continuous random variables, or mixed and continuous random variable as well. The only difference is in definition of cumulative density functions and bounds l_{W^k} and u_{W^k}.

Remark

For our application it is satisfactory to focus on mixed random variables with single point in Ω_d. Thus, we defined only operations over these special mixed random variables. Clearly, one can define all aforementioned operations also

for general mixed random variables. Notice that the complexity of the operations over such random variables depends on the number of non-zero steps but also on the cardinality of set Ω_d. Moreover, every applied convolution increases size of Ω_d rapidly what may cause significant slow down or even intractability of the method for general mixed random variables.

3.5 Non-rotational Delay Propagation

Late aircraft are not the only source of propagated delays. Operation of the legs may depend on other resources like crew, cargo, and passengers. If these resources get delayed on their previous legs covered by a different aircraft, the delay propagates to the current leg. All propagated delays not caused by the aircraft are called non-rotational propagated delays. Figure 3.4 shows example of non-rotational delay propagation due to the changing crew. Through out this section, we speak about the most common source of non-rotational delay, which is the crew; although the same applies for other resources changing between rotations.

3.5.1 Formal Computation of Propagated Delays

In addition to notation and variables introduced in Section 3.4.1, let us denote

L the set of all legs,

R the set of all rotations,

P the set of all pairings,

$\mathcal{R}$ set of rotations covering each leg exactly once, $\mathcal{R} \subset R$.

We consider a rotation $r = 1, 2, \ldots, k$ with k legs and $r \in \mathcal{R}$. Furthermore, along with random variables PD_i^r, G_i, B_i and, AD_i^r, we introduce

${}^{\mathcal{R}}APD_i^r$ the delay propagated to leg i in rotation r by the aircraft under consideration of set of rotations $\mathcal{R}$, for $i = 1, 2, \ldots, k$,

${}^{\mathcal{R}}CPD_i^r$ the delay propagated to leg i in rotation r due to the changing crew under consideration of set of rotations $\mathcal{R}$, for $i = 1, 2, \ldots, k$,

$b_{i,j}^c$ the scheduled buffer time for a crew member changing between legs i and j, where leg i is covered by different rotation than leg j, for $i, j \in L$.

Notice that delay propagation depends not only on currently considered rotation but also on rotations covering other legs. We discuss this topic later in Section 3.5.3. Thus, we use index $\mathcal{R}$ also for random variable of propagated ${}^{\mathcal{R}}PD_i^r$ and arrival delay ${}^{\mathcal{R}}AD_i^r$.

Let us recall that $b^c_{i,i+1}$ are constant values. Since the minimum ground time may differ from the minimum sit time, also $b^c_{i,i+1}$ may differ from $b_{i,i+1}$. We furthermore define functions

$$\begin{aligned} \text{crew} &: L \to 2^P \\ \text{rot} &: 2^R \times L \to R \end{aligned}$$

which for given leg $l \in L$ return the set of pairings, or the rotation, covering the leg under consideration of $\mathcal{R}$, respectively. We assume that every leg can be operated by several cockpit and cabin crew members. Each crew member has its own pairing, that is, the set of legs to operate. For accessing legs in the rotation or pairing, we define function

$$\text{pred} : P \cup R \times L \to \{L, \emptyset\} \tag{3.12}$$

that returns previous leg of given leg in pairing, rotation, respectively. The function returns empty set, if the leg on input is the first leg of the pairing or rotation, respectively. Furthermore, let $M_i \subset L$ be the set of all legs from which leg i in rotation r awaits an incoming crew member

$$M_i = \{l \in L | l = \text{pred}(p, i) \ \& \ p \in \text{crew}(i)\}\,. \tag{3.13}$$

Now, we show that propagated delay to leg i in rotation r is given by the recursion

$$\begin{aligned}
{}^{\mathcal{R}}APD_1^r &= 0, && && (3.14)\\
{}^{\mathcal{R}}APD_i^r &= \max\left\{{}^{\mathcal{R}}AD_{i-1}^r - b_{i-1,i}, 0\right\} && i = 2, 3, \ldots, k, && (3.15)\\
{}^{\mathcal{R}}PD_i^r &= \max\left\{{}^{\mathcal{R}}APD_i^r, {}^{\mathcal{R}}CPD_i^r\right\} && i = 1, 2, \ldots, k, && (3.16)\\
{}^{\mathcal{R}}AD_i^r &= {}^{\mathcal{R}}PD_i^r + G_i + B_i && i = 1, 2, \ldots, k, && (3.17)\\
{}^{\mathcal{R}}CPD_i^r &= \max_{l \in M_i \setminus \{\text{pred}(r,i)\}} \left\{\max\left\{{}^{\mathcal{R}}AD_l^{\text{rot}(\mathcal{R},l)} - b^c_{l,i}, 0\right\}\right\} i = 1, 2, \ldots, k. && && (3.18)
\end{aligned}$$

Clearly, there is no propagated delay to the first leg of the rotation due to the late aircraft; therefore, ${}^{\mathcal{R}}APD_1^r = 0$. But, the crew members operating

the first leg may have some legs assigned beforehand. The total propagated delay to the first leg equals the propagated delay due to the changing crew members.

$$^{\mathcal{R}}PD_1^r = {}^{\mathcal{R}}CPD_1^r = \max\left\{{}^{\mathcal{R}}APD_1^r, {}^{\mathcal{R}}CPD_1^r\right\}$$

Notice that the second equality follows from non-negativity of $^{\mathcal{R}}CPD_1^r$.

The arrival delay of leg 1 is the sum of the propagated delay to this leg, gate delay G_1, and block deviation B_1

$$^{\mathcal{R}}AD_1^r = {}^{\mathcal{R}}PD_1^r + G_1 + B_1.$$

Now, let us look at leg $i+1$, for $i \geq 1$. If the aircraft arrives with $^{\mathcal{R}}AD_i^r$ minutes of delay, $b_{i,i+1}$ minutes of the delay are absorbed by the buffer, the rest is propagated to the next leg. If the arrival delay is smaller than $b_{i,i+1}$, no delay propagates to leg $i+1$; therefore,

$$^{\mathcal{R}}APD_{i+1}^r = \max\left\{{}^{\mathcal{R}}AD_i^r - b_{i,i+1}, 0\right\}.$$

Leg $i+1$ may depart only if the aircraft is present and all crew members are on board. Thus, the propagated delay to leg $i+1$ in rotation r is the maximum of the propagated delay due to the aircraft and crew members

$$^{\mathcal{R}}PD_{i+1}^r = \max\left\{{}^{\mathcal{R}}APD_{i+1}^r, {}^{\mathcal{R}}CPD_{i+1}^r\right\}.$$

The propagated delay due to the crew is as big as the delay of the latest incoming crew member. M_i is the set of all legs from which at least one crew member changes to leg i. Clearly, we do not consider crew flying along rotation r. As we already explained for the case of rotational propagated delay, $\max\left\{{}^{\mathcal{R}}AD_l^{\mathrm{rot}(\mathcal{R},l)} - b_{l,i}^c, 0\right\}$ is the delay that propagates from leg l to leg i. Then the propagated delay due to the changing crew is

$$^{\mathcal{R}}CPD_i^r = \max_{l \in M_i \setminus \{\mathrm{pred}(r,i)\}} \left\{\max\left\{{}^{\mathcal{R}}AD_l^{\mathrm{rot}(\mathcal{R},l)} - b_{l,i}^c, 0\right\}\right\}.$$

Recursion Formula (3.14)-(3.18) is an extension of Formula (3.1)-(3.2) where in addition, propagated delay due to changing crew $^{\mathcal{R}}CPD_i^r$ is considered. In fact, if $^{\mathcal{R}}CPD_i^r = 0$, for every i, that is, no delay propagation due to crew occurs, $^{\mathcal{R}}PD_i^r = {}^{\mathcal{R}}APD_i^r$ and Formulae (3.14)-(3.18) reduce to Formulae (3.1)-(3.2).

3.5.2 Implementation

Section 3.4.2 documents the implementation of all operations necessary for evaluation of Formulae (3.14)-(3.18). Indeed, Formula (3.14) and (3.17) are similar to Formula (3.1) and (3.2), respectively.

Formulae (3.16) and (3.18) require, in addition, computation of the maximum of two or more random variables. This operation is defined for two random variables in Section 3.4.2. Computation of the maximum of more than two random variables is achieved by repeated application of this operation.

Nevertheless, computation of non-rotational delay propagation brings two more difficulties that need to be addressed, which are caused by possible interaction of the rotations.

3.5.3 Dependency of Random Variables

In our implementation of convolution and operation of maximum, we assume that input random variables are independent, see Section 1.4.3 and Section 3.4.2. Unfortunately, this assumption does not have to be fulfilled for operation of maximum when non-rotational delay propagation is assumed. In such situation our approach gives a lower bound of exact value, please see Observation 3.5.1.

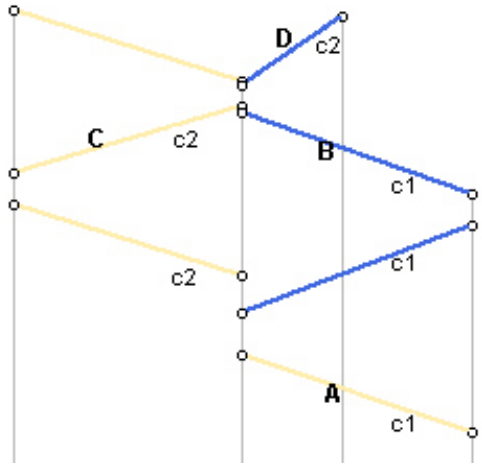


Figure 3.5: Arrival delay of legs B and C are dependent because of crew $c1$ and $c2$ that change the aircraft.

Figure 3.5 shows a situation where this is the case. The figure pictures the propagation of the primary delay from leg A to the consecutive leg of the rotation, and also to another rotation, due to changing crew $c1$. Therefore, the delays of both rotations are dependent; in particular, arrival delay of leg B and C are dependent. Hence, when we compute propagated delay to leg D the assumption about independence of propagated delay from leg B and leg C does not hold.

Solution methods from PERT Networks

Let us examine known solution techniques to tackle this situation. The similar problem arises in the computation of the distribution of completion time in a stochastic PERT (Program Evaluation and Review Technique) networks. PERT network models, for example, a project that is decomposed into tasks or jobs that may be performed in parallel or sequentially. Start of some tasks may depend on prior fulfilment of some other tasks. Every task has some stochastic duration characterised by some probability distribution. Knowledge of the project completion time distribution or the longest path in the graph helps to manage the project and to improve the planning of the project.

Formally, a PERT network is an acyclic oriented graph representing for example project divided into jobs. The vertices of the graph can be viewed as tasks, or jobs, in the project. Duration of each job is given by a random variable. The graph structure captures dependencies between the jobs by arcs; the job can be started only after all jobs on ingoing arcs are finished. When all ingoing jobs are finished, we say that the vertex, or job, is ready (to start). The goal is to find the distribution of the project duration which corresponds to the length of the path from the source to the sink. This problem was first stated by Malcolm et al. (1959) [81].

O'Connor (2006) [86] summarises approaches for solving this problem and recognises three basic classes of methods: bounding, approximate, and exact.

The **exact methods** compute the distribution of the completion time numerically under assumption that the task lengths follow discrete distributions and for each vertex all incoming sub-paths are independent. Dependencies of the paths are broken by fixing the length of some tasks to constant times. The set of arcs breaking all dependencies in the network is called the conditioning set (O'Connor (2006) [86]) or the uniformly directed cutset (Sigal, Pritsker & Solberg (1980) [98]). The choice of the conditioning set is not unique and

the literature proposes various algorithm for detection and selection of the conditioning set. In order to compute the completion distribution of the project, the completion time for all value combinations of the conditioning set arcs has to be computed. It is clear, that mentioned exact methods are very inefficient and applicable only on small networks of for very simple task length distributions.

The **approximate methods** estimate the distribution of the completion by simulation. Dependencies of the paths do not play role in this method but accurate estimates require many simulations. Knowledge of the conditional set may speedup the method since one can sample only on vertices from the conditioning set and the rest can be computed exactly. See, Burt & Garman (1971) [30].

Different approach present so called **bounding methods**. These methods compute approximate distributions of the completion time which are provable lower, or upper bounds on actual distributions. They do not rely on conditioning sets and are computable efficiently. Kleindorfer (1971) [68], Shogan (1977) [97], Fulkerson (1962) [58] and Dodin (1985) [44] propose various bounding methods. Shogan (1977) [97] studies tightness of bounds proposed in Kleindorfer (1971) [68] and Fulkerson (1962) [58] and provides small computational study.

Application in Airline Planning

Structure of the airline planning application differs from that of project scheduling in PERT Networks. Luckily, crew changes appear in the schedule seldom and thus, dependencies between rotations are observed rarely.

On the other hand, the problem of finding the distribution of the path length is only a sub-problem, which has to be solved many times during the optimisation, for further details see Chapter 4. The efficiency of applied method is, therefore, of very high importance.

Thus, we do not detect possible dependencies between the paths and we apply Formula (3.10) also in case considered random variables are dependent. In fact, this method is known as Kleindorfer's lower bound and is proposed by Kleindorfer (1971) [68]. He proves that this method delivers a lower bound on the distribution in the network.

Observation 3.5.1. *Formula (3.10) gives a lower bound on cumulative density function of maximum of two random variables. If the random variables are independent, the formula defines cumulative density function exactly.*

Proof. See Kleindorfer (1971) [68]. □

3.5.4 Dependency of Delay on Other Rotations

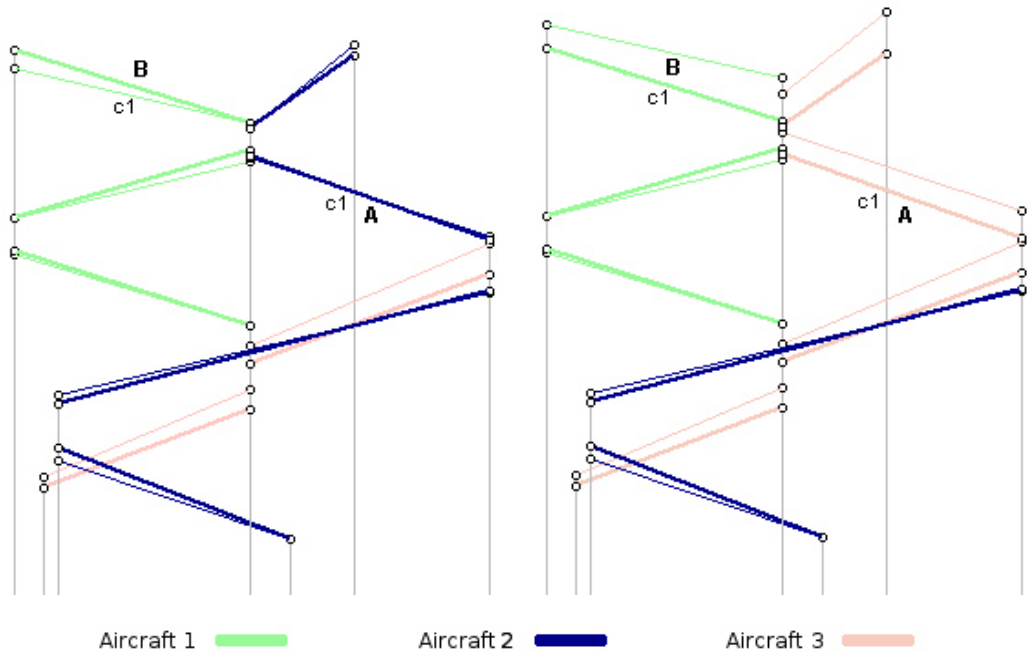


Figure 3.6: Demonstration of dependency of propagated delay of the rotation on another rotations.

Non-rotational delay propagation introduces also other difficulty, namely that the distribution of propagated delay of the rotation may depend on the choice of rotations covering other legs. Figure 3.6 illustrates such situation. There are three aircraft rotations covering the same set of legs in two different ways. The rotation of aircraft 1 stays in both cases unchanged, but the rotations of aircraft 2 and aircraft 3 are different. Due to crew $c1$, the arrival delay of leg A propagates to leg B. On the left, leg A is covered by aircraft 2 while on the right is covered by aircraft 3. Clearly, the arrival delay of leg A depends

Algorithm 2: The algorithm for computation of the propagated delay with consideration of non-rotational delay propagation.

Input : Set of legs L, set of pairings $\mathcal{P}$, set of rotations $\mathcal{R}$ covering each leg in L, constant values $b_{i,j}$, $b^c_{i,j}$ for $i, j \in L$, and random variables G_l, B_l for $l \in L$.

Output: Distribution of propagated delay to each leg in L.

1 sort L topologically as $l_1, \ldots, l_n$ by increasing scheduled arrival time;
2 **for** $l := l_1$ **to** l_n **do**
3 $\quad r := \mathrm{rot}\,(\mathcal{R}, l)$;
4 $\quad l_p := \mathrm{pred}\,(r, l)$;
5 $\quad$ **if** $l_p = \emptyset$ **then**
6 $\quad\quad {}^{\mathcal{R}}APD^r_l := 0$;
7 $\quad$ **else**
8 $\quad\quad {}^{\mathcal{R}}APD^r_l := \max\left\{{}^{\mathcal{R}}AD^r_{l_p} - b_{l_p,l}, 0\right\}$;
9 $\quad {}^{\mathcal{R}}CPD^r_l := \max_{q \in M_l \setminus \{l_p\}} \left\{\max\left\{{}^{\mathcal{R}}AD^{\mathrm{rot}(\mathcal{R},q)}_q - b^c_{q,l}, 0\right\}\right\}$;
10 $\quad {}^{\mathcal{R}}PD^r_l := \max\left\{{}^{\mathcal{R}}APD^r_l, {}^{\mathcal{R}}CPD^r_l\right\}$;
11 $\quad {}^{\mathcal{R}}AD^r_l := {}^{\mathcal{R}}PD^r_l + G_l + B_l$;
12 **return** $\left\{{}^{\mathcal{R}}PD^{\mathrm{rot}(\mathcal{R},l)}_l \mid l \in L\right\}$;

on preceding legs and therefore, may differ in both pictured routings as well the propagated delay to leg B. One such situation is demonstrated by thin lines, which represent one particular delay scenario applied to both routings. You can see that leg B departs on time on the left routing while there is some propagated delay due to crew $c1$ on the right routing.

This example shows that in order to compute propagated delay along a rotation with consideration of non-rotational delay propagation, we have to know rotations covering legs from which non-rotational delay propagates. Moreover, the distributions of the arrival delay on these legs have to be known.

Here, we propose an algorithm that for given rotations and pairings computes propagated delay of all rotations in parallel. We suppose that all legs are covered by exactly one aircraft rotation and each leg has assigned all necessary crew members. Steps 6-11 compute propagated and arrival delay of the leg as defined by Formulae (3.14)-(3.18). Algorithm 2 computes the

distribution of propagated delay of the legs in increasing order according to the arrival time of the leg (Step 1). This order of leg processing ensures that all distributions of arrival delay desired in Step 9 and Step 8 are already computed.

Clearly, if one requires to compute the propagated delay for a single rotation, it is not necessary to compute propagated delay for all legs and rotations. We can recursively identify only rotations that are influencing this rotation and then terminate the algorithm after the propagated delay of the last leg in the rotation is computed.

3.6 Performance and Accuracy

This section is devoted to study of computational performance of the discretisation method. We demonstrate behaviour of our method on large-scale real world instances of the tail assignment problem since the discretisation method is applied in the path generation. Further details about the tail assignment problem are subject of Chapter 4.

Here, we compare accuracy and computational efficiency of the method for various step sizes. Clearly, the smaller discretisation step we choose, the higher accuracy of the results we obtain, but the slower method gets. Our main interest is to prove that certain step sizes allow accurate and fast computation and thus the method is practical for real word applications. Further results showing practical impact on robustness of the planning is provided in Chapter 7.

3.6.1 Asymptotic Complexity

First, let us look on computational complexity of each of operations maximum, shifting, and convolution which combined allow computation of propagated delay in airline network. Complexity of these operations depends on the number of non-zero steps of discretised probability density functions.

Let us denote by s the maximum number of non-zero steps of probability density functions of B_i^k and G_i^k. Convolution is the most computationally intensive operation. It is easy to see that convolution of two discretised

probability density functions, with at most s non-zero steps, runs in $O(s^2)$ while operations of shifting and maximum can be evaluated in $O(s)$.

Let us look how this complexity translates to complexity of computation of propagated delay for whole rotation. As we proceed along the rotation, the span of resulting distribution of propagated delay grows with each computed convolution.

Shifting the probability density function does not influence the span; cutting off the negative part of the probability density function, shortens the span by some constant factor. Convolution of density functions with s_1 and s_2 non-zero steps produces probability density function with $s_1 + s_2$ non-zero steps. Thus, after computing i consecutive convolutions of probability density functions with at most s non-zero steps, we obtain a probability density function with at most is non-zero steps.

Convolution of resulting probability density function with at most is steps with another probability density functions with at most s requires $O(is^2)$ operations. Altogether, in the rotation with n legs, the computation of the last PD has complexity $O(ns^2)$ and computation of the propagated delay along the whole rotation can be done in $O(n^2s^2)$.

3.6.2 Speed-up

scenario	# legs	# aircraft	# crew changes
SC1	100	16	4
SC2	149	23	4
SC3	90	20	-
SC4	114	15	-
SC5	103	21	-

Table 3.1: Description of instances.

In our instances, n typically grows up to ten. Although it is rather small constant, it already causes significant slow down of the method. As an illustration, we show results for five standard one day instances of the robust tail assignment problem described in Table 3.1. In the second column of

Table 3.2, we show maximum number of non-zero steps of some probability density function that was generated during the optimisation. The third column shows total time, in seconds, spent on evaluation of operations over discretised random variables. All results are produced for discretisation with step size of one minute.

	full length		truncated length	
	maximum steps	time [s]	maximum steps	time [s]
SC1	2885	450	391	46
SC2	3018	214	402	24
SC3	2484	29	366	4
SC4	2933	107	392	18
SC5	2490	30	405	6

Table 3.2: Comparison of the maximum length of the probability density function and CPU time spent on operations over discretised random variables with step size of one minute.

You can see that maximum number of non-zero steps grows up to 3000. Obviously, the chance that delay of 3000 minutes propagates to some leg is purely theoretical. Actually, the great part of such distribution has very low probability. We, therefore, apply the same truncating procedure as we apply at the initial discretisation of the continuous random variables, see description of Operation (3.6). That is, we round the steps, in the tail of distribution, with value close to zero. Such steps do not have to be represented what fastens the operations over the probability density functions.

Repeated truncation increases approximation error. The resulting step function is not a probability density function any more since it does not integrate to one. To keep the error small we do not truncate the function after each convolution, but only when computing the distribution of propagated delay, namely when we take non-negative part of the function, see Operation (3.9).

Thus, in order to get properly defined probability density function, we add the truncated probability to $p_0^{Y^k}$. We reformulate the implementation of (3.9)

as follows. Let $Y^k = \max\left\{X^k, 0\right\}$ then

$$p_0^{Y^k} = \sum_{i=l_X^k}^{0} k\alpha_i^{X^k} + \sum_{j=q+1}^{u_{X^k}} k\alpha_j^{X^k} \tag{3.19}$$

$$f_{Y_c^k}(x) = \begin{cases} \frac{1}{1-p_0^{Y^k}}\alpha_i^{X^k} & \text{for } x \in (k(i-1), ki], \quad i = 1, 2, \ldots, q, \\ 0 & \text{else,} \end{cases}$$

$$\text{where } q = \min\left\{y \in l_{X^k}, l_{X^k}+1, \ldots, u_{X^k} \mid \forall x > yk: \ \alpha_{\lfloor \frac{x}{k} \rfloor}^{X^k} < \varepsilon\right\}.$$

Please note that another option, to overcome problems of truncation, is to define new distribution by scaling the truncated probability density function. Such approach is more natural, but according to our experiments performs worse due to numerical issues. These problems are caused by the scaling factor, which is very close to one.

Introduced truncation leads to significant speed up of the method. Table 3.2 shows the experimental results for five randomly chosen one day instances. The second and the third column refer to values obtained for the original method, that is, method where $Y = \max\{X, 0\}$ is implemented by (3.9). The fourth and the fifth column show values where the column generation framework for solving the robust tail assignment uses improved implementation of the method, which truncates the distribution of propagated delay, that is, implements $Y = \max\{X, 0\}$ as (3.19). In our experiments, we used truncation threshold $\varepsilon = 10^{-6}$ and all instances are discretised with the step of one minute. You can see that the truncation decreases length of the represented probability density functions significantly. Maximum represented length of the propagated delay is at about 400 minutes what is up to seven times less than before truncation. Finally, computational time drops down up to nine times. All presented computations are executed on a 64-bit desktop PC with two Intel(R) Core(TM) 2 processors with 3.0 GHz and 8 GB of RAM memory, running openSuse Linux 11.2. Our code is implemented in `C++` and was compiled using `g++` 3.4.

Description of distributions of random variables for the gate delay and block deviation are given in Chapter 5.

3.6.3 Computational Results

Application of the discretisation method to a real word application requires some tricks, for example aforementioned truncation, which makes theoretical analysis of the accuracy problematic. Nevertheless, we refer interested reader to Section 1.5 where theoretical analysis of the core of the method, computation of the convolution, is presented. Here, we focus on practical study of the accuracy of the method.

Our experiments show performance of the method within the column generation framework. We solve the robust tail assignment problem, which is introduced in Chapter 4. The discretisation method is implied in the path generation to compute the probability of delay propagation along the rotations.

We, therefore, use probability of delay propagation as a measure for benchmarking the accuracy of the discretisation. The accuracy and the running time of the discretisation method are controlled in terms of the step size parameter. The smaller the step size, the higher accuracy, but also higher computational effort. We now investigate the tradeoff between speed and accuracy, as well as accuracy per se.

We start with an experiment that is designed to demonstrate the speed of the discretisation algorithm. In order to eliminate influences from different execution paths of the column generation algorithm, we simply use the discretisation algorithm to compute the probability of delay propagation for fixed schedules. That is, we take the rotations of the schedule and compute the distributions of propagated delay. We know that distributions of propagated delay are mixed random variables $PD_j^{rk} = (p_0^{PD_j^{rk}}, PD_{j\,c}^{rk})$ then, the probability of delay propagation equals $1 - p_0^{PD_j^{rk}}$. Since the rotations are fixed, different value of the discretisation step size changes only accuracy of the computed probability and computation time, not the solution itself.

First, we compute probability of delay propagation (PDP) without considering the crew pairings; second, we consider also delay propagation due to the changing crew. We present results for instances SC1 and SC2 from Table 3.1. The instances include four crew changes and the rotations involve up to eight legs.

Table 3.3 shows the effect of six different step sizes ranging from 0.1 to 4.0 minutes. The second column in both subtables gives the step size in minutes.

		without crew			with crew		
	step size [m]	CPU [s]	PDP [#]	error [%]	CPU [s]	PDP [#]	error [%]
SC1	0.1	34.0	25.0586	0.09	39.2	26.0876	0.03
SC1	0.5	1.4	25.0672	0.06	1.6	26.1027	0.02
SC1	1.0	0.4	25.0917	0.04	0.4	26.1353	0.15
SC1	2.0	0.1	25.2227	0.56	0.1	26.2819	0.71
SC1	3.0	0.1	25.4775	1.58	0.1	26.5521	1.75
SC1	4.0	0.1	25.7657	2.73	0.1	26.9575	2.91
SIM			25.0820			26.0964	
		without crew			with crew		
	step size [m]	CPU [s]	PDP [#]	error [%]	CPU [s]	PDP [#]	error [%]
SC2	0.1	51.2	44.1227	0.04	58.9	45.4622	0.05
SC2	0.5	2.1	44.1360	0.01	2.4	45.4759	0.02
SC2	1.0	0.6	44.1743	0.08	0.6	45.5152	0.06
SC2	2.0	0.2	44.3906	0.57	0.2	45.7334	0.54
SC2	3.0	0.1	44.8245	1.55	0.1	46.1727	1.51
SC2	4.0	0.1	45.2921	2.40	0.1	46.5545	2.35
SIM			44.1410			45.4857	

Table 3.3: Comparison of accuracy and speed for various discretisation step sizes on schedules with fixed rotations.

The fourth, the fifth and the sixth column state the computation CPU time in seconds, computed value of PDP, and the error of the computation in percent respectively. Next three columns show the same for the scenario optimised with consideration of the crew pairings.

To determine the approximation error, we estimate the "true" probability of delay propagation of a schedule by averaging over the results of $500,000$ random runs of Ops Simulator. For details about Ops Simulator, we refer reader to Chapter 6. The simulated value is stated in the last row of each

		without crew			with crew		
	step size [m]	CPU [s]	PDP (opt)	PDP (sim)	CPU [s]	PDP (opt)	PDP (sim)
SC1	0.1	4446	19.7268	19.7478	4220	23.3942	23.3642
SC1	0.5	206	19.7362	19.7528	177	23.4234	23.3579
SC1	1.0	46	19.7450	19.7408	40	23.4288	23.3579
SC1	2.0	18	19.8693	19.7537	10	23.5460	23.3579
SC1	3.0	8	20.0651	19.7386	6	23.7779	23.3584
SC1	4.0	5	20.3353	19.7635	4	24.0664	23.3579
SC2	0.1	2796	34.8927	34.9200	1505	42.4405	42.4338
SC2	0.5	134	34.9236	34.9368	71	42.3651	42.3476
SC2	1.0	24	34.8938	34.8795	16	42.3906	42.3400
SC2	2.0	8	35.4122	35.2313	4	42.6030	42.3442
SC2	3.0	4	35.5942	35.0615	2	43.0354	42.3556
SC2	4.0	3	35.7466	34.9068	1	43.3621	42.3405

Table 3.4: Comparison of accuracy and running time for various discretisation step sizes on optimised schedules.

sub-table.

You can see, that the error is in general very small, even a relatively coarse discretisation step size of two minutes provides very accurate results. On the other hand, discretisation step sizes of three and more minutes do not bring any significant speed up of the method while approximation error increases.

Due to the truncation, the method converges to slightly smaller values than we estimated by the simulation. We also see that the PDP value grows with increasing discretisation step size. These two facts together explain surprising behaviour, where we observe more accurate results for discretisation step 0.5 and 1 than for 0.1.

Another important question is what happens when we apply the discretisation method with various step sizes in the optimisation is demonstrated by another experiment. We, again, use scenarios SC1 and SC2 with and without consideration of the crew pairings. The results are summarised in Table 3.4.

Throughout the column generation, many rotations are constructed and many delay distributions are computed. Therefore, the running times, listed in the columns labelled CPU, are much higher than those in Table 3.3. Due to the approximation error, the same column has different PDP value for each discretisation step size what may cause different execution paths of the algorithm and algorithm may converge to different solution. Therefore, we cannot directly compare solutions by the computed PDP. Columns labelled PDP (opt) give the PDP value of the solution generated by the optimiser. This value is, however, compared to the PDP value obtained from Ops Simulator as an average of $100,000$ random simulation runs. This value is listed in columns labelled PDP (sim).

You can be seen that solution quality is insensitive to the discretisation step size in tested range, that is, even with very coarse discretisations, the method computes very good solutions.

In other words, the important point is to take delay propagation into account. If one does so, the quality of the discretisation does not influence the quality of the solution much. However, the objective value computed by the optimiser becomes increasingly worse estimate of the "true" objective with growing step size.

We finally take a closer look at the probability of delay propagation on individual rotations. For practical application it is important to know that the computed values are good estimators of the "true" values for each leg and rotation, not only on average for the whole schedule due to averaging values with higher deviations.

Table 3.5 reports PDP values on the individual legs of three rotations from the optimised solution of instance SC2. We compare PDP for discretisation step sizes of one and four minutes, respectively. The columns of the table state, for each leg, the scheduled block time in minutes, ground buffer time after the leg in minutes, simulated PDP values, and optimised PDP values as well as the relative error for discretisation step sizes of one and four minutes, respectively.

The solution quality is again very good. For a discretisation step size of one minute, the PDP is forecast with an error less than one percent on each individual leg, except for legs with very small PDP in the order of the computational precision. Such errors are irrelevant in practise. Notice also that the approximation error does not grow with the length of the rotation.

Rotation 1

leg	duration	buffer	PDP	step size 1.0		step size 4.0	
	[m]	[m]	(sim)	PDP	error [%]	PDP	error [%]
1	60	25	0	0	0	0	0
2	100	10	0.0478	0.0470	1.67	0.0483	1.05
3	115	20	0.2144	0.2136	0.37	0.2213	3.22
4	40	0	0.0957	0.0949	0.84	0.0985	2.93
5	45	10	0.5521	0.5514	0.13	0.5608	1.58
6	75	20	0.3365	0.3367	0.06	0.3486	3.60
7	90	15	0.1959	0.1955	0.20	0.2035	3.88
8	70	-	0.1428	0.1432	0.28	0.1489	4.27

Rotation 2

leg	duration	buffer	PDP	step size 1.0		step size 4.0	
	[m]	[m]	(sim)	PDP	error [%]	PDP	error [%]
1	95	0	0	0	0	0	0
2	95	0	0.3177	0.3176	0.03	0.3267	2.83
3	65	0	0.4596	0.4591	0.11	0.4711	2.50
4	60	-	0.5746	0.5743	0.05	0.5873	2.21

Rotation 3

leg	duration	buffer	PDP	step size 1.0		step size 4.0	
	[m]	[m]	(sim)	PDP	error [%]	PDP	error [%]
1	95	55	0	0	0	0	0
2	100	35	0.0110	0.0109	0.91	0.0110	0
3	95	0	0.0195	0.0194	0.51	0.0195	1.53
4	85	35	0.4402	0.4395	0.16	0.4402	2.20
5	120	-	0.0600	0.0589	1.84	0.0600	2.17

Table 3.5: Probability of delay propagation for individual legs in rotations.

For a step size of four minutes, the results for individual legs become less reliable.

3.6.4 Conclusions

Presented computational study shows that discretisation with step sizes of 0.1 and 0.5 minutes are too slow for practical usage. Discretisation step of three minutes and more does not bring significant speed up of the method while the estimate of the probability of delay propagation is less accurate. Discretisation step size of one and two minutes provide fast computation of distributions of propagated delay and very high accuracy. In the rest of this work, we use discretisation step size of one minute as a standard setting for all computations.

3.7 Summary

In this chapter we introduced the model of aircraft operations and we showed how to compute propagated delay within this model. The implementation is based on the discretisation method, introduced in Chapter 1. Besides the computation of the propagated delay along the rotations, we also showed how to consider non-rotational delay propagation due to changing crew members, transferring passengers, or cargo.

This algorithm allows optimisation of various robustness measures which are based on knowledge of propagated or arrival delay of legs.

Developed method is suitable for application in column generation frameworks for solving real world large-scale instances. We demonstrated it on instances of the tail assignment problem. Discretisation method is applied in the path generation and provides fast and accurate computation of probability of delay propagation along rotations.

Chapter 4

The Robust Tail Assignment Problem

In Section 4.1, we shortly introduce the tail assignment problem and thereof derived the robust tail assignment problem.

We propose two mathematical models for the robust tail assignment problem. In Section 4.2, we show stochastic programming model RoTA which minimises rotational probability of delay propagation. In Section 4.3, we introduce two-stage stochastic programming model NoRoTA which minimises probability of delay propagation also with consideration of non-rotational delay propagation.

We also show how to solve both formulations by an extension of standard deterministic column generation algorithm where the central role plays discretisation method discussed in previous chapters.

4.1 Problem Definition

The tail assignment problem, as we understand it, first introduced Grönkvist (2005) [61]. The goal is to construct rotations for a set of aircraft in order to cover a set of flights. The rotations have to fulfil all maintenance rules and operational restrictions. Mathematical formulation of the tail assignment problem is alike to better known maintenance routing. The main difference between these problems is that the maintenance routing constructs the set of

virtual rotations for the aircraft set while the tail assignment creates "personalised" rotations for each concrete aircraft. This difference is also reflected in the name of the problem since individual aircraft are identified by the tail number.

The maintenance routing problem is solved a couple months before the day of operation. At that time exact aircraft positions of individual aircraft and their maintenance requirements are unknown. Therefore, the constructed routings need to ensure that a certain part of the fleet can be maintained each night and that the schedule is feasible.

The tail assignment is solved on a daily basis for following few days in order to adjust previously constructed rotations to current maintenance requirements. Information about current position of each individual aircraft allow to cover a wider set of restrictions and to satisfy all individual aircraft requirements. Some typical conditions reflected in the tail assignment problem formulation are: airport curfew, short to long-term maintenance requirement, or maintenance capacity of the airport. Although operational restrictions may seem fleet specific and thus equal for every aircraft, there are many aircraft dependent restrictions as well.

We formulate and solve the tail assignment for a single fleet, but it is possible to solve the problem for more fleets at the same time. This can be convenient if there are feasibility problems or if the fleets have easily interchangeable aircraft. Grönkvist (2005) [61] gives full details about the tail assignment together with suitable models and algorithms for its solving.

Nature of this problem does not give us a clear objective to optimise. Especially, there are no monetary costs to be minimised. In fact, aircraft of the same type have the same fuel consumption the set of legs to be flown is fixed and the choice of routings cannot influence it. Also crew pairings are fixed and thus crew costs cannot be changed either.

Clarke et al. (1997) [33] mention minimisation of so called through value, which is a bonus for preserving multi-leg passenger itineraries on the same aircraft. They also pose requirement for equal aircraft utilisation. Up to date literature, however, mostly focuses on maximisation of robustness which we also consider to be the key optimisation objective.

Meaning of robustness in context of the tail assignment differs from author to author but the central idea is the same. The goal is to decrease effects of delays on the aircraft, passengers, and whole airline network.

We refer reader to Section 2.8 for overview of methods and objective functions for improvement of robustness in the airline planning and in particular in the tail assignment and maintenance routing.

In order to minimise impact of delays various KPIs and approaches were introduced. Their goal is to estimate delay propagation effects throughout the network. From the mathematical point of view, stochastic programming provides the most satisfying approach to modelling the robustness of the schedules. The uncertainty in this case reflects aircraft delays.

In this chapter, we propose a stochastic programming models for solving the tail assignment problem where we use total probability of delay propagation, in short PDP, of the schedule as a measure of robustness. We show that these models are effectively solvable by the column generation framework enhanced by the discretisation method for computation of propagated delay introduced in Chapter 3.

4.2 Stochastic Programming Model RoTA

We formulate the robust tail assignment problem as a set partitioning problem with base constraints, see again Grönkvist (2005) [61].

$$\textbf{RoTA}: \quad \min \sum_{k \in K} \sum_{r \in R_k} d_r x_r^k \tag{4.1}$$

$$\sum_{k \in K} \sum_{r: l \in r, r \in R_k} x_r^k = 1 \qquad \forall l \in L \tag{4.2}$$

$$\sum_{k \in K} \sum_{r \in R_k} a_{br} x_r^k \leq s_b \qquad \forall b \in B \tag{4.3}$$

$$\sum_{r \in R_k} x_r^k = 1 \qquad \forall k \in K \tag{4.4}$$

$$x_r^k \in \{0, 1\} \qquad \forall k \in K, \forall r \in R_k. \tag{4.5}$$

Here, K denotes set of all aircraft, R_k is the set of feasible rotations for aircraft k, L is the set of all legs, and B is the set of all constraints. Fulfilment of individual aircraft constraints is guaranteed by construction of set R_k where rotations that are infeasible for aircraft k are not included in this set. Constraint (4.2) guarantees that every leg is covered by exactly one rotation. Constraint (4.3) ensures that the base constraints are satisfied.

These constraints can model, for instance, maintenance capacities at airports (bases). In such case constant s_b specifies maximum number of aircraft that may be maintained at particular airport at given night while a_{br} equals one if the rotation r ends with the maintenance check at that airport.

Difference between standard tail assignment model and RoTA lies in the objective function (4.1) which is stochastic. The objective maximises robustness in the sense of minimising the probability of delay propagation. The cost of rotation r is defined as

$$d_r = \sum_{i \in r} Pr[PD_i^r > 0] \tag{4.6}$$

where PD_i^r is random variable of propagated delay to leg i in rotation r.

4.2.1 Column Generation Approach

We solve RoTA model (4.1)-(4.5) by using a column generation algorithm. Although this model has stochastic objective function, it may be solved as a deterministic algorithm.

Here, we do not give technical details about solving the column generation models in general and we rather focus on problem specific enhancement of the algorithm. We show how to improve existing column generation framework for solving the tail assignment in order to incorporate stochastic objective function (4.6). Specifically, this improvement requires formulation of the pricing problem which can be implemented easily by the discretisation method.

The idea of the column generation is to solve the linear programming relaxation of the problem, the so-called master problem, in an iterative way. This algorithm for the robust tail assignment is summarised by Algorithm 3 in pseudocode.

The full formulation of the problem has exponentially many columns, therefore only a subset of columns is considered in each iteration. So constructed problem is called the restricted master problem (Step 4). In each iteration, we identify columns that may improve current solution of the restricted master problem. These columns are identified by solving the pricing problem where we search for columns with negative reduced cost (Step 5). The set

Algorithm 3: Column generation algorithm for RoTA.

Input : Set of aircraft K, set of legs L, base constraints B, rotation feasibility oracle, and random variables B_i^k, G_i^k, $i \in L$.

Output: Feasible schedule.

1 $newCols := \emptyset$;
2 generate initial RMP;
3 **repeat**
4 $\quad (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\nu}) :=$ solve the RMP;
5 $\quad newCols :=$ `solvePricingProblemForRoTA(`$\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\nu}$`)`;
6 $\quad$ add $newCols$ to the RMP;
7 **until** $newCols = \emptyset$;
8 heuristically find IP solution of the RMP;
9 **return** IP solution;

of such columns (denoted by $newCols$) is added to the restricted master problem (Step 6); then the problem has to be resolved.

This process is repeated until no improving column exists. Although we do not include all columns to the restricted master problem, we know that the master problem is solved to optimality since there is no column which can improve current solution of the problem. An integer solution is constructed by a rounding heuristic in a second phase (Step 8).

For more details about the column generation, please review publications Borndörfer et al. (2006) [25], Barnhart et al. (1998) [13], or Desaulniers, Desrosiers & Solomon (2005) [42].

4.2.2 Pricing Problem for RoTA

As we just mentioned, the goal of the pricing problem is to find columns (which correspond to feasible aircraft rotations) with negative reduced cost. Negativity of reduced costs means that the column can improve current solution of the restricted master problem. We show how to solve the pricing problem for a single aircraft however, the same problem needs to be solved for each aircraft.

We associate dual variables π_l, $l \in L$, with the leg covering Constraints (4.2), μ_b, $b \in B$, with the base Constraints (4.3), and ν_k, $k \in K$, with the aircraft Constraints (4.5). The pricing problem for aircraft k can be then stated as

$$\min_{r \in R^k} \bar{d}_r = \min_{r \in R^k} \left(d_r - \sum_{l \in r} \pi_l + \sum_{b \in B} a_{br} \mu_b - \nu_k \right).$$

Assuming that the consumption of base resources on a rotation for aircraft k can be expressed in terms of base resource consumptions a_{bl} on legs l, the pricing problem becomes

$$\min_{r \in R^k} \bar{d}_r = \min_{r \in R^k} \left(d_r - \sum_{l \in r} \pi_l + \sum_{b \in B} \sum_{l \in r} \mu_b a_{bl} \right) - \nu_k. \tag{4.7}$$

We also know that d_r can be expressed as the sum of robustness measure on individual legs which, in our case, is the sum of probability of delay propagation to legs in the rotation

$$d_r = \sum_{l \in r} Pr[PD_l^r > 0].$$

Altogether, the pricing problem can be rewritten as

$$\min_{r \in R^k} \bar{d}_r = \min_{r \in R^k} \sum_{l \in r} \left(Pr[PD_l^r > 0] - \pi_l + \sum_{b \in B} \mu_b a_{bl} \right) - \nu_k \tag{4.8}$$

which can be seen as the problem of finding the shortest path in a connection graph. ν_k is a problem specific constant that is not influencing the solution process.

Definition 4.2.1 (Connection Graph)**.** *The connection graph is an oriented graph with one vertex for each leg and two artificial vertices, source vertex s and target vertex t.*

An arc $a = (u, v)$, $u, v \in L$, exists if and only if leg v can succeed leg u in some rotation. That is, if leg u arrives to the airport before leg v departs from this airport.

Vertex s is connected to legs which may be considered as first legs of the day, that is, they originate from the airport where the aircraft stays over night. All legs that may be the last operated legs of the day are connected to t.

We define arc weights for arcs not including s and t as

$$w_{uv} = Pr[PD_v^r > 0] - \pi_v + \sum_{b \in B} \mu_b a_{bv} \tag{4.9}$$

where r is path from s to v. Weights of all arcs from the source as

$$w_{sv} = -\pi_v + \sum_{b \in B} \mu_b a_{bv} \tag{4.10}$$

since we know that there is no propagated delay to the first leg of the rotation, see Formula (3.1). All arcs to the sink have weight

$$w_{ut} = 0. \tag{4.11}$$

Please note that so defined weights w_{uv} depend on currently constructed su-path due to PD_v^r.

Finding solution of the pricing problem (4.8) therefore equals to finding the shortest path from s to t in the defined connection graph.

Implementation of The Pricing Problem

In Formula 4.9, resource consumptions a_{bv}, dual variables μ_b and π_v are the part of the problem's input but random variable PD_v^r for $v \in L$ as well as $Pr[PD_v^r > 0]$, are unknown since they depend on currently explored path.

It means that we cannot construct complete connection graph with fixed arc weights. Thus we compute distribution of PD_v^r and $Pr[PD_v^r > 0]$ "on the fly" during the path search and only for the path being currently generated. First, let us show how to compute arc weights w_{uv} and then we state whole path search algorithm.

Function `computeArcWeight` (see Algorithm 4) summarises computation of w_{uv}. According to (4.10) and (4.31), all arcs from from the source and to the sink do not have contribution from propagated delay and their weights are easy to compute, see Step 3 and 6.

Let p_{su} be currently constructed su-path which we are about to extend along arc uv and propagated delay $PD_u^{p_{su}}$ to leg u was computed beforehand. In order to compute w_{uv}, we have to compute $PD_v^{p_{uv}}$ first. As explained in Section 3.4.1, propagated delay PD_v^r for leg $v \in L$ with predecessor vertex

Algorithm 4: Algorithm for computation of the arcs weight in RoTA pricing problem.

Input : Discretised random variables B_i^k, $G_i^k, i \in L$, su-path p_{su} which is subpath of rotation r, distribution of propagated delay $PD_u^{p_{su}^k}$ for vertices along path r, and vertex v such that there is arc (u, v) in the connection graph.
Base constraints B, dual variables $\boldsymbol{\pi}$ and $\boldsymbol{\mu}$.

Output: Weight of arc (u, v) extending path p_{su} and distribution of propagated delay $PD_v^{p_{sv}^k}$ to vertex v along path $p_{sv} = p_{su} + v$ which is subpath of rotation r.

```
1  computeArcWeight(u, v, PD_u^{p_su^k},π,μ) begin
2      if u = s then
3          w_uv := −π_v + Σ_{b∈B} μ_b a_bv;
4          return (w_uv, 0);
5      else if v = t then
6          return (0, 0);
7      else
8          PD_v^{p_sv^k} := max{PD_u^{p_su^k} + G_u^k + B_u^k − b_uv, 0};
9          w_uv := Pr[PD_v^{p_su^k} > 0] − π_v + Σ_{b∈B} μ_b a_bv;
10         return (w_uv, PD_v^{p_sv^k});
```

u in rotation r is computed from primary gate delay G_v and block time deviation B_v and from propagated delay PD_u^r to vertex u, see Formula (3.2).

Computation of $PD_v^{p_{sv}^k}$ is done in Step 8 of Algorithm 4. Then in Step 9, we compute w_{uv} according to the definition of the arc weight (4.9). All operations over random variables are handled by the discretisation method which is explained in Section 3.4.2.

Note that computed $PD_v^{p_{sv}^k}$ has to be stored because it is required for computation of arc weights of further extensions of the path. Hence, function `computeArcWeight` returns $PD_v^{p_{sv}^k}$ together with w_{uv}.

Now, we know how to compute arc weights, let us show how to find the shortest path in the graph. Not every path can be considered during the path search due to rotation feasibility requirement. This itself significantly increases difficulty of the problem since this leads to solving the resource constrained shortest path problem which is known to be NP-hard.

Typically, the problem is solved by dynamic programming approach, see Desaulniers et al. (1997) [41], or by depth-first-search, see Klabjan et al. (2001) [66]. See overview of the resource constrained shortest path problem formulations and solution techniques in Desaulniers, Desrosiers & Solomon (2005) [42].

We solve the pricing problem by the depth-first search algorithm. Clearly, there is no efficient way to find the most negative path since it requires exploration of all paths, but we can find many negative paths fast. In each iteration of the column generation we identify more negative paths and add them to RMP. One can apply full depth-first search in the final iterations of the column generation if optimal solution of the RMP is required.

Algorithm 5 shows pseudocode of implementation of the pricing algorithm `solvePricingProblemForRoTA` which solves the pricing problem for every aircraft by calling the function `pathSearchRoTA` (Step 5).

If sink vertex t is reached and the cost of the current path is negative, we add the current path to the set of negative paths (Step 10). If we reach target count of negative paths we terminate search for this aircraft (Step 12).

In case the current path is incomplete, we try all possible extensions of the path (Step 14). If the extension is feasible, we compute distribution of propagated delay $PD_v^{r^k}$ and weight of the arc by method `computeArcWeight` specified by Algorithm 4. We recursively call `pathSearchRoTA` to the extended path with updated path cost (Step 18).

For simplicity, we handle feasibility rules of aircraft rotations in the algorithm by means of a feasibility oracle (Step 16). Note that in Step 14, we can also heuristically pick only the most promising extensions of the path instead of all possibilities.

Just to summarise, we recall that function `solvePricingProblemForRoTA` in Algorithm 5 is applied in Algorithm 3 in order to identify new rotations (columns of the master problem) with negative reduced costs for each aircraft, see Step 5 of Algorithm 3. These rotations are then included to the RMP in the next step of the column generation iteration.

Algorithm 5: Pricing Algorithm for RoTA problem `solvePricingProblemForRoTA()`

Input : Set of legs L, set of aircraft K, base constraints B, rotation feasibility oracle, dual variables $\boldsymbol{\pi}$, $\boldsymbol{\mu}$, and $\boldsymbol{\nu}$, and random variables B_i^k, G_i^k, $i \in L$.

Output: Set of feasible paths (rotations) with negative reduced cost.

```
1  solvePricingProblemForRoTA(π, μ, ν) begin
2      negativePaths := ∅;
3      for every aircraft a do
4          construct connection graph for aircraft a;
5          pathSearchRoTA(∅,s,0,0,π, μ, ν) ;
6      return negativePaths;
7  pathSearchRoTA(p_su,u, cost,PD_u^{p_su},π, μ, ν) begin
8      if u = t then
9          if cost < ν_k then
10             add p_su to negativePaths;
11             if found enough negative paths then
12                 terminate path search;
13         return;
14     for arcs (u, v) do
15         p_sv = p_su + v;
16         if p_sv is feasible then
17             (w_uv, PD_v^{p_sv}) :=computeArcWeight (u,v,PD_u^{p_su},π, μ);
18             pathSearchRoTA (p_sv,v,cost + w_uv,PD_v^{p_sv},π, μ, ν);
```

4.3 Two-stage Stochastic Programming Model NoRoTA

In the previous section we showed how to solve RoTA model for the robust tail assignment problem. Now, we look at an extension of this problem where also non-rotational delay propagation is considered.

Section 3.5 demonstrates that computation of the propagated delay for a single rotation with non-rotational delay propagation is not straightforward. Difficulties arise from dependency of propagated delay of one rotation on another rotations. It means that the change in the rotation of one aircraft may change the propagated delay to another aircraft, and thus cost of another aircraft's rotation while the rotation itself stays unchanged. Consequences of possible interactions between the rotations are explained and demonstrated on the examples in Section 3.5.3 and Section 3.5.4.

This has strong implication to the column generation approach. The objective function of the problem, which is a function of propagated delay, becomes a non-linear function and not directly decomposable to individual columns.

We therefore propose a two-stage stochastic programming model. In the first stage, we compute the expected cost of each rotation when considered independently of others. This is the same situation than we deal with in RoTA model and thus the formulation is so far the same as for RoTA model, see (4.1)-(4.5). In the second stage, we add probability of delay propagation due to non-rotational delay propagation only.

Yen & Birge (2006) [109] tackle alike problem by two-stage stochastic programming model with nonlinear recourse for minimisation of expected delay costs of the crew pairings. They demonstrate complexity of minimisation of such objective. However, the way we model delay propagation in the objective is alike to Yen & Birge (2006) [109], the solution methods are very different. In comparison to Yen & Birge (2006) [109], our recourse problem has different structure in order to match the structure of our stochastic model.

Equations (4.12)-(4.16) summarise the model and the recourse model is defined by (4.18)-(4.26).

$$\textbf{NoRoTA}: \quad \min \sum_{k\in K}\sum_{r\in R_k} d_r x_r^k + \mathbb{Q}(\mathbf{x}) \tag{4.12}$$

$$\sum_{k\in K}\sum_{r:l\in r, r\in R_k} x_r^k = 1 \qquad \forall l \in L \tag{4.13}$$

$$\sum_{k\in K}\sum_{r\in R_k} a_{br} x_r^k \leq s_b \qquad \forall b \in B \tag{4.14}$$

$$\sum_{r\in R_k} x_r^k = 1 \qquad \forall k \in K \tag{4.15}$$

$$x_r^k \in \{0,1\} \qquad \forall k \in K, \forall r \in R_k. \tag{4.16}$$

Please notice that this model is identical to the model (4.1)-(4.5) except the objective function. In addition to probability of rotational propagated delay d_r, the expectation of the recourse problem $\mathbb{Q}(\mathbf{x})$ through all disruption scenarios $\omega \in \Omega$ is minimised. That is

$$\mathbb{Q}(\mathbf{x}) = \mathbb{E}\left(Q\left(\mathbf{x},\omega\right)\right) = \int_{\Omega} Q\left(\mathbf{x},\omega\right) Pr[\omega] d\omega. \tag{4.17}$$

We state the recourse problem as follows

$$Q\left(\mathbf{x},\omega\right) = \min \sum_{j\in L} \left(ppd_j - ppd'_j\right) \tag{4.18}$$

$$ad'_j = pd'_j + B_j^{\omega} + G_j^{\omega} \qquad \forall j \in L \tag{4.19}$$

$$pd'_j = \sum_{k\in K}\sum_{r\in R_k} \left(ad'_{\mathrm{pred}(r,j)} - b_{\mathrm{pred}(r,j),j}\right) x_r^k \qquad \forall j \in L \tag{4.20}$$

$$pd'_j \geq 0 \qquad \forall j \in L \tag{4.21}$$

$$Mppd'_j \geq pd'_j \qquad \forall j \in L \tag{4.22}$$

$$ad_j \geq pd_j + B_j^{\omega} + G_j^{\omega} \qquad \forall j \in L \tag{4.23}$$

$$pd_j \geq \sum_{k\in K}\sum_{\forall r\in R_k} \left(ad_{\mathrm{pred}(r,j)} - b_{\mathrm{pred}(r,j),j}\right) x_r^k \qquad \forall j \in L \tag{4.24}$$

$$pd_j \geq ad_l - b_{lj}^c \qquad \forall j \in L, \forall l \in M_j \tag{4.25}$$

$$pd_j \geq 0 \qquad \forall j \in L \tag{4.26}$$

$$Mppd_j \geq pd_j \qquad \forall j \in L \tag{4.27}$$

$$ppd_j, ppd'_j \in \{0,1\} \qquad \forall j \in L \tag{4.28}$$

where $\mathbf{x} = \left(x_1^1, x_2^1, \ldots, x_1^2, x_2^2, \ldots, x_1^k, x_2^k, \ldots\right)$.

Recourse problem corresponds to the model of airline operations introduced in Section 3.3 and formally is described in Section 3.5.1. Every disruption scenario $\omega \in \Omega$ comprises of realisations of all random variables $B_i, i \in L$, of block time duration, and $G_i, i \in L$, of primary gate delay. We denote these realisations of random variables by ω by G_i^ω and B_i^ω. In order to remind definition and explanation of these random variables please see Chapter 3.

Just to remind, $b_{i,j}$ denotes ground buffer between legs i and j, and $b_{i,j}^c$ denotes ground buffer of the crew member changing between legs i and j. As we previously mentioned, G_i^ω, B_i^ω, $b_{i,j}^c$, and $b_{i,j}$ are constant values. Function $\mathrm{pred}(r, j)$ returns previous leg to leg j in rotation r, see formal definition (3.12) and M is some large constant. Variables x_r^k are the first stage decision variables of aircraft rotations. Let us denote $\mathcal{R}$ set of rotations chosen in the first stage corresponding to $\mathbf{x}$.

Variables ad_j and pd_j are second stage decision variables indicating arrival delay and propagated delay of leg j with consideration of non-rotational delay propagation, respectively. When coming back to the stochastic model, these variables can be also seen as realisations of random variables ${}^{\mathcal{R}}AD_j^{\mathrm{rot}(\mathcal{R},j)}$ and ${}^{\mathcal{R}}PD_j^{\mathrm{rot}(\mathcal{R},j)}$ from the stochastic model. Decision variables ad'_j and pd'_j denote the same but without consideration of non-rotational delay propagation and thus correspond to $AD_j^{\mathrm{rot}(\mathcal{R},j)}$ and $PD_j^{\mathrm{rot}(\mathcal{R},j)}$, respectively.

Binary decision variables ppd_j and ppd'_j model the probability that propagated delay pd_j and pd'_j to leg j is positive. The objective function of the recourse problem (4.18) computes only the contribution of non-rotational delay propagation. Therefore, we subtract the occurrences of delay propagation without consideration of non-rotational delay propagation from the occurrences of delay propagation under consideration of non-rotational delay propagation.

Inequalities (4.19)-(4.21) ensure correct computation of pd'_j, and they are equivalent to formulae describing formal computation of the propagated delay from Section 3.4. Inequality (4.19) corresponds to (3.3) and computes the arrival delay of the leg, which is the sum of the propagated delay, primary gate delay, and deviation of the block time. Inequalities (4.20) and (4.21) correspond to (3.4) and compute the propagated delay to the leg which depends on the arrival delay of the previous leg. Previous leg is determined by the choice of the rotation covering the leg. This choice is given by the

decision variables x_r^k from the first stage of the problem. Inequality (4.21) ensures non-negativity of propagated delay.

Conditions (4.22), (4.28) ensure that if pd_j, pd'_j is positive , then ppd_j, ppd'_j equals one otherwise it equals zero.

Inequalities (4.23)-(4.26) ensure correct computation of pd_j, that is, the propagated delay to leg j with consideration of non-rotational delay propagation according to Section 3.5. These inequalities are equivalent to Formulae (3.14)-(3.18) that formally describe computation of such propagated delay.

In addition to Inequalities (4.23), (4.24), and (4.26) which are similar to (4.19), (4.20), and (4.21), respectively; inequalities (4.25) address inclusion of non-rotational propagated delay defined by Formula (3.18). The model ensures that the propagated delay is maximum of propagated delay, (Inequalities (4.24)), and of propagated delay due to incoming crew to leg i (Inequalities (4.25)). We denote M_i set of all legs from which leg i awaits crew, see (3.13). Inequalities (4.25) ensure delay propagation from these legs. If there is an incoming crew also from the previous leg of the chosen rotation, the constraint for delay propagation from these leg is duplicate. Notice that this duplicity is irrelevant in the mixed-integer programming model while in Formulae (3.14)-(3.18) has to be omitted.

As we mentioned in Section 3.3, minimum change time may be greater than minimum ground time. If this is the case for legs i and j covered by some crew member, the aircraft has to cover legs i and j in the sequence as well. Otherwise, the crew does not have enough time to change the aircraft. Such crew connections are called short connections. During the solving the robust tail assignment, crew pairings are already generated and cannot be changed. Typically, optimisation models avoid violation of the minimum sit time by constraints for time critical connections ensuring "aircraft follows crew" policy.

We do not incorporate these constraints into our model explicitly since we do not consider such solutions to be infeasible. Unsatisfactory change time of the crew often causes delayed departure of the next leg. Since we minimise total probability of delay propagation, unsatisfactory change time is automatically penalised during the optimisation in the objective and such solution can be optimal only in very special situations.

Solution approach

The recourse problem makes NoRoTA difficult to solve. Even restriction of the scenario space of the recourse problem to rather small number of scenarios keeps the model intractable for practical application. We propose a method for approximate solving the NoRoTA model by a deterministic column generation framework which employs the discretisation method.

We do not define pricing problem for whole two stage stochastic model, i.e. the first and the second stage decision variables, but only for the first stage variables. That means we solve the pricing problem similar than for RoTA but with contribution of recourse problem to objective. After solving the restricted master problem we update the second stage solution in order to align with change of first stage variables. At this point of computation we have correct value of the objective and we proceed to the next iteration.

First let us show implications of two stage stochastic problem to the pricing problem based on which we state whole column generation algorithm later.

4.3.1 Pricing Problem for NoRoTA

Models RoTA and NoRoTA differ only in the objective function where we add the recourse problem on top of RoTA. We therefore modify column generation for RoTA model in a way that we include contribution of recourse problem to the objective. In order to incorporate the recourse problem within column generation we have to decompose it to individual columns and arc weights in the connection graph. Then, it is easy to see that this problem also reduces to the restricted shortest path problem and we solve it by the depth-first-search.

So, let us have a look on decomposition of objective of the recourse problem to arcs in the connection graph. Notice that the recourse problem is due to Inequalities (4.20) and (4.24) nonlinear, but for fixed first stage decisions it reduces to an easy to solve mixed-integer programming model. Fortunately, when looking for columns with negative reduced costs by solving the pricing problem, the fixed solution of RMP from the previous iteration is considered.

Let $\mathbf{x}^*$ be an integer solution of the problem and $\mathcal{R}^*$ set of rotations corresponding to $\mathbf{x}^*$. Furthermore, let us denote $\mathbb{Q}_r(\mathbf{x}^*)$ the value of recourse problem assigned to rotation r for fixed $\mathbf{x}^*$.

Please notice that the pricing problem is solved based on linear programming solution not integer one. How to tackle this difference is explained later in Section 4.3.2.

Since $\mathbb{Q}(\mathbf{x}^*)$ is the sum of contribution of propagated delay for each leg, we can easily decompose it for every rotation as simple sum for all legs in the rotation.

$$\begin{aligned}\mathbb{Q}_r(\mathbf{x}^*) =& \mathbb{E}\left(\sum_{j\in r} ppd_j - \sum_{j\in r} ppd'_j\right)\\ =& \sum_{j\in r} Pr[{}^{\mathcal{R}^*}PD_j^r > 0] - \sum_{j\in r} Pr[PD_j^r > 0]\end{aligned}$$

under constraints (4.19)-(4.28) where PD_j^r and ${}^{\mathcal{R}^*}PD_j^r$ are random variables of propagated delay to leg j in rotation r. PD_j^r considers no non-rotational delay propagation and ${}^{\mathcal{R}^*}PD_j^r$ considers non-rotational delay propagation under consideration of rotations $\mathcal{R}^*$.

Decomposition of the recourse problem to individual rotations gives us the following pricing problem

$$\begin{aligned}\min_{r\in R^k} \bar{d}_r(\mathbf{x}^*) &= \min_{r\in R^k}\left(d_r + \mathbb{Q}_r(\mathbf{x}^*) - \sum_{l\in r}\pi_l + \sum_{b\in B}\sum_{l\in r}\mu_b a_{bl}\right) - \nu_k\\ &= \min_{r\in R^k}\sum_{l\in r}\Bigg(Pr[{}^{\mathcal{R}^*}PD_j^r > 0] - Pr[PD_j^r > 0]\\ &\quad + Pr[PD_j^r > 0] - \pi_l + \sum_{b\in B}\mu_b a_{bl}\Bigg) - \nu_k\\ &= \min_{r\in R^k}\sum_{l\in r}\left(Pr[{}^{\mathcal{R}^*}PD_l^r > 0] - \pi_l + \sum_{b\in B}\mu_b a_{bl}\right) - \nu_k. \qquad (4.29)\end{aligned}$$

It means that in order to solve the pricing problem of NoRoTA we need to compute propagated delay with consideration of non-rotational delay propagation for given covering of legs along newly generated rotation in the connection graph (see Definition 4.2.1).

Similarly than for RoTA model, arc weights depend on currently constructed path and thus need to be computed along path construction. We also use discretisation method for representation of random variables and operations over them.

Algorithm 6: Algorithm for computation of the arcs weight in NoRoTA pricing problem.

Input : Discretised random variables B_i^k, G_i^k, $i \in L$, su-path p_{su} which is subpath of rotation r, distribution of propagated delay ${}^{\mathcal{R}^*}PD_u^{r^k}$ along path r, and vertex v such that there is arc (u, v) in the connection graph. Base constraints B and dual variables $\boldsymbol{\pi}$ and $\boldsymbol{\mu}$.

Output: Weight of arc (u, v) extending path p_{su} and distribution of propagated ${}^{\mathcal{R}^*}PD_v^{p_{sv}{}^k}$ delay to vertex v along path $p_{sv} = p_{sv} + v$ which is subpath of rotation r.

`computeArcWeightWithCrew(`u, v, ${}^{\mathcal{R}^*}PD_u^{p_{su}^k}$, $\boldsymbol{\pi}, \boldsymbol{\mu}$,$\{{}^{\mathcal{R}^*}AD_l^{*^k}\}$`)` **begin**

- **if** $v = t$ **then**
  - **return** $(0, 0)$;
- **else**
  - **if** $u = s$ **then**
    - ${}^{\mathcal{R}^*}APD_v^{p_{sv}^k} := 0$;
  - **else**
    - ${}^{\mathcal{R}^*}APD_v^{p_{sv}^k} := \max\left\{{}^{\mathcal{R}^*}PD_u^{p_{su}^k} + G_u^k + B_u^k + b_{uv}, 0\right\}$;
  - ${}^{\mathcal{R}^*}CPD_v^{p_{sv}^k} := \max_{l \in M_v}\left\{\max\left\{{}^{\mathcal{R}^*}AD_l^{*^k} - b_{lv}^c\right\}\right\}$;
  - ${}^{\mathcal{R}^*}PD_v^{p_{sv}^k} := \max\left\{{}^{\mathcal{R}^*}APD_v^{p_{sv}^k}, {}^{\mathcal{R}^*}CPD_v^{p_{sv}^k}\right\}$;
  - $w_{uv} := {}^{\mathcal{R}^*}Pr[PD_v^{r^k} > 0] - \pi_v + \sum_{b \in B} \mu_b a_{bv}$;
  - **return** $\left(w_{uv}, {}^{\mathcal{R}^*}PD_v^{p_{sv}^k}\right)$;

We therefore solve the shortest path problem in the connection graph with arc weights

$$w_{uv} = Pr[{}^{\mathcal{R}^*}PD_v^r > 0] - \pi_v + \sum_{b \in B} \mu_b a_{bv} \tag{4.30}$$

for arcs not including vertex t and where r is some path from s to v. All arcs to the sink have weight

$$w_{ut} = 0. \tag{4.31}$$

Algorithm 7: Pricing algorithm for NoRoTA problem `solvePricingProblemForNoRoTA()`

Input : Set of legs L, set of aircraft K, base constraints B, rotation feasibility oracle, dual variables $\boldsymbol{\pi}$, $\boldsymbol{\mu}$, and $\boldsymbol{\nu}$, random variables B_i, G_i, $i \in L$, and distributions of arrival delay ${}^{\mathcal{R}^*}AD_l^{*^k}$ for $l \in L$.

Output: Feasible paths (rotations) with negative reduced cost.

```
1  solvePricingProblemForNoRoTA(π, μ, ν, {R* AD*_l^k}) begin
2      negativePaths := ∅;
3      for every aircraft a do
4          construct connection network for aircraft a;
5          pathSearchNoRoTA(∅,s,0,0, π, μ, ν, {R* AD*_l^k} );
6      return negativePaths;
7  pathSearchNoRoTA(p_su,u, cost, R* PD_u^{p_su^k}, π, μ, ν, {R* AD*_l^k}) begin
8      if u = t then
9          if cost < ν_k then
10             add p_su to negativePaths;
11             if found enough negative paths then
12                 terminate path search;
13         return;
14     for arcs (u, v) do
15         p_sv = p_su + v;
16         if p_sv is feasible then
17             (w_uv, R* PD_v^{p_sv^k}) :=
               computeArcWeightWithCrew(u, v, R* PD_u^{p_su^k}, π, μ, {R* AD*_l^k});
18             pathSearchNoRoTA (p_sv,v,cost + w_uv, R* PD_v^{p_sv^k}, π, μ,
               ν, {R* AD*_l^k});
```

Notice that in comparison to connection graph for RoTA problem, there may be some propagated delay to the first leg of the rotation due to connecting crew.

Algorithm 6 shows computation of arcs weight in the connection graph for NoRoTA problem. It extends Algorithm 4 where in addition to rotational propagated delay, in Step 6 and 8, also non-rotational propagated delay due to changing crew, in Step 9, is computed. In Step 10 maximum of these delays is taken as total propagated delay. Steps 6-10 correspond to Equations (3.14)-(3.18).

Propagated delay of changing crew is computed from the arrival delay of crew's previous leg. Arrival delay ${}^{\mathcal{R}^*}AD_l^{*^k}$ of such leg l is given as an input of the pricing problem and we come to its computation later in this section. Similarly than in Algorithm 4, we return weight of the arc w_{uv} and distribution of propagated delay ${}^{\mathcal{R}^*}PD_v^{p_{sv}^k}$.

Algorithm for solving the pricing problem stays unchanged in comparison to RoTA model and it is summarised in Algorithm 7. It differs from Algorithm 5 only in the computation of arc weights in the connection graph and additional input parameter ${}^{\mathcal{R}^*}AD_i^{*^k}$, $i \in L$ needed for arc's weight computation.

4.3.2 Approximate Column Generation Approach

In order to conclude with complete column generation algorithm, let us explain how to compute ${}^{\mathcal{R}^*}AD_i^{*^k}$, $i \in L$ which we skipped in previous explanation. We also explain how we handle nonintegrality of current RMP solution.

Computation of ${}^{\mathcal{R}^*}AD_i^{*^k}$ is based on current solution $\mathbf{x}^*$ of RMP. As the solution of RMP changes with every iteration we should recompute ${}^{\mathcal{R}^*}AD_i^{*^k}$. As a consequence we should recompute prices of all columns that depend on ${}^{\mathcal{R}^*}AD_i^{*^k}$. This, actually, reflects connection of the first stage decision variables with recourse problem. We refer reader to Section 3.5.3 and Section 3.5.4 where possible interactions between the rotations are explained and demonstrated on examples.

Accuracy of prices of such columns is high as long as the solution of RMP (fixed first stage decisions for the recourse problem) matches to the previous RMP solution. Unfortunately, the optimal solution of the RMP changes with iterations significantly and the prices of the columns added to the RMP in early iterations become very inaccurate. Hence, we increase accuracy of the

Algorithm 8: Column generation algorithm for NoRoTA.

Input : Set of aircraft K, set of legs L, base constraints B, rotation feasibility oracle, and random variables B_i^k, G_i^k, $i \in L$.

Output: Feasible schedule.

1 $newCols := \emptyset$;
2 generate initial RMP;
3 **repeat**
4 $\quad (\mathbf{x}^*, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\nu}) :=$ solve the RMP;
5 $\quad (recalcCols, \left\{ {}^{\mathcal{R}^*}AD_l^{*^k} \right\}) :=$`updateProblem(`$\mathbf{x}^*, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\nu}$`)`;
6 $\quad newCols :=$ `solvePricingProblemForNoRoTA(`$\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\nu}, \{{}^{\mathcal{R}^*}AD_l^{*^k}\}$`)`;
7 $\quad$ add $newCols$ to the RMP;
8 $\quad$ update the RMP with $recalcCols$;
9 **until** $newCols = \emptyset$;
10 heuristically find IP solution of the RMP;
11 **return** IP solution;

the RMP by recalculating the column prices in order to match with current first stage decision. Accuracy of prices of these columns is high as long as solution of RMP (fixed first stage decisions for the recourse problem) matches to the previous RMP solution. Unfortunately, the optimal solution of the RMP changes with iterations significantly and the prices of the columns added to the RMP in early iterations become very inaccurate. Hence, we increase accuracy of the the RMP by recalculating the column prices in order to match with current first stage decision.

Our modified column generation approach summarises Algorithm 8 in pseudocode. In comparison to column generation algorithm for RoTA, stated in Algorithm 3, we add a new procedure `updateProblem` which updates the cost of columns that are already in the RMP as well as computes ${}^{\mathcal{R}^*}AD_i^{*^k}$, see Step 5 of the algorithm.

Algorithm 9: Algorithm for updating the cost of all rotations in the current solution `updateProblem()`.

Input : Set of legs L, random variables G_i^k, B_i^k, $i \in L$, rotations specified by $\mathbf{x}^*$ where x_r^* represents probability of choosing r.

Output: Total probability of propagated delay of rotations and distribution of arrival delay of $l \in L$ with respect to $\mathbf{x}^*$.

updateProblem$(\mathbf{x}^*, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\nu})$ **begin**
  sort L topologically as $l_1, l_2, \ldots, l_n$ by increasing scheduled arrival time;
  for $l := l_1$ **to** l_n **do**
    for $r \in \mathrm{rot}^*(\mathbf{x}^*, l)$ **do**
      $l_p := \mathrm{pred}(r, l)$;
      if $l_p = \emptyset$ **then**
        $APD_l^r := 0$;
      else
        $APD_l^{r^k} := \max\left\{AD_{l_p}^{r^k} - b_{l_p,l}, 0\right\}$;
      $CPD_l^{r^k} := \max_{q \in M_l \setminus \{l_p\}}\left\{\max\left\{AD_q^{*^k} - b_{q,l}^c, 0\right\}\right\}$;
      $PD_l^{r^k} := \max\left\{APD_l^{r^k}, CPD_l^{r^k}\right\}$;
      $AD_l^{r^k} := PD_l^{r^k} + G_l^k + B_l^k$;
    $AD_l^{*^k} := \bigoplus_{r \in \mathrm{rot}^*(\mathbf{x}^*, l)} \mathbf{x}_r^* AD_l^{r^k}$;
  for $r \in \mathcal{R}$ such that $x_r^* > 0$ **do**
    $\bar{d}_r(\mathbf{x}^*) := \sum_{i \in r}\left({}^{\mathcal{R}^*}Pr[PD_i^{r^k} > 0] - \pi_i + \sum_{b \in B} \mu_b a_{bi}\right) - \nu_k$;
  return $\bar{\mathbf{d}}(\mathbf{x}^*)$, $\left\{AD_l^k \mid l \in L\right\}$;

Column Updates

The pricing problem for integer $\mathbf{x}^*$ is equivalent to Formulae 3.14-3.18. therefore ${}^{\mathcal{R}^*}AD_l^{*^k}$ can be computed by Algorithm 2. We recall that we denote by $\mathcal{R}^*$ set of rotations corresponding to $\mathbf{x}^*$. Unfortunately, every iteration of the column generation algorithm provides us linear solution of the problem instead of integer solution.

Clearly, linear relaxation of the recourse problem does not give us direct opportunity to solve the problem by the discretisation method. Thus we always look at the recourse problem as a mixed-integer problem and fractionality of first stage decisions is considered as the probability of choosing the rotation into the solution. This point of view allows us to use again discretisation method for computation of ${}^{\mathcal{R}^*}AD_l^{*^k}$ by modified Algorithm 2. In parallel, we evaluate new prices for columns that are already part of RMP.

Algorithm 9 summarises this procedure. Similarly than in Algorithm 2 we compute propagated delay along legs in rotations in increasing order according to the scheduled arrival time. It guarantees that arrival delay of other legs, which may be required in Step 10, is already computed.

Steps 4-12 of Algorithm 9 correspond to steps 3-11 of Algorithm 2 and compute propagated and arrival delay to chosen leg. For detailed explanation we refer reader to Section 3.5 and in particular to description of Algorithm 2.

Since $\mathbf{x}$ is fractional, one leg may be covered by more rotations at the same time. Therefore we compute propagated delay along the leg for all these rotations at once (Step 4). In order to identify all legs covering the leg, we introduce function rot^* which is an extension of function rot that for set of rotations and leg $l \in L$ returns the subset of rotations which are covering leg l

$$\text{rot}^* : \ 2^R \times L \to 2^R.$$

When considering arrival delay of some leg which is not part of the rotation we are currently looking at, we take the mixture of arrival delay through all rotations covering this leg. We denote random variable for such arrival delay $AD_l^{*^k}$. For simplicity, we compute the mixture of arrival delays of the leg right after the computation of arrival delays for individual rotations (Step 13). This random variables are used for computation of non-rotational propagated in Step 10.

The algorithm outputs new values of the total propagated delay AD_l^{*k} for every $l \in L$ and updated prices (computed in Step 15) for all rotations in current optimal solution of RMP.

Chapter 5

Stochastic Model

In this chapter, we present a stochastic model of primary delays of airline operations. This model is used throughout this work for the optimisation and evaluation of the aircraft schedules and is derived based on a large set of historical data.

In Section 5.1, we discuss some stochastic models of aircraft operations presented in the literature and we outline a structure of our stochastic model which is alike to presented model of airline operations from Section 3.3 .

Then, in Section 5.2, we document historical data set which we apply as input data for our analysis. We describe the process of separating primary delays from propagated delays.

In Section 5.3, we explain statistical methods and techniques we use during the data analysis and for estimation of probability distributions of random variables in our stochastic model.

In Section 5.4 and Section 5.5, we provide details about chosen distributions and their parameters. We show that

- the probability of occurrence of the primary gate delay depends on the departure airport and on the departure time of the leg and it follows Bernoulli distribution
- the length of the primary gate delay is independent of all parameters and follows Log-normal distribution with tail following Power-law distribution

- deviation of the block phase follows Log-logistic distribution with parameters depending on the scheduled block duration.

Section 5.6 is dedicated to discussion about minimum ground and minimum sit time used in our model.

In Section 5.7, we list assumptions under which is our model valid and we justify their choice. We conclude the chapter with justification of the model by experiments comparing historically observed propagated delay with the propagated delay forecast by our model in Section 5.8.

5.1 Model Overview

5.1.1 Stochastic Models in Airline Optimisation

Most papers concerning robust optimisation in the field of airline scheduling propose some stochastic model of disruptions and delay propagation. Such models are used either directly in the mathematical optimisation model, or for the evaluation of robust solutions by simulation. These models vary in structure, distributions and their parameters, in the factors on which the parameters are dependent, and in the methods used for estimation of suitable distributions and their parameters.

Many models forecast delay of legs by a single random variable. Arikan et al. (2010) [8] model block duration of legs by truncated Log-Laplace distribution. The parameters of the distribution are obtained by regression that is parametrised by route, origin and destination of the leg, carrier, month, day of week, departure and arrival time, aircraft age, average number of passengers on board, and congestion of the departure and arrival airport. Sohoni, Lee & Klabjan (2008) [103] model the block duration using truncated Log-normal distribution without any other details about the parameters of the distributions.

AhmadBeygi, Cohn & Lapp (2008) [4] present a different perspective on delay generation. They model primary delays at the gate assuming that there is no deviation involving block durations. The authors rely on empirical distributions of primary gate delays, which are derived for every departure airport. Tam, Ehrgott & Zakeri (2009) [107] also model primary gate delays. The delays are forecast by two factors: a probability that the leg is delayed

and delays length distribution. Both factors are estimated for each weekday separately by regression, which is parametrised by the scheduled arrival and departure times and by the arrival and departure airports of the leg.

Schaefer et al. (2005) [94] forecast leg delay using two random variables: one for primary gate delay and the other for block duration. The length of primary gate delays follows a Beta distribution truncated at 5 hours. The parameters of the Beta distribution are the same for all airports and day times. The block duration is classified into three classes according to the scheduled leg duration: short, medium, and long. The Gamma distribution is chosen for the short and for the long legs and the Erlang distribution is chosen for the medium legs. The classes of distributions and specific parameters of the distributions are determined based on results of the chi-square test. They admit inaccuracy of this classification caused by lack of the data.

5.1.2 Structure of the Model

Our model follows model of airline operations explained in Section 3.3 where we also show how to compute the propagated delay under the model. We also demonstrate structure of the model on a small example.

We split the aircraft operations into two phases: a gate phase and a block phase. Random variables are linked to each phase in order to express the deviation between their scheduled and real durations.

Time between arrival and scheduled departure, or a moment when the aircraft is ready to depart in case of delayed arrival, is not considered to be stochastic. We assume that aircraft needs at least minimum ground time for the preparation and since an aircraft cannot depart earlier than scheduled leg departure, any spare time remains unused. Therefore, potential deviations of the gate phase are always non-negative and represent primary gate delays. We model this delay by two random variables: probability that leg gets delayed and length of this delay. For the block phase, the deviation may be either positive or negative.

We find it very important to keep our stochastic model simple. Therefore, we identify only the most important factors on which the distribution parameters depend. This approach is advantageous because it makes the model suitable for scenarios where the historical datasets are rather small. Besides, it is questionable whether more factors would bring higher accuracy of the forecast

since finer differentiation decreases amount of relevant historical data and, as a consequence, decreases the accuracy of estimates.

Moreover, we keep in mind that the modal has to be usable in situations where new legs are introduced into the schedule. It means that the model has to forecast delays also for connections without historical data, that is, it should rely on few information about the new leg available beforehand. For example, origin-destination pair should not be a parameter of the model since a new leg may connect a new origin-destination pair. For such leg our model could not forecast anything.

5.2 Historical Data

We analysed historical data of a European short haul carrier. The dataset contains flight legs of four subfleets within 28 months with 300-650 legs per day and approximately $350,000$ legs in total. The legs are connecting more than 400 different origin-destination pairs in the hub-and-spoke network with two hubs. Figure 5.1 illustrates the network with legs of one subfleet for one day.

For each leg, the historical data consist of records providing information about scheduled and actual time of departure, scheduled and actual time of arrival, departure airport, arrival airport, and the identification codes of the scheduled aircraft and the aircraft that actually operated the leg. If the aircraft departed with delay, the record contains the IATA delay code and duration of the delay associated to this code. If the leg was cancelled, we do not have any information about the leg.

International Air Transport Association (IATA) established the standard for classification of delays. Delays are categorised into ten groups, which are listed in Table 5.1.

Clearly, one leg may have more than one reason of delay. For example, a propagated delay from the previous leg and a new primary delay, or more reasons of the primary gate delay at the same time. In this case, we are provided all reasons with IATA delay code and associated delay duration.

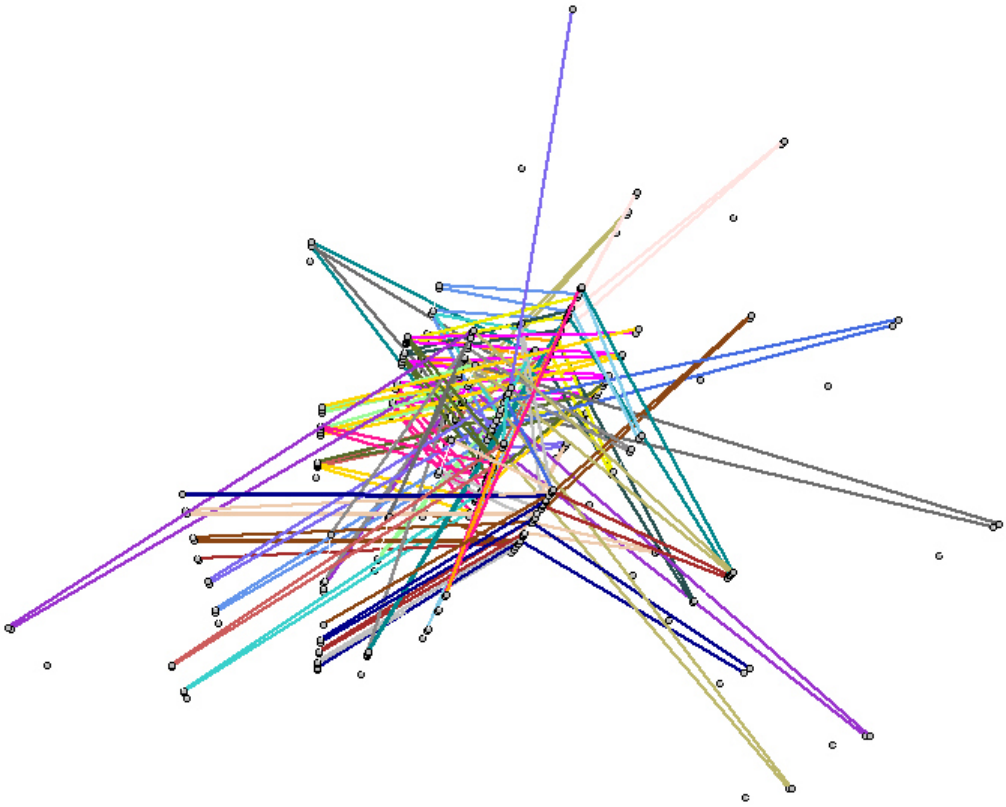


Figure 5.1: The flight network for one day of operation of a European short haul carrier (geographical location with time axis going up). Colours distinguish different aircraft, each line is one leg.

5.2.1 Data Filtering

In order to build the stochastic model of primary delay, we have to filter out propagated delays from historical data. Propagated delays result from primary delays and the airlines network and operational decisions. We explain this on the motivational example in Section 3.1.

Jetzki (2009) [64] provides an elaborate study on propagated delays; she analyses delay propagation for three business models: hub-and-spoke, point-to-point and low cost. The author reports that 40% of delays are propagated in hub-and-spoke networks and 45% in point-to-point networks. Low cost carriers have the highest share of propagated delay which is about 50%.

Effects of delay, of course, depend on various other factors like robustness of

IATA Code	Description
1-9	Internal airline codes
11-19	Passenger and baggage handling
21-29	Cargo and mail handling
31-39	Aircraft and ramp handling
41-49	Technical delays
51-59	Damage to aircraft and automated equipment failure
61-69	Operations and crew
71-79	Weather
81-89	Air Traffic Control and Airport or Governmental Authorities
91-99	Reactionary and Miscellaneous

Table 5.1: IATA Delay Codes classification.

the schedule, concrete network structure, congestion of served airports, etc. In our data set, we observed that 30% of all recorded delay minutes are due to propagated delays.

In the first step of the filtering, we reconstruct the scheduled and actual aircraft rotations based on historical data with IATA delay codes. A rotation is a chain of legs flown by a single aircraft during a single day. Since we analysed European continental data, two rotations are always separated by an overnight stay.

We decompose the activities of the rotations in an interlaced sequence of periods of scheduled (actual) ground time - which is the time between scheduled (actual) arrival and scheduled (actual) departure of the next leg - and scheduled (actual) block durations - which is the difference between scheduled (actual) arrival and departure time.

Availability of IATA delay codes in historical data makes decomposition of delays into primary and propagated easy. One has to omit delay records with delay codes $91 - 96$. Delay code 93 corresponds to rotational delay propagation and delay codes 91-92,94-96 correspond to non-rotational delay propagation. Table 5.2 provides definition of individual delay codes from 91 to 99. In our dataset, 81% of propagated delays account to rotational propagated delay and only 19% to non-rotational.

IATA Code	Description
91	load connection, awaiting load from another flight
92	through check in error, passenger and baggage
93	aircraft rotation, late arrival of aircraft from another flight
94	cabin crew rotation, awaiting crew from another flight, (flight deck or entire crew)
95	crew rotation, awaiting flight deck or entire crew from another flight (including deadheading crew members)
96	operations control, rerouting, diversion, consolidation, aircraft change for reasons other than technical
97	industrial action with own airline
98	industrial action outside own airline, excluding ATS
99	other

Table 5.2: IATA Delay Codes 91-99.

Notice that there is another method which can be used to purify data from the propagated delays instead of usage of IATA codes. Considering our stochastic model, one can compute primary delays directly from the arrival time, the departure time, and the minimum ground time of each airport. More precisely, the arrival delay minus buffer time is the propagated delay and thus, the departure primary gate delay of the next leg is departure delay minus propagated delay. Please note that the values of primary delays obtained by this way sometimes differ from the ones we see directly in historical data.

5.3 Methodology

Sections 5.4 and 5.5 are dedicated to the selection of suitable probability distribution classes and their parameters for the random variables of primary gate delay and block duration deviation. Before that, we discuss our approach, indicate the goals we want to reach, and we give an overview of the methods used during this process.

Choice distribution class

First, we pick a set of distribution classes for each random variable. These classes are chosen upon consideration of the shape, (a)symmetry, and domain of the historical data we want to fit.

We apply the maximum likelihood estimation as the basic method for the distribution fitting. This method returns parameters for the given distribution class which are most likely to produce the observed data. We apply fitting only on data sets with at least fifty observations.

Choice of parameters: chi-square test

If we have more distribution classes that are suitable to model the data, we find the best parameters for each class and compare them. We use the goodness-of-fit chi-square hypothesis test together with visual examination of quantile-quantile plots in order to choose the distribution that matches best the empirical data.

The goodness-of-fit chi square test decides whether some statement, also called null hypothesis, is plausible or not. In our case, we test whether a given data set comes from a population of a specific distribution. An advantage of this test is that it can be applied to any univariate distribution with known cumulative density function.

First, data has to be grouped into bins. The support of the distribution is divided into intervals. Each bin contains the samples with the value from the certain interval. Then the test computes probability with which the samples from theoretical distribution could be binned as we observe it in tested set. This probability is called p-value. We accept the null hypothesis if the p-value is greater than 0.01. Otherwise, we reject it. Please note that acceptance of the null hypothesis means that the observed data are drawn from the tested distribution with probability at least 1%. Typical acceptance levels recommended in the literature are 0.01 and 0.05.

The results of the chi square test depend on the choice of bins. By default, we pick uniform size of the bins. In order to achieve higher accuracy of the method, it is recommended to have at least seven samples in each bin. Therefore, we always merge a bin with its successor until it contains at least seven elements.

Choice of parameters: justification by visual inspection

In general holds that the more data samples available the more accurate the results of the test. However, due to the imprecision in real world data, which we explain in more details later, the chi square test may incorrectly reject the null hypothesis for large sample sets. In such cases we rather rely on our own judgement supported by graphical comparison of the real data and theoretical distribution.

Visual inspection is not a rigorous mathematical method, but it plays an important role during the fitting. We use histograms and quantile-quantile plots to compare historical data with fitted distributions. Quantile-quantile plot, or shortly QQ-plot, compares quantiles of historical data and theoretical distribution. If the points on the graph roughly lie on a plotted line, then we can say that the data comes from the theoretical distribution. On QQ-plots, in comparison to histograms, deviations in tail parts and shifts of the data are more notable.

During our work, we found e-Handbook of Statistical Methods eHa [1] very useful in sense of practical hints for work with statistical methods and their interpretation where we refer reader for further information.

5.4 Primary Gate Delays

First, we look at histograms of primary gate delays on airports. Table 5.2 shows histograms for two different airports. Legs with no primary delay are included in this table as legs with zero primary gate delay. It is possible to see that the histograms have significant peaks at zero. However, the histograms differ significantly in magnitude of this peak. Obviously, it is not possible to fit any continuous distribution to such data with satisfactory accuracy. Therefore, we split the forecast of primary gate delays into two random variables: a Bernoulli random variable that predicts whether primary delay occurs or not, and a continuous random variable that estimates the length of this delay.

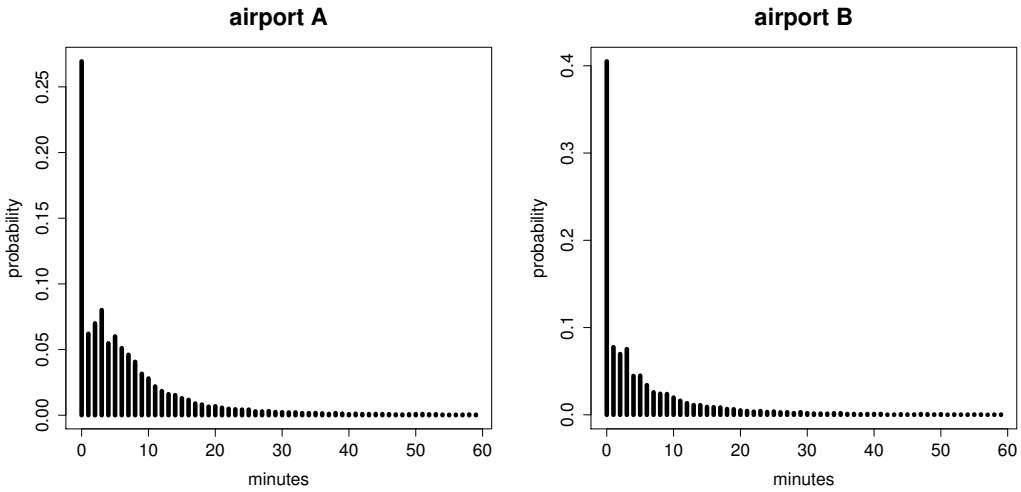


Figure 5.2: Histograms of primary gate delays at two airports including on time departures.

5.4.1 The Probability of Primary Gate Delays

First, we look at factors influencing the occurrence of primary gate delays. We identified the departure airport and the departure time of the leg as the most important factors.

Hence, we derived the probability of occurrence of a primary gate delay for each airport and for each hour of the day as the ratio of the number of departing legs with primary gate delays and total number of all departing legs.

Figure 5.3 presents derived probabilities by hour on three chosen airports during the day. You can see that the rate of delayed legs increases and decreases during the day. Besides, the graphic shows that there are significant variations in probabilities for different airports.

Airport Aggregation

Unfortunately, even a data set containing $350,000$ historical observations does not guarantee enough information for satisfactory estimation of the probability of primary gate delay for every airport and every required hour. Hence, we identify airports with few historical samples that may have similar

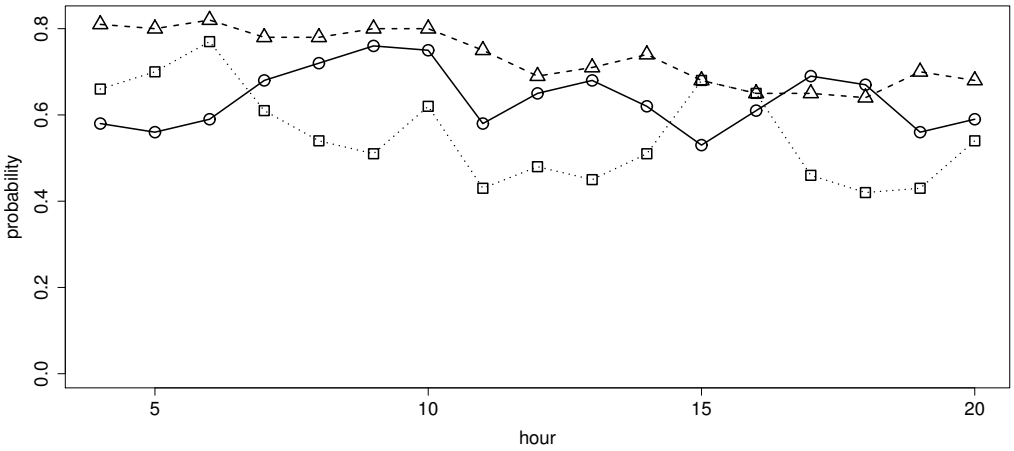


Figure 5.3: Probability of primary gate delay on three airports for different hours.

properties and aggregate them. Under assumption that the airports generate primary gate delays at about the same rate, we get an improvement in the quality of the probability estimates.

We group the airports with few samples into two groups based on their size measured by the number of transported passengers. We believe that small regional airports should have similar properties, as well as more congested airports should exhibit similar properties. We support our decision by observation that the probability of primary gate delay on small airports is much smaller than on large and highly congested ones. We think, that this aggregation leads to more realistic probabilities than derivations of the probability based on, say, ten to twenty samples.

Availability of historical data with primary gate delays from other airlines may help to overcome data shortage on occasionally served airports. But, before doing so, one should also examine whether frequency of primary gate delays is airline independent or not.

5.4.2 The Length of Primary Gate Delays

As for the probability of primary gate delay, we first focus on factors influencing the distribution. Our analysis showed that the distribution of the

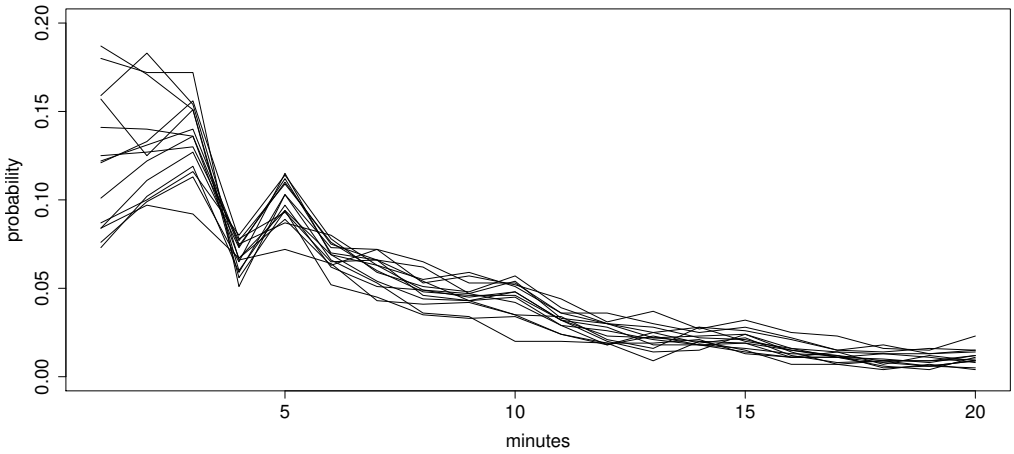


Figure 5.4: Histogram of primary gate delays at an airport for different hours.

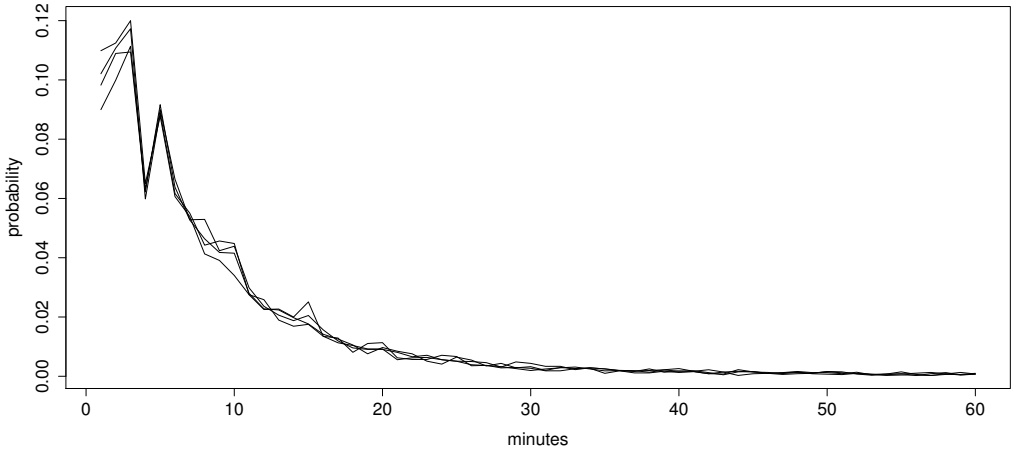


Figure 5.5: Histogram of primary gate delays at an airport for different weekdays.

length of the primary gate delay is independent of all studied factors.

We illustrate our conclusions on Figure 5.4 and Figure 5.5. Figure 5.4 demonstrates independence of primary gate delays on daytime. We present histograms for a chosen airport at different hours. Each curve represents a histogram for different one hour interval. The histograms exhibited in Figure 5.5 show independence on weekdays; that is, each curve represents a

different weekday. Similar results provide partitioning data according to seasons and months, which are discussed in Section 5.8.

Modelling of Heavy Tails

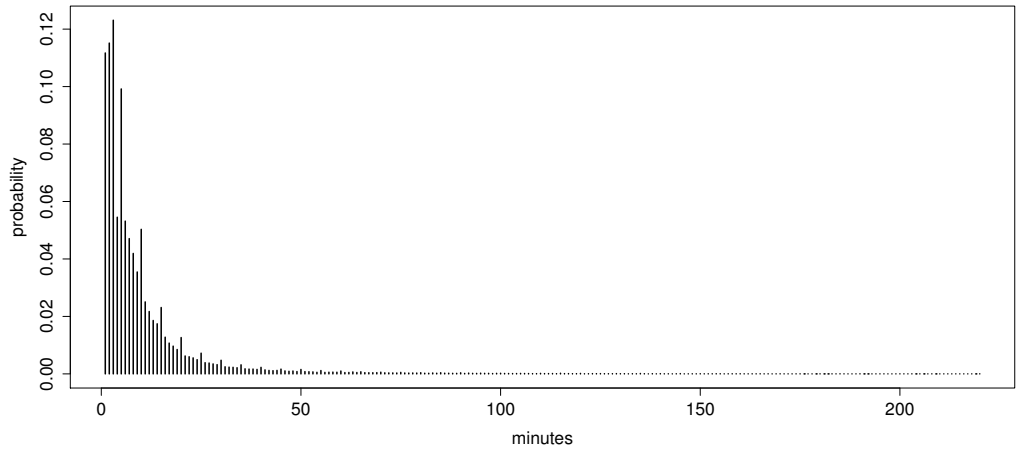


Figure 5.6: Histogram of historical primary gate delays.

Figure 5.6 shows a histogram of the length of the primary gate delay from historical data which we want to fit by some continuous probability distribution. It is possible to see that the probability of a particular delay decreases with length rapidly. However, the number of long delays is, still, significant. In fact, 2% of all primary delays are longer than one hour and 1% is longer than 80 minutes.

We are not aware of any probability distribution with such a heavy tail. This may appear irrelevant, but we believe that proper modelling of tail is important. Long delays are causing serious problems to airlines and thus, proper handling is of high importance for study of robustness in the planning.

Hence, we model the tail of the distribution separately. We examined different values ranging from 50 to 100 minutes and we conclude that 60 minutes provides the best threshold value separating the distribution into two parts, as this value yields the best quality of fit for both parts of the distribution.

In addition, we bound the maximal modelled delay to 220 minutes. Longer delays are barely observed in the historical data. Moreover, one does not have

accurate evidence of such long delays. Often, when so long delay occurs, the leg has to be cancelled which is then not recorded in our historical data.

Distribution Fitting

A thoughtful reader certainly noticed regular peaks in the histogram on Figure 5.6. These peaks occur every five minutes in values divisible by five. We believe that this imperfection is caused during data gathering. Indeed, not all airports suffer from this data corruption in the same way.

We tried to smooth the data to eliminate this problem. We applied various smoothing methods. Since it is questionable how to smooth first minutes of the distribution, we rather rely on the original, not smoothed, historical data. This data corruption results in poor performance of statistical tests, as the chi square test. Therefore, we often based our decisions on examination of QQ-plots or histograms.

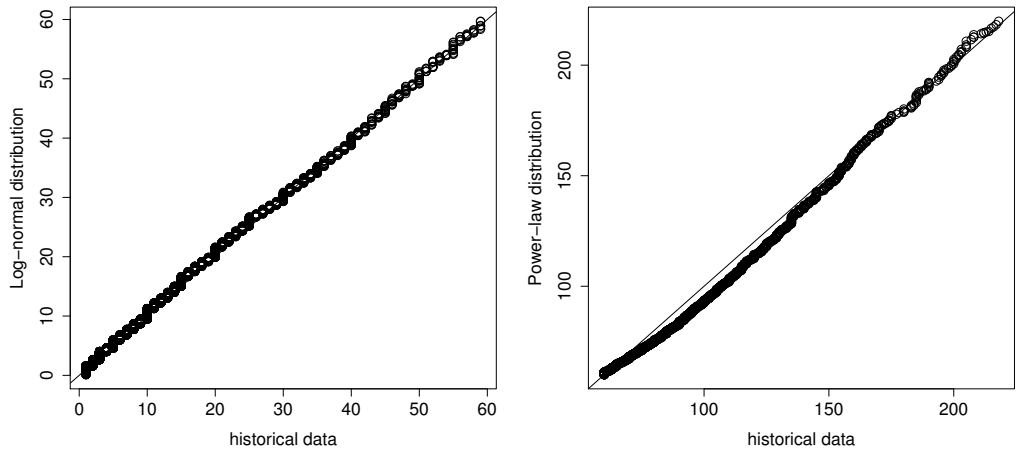
The shape and the range of the delay distribution suggest usage of the following distribution classes: Exponential, Log-normal, Gamma, Weibull, and Power-exponential distribution. We use R, a software for statistical computing and graphics, for distribution fitting (see R Development Core Team (2008) [89]). We apply the maximum likelihood estimation to fit distributions to the historical data.

Aforementioned data corruption causes poor performance of the chi square hypothesis test and of any other statistical test. Therefore, we choose suitable distribution by comparing QQ-plots. Log-normal distribution with $\mu = 1.66$ and $\sigma = 1.07$ performed the best. Although, Power-exponential distribution performed also very well. Figure 5.7a shows QQ-plot of the selected Log-normal distribution against historical data.

A natural candidate to model the tail of the distribution is Power-law distribution. The probability density function of Power-law distribution can be defined as

$$Pl(x) = \begin{cases} c\frac{x^{-\alpha}}{x_{min}} & x > x_{min}, \\ 0 & \text{else}, \end{cases}$$

where $x_{min} > 0$ is the lower bound and c is normalisation constant that ensures that the function integrates on the domain to one.



(a) (b)

Figure 5.7: QQ-plot of Log-normal and Power-law distribution and historical data.

In our case $x_{min} = 60$ and we introduce upper bound on domain since we do not want to model delays which are longer than 220 minutes. Hence, the normalisation constant is computed as

$$c = \frac{220^{-\alpha+1} - 60^{-\alpha+1}}{60\,(-\alpha+1)}.$$

Clauset, Shalizi & J. (2009) [34] discuss Power-law distribution and methods for fitting the distribution to the data. They propose a novel fitting method suitable especially for this distribution class. Unfortunately it does not work well for our data set. Similarly, the maximum likelihood estimation did not provide us meaningful results. Thus, we estimate parameters empirically by comparing QQ-plots and results of the chi-square hypothesis test. Figure 5.7b shows QQ-plot of Power-law distribution with $\alpha = 3.7$ which provides the best fit.

5.4.3 Summary

We conclude with a short formal summary of the presented results. A primary gate delay occurs with probability $p(d_i, t_i)$, and with probability $1 - p(d_i, t_i)$ there is no primary gate delay where p is a function of departure airport d_i, and hour of departure t_i of leg i. Let us denote by $f_{G_i}(x)$ the probability density function for a length of primary gate delay of leg i. Then,

$$f_{G_i}(x) = \begin{cases} k_1 Ln(x, \mu, \sigma), & 0 < x <= 60, \\ k_2 Pl(x, \alpha), & 60 < x < 220, \end{cases}$$

where $Ln(x, \mu, \sigma)$ denotes the probability density function of Log-normal distribution with mean $\mu = 1.66$ and standard deviation $\sigma = 1.07$. Further, $Pl(x, \alpha)$ denotes the probability density function of Power-law distribution with parameter $\alpha = 3.7$. Constants k_1 and k_2 are scaling factors which assure that

$$k_1 \int_0^{60} Ln(x, \mu, \sigma) dx = 0.98,$$
$$k_2 \int_{60}^{220} Pl(x, \alpha) dx = 0.02.$$

5.5 Block Time Deviation

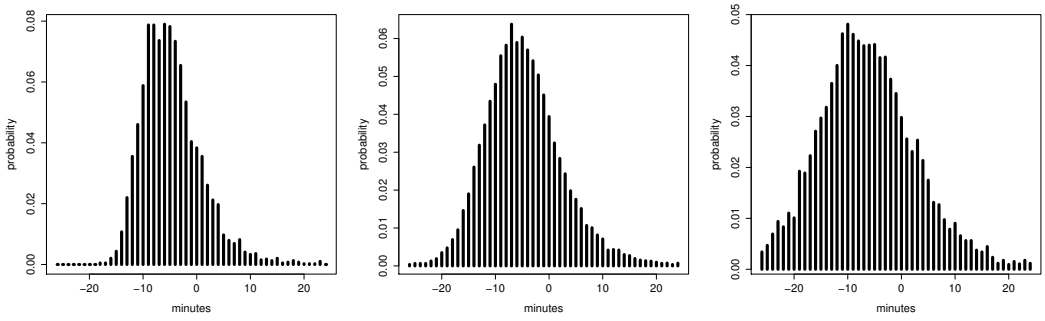


Figure 5.8: Historic block time deviation for legs with scheduled leg duration of 40, 80, and 120 minutes.

The most natural and accurate parametrisation of distributions of the block time deviation is the leg number. This is, unfortunately, very fine differentiation. For illustration, if a leg is flown once a week, we have 52 samples a year. In fact, only 72% of all leg numbers have more than 50 historical records, which, we believe, is the smallest reasonable data set for parameter estimation.

We looked for more rough data decomposition. We clustered data by origin-destination airport pairs. Such data aggregation still does not improve data availability for many cases. Moreover, we see that this aggregation is not valid, as different connections for the same origin-destination airport pair, for example morning and afternoon flight, may have even different scheduled block duration. Varying scheduled block duration of the same origin-destination pairs may be at first sight surprising, but airline planners are frequently forced to develop schedules where such variations appear in order to match scheduled and observed block times. It means that the same leg number may have different scheduled block time in consecutive seasons, if punctuality was not satisfactory. On the other hand, sometimes the same connection at different day times has different block time due to for example, longer taxi in/out caused by higher airport congestion. For these reasons, we decided not to use origin-destination airport pairs.

Clearly, scheduled block duration influences the distribution of deviation. For illustration, see Figure 5.8 picturing three histograms of block time deviation. You see that the longer the block time the greater the standard deviation of the distribution.

This observation supports our decision to take the scheduled block duration as the only factor influencing the parameters of distribution of the block time deviation. We believe that manual adjustments of scheduled block times, performed by the airline planners, eliminate significance of other effects.

Now, we address the selection of the distribution class and of the parameters for the random variable. We cluster historical observations according to the scheduled block duration into 32 groups. Each group contains actual block durations belonging to the legs whose scheduled block duration have certain value. These groups cover connections from 40 to 200 minutes of flight time. As candidate distribution, we have chosen: Weibull, Normal, Log-normal, Logistic, Log-logistic, and Gamma distribution. For each group, we estimated the parameters for these distribution classes using the maximum-likelihood estimation.

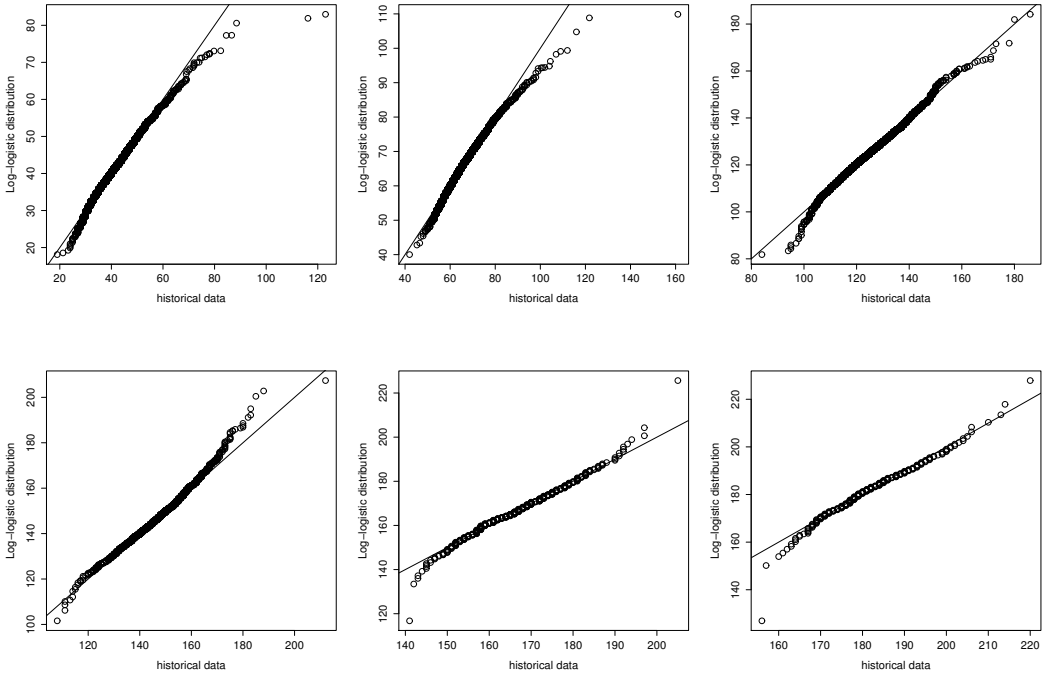


Figure 5.9: QQ-plots of observed data and Log-logistic distribution for scheduled block duration 45, 70, 130, 155, 175, and 190 minutes.

Based on results of the chi-square hypothesis test, with bin size of one minute, and on visual inspection, we pick Log-logistic distribution, as it provides very good quality of modelling for all groups. Just for illustration, see Figure 5.9 picturing quality of the fit for scheduled block time of 45, 70, 130, 155, 175, and 190 minutes.

Since Log-logistic distribution has support $\mathbb{R}^+$, we did not fit historical block deviation, which acquires also negative values, but historical block time. Distribution of the deviation is then obtained after subtraction of the scheduled block time.

Figure 5.10 summarises obtained parameters of Log-logistic distribution for all groups. Shape parameter ranges from 0.03 to 0.1 and scale parameter from 3.5 to 5.2.

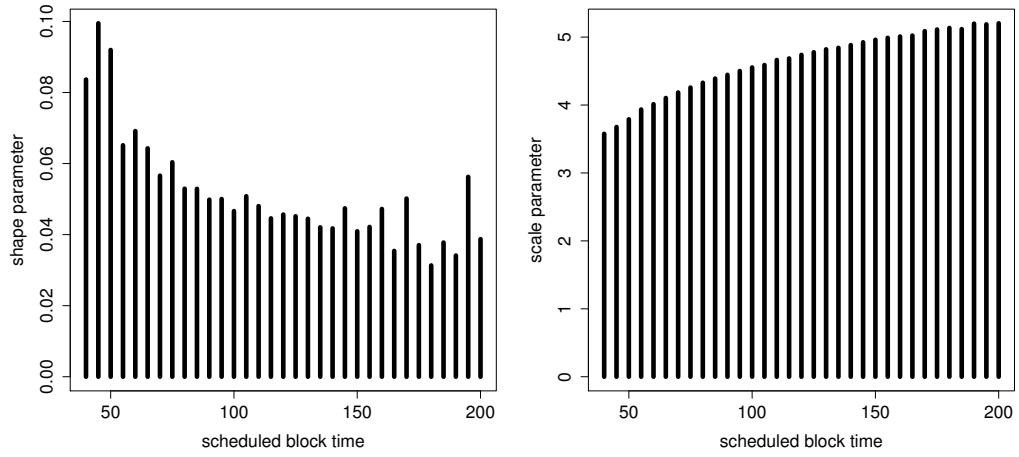


Figure 5.10: Shape and scale parameters of Log-logistic distribution for scheduled block times from 40 to 220 minutes.

5.5.1 Summary

We model block delays by a Log-logistic distribution with probability density function $f_{B_i}(x)$ of block time deviation of leg i

$$f_{B_i}(x) = Llg\big(x + \ell_i, \alpha(\ell_i), \beta(\ell_i)\big), \quad x \in \mathbb{R}.$$

Here, $Llg\big(x, \alpha(\ell_i), \beta(\ell_i)\big)$ is the probability density function of Log-logistic distribution with scale parameter $\alpha(\ell_i)$ and shape parameter $\beta(\ell_i)$, where ℓ_i denotes the scheduled block time of leg i. Figure 5.10 pictures values of estimated scale and shape parameters.

5.6 Minimum Ground Time and Minimum Sit Time

In our model, we assume that the minimum ground time is a constant varying from 25 to 40 minutes which depends on the aircraft type and on the airport. The values are given by our project partner and are estimated based on the experiences of the planners. The minimum sit time is considered to be the same for all aircraft types, airports, and crew members and equals 40 minutes.

In an ideal model, the minimum ground and sit time values should be modelled by a random variable. Unfortunately, the structure of available historical data does not give us the possibility to extract relevant historical data for such analysis.

If an aircraft departs on time, one cannot distinguish the time spent for the preparation for the next leg from idle time. Conversely, if primary delay at the gate occurs, we do not know whether it is due to unsatisfactory ground time or to some other event. It is questionable what is still a standard length of the ground time and what is already a primary delay.

Our analysis shows that the actual ground time is rarely markedly shorter than the minimum ground time. The historical data do not allow us to verify correctness of the considered minimum sit times.

5.7 Properties of the Stochastic Model

Just like every stochastic model, our model is a simplification of reality. Here, we discuss assumptions under which our model is valid, and their connection to assumptions required by the method of computation of propagated delay presented in Chapter 3.

Early Departures

The structure of the block phase does not allow early departures from the gate. We observed very few departures one to five minutes ahead of scheduled leg departure in the historical data, we think that such inaccuracy is not significant. Moreover, modification of the structure of the stochastic model can easily accommodate such requirement without influencing the discretisation method implied in the computation of propagated delay.

Additivity of Delays

The stochastic model assumes that the duration of each phase of the aircraft operation is independent of the actual delay of the leg. Thus, each phase can be forecast independently, and the propagated delay is then computed as a sum of primary delays (or deviations) of all phases.

Clearly, the validity of this assumption is not guaranteed and needs further exploration. We compared the frequency of primary gate delays of legs with propagated delay with the frequency of primary gate delays of legs without propagated delay. We could not identify any significant difference or any trend in the frequencies. The same experiment with histograms of the length of primary gate delay did not show any deviations as well. The average block times of legs with and without propagated delay differ in less than one minute, which is also irrelevant.

Nevertheless, the discretisation method can be adapted in order to handle situations where the additivity of delays is not valid.

Independence of Primary Delays of Legs

Further, we expect block deviations and primary gate delays of two legs to be independent. This does not have to be true in reality, since weather conditions, like strong wind, may influence the block duration of many legs at the same time. Moreover, some problems at the airport may delay many legs departing from this airport in certain time window.

Unfortunately, it is not possible to consider these aspects in the model without making the computation of propagated delay intractable.

Level of Detail

One can easily come up with a more accurate model where the probability of primary gate delay at different year seasons and week days can be distinguished, distribution of delay length for each airport can be derived, or deviation of block duration can be modelled for each leg separately. Also, it is possible to introduce more phases. However, it is important to remark that models with more parameters require larger data sets in order to get accurate estimates.

All these enhancements can be incorporated within usage of discretisation method.

5.8 Model Justification

In the previous sections we focused on finding accurate model for observed empirical data. However, this does not guarantee practical usability of the model. One has to show, whether the model is also able to forecast delays in the future with high accuracy. This section is dedicated to answering this question.

In Section 5.8.1, we examine the stability of all used parameters of the model in time, which is the key property for high quality forecasts. We conclude the chapter with a "proof of concept" experiment, in Section 5.8.2. This experiment is designed to question all assumptions taken by our stochastic model and to shows that the model is able to forecast probability of delay propagation along rotations with very high accuracy.

5.8.1 Stability of Parameters

Here, we show that historical data from one year allow us to forecast all parameters of the model with high accuracy for the next year. Since our historical data contain more than two years of records, we can compute the parameters derived for 2005 and 2006 separately and compare them.

The Probability of Primary Gate Delay

In order to demonstrate the stability of probability of primary gate delay, we compare computed probabilities for the seven most visited airports in our data set. Many airports, with less historical observations, cannot give clear evidences of stability of their parameters and also are not modelled separately in the stochastic model.

Table 5.3 lists derived probabilities at 8^{am} for year 2005 and 2006. For a clear picture about reliability of each computed value, we give rough number of observations available for each year in the last row of the table.

We believe that all studied airports with the exception of airport E demonstrate high stability of acquired probability of primary gate delay. Deeper analysis of airport E shows that there is an increasing amount of delayed legs during the whole day.

	A	B	C	D	E	F	G
2005	0.72	0.78	0.46	0.44	0.64	0.56	0.52
2006	0.72	0.78	0.54	0.56	0.46	0.60	0.50
samples	3000	1000	550	500	300	300	250

Table 5.3: Probability of primary gate delay at seven most congested airports at 8^{am}.

It means that there was some abrupt change in functioning of the airport from the point of view of delay generation. Obviously, such changes cannot be forecast by our model but may be incorporated manually by planners.

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2005	0.64	0.69	0.75	0.71	0.75	0.79	0.74	0.77	0.81	0.76	0.67	0.72
2006	0.70	0.70	0.67	0.72	0.77	0.72	0.79	0.73	0.72	0.78	0.72	0.72
2007	0.71	0.70	0.76	0.79								

Table 5.4: Probability of primary gate delay at 8^{am} on airport A in different months.

Furthermore, we show stability of probabilities for individual months as well. Table 5.4 summarises results for airport A at 8^{am}. The probability at each month is derived from 180 to 314 samples. It is possible to see that only small deviations happen when we compare the values for each month.

The Primary Gate Delay

Let us show the results of a similar experiment with length of primary gate delays. We derived a histogram of primary gate delays for 2005 and 2006. We observed very high persistence of distributions; moreover, different months during these years have also alike distributions. To manifest this conclusion, see Figure 5.11 which displays histograms of primary gate delays in chosen months.

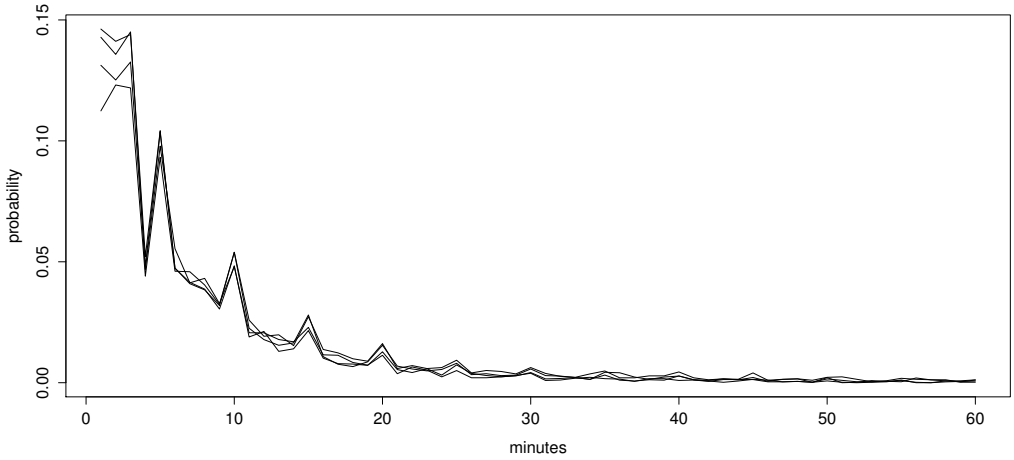


Figure 5.11: Histogram of primary gate delays in January 2005, August 2005, April 2006, and December 2006.

Further information regarding delay statistics on airports can be found on CODA [35] webpage. There are annual and monthly statistics available, regarding delays at airports and year-on-year changes.

The Block Time Deviation

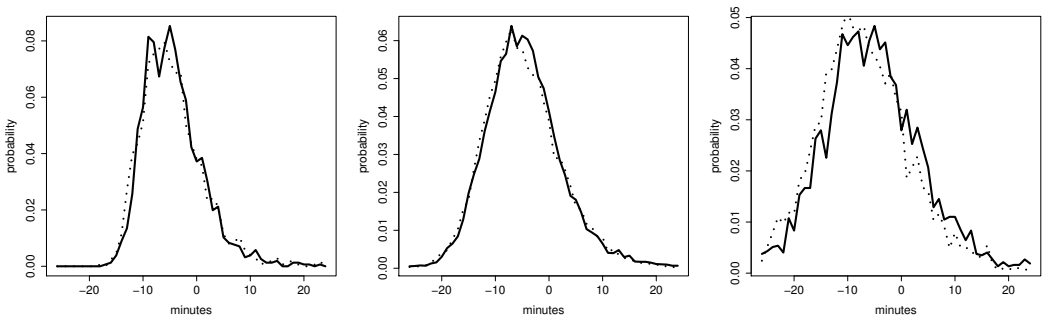


Figure 5.12: Historic block time deviation for legs with scheduled leg duration of 40, 80, and 120 minutes in year 2005 (solid line) and 2006 (dotted line).

Next, we study block times. We compare distributions for groups of scheduled block duration. This again shows a very high persistence. Three ran-

domly chosen histograms are provided in Figure 5.12. We went a step further and studied histograms of individual leg numbers. Figure 5.13 shows some examples which prove also very good year-on-year stability.

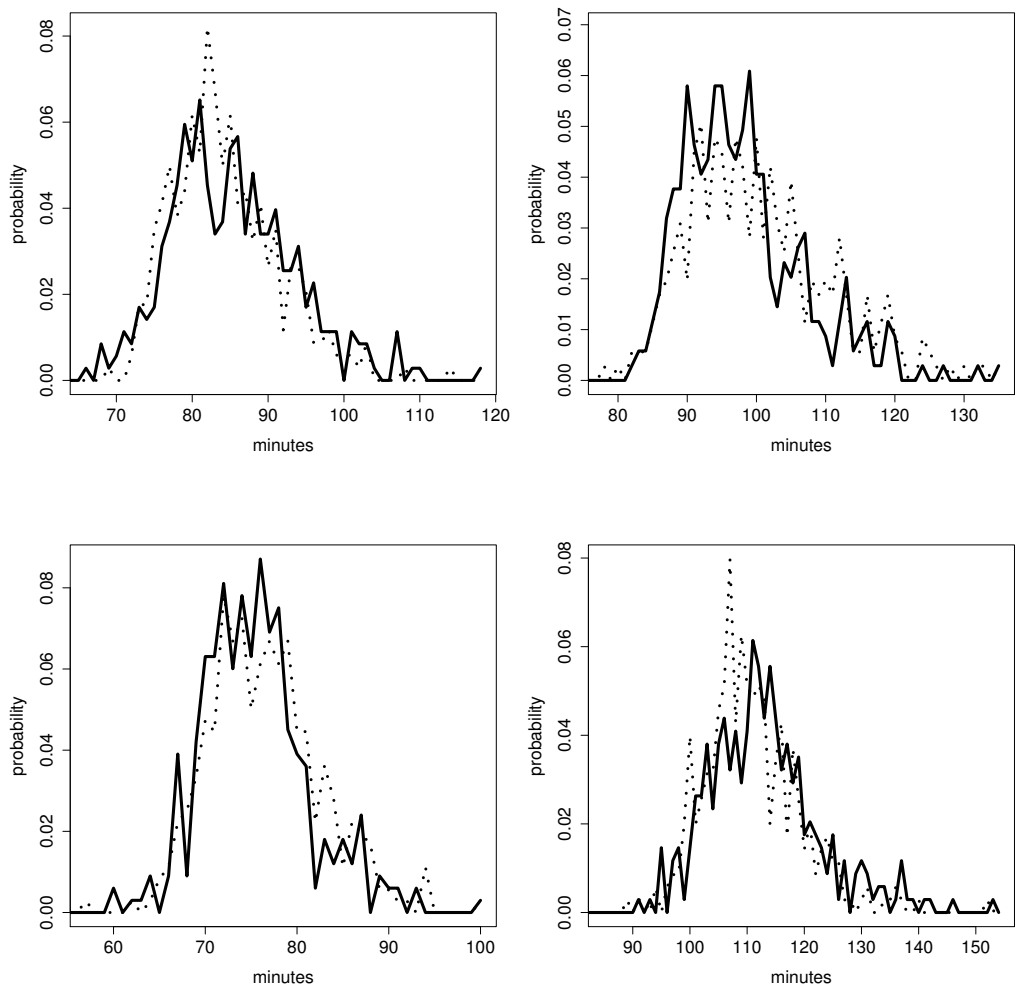


Figure 5.13: Histograms of chosen leg numbers in year 2005 (solid line) and 2006 (dotted line).

5.8.2 Proof of Concept

We conclude this chapter with an experiment that demonstrates the ability of our stochastic model to forecast delay propagation. We compare the delay propagation computed by our stochastic model directly with the observed delay propagation in historical rotations. In order to avoid taking any assumptions, we consider only days where the studied rotation was flown by single aircraft. That means, if all legs of the rotation were flown on certain day but by various aircraft, we do not consider such day.

This approach allows us to question all parts of the stochastic model at the same time, since data gathered from historical rotations are independent of the structure and assumptions of our stochastic model, that is, they really reflect what actually happens.

We remark that the only inaccuracy of this experiment is associated with the fact that some long disruptions may be filtered out. Namely, if there was some serious disruption on the studied rotation and planners have therefore changed the rotation or cancelled the leg, then this day, and consequently this delay, does not appear in our sample set.

This procedure requires sufficient number of stable rotations that are flown for many days. We, therefore, identified the most frequent rotations in the historical data. We concentrated on rotations with exactly five legs because such rotations are long enough to observe the effects of delay propagation. On the other hand, we could still find enough historical observations of such rotations. Clearly, the longer the rotation is, the less likely it is to be found in the schedule.

Table 5.5 gives description of four chosen rotations. We list the departure and the arrival airport in the first and the second column, respectively. Please notice that airport labels are not consistent across various tables in this chapter. Finally, the scheduled block time appears in the third column and the scheduled buffer in the last one.

Table 5.6 summarises results of the test. The fourth column shows the probability of delay propagation (PDP) computed based on the stochastic model, while the third column gives the average number of experienced delay propagations per day in historical data. The number of observations used to compute the average is presented in the second column.

dep apt	arr apt	block time	buffer	dep apt	arr apt	block time	buffer
Rotation 1				Rotation 2			
A	B	85	0	C	D	90	0
B	A	85	0	D	C	80	100
A	C	55	0	C	D	90	10
C	A	60	0	D	C	80	35
A	B	85	-	C	D	90	-
Rotation 3				Rotation 4			
A	E	90	5	G	H	80	10
E	A	85	45	H	I	150	5
A	F	150	10	I	H	150	55
F	A	150	5	H	J	150	10
A	E	90	-	J	H	150	-

Table 5.5: Description of studied rotations.

	historical data		model
	# samples	avg DP	PDP
Rotation 1	86	1.78	2.21
Rotation 2	172	0.64	0.61
Rotation 3	86	0.67	0.63
Rotation 4	71	0.44	0.49

Table 5.6: Average number of delay propagations for chosen rotations gathered from historical data vs. probability of delay propagation computed by the discretisation method under proposed stochastic model.

This experiment shows very good forecasting quality of the probability of delay propagation along rotations. Together with high stability of the parameters of the stochastic model, which we addressed in Section 5.8.1, we can conclude that our stochastic model is of practical relevance.

Chapter 6

Ops Simulator

Simulation plays an important role in robust planning. Without simulation, it is difficult to quantify the effect of improved robustness before actual employment of a new plan into practise. Increase of robustness is often achieved for the price of increased planning costs which makes difficult to convince practitioners that such plan will decrease operational costs when applied in operation. We believe that bad predictability of the real impact of the robust planning is the key reason discouraging practitioners from bringing robust plans into action. Simulation helps to overcome these problems and presents the best way how to assure practitioners of benefits of the robust planning before actual deployment. Besides, simulation is helpful in development and testing of new ideas in form of robustness objectives, KPIs, or recovery strategies.

We devote this chapter to discrete event-based simulator Ops Simulator which we use as a tool for benchmarking various robust schedules. The simulator is extensively used in Chapter 7, where we assess quality of various robust schedules based on simulation results. We also used the simulator in Section 3.6 for evaluation of the accuracy of the discretisation method.

First, after a short description of other known simulation frameworks in Section 6.1, we introduce a structure of Ops Simulator in Section 6.2. Important parts of the simulator are individually investigated in subsequent sections. In Section 6.3, we give an overview of the core of the simulation. We continue with key parts of the simulator. In Section 6.4 we document the cost model which is important for evaluation of benefits of robust planning. Section 6.5 details the recovery module.

Possibility of the simulator to visualise the schedule and operation turned out to be also very useful during the research. More about the visualisation is provided in Section 6.6.

6.1 Overview of Simulation Frameworks

The simulation of airports and flight legs is a very complex task. To build an universal simulator capturing all aspects of the simulation in high details is virtually impossible. Therefore, known models differ in their structure, which is influenced by the aim of the simulation. Before introducing Ops Simulator, we shortly describe two simulation frameworks closely related to ours: Simulation of Airline Operations (SimAir) and MIT Extensible Air Network Simulation (MEANS). We discuss their structure as well as their field of application.

The main objective of SimAir, documented in Rosenberger et al. (2002) [91] and SimAir [100], is to evaluate various recovery algorithms and to evaluate robustness of plans. SimAir decomposes aircraft activities into the following time segments: ground delay, taxi-out duration, leave ground duration, flying time, touch down duration, and taxi-out duration. It is possible to assign a probability distribution representing duration of each time segment. Furthermore, every airport contains a departure and an arrival queue where the aircraft are queued before take-off and landing, respectively. The queues increase accuracy of modelling the airport's congestion since the waiting time in the queue depends on the activity of other aircraft. The disruption model of SimAir also covers disruptions such as an unscheduled maintenance and an airport closure. The movement of a crew and passengers may be simulated as well.

MEANS, described in Melconian & Clarke (2001) [82] and Clarke et al. (2007) [32], has alike structure as SimAir, from the point of view of simulation of aircraft operations. The leave ground duration and the touch down duration are not explicitly represented, but included in the flying time. On the other hand, some properties of the airports are modelled in much higher details. For instance, they incorporate the capacity of each airport or the influence of each terminal on taxi in/out time. MEANS is, therefore, suitable for studies concerning air traffic management or testing schedules with new leg structure.

6.2 Structure of the Ops Simulator

In the event based simulator Ops Simulator, we focus on the simulation of aircraft and crew operations of a single airline. We do not perform any changes in the schedules in the sense of introducing new legs or rescheduling the existing ones. For these reasons, it is superfluous to simulate the influence of other fleets and airlines explicitly. Therefore, in comparison to MEANS or SimAir, we omit take-off and landing queue from our simulation. Additionally, we simplify the structure of simulated aircraft activities. We do not simulate taxi-in and taxi-out time, which we take as a part of the ground time. So defined simulation model is identical to our model of aircraft operations introduced in Section 3.3 as well as structure of historical data.

Ops Simulator consists of modules that allow us to define the functionality of each component independently and gives us higher flexibility of usage of the simulator. The simulator is divided into the modules shown on Figure 6.1.

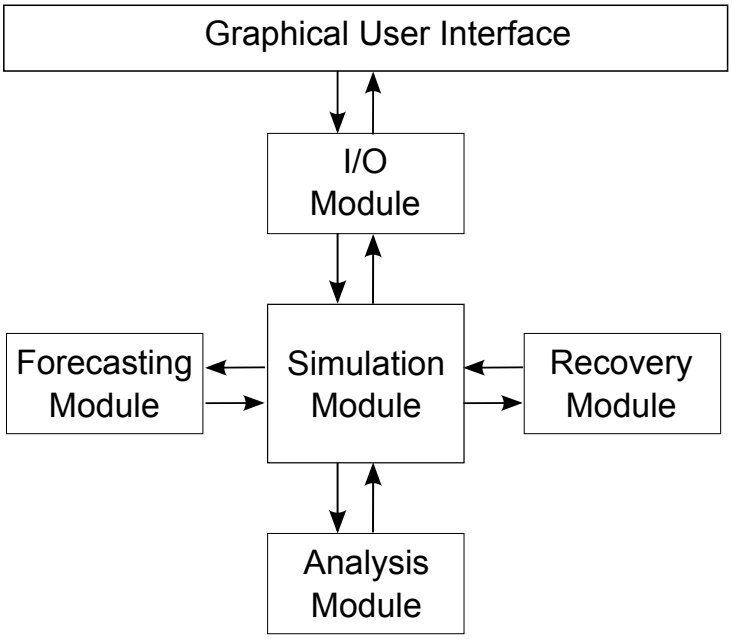


Figure 6.1: Modular structure of Ops Simulator.

The simulation module is the core of Ops Simulator. It performs discrete-event simulation and communicates with other modules. In Section 6.3, we give a deeper description of this module and an overview of the events.

The forecasting module is responsible for determining the primary gate delay and the block duration of each leg. We use two implementations of this module. The first implementation loads the data from an XML file. This gives us the chance to reuse the same disruptions repeatedly or to test schedules under "hand made" disruption. We also run the simulation with historical data that are, at first, loaded from the database to the XML file. The second implementation generates delays at random by sampling from distributions of the stochastic model described in Chapter 5.

The recovery module simulates the function of Operation Control Centre, which monitors the execution of the schedule and performs changes in the schedule when necessary. Section 6.5 discusses the main aspects involved in the implementations of this module.

The results of the simulation are collected by **the analysis module**, which produces various statistics such as total incurred primary and propagated delay, and evaluates robustness by means of punctuality, number of performed swaps and cancellations, number of broken crew connections, number of reserve crew members called up, additional recovery costs, etc. Additional recovery costs are important for practitioners since it allows them to judge the performance of the schedule. Thus, we discuss the development of realistic cost models in Section 6.4.

The input/output module is responsible for loading the input schedules and parameters of the simulation. The results of the optimisation, as well as the input schedules, are written to XML files. The whole simulation can be controlled from the command line or from **a graphical user interface**. The graphical user interface also provides the visualisation of the schedules and the results of the simulation. More details are available in Section 6.6.

6.3 Simulation Module

The structure of the simulation is alike to the model of airline operations defined in Section 3.3. The operations of the simulator can be described by the following main events and corresponding states of the aircraft.

We represent the arrival of the aircraft to the gate by an **Aircraft Arrival Event**. Afterwards, passengers have to be deboarded, the aircraft has to be prepared for the next departure, and new passengers have to be boarded. For all these actions, we account the minimum ground time. Then, the

aircraft is available for the next departure which is reflected by **Aircraft Available Event**. If all crew members are on board but not earlier than at the scheduled departure time of the leg, we say that the aircraft is ready to depart (**Aircraft Ready Event**). Then, the forecasting module assigns a gate primary delay to the leg, if any; after which aircraft departs the airport (**Aircraft Departure Event**).

At this moment, the recovery module is asked to decide whether to perform a recovery action to resolve the problems caused by the possible delayed arrival. This simulates reality where some time for decision making and accommodation of changes is required. If there is no departure delay there is no recovery action performed by the recovery module. Notice that recovery module can not influence operation of already departed legs. Next event, after departing the airport, is the **Aircraft Arrival Event**, which denotes arrival of the aircraft to the gate. Time between the aircraft departure and aircraft arrival event is called block time and is given by the forecasting module.

The structure of the module grants potential to very fast simulation. Indeed, the simulation of one day instances with 100 to 150 legs under $500,000$ randomly generated instances is on average computed in less than one hour on a desktop PC if no recovery is applied by recovery module. Clearly, the employment of the sophisticated recovery module significantly slows down the simulation.

6.4 Cost Model

Delays, and especially cost of delays are topic of numerous papers. Boydell (2005) [27] compiles many reports in order to derive a set of "standard inputs" which Eurocontrol recommends to use in the development of cost-benefit analysis. One of these standard inputs is a delay cost per minute. For delays longer than 15 minutes, the estimated cost of one minute of arrival delay is \$72. This evaluation is made based on the work of Cook, Tanner & Anderson (2004) [38]. Furthermore, Boydell (2005) [27] lists other interesting works on this topic and summarises their suggestions and conclusions.

Clearly, not every delay causes the same problems, and thus not all delays inflict equivalent additional costs. Furthermore, average cost of delay value is compiled from many factors. Extra cost of each contributing factors is

balanced in correspondence to the frequency of their occurrence. There is also no difference between primary or propagated delay which also differ in incurred cost. The application of such value is, therefore, more suitable in situations where no other information about the character of delays and schedule is provided.

Since we do have the information about the primary delays, the crew pairings, and the aircraft rotations, we can apply more accurate model that distinguishes each cost factor individually.

We use the results presented by Cook, Tanner & Anderson (2004) [38], which builds delay costs from various factors and estimate the contribution of each factor separately. Since there is no way how to describe the delay costs in general for any delay length and particular situation, they analyse six different delay scenarios; a low, a base, and a high cost scenario of 15 and 65 minutes of primary delay, respectively. For each scenario, they show the cost analysis of the primary ground and the primary airborne delay for various aircraft types. The costs are derived upon experiences of many practitioners.

They classify cost factors causing additional costs as: airport charges, fuel costs, station expenses, maintenance, passenger costs, and extra crew salaries. Estimation of the cost of every factor for each scenario is given. Furthermore, they analyse factors contributing to the cost of propagated delays as well.

Based on this detailed study, we were able to identify situations where each factor contributes to the total cost as well as the incurred additional costs. The report contains cost analysis for many aircraft types. As a reference, we have chosen the Airbus 320, which is very common aircraft type. In order to estimate the costs, we use a base scenario for 15 and 65 minutes, respectively.

Since we are interested in the evaluation of savings of robust planning, we do not include the costs of primary delays per se in our cost model, we just measure their consequences. Therefore, we omit extra fuel costs incurred by the delay. Notice that such costs are independent of the schedule, which is flown under the given disruption scenario; they contribute in all schedules with the same value (under assumption that leg is flown in all schedules and thus delay occurs).

The biggest contribution to the delay costs comes from the passenger costs, which are a consequence of the late arrival of the aircraft. These costs include rebooking costs for missed connections, passenger compensations, and also potential loss of market share. In considered scenarios they estimate per

minute cost to 43.2 € for 65 minutes long delay. For 15 minutes long delay, in base scenario, no extra passenger costs are incurred. However, we decided to include 1.2 € per minute also in this case. Based on fact that the high cost scenario of 15 minutes long delay incurs extra passenger cost 6.9 € per minute.

The second highest cost involves extra crew salaries, which incorporate overtime crew and reserve crew costs. Since we have exact information about crew overtimes and usage of reserve crews during the simulation, we do not include these costs to our cost per minute value. Number of reserve crews and overtime crew working minutes are stated separately in the simulation statistics.

In studied base scenarios we observed very small extra airport charges and station expenses and we do not include them to the model either.

Altogether, we account 1.2 € per minute for delay of 15 minutes and 43.2 € per minute for 65 minutes long delay with these costs representing the passenger costs only.

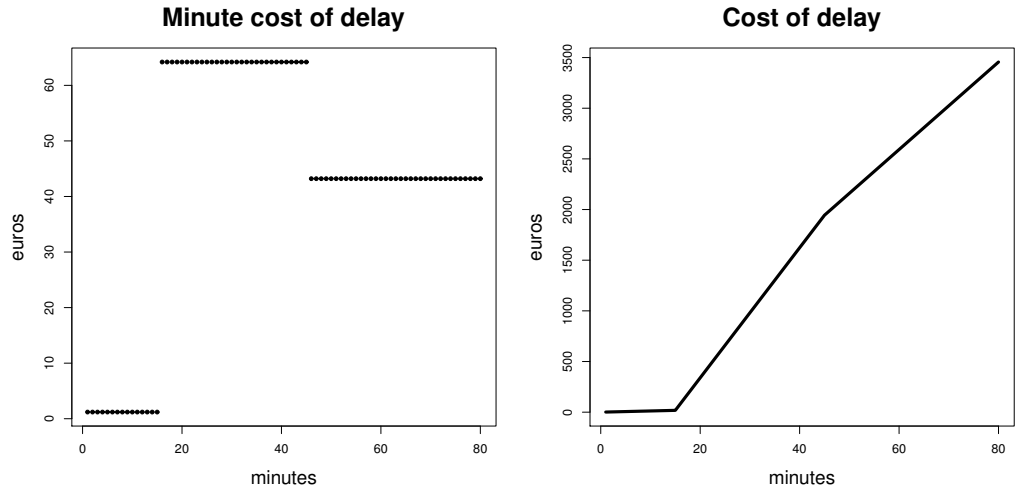


Figure 6.2: Cost of a delay depending on the length.

Cook, Tanner & Anderson (2004) [38] do not propose a method to estimate cost of delay for other delay lengths. Dependence of delay cost on the length is discussed in Eurocontrol (2000) [51]. They show that the costs do not grow linearly. First, the per minute cost grows slowly and after steep growth again slows down, or even decreases. Actually, a few minutes of the delay

almost never cause a problem; conversely, additional minutes of delay to legs which are already very late (by, say, more than three hours), do not bring significant increases in costs either.

We, therefore, propose three distinct cost per minute values for Airbus 320 depending on the length of the delay. For the first 15 minutes of delay we propose cost of 1.2 € per minute. Then for each minute from 15 up to 45 minutes we charge 64.2 €. It makes the average per minute cost of a 45 minutes long delay to be 43.2 €. For each delay minute over 45 minutes we charge 43.2 € which is also the average cost per minute of 65 minutes long delay of our modified cost scenario. Left part of Figure 6.2 summarises cost of minute of delay depending on the length of the delay. Right part of Figure 6.2 shows total cost of delay.

6.5 Recovery in the Ops Simulator

Every day, disruptions enforce changes in the schedules. In order to obtain a realistic picture of the impact of robust planning and to assess its positive effects on the recoverability of schedules, it is important to simulate these changes as well. Hence, the simulator contains the recovery module which verifies and, when necessary, modifies the original schedule.

Alone, the question whether a recovery action is necessary or not is difficult to answer. Therefore, we decided to adopt a simple policy, in which the recovery algorithm is executed if an aircraft departs with more than 30 minutes of delay.

6.5.1 Netline/Ops Solver xOPT

In order to make the recovery decisions realistic, we use the commercial aircraft recovery tool Netline/Ops Solver xOPT, provided by Lufthansa Systems. Netline/Ops Solver xOPT is a column generation based solver. Here, we address the functionality of the solver and the settings of its most important parameters, that it, the ones which determine the behaviour of the tool.

The purpose of our recovery module and of Netline/Ops Solver xOPT is primarily to recover the aircraft from delay. To do so, the solver builds new

aircraft rotations which are brought into operation by the series of aircraft swaps and leg cancellation. In our settings we do not allow usage of ferry legs.

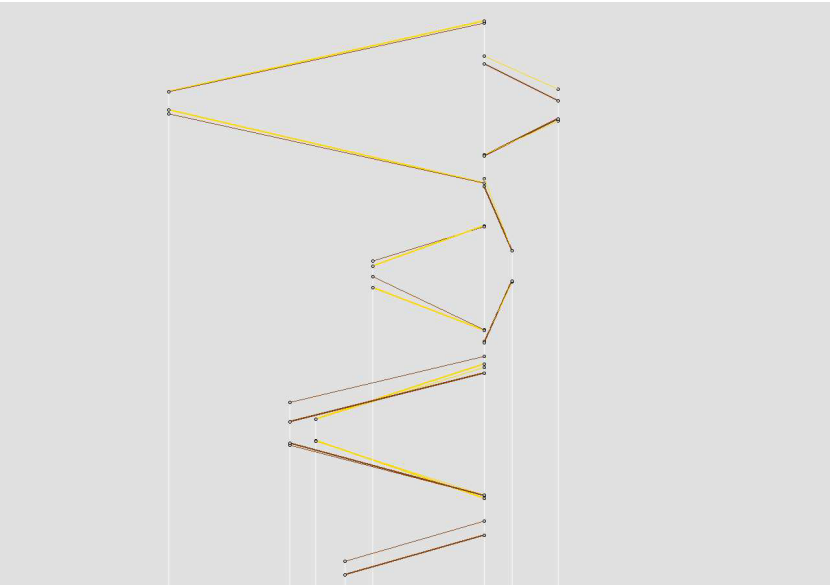


Figure 6.3: Visualisation of an aircraft swap. Each colour represents one aircraft. Thick lines are scheduled rotations, thin lines are operated rotations. Vertical axis represents time.

Figure 6.3 shows an example of an aircraft swap enforced by the recovery module. The thick lines visualise two scheduled rotations. The colour represents the aircraft which should operate the leg. The operated rotations are visualised by the thin lines. You can see that at a certain point the colours of thin and thick line of the leg do not match, what pictures the change of the aircraft.

Furthermore, if the simulation involves crew movement, propagated delay due to crew is considered as well. Thus, the recovery algorithm is allowed to decide whether to wait for the delayed crew members or to break a crew connection and use a reserve crew instead. Realistic usage of the reserve crew is achieved by forbidding to break the crew connections on non-base airports where no reserve crew is assumed. For simplicity, we do not check violation of crew legality rules and we do not track how to fix changed crew pairings.

The objective of NetLine/Ops Solver xOPT is to minimise the sum of penalty for propagated delay, leg cancellation, broken crew connection, and aircraft

	Minute of propagated delay		Swap	Cancellation	Broken
	0-15	15+			crew
Easy	0.1	1	8	150	16
Normal	0.1	1	16	150	32
Difficult	0.1	1	23	150	46

Table 6.1: Overview of key parameters defining three settings used for simulation under recovery

swaps. Behaviour of NetLine/Ops Solver xOPT is controlled by the adjustment of these parameters.

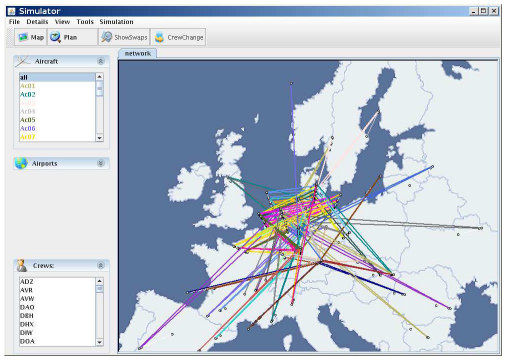
We perform our experiments under three different settings. Our aim is to simulate different settings from a swap vulnerable behaviour to very conservative behaviour unwilling to change the schedule.

Table 6.1 summarises used values. In all three scenarios we consider same values for propagated delay penalty. Following the logic of cost of delay discussed in Section 6.4, we use penalty 0.1 for the first 15 minutes of delay. Each additional minute of delay is penalised by 1. Penalty for each aircraft swap ranges from 8 to 23 in our settings. Please note that in case of one swap between two aircraft is this penalty accounted twice - once for each aircraft. Swap back to original rotation is not penalised. We use penalty of 150 for each cancelled leg in all three scenarios. Penalty for breaking crew connection ranges from 16 to 46 which is twice as much as swap penalty. Please note that this penalty is accounted for each broken connection regardless how many crew members follow this connection.

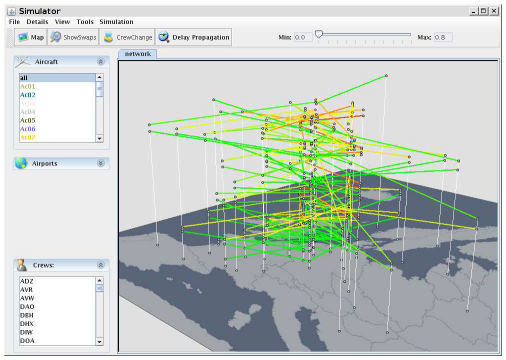
6.6 Visualisation

Ops Simulator is designed to visualise schedules in two or three dimensions. This turned out to be very useful for the analysis of robustness of the schedules and provided additional insight to recovery decisions and schedule structure.

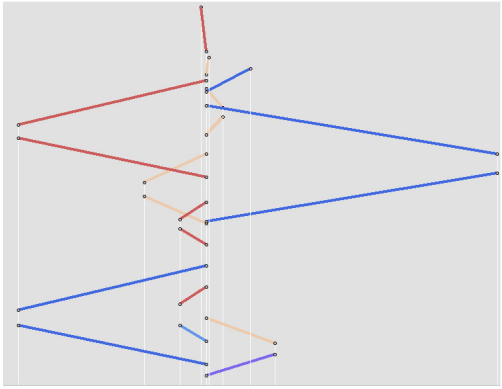
In the three-dimensional view, length and width represent geographical position, while height represents time. This view shows the structure of the



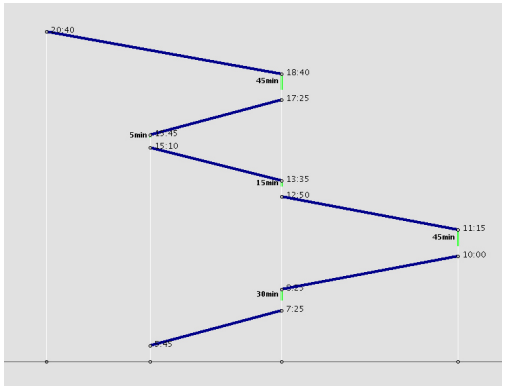
(a) 3D view of an instance. Each line represents a legs; colour of a leg indicates an aircraft covering the leg.



(b) 3D view of an instance. Colour represents probability of delay propagation to the leg (green/low, red/high).



(c) A detailed view of an airport. Only incoming and outgoing legs of the airport are displayed.



(d) Visualisation of a single aircraft rotation with information on departure time, arrival time and buffer time between the legs.

Figure 6.4: The screenshots of the graphical user interface of the Ops Simulator showing numerous uses.

network spread in time. On Figures 6.4a, 6.4b, or 5.1 you can see a screenshots of the three dimensional view.

In the two dimensional view, the horizontal direction represents projection of geographical position and the vertical direction represents time. See, for instance, Figures 6.4c and 6.4d.

The visualisation of the aircraft rotations, buffer times, and crew pairings is helpful for understanding the structure of the network, see Figure 6.4d. Together with visualisation of some robustness indicator, it is very useful for comparison of different schedules from the robustness point of view.

Ops Simulator offers visualisation of planned and operated legs. This provides insight into the understanding of recovery decisions, taken by the operators, when visualising historical schedules. If we visualise simulated schedules, it is easy to identify operations taken by the recovery module during the simulation, see Figure 6.3.

Ops Simulator can also picture some robustness indicators. Value of the indicator is assigned to each leg and is expressed by the colour. Green leg means low value, red stands for high value of the indicator. In our case, we display probability of delay propagation to the leg, see Figure 6.4c The value of the indicator is part of the input. Besides, the simulator can get estimates for the probability of delay propagation from the simulation. Comparison of the values computed by the optimiser and the values estimated by the simulation also brings insight into the accuracy of discretisation method.

Free-to-download version of Ops Simulator allowing visualisation of schedules and arbitrary measure on the legs is available on AirVis [6].

Chapter 7

Computational Results

In this chapter we present computational results for real-world large-scale tail assignment instances. We present results for both stochastic programming formulations RoTA and NoRoTA of the robust tail assignment defined in Chapter 4.

Both models minimise probability of delay propagation where RoTA model considers rotational propagated delay only, while NoRoTA model includes also non-rotational delay propagation due to the changing crew. In Chapter 4, we showed that both models can be solved by an extension of standard column generation framework for solving the tail assignment problem.

We present computational results for an extension of tail assignment optimiser NetLine/Ops Tail xOPT from the NetLine planning system of Lufthansa Systems. Our extension is compared to results of the same optimiser with no extensions where standard KPI objective function is optimised. This allows us to compare benefits of the stochastic programming approach over KPI approach and prove computational tractability of our approach at the same time.

Section 7.1 is dedicated to full description of the testing approach. Especially, in Section 7.1.2, we provide description of the KPI objective we benchmark RoTA and NoRoTA against. In Section 7.1.3, we introduce testing instances used throughout this chapter.

In Section 7.2 and Section 7.3 we give performance results of RoTa and NoRoTa model, respectively. We show robustness improvement and comparability of running times of our extension to the standard KPI approach.

Furthermore, in Section 7.4, we take a closer look on benefits of the stochastic programming approach in practise. We use simulation tool Ops Simulator in order to provide an estimation of the practical impact of the approach in means of monetary savings and decreased arrival delay. Moreover, we show superiority of our method for majority of disruption scenarios not only in average case. We also observe that the benefits of such robust schedules grow with amount of disruptions.

Section 7.5 goes even further in study of practical relevance of the results. We apply recovery tool Netline/Ops Solver xOPT during the simulation. This emulates actions taken by the operation control centre and make the simulation more lifelike. This, again, confirms benefits of the stochastic optimisation approach over KPI approach.

Finally, in Section 7.6, we benchmark results on historic disruptions in order to show that our approach indeed "captures reality". This questions our approach as whole together with the stochastic model of primary delays from Chapter 5.

7.1 Overview

7.1.1 Testing Methodology

As a basis for the implementation of RoTA and NoRoTA models, we take the commercial state-of-the-art tail assignment optimiser NetLine/Ops Tail xOPT from the NetLine planning system of Lufthansa Systems, see Schickinger (2008) [95].

We extend the column generation algorithm, which is part of the optimiser, in order to handle computation of distribution of propagated delay along rotations. For implementation of RoTA model, we use the standard column generation algorithm used in NetLine/Ops Tail xOPT with our pricing algorithm as describes Algorithm 5.

In case of NoRoTA model, we use adjusted column generation algorithm as states Algorithm 8. That means we use implementation of the pricing algorithm as describes Algorithm 7 and in addition, we use the method for updating column costs between the column generation iterations as states Algorithm 9.

Implementation of both models is based on the discretisation method introduced in Chapter 1. This method is extended for applications in airline scheduling in Chapter 3. In all experiments, we use a discretisation step size of one minute.

In order to get valuation of practical applicability of the models, we give a comparison of implementations of RoTA and NoRoTA models and standard tail assignment optimiser. We use the same optimiser with no extensions where simple KPI robustness objective is optimised. This KPI is described in detail in Section 7.1.2 for description of the KPI. Such comparison gives a fair benchmark of impact of our extensions on tractability of real-world instances while at the same time it documents benefits of the stochastic planning over current practise.

Throughout this chapter we extensively use simulation for documenting a practical benefits of our approach. We use a fast, event-based simulator Ops Simulator described in Chapter 6. The stochastic programming approach as same as simulation are using a stochastic model of disruptions presented in Chapter 5.

All computations are done on a 64-bit desktop PC with two Intel(R) Core(TM) 2 processors with 3.0 GHz and 8 GB of RAM memory, running openSuse Linux 11.2. Our code is implemented in `C++` and compiled using `g++` 3.4.

7.1.2 KPI Objective (ORC)

The KPI objective rewards ground buffer times between two successive legs in a rotation up to a maximum of fifteen minutes, longer buffer is not considered to be beneficial. This is a standard practitioner's approach; it aims at distributing ground buffers of reasonable length in a homogeneous way. We further referred to the KPI method as ORC (Ordinary Robustness Costs).

In order to ensure satisfactory time for preparation of the aircraft for the next leg, optimiser ensures minimum ground time between consecutive legs. However, sometimes a small violation is required in order to construct feasible routing. Such violation is handled in ORC objective by a penalty. We set this penalty to value ten times higher than the value of one minute of the buffer. This should prevent the optimiser from violating the minimum ground time when it is not necessary. This violation rarely exceeds ten minutes.

For solving the tail assignment problem with consideration of crew pairings, we, in addition, introduce a penalty for crew changing the aircraft. We set it to fifty times higher value than the bonus of one minute of buffer. It practically means that the aircraft follows crew as much as it is possible.

Parameter	Value
ground buffer (up to 15 minutes)	-1 per minute
minimum ground time violation penalty	10 per minute
crew change penalty	50 per crew change

Table 7.1: Key parameters of ORC objective.

By a crew change, we mean the situation where at least one crew member changes aircraft. It is not important how many crew members has to take the same aircraft change. However, if two crew members continue on two different aircraft, the penalty is accounted twice.

Presented settings are derived based on discussion with practitioners and make good sense in practise. Summary of key parameters of the KPI is listed in Table 7.1.

Just to remind, in RoTA and NoRoTA models, we do not guarantee minimum ground time between legs, since the violation of the minimum ground time is allowed and it is not penalised. However, every violation of the minimum ground time leads to delay propagation with high probability and thus increases value of objective function. Therefore such violation rarely lead to an optimal solution of the problem.The same applies also for crew changes which we also do not penalise explicitly.

7.1.3 Instances

In this chapter, we use testing data set from continental short-haul airline. Daily schedules are, therefore, always separated by a long overnight stay which prevents aircraft delay propagation between individual days. We do not consider maintenance rules in our experiments. Under these conditions, the optimisation of few days together produces the same results as their separate optimisation. Hence, we always optimise, simulate, and report results for

		Fleet F1					
		Aircraft			Legs		
	Days	Min	Max	Avg	Min	Max	Avg
January	20	20	24	22	82	153	119
February	22	19	24	22	88	160	136
March	26	18	25	22	88	168	140
April	16	17	25	23	64	174	143

		Fleet F2					
		Aircraft			Legs		
	Days	Min	Max	Avg	Min	Max	Avg
January	22	13	17	15	60	106	92
February	18	15	17	16	103	116	111
March	17	15	17	16	100	121	110
April	20	15	18	16	93	118	105

Table 7.2: Description of all considered monthly tail assignment instances. Minimum, maximum, and average number of legs and aircraft per one day.

one day instances. Results for monthly instances are therefore combination of results for single day instances.

Table 7.2 summarises basic information about our testing set. We study instances from January 2007 to April 2007 for two fleets. Due to some data inconsistencies we are not able to optimise every day in this time period. Thus, we work with 16 to 26 days in each month. Number of used aircraft and covered legs differs between days. Fleet F1 consists of 17 to 25 aircraft which cover from 64 to 174 legs. Fleet F2 is smaller and has 13 to 18 aircraft which cover 60 to 121 legs a day.

When optimising with crew we always cover every leg by three different crew member types: pilot, first officer, and cabin crew.

Some detailed experiments, as analysis of individual simulation runs or simulations under recovery, are very time consuming and cannot be performed

Instance	Crew Pairings	Aircraft	Legs	Avg rotation length
F1 20070110	No	21	128	6.1
F1 20070326	No	24	168	7
F1 20070404	Yes	24	165	6.9
F2 20070108	Yes	16	97	6.1
F2 20070128	Yes	17	96	5.6
F2 20070312	No	17	110	6.5

Table 7.3: Description of one day instances chosen for detailed analysis.

for whole dataset. We, therefore, restrict our dataset to six instances thereof three are used for experiments with and three without consideration of crew pairings. See their overview in Table 7.3

7.2 Performance of the RoTA Model

As first, let us take a look on performance of RoTA model in relation to ORC method (which uses no stochastic information). RoTA model minimises probability of delay propagation, in short PDP, see Section 4.2 for detailed description of this model.

Table 7.4 summarises results for all instances. The first two columns of the table identify the testing instances from Table 7.2. The columns labelled PDP show total average probability of delay propagation per one day instance. The columns labelled CPU list the average computation time per one day instance in CPU seconds. We state values for implementation of RoTA model on the left and for ORC on the right. Probability of delay propagation (PDP) of ORC optimised solutions is computed a posteriori, for the RoTA optimised schedules, it constitute the objective. Please notice that we do not state average values based on simulation since they do not differ from computed once. See Section 3.6 for discussion about accuracy of the method.

Last two columns of the table summarise the average improvement of PDP value both absolute and relative. We observed an average decrease of 0.8% to 3.0% of PDP for each monthly instance; impact of this improvement is

Instance		RoTA (PDP)		ORC		Improvement	
		PDP (opt)	CPU [s]	PDP (opt)	CPU [s]	PDP [#]	PDP [%]
F1	Jan	23.26	10	23.61	7	0.35	1.5
F1	Feb	32.41	11	32.76	9	0.35	1.1
F1	Mar	33.21	14	33.61	10	0.39	1.2
F1	Apr	34.12	16	34.67	12	0.55	1.6
F2	Jan	15.78	12	16.33	9	0.56	3.4
F2	Feb	24.70	14	25.18	10	0.48	1.9
F2	Mar	24.03	18	24.38	13	0.35	1.4
F2	Apr	17.16	17	17.69	11	0.54	3.0

Table 7.4: Comparison of the robustness of airline rotations using a KPI method ORC and a stochastic optimisation method (RoTA).

further studied in Section 7.4. You can also see that computation time for the PDP method is on average higher, but the increase is quite low.

7.3 Performance of the NoRoTA Model

Now, let us present the same comparison on these instances with inclusion of delay propagation along crew pairings. Again, we compare a stochastic approach that minimises total probability of delay propagation, see NoRoTA model in Section 4.3, with the KPI approach.

Table 7.5 documents the results for instances from Table 7.2 in similar structure as in Table 7.4. Column PDP shows an average probability of delay propagation per one day. As we explained in Section 3.5.4, discretisation method may under certain conditions produce inaccurate results. Hence, we present in PDP columns simulated values instead of optimised. Each value is computed as an average of 100,000 simulations. CPU column gives an average computation time in seconds. Column CC states number of crew changes appearing on average in one day schedule of the instance.

Instance		NoRoTA (PDP)			ORC			Improvement	
		PDP (sim)	CC [#]	CPU [s]	PDP (sim)	CC [#]	CPU [s]	PDP [#]	PDP [%]
F1	Jan	25.92	1.0	9	26.20	0.6	6	0.28	1.1
F1	Feb	35.83	0.8	10	36.10	0.7	7	0.27	0.7
F1	Mar	37.18	0.5	12	37.51	0.5	8	0.33	0.9
F1	Apr	38.00	1.6	14	38.55	1.4	9	0.55	1.4
F2	Jan	20.01	2.9	14	20.53	1.9	7	0.52	2.6
F2	Feb	29.80	3.0	16	30.19	2.4	7	0.39	1.3
F2	Mar	28.76	2.1	19	29.03	2.1	9	0.27	0.9
F2	Apr	21.75	3.8	20	22.25	3.1	6	0.5	2.3

Table 7.5: Comparison of the robustness of airline rotations for a KPI method ORC and a stochastic optimisation method NoRoTA.

Improvement provided by the PDP method is stated in the last two columns of the table. You can see that PDP method decreases probability of delay propagation from 0.7% to 2.6%. Computational effort of the PDP method are still very good and acceptable for real applications.

Moreover, you can see that PDP schedules encounter more crew changes than ORC schedules. This contradicts accepted logic that less crew changes lead to more robust schedule.

Instance	NoRoTA (PDP)		ORC		Improvement	
	PDP [#]	CC [#]	PDP [#]	CC [#]	PDP [#]	PDP [%]
F1 20070107	14.92	2	15.80	1	0.88	5.6
F1 20070114	21.39	3	21.52	2	0.13	0.6
F2 20070109	23.84	5	24.75	3	0.91	3.7
F2 20070131	25.56	7	25.67	5	0.11	0.4

Table 7.6: Probability of delay propagation and number of crew changes for chosen one day instances.

We give a short overview of instances supporting our observation in Table 7.6. We show few instances where PDP optimised schedules experience less delay propagation while having more scheduled crew changes. Please notice that presented instances are no rarities in any means. Just in January instances of both fleets, we identified 19 such cases out of 42.

7.4 Impact

So far, we presented gain of our approach in terms of decreased probability of delay propagation. Nevertheless, it still does not provide insight into practical gain of this method. One can hardly imagine what improvement of average number of delay propagations by, for example, 0.3% or 2% mean.

Moreover, it is unclear where this gain comes from. Our method may outperform ORC approach under all disruption scenarios or only in some special cases, for example, with very few disruptions. In this section we take a look on these two aspects.

7.4.1 Average Savings

Let us translate benefits of the stochastic optimisation method to arrival leg delay and estimated extra costs. For estimation of monetary savings, we use the cost model explained in Section 6.4 which is built in into Ops Simulator.

Presented measures are a direct output of the simulation. In order to obtain good estimates of the expected gain of the method we simulate schedules of every instance 100, 000 times. Please notice that it would be also possible to get these values directly from the optimiser.

Tables 7.7 and 7.8 summarise obtained results with and without consideration of crew pairings, respectively. Each value in the tables represents an average value per one day of the monthly instance. For both schedules RoTA/NoRoTA and ORC we show probability of delay propagation (in PDP column), we present average total arrival delay (in column AD), and average total extra expenses (in column EC). The last three columns show average improvement of the RoTA/NoRoTA models over ORC approach.

You can see that there is not direct correlation between PDP, arrival delay, and extra costs. This stems from non-linearity of cost model as well as

Instance	RoTA			ORC			Improvement		
	PDP [#]	AD [min]	EC [€]	PDP [#]	AD [min]	EC [€]	PDP [#]	AD [min]	EC [€]
F1 Jan	23.23	848	25,535	23.59	857	25,995	0.36	9	460
F1 Feb	32.38	1,083	34,388	32.74	1,092	34,802	0.36	9	414
F1 Mar	33.18	1,112	35,046	33.58	1,124	35,596	0.40	12	550
F1 Apr	34.09	1,169	36,705	34.64	1,184	37,437	0.55	15	732
F2 Jan	15.77	676	20,273	16.34	688	20,822	0.57	12	549
F2 Feb	24.67	916	29,342	25.14	926	29,795	0.47	10	453
F2 Mar	24.03	904	28,820	24.43	911	29,149	0.40	7	329
F2 Apr	17.14	779	23,658	17.68	790	24,164	0.54	11	506

Table 7.7: Comparison of practical benefits of the stochastic optimisation method against KPI method.

Instance	NoRoTA			ORC			Improvement		
	PDP [#]	AD [min]	EC [€]	PDP [#]	AD [min]	EC [€]	PDP [#]	AD [min]	EC [€]
F1 Jan	25.92	891	27,406	26.20	896	27,654	0.28	5	248
F1 Feb	35.83	1,138	36,788	36.10	1,143	36,975	0.27	5	187
F1 Mar	37.18	1,178	37,920	37.51	1,188	38,386	0.33	10	466
F1 Apr	38.00	1,238	39,750	38.55	1,253	40,477	0.55	15	727
F2 Jan	20.01	748	23,481	20.53	759	24,001	0.52	11	520
F2 Feb	29.80	1,011	33,686	30.19	1,019	34,035	0.39	8	349
F2 Mar	28.76	986	32,479	29.03	992	32,733	0.27	6	254
F2 Apr	21.75	858	27,179	22.25	870	27,739	0.50	12	560

Table 7.8: Comparison of practical benefits of the stochastic optimisation method against ORC method with consideration of crew parings

from the fact that delay of the length is not directly reflected by probability indicator.

Stochastic approach brings an average daily savings 400-700€ and decreases arrival delay by 7-15 minutes when crew pairings are not considered. In the optimisation with crew pairings, savings vary from 200€ to 700€ per day and we save 5-15 of delay on the arrival which corresponds to approximately 300,000€ per annum for studied fleets.

7.4.2 Analysis Of Individual Simulations

Knowing that the method saves delays and extra expenses on average is not everything one wants to know before deployment of the method. It is important to know that the method dominates in the most cases and not, for example, under special circumstances, while in the majority of cases performs worse. Such properties, of course, would make contribution of the method questionable.

Instance	RoTA/NoRoTA				ORC				Impr.	
	PDP [#]	AD [min]	EC [€]	CC [#]	PDP [#]	AD [min]	EC [€]	CC [#]	PDP [#]	AD [min]
F1 20070110	30.93	975	30,192	-	31.35	984	30,650	-	0.42	9
F2 20070312	19.14	806	24,034	-	20.13	828	25,078	-	0.99	22
F1 20070326	38.15	1,338	41,600	-	38.54	1,350	42,171	-	0.39	12
F2 20070108	23.86	847	27,557	4	24.51	853	27,780	2	0.65	6
F2 20070128	18.00	746	22,991	6	19.66	783	24,700	3	1.66	37
F1 20070404	45.20	1,434	46,542	2	46.26	1,466	48,056	1	1.06	32

Table 7.9: Comparison of schedules for one day instances chosen for detailed analysis based on 100,000 simulations without recovery

We deal with this question by studying six one day instances described in Table 7.3. For the first three of them, we do not consider crew pairings during the optimisation and simulation while for the other three we consider also crew. Table 7.9 summarises these instances in terms of probability of delay propagation (in column PDP), scheduled number of crew changes (in column CC), average arrival delay (in column AD), and extra expenses due to delays (in column EC). Last two columns show improvement of probability of delay

propagation (PDP) and arrival delay (AD) of RoTA and NoRoTA optimised schedules. All values are based on 100, 000 simulation runs.

Please notice that we have chosen on purpose range of instance with various average frequency of delay propagations and difference between quality of PDP and ORC optimised schedules.

Instance	All simulations		
	OP [%/%]	ΔAD [min]	ΔEC [€]
F1 20070110	40 / 30	8	391
F2 20070312	54 / 25	23	1,149
F1 20070326	43 / 29	15	792
F2 20070108	47 / 24	5	139
F2 20070128	77 / 10	36	1,667
F1 20070404	73 / 6	30	1,380

Table 7.10: Performance comparison of schedules under individual simulations.

Now, we compare results of 500 randomly generated simulations individually. Table 7.10 quantifies our finding based on 500 simulations. The first number in column OP gives the percentage of disruption scenarios where PDP outperforms ORC, the second number gives percentage where it is vice versa. The remaining percentage missing to 100% represents cases when both schedules perform equally. Column AD gives average saving in arrival delay and column EC gives estimated monetary savings. Notice that these numbers slightly differ from numbers in Table 7.9 due to lower sample size in our experiment.

PDP optimised schedules outperform ORC more often but further analysis shows that in simulations with low level of disruptions, both schedules perform similarly quite often. In order to make this differences more visible, we derive statistics only for "heavier" part of disruption scenarios. We take the half of the disruption scenarios with more total primary disruptions. Statistics for these disruption scenarios are given in Table 7.11. Increasing trend of impact of stochastic optimisation is observable in all tested instances and

Instance	High disruption simulations		
	OP [%/%]	ΔAD [min]	ΔEC [€]
F1 20070110	44 / 30	8	371
F2 20070312	60 / 27	31	1,585
F1 20070326	44 / 32	18	959
F2 20070108	46 / 22	5	152
F2 20070128	74 / 11	44	2,073
F1 20070404	71 / 6	37	1824

Table 7.11: Performance comparison of schedules under individual simulations with heavy disruptions.

average cost savings increase up to 50% in comparison to average savings over all disruption scenarios.

On Figure 7.1 we provide a graphical overview of these findings. Each point in the sub-graphs represents arrival delay of the ORC and the PDP optimised schedule for one simulation run. Points on the right of the diagonal represent simulation runs in which ORC optimised schedule outperformed PDP optimised schedule, points on the left of the diagonal represent scenarios where the opposite occurs. Simulations with less primary disruptions, and thus less arrival delays lie on the left down part of the diagonal.

As mentioned previously, scenarios with few disruptions often perform alike since few and short disruptions do not influence arrival and propagated delays much (points on the bottom left lie close to diagonal). However, as the amount of disruptions increases, variations in the performance of both schedules grow (points on the top right are much more spread). On graphs where PDP optimised schedule significantly outperforms ORC it is visible that majority of the points lie on the left from the diagonal.

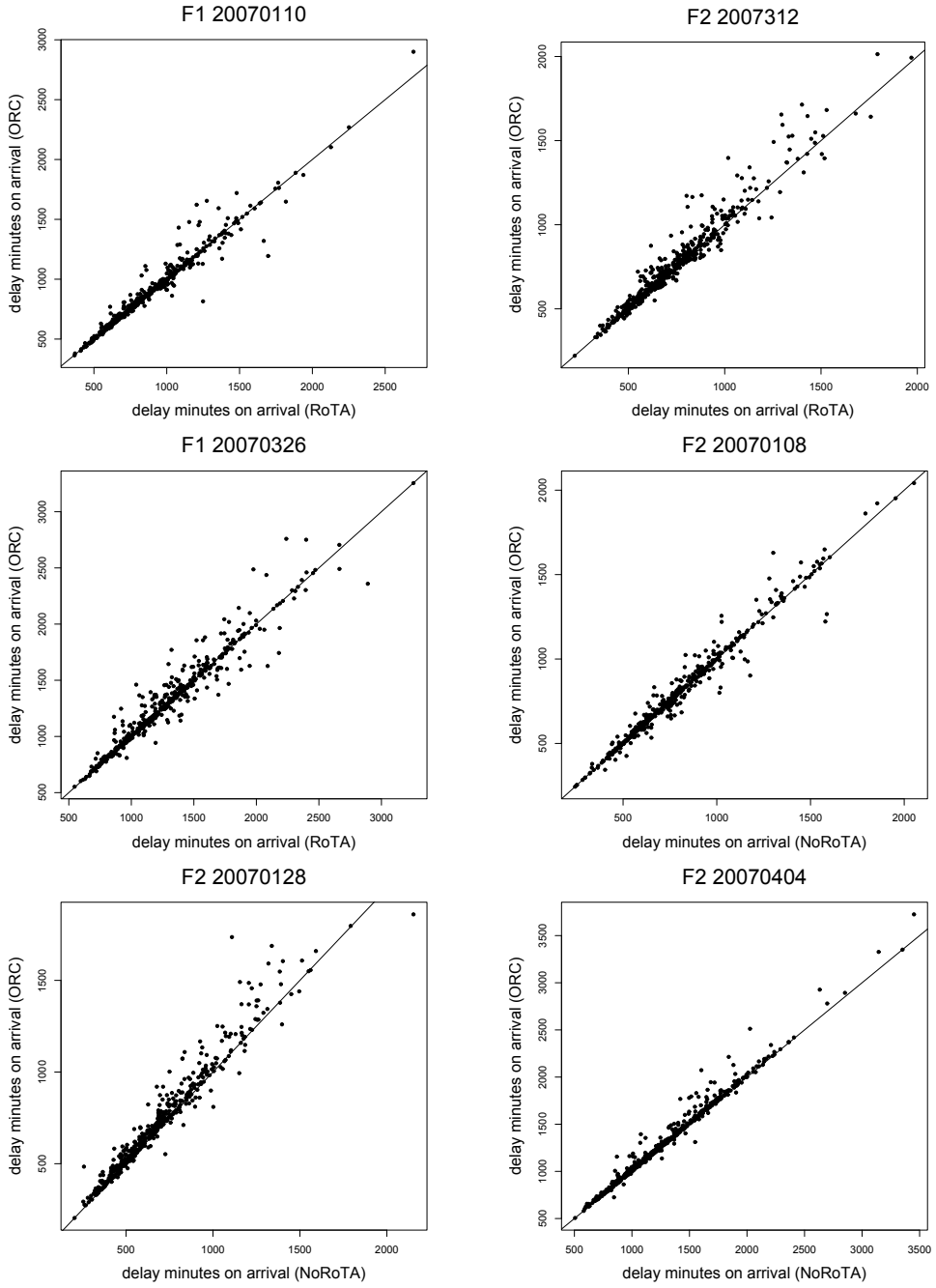


Figure 7.1: Plot of arrival delay minutes for 500 simulations of the ORC and the PDP optimised schedules of six testing instances.

7.5 Performance under Recovery

Clearly, the schedules are rarely operated as scheduled. Especially, in days with high level of primary disruptions, planners are forced to ad hoc changes in schedules during the day of operation. Naturally arises question how, so far, presented results transfer into real operation where recovery actions may be taken.

We examine performance of our schedules in simulation under recovery. We use the instances from Table 7.3 which we studied in Section 7.4.2 but without recovery. We us Ops Simulator, introduced in Chapter 6, with integrated recovery tool NetLine/Ops Solver xOPT. We use this tool to recover disturbed schedules in a realistic way. For more details about recovery in Ops Simulator see Section 6.5.

We use the same 500 randomly generated disruption scenarios as in previous section. For better understanding of recovery effects, we present results for three various settings of recovery tool from swap vulnerable (denoted as "easy") to very conservative recovery settings (denoted as "difficult"). We refer reader to Table 6.1 in Section 6.5.1 where important parameters of recovery algorithm are explained. For comparison, we show results of simulation without any recovery allowed, denoted as "forbidden".

	F1 20070110														
	RoTA					ORC					Improvement				
Recovery algorithm	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	ΔDP [#]	ΔEC [€]	ΔAD [min]	ΔSW [#]	ΔCL [#]
Easy	27.75	25,658	865	0.6	15	28.04	25,701	867	0.7	13	0.29	42	2	0.1	-2
Normal	27.77	25,923	869	0.4	10	28.10	26,020	873	0.4	10	0.33	97	3	0.1	0
Difficult	27.76	26,026	871	0.3	10	28.09	26,216	876	0.3	10	0.33	190	5	0.1	0
Forbidden	27.85	27,169	891	0	0	28.19	27,560	899	0	0	0.34	391	8	0	0

Table 7.12: Performance of ORC and RoTA optimised schedules for $F1$ 20070110 instance under recovery algorithms.

	F2 20070312														
	RoTA					ORC					Improvement				
Recovery algorithm	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	ΔDP [#]	ΔEC [€]	ΔAD [min]	ΔSW [#]	ΔCL [#]
Easy	16.65	20,177	708	0.8	7	17.28	20,403	714	1.1	3	0.64	227	6	0.3	-4
Normal	16.76	20,457	714	0.6	7	17.44	20,845	722	0.8	2	0.68	389	9	0.2	-5
Difficult	16.83	20,756	719	0.4	3	17.57	21,187	729	0.5	2	0.74	432	10	0.1	-1
Forbidden	16.96	22,062	742	0	0	17.94	23,212	766	0	0	0.98	1,149	23	0	0

Table 7.13: Performance of ORC and RoTA optimised schedules for $F2$ 20070312 instance under recovery algorithms.

	F1 20070326														
	RoTA					ORC					Improvement				
Recovery algorithm	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	ΔDP [#]	ΔEC [€]	ΔAD [min]	ΔSW [#]	ΔCL [#]
Easy	33.92	36,013	1,192	1.5	21	34.23	36,321	1,199	1.6	71	0.31	309	7	0.1	50
Normal	33.98	36,396	1,199	1.0	20	34.32	36,799	1,207	1.1	73	0.34	403	8	0.1	53
Difficult	34.06	36,785	1,206	0.8	20	34.35	37,013	1,212	0.9	53	0.30	228	5	0.1	33
Forbidden	34.32	40,865	1,278	0	0	34.81	41,658	1,294	0	0	0.50	792	15	0	0

Table 7.14: Performance of ORC and RoTA optimised schedules for $F1$ 20070326 instance under recovery algorithms.

	F2 20070108																
	NoRoTA						ORC						Improvement					
Recovery algorithm	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	BC [#]	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	BC [#]	ΔDP [#]	ΔEC [€]	ΔAD [min]	ΔSW [#]	ΔCL [#]	ΔBC [#]
Easy	21.74	24,501	766	2.2	27	0.1	22.23	24,455	768	2.1	24	0.1	0.49	-45	1	-0.1	-3	0.0
Normal	21.68	24,740	771	1.3	13	0.0	22.25	24,708	773	1.3	10	0.0	0.56	-32	2	-0.0	-3	0.0
Difficult	21.64	24,709	770	0.9	11	0.0	22.22	24,721	773	0.9	8	0.0	0.58	12	3	0.0	-3	0.0
Forbidden	21.66	25,422	783	0	0	0.0	22.21	25,561	788	0	0	0.0	0.55	139	5	0	0	0.0

Table 7.15: Performance of ORC and NoRoTA optimised schedules for $F2$ 20070108 instance under recovery algorithms.

	F2 20070128																	
	NoRoTA						ORC						Improvement					
Recovery algorithm	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	BC [#]	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	BC [#]	ΔDP [#]	ΔEC [€]	ΔAD [min]	ΔSW [#]	ΔCL [#]	ΔBC [#]
Easy	15.88	20,283	668	1.6	48	0.3	17.13	20,796	683	1.8	47	0.3	1.24	513	15	0.2	-1	0.0
Normal	15.89	20,336	670	1.1	43	0.3	17.19	21,060	688	1.2	44	0.3	1.29	724	18	0.1	1	0.0
Difficult	15.89	20,456	672	0.9	39	0.2	17.25	21,376	694	1.0	37	0.2	1.35	920	22	0.1	-2	0.0
Forbidden	15.90	21,238	686	0	0	0.0	17.44	22,905	722	0	0	0.0	1.54	1,667	36	0	0	0.0

Table 7.16: Performance of ORC and NoRoTA optimised schedules for $F2$ 20070128 instance under recovery algorithms.

	F1 20070404																	
	NoRoTA						ORC						Improvement					
Recovery algorithm	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	BC [#]	DP [#]	EC [€]	AD [min]	SW [#]	CL [#]	BC [#]	ΔDP [#]	ΔEC [€]	ΔAD [min]	ΔSW [#]	ΔCL [#]	ΔBC [#]
Easy	40.82	40,540	1,282	2.0	203	1.2	41.83	41,247	1,300	2.0	216	1.1	1.01	707	18	0.1	13	-0.1
Normal	40.70	40,779	1,286	1.3	154	0.9	41.70	41,432	1,303	1.4	159	0.9	1.00	653	17	0.1	5	0.0
Difficult	40.90	41,979	1,309	1.1	80	0.5	41.87	42,737	1,328	1.1	90	0.4	0.97	757	19	0.0	10	0.0
Forbidden	40.45	41,990	1,308	0	0	0.0	41.57	43,370	1,338	0	0	0.0	1.12	1,380	30	0	0	0.0

Table 7.17: Performance of ORC and NoRoTA optimised schedules for $F1$ 20070404 instance under recovery algorithms.

Tables 7.12, 7.13, and 7.14 show results for instances without consideration of crew pairings. In each table, we state average number of delay propagations (in column DP), average extra costs (EC), average number of arrival minutes (in column AD), and average count of performed swaps (in column SW) per one simulation based on 500 simulation runs. Since the number of leg cancellations is very low, we do not show an average but total number of cancelled legs in 500 simulations in column CL. Tables show all these values for RoTA and ORC schedules as well as their difference.

You can see that as recovery actions are getting easier to take - from no recovery at all (row Forbidden) through Difficult to Easy settings - overall delay costs are decreasing. Furthermore, benefit of stochastic approach decreases in terms of lower gap in delay propagation (column ΔDP), extra costs (column ΔEC), and arrival delay (column ΔAD). On the other hand amount of operational changes and gap in terms of more swaps (column ΔSW) is growing. Cancellations are very rare, and we do not observe any trend in their development.

Please notice that in simulations of instance $F1$ 20070326 in Table 7.14, we recorded much higher number of cancellations in case of ORC optimised schedule. This, obviously, significantly improves performance in terms of lower delay propagation, arrival delay, and extra costs (not including cancellation costs).

Tables 7.15, 7.16, and 7.17 present the same results for instances with consideration of crew pairings. In addition to aforementioned columns, we state average number of broken crew connections by recovery in column BC. You can observe the same conclusions as for the the previous instance. Here, number of broken crew connections is growing together with increasing number of swaps.

Simulation of instance $F1$ 20070404 in Table 7.17 shows interesting behaviour where average number of delay propagations grows with employment of recovery tool. This can be explained by the fact that objective of recovery tool favours two shorter delays to one longer. Nevertheless, you can see continuous decrease of arrival delay and extra expenses.

7.6 Proof of Concept

7.6.1 Stochastic Model

All hitherto presented results depend on the correctness of our stochastic optimisation model which is used in RoTA and NoRoTA models as well as in the simulator. In order to verify the superiority of our approach over KPI approach independently of the stochastic model, we run a direct simulation on the basis of historic data. That is, we do not randomly generate primary delays but we use historical primary delays as recorded in our data.

	RoTA			ORC			savings		
Instance	DP [#]	AD [min]	EC [€]	DP [#]	AD [min]	EC [€]	DP [#]	AD [min]	EC [€]
F1 Jan	468	27,878	1,182,422	477	28,844	1,237,338	9	966	54,916
F1 Feb	740	36,492	1,523,586	757	37,013	1,559,979	17	521	36,393
F1 Mar	1,009	47,257	1,951,834	1042	48,699	2,013,555	33	1,442	61,721
F1 Apr	464	17,238	591,553	467	17,080	589,207	3	-158	-2,346
F2 Jan	502	46,071	2,154,893	510	47,289	2,213,557	8	1,218	58,664
F2 Feb	444	22,159	892,808	442	22,423	905,591	-2	264	12,783
F2 Mar	519	32,312	1,416,447	541	33,402	1,471,276	22	1,090	54,829
F2 Apr	335	20,560	781,192	347	20,969	800,461	12	409	19,269

Table 7.18: Performance of ORC and PDP schedules under historical disruptions without consideration of crew pairings.

Table 7.18 and 7.19 summarise our experiment where ORC and PDP optimised schedules are evaluated with respect to the historic disturbances as observed on the respective days of operation. It means that each daily instance is simulated only once.

The first column of the tables identifies the tested instance. Columns labelled DP and AM give the number of delay propagations and arrival delays in minutes that would have resulted in the historic situation. Columns EC state estimated extra costs incurred during the tested month. The last three columns give savings that would PDP schedules bring over ORC schedules.

	NoRoTA			ORC			savings		
Instance	DP	AD	EC	DP	AD	EC	DP	AD	EC
	[#]	[min]	[€]	[#]	[min]	[€]	[#]	[min]	[€]
F1 Jan	594	40,195	1,818,697	624	42,132	1,916,175	30	1,937	97,478
F1 Feb	787	38,087	1,622,836	805	39,730	1,713,965	18	1,643	91,129
F1 Mar	1,086	51,999	2,216,132	1,128	52,878	2,255,378	42	879	39,246
F1 Apr	547	20,086	732,783	574	20,751	764,157	27	665	31,374
F2 Jan	589	52,557	2,509,363	608	55,859	2,690,096	19	3,302	180,733
F2 Feb	508	25,121	1,053,294	517	25,403	1,064,722	9	282	11,428
F2 Mar	617	37,017	1,646,459	627	38,241	1,712,781	10	1,224	66,322
F2 Apr	461	21,624	810,846	474	22,909	875,994	13	1,285	65,148

Table 7.19: Performance of ORC and PDP schedules under historical disruptions with consideration of crew pairings.

You can see that the PDP schedules, again, outperform the ORC schedules only with few exceptions where the difference in favour of ORC is rather small.

Chapter 8

Perspectives

We see a huge potential in application of introduced discretisation method in various transportation problems which are being solved by the column generation algorithm. Our approach is useful throughout various stages of planning and scheduling in order to gain, or ensure, wished robustness.

In case of airline planning we see possible application starting with assessment of robustness of various fleeting options and continuing in crew pairing and maintenance routing. Such approach would boost overall robustness of schedules immensely.

Just to document this potential let us come back to our experiments. We already showed potential in robustness improvement by $1\% - 3\%$ when measured by probability of delay propagation. In time when the tail assignment problem is solved, pairings are already fixed. Obviously, pairings impose significant limits for potential increase of robustness. Minimisation of total delay propagation forces aircraft routes to follow the pairing in the most of the cases. Clearly, when pairings are built without consideration of the robustness, which is the case in our data set, we have limited possibilities to improve total robustness of the schedule at tail assignment stage of planning. We believe that robust construction of the crew pairings could bring overall robustness to values where we observe it now when not considering crew. Tables 7.4 and 7.5 show difference in robustness by more than 10% which approves clear potential for robustness increase when our method applied at crew pairing construction.

We also believe that introduced testing framework based on stochastic model and simulation is an important part of any research focusing on robustness. It turned out to be a powerful tool for interpretation of results. Without this tool we would not be able to understand that improvement in orders of few percent already brings real impact in practice. Simulation is also useful for study and understanding of roots of bad performance and high delays in operation.3.

Bibliography

[1] *NIST/SEMATECH e-Handbook of Statistical Methods*, 2010. http://www.itl.nist.gov/div898/handbook/.

[2] J. Abara. *Applying integer linear programming to the fleet assignment problem.* Interfaces 19(4):20–28, 1989.

[3] S. AhmadBeygi, A. Cohn, Y. Guan & P. Belobaba. Analysis of the potential for delay propagation in passenger airline networks. Technical report, Sloan Industry StudiesWorking Paper Series, 2007.

[4] S. AhmadBeygi, A. Cohn & M. Lapp. *Decreasing airline delay propagation by re-allocating scheduled slack.* In *AGIFORS*, 2008.

[5] S. AhmadBeygi, A. Cohn & M. Weir. *An integer programming approach to generating airline crew pairings.* Computers & Operations Research 36(4):1284 – 1298, 2009. ISSN 0305-0548.

[6] AirVis. *www.zib.de/airvis.*

[7] M. F. Argüello, J. F. Bard & G. Yu. *A grasp for aircraft routing in response to groundings and delays.* Journal of Combinatorial Optimization 1:211–228, 1997. ISSN 1382-6905.

[8] M. Arikan, V. Deshpande, & M. Sohoni. *Building reliable air-travel infrastructure using stochastic models of airline networks.* AGIFORS , 2010.

[9] R. D. Badinelli. *Approximating probability density functions and their convolutions using orthogonal polynomials.* European Journal of Operational Research 95(1):211–230, November 1996.

[10] M. Ball, C. Barnhart, G. L. Nemhauser & R. Odoni. *Transportation*, vol. 14 of *Handbooks in Operations Research and Management Science.* North Holland, 2006.

[11] M. Ball, C. Barnhart, M. Dresner, M. Hansen, K. Neels, A. Odoni, E. Peterson, L. Sherry, A. Trani & B. Zou. *Total Delay Impact Study A Comprehensive Assessment of the Costs and Impacts of Flight Delay in the United States*, October 2010.

[12] C. Barnhart, N. L. Boland, L. W. Clarke, E. L. Johnson, G. L. Nemhauser & R. G. Shenoi. *Flight string models for aircraft fleeting and routing.* Transportation Science 32(3):208–220, 1998.

[13] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh & P. H. Vance. *Branch-and-price: Column generation for solving huge integer programs.* Operations Research 46:316–329, 1998.

[14] C. Barnhart, T. S. Kniker & M. Lohatepanont. *Itinerary-based airline fleet assignment*. Transportation Science 36(2):199–217, 2002.

[15] C. Barnhart, P. Belobaba & R. Odoni. *Applications of operations research in the air transport industry*. Transportation Science 37: 368 – 391, 2003.

[16] C. Barnhart, A. Farahat & M. Lohatepanont. *Airline fleet assignment with enhanced revenue modeling*. Operations Research 57 (1):231–244, 2009.

[17] P. Belobaba, R. Odoni & C. Barnhart, (Eds.). *The Global Airline Industry*. John Wiley & Sons, 2009.

[18] A. Ben-Tal & A. Nemirovski. *Robust convex optimization*. Mathematical Operations Research 23(4):769–805, 1998. ISSN 0364-765X.

[19] A. Ben-Tal & A. Nemirovski. *Robust solutions of uncertain linear programs.* Operations Research Letters 25:1–13, 1999.

[20] I. Bertsimas & M. Sim. *Robust discrete optimization and network flows*. Mathematical Programming Series B 98:49–71, 2003.

[21] I. Bertsimas & M. Sim. *The price of robustness*. Operations Research 52(1):35–53, 2004.

[22] F. Bian, E. Burke, S. Jain, G. Kendall, G. Koole, J. L. Silva, J. Mulder, M. Paelinck, C. Reeves, I. Rusdi & M. Suleman. *Measuring the robustness of airline*, 2003.

[23] J. R. Birge & F. Louveaux. *Introduction to Stochastic Programming*. Springer Series in Operations Research and Financial Engineering. Springer, 2000.

[24] S. Bisaillon, J.-F. Cordeau, G. Laporte & F. Pasin. *A large neighbourhood search heuristic for the aircraft and passenger recovery problem*, April 2010.

[25] R. Borndörfer, U. Schelten, T. Schlechte & S. Weider. *A column generation approach to airline crew scheduling*. In H.-D. Haasis, H. Kopfer & J. Schönberger, (Eds.), *Operations Research Proceedings 2005*, vol. 2005 of *Operations Research Proceedings*, pp. 343–348. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-32539-0.

[26] R. Borndörfer, I. Dovica, I. Nowak & T. Schickinger. *Robust tail assignment*. In *AGIFORS*, 2010.

[27] G. Boydell. Standard inputs for eurocontrol cost benefit analyses. Technical report, EUROCONTROL, 2005.

[28] S. Bratu & C. Barnhart. *Flight operations recovery: New approaches considering passenger recovery.* Journal of Scheduling 9(3): 279–298, 2006. ISSN 1094-6136.

[29] E. K. Burke, P. D. Causmaecker, G. D. Maere, J. Mulder, M. Paelinck & G. V. Berghe. *A multi-objective approach for robust airline scheduling.* Computers & Operations Research 37(5):822 – 832, 2010. ISSN 0305-0548.

[30] J. Burt, John M. & M. B. Garman. *Conditional monte carlo: A simulation technique for stochastic network analysis.* MANAGEMENT SCIENCE 18(3):207–217, 1971.

[31] A. Caprara, L. Galli, S. Stiller & P. Toth. Recoverable-robust platforming by network buffering. Technical report, ARRIVAL Project, 2008.

[32] J.-P. Clarke, T. Melconian, E. Bly & F. Rabbani. *Means-mit extensible air network simulation.* Simulation 83(5):385–399, 2007. ISSN 0037-5497.

[33] L. Clarke, E. Johnson, G. L. Nemhauser & Z. Zhu. *The aircraft rotation problem.* Annals of Operations Reserach 69:33–46, 1997.

[34] A. Clauset, C. R. Shalizi & N. M. E. J. *Power-law distributions in empirical data.* arXiv:0706.1062v2, June 2009.

[35] CODA. *http://www.eurocontrol.int/coda.*

[36] CODA. *Delay to air transport in europe*, 2009.

[37] A. M. Cohn & C. Barnhart. *Improving crew scheduling by incorporating key maintenance routing decisions.* Operations Research 51: 387–396, 2003.

[38] A. Cook, G. Tanner & S. Anderson. Evaluating the true cost to airlines of one minute of airborne or ground delay. Technical report, EUROCONTROL, 2004.

[39] J.-F. c. Cordeau, G. Stojković, F. c. Soumis & J. Desrosiers. *Benders decomposition for simultaneous aircraft routing and crew scheduling.* Transportation Science 35(4):375–388, 2001.

[40] P. R. Day & D. M. Ryan. *Flight Attendant Rostering for Short-Haul Airline Operations.* Operations Research 45(5):649–661, 1997.

[41] G. Desaulniers, J. Desrosiers, Y. Dumas, S. Marc, B. Rioux, M. Solomon & F. Soumis. *Crew pairing at air france.* European Journal of Operational Research 97(2):245 – 259, 1997. ISSN 0377-2217.

[42] G. Desaulniers, J. Desrosiers & M. M. Solomon, (Eds.). *Column Generation*. Springer-Verlag New York Inc., 2005.

[43] B. Dodin. *Approximating the distribution functions in stochastic networks*. Computers & Operations Research 12(3):251–264, 1985.

[44] B. Dodin. *Bounding the project completion time distribution in pert networks*. Operations Research 33(4):862–881, 1985.

[45] V. Dück. *Increasing Stability of Aircraft and Crew Schedules*. PhD thesis, University Paderborn, 2010.

[46] M. Dunbar, G. Froyland & C.-L. Wu. *Robust airline schedule planning: Minimizing propagated delay in an integrated routing and crewing framework*, March 2010.

[47] N. Eggenberg & M. Salani. Uncertainty feature optimization for the airline scheduling problem. Technical report, 2009.

[48] M. Ehrgott & D. M. Ryan. *Constructing robust crew schedules with bicriteria optimization*. Journal of Multi-Criteria Decision Analysis 11 (3):139–150, 2002.

[49] A. Eiger, P. B. Mirchandani & H. Soroush. *Path preferences and optimal paths in probabilistic networks*. Transportation Science 19: 75–84, 1985.

[50] L. El Ghaoui & H. Lebret. *Robust solutions to least-squares problems with uncertain data*. SIAM Journal on Matrix Analysis and Applications 18(4):1035–1064, 1997. ISSN 0895-4798.

[51] Eurocontrol. Costs of air transport delay in europe. Technical report, EUROCONTROL, 2000.

[52] Eurocontrol STATFOR. *Eurocontrol long-term forecast: Ifr flight movements 2010-2030*, December 2010.

[53] Y. Y. Fan, R. E. Kalaba & J. E. Moore II. *Arriving on time*. Journal of Optimization Theory and Applications 127:497–513, 2005.

[54] M. Fischetti & M. Monaci. Light robustness. Research paper, ARRIVAL project, 2008.

[55] M. Fischetti, D. Salvagnin & A. Zanette. *Fast approaches to improve the robustness of a railway timetable*. Transportation Science 43:321–335, 2009.

[56] H. Frank. *Shortest paths in probabilistic graphs*. Operations Research 17:583–599, 1969.

[57] B. Fuhr. *Robust flight scheduling - an analytic approach to performance evaluation and optimization*. In *AGIFORS*, 2007.

[58] D. R. Fulkerson. *Expected critical path lengths in pert networks.* Operations Research 10(6):808–817, 1962.

[59] M. Gamache, F. Soumis, G. Marquis & J. Desrosiers. *A column generation approach for large-scale aircrew rostering problems.* Operations Research 47(2):247–262, 1999. ISSN 0030-364X.

[60] C. Gao, E. L. Johnson & B. C. Smith. *Integrated airline fleet and crew robust planning.* Transportation Science 43:2–16, 2009.

[61] M. Grönkvist. *The Tail Assignment Problem.* PhD thesis, Göteborg university, 2005.

[62] C. A. Hane, C. Barnhart, E. L. Johnson, R. E. Marsten, G. L. Nemhauser & G. Sigismondi. *The fleet assignment problem: Solving a large-scale integer program.* Mathematical Programming 70: 211–232, 1995. ISSN 0025-5610.

[63] K. L. Hoffman & M. Padberg. *Solving airline crew scheduling problems by branch-and-cut.* Management Science 39(6):657–682, 1993.

[64] M. Jetzki. *The propagation of air transport delays in Europe.* PhD thesis, RWTH AACHEN UNIVERSITY, 2009.

[65] D. Klabjan. *Column Generation*, chap. Large-Scale Models in the Airline Industry, pp. 163–195. Springer US, 2005. ISBN 978-0-387-25486-9.

[66] D. Klabjan, E. L. Johnson, G. L. Nemhauser, E. Gelman & S. Ramaswamy. *Solving large airline crew scheduling problems: Random pairing generation and strong branching.* Computational Optimization and Applications 20:73–91, 2001. ISSN 0926-6003. 10.1023/A:1011223523191.

[67] D. Klabjan, E. L. Johnson, G. L. Nemhauser, E. Gelman & S. Ramaswamy. *Airline crew scheduling with regularity.* Transportation Science 35(4):359–374, 2001.

[68] G. B. Kleindorfer. *Bounding distributions for a stochastic acyclic network.* Operations Research 19(7):1586–1601, 1971.

[69] A. J. Kleywegt, A. Shapiro & T. Homem-de Mello. *The sample average approximation method for stochastic discrete optimization.* SIAM Journal on Optimization 12:479–502, 2001.

[70] N. Kohl & S. E. Karisch. *Airline crew rostering: Problem types, modeling, and optimization.* Annals of Operations Research 127:223–257, 2004. ISSN 0254-5330.

[71] N. Kohl, A. Larsen, J. Larsen, A. Ross & S. Tiourine. *Airline disruption management - perspectives, experiences and outlook*. Journal of Air Transport Management 13(3):149–162, May 2007.

[72] A. R. Krommer & C. W. Ueberhuber. *Computational Integration*. Society for Industrial Mathematics, 1987.

[73] A. R. Krommer & C. W. Ueberhuber. *Numerical Integration on Advanced Computer Systems*. Springer-Verlag, 1994.

[74] L. Kroon, R. Dekker, G. Maroti, M. R. Helmrich & M. Vromans. Stochastic improvement of cyclic railway timetables. Technical report, ARRIVAL, 2007.

[75] S. Lan. *Planning for robust airline operations : Optimizing aircraft routings and flight departure times to achieve minimum passenger disruptions*. PhD thesis, Massachusetts Institute of Technology, Dept. of Civil and Environmental Engineering., 2003.

[76] S. Lan, J.-P. Clarke & C. Barnhart. *Planning for robust airline operations: Optimizing aircraft routings and flight departure times to minimize passenger disruptions*. Transportation Science 40(1):15–28, February 2006.

[77] L. Lettovsky, E. L. Johnson & G. L. Nemhauser. *Airline crew recovery*. Transportation Science 34(4):337–348, 2000.

[78] C. Liebchen, M. Lübbecke, R. H. Möhring & S. Stiller. Recoverable robustness. Technical report, ARRIVAL-Project, August 2007.

[79] M. Lohatepanont & C. Barnhart. *Airline schedule planning: Integrated models and algorithms for schedule design and fleet assignment.* Transportation Science 38:19–32, 2004.

[80] R. P. Loui. *Optimal paths in graphs with stochastic or multidimensional weights*. Communications of the ACM 26(9):670–676, 1983. ISSN 0001-0782.

[81] D. G. Malcolm, J. H. Roseboom, C. E. Clark & W. Fazar. *Application of a technique for research and development program evaluation*. Operations Research 7(5):646–669, 1959.

[82] T. Melconian & J.-P. Clarke. Effects of increased nonstop routing on airline cost and profit. Technical report, MIT, 2001.

[83] A. Mercier, J.-F. Cordeau & F. Soumis. A computational study of benders decomposition for the integrated aircraft routing and crew scheduling problem. Technical report, Les Cahiers du GERAD, 2004.

[84] I. Murthy & S. Sarkar. *A relaxation-based pruning technique for a class of stochastic shortest path problems.* Transportation Science 30 (3):220–236, 1996.

[85] E. Nikolova, M. Brand & D. R. Karger. *Optimal route planning under uncertainty.* In *International Conference on Automated Planning and Scheduling*, 2006.

[86] D. O'Connor. *Exact and approximate distributions of stochastic pert networks.* 2006.

[87] N. Papadakos. *Integrated airline scheduling.* Computers & Operations Research 36(1):176–195, 2009. ISSN 0305-0548.

[88] J. D. Petersen, G. Sölveling, , E. L. Johnson, J.-P. Clarke & S. Shebalov. *An optimization approach to airline integrated recovery.* In *AGIFORS*, 2010.

[89] R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[90] B. Rexing, C. Barnhart, T. Kniker, A. Jarrah & N. Krishnamurthy. *Airline fleet assignment with time windows.* Transportation Science 34(1), 2000.

[91] J. M. Rosenberger, A. J. Schaefer, D. Goldsman, E. L. Johnson, A. J. Kleywegt & G. L. Nemhauser. *A stochastic model of airline operations.* Transportation Science 36(4):357–377, 2002.

[92] J. M. Rosenberger, E. L. Johnson & G. L. Nemhauser. *A robust fleet-assignment model with hub isolation and short cycles.* Transportation Science 38(3):357–368, 2004. ISSN 1526-5447.

[93] A. Sarac, R. Batta & C. M. Rump. *A branch-and-price approach for operational aircraft maintenance routing.* European Journal of Operational Research 175(3):1850 – 1869, 2006. ISSN 0377-2217.

[94] A. Schaefer, E. L. Johnson, A. Kleywegt & G. L. Nemhauser. *Airline crew scheduling under uncertainty.* Transportation Science 39(3):340–348, 2005.

[95] T. Schickinger. *The potential of tail assignment optimization.* In *AGIFORS Airline Operations*, 2008.

[96] S. Shebalov & D. Klabjan. *Robust airline crew pairing: Move-up crews.* Transportation Science 40:300–312, 2006.

[97] A. W. Shogan. *Bounding distributions for a stochastic pert network.* Networks 7(4):359–381, 1977. ISSN 1097-0037.

[98] C. E. Sigal, A. A. B. Pritsker & J. J. Solberg. *The stochastic shortest route problem.* Operations Research 28:1122–1129, 1980.

[99] M. Sim. *Robust Optimization.* PhD thesis, Massachusetts Institute of Technology, Sloan School of Management, 2004.

[100] SimAir. *http://www.ise.nus.edu.sg/project/simair/.*

[101] B. C. Smith & E. L. Johnson. *Robust airline fleet assignment: Imposing station purity using station decomposition.* Transportation Science 40(4):497–516, November 2006.

[102] G. K. Smyth. *Numerical integration.* In *In Encyclopedia of Biostatistics*, pp. 3088–3095, 1998.

[103] M. Sohoni, Y. C. Lee & D. Klabjan. *Block time estimation and robust airline scheduling.* In *AGIFORS*, 2008.

[104] A. L. Soyster. *Convex programming with set-inclusive constraints and applications to inexact linear programming.* Operations Research 21(5):1154–1157, 1973. ISSN 0030364X.

[105] H. Srivastava & R. Buschman. *Theory and applications of convolution integral equations.* Mathematics and its applications. Kluwer Academic Publishers, 1992. ISBN 9780792318910.

[106] S. Stiller. *Extending Concepts of Reliability.* PhD thesis, TU Berlin, 2008.

[107] B. Tam, M. Ehrgott & G. Zakeri. *A comparison of stochastic programming and bi-objective optimisation approaches to robust airline crew scheduling.* OR Spectrum pp. 1–27, 2009. ISSN 0171-6468.

[108] O. Weide. *Robust and Integrated Airline Scheduling.* PhD thesis, Department of Engineering Science, School of Engineering, The University of Auckland, 2009.

[109] J. W. Yen & J. R. Birge. *A stochastic programming approach to the airline crew scheduling problem.* Transportation Science 40(1):3–14, 2006.

[110] G. Yu. *Operations Research in the Airline Industry.* Kluwer Academic Publishers, 1997.

[111] G. Yu, M. Argüello, G. Song, S. M. McCowan & A. White. *A new era for crew recovery at continental airlines.* Interfaces 33:5–22, 2003.