ASSIGNING INBOUND FLIGHTS TO BAGGAGE RETRIEVAL BELTS

A simulation study on the impact of stochasticity on the performance of different assignment heuristics

July 2020

Author:

Leo van Zadelhoff

l.d.vanzadelhoff@student.utwente.nl

Supervisors:

Dr. ir. E.A. Lalla,	University of Twente
Dr. ir. M.R.K. Mes	University of Twente
Dr. M. Verbin	ORTEC B.V.

UNIVERSITY OF TWENTE.

ORTEC
OPTIMIZE YOUR WORLD

Preface

For over the last six months I have been working on this thesis, and its completion will mark the end of my time as a student. The first months I started at ORTEC's Zoetermeer office, where I could always count on everyone at ORTEC's Centre of Excellence to provide me with help on practical issues and with insights on possibilities for this thesis. Later, due to the Dutch 'Intelligent Lockdown', my home became my office. Even though there were a lot less flexi-workspaces and my coffee was downgraded to Senseo coffee, I was lucky to still be surrounded by people with similar backgrounds in studies, always willing to help.

From the University of Twente, I would like to thank Eduardo for his guidance throughout the whole project, and for always providing extensive feedback on my work, both on the structure and on the contents. I would also like to thank Martijn, for his thorough feedback and encouraging me to implement extra approaches. You both put a lot of your valuable time in providing me feedback, and allowed me to send my work quite shortly before our meetings, which had a positive impact on my workflow. I am sure that the quality of this thesis has improved substantially due to you.

At ORTEC, many people have contributed to my research some way or another, but special thanks go out to Mor, who has always helped me give direction to the project. You had answers to most of my questions, and if you did not, you made sure you would get these answers as soon as possible. It was a great pleasure working with you.

I would also like to thank my friends, who motivated me to push my boundaries throughout my entire studies, and everyone in the so-called 'Master Corner' at the university for being great study and project companions. Finally, I would like to thank my family and my girlfriend for their love and unconditional support.

Leo van Zadelhoff

Management Summary

In this thesis we analyze the assignment of flights to baggage belts for a large European airport, on behalf of ORTEC. First we analyze the data, the airports objectives for the solution, and the current solution approach. Then, we formulate alternative solution approaches, and compare their performance, based on both current data quality and improved data quality.

If the baggage of two flights (a flight pair) is present on the baggage retrieval belts at the same time, we say the flight pair has overlapping on-belt time windows. These on-belt time windows are based on the expected arrival time of a flight at the airport, the expected time it takes to move the baggage from the aircraft to the baggage retrieval belt, and the expected time a flight's baggage remains on the baggage retrieval belt. One of the objectives of the flight-to-belt assignment is to minimize the total time that overlapping flights are assigned to the same baggage retrieval belt, by only changing the belt to which a flight is assigned, and not by changing the time at which a flight's baggage is put on the belt.

From a literature review, we find that this problem can be formulated as a maximization variant of the Process Allocation Problem, which is a special case of the Generalized Quadratic Assignment Problem. Currently, literature only reports on solution approaches for the minimization version of the problem. These solution approaches focus primarily on clustering, but clustering approaches were found to be impractical for the airport's problem.

The airport uses a Simulated Annealing (SA) approach to assign flights to baggage belts. Every ten minutes, a problem instance is created based on the estimated times for the upcoming three hours of flights, and for this problem instance all flights are assigned to a belt using SA. When a flight arrives, the assignments of the latest completed SA run are retrieved, and the flight is conclusively assigned to the belt from that assignment. This is called Rolling Horizon Optimization.

It was not known how well this SA algorithm performs with respect to the optimal solutions of the problem instances based on estimates, nor on the realized times. After creating a mathematical model formulation of the problem and being able to solve some problem instances to optimality, we found that the SA structurally came very close to, and often matched, the optimal solutions for the problem instances using the estimates.

We also found that there was a large difference in a schedules expected performance (based on up-front time estimates) and the realized performance of the schedules (based on the realized times). This is primarily due to the fact that 40% of the flight pairs that ended up overlapping were not estimated to overlap, and out of all flight pairs that were estimated to overlap, only 47% did end up overlapping. This problem was found to be caused by large standard deviations, and often bias, in the estimators used

to determine the time windows in which flights are on-belt, and the stochastic behavior not being incorporated in the solution approach.

Three methods were implemented to deal with the problem of stochasticity:

- First, a solution approach was designed that assigns each flight upon arrival, instead of creating schedules every ten minutes. Due to this, the stochasticity in the arrival time estimates has no impact on the quality of the schedules. The approach was implemented in a variant that incorporates future knowledge, and one that does not. The former is referred to as 'First Come First Served incorporating future demand' (FCFS+) and the latter as the regular 'First Come First Served' (FCFS) approach.
- Second, safety times were added to the expected on-belt time duration of each flights, so that more flights will be classified as overlapping.
- Third, the expected overlap between two flights are calculated, instead of using the overlap of the expected on-belt time windows. To do so, distributions and parameters are fitted for each of the estimators per flight. Then, using Monte Carlo (MC) simulation, the overlap between all combinations of flights are simulated a large number of times. For each combination of flights, we average the resulting overlap in all MC simulations, and use this as the expected overlap between the flight pair. This new method to calculate the expected overlap between flight pairs, is referred to as 'MC overlap estimation'. The old method, in which the overlap of the expected times are used, is referred to as 'basic overlap estimation'.

Separately, each of these approaches improve the performance based on the current situation by an equal amount. However, by combining the safety times and the MC overlap estimates, performance increases further. Other combinations of measures do not increase performance. The weekly results using unbiased estimates (but with stochasticity) are shown Table 1, in which a lower score is better. The historical solution value on the dataset was -6,730. Using MC overlap estimates, a safety factor of 2 minutes and the SA solution approach, we find a solution value of -8300, which is an improvement of more than 20%.

Table 1: Performance of the SA approach and FCFS approaches, using basic and MC overlap estimates and various safety factors

		Safety factor							
		0	1	2	3	4	5	6	7
basic overlap estimates	*SA*	-7,280	-7,343	-7,493	-7,651	-7,770	-7,745	-8,055	-7,882
	FCFS	-7,592	-7,578	-7,534	-7,452	-7,530	-7,392	-7,457	-7,203
	FCFS+ (large)	-8,046	-7,945	-8,000	-7,894	-7,702	-7,620	-7,525	-7,170
MC overlap estimates	*SA*	-8,060	-8,131	-8,300	-8,236	-8,223	-8,299	-8,170	-8,232
	FCFS	-7,972	-8,057	-7,924	-7,877	-7,857	-7,621	-7,572	-7,649
	FCFS+ (small)	-8,057	-8,037	-7,816	-7,719	-7,768	-7,802	-7,663	-7,620

The airport expects that in the future the estimates will become better and uncertainty in the estimates would therefore become smaller. A simulation study was conducted on the performance of the solution approaches under improved estimates. The results of the simulation showed that uncertainty in the transportation time has the largest impact on the performance of all solution approaches, and that while this standard deviation in the transportation time estimates is not improved from around 6 minutes to less than 4 minutes, the SA approach with MC overlap estimates is preferable for all cases. With less uncertainty in the transportation times and a similar level of arrival time uncertainty to the current situation, the FCFS approach with future demand and MC overlap estimates performs better. If arrival time uncertainty also decreases, either the SA approach with basic overlap estimates, or with MC overlaps is better.

Although methods using the MC overlap estimates are often better than methods using basic overlap estimates, it was found that it is much harder to visualize the MC overlap interactions, than the basic interactions. This makes it hard to show why a solution created using MC overlap estimates would outperform a solution created using basic overlap estimates, even though it generally does. Therefore, a trade-off must be made between the interpretability of a solution and the quality of a solution.

One of the limitations of the research is that we were not able to include the rolling horizon optimization, as it would take too much time to simulate these approaches, and it would greatly increase the complexity of the experiments. Instead, we solved one problem instance per day. We saw however that the difference between solving one problem instance and using the rolling horizon approach was small over one week of data. Nonetheless, we recommend that any alternative solution approach that is being considered for implementation is first run in parallel with the current solution approach to assess its performance under rolling horizon optimization.

Table of contents

Abbreviations

DCOP	Deterministic Combinatorial Optimization Problem
FCFS	First Come First Served
FDR	False Detection Rate
GAP	Generalized Assignment Problem
GQAP	Generalized Quadratic Assignment Problem
MC	Monte Carlo
MIP	Mixed Integer Problem
PAP	Process Allocation Problem
QMKP	Quadratic Multiple Knapsack Problem
SA	Simulated Annealing
SCOP	Stochastic Combinatorial Optimization Problem
TPR	True Positive Rate

Chapter 1: Introduction

In this chapter we introduce ORTEC, the company on behalf of which this research was conducted, in Section 1.1. Next, we will explain the problem context in Section 1.2. Finally, we formulate the goal of this research, as well as the research questions that must be answered to reach this goal, in Section 1.3. In Section 1.3 we will also provide an overview of the structure of the report.

1.1 – Company and department background

ORTEC is one of the world's leading optimization software and analytics solutions developers. It does so by combining operation research, IT and business process knowledge. ORTEC has around 800 employees and offices in 13 countries. The company is divided into two large business units, the first being ORTEC Consulting and the second being ORTEC Products. ORTEC Consulting develops tailored solutions for customer's challenges, while ORTEC Products is responsible for research, development and maintenance of ORTEC's standard software products.

The Center of Excellence is part of ORTEC's Consulting branch and aims to improve and maintain the knowledge base for ORTEC Consulting. The Center of Excellence explores data science and operation research methods that can be applied to customer's problems. The problem that we are about to present is based on an existing custom-made optimization solution developed by ORTEC. This solution was developed within a limited timeframe, and now ORTEC is in search for better methods to tackle the same or similar problems. As the assignment is related to exploring new methods for the customer's problem, it falls under the Center of Excellence division of ORTEC.

1.2 – Problem background and description

The customer we mention is a European airport that serves more than twenty million passengers annually. According to Borille and Correia (2013), one of the factors that affect the passenger's perceived service level of an airport is the availability of space around the baggage belts in the baggage reclaim area. It is considered a problem when the bags of different flights are put on the same belt at the same time, as this causes more people to gather along the baggage belts and thus more people experiencing less available space and thereupon a lower service level.

Our problem revolves around the implementation of a tool created by ORTEC for this airport. The tool automatically assigns flights to baggage belts at the airport. The main objective of the tool is to reduce the number of flights assigned to the same belt at the same time as much as possible, to provide a higher perceived service level to the passengers. The output of this tool is an assignment of each flight to a belt. A global overview of the tool is shown in Figure 1.

To reach the main objective, the tool must solve an assignment problem. As the flights' time estimates that are needed to solve the problem can constantly change during the day, parts of the problem are solved multiple times per hour. Every time the tool solves the problem, it takes the most recent information on all the flights of a certain period ahead into account, plus previous flights for which the baggage is still on the belts. The period of time that we look ahead is what we call the planning horizon.

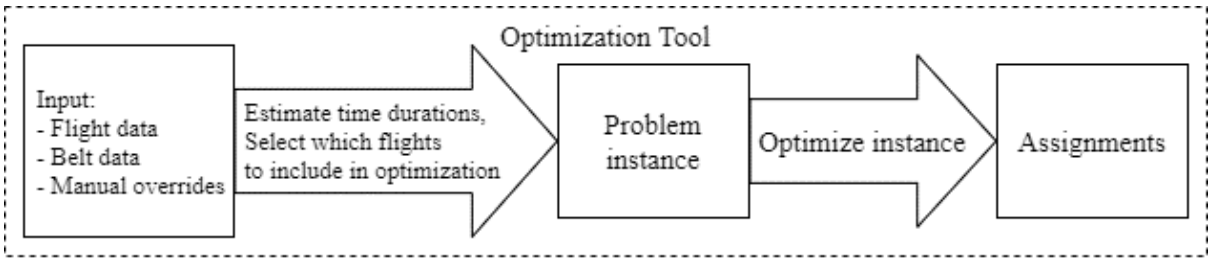


Figure 1: General overview of the structure of the optimization tool

If we would optimize based on a horizon of a few minutes, then every time we run the algorithm, we do our optimization based on a very small amount of flights. We might easily find an optimal allocation for the coming flights, but the overall quality of the resulting solution for an entire day could still be quite bad. The more flights we use to determine the allocation of the coming flights, the better the overall solution quality can be. However, the further we look ahead, the more complicated the problem to be solved gets, and the less likely it is to find an optimum solution quickly for each problem instance. Besides, there will be more uncertainty in the estimation of times for flights further ahead.

Due to the complexity of the problem, the size of the problem and the number of times the problem must be evaluated during a day, a heuristic approach is currently implemented. The tool uses a Simulated Annealing (SA) approach to create schedules. Every ten minutes, the heuristic gets five minutes to optimize the schedule of a fixed time horizon of three hours. For a longer run meant to create a daily backup, the heuristic gets an hour to optimize the schedules of a 48-hour horizon.

Besides the objective of reducing the number of flights assigned to the same belt at the same time, the heuristic must take several technical requirements and business rules into account, as required by the customer. Stakeholders around the airport would also like their preferences incorporated in the model.

It is not clear exactly how good the Simulated Annealing algorithm solves the problem instance that it tries to solve. The algorithm's performance would be very good if it would find the optimal assignment most of the times, and it would be very bad if it does not outperform random assignment methods. However, as this heuristic is currently considered the way to find the best results, we do not know how good certain problems can be solved. We can calculate an objective value for each solution of the algorithm, which indicates the absolute performance of the solution, but without knowing the objective value for the optimum solution of the problem, we do not know the relative performance of the solution.

From now on, we refer to this 'relative performance' as a solutions quality. Without knowing the quality of the generated solutions, we cannot assess the performance of the algorithm that generates these solutions. So, to assess the performance of the current algorithm (including its parameters), we want to compare the output of the tool to optimal solutions for the problem instances. This part of the problem does not revolve about changing the situation, but about gaining knowledge about the current solution approach. Therefore, this is a knowledge problem (Heerkens et al., 2017).

If the problem instances can be solved to optimality, this does not necessarily mean that the flights will be assigned to optimality, because there can be a discrepancy between the defined problem instances and reality due to uncertainty in the time estimates used to create the problem instances.

We assume the current heuristic does not (always) find optimal configurations for the global optimization problem. So, once we can assess the performance of the tool, we would like to improve this performance. This can be done by finding and implementing a heuristic or an improvement on the current Simulated Annealing heuristic that solves the problem instances better, or by finding a way to define the problem instances better. The goal is to change to current situation into a better situation. As there is a difference between the current and the wanted situation, this is an action problem. We see this problem as our core problem.

1.3 – Objective and research questions.

From the problem background and description follows our main research goal:

> *'Enhance the performance of ORTEC's baggage belt assignment algorithm for current, and improved time estimates, by developing a heuristic that outperforms the current assignment heuristic'.*

To reach our goal, we measure the performance of the current algorithm and create promising alternative approaches to solve the problem instances. Besides improving the solving of problem instances, we want to find ways to improve the creation of these problem instances. To achieve this in a structured way, we define research questions to be answered throughout the report.

In Chapter 2.1, we analyze the structure of the data that the tool has to its disposal and explain the workings of the implemented Simulated Annealing algorithm. Next, by analyzing the input data and comparing time estimates with realized times, we want to gain insights into the demand characteristics and evaluate the quality of the estimates. The first question to be answered in Chapter 2 is:

1) What are the current flight characteristics, and how does the current solution approach create solutions?
    a. How is the available data structured?

b. How does the current solution approach work?
c. How does the required data change during the day?
d. How do time estimates differ from the realized times?
e. How do changing time estimates influence the outcomes of consecutive schedules?

Next, we want to assess the performance of the current solution approach. First, based on the input data and the preferences of the airport, we introduce the objective function. To determine how well the current solution approach performs, we want to compare its results with the optimal results of the problem instances, as this gives a clear view of how well the instance is solved. Besides, we want to find the optimal solution to the actual problem, so based on the realized times, to see how close solving the problem instances will get us to the global optimal solution value.

While short-horizon problems (therefore with a few flights) can be solved to optimality in a reasonable amount of time, solving longer horizons will take too long to be a useable approach in operations, and therefore a heuristic approach was chosen. However, when generating optimal solutions in retrospect as a baseline to assess the performance of the heuristic, we would not need the model to solve quickly.

If we want to solve the problem to optimality, we need to translate the problem into a mathematical model formulation. We can then use the data from Question 1 as input for this model and determine optimal solutions by using a mathematical programming solver (CPLEX, a mathematical programming solver developed by IBM) to find the best value for the objective function. We are interested in finding the performance of the current solution approach relative to the best possible assignments for a problem instance, and in finding the performance of the current solution approach relative to the best possible assignment for the realized times. The former will give us information on the suitability of using the current Simulated Annealing implementation to solve a problem instance, while combining this with the latter can give us information on the suitability of using the problem instances with their estimates to solve the actual problem, with realized times. Therefore, the second question to be answered in Chapter 2 is:

2) How well does the current solution approach perform?
    a. How do we measure the performance of a solution?
    b. What does the mathematical model formulation of the problem look like?
    c. How well does the Simulated Annealing approach perform on the defined problem instances?
    d. How well does using the current solution approach perform on the realization of the expected times?

Next, we will be looking into literature to find suitable alternative solution methods. Our first section will focus on airport optimization problems, related to our problem. Next, we look at traditional

optimization problems that are similar to our problem, and if there are problems that are very similar, we look at best practices for the solution approaches. Finally, we investigate how we can implement the stochastic nature of the problem into the solution approaches. Therefore, the research questions to be answered in Chapter 3 are:

3) What can we learn from literature regarding:
    a. Airport optimization problems related to our problem?
    b. Traditional optimization problems related to our airport's problem?
    c. Incorporating stochasticity in combinatorial optimization problems?

After the analyses in Chapter 2 we have a clear view of the performance of the current solution approach and we should have a general idea about why current performance is good or bad. In Chapter 3, literature shall help us find promising alternative solution approaches.

In Chapter 4, we use the gained knowledge to discuss and implement alternative heuristics to solve the airport's problem. Next, we use the outcomes of the literature research to find a way to incorporate stochasticity into the optimization approaches. Finally, we can compare the performance of the solution approaches based on historic data. The research question to be answered in Chapter 4 is:

4) Which alternative solution approaches should we assess and how do they perform on historic data?
    a. Which promising heuristics should we implement and test for the given problem, and how can we implement these?
    b. How do we incorporate the stochastic behavior into the solution approaches?
    c. How do the different solution approaches perform on historic data?

We have reason to believe that the quality of time estimates will improve, as better estimators will become available to the assignment tool in the future. After Chapter 4 has given us an overview of the performance of the solution approaches for the current quality of time estimates, we will use Chapter 5 to set up experimentation that enables us to assess the performance of the solution approaches under these improved time estimates. The research question to be answered in Chapter 5 is:

5) How can we set up experimentation to assess solution approach performance under improved estimates?
    a. Which factors should we adjust throughout experimentation?
    b. How can we artificially change the quality of time estimates?
    c. How do we run the experiments in a structured way?

After answering Question 4 and 5, and implementing the approaches in a structured way, we can assess the performance of the different approaches. First we want to know what the best solution approach per

level of stochasticity is. Then, we want to analyze why this is the case. Therefore, we analyze the impact of stochasticity on each of the solution approaches, and assess how each proposed method for dealing with stochasticity, actually deals with the stochasticity. The research question to be answered in Chapter 6 is:

6) How well do the alternative approaches perform compared to the current approach under improved estimates?
    a. What is the best solution approach per level of stochasticity?
    b. How does stochasticity impact the performance of our solution approaches?
    c. How do our methods designed to mitigate the effects of stochasticity perform under improved estimates?

In Chapter 7, we formulate our conclusions, limitations and our recommendations for further research. The structure of answering the research questions that were discussed above is depicted in Figure 2.

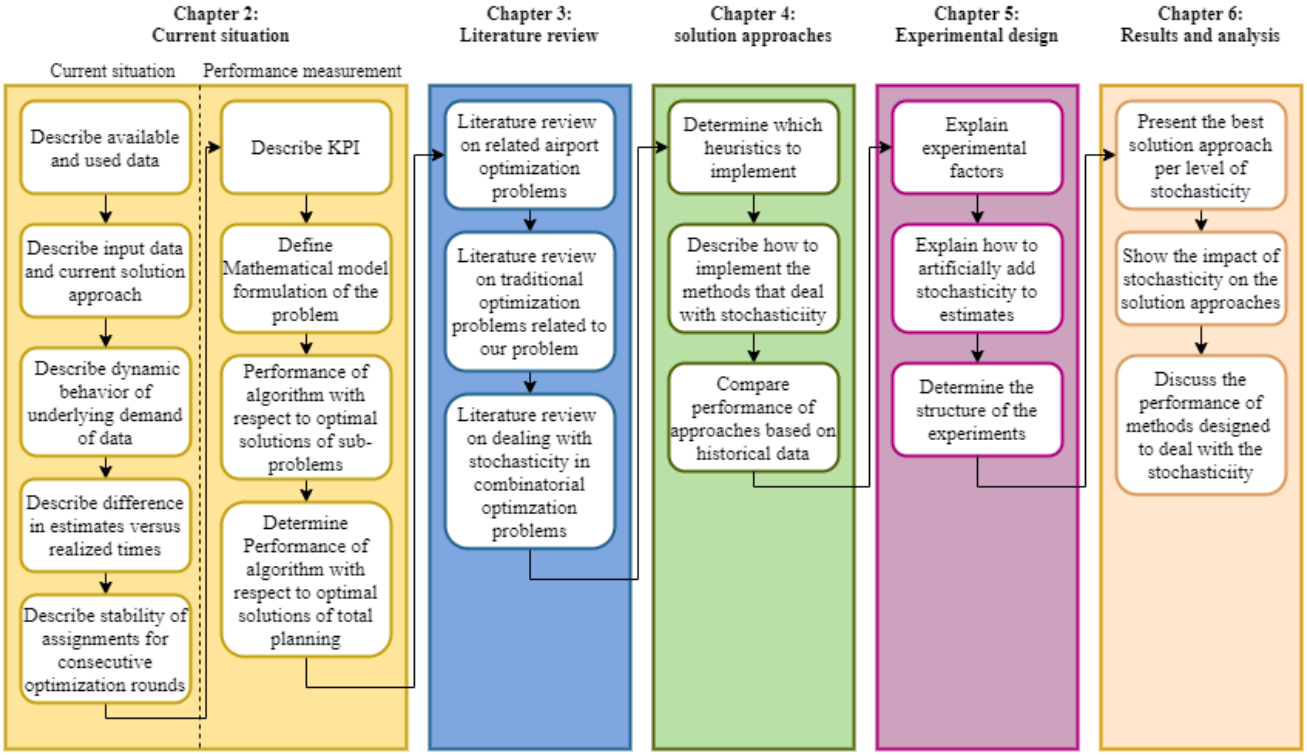


Figure 2: Structure of answering research questions per chapter

Chapter 2: Current Situation

This chapter should provide a clear view of the current situation. In Section 2.1, we describe the problem and analyze the input data, the current problem-solving method, and the output. In Section 2.2, we focus on assessing the performance of the current approach.

2.1 – Input and current solution approach analysis

In Section 2.1.1, we start with a brief overview of the problem and the solution approach to provide context for the rest of the chapter. Next, in Section 2.1.2, we describe the information that is available for the model, how it is generated, and how it is used. Then, in Section 2.1.3, we explain how this data is used to generate schedules. In Section 2.1.4 we analyze the estimates for input data over time and in Section 2.1.5, we analyze the differences between the data estimates and the realized data. Finally, in Section 2.1.6 we present a set of problems that represent the entirety of the data well, and that serve as a test set.

2.1.1 – Problem description

Every day, hundreds of flights arrive at the airport carrying passengers and their hold baggage. This is baggage that passengers checked in at their origin airport and must retrieve from the baggage retrieval belts at our destination airport.

The problem revolves around the assignment of the flights to the baggage belts. The goal of the assignments is threefold. First, to minimize the amount of flights that will be present at the same belt at the same time, as this will increase customer satisfaction. Second, to reduce the amount of times the baggage of a special class of flights arrive at an occupied baggage belt, to increase service differentiation possibilities for this class of flights. Third, to maximize the number of flights assigned to one of their preferred baggage belts. In Section 2.2.1, this objective, and the reasoning behind it, will be discussed in more detail. For the assignments, multiple flights can be assigned to the same baggage belt at the, but only one baggage belt can be assigned per flight.

When an incoming flight arrives at its parking place (its ‘apron’) at the airport, the decision to which baggage belt the baggage must be transported, must be made. The baggage is then removed from the aircraft and transported to the assigned baggage belt. Upon arrival at the infeed station of the baggage belt, it will take some time before all baggage of the flight is put on the baggage belt, and before it is all retrieved from the baggage belt.

For all incoming flights, there are estimates for the arrival times of the flight, which change over time. There are also estimates for the transportation time and for the total time a flight’s baggage will be on

the baggage belt. The latter two estimates are based on the flight's baggage class and the flight's assigned apron. Both the baggage class and the apron generally do not change in the 24 hours prior to arrival of the flight, and therefore these estimates also do not change.

To solve the assignment problem, the airport uses an optimization tool. For the current solution approach, the tool works with optimization rounds of ten minutes per round. At the start of every optimization round (so every ten minutes), the tool retrieves the latest information on upcoming flights and flights from which the baggage is already being handled at the airport from another system. This data and the used time estimates will further be discussed in Section 2.1.2. The tool then creates a problem instance, in which all flights that are being handled at the airport, plus all flights expected to arrive in the upcoming three hours, are incorporated with their time estimates (and realized times, if available).

Based on the realized times, the time estimates, and the belt preferences of the flight, the tool creates a new schedule using a random constructive heuristic and then improves the assignments using a five-minute SA run. When a flight arrives at its apron, the results of the most recently completed SA run are retrieved, and the flight is assigned to the belt it was assigned to by the SA algorithm.

2.1.2 – Exploring the input data

In this section we give an overview of what data is available and used in the current solution approach. Table 2 shows the data structure of the optimization tool and describes the variables.

Table 2: Overview of relevant input data

Variable	Description	Type
Id	Unique flight identifier	string
Scheduled_date_of_arrival	Scheduled local date of arrival	string
Flight_number	The number of the flight (3-4 digits)	string
Apron	The apron that was assigned for this flight	string
Aircraft_baggage_class	A, B or C, baggage capacity of airplane used for the flight (A = small, B = medium, C = large)	string
Alliance_code	The airline-alliance from which the carrier is part (if any)	string
Assigned_belt_id	The baggage belt that was assigned to the flight in the previous optimization round	nullable integer
Best_known_on_block_time	The estimate of the time at which the aircraft gets on-block (and equal the realized time, once the aircraft gets on-block). This is the moment that blocks get put under the wheels of the aircraft, so it will not be able to move	datetime

Actual_on_block_time	The time at which the aircraft actual gets on-block. Is empty if this has not happened yet	nullable datetime
Assigned_belt_from	Time estimate of the moment that the first piece of baggage arrives on the belt (equal to actual time when known)	nullable datetime
Actual_first_bag_on_belt_time	Time that the first piece of baggage arrives on the belt. Is empty if this has not happened yet	nullable datetime
Assigned_belt_to	Time estimate of the moment that the last piece of baggage is taken from the belt (equal to actual time when known)	nullable datetime
Actual_last_bag_on_belt_time	Time that last piece of baggage arrives on the belt. Is empty if this has not happened yet	nullable datetime
Fixed_belt_assignment	True if the assigned belt cannot be changed anymore (due to hard requirements)	boolean
Last_allocation_modified_when	The most recent moment that the belt occupation (in time) estimates were adjusted	nullable datetime
Preferred_belts	A list of belts that are preferred for the flight. Generally, these preferences are fixed per carrier, but sometimes specific flight numbers have more specific preferences or requirements. ("All flights of carrier *XX* are preferred on belts A,B or C, but flight *XX_1234* is required on belt C").	Nullable list of strings
Fixed_belt	The ID of the baggage belt that is required for the flight, if there is any. If a belt is required, then Preferred_belts will be empty.	Nullable string

Figure 3 shows the flight and baggage processes that are relevant to us, including input data changes that are triggered by the process. The solution approach currently used by ORTEC, which will be described in Section 2.1.3, uses eight input parameters per flight:

1. Whether or not the flight's assignment may be changed (**Fixed_belt_assignment)**
2. Whether or not the flight has already arrived (**Actual_on_block_time)**
3. The flight's belt assignment in the prior optimization round (**Assigned_belt_id**)
4. Whether or not the flight's carrier is part of an alliance (**Alliance_code**)
5. A set of preferred belts or a fixed belt, if any (**Preferred_belts)**
6. The belt that is required for this flight, if any (**Fixed_belt)**
7. The (estimated) starting time of the flight's belt occupation (**Assigned_belt_from**)
8. The (estimated) ending time of the flight's belt occupation (**Assigned_belt_to**)

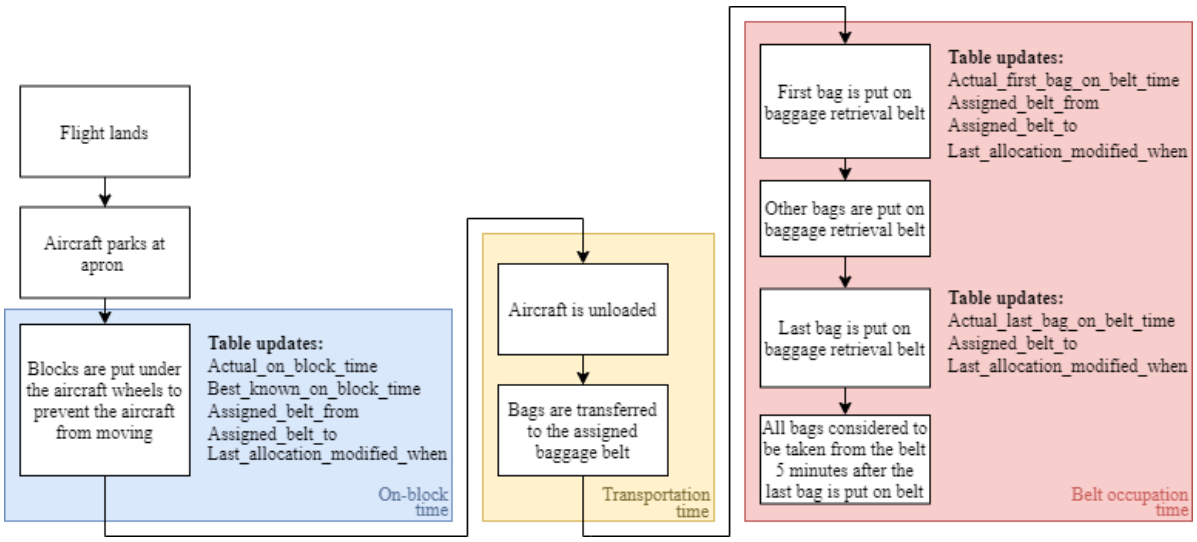


Figure 3: Overview of baggage handling steps and information updates

A flight's belt assignment may not be changed by the algorithm if a specific belt is needed for the flight. In this case, the Fixed_belt_assignment variable will be TRUE. The flight's belt assignment may also not be changed anymore if the flight has arrived in the airport and is on-block (in this case, the Actual_in_block_time will have a value). In these cases, the flight will be assigned to its previous assignment. Whether or not the flight's carrier is part of an alliance is static data that does not change, and neither do the belt preferences change. These belt preferences are mainly based on the carrier of the flight, but sometimes a specific flight must be put on a specific belt, due to special baggage handling facility only available at that belt.

The estimated starting time of the flight's belt occupation is based on the time that an aircraft is on-block (the Actual_on_block_time variable), and the estimation of the handling time is needed to estimate when its first bag is put on the belt. This includes the time it takes to unload bags from a flight entirely and the time it takes to move the bags from the plane to the belt systems. The time it takes to unload a flight is dependent on the number of bags that must be unloaded and the number of people unloading the flight. The time it takes to move the bags mainly depends on the distance to be covered. Currently, for the unloading time estimator, the baggage class of the aircraft is used. This baggage class is only based on the type of airplane that executes the flight, so it does not incorporate the number of bags or the number of passengers. For the transportation time estimator, the airport currently uses the apron at which the aircraft gets on block. For each of these *Apron/Class*-combinations, a fixed number of minutes is used for the transport time estimation. Aprons closer to the baggage retrieval belts have lower transportation time, and flights of smaller baggage classes are also expected to have lower transportation times.

The estimated ending time of the flight's belt occupation is based on the estimated starting time and the estimated duration on the belt. This estimated duration is also based on the aircraft's baggage class. The following times are used:

- baggage class A (about 10% of the flights in the analyzed week): 8 minutes,
- baggage class B (about 65% of the flights in the analyzed week): 10 minutes,
- baggage class C (about 25% of the flights in the analyzed week): 20 minutes.

There is no precise measurement for how long bags are on the baggage belts, as it would take a lot of effort to keep track when the last bag of a specific flight leaves the belt, especially when there are also bags of other flights on the same belt. However, the moment the last bag of a flight is put on the belt, a physical button is pressed by the baggage handlers which stores the Actual_last_bag_on_belt_time. The airport uses this time and adds five minutes to it to determine the realized end of the belt assignment of a flight.

The baggage class of the used aircraft type is known long in advance and the apron is assigned to the flight about 24 hours in advance. Only about 1% of the assigned aprons change within 1 hour prior to arrival, and the impact on the arrival time estimation is a few minutes at most. Therefore, we consider the transportation times and on-belt time estimates to be static. As a result, only a change in the expected arrival on-block time impacts the estimated start and completion time of a flight on a belt.

2.1.3 – Exploring the current solution approach

In this section we explain how the available data is used to create problem instances, and how these problem instances are solved. First, the structure of the problem instances is explained, and second, the method for finding solutions for these problem instances is explained.

Problem Instances: Rolling horizon optimization

We will refer to the problem that an optimization algorithm tries to solve as the problem instance, whether the problem that an algorithm tries to solve is the actual problem that arises or not. The tool makes use of a rolling horizon optimization approach, so the problem that the algorithm tries to solve is based on all flights that will be arriving within a certain period ahead. Our problems consist of all flights that are expected to arrive within the upcoming three hours, plus the flights that are still present on the baggage belts. Every time a flight gets put on block (the moment at which actual blocks are put underneath the aircraft's wheels), the solution of the most recent completed optimization run is retrieved, and the flight is assigned to the belt it was assigned to in that previous solution.

Every ten minutes, a problem instance is created, which results in much overlap between these problem instances. With respect to the preceding problem instance:

1. the flights that are not on the baggage belt anymore are dropped from the problem,
2. the flights that have arrived on block since the last optimization round are ‘frozen’ to the belt that they were assigned to in the last round,
3. the next ten minutes of flight arrivals are added to the problem. For these flights, a random belt will be temporary assigned, keeping in mind the constraints presented in Section 2.1.2.

So, if we look at Figure 4, the ‘Free Interval’ length is three hours, the length of the ‘Frozen Interval’ runs back from the current moment, including all flights that are already block, but not yet completely processed on the baggage belts, and the length of ‘New Flights’ is ten minutes.

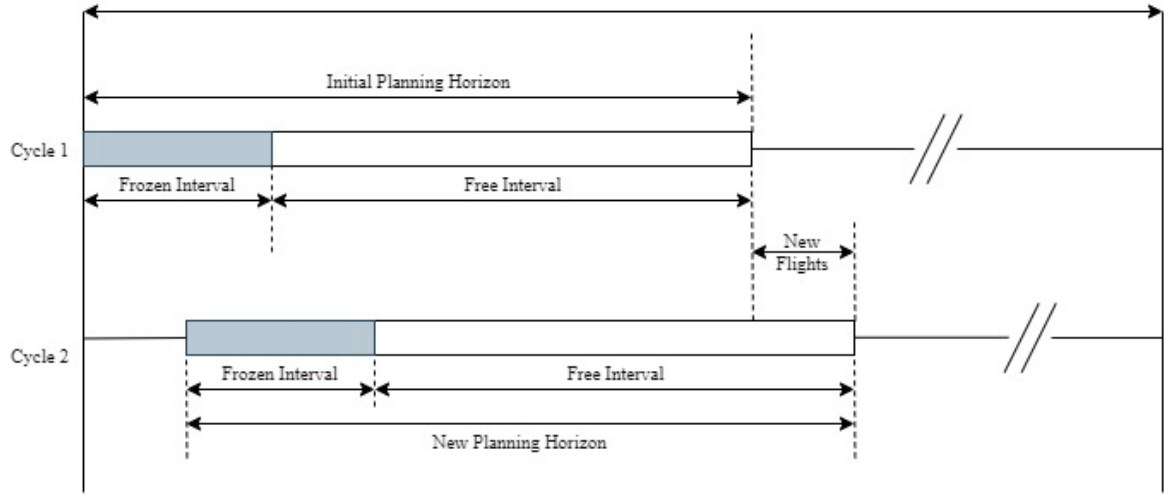


Figure 4: Rolling horizon planning concepts (adapted from Narayanan and Robinson 2010).

Then, using Simulated Annealing, a global search for a better solution to the problem instance is initiated. This search has a time limit of around five minutes. We call this process of fetching new data, running a five-minute optimization followed by a five-minute buffer, an *optimization round*.

In Figure 5, we visualize how a single flight is incorporated in these optimization rounds. In the figure we see the process of a flight’s data changing from estimates to realized times. Every optimization round (so every ten minutes) the system receives (new) estimates for the arrival time of the airplane on-block (A). Together with the static estimates of the internal transportation time and the time that baggage will be on the belt, we find the expected moment in time that the belt is occupied defined by a start moment (s) and an ending moment (e). These starting and ending moments of the flight’s belt occupation are incorporated in the optimization.

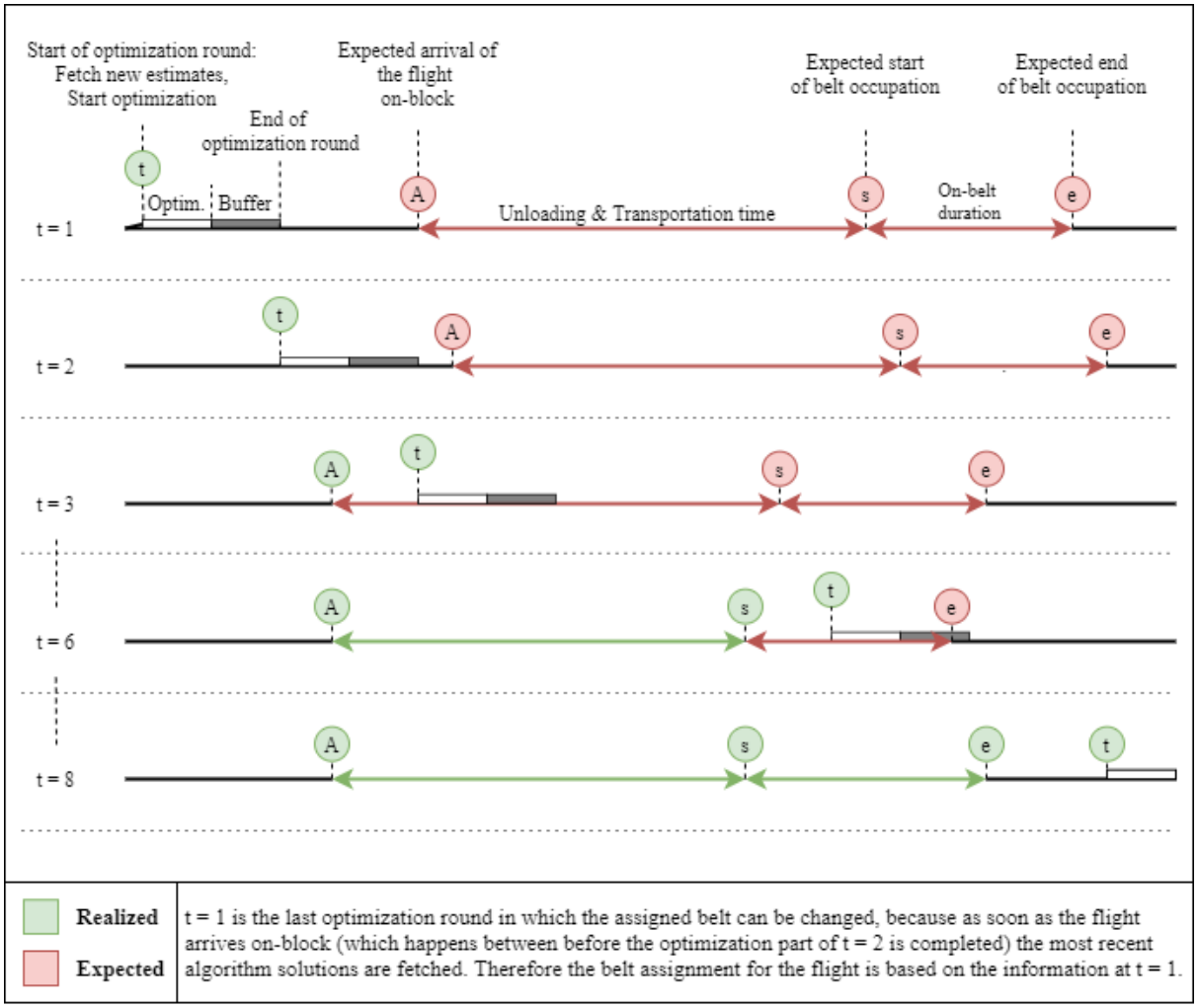


Figure 5: Relation between optimization round, time estimates and realizations.

At the data update at *t=2*, we see that the flight's estimated on-block arrival is postponed. Between *t=2* and *t=3*, the flight arrives on-block. However, we see that the optimization of the optimization round that started at *t=2* has not finished yet. Therefore, most recent completed optimization was the optimization of *t=1*, so the flight will be assigned to its result in optimization round *t=1*. Throughout the next optimization rounds, the starting and ending moment of the flight's belt occupation will change when time estimates change into realizations. Finally, at *t=8*, the problem will be no longer incorporated in the optimization heuristics, as the flight is no longer present at the baggage belts at the start of the optimization round.

In the next section, we will explain what happens during these optimization rounds.

Simulated Annealing approach

The algorithm only looks at the expected blocks of belt occupation, at whether the flight's airline is part of an alliance, and to the preferred (or required) belts of the airline. The SA approach uses swap and move operators to determine a new planning. If more than one flight in the problem is changeable, the

swap and move heuristics have an equal chance of being picked as the neighborhood operator for the next iteration.

Each iteration consists of determining the neighborhood operator, picking a random flight and a random new belt to move it to in case of the move operator, or picking two random flights to which belts will be swapped in case of the swap operator. Next, the expected change in the objective value will be calculated and based on this change and the current temperature of the SA algorithm, it is determined whether this swap or move should be executed. The objective function will be explained in Section 2.2.

Running the simulated annealing approach on a quadcore Intel Core i7-8650U with 16GB RAM, we find that on average for five experiments around 130,000 iterations per second are made by the Simulated Annealing algorithm. The number of iterations per second lie close to each other for the same problem instance, but if we look at two days' worth of problem instances of the airport's data logs, we see in Figure 6 that the number of iterations per second vary between 130,000 and 550,000 iterations per second in the online tool (with unknown computer specifications). There is a clear relation between the number of movable flights in a problem instance and the number of iterations per second, with a logarithmic ($R^2 = 0.964$) relation. As the density of flights in a problem increases, a shift in the schedule will impact more flights and it will take more computational power to assess the impact on the objective value .

The Markov chain length is 1, so after every iteration the temperature gets updated, using a cooling factor. The algorithm runs with a variable cooling factor, which gets recalibrated every 100,000 iterations. To determine the new cooling factor after these 100,000 iterations, the tool calculates the run's average number of iterations per second and (using this number) how many iterations are expected to be completed in the remainder of the run time. Considering the current temperature, the predefined ending temperature and the expected number of iterations left, the cooling factor is updated.

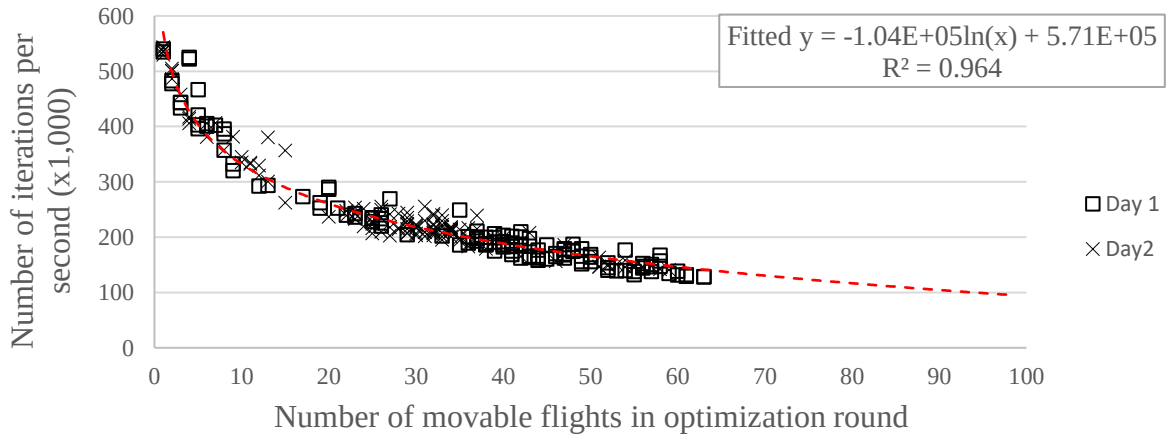


Figure 6: The number of iterations per second, given the number of movable flights in an optimization problem

In Appendix A, we present figures on the behavior of the objective values, best-known objective value, temperature, average acceptance probabilities and the cooling factor over the course of a simulated annealing run, for five replications. We see that the replications of the optimization behave similar and that they find the same solution value for the same sub-problem.

Figure 7: The planning of an arbitrary morning worth of flights

The realization of a morning's planning is shown in Figure 7. Flights that are assigned to one of their preferred belts are represented with a green time window, grey time windows represent flights that must be mandatorily assigned to their fixed belt, red time windows are flights that are assigned to a belt that did not have their preference even though they had a (set of) preferred belt(s), and flights that did not have any belt preference are represented in blue. Each flight with a blue 'A' in it, represent a flight whose carrier is part of an alliance, as these flights are penalized when their assigned belt is occupied upon their arrival.

2.1.4 – Dynamic behavior of the data

For our data analysis in Sections 2.1.4, 2.1.5 and 2.1.6, we have used the data of an arbitrary week in 2019. We will use this data to explore the dynamism and stochasticity in the input data and the resulting estimates. In these sections we do not aim to improve the estimation method nor to provide numerical recommendations to improve estimates, but we are interested in exploring and explaining the behavior of the data to be able to improve the solution approach later on.

When we assess the dynamism in this section, we are interested in how flights' start and completion on belt time estimates change before their arrival on-block. Recall from Section 2.1.2 that the aircraft's baggage class and the flight's assigned apron are fixed, so a flight's expected start and completion time on the baggage belt only will only change due to a change in the estimated arrival time. This might be due to (amongst others) delayed departure, weather conditions and air traffic control decisions. When we assess the stochastic behavior in Section 2.1.5, we focus on the differences between estimated and the realized times.

To determine how the arrival time estimates change over time, we first find for each flight the optimization round in which the flight has arrived on-block and, therefore, the arrival on-block time is not an estimate anymore. From now on, will refer to this as the flight's $t_{arrival}$, or t_0 (for the flight in in Figure 5, $t_{arrival} = 3$). Then we look at the arrival time estimates of the flight at its preceding optimization rounds $t_{arrival}$ - 1, …, $t_{arrival}$ – N (t-1,…, t-N in short), and calculate the difference between each round's estimate and that of the flight's arrival time estimate in the consecutive optimization round for each flight. For a large set of flights, these changing estimates are shown in Table 3.

We find that from one-and-a-half hour before the actual arrival until one hour before the actual arrival, every ten minutes the expected arrival moment of about 15% of the flights change, with on average 7 to 9 minutes. The majority of the 15% of flights with a changing expected arrival time, get an earlier expected arrival time. The remaining 85% of flights keep the same arrival time estimation.

Table 3: Statistics on changing arrival moment estimates for consecutive optimization rounds (n > 1,000)

Time slot progression: from, to	t-9, t-8	t-8, t-7	t-7, t-6	t-6, t-5	t-5, t-4	t-4, t-3	t-3, t-2	t-2, t-1	t-1, t0
% of flights with changing arrival estimates	10%	13%	16%	14%	32%	66%	71%	85%	76%
Flights with belated estimates (given there is change)	40%	42%	35%	31%	69%	79%	43%	26%	33%
Average change*	-0.22	-0.25	-0.50	-0.50	0.55	2.68	-0.28	-1.69	-0.59
Average change* (given there is change)	-2.17	-1.99	-3.02	-3.67	1.75	4.04	-0.39	-1.98	-0.79
Average absolute change (given there is change)	9.25	9.00	8.35	7.19	5.12	5.39	3.75	3.39	2.03

*a negative value implies that in the new optimization round, the flight is expected earlier than in the preceding optimization round.

Between $t_{arrival}$-4 (which corresponds to the window of 30 to 40 minutes prior arriving on-block) and $t_{arrival}$-3 (which corresponds to the window of 20 to 30 minutes prior arriving on-block), the expected arrival is structurally postponed. The arrival expectation of more than half of all incoming flights are postponed in this time window. For the estimation change between $t_{arrival}$-2 and $t_{arrival}$-1, the opposite

holds: more than half of all flights are now expected to land earlier. From this clear trend it seems likely that the estimation method changes 20-40 minutes prior actual arrival. Therefore, it is hard to say which changes in expected arrival time are due to external effects on the actual arrival moment (when an arrival estimate is belated because the flight is delayed) and which are due to changing estimation methods.

2.1.5 – Uncertainty in time estimates

In this section the question of how the deterministic estimates differ from the realized times is assessed. First, we will look at the on-block time of the flights, next at the transportation times and third at the on-belt duration of the flights. Finally, we will see how these estimation errors impact the overlapping periods of each flight combination. In these sections, we will try to fit distributions over the realized times and the differences between the estimated and the realized times. In our solution approach (Chapter 4), we will use these distributions to assess the impact of reducing uncertainty in these estimates.

On-block time

To assess how the deterministic estimates differ from the realized on-block time, we look at the estimates of the last optimization round before arrival, because that is the last moment in which a flight's assignment may still change. This last optimization round before arrival is between 1 and 10 minutes prior to arrival. An overview of these estimates is shown in Figure 8.

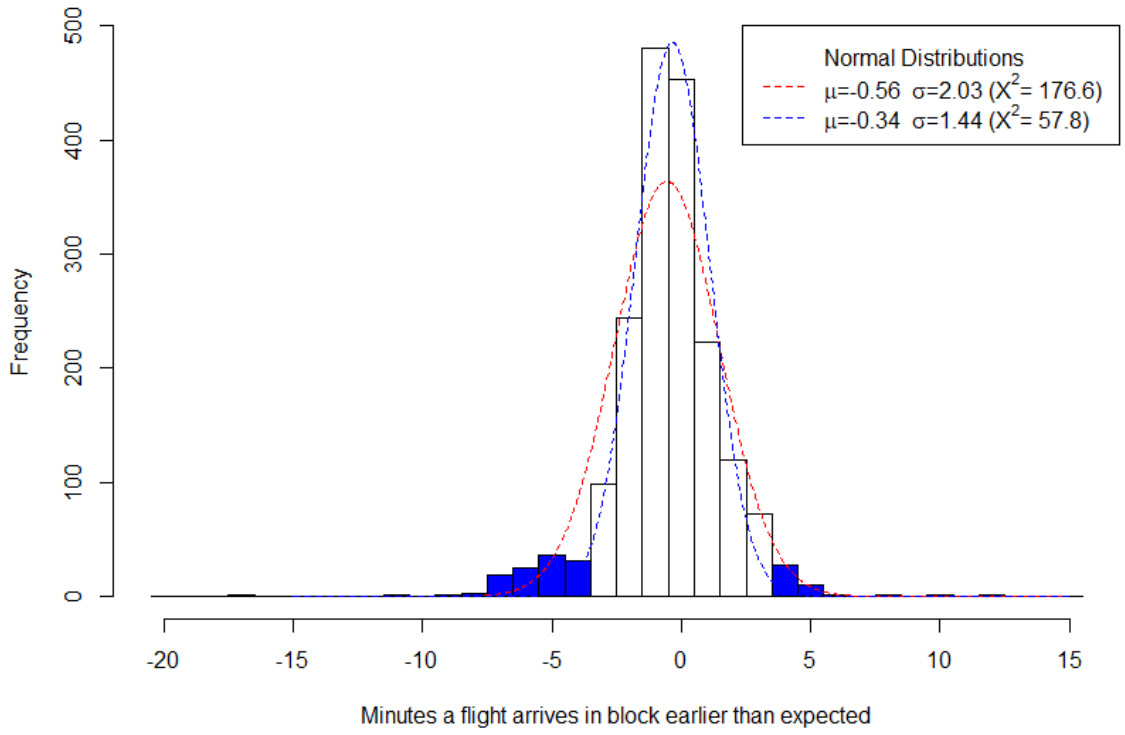


Figure 8: Deviations in arrival on block time estimates, with two fitted normal distributions

The normal distribution was fitted over the datapoints, but in neither sets with outlier removal (red contains all values in the range of [-10,10] and blue only the values in the range of [-3;3]), Chi-Square values below the test statistic could be obtained (19.02 and 14.45 respectively). Therefore we conclude that the deviation from the actual arrival time is not normal distributed. Appendix B contains all Chi-Square calculations.

Table 4: The number of minutes a flight was expected later than its actual arrival

Time	Mean overestimation of arrival time	Standard Deviation
$t_{arrival} - 1$	0.59	2.38
$t_{arrival} - 2$	2.29	3.97
$t_{arrival} - 3$	2.56	3.76
$t_{arrival} - 4$	-0.12	4.76
$t_{arrival} - 5$	-0.68	5.73
$t_{arrival} - 6$	-0.18	6.16
$t_{arrival} - 7$	0.33	7.30
$t_{arrival} - 8$	0.59	7.96
$t_{arrival} - 9$	0.84	8.52

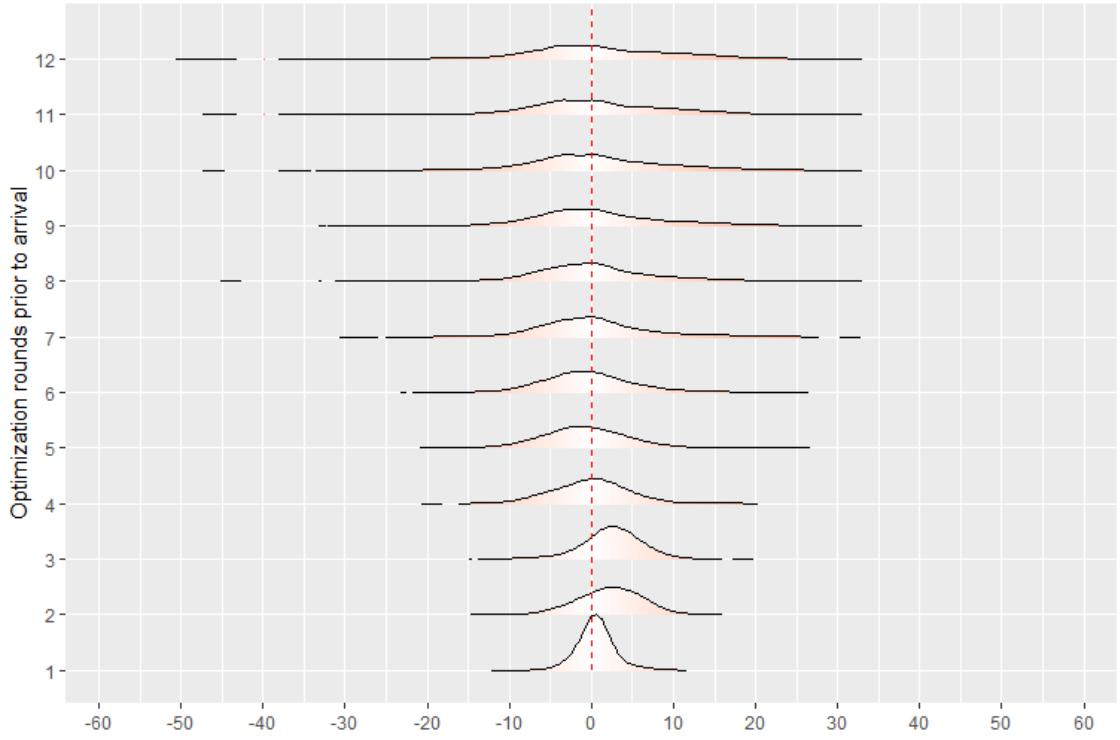


Figure 9: Density plot of the number of minutes flights are expected to arrive later than their actual arrival, over time

In Figure 9, the development of the deviation in arrival time over time is shown, and Table 4 shows the bias and standard deviation for the arrival time estimation. We see that upon the fourth optimization round before the actual arrival of the plane, the predictions are becoming more precise over time.

However, in line with the results from Section 2.1.4, we see that the arrival time estimation at $t_{arrival} - 3$ has a worse accuracy than the preceding optimization rounds.

Transportation times

The time between the flight arriving on block and the first baggage of the flight arriving on belt is what we call the transportation time. The time includes unloading the aircraft and transporting the bags from the apron to the baggage retrieval belt. The apron at which the aircraft is parked is known a day in advance. For the same week of data as addressed in the on-belt duration explanation, we plotted the estimated transportation times versus the corresponding realized transportation times, as depicted in Figure 10. These time estimates follow from the aircraft's baggage class and the apron at which the aircraft gets on-block.

We note that besides variance, there is also bias in these estimates. The Pearson's correlation between the estimated and the realized transportation times is -0.01 in this week's data. Thus, there is no linear correlation between the expectation and the realization of the transportation time for these flights. It is however not yet clear if this is due to the use of non-distinctive estimators, or because the used transportation time estimates for each group do not reflect the group averages correctly.

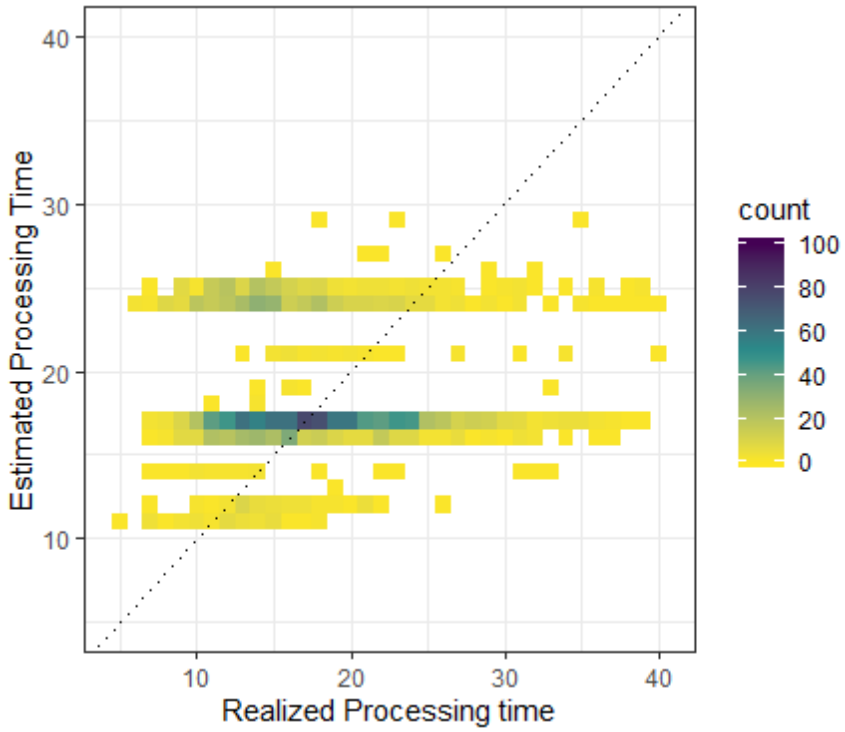


Figure 10:Realized versus estimated processing times

In Table 5 we have calculated for each baggage class the average expected transport duration, the realized transport duration and the weighted average standard deviation of the actual transportation

times based on the apron/class estimators. We find that class A and Class B transport estimates are structurally underestimates, while class C flight transportation times are overestimated.

Table 5: Overview of expected and realized average transportation times

Flight's baggage class	# flights	Expected	Actual Transport time	
	Total	Mean	Mean	St. Dev.
A	**~10%**	12.7	14.6	4.0
B	**~65%**	16.9	18.3	5.9
C	**~25%**	24.4	16.9	6.4

To assess the potential performance of the *Apron/Class*-combinations as estimators we look at the realized average transportation time for each combination of apron and baggage class, and use these realized averages as the new transportation estimate. Using the same data for parameter estimation and performance measurement will give us an (too) optimistic estimation of the potential performance of the estimators. We find a Pearson's correlation value of about 0.25, indicating that using best-case estimates, in which the performance estimation is too optimistic, would result in small correlation. A visual representation of the correlation between the realized and the estimated processing times is shown in Figure 11.

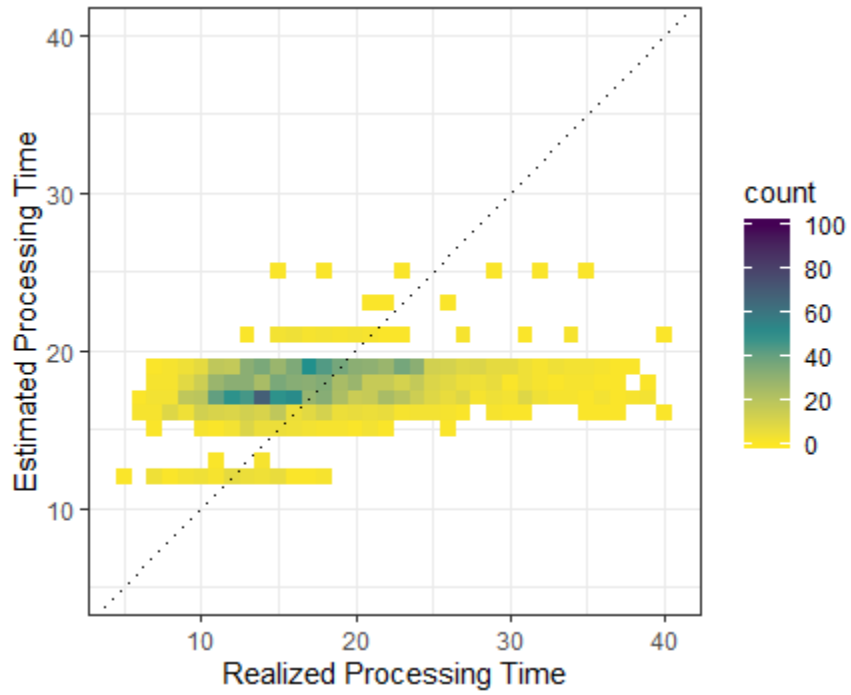


Figure 11: Realized versus estimated processing times, for bias minimizing time estimates

For each of these baggage classes, we have fitted a normal distribution and a gamma distribution over the transport duration data. Using the Chi-Square test to perform statistical comparison, we find that for

baggage class A and baggage class B, the data, with outliers removed, does not significantly (α=0.05) differ from the fitted gamma distributions, in Figure 12 we show the data including the fitted distributions of baggage class B. For baggage class C, we need to remove more data to find a proper fit, which indicates that either those flights are not gamma distributed, or that we need to make further distinction within the flights of baggage class C to be able to find a good fit. Appendix B contains plots of the fitted distributions and calculations of these Chi-Squared values.

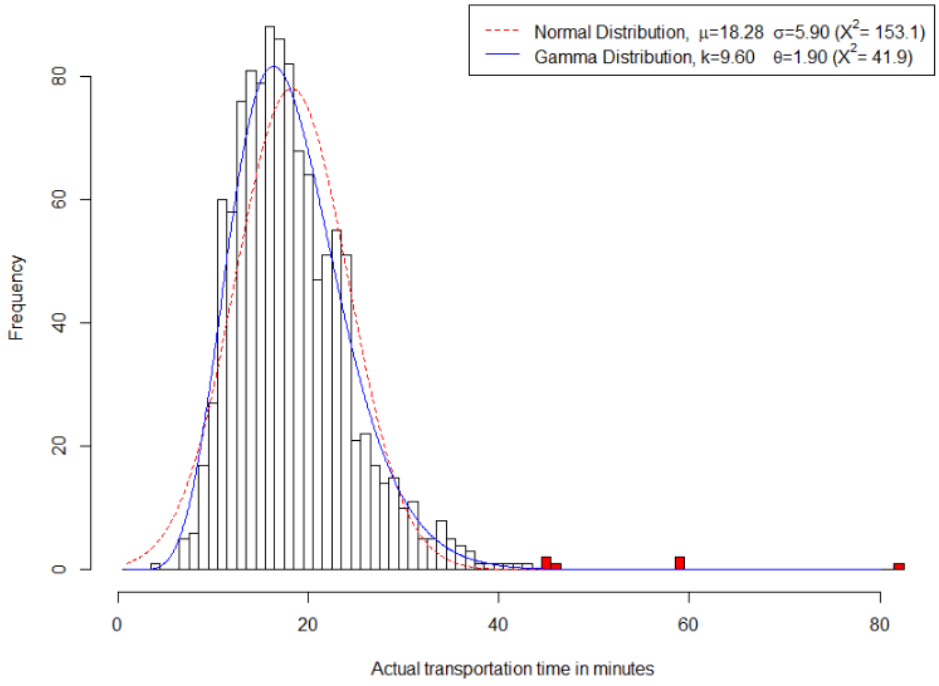


Figure 12: Internal transportation duration for baggage class B, distributions fitted with outlier removal (outliers in red)

On-belt times

The on-belt time estimates are based on the baggage class of the operating aircraft, for baggage class A, B and C, respectively 8, 10 and 20 minutes are used as estimates. The baggage class is linked to the type of aircraft used, with classification 'A' for small aircrafts and 'C' for large aircrafts. The classification does not consider the actual number of passengers (or even better, the number of bags on board). The realized on-belt times are calculated by determining the time between the first bag on the belt and the last bag on the belt for each flight and adding five minutes to this. A distribution could neither be fit over these realized on-belt times (shown in Figure 13, Figure 14 and Figure 15), nor on the difference between a flight's first and last bag on the belt. The on-belt duration times are deliberately over-estimated to create a small buffer for each flight's uncertainty. However, the high on-belt duration estimates do not cover the actual on-belt durations for all flights. For baggage class A, B and C, 86%, 74% and 76% of the on-belt time windows fall below or are equal to their threshold value, respectively.

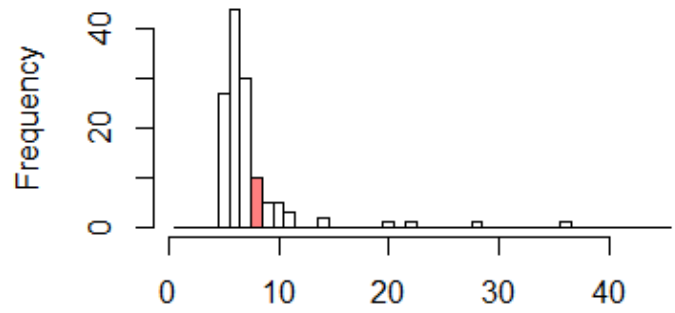


Figure 13: Realized on-belt duration for baggage class A

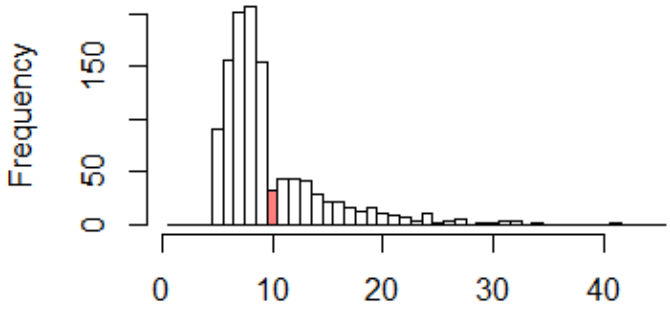


Figure 14: Realized on-belt duration for baggage class B

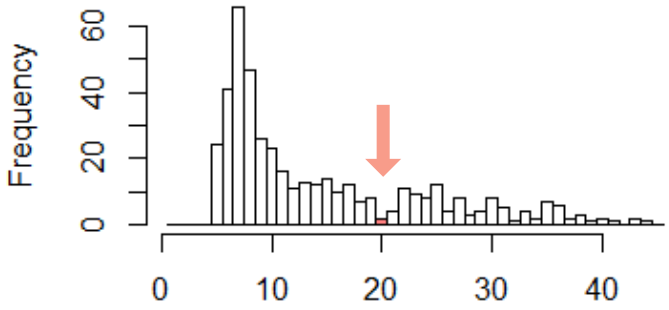


Figure 15: Realized on-belt duration for baggage class C

Overlap times

Uncertainty in the flight's arrival time, transportation time and on-belt duration time, cause uncertainty in the time window in which the flight is expected to be on the baggage belt. As the goal of the assignment tool is to reduce the number of flights assigned to the same belt at the same time, it is important to properly predict which flights will be on the same belt at the same time. The total number of minutes that two flights (a flight combination) are on a baggage belt at the same time is what we call overlap. The overlap between the two flights is only penalized if the flights are assigned to the same belt.

Out of the one-week sample of more than 1.000 flights, looking at each flight's expected time window at that flight's last optimization window before actual arrival, there were 5330 flight combinations expected to have overlapping time windows for their expected baggage belt occupation. Of these flight combinations 2527 (47%) did end up overlapping. The overestimation of overlapping flight pairs is likely due to the overestimation of the on-belt duration of the flights and the poor time estimates.

In Table 6, we show statistics regarding overlap for flights that were expected to overlap for a given number of minutes. We see a strong decline in expected number of flights to overlap for overlap of more than 10 minutes, which makes sense, because only a combination of two flights of the largest baggage class can overlap for 11 minutes. There are less flights with the highest baggage class and thus even less flight combinations with two flights of the highest baggage class. For flight pairs with an expected overlap of less than 11 minutes, we see little connection between the expected overlap duration, and the average realized overlap, given there is overlap.

Using the current estimates, we previously found that 2527 flight combinations were rightfully identified to be overlapping. Besides these 2527 flight combinations, there are also flight combinations that ended up being on a baggage belt at the same time, while they were not expected to overlap: an additional 1716 flight combinations. This makes the total number of overlapping flight combinations 4243.

Table 6: Overlap statistics for flight combinations that were expected to overlap a certain amount of minutes

Expected overlap in minutes	1	2	3	4	5	6	7	8	9	10	11	12	13 to 20 (totals)
Number of flight combinations expected with this overlap	463	463	464	468	472	459	491	634	390	783	24	21	198
Number of flights that will actually overlap	183	193	200	219	229	231	263	324	208	299	16	14	148
A verage overlap in minutes	2.11	2.25	2.31	2.61	2.62	2.90	2.91	2.86	3.18	2.33	5.75	10.28	9.46
Average overlap, given there is overlap	5.35	5.41	5.46	5.58	5.40	5.77	5.44	5.60	5.96	6.12	8.63	15.43	12.66
% of flights that will actually overlap	0.40	0.42	0.43	0.47	0.49	0.50	0.54	0.51	0.53	0.38	0.67	0.67	0.75

If flights are not overlapping, we say that flights have a gap between them: the expected gap is the number of minutes the later starting flight is expected to start after the earliest flight has already ended. If one flight of the flight combination arrives on belt as soon as the other flight is finished on the belt,

both the overlap and the gap between the flights is zero. In Figure 16, we show the expected gap for the flight pairs that ended up overlapping, while they were not expected to.

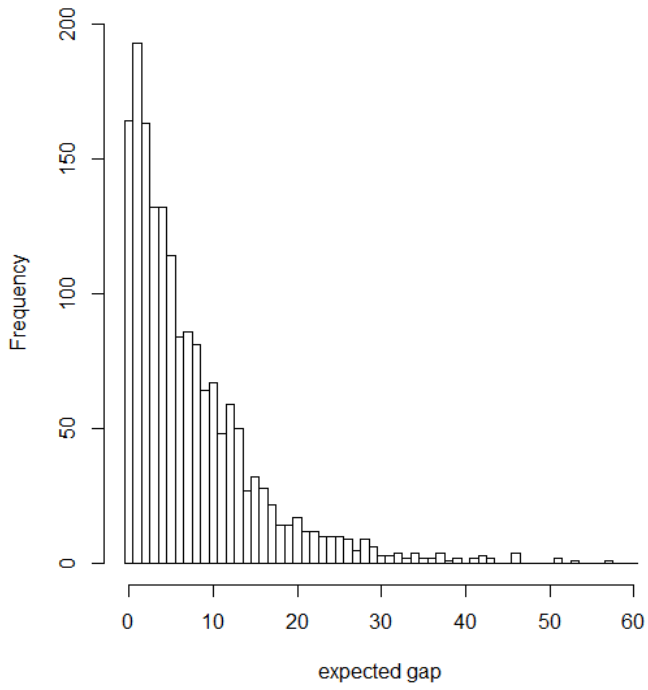


Figure 16: Expected gap (in minutes) between flights that ended up overlapping

As around 40% of the overlaps is not considered in the solution approach, there might be a large difference between the expected performance and the actual performance of the generated solution, and a large difference between the performance of the generated solution and the best possible performance. The impact of the overestimation of the flights on the other hand is not yet clear.

2.1.6 – Stability of assignments in consecutive solutions

The Simulated Annealing optimization approach shuffles the starting solution around due to the high acceptance probability and the large number of swaps and moves at this high acceptance probability. Besides this shuffling, arrival time estimates of flights that will be arriving in the hour following the optimization moment are changing often. As the upcoming three hours of flights are considered in the problem, we want to know whether the assignments of these flights are stable over the results of consecutive SA optimization rounds. If the flights' assignments constantly change, this might be a sign that we do not benefit from incorporating a large time window.

To test the stability of flight assignments, we looked at the result of every optimization round and determined whether the flight's assignment differs from the result of the preceding optimization round. We further looked at the latest moment at which each flight changes belt, to assess the portion of flights

that will remain on the same belt at each time slot. Finally, we looked at the portion of flights in each time slot that are assigned to the belt that they will eventually be assigned to, the results are presented in Table 7.

Table 7: Statistics of change in assigned belts (n >1,000)

time window	Flights that change assignment with respect to the previous assignment	% of flights that change assignment in future optimization rounds at least once	% of flights assigned to their eventual belt at end of time window
$t_{arrival}$-9	51.2%	90.0%	37.3%
$t_{arrival}$-8	51.8%	88.8%	36.8%
$t_{arrival}$-7	52.4%	87.7%	38.6%
$t_{arrival}$-6	52.5%	86.4%	39.4%
$t_{arrival}$-5	53.0%	83.5%	40.3%
$t_{arrival}$-4	53.8%	79.1%	41.7%
$t_{arrival}$-3	56.0%	69.3%	43.9%
$t_{arrival}$-2	53.6%	47.3%	52.7%
$t_{arrival}$-1	45.1%	19.0%	83.1%
$t_{arrival}$	19.0%	2.2%	97.8%

We find that the individual flight assignments of consecutive optimization windows change a lot, and that therefore the output of consecutive optimization windows also change a lot. The fact that between 10 and 20 minutes before the arrival of the flight, more than half of the flights is still assigned to another belt than the one they will eventually be assigned to, calls into question the relevance of using data of the upcoming three hours' worth of flights.

Analyzing the on-block arrival times estimates of the aircrafts and the outcomes of the optimization rounds, we have found that measures were taken to prevent flights switching assignments during the optimization round in which they are expected to arrive on-block. For instance, if a flight is expected to arrive at 17:03, the optimization round starting at 17:00 is not allowed to move the flight to another belt. However, the airport does allow flights that are expected to arrive at 17:08 to change belt assignments in the optimization round starting at 17:00. This is due to the fact that the flight is expected to arrive after the optimization part of the optimization round starting at 17:00. This results in the fact that sometimes a flight is allowed to change assignment (because they are expected to arrive at 17:08), while they actually arrive during the optimization part of the optimization round (for instance, at 17:03). The flight will in reality be assigned to its assignment resulting from the optimization round of 16:50, but in the new optimization round it might get another assignment. This happens for the 2.2% of flights, shown in Table 7.

2.2 – Performance Measurement

First, in Section 2.2.1, we describe the optimization objective, as decided by the airport. In Section 2.2.2 we define the mathematical model formulation for the problem. In Section 2.2.3, we look at the performance of the current solution approach for the sub-problems created every optimization round. In Section 2.2.4, we look how well the method of solving the collection of sub-problems scores on the global problem, including the stochasticity.

2.2.1 – The optimization objective

The performance of a planning in ORTEC's current Simulated Annealing algorithm implementation is measured by the score of the objective function. The objective function is a sum of the following three components:

1. **The total number of minutes that overlapping flight combinations are handled at the same belt**. This results in the fact that if three flights are handled at the same belt, each two-way interaction ([A-B], [A-C], [B-C]) is penalized. So, for every minute four flights are on one belt, we incur a penalty of 3+2+1=6.

2. **The number of times the baggage of an alliance flight does not arrive at an empty belt (**times a tuning factor $\beta = 9$). This measure follows from the fact that the airport aims to provide additional services for alliance airlines. Airlines want to offer additional premium services to its customers, as this helps with service differentiation. If passengers pay additional fees for extra benefits, this also includes their baggage being put on the baggage belt first. According to the airport and the alliance airlines, the perceived level of luxury is then higher if the premium baggage arrives at an empty belt.

3. **<u>Minus</u> the number of times a flight gets assigned to one of the carrier's preferred belts (**times a tuning factor $\gamma = 8$). This component is also in the goal function to incorporate a preference. Each carrier (airline), works with one of the airports' two baggage handling companies. The baggage handling companies are used to working with their own set of belts for operational reasons, and the airport tries to incorporate these preferences as much as possible.

As overlap on belts, Alliance flights arriving at occupied belts and flights not arriving at their preferred belts are all undesired (the preferred belt term is negative in the objective function), we find that we want the objective function to be as low as possible. The airport itself has decided upon these components and their respective weights in the objective function and tried to incorporate the

preferences of the stakeholders in a fair way by doing so. Therefore, we will not be looking to change this objective function in this research.

2.2.2 – Mathematical model formulation

We first translate the problem into a mathematical formulation, so that a mathematical programming solver (IBM CPLEX 12.6 in our case) can use the formulation to solve the problem to optimality. To formulate the problem, the following notations are defined.

Indices

i, j Represent flights

b Represents belts

Sets

F Set of flights

B Set of belts

Parameters

β Penalty factor for when a belt is occupied at the start of an alliance assignment

γ Bonus factor for each flight that is assigned to one of its airline's preferred belts

s_i $\begin{cases} 1 & \text{if flight } i \text{ is from an alliance} \\ 0 & \text{otherwise} \end{cases}$

r_{ib} $\begin{cases} 1 & \text{if flight } i \text{ is fixed on belt } b, \\ 0 & \text{otherwise} \end{cases}$

l_{ib} $\begin{cases} 1 & \text{if flight } i \text{ is allowed on belt } b \\ 0 & \text{otherwise} \end{cases}$

a_{ib} $\begin{cases} 1 & \text{if belt } b \text{ is available during the time required by flight } i \\ 0 & \text{otherwise} \end{cases}$

p_{ib} $\begin{cases} 1 & \text{if belt } b \text{ is prefferable for flight } i \\ 0 & \text{otherwise} \end{cases}$

t_{ij} The total overlap time between flight i and flight j, appointed to the flight that starts latest.

t_{ij} $\begin{cases} \max\left(0, min\left(end_i, end_j\right) - max\left(begin_i, begin_j\right)\right) & \text{if } begin_i < begin_j \\ 0.5 * \max\left(0, min\left(end_i, end_j\right) - (begin_i)\right) & \text{if } begin_i = begin_j \\ 0 & \text{otherwise} \end{cases}$

In which $begin_i$ and end_i represent the (estimated) begin and the end time of the belt allocation for flight i respectively. These estimated begin and end time are input for the model.

$bigM$ Large positive constant, is (as least) as big as the largest t_{ij}

Variables

X_{ib} $\begin{cases}1 & \text{if flight } i \text{ is assigned to belt } b,\\ 0 & \text{otherwise}\end{cases}$

O_{ij} $\begin{cases}1 & \text{if flight } i \text{ is assigned to same belt as flight } j\\ 0 & \text{otherwise}\end{cases}$

V_i $\begin{cases}1 & \text{if flight } i \text{ arrives at an occupied belt}\\ 0 & \text{otherwise}\end{cases}$

Q_i $\begin{cases}1 & \text{if flight i is put on one of its prefferd belts}\\ 0 & \text{otherwise}\end{cases}$

The model

$$\min \sum_{i\in F}\sum_{j\in F} O_{ij}t_{ij} \;+\; \beta \sum_{i\in F} s_i * V_i - \gamma \sum_{i\in F} Q_i \qquad (1)$$

Subject to

$$\sum_{b\in B} X_{ib} = 1 \qquad \forall i \in F \qquad (2)$$

$$X_{ib} + X_{jb} - O_{ij} \leq 1 \qquad \forall b \in B, (i, j \neq i) \in F \qquad (3)$$

$$V_j * bigM \;\geq O_{ij} * t_{ij} \qquad \forall (i, j \neq i) \in F \qquad (4)$$

$$\sum_{b\in B}(X_{ib} * p_{ib}) - Q_i \geq 0 \qquad \forall\, i \in F \qquad (5)$$

$$X_{ib} \geq r_{ib} \qquad \forall b \in B,\; i \in F \qquad (6)$$

$$a_{ib} \geq X_{ib} \qquad \forall b \in B,\; i \in F \qquad (7)$$

$$l_{ib} \geq X_{ib} \qquad \forall b \in B,\; i \in F \qquad (8)$$

$$X_{ib} \in \{0,1\} \qquad \forall b \in B, i \in F \qquad (9a)$$

$$V_j \in \{0,1\} \qquad \forall j \in F \qquad (9b)$$

$$O_{ij} \in \{0,1\} \qquad \forall (i, j) \in F \qquad (9c)$$

$$Q_i \in \{0,1\} \qquad \forall i \in F \qquad (9c)$$

The objective function (1) aims at minimizing A) The sum of minutes that flights that are put on the same belt overlap, B) The number of times the baggage of an alliance flight does not arrive at an empty belt (times a tuning factor β = 9), and C) minus the number of flights that are assigned to one of their preferred belts (times a tuning factor $\gamma = 8$). Constraints (2) ensure that each flight is assigned to exactly one belt. Constraints (3) set the overlap variable O_{ij} to one if two flights are assigned to the same belt. Constraints (4) set the start-at-occupied-belt variables V_j to one if at least one other flight is still on the belt when and where flight j starts. If two flights start at an empty belt at the same time, they are considered both not to start at an empty belt. Constraints (5) set the preferred-belt variables Q_i to one, if flight i starts at one of its preferred belts. Constraints (6) ensure that required assignments will be

assigned. Constraints (7) ensure that belts must be available throughout the flights' needed timeslot to be assigned. Constraints (8) ensure that flights are allowed on the belt that they are assigned to. Constraints (9a-d) specify the binary nature of the variables.

Both formulations are implemented in C#, and IBM's CPLEX 12.6 will be used to solve the problem instances with a branch-and-bound technique. The maximum size of the branch-and-bound tree is 10GB. When the tree becomes larger, the branch-and-bound algorithm is terminated.

2.2.3 – SA performance based on the expected times

In this section we assess the performance of the SA approach, relative to that of the Mixed Integer Problem (MIP) solver approach. The performance in this section is measured based on the estimated times. In Section 2.2.4 we assess how well a solved problem instance (based on the airport's estimates) will perform when measuring performance based on the realization of the times.

For the comparison, we use a week of data. Seven problem instances of one day each are created and optimized using the SA approach and the MIP solver. We limit the SA run to one minute and the MIP solver to five minutes. For the SA approach we use five replications. The results are shown in Table 8. In each column, the best results are highlighted, and the objective function of the expected best SA replication is shown in the bottom row.

Table 8: Expected Simulated Annealing results using current estimates

Approach	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Total
MIP UB	-1,457	**-1,334**	**-1,403**	-1,539	-1,357	-1,492	-1,474	-10,056
gap	13.0%	2.1%	9.3%	21.5%	23.7%	14.8%	4.6%	
SA rep. 1	-1,458	**-1,334**	-1,399	**-1,551**	**-1,373**	-1,504	-1,475	-10,094
SA rep. 2	**-1,463**	**-1,334**	-1,401	-1,548	-1,364	-1,497	**-1,476**	-10,083
SA rep. 3	-1,460	**-1,334**	-1,402	-1,547	-1,369	**-1,506**	-1,468	-10,086
SA rep. 4	-1,458	**-1,334**	**-1,403**	-1,547	-1,369	-1,505	-1,475	-10,091
SA rep. 5	-1,457	**-1,334**	-1,402	-1,543	-1,372	-1,502	-1,474	-10,084
Best SA rep.	**-1,463**	**-1,334**	**-1,403**	**-1,551**	**-1,373**	**-1,506**	**-1,476**	-10,106

The MIP solver is never able to reduce the gap to 0%, and only for two out of the seven days, the MIP solver finds the same upper bound as the best SA replication. On the remaining five days, the solutions found by the MIP solver are generally worse than all the SA replications (except for day 7).

The SA approach does not structurally find the same solution values. It does however come close to the best-known solution on a structural basis. In the next section we will discuss whether extending the runtime of the SA algorithm will generate schedules with better performance when considering the realized times.

2.2.4 – Solution performance based on the realized times

To determine the performance based on the realized times, we change the times in the assignments created in Section 2.2.3 to the realized times, without changing the belt assignments. In Table 9, we show the realized results of the assignments represented in Table 8, including the realized value of the expected SA replication. Furthermore, in the bottom row, we show the correlation between the expected and the realized performance for a one-day problem instance.

We find that there is no clear relation between the expected objective value and the realized objective value for different SA replications of the same dataset. Because of this, and the expected results per day being relatively close to one another over different SA replications, there is no need to increase the run length of the SA algorithm.

Table 9: Realized Simulated Annealing results using current estimates

	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Total
MIP solver	-923	-736	-873	-1,044	**-991**	-1,156	-1,037	-6,760
SA rep. 1	-975	**-809**	-865	-1,054	-903	-1,165	**-1,165**	-6,936
SA rep. 2	**-1,072**	-761	-864	-1,014	-927	**-1,175**	-1,088	-6,901
SA rep. 3	-934	-753	-882	-998	-889	-1,119	-1,039	-6,614
SA rep. 4	-977	-739	-844	-942	-983	-1,062	-1,160	-6,707
SA rep. 5	-992	-802	**-897**	**-1,063**	-917	-1,226	-1,078	-6,975
Realized value of the expected best replication	**-1,072**	-773	-844	-1,054	-903	-1,119	-1,088	-6,853
Correlation with expected values	0.79	-	-0.18	-0.08	-0.68	-0.41	0.52	0.19

Next, the assignments in the problem instances are optimized based on the realized times to determine how good the approaches could have done if estimates were to be perfect. We limit the MIP approach to five minutes, and for the SA approach, we use the maximum value of five replications of one minute each. The results are shown in Table 10. Notable is the performance of the MIP solver: for each of the problem instances it finds values at least as good as the SA approach, and for four out of the seven days it finds the optimum solution. In all four of the cases, the SA approach also manages to find the optimum solution value. For the remaining three problem instances, there is very small difference between the MIP solver and the SA approach.

The difference between the expected objective function from the optimization based on the expected times (-10,056 for the MIP solver) and the realized objective function from the optimization based on the realized times (-11,398 for the MIP solver), can be explained by the fact that in the used estimates, the on-belt duration is deliberately overestimated.

Table 10: Optimization results based on the realized times

Approach	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Week total
MIP UB	**-1,628**	**-1,350**	**-1,513**	**-1,859**	**-1,698**	**-1,751**	**-1,599**	**-11,398**
gap	0%	0%	3.0%	7.5%	1.5%	0%	0%	
SA	**-1,628**	**-1,350**	**-1,513**	-1,858	-1,696	**-1,751**	**-1,599**	-11,395

In Table 11 we break down the goal function for the MIP solver outcomes, for the assignments generated based on the expected times (corresponding to Table 8 and Table 9), and the assignments generated based on the realized times (corresponding to Table 10).

Table 11: Objective function breakdown for the expected results, realized results and best possible known solution results

		Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Week
Expected performance of MIP solver	Total	-1,457	-1,334	-1,403	-1,539	-1,357	-1,492	-1,474	-10,056
	Overlap minutes	174	90	154	284	271	156	91	1,220
	Pref. belt bonus	-1,784	-1,496	-1,656	-2,048	-1880	-1,792	-1,664	-12,320
	Alliance empty	153	72	99	225	252	144	99	1,044
Realized performance of MIP solver	Total	-923	-736	-873	-1,044	-991	-1,156	-1,037	-6,760
	Overlap minutes	366	337	396	419	421	276	267	2,482
	Pref. belt bonus	-1,784	-1,496	-1,656	-2,048	-1880	-1,792	-1,664	-12,320
	Alliance empty	495	423	387	585	468	360	360	3,078
Best known solution based on realized times	Total	-1,628	-1,350	-1,513	-1,859	-1,698	-1,751	-1,599	-11,398
	Overlap minutes	82	74	119	106	84	35	34	534
	Pref. belt bonus	-1,800	-1,496	-1,704	-2,064	-1,872	-1840	-1,696	-12,472
	Alliance empty	90	72	72	99	90	54	63	540

Compared to the expected times, the penalty incurred for overlap minutes when considering the realized times is twice as high. For the penalty for alliance flights not starting at empty belts, this is around three times as high. The objective of flights being assigned to their preferred belt per definition does not change.

In Appendix C, we illustrate the problem by showing the different schedules for one morning worth of flights for the expected best solution, the realization of that expected best solution, and the actual best solution.

Comparing the objective function breakdown in Table 11 with the optimality gaps shown in Table 8 and Table 10, a strong correlation can be observed between the optimality gap of the MIP and the overlap in the resulting schedule (Pearson's correlation = 0.96). Also, there is a strong correlation between the optimality gap and the penalty for alliance flights not starting at empty belts (Pearson's

correlation = 0.96). This could mean that for MIPs with large optimality gaps either the upper bound of the MIP is far from the optimal solution (so, in the optimal solution the gap would be smaller, and so would be the resulting minutes of overlaps and the number of flights not starting at empty belts). Another explanation is that when there is more overlap in the optimal schedule, the MIP solver has trouble increasing the lower bound.

2.3 – Conclusion

In Section 2.1, we have explored the structure of the input data, the estimation process of the values, and how the data is used to generate solutions. We also looked at the stochasticity involved in these estimates, and at the difference between the expected times and the realized times. We found that there is a lot of uncertainty involved in the processes that are being used to optimize the baggage belt assignments. The estimated arrival times structurally worsen around half-an-hour prior to the actual arrival of the planes, as the bias in the estimates jumps from -0.12 to +2.56 minutes. Furthermore, the estimates for the transportation times are so far off that there is no correlation between the baggage's expected transportation time and the realized transportation time. The potential of the estimators used to determine the transportation time was found to be low.

Due to all the uncertainties in the planning, it is hard to estimate for which flights the baggage will be on the baggage retrieval belts at the same time, and for which flights this will not be the case. Out of all flight pairs that are expected to overlap, less than 50% of the flight combinations will actually overlap, and from the overlapping flight combinations, only 60% of combinations were expected to do so. This means that even though problem instances are solved perfectly, results can end up poor due to the problem instances using time estimates that do not reflect reality close enough.

In Section 2.2, we introduced the optimization objective that is being used by the airport. Based on this objective, the MIP formulation was introduced and solved using IBM's CPLEX (12.6). For more congested problems (problems with more overlapping flights), the optimality gap could not be reduced to 0% within the allowed calculation time, and often the SA approach found better solutions in less time. CPLEX was able to solve the problem instances using the realized times to create assignments, and in these cases the SA approach often also found the optimum result. When comparing the expected performance of the assignment algorithm with the realized performance, we found that there is indeed a large difference between the two. There is also a large difference between the realized performance and the best possible performance.

In the coming chapters, we will need to find a way to increase the number of flight pairs that will be correctly classified as overlapping while making sure that not too many flights will be marked as overlapping while they are not overlapping in order to reduce the gap between the expected values and

the realized values of the assignments, and thereby reducing the gap between the realized values and the best possible assignment values. Considering all flight combinations to have overlapping time windows can be just as bad as considering no flight combinations at all to have overlapping time windows. So, besides finding methods for determining whether flights overlap, we will also need to find a balance between over- and underestimation of the overlap.

Chapter 3: Literature Review

In this chapter we study the literature to find airport related problems that are linked to the baggage belt assignment problem in some way, in Section 3.1. Next, we look at a set of traditional problems related to our problem formulation and the solution methods for these problems in Section 3.2. Finally, we look at methods to incorporate stochasticity into combinatorial optimization problems in Section 3.3.

3.1 – Airport ground handling problems in literature

A large amount of research has been done on the optimization of airport operations. In this section we have a look at the two problems most related to our problem. The first is the belt allocation problem itself. The second is the gate assignment problem.

Belt allocation problems

Frey et al. (2017) claim to be the first to provide a mathematical model formulation and a solution approach for a belt allocation problem in inbound baggage handling. The goal is balancing usage of the baggage carousels and minimizing passenger waiting times. In the paper, they describe a baggage handling system with multiple remote infeed stations, each being able to reach a set of baggage retrieval belts, as well as the possibility to put the baggage on a direct infeed station that can only reach one belt (which takes considerably less transportation time). Moreover, they incorporate the passengers traveling distance to the belt, to ensure passengers do not arrive at the retrieval belt much earlier than their baggage (resulting in waiting time) or much later (resulting in congested baggage belts). For the solution approach, an initial solution is constructed by first sorting flights based on expected arrival time at the infeed stations, and for each combination of infeed station and baggage retrieval belt, the expected impact on belt occupation and waiting times are calculated. Then, the assignment is improved using randomized local search.

Frey et al. consider the capacity of the belts, the intensity of the baggage retrieval according to the modeled number the passengers around the belt, and the transportation time of the baggage due to the number of bags on board of the aircraft and the capacity of a transportation container.

In our problem, only rough estimates on the number of bags are available, based on the baggage class of the aircraft. We also estimate the time of the arrival of the baggage at the baggage belts and the time it takes to put the baggage on the belt, but less in detail. Whereas Frey et al. consider the supply capacity and the retrieval intensity, we use a fixed duration per flight baggage class. In addition to Frey et al., we also consider belt preferences for certain flights, and awarding a bonus for starting flights at empty belts for stakeholder management. Besides, we use estimates to create schedules and evaluate

performance based on realized times, instead of optimizing the realized times, thereby having to incorporate the effects of stochasticity.

Gate assignment problem

A widely studied field within airport planning is the Gate Assignment Problem. In this assignment problem, gates are assigned to aircrafts. Earlier variants of the problem focus solely on minimizing the walking distance for passengers to improve passenger satisfaction, now referred to as the Static Gate Assignment Problem. Later, as the number of flights increased, the idle time of the gate, the negative effects of flight delay, and gate conflicts are incorporated in the problem. The Gate Assignment Problems that also focus on the performance of the assignment under uncertainty are referred to as Robust Gate Assignment Problems (Deng et al., 2016).

Babic et al. (1984) used a branch-and-bound approach to reduce the walking distance on an airport for arriving and departing passengers only, which resulted in planes with more passengers being assigned to the most central gates. Mangoubi & Mathaisel (1985) formulated the Static Gate Assignment Problem including transfer passengers, assuming a transfer passenger is equally likely to board his next flight at any gate. Haghani & Chen (1998) incorporated actual passenger transfer flows, creating a Quadratic Assignment Problem, and added time constraints to the problem. As the Quadratic Assignment Problem is NP-Hard, Haghani and Chen chose to use a greedy constructive heuristic approach for the assignment of gates.

Bolat (2001), introduced a model with the objective of minimizing the variance of idle times to create assignments as robust as possible, and solves the model using a Genetic Algorithm. Many variants have been introduced since this first robust approach. Amongst those are the approach of Kim & Feron (2011), who introduced a variant maximizes that the time gap between consecutive flights at each gate with the goal of minimizing gate conflicts and Deng et al (2016), who formulated the Robust Gate Assignment Problem incorporating a measure for balanced idle time for each gate and reducing the walking distance of passengers, and solved the problem using CPLEX.

At the airport in our problem, we do not consider the assigned gate when determining the transportation times for the baggage, but we use the assigned aprons as transportation time estimators. Each apron is a set of aircraft parking spaces, and each parking space is linked to a gate. If all gates linked to the apron are occupied, it might happen that a flight's assigned apron needs to change due to stochastic arrival times of itself or other flights. As mentioned in Section 2.1.2, only about 1% of the flights were reassigned to another apron within one hour prior to arrival in the considered week and therefore we neglect this factor. If the airport's gate assignments (and as a result the apron assignments) would be less stable, the impact of reassignment of aprons would be larger and our baggage belt assignment

problem would become more complicated. Vice versa, the baggage retrieval assignment problem does not impact the gate assignment problem, as all baggage retrieval belts are in the same area.

Although the gate assignment problem and the baggage belt assignment problems might seem somewhat similar in the sense that we assign flights to a processor (either a gate or a baggage belt), with a quadratic penalization function (either by overlap or walking distance), the actual problems differ a lot. First, there are a lot more gates than there are baggage belts, resulting in the fact that we need to assign flights of multiple gates to the same baggage belt. The time between consecutive flights at the same gate is therefore on average larger than the time between consecutive flights at the same belt. The same minute of delay of a flight therefore has a larger probability of resulting in overlap at a belt than at a gate. however, overlap at a gate will cause an infeasible schedule, whereas overlap on the belt will only be penalized. Despite the overlap penalization, in the baggage belt assignment problem it might be better to plan flights to overlap.

3.2 – Traditional optimization problems

In our problem, we must assign flights to baggage belts. This is some form of an assignment problem. The first appearance of the name 'assignment problem' (AP) dates to a paper from Votaw and Orden in 1952. However, in 1946 a first algorithm was proposed by Easterfield to solve the classical version of the AP, this classic version is also referred to as the Linear Sum Assignment Problem (LSAP) (Burkard, Dell'Amico, Martello, 2012). The problem is finding the lowest cost assignment between n tasks and n agents, with one task per agent and one agent per task. The costs of assigning an agent to a task are defined by an $n \times n$ cost matrix $C = (c_{ij})$. The LSAP can also be formulated in terms of graph theory. Define a bipartite graph $G = (U, V; E)$ with a vertex of U for each row, a vertex V for each column and cost c_{ij} associated with edge $[i,j](i,j = 1,2,\ldots,n)$. The problem is then to determine a minimum weight perfect matching in G.

In 1955, Kuhn published the Hungarian Method, an algorithm that can solve the LSAP in polynomial time: $O(n^4)$. It was later found that this could be further reduced to $O(n^3)$ by Edmonds and Karp (1972).

The Generalized Assignment Problem

Since its introduction, many variations on the LSAP have been introduced. One of these variations is the Generalized Assignment Problem (GAP) first presented by Ross and Soland (1975). For the GAP, the goal is finding the lowest cost assignment between n tasks and m agents, with multiple tasks per agent and one agent per task. In the airport's case, the tasks translate into flights and the agents into the baggage retrieval belts. The GAP is NP-Hard (Fisher et al, 1986) and the simpler problem of determining the existence of a feasible solution for GAP is NP-complete as the decision version of the

bin packing problem can be reduced to this problem and therefore can only be solved to optimality within reasonable time for small problem instances (Martello and Toth, 1990).

The mathematical formulation of the GAP is:

$$\text{Minimize} \quad \sum_{i\in I}\sum_{j\in J} c_{ij}\, x_{ij} \tag{1.1}$$

$$\textit{Subject to} \quad \sum_{j\in J} r_{ij} x_{ij} \le b_i \qquad \forall i \in I \tag{1.2}$$

$$\sum_{i\in I} x_{ij} = 1 \qquad \forall j \in J \tag{1.3}$$

$$X_{ik} = \begin{cases} 1, & if\ process\ i\ on\ processor\ k \\ 0, & otherwise \end{cases} \qquad \forall j \in J,\ \forall k \in K \tag{1.4}$$

In which I = {1,2,…,*m*} is a set of agent indices, J = {1,2,…,*n*} is a set of task indices, c_{ij} are the costs of assigning agent *i* to task *j*, r_{ij} are the resources required by agent *i* to perform task *j* and b_i is the amount of resources available to agent *i*. The goal is to minimize the total costs incurred by the assignments. Constraints (1.2) ensure capacity of the agent is not exceeded, and constraints (1.3) make sure that every task is assigned to an agent. Constraints (1.4) are binary constraints for the decision variables.

In the airport's case, assigning a flight to a specific belt will only incur costs (or miss a bonus, according to the formulation in Section 2.2.1) if the flight is not assigned to its preferred belts. The rest of the objective function depends on flights being on the same belt at the same time.

The Process Allocation Problem

In 1969, Chu introduced a problem was that later called the Process Allocation Problem (PAP) (Sofianopoulou, 1990). This problem followed from the distributed system advances in computer hardware and software. In a distributed system, hardware consists of network of processors, and software consists of number of processes. Each software process is assigned to a hardware processor, with limited availability of space on each hardware processor. Costs incur from the assignment of a process to a processor, and from the time it takes for different processes to communicate between different processors. If processes are assigned to the same processor, this communication time is considered negligible.

The mathematical formulation of the PAP is:

$$\text{Minimize} \quad \sum_{i=1}^{N}\sum_{k=1}^{M} e_{ik} X_{ik} + \sum_{i=1}^{M-1}\sum_{j=i+1}^{M}\sum_{k=1}^{N}\sum_{\substack{q=1 \\ q\neq k}}^{N} c_{ij} X_{ik} X_{jq} \tag{2.1}$$

$$\textit{Subject to} \quad \sum_{i\in I} r_i X_{ik} \le b_k \qquad \forall k \in K \tag{2.2}$$

$$\sum_{i\in I} X_{ik} = 1 \qquad \forall j \in J \tag{2.3}$$

$$X_{ik} = \begin{cases} 1, & if\ process\ i\ put\ on\ processor\ k \\ 0, & otherwise \end{cases}, \qquad \forall j \in J,\ \forall k \in K \qquad (2.4)$$

In which $i,j = \{1,2,\ldots,M\}$ is a set of processes, $k,q = \{1,2,\ldots,N\}$ is a set of processors, e_{ik} are the costs of assigning process i to processor k, c_{ij} are the costs of putting processes i and j on different processors, r_i are the resources required by process i and b_k is the amount of resources available by processor k. Constraints (2.2) ensure capacity of the processor is not exceeded, and constraints (2.3) make sure that every process is assigned to an agent. Constraints (2.4) are binary constraints for the decision variables. With the introduction of a new variable $Y_{ikjq} = X_{ik}X_{jq}$, and additional constraints the problem can be linearized. The problem then becomes:

Minimize $\sum_{i=1}^{N}\sum_{k=1}^{M} e_{ik}X_{ik} + \sum_{i=1}^{M-1}\sum_{j=i+1}^{M}\sum_{k=1}^{N}\sum_{\substack{q=1 \\ q\neq k}}^{N} c_{ij}Y_{ikjq}$ (2.5)

Subject to Constraints (2.2), (2.3), (2.4), and additionally:

$$X_{ik} + X_{jq} - 1 \leq Y_{ikjq} \quad \forall (j, i < j) \in J\ ,\ \forall (k, q \neq k) \in K \qquad (2.6)$$

$$Y_{ikjq} = \{0, 1\} \qquad (2.7)$$

Considering the airport problem, the processors in the PAP translate into baggage belts and the processes translate into flights' baggage. The assignment costs can then be considered the (absence of the) preferred belt bonus, and the costs of communication between the processors are translated into the overlap between two flights. The difference here is that in the airport's problem, the costs are incurred if flights are assigned to the same belt, while in the case of the PAP, costs are incurred if processes are assigned to different processors. The airport's problem and the PAP can be formulated equivalent when we do not consider the airport's additional non-empty belt penalty in the objective function described in Section 2.2.1, by setting the required resources r_i to 0 and making the costs c_{ij} in the goal function (constraints 2.1 and 2.5) negative. Doing so, process-pairs that interact (flight-pairs that overlap) will now be rewarded if they are assigned to different processors.

For three processors, the PAP is NP-complete, as the minimum 3-cut problem is NP-complete (Dahlhaus et al., 1987) and can be polynomially transformed to the three-processor problem (Magirou, Milis 1988). Bokhari (1981) found earlier that the PAP is NP-complete for four or more processors.

For the PAP, a greedy clustering algorithms was applied by Chu et al. (1980), a local search method was applied by Price (1981), a SA approach by Sofianopoulou (1992) and a memetic algorithm by Vigo and Maniezzo (1997). No performance comparison was conducted between these approaches.

The Generalized Quadratic Assignment Problem

Seven years after the last paper on the PAP (Vigo, Maniezzo, 1997), the Generalized Quadratic Assignment Problem (GQAP) was introduced (Lee and Ma, 2004) as a generalization of the Quadratic Assignment Problem. The problem is to optimally assign *m* pieces of equipment to *n* locations, in which parts must be moved between the equipment to perform a sequence of operations for each part. Each piece of equipment must be assigned to exactly one location and multiple pieces of equipment can be put on one location. The goal is to reduce the total costs, consisting of assignment costs and transportation costs.

The mathematical formulation for the QGAP is:

$$\text{Minimize} \quad \sum_{i=1}^{N}\sum_{k=1}^{M} e_{ik}X_{ik} + v\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{k=1}^{N}\sum_{q=1}^{N} c_{ij}d_{kq}X_{ik}X_{jq} \tag{3.1}$$

Subject to Constraints (2.2), (2.3) and (2.4).

In which $i,j = \{1,2,\ldots,M\}$ is a set of equipment, $k,q = \{1,2,\ldots,N\}$ is a set of locations, e_{ik} are the costs of assigning equipment i to location k, c_{ij} is the flow volume from equipment i to equipment j, d_{kq} is the distance between location k and location q. Parameter v is the travel cost per unit distance per unit flow volume. If we set $v = 0.5$, $d_{kq} = 1$ for all combinations in which $k \neq q$, and make c_{ij} the sum of the unit flow from equipment i to equipment j and from equipment j to equipment i, then the GQAP is equal to the PAP. The PAP is therefore considered a special class of the GQAP (Hahn, 2007).

For the GQAP, amongst others, memetic algorithms have been applied (Cordeau et al., 2006), GRASP approaches with Path-relinking (Mateus et al., 2011), (Silva et al., 2013), (Morán-Mirabal et al., 2013), TABU-search (McKendall and Li, 2017) and TABU-search variations with SA (Gunawan et al, 2014) and biogeography optimization (Lim et al., 2015) have been applied. A comparison by Beham et al. (2018), shows that GRASP-algorithms with path relinking, Iterated Local Searches, and Genetic Algorithms perform well on the GQAP.

The Quadratic Multiple Knapsack Problem

In 2006, a problem very similar to the PAP was introduced: The Quadratic Multiple Knapsack Problem (QMKP) (Hiley and Julstrom, 2006). The QMKP followed as a generalization and combination of the multiple knapsack problem (Hung and Fisk, 1978) and the quadratic knapsack problem (Gallo et al. 1980). The QMKP is NP-Hard (Hiley and Julstrom, 2006).

The mathematical model formulation of the QMKP is:

$$\text{Minimize} \quad \sum_{i=1}^{N}\sum_{k=1}^{M} e_{ik}X_{ik} + \sum_{i=1}^{M-1}\sum_{j=i+1}^{M}\sum_{k=1}^{N}\sum_{\substack{q=1 \\ q\neq k}}^{N} c_{ij}X_{ik}X_{jq} \tag{4.1}$$

$$\textit{Subject to} \quad \sum_{i\in I} r_i X_{ik} \leq b_k \qquad \forall k \in K \tag{4.2}$$

$$\sum_{i \in I} X_{ik} \leq 1 \qquad \forall j \in J \qquad (4.3)$$

$$X_{ik} = \begin{cases} 1, & if\ process\ i\ put\ on\ processor\ k \\ 0, & otherwise \end{cases}, \qquad \forall j \in J,\ \forall k \in K \qquad (4.4)$$

This formulation is almost equal to that of the PAP, except for the assignment constraints (constraints 2.3 and constraints 4.3), For the PAP, these constraints ensure that each task is assigned exactly once to agent. For the QMKP, these constraints ensure that each task is assigned to at most one agent. In the airport's problem, we need all tasks (flights) to be assigned to exactly one agent (belt). If flights were to be assigned to at most one agent, then flights with a negative impact on the goal function would simply not be assigned.

Upon introduction of the QMKP, Hiley and Julstrom applied a greedy algorithm, a local search algorithm (stochastic hill climbing), and a genetic algorithm approach. The stochastic hill climbing approach gave the best results. After the introduction, many nature-inspired inspired metaheuristics being applied. Amongst others, two genetic approaches (Singh, Baghel, 2007), (Sarac, Sipahioglu, 2007), an Artificial Bee Colony algorithm (Sundar, Singh, 2010) and an Evolutionary Path Relinking approach (Chen et al., 2015). Each outperforming the algorithms that were published before them, for a majority of the tested problem instances.

The PAP, QGAP and QMKP all have in common that they work with capacity constraints and that separation of related tasks over different agents is penalized (or equivalently assigning related tasks to the same agent is rewarded), instead of the assignment of related tasks (overlapping flights) to the same agent (belt), like in the airport's case. Due to the latter, the capacity constraints cannot simply be set to a very large number, as for the abovementioned problem formulations (PAP, QGAP and QMKP) this would mean that all tasks would be assigned to the same agent. Of course, this behavior will change if the costs are set to negative numbers or we set the goal functions to maximize, but to the best of our knowledge, solution approaches for the maximization variants are not reported. Solution approaches that are successful in clustering related tasks may not be as successful in finding the best configuration that separates related tasks, as they are designed to do the exact opposite.

3.3 – Stochastic optimization

As was found in Chapter 2, a flight's arrival on-block time, its baggage transportation time, and its on-belt time duration are not known with 100% accuracy at the time a flight is assigned to a belt. As the processes are not fully predictable, the resulting on-belt time window of a flight is stochastic, therefore the overlap between two flights is stochastic and whether or not a flight starts at an empty belt is stochastic. In this section we look at three different methods that deal with the stochastic nature of the airport's problem. We start off by looking at a simple way to mitigate the impact of variability by adding

time buffers. Next, we look at the usage of expected values of stochastic processes. Third, we explore the concept of simheuristics.

Adding time buffers

A simple way to deal with stochasticity is to add some sort of buffer to schedules: so-called safety times. In scheduling, safety times can be determined based on the wanted service level of a process when the distribution of uncertainty is known (Baker, 2009). However, still a decision must be made on the wanted service level. In more complex systems, policies for the safe time can be simulated to assess their performance (Lambrecht et al., 1985). This approach has the advantage that it can be used without knowing anything about the distribution of the stochastic times. The trade-off to be made is that between disruption costs and safety time costs.

For our airport's problem, the disruption costs consist of the penalty of overlap between two flights and the penalty of flights blocking the empty arrival of other flights. For the safety time costs, we considered the opportunity costs, of for instance not placing a flight on one of its preferred belts due to expected overlap on that belt.

In systems where the impact of disruption is large (for instance, loss of feasibility), maximizing the safety times and minimizing the standard deviations of these safety times can be goals of the optimization. Examples of this are found for the Robust Gate Assignment Problem Kim and Feron (2011), which we saw in Section 3.1, and for the Berth Allocation Problem by Rodriguez-Molins et al. (2014).

Deterministic optimization methods using expected values

Bianchi et al. (2008) suggest the use of metaheuristics suitable for Deterministic Combinatorial Optimization Problems (DCOP), for solving the Stochastic Combinatorial Optimization Problems (SCOP), to which our problem belongs. The SCOP formulation involves the computation of one or more expected values for evaluating the objective function. There are three scenarios when assessing the expected values:

1. Closed-form expressions (which are expressions in terms of mathematical functions and operations) for the expected values are available, and the objective function is computed exactly based on these objective values
2. Closed-form expressions are available, but the objective function is considered too time consuming to be always computed during optimization.
3. The problem is so complex that no closed-form expression exists, therefore the objective function must be estimated by simulation.

In Appendix D we work out the possibility of using a closed-form expression, and find that this is not possible for our problem at hand. Considering the three scenario's, without the closed-form formulation for the expected overlapping periods, a simulation of these overlapping periods is needed to incorporate the problem's stochasticity. This can easily be done using Monte Carlo (MC) Simulation. The description of the needed random variables has already been done in Section 2.1. Using the MC approximations and solving the resulting deterministic problem is also known as "sample path optimization" or "sample average approximation" (Homen-de-Mello, 2003).

Due to the use of simulation there is a degree of uncertainty in the estimates, and therefore a statistical comparison is needed to determine whether one solution is better than another. Alkhamis et al. (1999) incorporate this in a Simulated Annealing algorithm, where the sample average of the current solution is compared with the estimate of the optimal solution, with a number of samples increasing linear with the number of iterations, Homem-de-Mello (2003) also incorporated simulated expected values in a Simulated Annealing algorithm by comparing the sample average of the current solution with the estimate of the optimal solution, but based the number of samples on a statistical test.

According to Binachi et al (2008), it is of high importance to investigate the performance of the used heuristics with respect to the level of stochasticity and to compare the performance of the metaheuristic with the performance of other available algorithms for the combinatorial optimization problem. These alternative algorithms will be covered in Chapter 4.

The method seems appropriate to use in the situation of the airport, as the method focusses on the expectations of the interaction between flights, and this interaction between flights, is what is being penalized in the goal function. Regarding the statistical comparison between different solutions, for the method by Alkhamis et al. (1999), we must calculate interactions at every neighborhood operator. For the method by Homem-de-Mello (2003) we can calculate each interaction before the optimization starts. Therefore we can use these expected overlap values also in other optimization techniques. We will therefore incorporate this method, in which the number of Monte Carlo replications is based on a statistical test up front.

Simheuristics

The term simheuristics was first introduced in 2014 by Juan et al. Simheurstics are an extension to the metaheuristic approach in which simulation is used to solve SCOPs. Simheuristics rely on the facts that the deterministic variant of the SCOP can be solved efficiently and that a strong correlation exists between high-quality solutions for the deterministic and the stochastic version of the combinatorial optimization problem (Juan et al., 2015).

First the SCOP is translated into a DCOP, by for instance replacing all random variables by their expected values. Then, a metaheuristic is used to find promising solutions for the resulting DCOP fast.

The algorithm decides on whether the solution is promising or not. Bad solutions are rejected, and the promising solutions are evaluated by a small simulation (like a MC simulation) that incorporates the stochasticity of the SCOP. When a stopping criterium is met, the elite solutions (a subset of all solutions evaluated by simulation) are evaluated by a more extensive simulation. The outcomes of the simulation are then used to perform a risk analysis on these elite solutions.

Applications of the simheurstic include a permutation flow shop problem with random processing times, in which a Greedy Randomized Adaptive Search was used to determine production schedules and simulation was used to determine the makespan of the schedules (Juan et al., 2014). Furthermore, a Multi-depot Waste Collection Problem with stochastic demands, in which an Iterated Local Search heuristic is used to assign waste containers to depots, and simulation is used to generate demands that will result in the costs of the routes (Gruler et al., 2017). A third application is found in a Capacitated Facility-Location Problem with stochastic demands, with applications for the e-Commerce industry, in which an Iterated Local Search heuristic is used to assign customers to depots and a simulation is used to generate demand and multiple sources of costs (Pages-Bernaus, 2019).

The simulation of stochastic data in the simheuristic approach enables the user to assess the impact of stochasticity on the complicated interactions in the SCOP. In the airport's problem, these interactions are a lot less complicated as there are no capacity restrictions that can be violated, and a flight only impacts the performance of flights with which there is direct overlap. It also requires a strong correlation between high-quality solutions for the deterministic and the stochastic variant of the combinatorial optimization problem, and this is not the case in our problem, as was shown in Section 2.2.4. Therefore, we will not use simheuristics for our problem.

3.4 – Conclusion

Very little research has been published about solution approaches on belt allocation in inbound baggage handling. Only Frey et al. (2017) proposed a mathematical model formulation and a solution approach for the belt allocation problem, but their problem focusses on optimizing transportation times and on-belt times of the inbound baggage handling, while our problem has a different optimization goal (defined in Section 2.2.1) and uses estimates for these transportation times and on-belt times. Furthermore, they do not pay attention to the discrepancy between the expected and the realized times.

A problem related to the baggage belt assignment problem is the gate assignment problem, for which stochasticity has become an important factor in the optimization of the problem. Although there seem to be some similarities between the gate assignment problem and our belt allocation problem, the optimization goals are different, and so is the impact of stochastic behavior on the solutions and their feasibility. The assignment of gates will therefore not help us with the assignment of belts. However,

the airport's gate assignment does have an impact on our belt allocation problem. We need the apron assignments, which follow from the gate assignments, to be robust, for the transport time estimates to be stable.

In terms of classical problems, our problem can be formulated as a maximization variant of the Process Allocation Problem (or the Generalized Quadratic Assignment Problem, of which the PAP is a special case) with side constraints. Although a broad range of solutions are offered for the minimization problem, solution approaches for the maximization variant, which would be more relevant for our problem, are not reported. In these minimization variants, the solution approach can focus on clustering related processes (like software processes that would have a large communication time if assigned to different hardware processes), but for the maximization variants it makes no sense to cluster flights that are non-overlapping, as most flight combinations are non-overlapping.

Stochasticity can be incorporated into the solution approaches in multiple ways. The most basic approach is adding a fixed amount of safety time to each on-belt duration. Another way of incorporating stochasticity is using MC simulations to estimate the expected overlap between a flight pair and use these estimates in our optimization. A third possibility is using a simheuristic approach, in which promising solutions are generated using a deterministic optimization approach and the quality of each of the solutions is assessed based on a large number of simulations of the time durations.

As in the airport's problem flights only impact other flights with which there is direct overlap, having better estimates of this overlap and the blocking probabilities will be very beneficial. Therefore, we will continue with the implementation of the MC simulation to estimate the overlaps. Furthermore, we will investigate performance of the optimization models for different levels of safety times, as this is a less intensive method to cope with stochasticity, and the resulting schedules are straightforward. We will not use the simheuristics approach, as it relies on the fact that a strong correlation exists between high-quality solutions for the deterministic and the stochastic variant of the combinatorial optimization problem, and this does not hold for our problem, as was shown in Section 2.2.4.

Chapter 4: Solution Approaches

In Section 4.1, we discuss the heuristics implemented to solve the problem instances. In Section 4.2, we explain how to find the expected overlap between two flights when considering the stochasticity of the estimates for an individual flight. In Section 4.3, we explain how to incorporate these MC overlap estimates in the optimization methods. In Section 4.4, we compare the performance of the methods based on historical data.

4.1 – Heuristics to be assessed

As previously noted, the current SA approach finds good solutions for situations where the realized times equal the expected times. The performance for the realization of these times is not necessarily good. Besides the currently implemented SA approach, that we discuss in Section 4.1.1, three new approaches to be implemented and included in the experiments are introduced. These new approaches are a First Come First Served appointment strategy in Section 4.1.2, a greedy appointment strategy based on a regret factor in Section 4.1.3, and a First Come First Served strategy that includes future demand in Section 4.1.4.

4.1.1 – Simulated Annealing

The simulated annealing algorithm is the algorithm currently being used in operations. It was already discussed in Chapter 2.1.3, but now we will have a closer look at the workings of the algorithm. Figure 17 shows the pseudocode of the algorithm, and to run it, the following input is needed:

- List of all flights: *AllFlights*, and for each *flight* that is part of *AllFlights*:
  - a mandatory belt assignment, if applicable: *flight.mandatoryBelt*
  - the preferred belts for the flight, if applicable: *flight.preferredBelts*
  - whether or not the flight is part of an alliance: *flight.isAlliance*
  - belts that are closed during a flight's on-belt time: *flight.closedBelts*
  - a lookup table of flights that overlap with the flight, and how much:
    - *flight.overlapDurationDictionary* that consists of:
      - Key: *flight2.Id*
      - Value: the minutes of overlap
  - a list of flights that are expected to be on belt at the start of the flight's on-belt window
    - *flight.FlightsThatBlockMe*
  - a list of alliance flights that are expected to arrive during the flight's on-belt window
    - *flight. AllianceFlightsBlockedByMe*
- The weights of the component of the objective function:

- *allianceOccupiedBeltPenalty*
- *preferredBeltBonus*
- *penaltyPerMinuteOverlap*

- List of all belts: *AllBelts*, and for each *belt* that is part of *AllBelts:*
  - a list of flights that is assigned to the belt. *belt.assignedFlights*
- Parameters for the SA algorithm:
  - The allowed run length *allowedCalculationTime*
  - A starting temperature *startingTemperature*
  - An ending temperature *endingTemperature*
  - A temperature update count *temperatureUpdateCount*

The algorithm starts with creating an initial solution by fetching the previous assignment results and randomly assigning the new flights (lines 2 and 3 in Figure 17). The starting values are then initialized for the temperature, the starting solution value and therefore the best-known solution value (lines 3-8). Next, the algorithm will keep evaluating the impact of assignment changes, until the prespecified optimization time has elapsed.

At each iteration (starting at line 10), it is determined at random which neighborhood operator is used: moving a flight to another belt or swapping the assignment of two flights. The swap mutator and the move mutator have an equal probability of being picked. If the move mutator is used, a random flight and a random new belt are picked. If the swap mutator is used, first a random flight is picked, and for this flight a list is created containing all flights that overlap with the randomly picked flight. Next, for each flight in the list, again all overlapping flights are determined. Out of all these flights that either directly overlap with the random flight or overlap with the flights overlapping with the random flights, a second flight is picked. If the list of overlapping flights is empty, a random second flight is picked out of all flights. These flight selections are carried out in the `CreateNeighbourSolution` method that is called in line 12. In line 13, the method `CalculateChangeInObjective` is called, that will determine the impact of the move or swap on the objective function. This method is described in detail in Appendix E. If the mutation results in an improvement or an equal objective, the change is always accepted (lines 14 and 15) and if the new solution value is better than the best known, the resulting assignments should be saved (lines 16 and 17).

If the mutation results in an objective worse than that of the current assignment, the change is accepted with a probability based on the change in objective and the temperature of the algorithm (lines 20 and 21). At the start, this temperature is high, and the probability of a worse solution being accepted is high. As the algorithm progresses, this temperature will decrease, and therefore the same negative impact on the objective function is less likely to be accepted. Next, if the change is accepted, the move or swap is

made (line 23 to 25) and the resulting assignments are saved in case a new best solution was found (lines 26 to 29).

```
1.  calculationStartTime ← DateTime.Now
2.  FetchPreviousAssignment()
3.  AssignAllUnAssignedFlightsToRandomBelt()
4.  InitializeTemperature()
5.  startingSolutionValue ← CalculateObjectiveFunction(Assignments)
6.  currentSolutionValue  ← startingSolutionValue
7.  bestSolutionValue     ← startingSolutionValue
8.  iterationCount        ← 0
9.  continueIterating     ← TRUE
10. while(continueIterating == TRUE){
11.   mutator           ← SelectRandomMutator()
12.   neighbourSolution ← mutator.CreateNeighbourSolution()
13.   objectiveChange   ← CalculateChangeInObjective(NeighborSolution)
14.   if(objectiveChange <= 0){
15.     acceptNeighbour ← TRUE
16.     if(currentSolutionValue + objectiveChange < bestSolutionValue){
17.       newBestSolution ← TRUE
18.     }
19.   }else{
20.     acceptanceProbability ←  e^(- objectiveChange/currentTemperature)
21.     acceptNeighbour       ← acceptanceProbability > randomBetween(0,1)
22.   }
23.   if(acceptNeighbour == TRUE){
24.     neighbourSolution.MakeTheChange()
25.     currentSolutionValue += objectiveChange
26.     if(newBestSolution == TRUE){
27.       solution.UpdateBestSolution()
28.       bestSolutionValue ← currentSolutionValue
29.       newBestSolution ← FALSE
30.   } }
31.   remainingCalculationTime = calculationStartTime +
         allowedCalculationTime - DateTime.Now
32.   if(RemaingCalculationTime <= 0){
33.     continueIterating ← FALSE
34.     SaveAllResults()
35.   }else{
36.     if(iterationCount % temperatureUpdateCount == 0 AND iterationCount > 0){
37.      averageIterationsPerSecond ← iterationCount/
             (DateTime.Now - calculationStartTime).inSeconds
38.       remainingIterations ← averageIterationsPerSecond -
              remainingCalculationTime
39.       coolingFactor ←
             endingTemperature/currentTemperature)^(1/remainingIterations)
40.     }
41.     currentTemperature ← currentTemperature * coolingFactor
42.     iterationCount += 1
43. } }
```

Figure 17: Pseudocode of the Simulated Annealing approach

After the iteration, the remaining calculation time is checked. If the total time is exceeded, the improvement algorithm stops, and the best assignment is (line 33 and 34) is retrieved. If there is

calculation time left, the cooling factor is updated once every certain number of iterations (which is the `temperatureUpdateCount`). Finally, the temperature is updated by multiplying the temperature with the cooling factor (which is < 1) and if there is time left, the new iteration is started with a lower temperature.

4.1.2 – First Come First Served heuristic

First Come First Served (FCFS) is a simple appointment strategy. Every time a flight comes on block, it is decided which belt is the most favorable to assign the flight to, by looking at the expected impact on the objective value for an assignment to each of the available baggage belts. This is based on the flight's belt preferences and the time estimates of flights that have already arrived and the flight that is being planned. Figure 18 shows the pseudocode to create schedules according to the FCFS heuristic. The following input is needed:

- a set of flights, sorted by ascending on-block time: *AllFlights*
- For each flight that is part of *AllFlights*:
  - the expected start and end times of belt occupation: *flight.start* / *flight.end*
  - a mandatory belt assignment, if applicable: *flight.mandatoryBelt*
  - the preferred belts for the flight, if applicable: *flight.preferredBelts*
  - whether or not the flight is part of an alliance: *flight.isAlliance*
  - belts that are closed during a flight's on-belt time: *flight.closedBelts*
- The weights of the component of the objective function:
  - *allianceOccupiedBeltPenalty*
  - *preferredBeltBonus*
  - *penaltyPerMinuteOverlap*
- List of all belts: *AllBelts*

The algorithm assigns flights in order of arrival on-block. In the process of assigning a certain flight, the algorithm first checks if there is a specific belt to which the flight must be assigned, if this is the case, the algorithm will store this belt (line 10-11). If not, the algorithm calculates the expected impact on the goal function for each allowed baggage belt. This is done by checking whether the belt is a preferred belt (line 18), determining the expected total minutes of overlap between the flight and every flight that is yet assigned to the belt (lines 19-22), determining whether the flight to be assigned would arrive at an empty belt (lines 23-25) and by determining the number of yet assigned flights that will not arrive on an empty belt anymore due to the assignment of the flight to the candidate belt (lines 27-28). The algorithm stores the belt that is expected to have the most favorable impact on the goal function,

or if multiple belts are equally favorable, out of those belts the belt that is expected to be empty the earliest (lines 31-34).

```
1.  assignedFlightList ← new list of flights()
2.  for every(belt part of AllBelts){
3.    belt.expectedFirstEmpty ← DateTime.MinValue
4.  }
5.  for every(flight1 part of AllFlights){
6.  bestBeltScore              ← int.MaxValue
7.  bestBelt                   ← NULL
8.  bestBeltExpectedEmptyAt    ← DateTime.MaxValue
9.  flight1.overlapProbaiblity ← 0
10.  if(flight1.mandatoryBelt ≠ NULL AND
        !(flight1.closedBelts.contains(flight1.mandatoryBelt))){
11.    bestBelt ← flight1.mandatoryBelt
12.  }else{
13.    for every(belt part of AllBelts){
14.      if(!(belt in flight1.closedBelts)){
15.        beltOverlapScore        ← 0
16.        occupiedBeltPenalty     ← 0
17.        flight1BlockedAtNewBelt ← 0
18.        beltIsPreferred ← flight1.preferredBelts.contains(belt)
19.        for every (flight2 part of assignedFlightList){
20.          if(flight2.assignedBelt == belt){
21.            overlapFlight1Flight2 ← CalculateOverlap(flight1,flight2)
22.            beltOverlapMinutes += overlapFlight1Flight2
23.            if(flight1.start >= flight2.start AND flight1.isAlliance AND
                  flight1BlockedAtNewBelt == 0 AND overlapFlight1Flight2 > 0){
24.              flight1BlockedAtNewBelt ← 1
25.              occupiedBeltPenalty += 1
26.            }
27.            if(flight2.start >= flight1.start AND flight2.isAlliance AND
                  !flight2.expectedBlocked AND overlapFlight1Flight2){
28.              occupiedBeltPenalty += 1
29.      } } } }
30.      totalBeltScore ← beltOverlapMinutes * penaltyPerMinuteOverlap -
            beltIsPreferred * preferredBeltBonus +
            newAllianceFlightsBlocked * allianceOccupiedBeltPenalty
31.      if(totalBeltScore < bestBeltScore OR (totalBeltScore == bestBeltScore
            AND belt.expectedFirstEmpty < bestBeltexpectedemptyAt)){
32.        bestBelt                  ← belt
33.        bestBeltExpectedEmptyAt   ← belt.expectedFirstEmpty
34.  } } }
35.  flight1.assignedBelt              ← bestBelt
36.  bestBelt.firstExpectedEmptyMoment ← maximum(bestBelt.expectedFirstEmpty,
        flight1.end)
37.  assignedFlightList.append(flight1)
38.  UpdateFlightsStartsAtOccupiedBelts(flight1, bestBelt, assignedFlightList)
39. }
```

Figure 18: Pseudocode of the First Come First Served approach

After all belts are checked, the flight is assigned to the belt expected to have the best impact on the objective function, or to the mandatory belt if there is one (line 35). After, the belt's expected empty moment is updated to include the newly assigned flight (line 36) and for all of the flights that were already assigned to this belt, the '`expectedBlocked`' variable is updated for flights that are now blocked by the new flight.

The heuristic does not incorporate knowledge of upcoming flights when assigning a belt to a flight, but the algorithm does eliminate arrival time uncertainty. The Simulated Annealing approach must deal with the arrival time uncertainty while incorporating knowledge of upcoming flights. The trade-off to be made is between usage of future data and the quality of the time estimates. If the time estimates were to be perfect, the Simulated Annealing would most likely outperform the FCFS heuristic, as the FCFS heuristic has the disadvantage that it does not keep belts free for upcoming alliance flights, so these alliance flights might be forced to be put on occupied belts later. However, as the quality of the estimates would decrease, there is less to be gained from incorporating future demand and it would become more beneficial to eliminate a part of this uncertainty (namely the arrival time uncertainty). Comparisons between the approaches will follow in Section 4.4.

4.1.3 – Greedy heuristic

A third heuristic that we implement is a greedy heuristic. The goal of its implementation is to find out how good the performance of an approach can be that generates results almost instantaneously, due only being a constructive heuristic. Martello and Toth (1992) proposed multiple types of evaluation functions for a Greedy approaches, but for our problem, it only makes sense to use the costs, as defined in the optimization objective, of assigning a job j to a processing station i: c_{ij}.

The assignment values for each combination of flights and belts and the minimum value can then be evaluated, and the combination that results in the lowest c_{ij} can be chosen, but another common technique is to calculate the 'desirability measure' (Martello and Toth, 1992) for each job, also known as the 'regret factor'. To calculate this measure, for each job the difference in the evaluation function between its most desirable assignment and its second most desirable assignment is calculated. The job for which this difference (so the desirability measure) is the highest, is assigned to its most favorable processing station. Using the regret factor instead of using the lowest cost assignment will generally improve the performance of the heuristic (Martello and Toth, 1992).

We need the same type of input for the Greedy approach as for the FCFS approach, but while for the FCFS approach the arrival times on-block are known, for the Greedy approach only an estimation is available. Therefore, the actual data that goes in the heuristic differs from that of the FCFS approach. The expected arrival time and the expected start and end times of belt occupation can differ, as those times all include the arrival time uncertainty for the Greedy approach.

```
totalScore ← 0
unassignedFlightList ← AllFlights
assignedFlightList ← new list of flights()
assignAllFlightsToTheirMandatoryBelt(AllFlights)
createOverlapTableOfAllFlights(AllFlights)
while(length(unAssignedFlightList) > 0) do{
  bestFlightToAssign ← NULL
  bestBeltToAssignBestFlightTo ← NULL
  highestRegretFactor ← int.MinValue
   for every(flight1 part of unassignedFlightList){
    bestBeltScore ← int.MaxValue
    bestBelt ← NULL
   vectorOfBeltScores ← new vector()
   flightRegretFactor ← 0
   for every(belt part of AllBelts){
     if(!(flight1.closedBelts.contains(belt))){
       beltOverlapScore ← 0
       flight1BlockedAtNewBelt ← 0
       beltIsPreferred ← flight1.preferredBelts.Contains(belt)
       for every(flight2 part of assignedFlightList){
         if(flight2.AssignedBelt == belt){
           overlapFlight1Flight2 ← lookupOverlap(flight1,flight2)
           beltOverlapMinutes += overlapFlight1Flight2
           if(flight1.start >= flight2.start AND flight1.isAlliance AND
              flight1BlockedAtNewBelt == 0 AND overlapFlight1Flight2 > 0){
             flight1BlockedAtNewBelt ← 1
             occupiedBeltPenalty += 1
           }
           if(flight2.start >= flight1.start AND flight2.isAlliance AND
              !flight2.expectedBlocked AND overlapFlight1Flight2){
             occupiedBeltPenalty += 1
     } } } }
     totalBeltScore ← beltOverlapMinutes * PenaltyPerMinuteOverlap -
        beltIsPreferred * preferredBeltBonus +
        newAllianceFlightsBlocked * allianceOccupiedBeltPenalty
     if(totalBeltScore < bestBeltScore){
       bestBeltScore ← totalBeltScore
       bestBelt ← belt
     }
     vectorOfBeltScores.append(totalBeltScore)
   }
   vectorOfBeltScores.sort(descending)
   flightRegretFactor ← vectorOfBeltScores[1] - vectorOfBeltScores[2]
   if(flightRegretFactor > highestRegretFactor){
     bestFlightToAssign ← flight1
     bestBeltToAssignBestFlightTo ← bestBelt
     highestRegretFactor ← flightRegretFactor
     assignmentScore ← vectorOfBeltScores[1]
 } }
 bestFlight.assignment ← bestBelt
 assignedFlightList.append(bestFlight)
 UpdateFlightsStartsAtOccupiedBelts(bestFlight, bestBelt, assignedFlightList)
 unassignedFlightList.remove(bestFlight)
 totalScore += assignmentScore }
```

Figure 19: Pseudocode of the Greedy approach

The pseudocode to create assignments for a problem instance is given in Figure 19. First, all flights with a mandatory belt assignment are assigned to their mandatory belt (line 4), these flights will also be removed from the `unAssignedFlightList` and added to the `assignedFlightList`. Then, for every flight combination it is estimated how much overlap there is (line 5), as this value is needed multiple times, the value is stored and retrieved (later, in line 22), instead of recalculating it every time. Next, while there are unassigned flights, the expected impact of all flights to each of the belts is determined, in a similar as with the FCFS approach (line 15-35).

For each flight, all belt scores are saved in the '`vectorOfBeltScores`' vector (line 36). After the impact on the goal function by assigning the flight to each of the belts is assessed, the best and the second-best belt scores are determined by sorting the vector ascending (line 38) and taking the first and second entry. The difference between these is the regret factor (line 39). If the regret factor is higher than the highest regret factor found so far, the regret factor, the flight, and the belt that the flight was assigned to are saved. After all flights were assessed, the flight with the highest regret factor is picked, and the process is repeated until all flights are assigned.

4.1.4 – FCFS incorporating future demand

The final new solution approach that we implement mitigates arrival uncertainty on one hand, while on the other hand also looks ahead. To eliminate the arrival time uncertainty partially, the decision is made upon the flight's arrival on-block, just like in the FCFS algorithm. The resulting algorithm is shown in Figure 20.

If the flight does not have a mandatory belt assignment, the flight's expected impact on the goal function is determined for each of its allowed belts (line 14 and 15). A selection of upcoming flights is made (line 16) to create a subproblem. In the subproblem, the belt that is being assessed is treated as the mandatory belt for the flight that is being assigned (line 19). The subproblem is solved using the greedy approach (line 19), and the flight is then assigned to the belt that results in the best total score for this Greedy approach. We will return to the forming of these sub-problems shortly.

For this method three different overlap estimates are needed per flight combination. First, when calculating the expected overlap between the flight to be planned and a flight that has yet been assigned, the arrival time uncertainty should be excluded from both flights, as for both flights the actual arrival times are known at the moment of assignment. Next, when determining the expected overlap between the flight to be planned and an upcoming flight, arrival time uncertainty should be excluded from the flight to be planned, but not from the upcoming flight. Finally, when determining the expected overlap between two upcoming flights, the arrival time uncertainty should be included for both flights.

```
1.  assignedFlightList ← new list of flights()
2.  unassignedFlightList ← AllFlights
3.  for every(belt part of AllBelts){
4.    belt.expectedFirstEmpty ← DateTime.MinValue
5.  }
6.  for every(Flight part of AllFlights){
7.    bestBeltScore                 ← int.MaxValue
8.    bestBelt                      ← NULL
9.    bestBeltExpectedEmptyAt       ← DateTime.MaxValue
10.   flight.overlapProbaiblity ← 0
11.   if(Flight.mandatoryBelt ≠ NULL AND
         !(flight.closedBelts.contains(flight.mandatoryBelt))){
12.     bestBelt ← flight.mandatoryBelt
13.   }else{
14.     for every(belt part of AllBelts){
15.       if(!(flight.closedBelts.contains(belt))){
16.         subProblemOfFlights ← DetermineFlightsThatWillArriveShortly()
17.         assignedFlightListWithCurrentFlight ← assignedFlightList.append(flight)
18.         flight.mandatoryBelt ← belt
19.         greedySolutionValue ← determineGreedyPlanning(subProblemOfFlights,
               flight, belt, assignedFlightListWithCurrentFlight)
20.         totalBeltScore ← beltScoreBasedOnAssignedFlights + greedySolutionValue
21.         flight.mandatoryBelt ← NULL
22.         if(totalBeltScore < bestBeltScore OR (totalBeltScore == bestBeltScore AND
               belt.expectedFirstEmpty < bestBeltexpectedemptyAt)){
23.           bestBelt                  ← belt
24.           bestBeltExpectedEmptyAt   ← belt.expectedFirstEmpty
25.   } } } }
26.   flight.assignedBelt               ← bestBelt
27.   bestBelt.firstExpectedEmptyMoment ← maximum(bestBelt.expectedFirstEmpty,
         flight.end)
28.   assignedFlightList.append(flight)
27.   UpdateFlightsStartsAtOccupiedBelts(flight, bestBelt, assignedFlightList)
28. }
```

Figure 20: Pseudocode of the First Come First Served approach incorporating future demand

When defining the sub-problem (in line 16), the flight that is being planned and at least all future flights that are expected to overlap with the flight should be included. In Section 4.4, we will also assess the performance of sub-problems that includes more flights: additional to the overlapping future flights, we also include future flights that overlap with overlapping flights, to see which sub-problem selection criterium performs better.

4.2 – Calculating the Monte Carlo overlap estimates

For the current (basic) overlap estimation, the overlap between the expected on-belt time windows of the two flights is considered the expected overlap. In Chapter 3.2 it was concluded that we can incorporate the stochasticity of the time estimates to find the expected overlap between two flights using MC simulations. In this section we explain the process of using the MC simulation to find new overlap

expectancies. The MC simulation requires three terms: the arrival time distribution, the transportation time distribution, and the on-belt time duration distribution.

For modelling the arrival time, we use a normal distribution with the expected arrival time as a mean. The normal distribution has the advantage that random variables from the distribution can take on negative values (which corresponds to a plane landing early). Additionally, in Figure 8 (in Section 2.1.5) it was shown that the normal distribution seemed like a reasonable fit for the examined data. Therefore, the arrival time should behave according to:

$$actual\ arrival\ time \quad = X \sim N(arrival\_time\_estimation, \sigma^2_{arrival\_time})$$

For the transportation time duration and the on-belt duration, a gamma distribution is assumed. Even though Section 2.1.5 stated that the on-belt time estimates did differ significantly from the Gamma distributions, the Gamma distribution seemed to be a better fit than the Normal distribution and the Gamma distribution is generally used to model time durations. Besides, the Normal distribution would need to be truncated to prevent negative time durations.

As on-belt duration times are determined by adding five minutes to the time it takes to put all bags of a flight on the belt, the total time it takes putting the bags on the belt is assumed to be Gamma distributed, rather than the assumed on-belt duration time being gamma distributed. This results into the following relations:

$$actual\ transportation\ time \quad = X \sim \Gamma\left(k = \left(\frac{transport\_estimate}{\sigma_{transport}}\right)^2, \theta = \frac{\sigma^2_{transport}}{transport\_estimate}\right)$$

$$actual\ on\ belt\ time \quad = 5 + X \sim \Gamma\left(k = \left(\frac{on\_belt\_estimate - 5}{\sigma_{on\_belt}}\right)^2, \theta = \frac{\sigma^2_{on\_belt}}{on\_belt\_estimate - 5}\right)$$

There is a probability that a flight has an on-belt duration of five minutes, in the case it takes less than half a minute to put all the bags on the baggage belt. Therefore, it should also be possible to have an estimation for the on-belt time duration of five minutes. Using this estimation would result in an infeasible Gamma distribution ($k=0$ and θ=infeasible). Normally, a duration of 6 minutes would be assigned to any duration between 5.5 minutes and 6.5 minutes, but an estimation of 5 minutes only spans the range of 5.0 to 5.5 minutes, as putting the bags on the baggage belt cannot have a negative duration. The mean duration of the 5-minute estimation is therefore assumed to be 5.25 minutes.

Table 12: Example characteristics of two flights

	Arrival		Transport time		On-belt duration	
	Expected	σ in estimation	Expected	σ in estimation	Expected	σ in estimation
Flight A	10:00	2	10	5	10	3
Flight B	10:05	2	8	3	8	2

To calculate the expected overlap using the MC simulation method for a flight pair with the characteristics shown in Table 12, the following steps are taken, corresponding to the steps in Figure 21:

1. The expected arrival time, the expected transportation time, and the expected on-belt duration are determined for both Flight A and Flight B, based on some estimator (for example “baggage class A will be on-belt for 10 minutes on average”).
2. The standard deviations for each of the estimates should be assumed based on a data analysis of the estimators, like in Section 2.1.5 (for example, “the on-belt time for all flights with baggage class A historically have a standard deviation of 3 minutes”). Using the expected times and the assumed standard deviations, Distributions are created for the arrival time, the transportation time, and the on-belt duration of both flights. For each of these distributions, a random value is sampled, to find a random arrival time, transportation time, and on-belt duration of both flights.
3. From these simulated times and durations, the resulting time windows for which the flights would be on the baggage belt are calculated.
4. The overlap between the two on-belt time windows is determined, as well as whether flight A blocks flight B (in case flight A is already on block when flight B arrives, therefore resulting in B arriving at an occupied belt), flight B blocks flight A, or they arrive within the same minute (we consider them to block one another).
5. Steps 2-4 are repeated a large amount of times (1,000+ times) and the resulting overlap and blocking probabilities are stored each time. In Figure 21, a histogram of the resulting overlaps is shown.
6. The average of the 1,000+ newly generated overlaps is taken. This is considered the expected overlap in MC overlap calculation. Also, the percentage of simulations in which flight A blocks flight B, and vice versa is calculated.

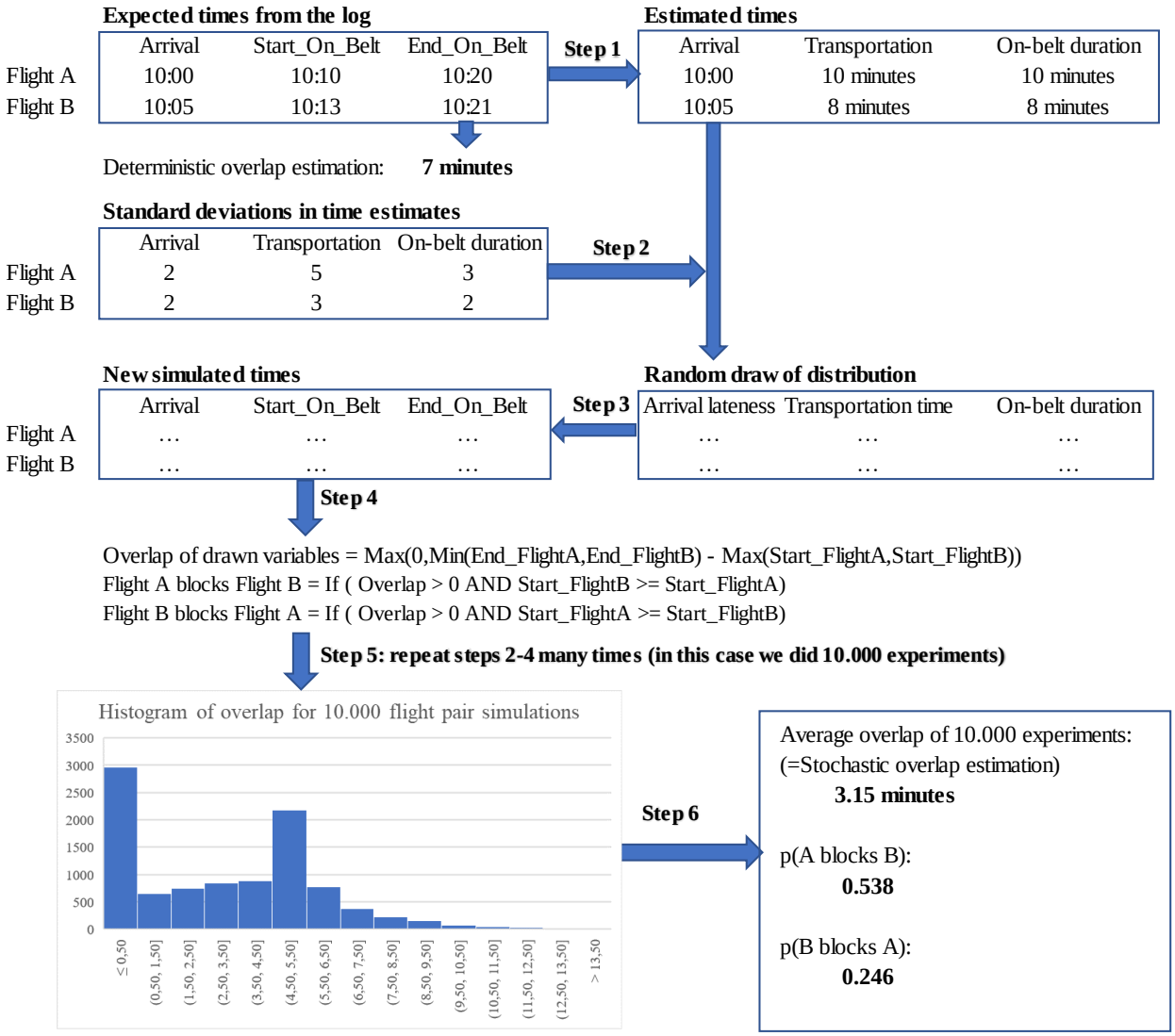


Figure 21: Calculating the expected overlap between two flights via MC simulations

A cut-off point to prevent the problems to become too complicated

Due to the stochastic nature of the problem, any given set of two flights has some (although maybe very small) probability of overlapping. Besides, there is no negative overlap. If there is a large gap of for instance one hour between the on-belt time windows, we consider the overlap to be 0, and not minus one hour. So, on average, not only the probability of overlap between two flights is always larger than 0, but also the expected overlap. Because this would make the number of interactions in the model very large, run-times of algorithms would increase, or in case of an algorithm that has a maximum runtime, the explored solution space would decrease. To counteract this effect, a cut-off point of 0.5 minutes is used for the MC simulated overlap. If the expected overlap is less than 0.5 minutes, we exclude the expected overlap and the blocking probabilities from the model. The cut-off point of 0.5 minutes is chosen because everything below that threshold would round to 0 minutes.

The number of MC replications for an estimation of the overlap between two flights

To determine how many overlap simulations should be done per flight pair to have a stable estimation, a hundred flight pair samples are taken. All these flight pairs have a gap between their on-belt time windows of at most 20 minutes (so Flight A of the flight pair will arrive on the baggage reclaim belt at most 20 minutes after Flight B's baggage belt occupation has ended and vice versa, in the log files). For each of these flight pairs, 50,000 overlap durations are simulated for σ_arrival = 3, σ_transport = 6, and σ_on-belt = 6. For each of these simulations we have calculated the standard error relative to the mean of the overlap. The results are shown in Appendix F, and for n = 20,000, the results are shown in Figure 23.

The number of replications is increased until the error relative to the expected value (the relative error) will stay below the threshold of 0.05 for the flight pairs. When the expected overlap is close to zero, the relative error becomes larger fast, as can be seen in Figure 22. Therefore, we base the number of replications only on the flights that are considered overlapping (with an expected overlap at least 0.5 minutes). In Figure 22, this means that the upper right quadrant, with quadrants defined by the overlap threshold and the relative error threshold, should be empty. As this is not the case, more replications are needed.

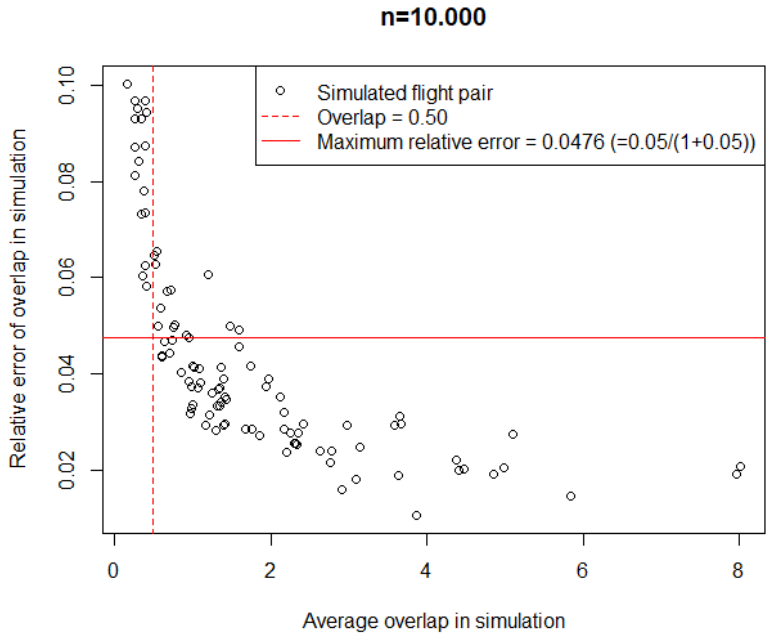


Figure 22: Relative error versus the average estimated overlap for 10.000 MC replications (σ_arrival = 3, σ_transport = 6, σ_on-belt = 6, safety factor = 0)

For n = 20,000, all randomly chosen experiments with an expected overlap of more than 0.5 stay below a maximum relative error of 0.0476. Therefore, we choose to do experimentation with 20.000 overlap simulations per estimation of overlap for a flight pair. If instead of using an expectancy of 0.5 minutes of overlap as cut-off point, we use the upperbound of the confidence interval of the overlap, this property still holds, as can be seen in Appendix F.

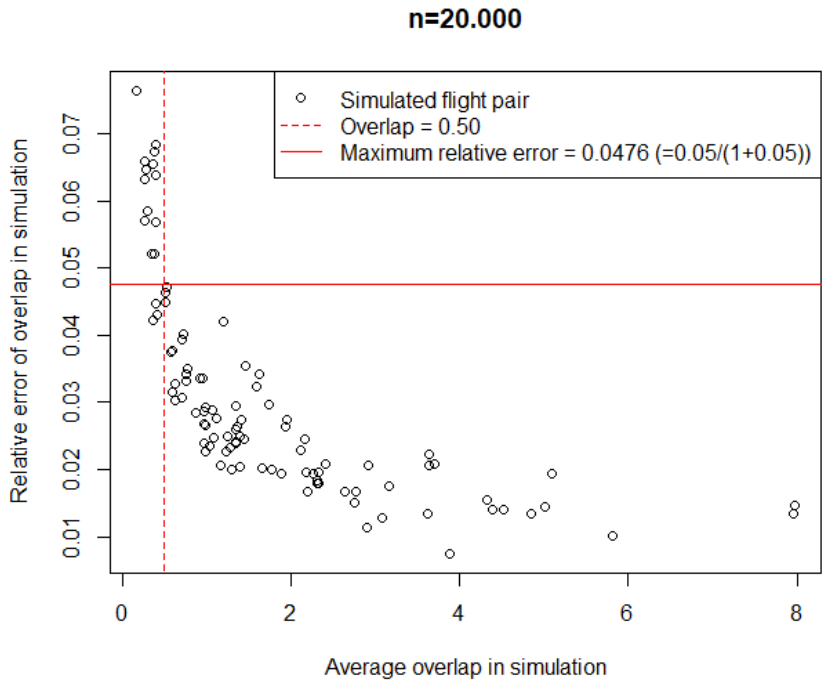


Figure 23: Relative error versus the average estimated overlap for 20.000 MC replications (σ_arrival = 3, σ_transport = 6, σ_on-belt = 6, safety factor = 0)

4.3 – Incorporating ways to deal with stochasticity

In this section we explain how we incorporate the methods that deal with stochasticity. In Section 4.3.1 we explain how the estimates from Section 4.2 will be implemented in the solution approaches, except for the MIP formulation, and in Section 4.3.2, we explain how the estimates are implemented in the MIP formulation. In Section 4.3.3, we explain the implementation of the safety times for the basic overlap estimates and the MC overlap estimates.

4.3.1 – Incorporate the Monte Carlo overlap estimates

In all optimization methods introduced in Chapter 4.1, we use the expected impact on the goal function of (re-)assigning a flight to a certain belt to make decisions. In this section we discuss how using the MC estimates will change the way the evaluate objective function and expected change in the objective function are determined. It must be noted that all MC overlap estimates are made prior to the optimization.

To determine the change in objective function evaluation, we split the objective function into its three components and look at the impact of using the MC overlap estimates per component. The three components of the objective function are the expected overlap durations, the number of alliance flights that start at an occupied belt and the number of flights that are assigned to their preferred belt.

Whether or not a flight starts at a preferred belt does not depend on time estimates, so for this component nothing will change when using the MC overlap estimates. The values of the expected overlap between two flights will differ between the stochastic and the deterministic approach, but once we have determined the expected overlap for every flight combination (which happens before the optimization), the way we calculate the impact of overlap on the objective function also does not change: if two flights have an expected overlap of more than 0.5 minutes and they are assigned to the same belt, this expected overlap is added to the objective function.

Incorporating the change in the goal function component that penalizes the number of alliance flights that start at an occupied belt, will require more change. To illustrate this, let us consider a situation with three flights, with deterministic time estimates as shown in Figure 24.

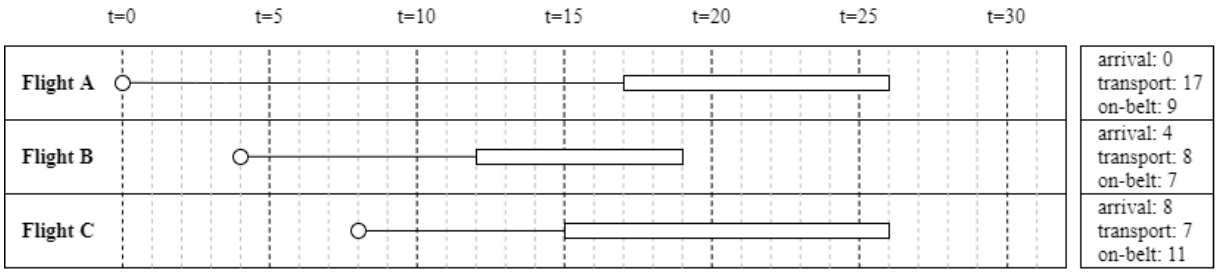


Figure 24: Three example flights including time estimates

In case we use the current overlap estimation method, we find the values for the expected overlap and whether a certain flight is on the belt upon arrival of another flight (also referred to as blocking) in Table 13 and Table 14 respectively. If we use the method explained in Chapter 4.2, assuming all estimated times have a standard deviation of 2 minutes, we find the expected overlap durations and blockage probabilities shown in Table 15 Table 16 respectively.

If we would consider any flight with a small probability of blocking another flight to be actually blocking the flight, then all flights relatively close to each other would be considered to block one another, as can be seen in the example in Table 16. Therefore, we work with the blockage probabilities, and calculate the expected impact on the objective function of assigning a flight to a certain belt by calculating the expected total increase in blockage probability of alliance flights on that belt.

Table 13: Deterministic estimates of overlap duration

	A	**B**	**C**
A	-	2	9
B	2	-	4
C	9	4	-

Table 14: flights considering blocking other flights when using deterministic estimates

blocks	**A**	**B**	**C**
A	-	1	1
B	0	-	0
C	0	1	-

Table 15: MC estimates of overlap duration

	A	**B**	**C**
A	-	2.53	6.30
B	2.53	-	3.69
C	6.30	3.69	-

Table 16: MC estimates of probability of flights blocking other flights

blocks	**A**	**B**	**C**
A	-	0.58	0.72
B	0.13	-	0.26
C	0.34	0.63	-

The probability of flight *i* is being blocked by any flight assigned to the same belt, is equal to 1 minus the probability that flight *i* is not blocked by any of the flights assigned to the same belt, assuming the probabilities being independent. Returning to our example flights from Figure 24 and assuming all flights are alliance flights, this would mean that when flight A and flight B are assigned to the same belt (and no other flights), the impact of adding flight C to that same belt can be calculated in the following manner:

- From Table 16, we see that the probability of flight A blocking B is 0.13, and the probability of flight B blocking flight A is 0.58. As Flight C is not planned yet, the current probability of this flight being blocked is 0.
- Adding flight C, which has a 0.72 probability of blocking Flight A, increases the probability of Flight A being blocked to: 1 - (1 - 0.58) * (1 - 0.72) = 0.8824.
- The probability of flight A being blocked has increased from 0.58 to 0.88, so an increase of 0.3. In the same way we can find that the increase for Flight B is 0.23, and 0.76 for Flight C.
- The total increase in probabilities of flights being blocked is 0.3 + 0.23 + 0.76 = 1.29. Assuming the penalty per blocked flight to be 9, we find an expected impact of $9 * 1.29 = 11.61$ for the additional blockage of alliance flights.
- The expected overlap between flight A and flight C is 6.3 and the expected overlap between flight B and flight C is 3.69, making the total expected additional overlap 9.99.
- Therefore, the expected impact of adding flight C to the same belt as flight A and flight B is 21.6.

To incorporate this method, the optimization methods using the MC estimates require a change in input. Previously, part of the required input were for each flight 1) a list of flights that are expected to be on

belt at the start of the flight's on-belt window and 2) a list of alliance flights that are expected to arrive during the flight's on-belt window. In the optimization based on the MC estimates we need lookup tables that for any flight show the probability of every other flight blocking the flight, like Table 16. Appendix E shows the pseudocode of determining the expected change in objective function for both deterministic and MC overlap estimation approaches for the SA algorithm that was described in Figure 17.

4.3.2 – Monte Carlo overlap estimates and the Mixed Integer Problem

As noted in the previous section, the probability of flight *i* is being blocked by any flight assigned to the same belt, is equal to 1 minus the probability that flight *i* is not blocked by any of the flights assigned to the same belt, assuming the probabilities being independent. Therefore, we find the probability of flight *j* being blocked by any flight: 1 - $\prod_{i \in F}(1 - b_{ij} * O_{ij})$), in which the variable O_{ij} is 1 if flight *i* and flight *j* are assigned to the same belt. b_{ij} is the probability that flight *i* is on the belt at the arrival of flight *j*, and thus blocking the empty belt arrival of flight *j*. The variable b_{ij} is calculated using the MC simulations.

Including the process described above into the Mixed Integer Program, the goal function for optimization with the MC estimates would transform into the following:

$$\min \sum_{i \in F} \sum_{j \in F} O_{ij} t_{ij} \;+\; \beta \sum_{j \in F} s_j * (1 - \prod_{i \in F}(1 - b_{ij} * O_{ij})) - \; \gamma \sum_{i \in F} Q_i,$$

In which t_{ij} is the expected overlap between flight *i* and flight *j*, following from the average overlap of the MC simulations as described in Section 4.3. The other symbols will retain their definition from Section 2.2.2. The quadratic formulation will not be solvable by the MIP solver. To still be able use the power of the solver, we change its objective function, so that instead of multiplying the probabilities, the probabilities are added:

$$\min \sum_{i \in F} \sum_{j \in F} O_{ij} t_{ij} \;+\; \beta \sum_{j \in F} s_j * \sum_{i \in F} b_{ij} * O_{ij} - \; \gamma \sum_{i \in F} Q_i$$

This can be further simplified by calculating the penalty of two flights being assigned to the same belt, in which the expected overlap between the two flights is equally divided over the two flights. The new objective function will then become:

$$\min \sum_{i \in F} \sum_{j \in F} O_{ij} v_{ij} - \; \gamma \sum_{i \in F} Q_i, \text{ with } v_{ij} = \frac{1}{2} t_{ij} + \beta * s_j * b_{ij}.$$

This formulation will result in a different model than the formulation that the SA approach will solve. In Section 4.4 and in the experimentations, we will see how big the impact of this simplifying the goal function will be.

4.3.3 – Safety times

In Section 4.2 we have investigated how to find the expected overlap between two flights using MC overlaps. As mentioned in Chapter 2, the overestimation and underestimation of flight pairs overlapping must be balanced, both for basic overlap estimation and MC overlap estimation. Therefore, using the expected values for the overlap does not necessarily give the best results, however.

Instead of classifying flights as overlapping when their estimated time windows overlap (or according to the percentage of the stochastic cases that overlap in case of using the MC overlap estimation method), we might consider flights overlapping if they almost overlap, in order to mitigate the effects of stochasticity, like discussed in the literature review in Section 3.3. To incorporate this overestimation, the expected length of each flight's on-belt duration can simple be increased by a fixed duration. This increase is what we call the safety factor. For MC overlap estimation, this safety factor is also implemented in the experimentation, by adding the safety factor to each random drawn on-belt time in every MC simulation.

By increasing the safety factor, the number of flight combinations rightfully labeled as overlapping will increase, but on the other hand the number flight combinations that are wrongfully labeled as overlapping will increase. To find the best safety factors for the different solution approaches at different levels of stochasticity, we incorporate these factors into the experiments.

If optimization with the basic overlap estimates using a safety factor would give assignments with equal performance to that of optimizations with the MC overlap estimates, then this deterministic approach using the safety factor would be strongly preferred, as it would be much faster, and the resulting schedules would be easier to interpret.

4.4 – Baseline performance

In this section we will see how the introduced algorithms perform on historical data. For this, we repeat the procedure of Section 2.2.3 and Section 2.2.4: We take 1 week of data, containing more than 1,000 flights (the same week as in Chapter 2), to create seven problem instances of one day each. For each of the problem instances, assignments are be created for three different types of times: First, in Section 4.4.1, the realized times of the baggage handling process are used to generate assignments. Next, in Section 4.4.2, the estimates that are being used by the airport (described in Chapter 2.1) are used to create assignments. In Section 4.4.3, we use our own time estimates to generate assignments, which are average times of the estimators, so that we have unbiased estimates. Finally, in Section 4.4.4, we discuss the limitations of the used approach.

4.4.1 – Optimizing based on the realized times

If the time estimates would be perfect, they would be equal to the realized times and there would be no stochasticity. Without stochasticity, the MC simulations averages would be equal to the overlap retrieved by the basic way of calculating overlap. Therefore, we do not incorporate the MC simulation overlap estimation in this section.

The results of optimization based on the perfect estimates are shown in Table 17. Each column, corresponding to a problem instance, is color coded with green corresponding to relatively good values and red being relatively bad values. The best solution value of each problem instance is displayed in bold.

Both the SA results and the MIP solver results are the same as those in Chapter 2. The "FCFS + (small)" approach refers to the FCFS approach incorporating future demand, in which subproblems are formed based on only the flight that is being planned and the future flights expected to overlap with this flight. "FCFS + (large)" considers all the flights from "FCFS + (small)", and additionally all future flights that are expected to overlap with any of the flights in the "FCFS + (small)" set.

Table 17: Performance of algorithms when using flights' realized times

Approach	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Week total
MIP UB	**-1,628**	**-1,350**	**-1,513**	**-1,859**	**-1,698**	**-1,751**	**-1,599**	**-11,398**
SA	**-1,628**	**-1,350**	**-1,513**	-1,858	-1,696	**-1,751**	**-1,599**	-11,395
FCFS	-1,426	-1,148	-1,361	-1,664	-1,567	-1,643	-1,441	-10,250
Greedy	-1,573	-1,329	-1,469	-1,821	-1,647	-1,714	-1,564	-11,117
FCFS + (small)	-1,533	-1,188	-1,366	-1,750	-1,581	-1,693	-1,488	-10,599
FCFS + (large)	-1,568	-1,262	-1,388	-1,776	-1,614	-1,669	-1,458	-10,735

The MIP solver gives the best results, followed by the SA approach, the Greedy approach, the FCFS approach incorporating future demand with the large subproblem selection, the FCFS approach incorporating future demand with the small subproblem selection, and the FCFS approach not incorporating future demand. The performance for the newly implemented heuristics relative to that of the MIP solver and one another is in line with expectations, as the solution approaches that make assignments based upon more future information generate better results, and all used information is accurate.

4.4.2 – Optimizing based on current estimates

Now, we compare performance of the algorithms based on the current estimates. As the time estimates used for the optimization are very biased (see Chapter 2.1), it makes little sense to incorporate the stochasticity using MC simulations. Therefore, again, we only use the basic overlap estimation methods

for the algorithms. The realized performance of the assignments created with the current estimates are shown in Table 18. We have also included the realized results from the assignments that the airport has done for the data, using the SA approach with rolling horizons.

The FCFS approach incorporating little future demand scored the best overall, followed by the simple FCFS approach and the FCFS approach incorporating more future demand, all clearly outperforming the MIP, SA, and the greedy approach. Under current estimates, using less information of higher quality clearly results in better assignments.

The difference between the SA in one problem instance and the performance of SA using rolling horizon is not that large for the week of data. However, on a daily basis the differences can be quite large. This is presumably because assignments of which the expected values are pretty close, the realized values may differ a lot. However, this difference averages out if we use multiple days of data.

Table 18: Realized performance comparison using current estimates

Approach	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Week total
SA (rolling horizon)	-990	-769	-952	**-1,171**	-811	-1,122	-915	-6,730
MIP	-923	736	-873	-1,044	-991	-1,156	1,037	-6,760
SA	-990	-772.8	-870	-1,014	-924	-1,149	-1,106	-6,827
FCFS	**-1,081**	-835	-986	-1,101	-1,068	-1,219	-1,097	-7,387
Greedy	-963	-781	-809	-992	-826	-1,108	-931	-6,410
FCFS + (small)	-1,028	-866	-890	-1,168	**-1,112**	-1,236	**-1,154**	**-7,454**
FCFS + (large)	-974	**-875**	**-995**	-1,096	-1,040	**-1,249**	-1,091	-7,320

4.4.3 – Optimizing based on unbiased estimates

In this section we use new estimates for the current data, which are based on the group averages of the data. The predictors currently used for time estimation are the airport's apron and the baggage class of the aircraft that executes the flight. In Table 5, there is no clear connection between the apron and the realized average transportation time, nor between the apron and the standard deviation of the estimation. For the baggage class a clearer distinction can be found between the averages and the deviations. Using only one estimator also prevents creating small groups that would give overly accurate estimates of the processing times, therefore only the baggage class of the aircraft is used as predictor. The estimates that are used, are denoted in Table 19. For the approaches that use the basic overlap estimation (in which the expected overlap is defined by the overlap of the expected on-belt time windows), all times are rounded to full minutes. In Appendix G, we compare the unbiased estimates of the used week with that of another week of historical data, and find that the estimates are similar.

Table 19: Unbiased time estimates

Flight's baggage class	Transport time		On-Belt duration		Arrival time
	Mean	St. Dev.	Mean	St. Dev.	St. Dev.
A	14.6	4.0	7.3	4.0	2.4
B	18.3	5.9	8.8	3.2	2.4
C	16.9	6.4	13.0	9.1	2.4

The results of the experiment with unbiased estimates are shown in Table 20 and an overview of the expected results and of each SA replication are shown in Appendix H. There we see that the expected results of the SA are close to that of the optimization based on the realized times in Section 4.4.1. The MIP solver using the basic overlap estimates has a slightly higher expected performance than the SA approach, but looking at the realized performance, the SA performs better. Even though both algorithms still do not result in good assignments.

For the solution approaches using basic overlap estimates, all estimation methods show improvement when comparing them with solutions generated based on the airport's basic estimates in Section 4.4.2. Especially the MIP solver, the Simulated annealing approach and the FCFS approaches that incorporate future flights benefit from the unbiased estimates.

Including the stochasticity in the overlap by using the MC overlap estimates generally improves the total week scores for all the assignment methods, except for the FCFS approach incorporating future demand, which gives similar results. All methods using the MC overlap estimates perform about equally well. The MIP solver with the MC overlap estimates performs comparable to the SA approach, while they use different goal functions in their optimization, as explained in Section 4.3.

Table 20: Optimization approach comparison for unbiased estimators without the use of safety times

	Approach	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Week
Basic Overlap estimates	MIP Solver	-977	-688	-897	-1,213	-1,026	-1,244	-1,033	-7,078
	SA	-1,003	-782	-1,005	-1,163	-991	-1,230	-1,107	-7,280
	FCFS	-1,049	-825	-1,079	-1,191	-1,007	-1,348	-1,093	-7,592
	Greedy	-955	-613	-879	-1,073	-893	-1,197	-916	-6,526
	FCFS + (small)	-1,154	-906	-1,078	-1,224	-1,086	**-1,411**	-1,150	-8,009
	FCFS + (large)	**-1,220**	-909	-1,042	-1,249	-1,107	-1,355	-1,164	-8,046
MC overlap estimates	MIP Solver	-1,184	-935	**-1,218**	-1,233	-1,088	-1,266	-1,106	-8,030
	SA	-1,128	-908	-1,187	**-1,298**	-1,092	-1,304	-1,143	**-8,060**
	FCFS	-1,102	-799	-1,129	-1,272	-1,118	-1,377	**-1,175**	-7,972
	Greedy	-1,160	-876	-1,178	-1,220	-1,022	-1,293	-1,156	-7,905
	FCFS + (small)	-1,186	-965	-1,143	-1,294	-1,057	-1,283	-1,129	-8,057
	FCFS + (large)	-1,096	**-974**	-1,151	-1,145	**-1,138**	-1,277	-1,145	-7,926

Next, various levels of safety times are added to the expected on-belt time windows of the flights, as explained in Section 4.3.3, and the optimization methods are tested again. The safety time duration was increased by steps of one minute until the results of al solution approaches started to worsen again. The results for one week of data are shown in Table 21.

The MIP solver does not benefit from the safety times and generates similar results for the basic overlap estimation method, for all safety levels, while the performance of the MIP solver using MC overlap estimates decreases. Due to more flights being considered overlapping, the problem becomes too complicated for the MIP solver to generate good upper bounds in the limited time. The FCFS approaches also do not seem to benefit much from using the safety times.

For the SA and Greedy approach, we see clear improvements due to the added safety factors, for both the basic overlap estimation methods and the MC overlap estimation methods. The basic overlap methods need a higher safety factor to reach their best results than the MC overlap method.

Table 21: Optimization approach comparison for unbiased estimators with the use of safety times (average weekly realized results)

		Safety factor (in minutes)							
		0	1	2	3	4	5	6	7
Basic overlap estimates	*MIP*	-7,078	-7,031	-7,030	-7,115	-7,090	-7,105	-7,109	-7,043
	SA	-7,280	-7,343	-7,493	-7,651	-7,770	-7,745	-8,055	-7,882
	Greedy	-6,526	-6,795	-7,125	-7,447	-7,303	-7,386	-7,335	-7,230
	FCFS	-7,592	-7,578	-7,534	-7,452	-7,530	-7,392	-7,457	-7,203
	FCFS+ (small)	-8,009	-7,880	-7,946	-7,768	-8,032	-7,751	-7,845	-7,684
	FCFS+ (large)	-8,046	-7,945	-8,000	-7,894	-7,702	-7,620	-7,525	-7,170
MC overlap estimates	*MIP*	-8,030	-7,949	-7,857	-7,696	-7,684	-7,335	-7,571	-7,343
	SA	-8,060	-8,131	-8,300	-8,236	-8,223	-8,299	-8,170	-8,232
	Greedy	-7,905	-8,029	-8,235	-7,993	-8,188	-7,905	-7,721	-7,791
	FCFS	-7,972	-8,057	-7,924	-7,877	-7,857	-7,621	-7,572	-7,649
	FCFS+ (small)	-8,057	-8,037	-7,816	-7,719	-7,768	-7,802	-7,663	-7,620
	FCFS+ (large)	-7,926	-7,982	-7,869	-7,594	-7,638	-7,767	-7,629	-7,610

In the analyzed week of data, the SA approach using MC overlap estimates and a safety factor of two minutes resulted in the best assignments. The results are close to that of the same approach using other safety factors, and the results do not point to a clear best safety factor (for the SA using MC overlaps, safety factor two and five are better than three and four). However, it is clear that the MC overlap estimation methods and the safety factors clearly allow for better performance.

In Table 22 we show the breakdown of the objective function of the historical solution of the data, and the best new solution, from the SA approach with MC overlap estimates and a safety factor of two minutes. The optimized solution has, for one week of flights, almost 800 minutes less overlap in its

realized schedule, and 125 less alliance flights that start at occupied belts. In the historical schedule however, 44 more flights were assigned to their preferred belts.

Table 22: Objective function breakdown of historical performance and best new performance

	Optimized	Historical	Change
Total	-6,730	-8,300	-1,570
Overlap minutes	2,691	1,894	-797
Preferred belt bonus	-12,832	-12,480	352
Alliance starting at empty belts penalty	3,411	2,286	-1,125

Regarding the FCFS approach using future demand, we find that for the basic overlap estimation methods, the larger sub-problem selection results in a slightly better week total, so it seems that the small overlap selection will create problem instances with too little future flights. For the MC overlap estimation, the smaller sub-problem selection gives better results. As with MC overlap estimates, each flight has, on average, expected overlap with a larger number other flights than the basic overlap estimates will have. A large sub-problem selection for the MC overlap estimates may be too large, because we would consider too many future flights. Although there is not much difference on a daily nor a weekly level, we will continue to use the best performing approaches. For the FCFS with future demand approach, using basic overlap estimates, that is the large subproblem selection. For the FCFS with future demand based on MC overlaps, that is the small-subproblem selection. The chosen approaches create similar-sized sub-problems, as can be seen in Figure 25.

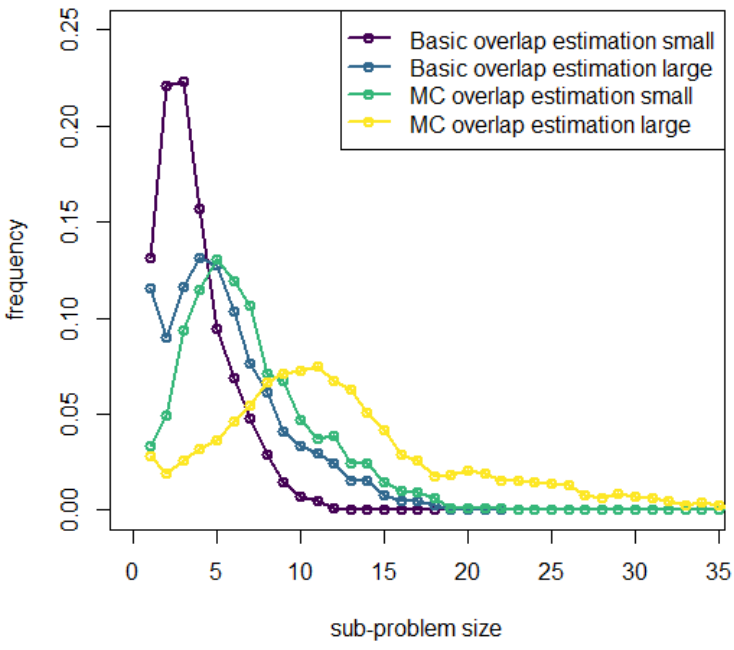


Figure 25: Size of sub-problems in FCFS approach used to assign a single flight, for large and small sub-problem selections when not using safety times

4.4.4 – Limitations of the performance measurement

In the performance comparison of the previous sections and in the upcoming chapters, we optimize the schedule based on sub-problems. The sub-problems are solved just once, and not according to the rolling horizon method in which every five minutes the information is updated. Due to this, we cannot incorporate the progression of arrival time stochasticity in the solution approaches.

The main reason for not using the rolling horizon approach is that we would need a very long time to simulate one day of data. In the current implementation for every 24 hours of data, the SA algorithm is running 12 hours (5 minutes per 10-minute optimization round). As the rolling horizon method would need to be applied to the MIP, SA and Greedy approaches for both the basic overlap and MC overlap estimates, just for the results of Table 21, we need to simulate 294 days of data, taking 147 days.

In Section 4.4.2 we saw that the performance of one problem instance and that of the rolling horizon approach is pretty similar for one week of data, with the performance of one problem instance being slightly better. However, on a daily basis, the differences can be quite large.

4.5 – Conclusion

In this chapter we have explained the heuristic solution approaches: the SA algorithm, a FCFS approach, a Greedy heuristic, and a FCFS approach that incorporates future demand. Next, a way to determine the expected overlap between two flights, using MC simulations was proposed. The implementation of these MC overlap estimates and the use of safety factors have been explained for all solution approaches.

All solution approaches were then applied to historical data of the airport. First, based on the situation in which perfect estimates are available. Second, based on the time estimates that are currently being used by the airport. Finally, the solution approaches have been applied using an unbiased version of the estimates, for which also the solution approaches with MC overlap estimates were applied, using historical variability to determine the expected overlap.

In the first set of experiments, with the perfect estimates, a clear trend was visible in which the solution approaches that incorporate more future knowledge will perform better. The MIP solver and the SA approach therefore clearly gave the best results.

For the second set of experiments, using the airport's current estimates, it was observed that solution approaches that assign flights to the baggage belt upon arrival of the plane (thus, removing the arrival time uncertainty), clearly outperformed the other approaches.

In the third set of experiments, we derived unbiased estimates from the data analysis in Chapter 2, and used these times in our optimization. For these estimates, only the flight class of the aircraft is used as an estimator. The SA approach benefitted from both the MC overlap estimates, added safety time, and the combination between both. With this combination of methods to mitigate the impact of stochasticity, the SA approach was clearly superior to the other solution approaches.

While the MIP solver returned optimal results for some of the problem instances in Section 4.4.2, the solver proved to be unfit to work with more congested problems, like the problems with larger safety factors resulting in more flights to be considered overlapping.

For the FCFS approach incorporating future demand, we found that in case the basic overlap estimates are used, the larger sub-problems performs better. In case the MC overlap estimates are used, the small sub-problems perform better. Presumably because the MC approach results in more flights classified as overlapping, and in the large sub-problem selection, too much future flights would be used in the decision making.

Chapter 5: Experimental Design

In the future, the accuracy of estimating time durations is expected to get better. This is due to importation information being unavailable to the estimation process, like the total number of bags on an airplane. In the (near) future however, this is expected to become available. The number of bags on a plane impacts the amount of time it takes to unload an airplane and the time it takes to load the baggage onto a baggage belt and would therefore be a more suitable estimator than the capacity or size of the plane. It is not clear how much better the time estimates can get due to these better predictors, and neither it is clear to which extent better time predictions will contribute to better assignments. In our experimentation we assess this impact of estimation quality onto the performance of different algorithms, to find out what would be the best belt assignment method for given levels of uncertainty that correspond to improved estimates.

In this chapter, we describe the experimental design for finding the best planning strategy for the assignment problem under improved estimators. In Section 5.1, we discuss what factors will be changed throughout the experimentation, why these factors were chosen, and how these factors can be implemented. In Section 5.2, we explain the parameterization of the optimization methods. In Section 5.3, the structure of the experimentation is explained.

5.1 – Experimental factors

In this section, we explore which factors we will change throughout the optimization, how these factors are implemented, and why these factors are chosen. We start off with looking at how we can change the level of uncertainty in the estimates in Section 5.1.1, followed by explaining the use of a safety factor in Section 5.1.2.

5.1.1 – Adding stochasticity to the time estimates

As pointed out in Section 3.3, the performance of each solution approach should be determined for different levels of stochasticity in the time-estimation process. Besides, we are interested in the performance of the algorithms under improved estimates.

In the experimentation, the $\sigma_{arrival\_time}$, $\sigma_{transport}$ and σ_{on_belt} must be adjustable to a wanted level of stochasticity. The standard deviations that are used in the experimentation are based on the current maximum stochasticity in the time estimates and lay within the following bounds:

$0 \leq \sigma_{arrival\_time} \leq 3, 0 \leq \sigma_{transport} \leq 6, 0 \leq \sigma_{on\_belt} \leq 6.$

In Chapter 4.2, it was explained that the actual times and the estimated times are assumed to have the following relationship:

$$actual\ arrival\ time \quad = X \sim N(arrival\_time\_estimation, \sigma^2_{arrival\_time})$$

$$actual\ transportation\ time \quad = X \sim \Gamma\left(k = \left(\frac{transport\_estimate}{\sigma_{transport}}\right)^2, \theta = \frac{\sigma^2_{transport}}{transport\_estimate}\right)$$

$$actual\ on\ belt\ time \quad = 5 + X \sim \Gamma\left(k = \left(\frac{on\_belt\_estimate - 5}{\sigma_{on\_belt}}\right)^2, \theta = \frac{\sigma^2_{on\_belt}}{on\_belt\_estimate - 5}\right)$$

For each flight we use the log files to derive the realized arrival time, the transportation time, the on-belt time duration, and the current estimates for each of these times. However, better estimates are not available from the current data (otherwise the airport would have used those). Therefore, these improved estimates must be created in some way.

For the arrival time, which is considered to be normal distributed, an estimate for which the above-mentioned arrival time relationship holds is simple to create. To explain this, let us introduce the following notations:

σ_{new} is the standard deviation of the estimate that we want to create

α represents the actual time. For instance the actual arrival time or the actual transportation time.

a(x) is the probability density function with α as mean, and standard deviation σ_{new}.

ε is a random draw of the distribution α as mean, standard deviation σ_{new}.

e(x) is the probability density function with ε as mean, and standard deviation σ_{new}.

We know that ε is, by definition, a random variable from the distribution with mean α and standard deviation σ_{new}. However, we do not want the estimate to be a normally distributed value based on the realized times, but we want the realized time to be a normally distributed value based on the estimates.

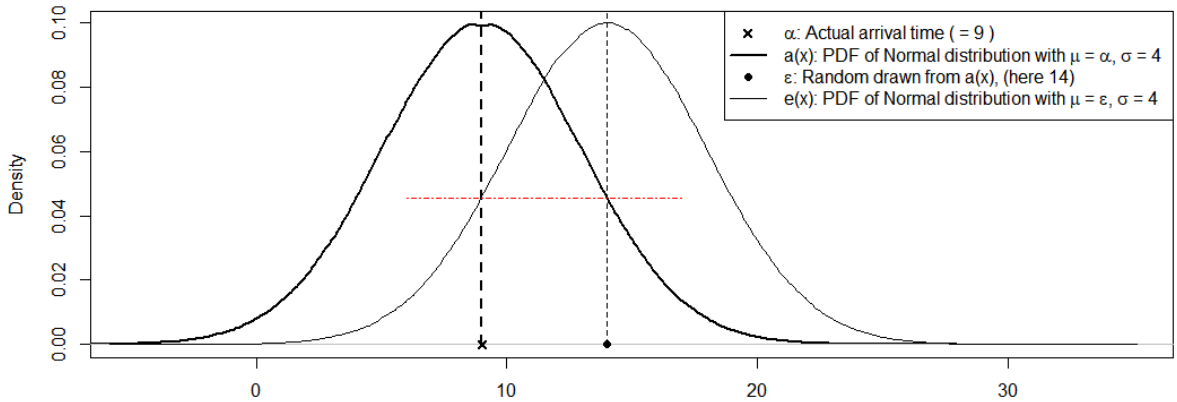


Figure 26: Overview of α, a(x), ε and e(x) for Normal distributions

Because the normal distribution is a symmetric distribution, we know that for any randomly drawn ε, it holds that *a(ε) = e(α)*. In Figure 27, we show an example of this. Therefore, the realized time will also be a normally distributed value from ε. So, if we use a randomly drawn ε as estimate, then α is a normally distributed value based on the estimate.

The realized transportation times and realized on-belt durations are assumed to be Gamma distributed. For the gamma distribution, it does not hold that *a(ε) = e(α)* for every value of ε, as shown in Figure 27. As ε, by definition, is a gamma distributed value with mean α, and we use the randomly drawn ε as estimate, then α cannot be a gamma distributed value based on the estimate.

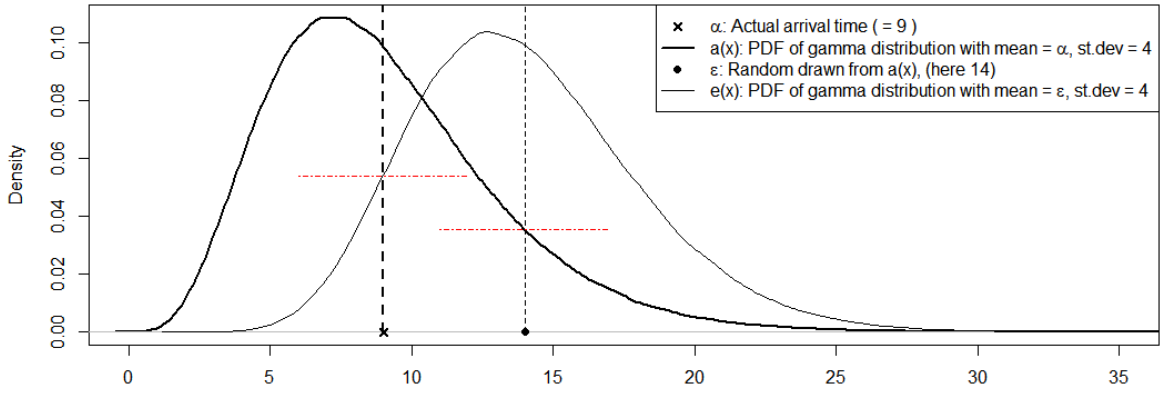


Figure 27 Overview of α, a(x), ε and e(x) for Gamma distributions

In Figure 27, we show the relationship between *a(ε)* and *e(α)*, by plotting the probability of α being drawn from *e(x)* is versus the probability of ε to be drawn from *a(x)* for a realized time of 9 minutes and a standard deviation of 4 minutes.

In the plot, $y = E\left(9; k = \left(\frac{a\left(x; k=\left(\frac{9}{4}\right)^2, \theta=\frac{4^2}{9}\right)}{4}\right)^2, \theta = \frac{4^2}{a\left(x; k=\left(\frac{9}{4}\right)^2, \theta=\frac{4^2}{9}\right)}\right)$, in which $E(9; k, \theta)$ represents the cumulative gamma distribution with shape k and scale θ, evaluated at 9.

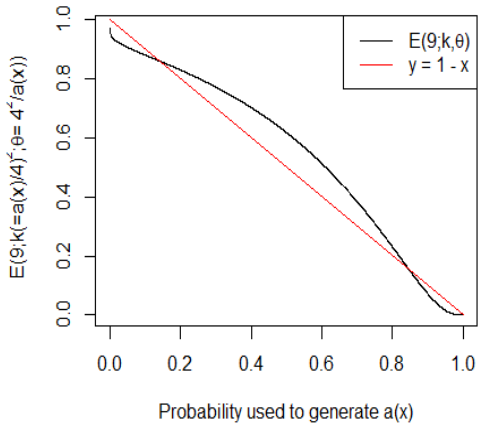


Figure 28: Relation between a(x) & ε versus the relation between e(x) & α

As using this method does not result in the wanted relationship between the estimates and the realized times, we present three options for creating the time estimates.

Option 1: The Gamma distribution using a normal distribution-based method

The first option is to use the same method as for the normal distribution, even though we know this creates estimates that cause the realized times to not be gamma distributed values from these estimates.

A gamma distribution *a(x)* is created based on the realized time duration α from the airport's logs and σ_{new}, then a random value ε is drawn from this distribution. This drawn value is our new time estimated duration. From this random variable a new distribution *e(x)* is created, using the random variable as mean and using σ_{new}. The problem is optimized based on the collection of all new time estimates ε and the performance is evaluated based on the realized times α. For the MC overlap determination, its distribution *e(x)* is used.

As previously noticed, this approach results in the problem that the probability density of α in *e(x)* is not equal to the probability density of ε in *a(x)*, and because the process of drawing ε from *a(x)* is according to our wanted Gamma distribution, this means that α is not a representative Gamma distributed draw from the newly generated *e(x)*.

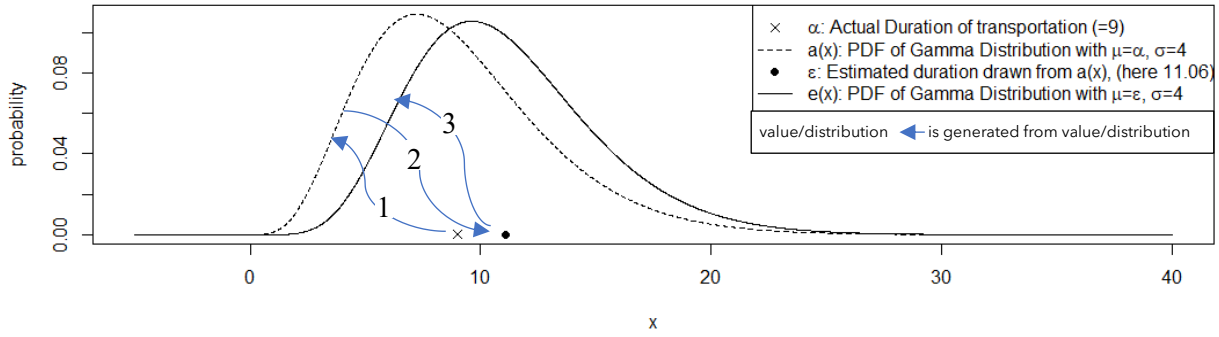


Figure 29: Option 1 for incorporating stochasticity

Option 2: Use the normal distribution

The second option is to use the same method as in the first option, but instead of using the Gamma distribution, using the normal distribution to generate new durations. As normal distributions are symmetric, the probability density of α in *e(x)* is equal to the probability density of ε in *a(x)*.

A drawback of this approach is that we know that for the current classifiers that are used to determine the expected time durations, the resulting groups tend more towards gamma distributed duration times (see Section 2.1.5). Another drawback of using the Normal Distribution is that values can become negative at higher levels of stochasticity. If we want to ensure values cannot become negative by using the truncated normal distribution, then the distribution is not symmetric anymore.

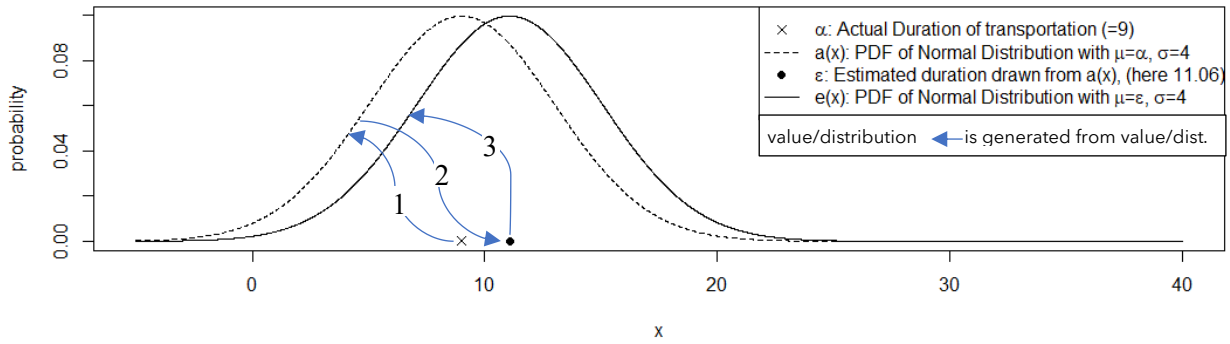


Figure 30: Option 2 for adding stochasticity

Option 3: Treat the realized values as predictions

The third option is to treat the realized times in the airport's logs as estimates, and these realized times to optimize the assignments and then generate new times that we will use to evaluate performance of the assignment. This is done by creating a gamma distribution based on the realized time and the wanted level of stochasticity and draw a random variable from this distribution that will be used to evaluate the performance of the assignment.

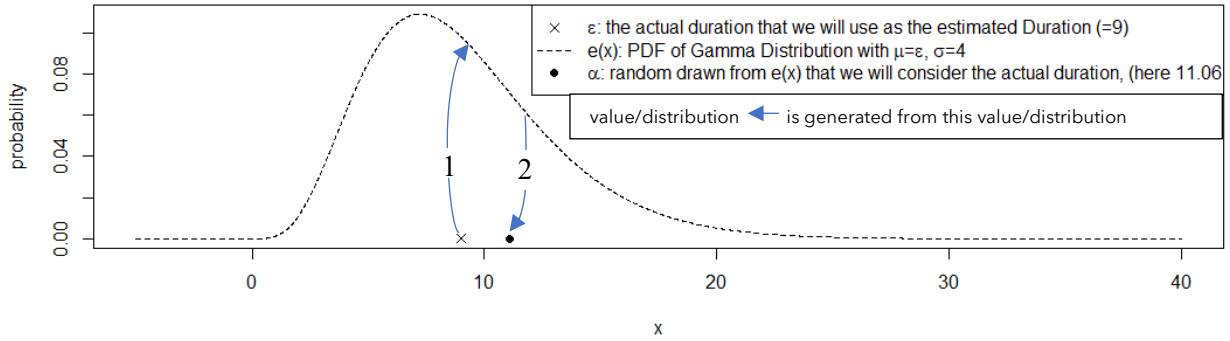


Figure 31:Option 3 for incorporating stochasticity

Using this approach, the relationship between the time estimation (ε) and the actual duration (α), is equal to the real life situation that we try to model, in which flights with a certain set of characteristics are expected to have a specific distribution for the estimation of the duration time.

Another advantage of this approach is that optimization is only needed once for each scenario (for instance, one day of flights with $\sigma_{arrival\_time} = 1$, $\sigma_{transport} = 3$, $\sigma_{on\_belt} = 3$), after which we can draw multiple sets of actual durations. As the optimization process takes much longer than performance evaluation, we can test performance for a larger number of experiments in the same amount of time.
A drawback of this approach is that we assess the quality of the assignments (and thus the performance of the algorithms) based on newly generated estimates, so we must validate whether the newly created set of artificial durations still reflect the actual data properly.

Our first question is whether the newly generated 'realized times' are comparable to the log's realized times. To do so, we investigate the number of flight pairs overlapping in the log files and in the newly

generated set of flight times. Flights do not need to be assigned to the same belt to be considered overlapping. For one week of flights, we have simulated new arrival times, transportation times and on-belt duration times according to the process described in Section 5.1.1, with various values for the standard deviations and five replications per experiment. The number of flight pairs that will overlap have been determined for both the original flight times and the newly generated flight times. In Table 23 we show the increase in overlapping flight pairs from the original times to the newly generated times. The table for instance shows that when we create a new dataset based on a σ_arrival = 3, σ_transport = 6 and σ_on-belt = 6 (the bottom-right cell), the new dataset has 0.5% less flight pairs overlapping (averaged over 5 replications) than the original dataset. The total number of overlapping flight pairs (originally 4200+), does not structurally increase or decrease after adding the stochasticity.

Table 23: percentage of extra flight pairs considered overlapping, averaged over 5 replications

	σ_Transport & σ_on-belt						
σ_arrival	0	1	2	3	4	5	6
0	0.0%	-0.2%	-0.1%	0.1%	0.1%	0.5%	-0.2%
1	0.3%	-0.1%	-0.4%	0.3%	-0.3%	0.0%	0.6%
2	-0.3%	-0.4%	0.5%	-0.5%	0.2%	-0.6%	-0.6%
3	0.3%	0.3%	-0.5%	0.6%	-0.3%	-0.1%	-0.5%

In Figure 32, we show the durations of the overlaps after adding stochasticity. The overlap of the flights that do overlap, shows similar results for the newly generated times as for the original times. As the number of flights overlapping, and the duration of overlap for the flights is very similar for the original times and the newly generated times, we consider the third method introduced in Section 5.1.1 to be a valid method and suitable for use in the experimentation.

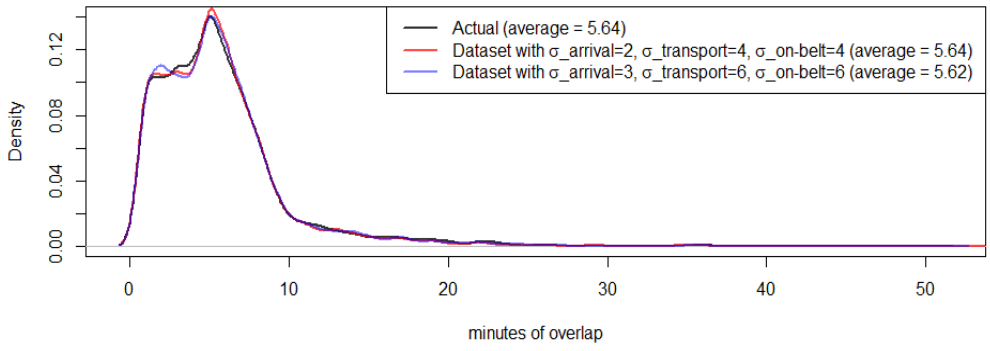


Figure 32: The density of the overlap durations for one week of data, before and after adding stochasticity

Conclusion and implementation

The third approach seems most suitable to use in the experimentation as it is the only approach that incorporates the impact of the stochasticity of the time estimates correctly, while also reducing the number of times the optimization algorithms must be run. We found that the newly generated dataset has a comparable amount of overlapping flight combinations and that the duration of the overlap is also

comparable the original data. Therefore, we use the third approach and the times from the logs will now be considered estimates and a random draw from the generated distribution will be considered the realized time.

5.2 – The number of replications in the experiments

In this section we determine the needed number of replications in the experiments. First, we determine the number of validation sets we must generate to create stable outcomes. Next, we investigate the impact of using different MC simulated overlap estimates. If there is a significant difference between the outcomes of different sets of MC simulated overlap, then we need to run multiple replications of overlap simulation or increase the number of samples in each MC overlap estimation.

5.2.1 – The number of new datasets to replicate

For each of the optimization methods, one week of data was optimized (without a safety factor) and 10 evaluation sets were formed, based the highest level of stochasticity to be simulated (σ_arrival = 3, σ_transport = 6, σ_on-belt = 6). These parameters were chosen because they are expected to generate the highest level of stochasticity for the output. The outcomes of these 10 evaluation sets are shown Table 24. The standard errors of these results are shown in Table 25. Here we see that at least 5 evaluation sets are needed to realize a standard error below 5% for all solution approaches. Therefore, we will use 5 evaluation sets for performance evaluation times.

Table 24: Results of 10 evaluation sets for performance evaluation times (σ_arrival = 3, σ_transport = 6, σ_on-belt = 6 and safety factor = 0)

	Basic overlap estimates					MC overlap estimates				
Rep.	*MIP*	*SA*	*Greedy*	*FCFS*	*FCFS +*	*MIP*	*SA*	*Greedy*	*FCFS*	*FCFS +*
1	-7,136	-7,757	-6,974	-7,440	-7,785	-7,296	-7,738	-7,597	-7,424	-7,795
2	-7,204	-7,091	-7,130	-7,051	-7,719	-7,755	-7,660	-7,492	-7,451	-7,385
3	-7,186	-7,213	-6,638	-7,035	-7,606	-7,257	-7,459	-7,058	-7,090	-7,299
4	-7,328	-7,113	-6,828	-7,001	-7,129	-7,470	-7,421	-7,130	-7,267	-7,228
5	-7,115	-7,277	-6,752	-7,038	-7,489	-7,467	-7,712	-7,340	-7,236	-7,529
6	-7,229	-7,363	-6,394	-6,931	-7,540	-7,679	-7,405	-7,168	-7,225	-7,245
7	-7,437	-7,646	-7,167	-7,235	-7,704	-7,020	-7,931	-7,527	-7,295	-7,459
8	-7,371	-7,616	-6,854	-7,029	-7,550	-7,315	-7,617	-7,359	-7,276	-7,422
9	-7,329	-7,532	-6,745	-7,081	-7,397	-7,382	-7,662	-7,504	-7,451	-7,422
10	-6,870	-6,918	-6,348	-6,477	-7,008	-7,343	-7,065	-6,798	-6,927	-6,843

Table 25: Standard errors for n replications of performance evaluation times

	Basic overlap estimates					MC overlap estimates				
Rep.	*MIP*	*SA*	*Greedy*	*FCFS*	*FCFS +*	*MIP*	*SA*	*Greedy*	*FCFS*	*FCFS +*
1	-	-	-	-	-	-	-	-	-	-
2	0.060	0.597	0.139	0.350	0.054	0.194	0.065	0.089	0.023	0.353
3	0.012	0.122	0.094	0.081	0.030	0.039	0.048	0.101	0.070	0.090
4	0.018	0.070	0.049	0.047	0.066	0.020	0.033	0.059	0.036	0.056
5	0.014	0.046	0.035	0.032	0.043	0.014	0.024	0.039	0.025	0.037
6	0.011	0.035	0.042	0.027	0.032	0.011	0.022	0.032	0.002	0.031
7	0.014	0.031	0.036	0.022	0.026	0.009	0.023	0.026	0.016	0.025
8	0.013	0.028	0.031	0.019	0.023	0.008	0.020	0.023	0.013	0.021
9	0.012	0.025	0.027	0.017	0.021	0.007	0.017	0.02	0.012	0.018
10	0.016	0.029	0.031	0.027	0.026	0.006	0.024	0.027	0.017	0.026

5.2.2 – The number of overlap sets to replicate using Monte Carlo

Because we perform 20.000 MC simulations per flight pair to determine the expected overlap between two flights, the relative error of the estimation becomes small, and as performing 20.000 MC replications per flight combination is a time intensive process, we want to use only one MC overlap estimation per scenario of standard deviations. For any flight pair, the expected overlap should not differ much if we were to calculate it again. However, the outcomes of optimization with new overlap estimates should also not differ significantly. To test this, for one week of data we have calculated the expected overlap between the flight combinations five times. For each of these overlap estimation replications, we have run the SA algorithm five times and determined the realized performance, of which the results are shown in Table 26. These runs are based on the unbiased estimates as explained in Section 4.4.3.

Table 26: SA results for multiple replications of determining the overlap between flight pairs

	Overlap Replication 1	Overlap Replication 2	Overlap Replication 3	Overlap Replication 4	Overlap Replication 5
SA Rep 1	-8,120	-7,972	-8,221	-8,057	-8,213
SA Rep 2	-8,287	-7,956	-8,023	-8,042	-8,207
SA Rep 3	-7,816	-8,119	-7,988	-8,071	-8,125
SA Rep 4	-8,014	-8,115	-8,114	-8,194	-7,949
SA Rep 5	-8,061	-8,070	-8,309	-8,189	-8,099

Performing an analysis of variance (ANOVA), it is found that there are no significant differences (α=0.05) between results following from different overlap estimates. Therefore, we will not create multiple instances of expected overlap in the experiments.

Table 27: Analysis of variance for overlap estimation

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	28,479.76	4	7,119.94	0.506086	0.731774	2.866081
Within Groups	28,137.,8	20	14,068.64			
Total	309,852.6	24				

5.3 – Running the experiments

In the experimentation, common random numbers are used where possible. This means that all algorithms will create schedules based on the same estimates and the resulting assignments will be evaluated based on the same newly generated times. The experiment structure is shown in Figure 33.

The deterministic approaches are executed only once per day per safety factor (lines 1-6), except for the FCFS approach incorporating future demand. The stochastic approaches are executed once for every combination of standard deviations and the safety factor within those days, as each combination of standard deviations and safety factors will result in different MC overlap estimates (lines 7-12). Also, estimates used for the FCFS approach exclude arrival time uncertainty (line 9).

For the given number of replications (5, following from Section 5.2), new realized times are created (lines 13-16). Once these new times are known, these can be used to create estimates for the deterministic and stochastic FCFS approach incorporating future demand (line 17). This is the only approach that requires the new realized times, as the solution approach uses a mix of known arrival times and arrival time estimates. After optimizing the FCFS approaches incorporating future demand (lines 18 and 19), we use the generated times to evaluate the performance of the Greedy, SA and FCFS approach incorporating future demand. As the regular FCFS approach is not subject to arrival time uncertainty, the arrival time uncertainty is removed from the new times (line 21) before evaluating the performance of the approach (line 22).

```
For every data set of one day that we will assess{
  For every safety factor{
    Determine the expected overlap based on the overlap in expected on-belt times
    Optimize FCFS based on the basic overlap estimates
    Optimize Greedy based on the basic overlap estimates
    Optimize SA based on the basic overlap estimates
    Optimize with MIP solver based on the basic overlap estimates
    For every combination of standard deviations that we want to assess{
      Determine Expected overlap with MC simulations inc. arrival time uncertainty
      Determine Expected overlap with MC simulations exc. arrival time uncertainty
      Optimize with FCFS based on MC overlap estimates
      Optimize with Greedy based on MC overlap estimates
      Optimize with SA based on MC overlap estimates
      Optimize with MIP solver based on MC overlap estimates
      For every replication{
        For every flight{
          Generate the new actual arrival times, used for performance measurement
        }
        Determine basic & MC estimates for FCFS+Future demand*
        Optimize FCFS with future demand based on basic overlap estimates
        Optimize FCFS with future demand based on MC simulated overlap estimates
        Determine performance of Greedy, SA & FCFS+ assignments based on new times
        Remove arrival time uncertainty out of the new actual times
        Determine performance of regular FCFS assignments
} } } }
```

Figure 33: Schematic overview of running the experiments

Preliminary experimentation has shown that the FCFS algorithm and the Greedy algorithm both only take a maximum of a few seconds for one day of data. The Simulated Annealing approach is normally set to a cooling scheme of five minutes for a smaller problem instance. However, as we saw in Section 2.2.3, even with a much shorter cooling scheme, we get relatively close to the values we find at the five-minute scheme. To be able to do a broader range of experiments, the cooling scheme is set to one minute.

Table 28: All factors used in experimentation

Experimental factor	Values
Day	Seven consecutive days of historical data: Day 1, Day 2, Day 3, Day 4, Day 5, Day 6, Day 7.
σ_arrival (in minutes)	0, 1, 2, 3
σ_transport (in minutes)	0, 2, 4, 6
σ_on-belt (in minutes)	0, 2, 4, 6
Safety factor (in minutes)	0, 2, 4, 6
Overlap estimation	Basic (= based on overlap of expected on-belt time windows), Based on the average of 20,000 MC simulations.
Solution Approach	MIP solver, SA, FCFS, Greedy, FCFS + future demand

The standard deviation of the on-belt duration and that of the transportation time are incorporated in the experimentation in steps of 2 minutes, to reduce the total runtime of the experiment by about two-third. In Table 28 we show all experimental factors and their possible values. Every combination of items in this list will be run.

The calculation time of the solution approaches and the time it takes to determine the MC overlap estimates using the 20.000 MC simulations is shown in Table 29. The values processing times are for one day of data, based on weekly averages. Safety factors are not considered in this overview. The calculation times were determined for three different levels of stochasticity, for which the specific standard deviation values also reported in the table.

Table 29: Calculation times in seconds to process one day of data, for processes used for solution creation

		Little uncertainty	**Medium uncertainty**	**High uncertainty**
	σ arrival time	1 minute	2 minutes	3 minutes
	σ transportation time	2 minutes	4 minutes	6 minutes
	σ on-belt duration	2 minutes	4 minutes	6 minutes
Basic Overlap estimates	MIP Solver	60	60	60
	SA	60	60	60
	FCFS	0.01	0.01	0.01
	Greedy	0.8	0.8	0.8
	FCFS +	2.5	2.5	2.6
	Estimate MC overlap	71	75	81
MC overlap estimates	MIP Solver	60	60	60
	SA	60	60	60
	FCFS	0.01	0.01	0.01
	Greedy	0.8	0.8	0.8
	FCFS +	1.1	1.5	2.6

We see that generally, the degree of uncertainty does not have a large impact on the computation times. The MIP solver and the SA approach are both time restricted, and the regular FCFS approach takes an equal amount of time for all levels of uncertainty for both the basic overlap estimation method and the MC overlap estimation methods. For the Greedy approach this also holds.

The time it takes to calculate the MC overlap estimates, and the time the FCFS with future demand takes when using MC overlap estimates, are however both subject to the level of uncertainty. When there is less uncertainty, less flights are considered as possibly overlapping and therefore the sub-problems become smaller in the FCFS with future demand approach.

The regular FCFS approach is clearly the fastest solution approach, followed by the Greedy approach and the FCFS incorporating future demand.

5.4 – Conclusion

As the airport expects to increase the accuracy of the time estimates, we want to assess the performance of the solution approaches from Chapter 4 for these improved time estimates. Therefore, we presented a set of experimental factors that will be adjusted throughout the experimentations, and we presented the values that will be used. We have also explored possibilities to incorporate any wanted level of stochasticity in the airport's estimates. With the lack of suitable estimators in the current data, the option to treat the realized times in historical data as time estimates, and creating 'new' realized times by sampling the same distributions as used by the MC overlap estimation, seems the best way to create the right relationship between the estimates and the realized times.

The approach was validated by comparing the number of flight pairs considered overlapping (without taking into account the assigned belts) in the original data, and in the newly generated datasets. Also, the quantity of overlap between overlapping flights in the original data and the newly generated datasets. Both the number of flight pairs overlapping and the amount of overlap between the overlapping flight pairs, seem to be comparable, even when with newly generated datasets with a high level of stochasticity.

Next, we presented a structure to run the experiments in a structured way, using the power of common random numbers where possible, and we defined the values of the factors that we will adjust. Finally, we showed the average computation times for different solution approaches.

Chapter 6: Results and Analysis

In this chapter we discuss the results of the experimentation, and analyze the general behavior of the solution methods under different levels of stochasticity. First, in Section 6.1, we give a short explanation on the visualization of the outcomes that are used throughout this chapter. Second, in Section 6.2, we present the best solution approaches for each combination of stochasticity in the estimates. In section 6.3 we explore the impact of stochasticity on the optimization methods. Next, we evaluate at all measures taken to combat stochasticity in the estimates: In section 6.4, we discuss the trade-off between using basic overlap estimates and MC overlap estimates. In section 6.5 we discuss the impact of using safety factors, and in Section 6.6 we show the differences in performance between the FCFS results and the SA results.

6.1 – Data presentation

As every experiment has multiple input factors that might interact with one another, it is complicated to show the impact of the input data on the output in a straightforward manner. To provide a comprehensive view of the results, we have implemented the table structure as shown in Table 30, which is a table of tables.

Table 30: Structure of output representation, to show effects and interactions of four input variables

Safety factor		σ Arrival time															
		0				1				2				3			
		σ On-belt				σ On-belt				σ On-belt				σ On-belt			
		0	2	4	6	0	2	4	6	0	2	4	6	0	2	4	6
0	σ Transport 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	σ Transport 2	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
	σ Transport 4	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
	σ Transport 6	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
2	σ Transport 0	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
	σ Transport 2	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
	σ Transport 4	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112
	σ Transport 6	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128
4	σ Transport 0	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144
	σ Transport 2	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160
	σ Transport 4	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176
	σ Transport 6	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192
6	σ Transport 0	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208
	σ Transport 2	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224
	σ Transport 4	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240
	σ Transport 6	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256

In the example, each column under the "σ Arrival time", consists of four columns with on-belt duration standard deviations. Each row of safety factors, consist of four rows of transportation time standard deviations. The highlighted cell in the center of the table (with '72' in it), represents the output of an experiment with σ Arrival time = 1, σ On-Belt = 6, σ Transport = 0, and a safety factor of 2 minutes. The tables are often color-coded to make relations clear. The color is always linked to the value in the cell, and sometimes this is relative to the other values in the table. When a red-green scale is used, green values are preferable considering the objective.

6.2 – The best solution method per stochasticity level

In this section, the best solutions and their approaches are presented, for the different combinations of stochasticity in the estimators. In Appendix I, we show all experimental outcomes, averaged per week, for all solution approaches, combinations of stochasticity, and safety factors. For each combination of stochasticity, we determined the combination of solution approach and safety factor with the best average weekly results, shown in Table 31.

Table 31: The best scoring solution approaches per combination of stochasticity

σ Arrival time		σ Transport		σ On-belt 0	σ On-belt 2	σ On-belt 4	σ On-belt 6
	0		0	MIP_basic (0)	SA_basic (4)	SA_MC (0)	SA_MC (0)
			2	SA_MC (0)	SA_basic (6)	SA_MC (0)	SA_MC (4)
			4	SA_MC (2)	SA_MC (2)	SA_MC (2)	SA_MC (4)
			6	SA_MC (0)	SA_MC (2)	SA_MC (0)	SA_MC (2)
	1		0	SA_basic (2)	SA_basic (4)	SA_basic (6)	SA_MC (0)
			2	SA_MC (0)	SA_MC (0)	SA_MC (0)	SA_basic (6)
			4	SA_MC (2)	SA_MC (4)	SA_MC (2)	SA_MC (2)
			6	SA_MC (2)	SA_MC (4)	SA_MC (2)	SA_MC (6)
	2		0	FCFS+fut_MC (0)	SA_basic (4)	SA_basic (4)	SA_basic (4)
			2	SA_MC (2)	SA_basic (4)	SA_MC (0)	SA_MC (2)
			4	SA_MC (2)	SA_MC (4)	SA_MC (2)	SA_MC (4)
			6	SA_MC (0)	SA_MC (2)	SA_MC (4)	SA_MC (2)
	3		0	FCFS+fut_MC (0)	FCFS+fut_MC (0)	FCFS_MC (0)	FCFS_MC (0)
			2	FCFS+fut_MC (0)	FCFS+fut_MC (0)	FCFS+fut_MC (0)	FCFS+fut_MC (0)
			4	SA_MC (2)	SA_MC (2)	SA_MC (2)	SA_MC (0)
			6	SA_MC (2)	SA_MC (2)	SA_MC (6)	SA_MC (2)

Solution approach	Overlap estimation
MIP	basic
SA	basic
SA	MC simulations
FCFS	MC simulations
FCFS+future demand	MC simulations

We find that for the analyzed cases, the MIP solver approach is only preferable if there is no stochasticity at all. The SA approach using basic overlap estimates is generally preferable when the uncertainty of the arrival time is 2 or lower and that of the transportation time is around 0.

The FCFS approaches with MC overlap estimates are preferred when there is a lot of arrival time uncertainty, but little transportation time uncertainty. The safety factor does not increase performance for the FCFS approach in these situations. When the on-belt duration uncertainty is low, it is better to include future demand. In all other cases, the SA approach using MC overlap estimates is generally the best solution method.

Table 32 shows the average performance over five replications for the solution approaches of Table 31. In Appendix J, we show what happens if instead of picking the solution approach with the best average result, we select the solution approach with the best worst-case result. It also shows the performance difference between the average of the best average solutions and the worst case of the best worst-case solutions. There is very little difference in the preferred overlap estimation approach and the solution approach when using the worst-case results.

Table 32: The realized performance of the best scoring solution approaches per combination of stochasticity

				σ On-belt			
				0	2	4	6
σ Arrival time	0	σ Transport	0	-11,398	-10,832	-10,381	-10,125
			2	-10,412	-10,181	-9,869	-9,541
			4	-9,649	-9,650	-9,298	-9,095
			6	-9,191	-8,999	-8,891	-8,681
	1	σ Transport	0	-10,902	-10,523	-10,145	-9,770
			2	-10,273	-10,037	-9,860	-9,466
			4	-9,627	-9,524	-9,269	-9,024
			6	-9,063	-8,945	-8,852	-8,584
	2	σ Transport	0	-10,495	-10,180	-9,823	-9,525
			2	-10,038	-9,824	-9,637	-9,387
			4	-9,491	-9,404	-9,059	-8,887
			6	-8,994	-8,893	-8,882	-8,596
	3	σ Transport	0	-10,487	-9,939	-9,655	-9,370
			2	-9,712	-9,512	-9,368	-9,243
			4	-9,328	-9,181	-8,948	-8,668
			6	-8,758	-8,722	-8,576	-8,454

6.3 – The impact of stochasticity on the solution approaches

In this section we fit a simple linear regression model over the outcomes of each of the solution approaches, using the values of the stochasticity as predictors. Only experiments without a safety time are used for this analysis. For variable selection, every variable with a *p* value below 0.05 is included in the model. The goal of this analysis is to show the impact of different kinds of stochasticity on the outcomes of the solution approaches. To keep outcomes of the models simple, interactions between the different levels of stochasticity are not considered.

Table 33: Linear regression models for different solution approaches (fitted for solutions without safety factors and without interaction)

		Intercept	Arrival	Transport	OnBelt	Adj. R^2
basic overlap estimation	MIP	-10,474	251	354	117	0.93
	SA	-10,441	238	339	121	0.93
	Greedy	-10,142	252	359	125	0.92
	FCFS	-10,019	0	291	120	0.98
	FCFS+	-10,253	81	302	121	0.96
MC overlap estimation	MIP	-10,369	167	211	105	0.95
	SA	-10,774	193	234	119	0.94
	Greedy	-10,441	184	222	108	0.93
	FCFS	-10,026	0	252	105	0.96
	FCFS+	-10,204	0	229	110	0.95

First off, in Table 33, we see that for the FCFS approaches without future demand, the standard deviation in the arrival time had no significant impact on the outcomes and was therefore excluded as an estimator. As the purpose of the method is to mitigate the arrival time uncertainties, this is in line with expectations. For the FCFS approaches that consider future demand, we see that for the basic overlap estimation method, the stochasticity in the arrival time estimates is included in the model, but with a smaller coefficient than the Greedy, SA and MIP approach. As in the solution approach, for the flight that is being planned the arrival time is known, but future flights with uncertain arrival times are also considered while making the assignment, the inclusion of the estimator with a relatively low coefficient is also in line with expectations.

Besides the arrival times coefficients of the FCFS approaches, the coefficients per type of overlap estimation are all pretty similar. Uncertainty in transportation times has the biggest impact on the results for all solution approaches, followed by arrival time uncertainty (except for the FCFS approaches) and uncertainty in the on-belt duration. The impact of each type of uncertainty is higher in almost all basic overlap estimation variants of the solution approaches than in the MC overlap estimate variants of the solution approaches.

Linear regression models considering interactions and the safety factor, are stated in Appendix K. The predictive power of those models is higher. Appendix K shows that for most models, there is some interaction between the levels of stochasticity for the different estimators. This interaction effect always decreases the goal function. Presumably because the different sources of uncertainty weaken one another. Especially between the arrival time uncertainty and transportation time uncertainty the interaction coefficient is large. As the safety factor increases however, the interaction effects become insignificant for most approaches.

6.4 – Basic overlap estimates versus MC overlap estimates

Although the MC overlap solutions give better results for most stochasticity levels (especially the higher levels of stochasticity, like the current levels), there are also downsides on the usage of the MC overlaps. Not only is it more time intensive to create the estimates, but a clear visualization of the problem and its solution is only possible for the basic overlap estimates. Therefore, it may be hard to understand why a planning is good.

To clarify this visualization problem, the results of a day's planning based on the basic overlap estimates (optimized with SA) are presented in Figure 34, and that based on MC overlap estimates (with σ arrival = 3, σ transport = 6, σ on-belt = 6) are presented in Figure 35. For both schedules, we have calculated the expected goal function based on basic overlap estimates, and based on the MC overlap estimates, shown in Table 34.

Table 34: The expected performance for basic overlap and MC overlap optimizations, for the basic and the MC goal function

		Basic overlap SA optimization	*MC overlap SA optimziation*
Expected results based on the basic goal function	Minutes of overlap	55	85
	Alliance flight arriving at occupied belt penalty	63	117
	Belt preference bonus	-320	-328
	Total	**-202**	**-126**
Expected results based on Monte Carlo goal function	Mins overlap	95.2	92.3
	Alliance empty belt	128.1	115.6
	Belt preference bonus	-320	-328
	Total	**-96.7**	**-120.1**

Comparing Figure 34 and Figure 35 (which correspond to the 'Expected results based on the basic goal function' in Table 34), it seems like the assignment based on the MC overlaps has a lot more overlap (85 minutes versus 55), and has almost double the number of alliance flights that will arrive at an occupied belt (13 versus 7). However, according to the MC overlap estimates and blocking probabilities, the assignments based on the basic overlap estimates will result in more overlap, and more alliance flights to arrive at occupied belts.

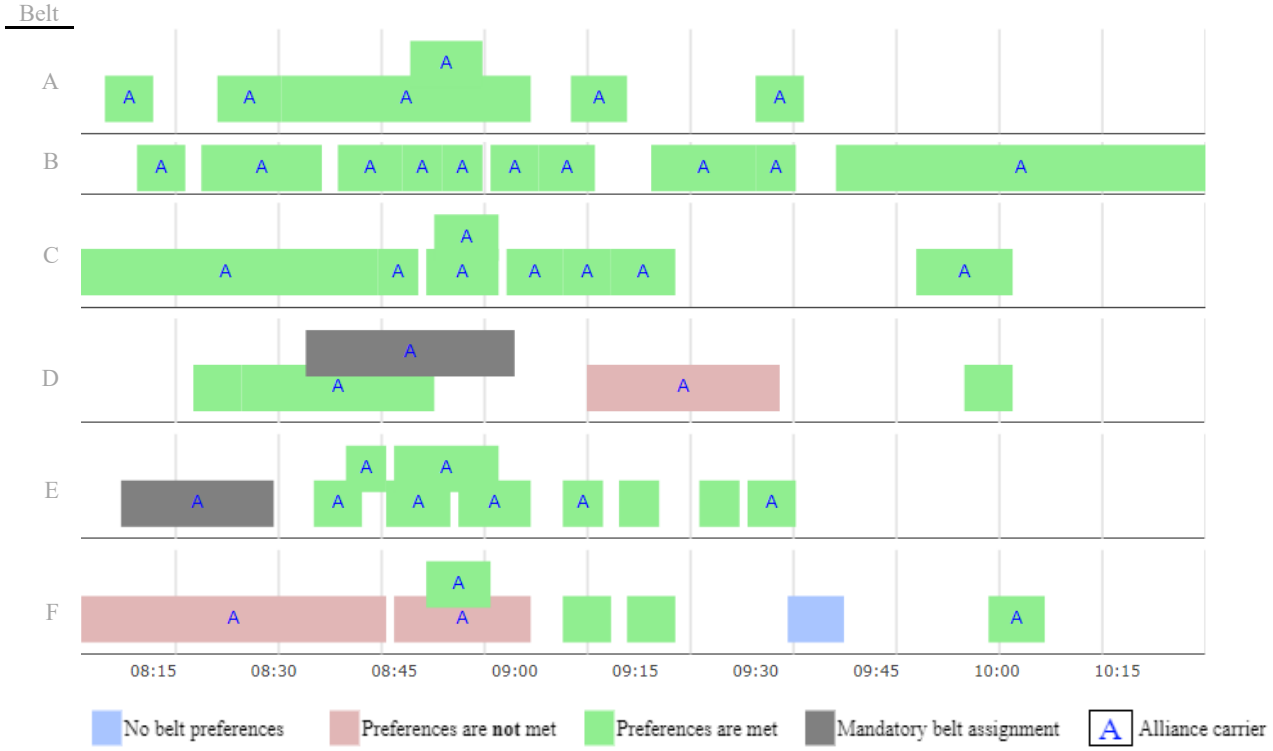


Figure 34: Resulting assignments of SA optimization based on basic overlap estimates

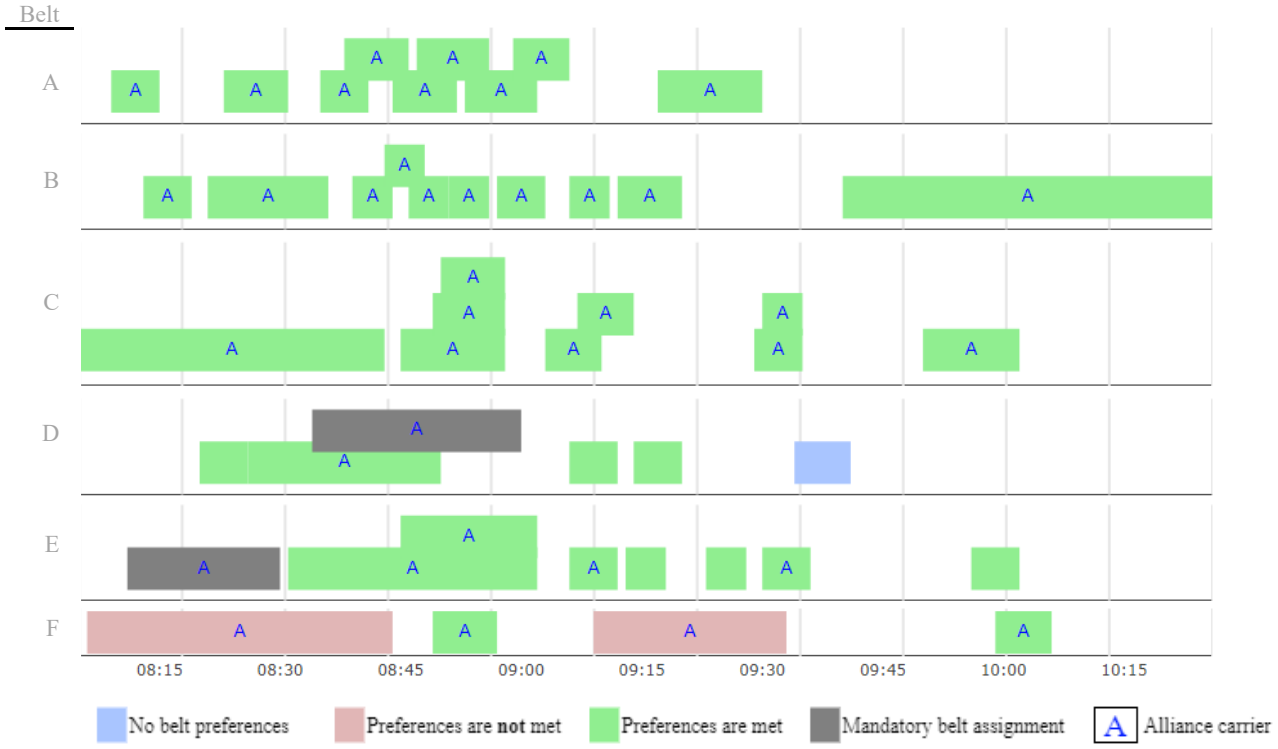


Figure 35: Resulting assignments of SA optimization based on Monte Carlo overlap estimates

In Table 35 we show the realized performance for ten replications of newly generated realized times. Corresponding to the results of Section 6.2, the assignments based on the MC overlap estimates clearly generate better results. So, there is a trade-off to be made between being able to visualize the problem and the solutions in a straightforward manner, and the performance of the assignments.

Table 35: Realized performance of basic overlap schedules and MC overlap schedules for two hours of data, with σ_arrival = 3, σ_transport = 6, σ_on-belt = 6

	Basic overlap estimates optimation				MC overlap estimates optimization			
	Minutes of overlap	Alliance flight at occupied belts penalty	Belt preference bonus	Total objective value	Minutes of overlap	Alliance flight at occupied belts penalty	Belt preference bonus	Total objective value
Rep1	96	135	-320	-89	65	144	-328	**-119**
Rep2	92	162	-320	-66	110	135	-328	**-83**
Rep3	89	126	-320	**-105**	111	153	-328	-64
Rep4	78	126	-320	-116	60	117	-328	**-151**
Rep5	78	117	-320	**-125**	89	135	-328	-104
Rep6	175	189	-320	44	161	171	-328	**4**
Rep7	84	153	-320	-83	75	126	-328	**-127**
Rep8	98	162	-320	-60	101	135	-328	**-92**
Rep9	123	162	-320	-35	97	126	-328	**-105**
Rep10	140	162	-320	**-18**	149	171	-328	-8
Average	105.3	149.4	-320	-65.3	101.8	141.3	-328	**-84.9**

Therefore, in Table 36, we show the performance loss when using basic overlap estimates: the difference between the best solution following from basic overlap estimates (including the best safety time) and the best solution following from MC overlap estimates, based on one week of data. A positive value means that the MC overlap estimates performed better, and a negative value means that the basic overlap estimates performed better. The higher the standard deviations, the more beneficial it is to use the MC overlap estimates. Especially the standard deviation in the transportation estimate has a large impact on the difference between the estimation approaches.

The performance of the basic overlap estimates can however be increased if the safety factors are further optimized for the basic overlap estimates. Now, safety times were static per experiment, and steps of two minutes were used. It might for instance be interesting to look at safety factors that depend on the baggage class of the flights, because as shown in Chapter 2, there is more uncertainty transportation times and on-belt durations for baggage class C flights. With a better policy, the values in Table 36 could therefore be decreased.

Table 36: The performance loss when using basic overlap estimates relative to best MC overlap estimate solution

				σ On-belt			
				0	2	4	6
σ Arrival time	0	σ Transport	0	-93	-211	81	220
			2	30	-55	47	139
			4	260	401	230	272
			6	560	374	470	644
	1	σ Transport	0	-63	-72	-10	8
			2	57	4	163	-29
			4	221	271	207	122
			6	444	339	523	297
	2	σ Transport	0	60	-114	-70	-86
			2	10	-39	191	140
			4	246	265	249	317
			6	432	444	539	516
	3	σ Transport	0	238	89	170	187
			2	178	52	89	214
			4	383	268	377	281
			6	524	397	477	374

6.5 – The impact of not using the best-known safety factor for SA optimization

In Section 6.1, it was shown that the SA approaches (either based on basic overlap estimates or on MC overlap estimates) often provide the best solutions. In this section, we will therefore use the outcomes of the SA approaches to show the impact of not using the best-known safety factor for every combination of standard deviations. In Table 37 we show the performance losses for optimization using basic overlap estimates, and in Table 38 for optimization using MC overlap estimates. Empty cells indicate that the safety factor corresponding to the cell gave the best results for the corresponding standard deviations.

For optimization with the basic overlap estimates, we find that as soon as there is some uncertainty present in the estimates, we immediately need a safety factor of at least 4 minutes, and when the estimates of the transportation time have a standard deviation of 4 minutes or more, the optimizations with a safety factor of 6 minutes perform best. Not using a safety factor has a large impact on the performance of optimization for all cases in which there is uncertainty in the estimates. The impact is probably even larger, as we have not optimized the safety factors and used steps of two minutes.

Table 37: Performance loss for one week of data when a non-optimal safety factor is used, for SA optimization based on basic overlap estimates

Safety factor	σ Transport	σ Arrival time	0				1				2				3			
			σ On-belt				σ On-belt				σ On-belt				σ On-belt			
			0	2	4	6	0	2	4	6	0	2	4	6	0	2	4	6
0	0			262	359	504	266	265	339	287	495	390	581	396	759	614	570	459
	2		550	603	489	368	524	597	569	634	697	727	538	614	695	689	929	779
	4		679	662	802	672	728	759	772	815	716	615	532	506	747	766	651	534
	6		480	764	673	483	648	673	611	676	741	584	726	679	512	640	458	820
2	0		28	107	144	144		30	47	105	160	105	195	264	361	324	303	143
	2		163	266	350	209	209	204	115	312	292	387	211	390	311	286	348	738
	4		259	136	417	397	384	315	441	592	401	303	321	316	307	421	248	367
	6		322	530	398	370	375	419	304	470	395	362	463	227	345	338	275	496
4	0		107		16		21		94	156	31				209	149	92	65
	2			7	133		13		46	94			118		53	93	99	160
	4		103		228	82	115	106	221	332	280	212	47	149	29	54	83	204
	6			259	122	0	168	173	343	384	144	223	0	110	24	178	111	0
6	0		230	123		22	110	19				55	15	165				
	2		35			29		30			28	10		42				
	4			30														
	6		54			15							103					60

Table 38: Performance loss for one week of data when a non-optimal safety factor is used, for SA optimization based on Monte Carlo overlap estimates

Safety factor	σ Transport	σ Arrival time	0				1				2				3			
			σ On-belt				σ On-belt				σ On-belt				σ On-belt			
			0	2	4	6	0	2	4	6	0	2	4	6	0	2	4	6
0	0													18	97			12
	2					25				29	48			224				
	4		95	215	55	142	21	157	100	134	42	45	95	27	200	189	79	
	6			94		164	100	110	133	101	0	98	116	245	27	4	95	219
2	0		666	163	124	335	263	113	64	72	37	16	47			53	64	
	2		119	31	24	37	36	24	112			1	119		19	110	75	140
	4					82		79				69		1				8
	6		77		123			44		173	68		198				17	
4	0		1123	533	469	591	752	527	365	389	296	234	176	113	174	167	116	99
	2		453	208	244		380	264	269	259	198	166	208	208	33	90	239	151
	4		70	289	86		83		2	48	98		1		89	56	122	45
	6		207	18	216	206	75		191	55	98	155		199	22	81	188	23
6	0		1709	966	845	719	1333	897	775	543	860	645	318	177	394	477	278	215
	2		849	590	611	306	715	590	434	172	442	447	301	433	360	205	176	145
	4		258	298	114	220	253	233	184	186	184	178	227	198	282	140	62	195
	6		279	86	152	159	182	108	69		144	188	342	111	124	42		104

For optimization with MC overlap estimates, when in the estimates the standard deviation of the transportation time is 2 minutes or less, not using a safety factor generally results in the best solutions. Using a large safety factor (of 4 or 6 minutes) will have a large negative effect. For estimates with a standard deviation of 4 or more for the transportation time, a safety factor of 2 minutes will generally give better results, but not using safety factors will not have as large of an impact as in the case with basic overlap estimates.

In Appendix L, we show how the combination of uncertainties and safety factors impact the percentage of flights wrongfully expected to overlap, and wrongfully expected to not overlap.

6.6 – Simulated Annealing versus FCFS approaches

In this section, we compare the performance of the SA optimization, with that of the FCFS techniques. In Table 39 and Table 40 we compare performance of the SA with the FCFS that does not incorporate future demand, based on basic overlap estimates and MC overlap estimates, respectively. In Table 41 and Table 42 we do the same, but for FCFS approaches that incorporate future demand. For the results, the safety factors with the best outcomes were used for each combination of standard deviations and for each solution approach. Cells with positive values (in blue) mean that the SA approach performs better, and negative values (red cells) mean that the FCFS approach performs better.

Table 39: Performance loss when using FCFS instead of SA for basic overlap estimates

		σ Arrival time																	
		0				1				2				3			
		σ On-belt				σ On-belt				σ On-belt				σ On-belt			
		0	2	4	6	0	2	4	6	0	2	4	6	0	2	4	6
σ Transport	0	1139	1022	776	700	653	694	596	645	186	375	398	452	-212	64	-42	61
	2	976	875	843	655	797	760	713	762	598	550	559	535	109	181	393	250
	4	752	622	716	743	726	695	695	828	512	563	466	463	241	367	180	219
	6	489	504	569	375	466	543	404	644	410	441	508	344	73	250	260	292

Table 40: Performance loss when using FCFS instead of SA for Monte Carlo overlap estimates

		σ Arrival time																	
		0				1				2				3			
		σ On-belt				σ On-belt				σ On-belt				σ On-belt			
		0	2	4	6	0	2	4	6	0	2	4	6	0	2	4	6
σ Transport	0	1023	829	786	905	557	661	508	561	37	269	203	317	-343	-35	-168	-159
	2	974	779	676	647	817	717	685	501	648	484	366	509	249	141	181	118
	4	785	887	712	793	739	774	794	644	597	621	624	431	443	294	377	349
	6	789	628	643	564	590	617	683	576	485	426	725	401	301	388	415	447

Unless the arrival time has a standard deviation of more than two minutes, the SA approach outperforms the standard FCFS approach, both based on basic overlap estimates and FCFS overlap estimates. When the standard deviation of the arrival time is larger than two minutes, it still only makes sense to the regular FCFS over the SA approach if the standard deviation in the transportation time estimates is low. For the FCFS approach that incorporates future demands, we see similar behavior as for the regular FCFS approach. Only, the performance loss when using MC overlap estimates is less than when using basic overlap estimates.

Table 41: Performance loss when using FCFS + future demand instead of SA for basic overlap estimates

		σ Arrival time																	
		0				1				2				3					
		σ On-belt				σ On-belt				σ On-belt				σ On-belt					
		0	2	4	6	0	2	4	6	0	2	4	6	0	2	4	6		
σ Transport	0	661	684	662	595	423	495	489	432	157	356	464	343	-81	7	67	-60		
	2	697	683	603	435	560	599	528	575	500	500	328	465	154	231	433	416		
	4	511	397	605	555	564	606	679	684	582	463	448	255	316	326	196	260		
	6	304	494	334	266	356	449	304	460	410	380	327	383	215	326	132	412		

Table 42: Performance loss when using FCFS+ future demand instead of SA for Monte Carlo overlap estimates

		σ Arrival time																	
		0				1				2				3					
		σ On-belt				σ On-belt				σ On-belt				σ On-belt					
		0	2	4	6	0	2	4	6	0	2	4	6	0	2	4	6		
σ Transport	0	822	642	621	673	331	519	457	384	-176	136	104	73	-548	-159	-87	-130		
	2	674	460	519	421	481	457	402	425	340	201	303	294	-26	-3	-37	-139		
	4	516	541	395	487	487	464	473	433	376	348	212	302	176	75	160	24		
	6	464	273	382	205	349	262	320	244	291	346	367	369	136	97	157	120		

6.7 – Conclusions

In this chapter we have shown the best solution approaches for all combinations of standard deviations in the estimates. FCFS approaches turned out to be preferable in case the arrival time estimates had a standard deviation of 3 minutes and the transportation time estimate a standard deviation of 2 minutes or smaller. In cases with a lower arrival time estimate standard deviation, either the MC overlap approach or the SA approach with basic overlap estimates are preferred. In cases the transportation time estimate standard deviation was 4 or larger, the SA approach with MC overlap estimates always showed superior performance.

Section 6.3 showed that uncertainty in the transportation time estimates has the largest impact on all solution approaches per unit of standard deviation. For non-FCFS approaches this is followed by the arrival time uncertainty and then the on-belt duration uncertainty. For FCFS, only the FCFS

incorporating future demand with basic overlap estimates, the arrival time uncertainty has a significant impact on the performance, but with a much lower coefficient.

In Section 6.4, we explained the trade-off between using basic overlap estimates and MC overlap estimates. For the basic overlap estimates, it is much easier to visualize the problem and the quality of the solution approach. However, this quality is generally lower than that of solution approaches using MC overlap estimates. For experiments with the standard deviation of the transportation time estimate of 4 or larger, this difference quickly becomes larger.

Based on the SA results, we found that solution methods using basic overlap estimates, require at least a safety factor of 4 minutes and in case of much stochasticity even 6 minutes. Not using a safety factor has a large negative effect on the performance. For approaches using MC estimates, a safety factor of around 2 generally gives good results. Not using a safety factor has not a major impact on performance.

The FCFS approaches only increase performance if the arrival time estimator has a high level of uncertainty, while the other estimators have low uncertainties.

Chapter 7: Conclusions and recommendations

The goal of this research was formulated as:

> *'Enhance the performance of ORTEC's baggage belt assignment algorithm for current, and improved time estimates, by developing a heuristic that outperforms the current assignment heuristic'*

To reach this goal, we formulated six research questions in Section 1.3, that were answered throughout the report. In Chapter 2, we explored the current flight characteristics and the solution approach, and we determined the performance of this solution approach. In Chapter 3, we conducted a literature research on related airport problems, related classical optimization problems and ways to incorporate stochasticity in the solution approaches. In Chapter 4 the set of implemented solution approaches was introduced, and their performance based on historical data was shown. In Chapter 5, an experimental design was introduced to determine the performance of the algorithms on improved estimates, of which the results were presented in Chapter 6. The conclusions per chapter are stated at the end of each of the chapters.

This chapter concludes the research, with in Section 7.1 our main conclusions, in Section 7.2 our recommendations, in Section 7.3 a discussion on the limitations of the research, and in Section 7.4 suggestions for further research.

7.1 – Conclusions

In Chapter 2, we have explored the data that the assignment tool uses to make decisions, and found that there was often a large difference between the expected time durations and realized time durations of processes. Not only was there some degree of uncertainty in the estimates, but transportation times were structurally estimated too high or too low.

Next, we explained the current solution approach and formulated the optimization objective of this solution approach. If the baggage of two flights (a flight pair) is present on the baggage retrieval belts at the same time, we say the flight pair has overlapping on-belt time windows. It is undesired to have these overlapping flight pairs assigned to the same baggage retrieval belt. In two out of the three factors of the optimization objective, the assignment of overlapping flights to the same baggage belt is penalized in some way. Because of the uncertainties in the estimates, 40% of flight pairs that ended up overlapping were not estimated to overlap, and out of all flights that were expected to overlap, only 47% did end up overlapping.

Thereafter, we introduced the difference between the 'expected performance' and the 'realized performance' of an assignment. The expected performance of an assignment is the objective function score of the assignment, based on the expected time durations. The realized performance is the objective function of an assignment, based on the realized time durations. This realized performance can therefore only be determined in hindsight, and if estimates were to be perfect, the expected and the realized performance would be the same.

In Chapter 2, we also formulated a mathematical model that was able to solve problem instances to optimality. Comparing the expected performance of the SA approach with the best possible expected performance, we found that the SA approach is able to solve the problem instances very well. The realized performance of optimization based on the current time estimates, in which a lower score is better, is around -6,800, both for the exact solutions found using the mathematical model and for the SA approach. The performance of the optimization based on the realized times is nearly -11,400. This is a large difference and thus we concluded that the current solution approach, using the airport's current estimates, creates schedules of poor quality. Therefore, a way to handle uncertainty in the time estimates had to be found.

In Chapter 3 a literature research was conducted on related airport problems, related traditional optimization problems, and ways to incorporate stochasticity in combinatorial optimization problems. We found that the only research that has been published about solution approaches for the belt allocation problem for inbound baggage handling, is structured in a very different way.

In terms of classical problems, our problem can be formulated as a maximization variant of the Process Allocation Problem (or the Generalized Quadratic Assignment Problem, of which the PAP is a special case) with side constraints. Current literature only focusses on the minimization variant of this problem, which is generally solved by clustering approaches. Clustering is however not suitable for our maximization variant.

From the literature research, two promising ways of dealing with stochasticity were found. For the first method, a so-called safety time was added to the estimated on-belt time duration. This makes the expected time window that a flight's baggage will be present on the baggage retrieval belt longer, and therefore more flight pairs were considered to be overlapping. For the second method, the way the expected overlap between a flight pair is calculated, was changed.

Originally, the expected overlap duration between two flights was calculated by determining the overlap between the time windows that the two flights were expected to be on the baggage retrieval belt. In the new expected overlap determination method, distributions and their parameters are determined for each of the estimators. Then, using Monte Carlo (MC) simulation, the overlap between all combinations of flights are simulated a large number of times. For each combination of flights, we average the resulting

overlap in all MC simulations, and use this as the expected overlap between the flight pair. This new method to calculate the expected overlap between flight pairs, is referred to as 'MC overlap estimation'. The old method, in which the overlap of the expected times are used, is referred to as 'basic overlap estimation'.

In Chapter 4, three new solution approaches were introduced. First, a First Come First Served (FCFS) approach. In the FCFS approach, instead of using rolling horizon optimization, flights are assigned to a baggage belt upon the arrival of the flight. This way, uncertainty in arrival time is eliminated, and the time at which a flight is present on the baggage retrieval belt becomes more accurate. The assignment is based on the expected occupancies of all belts, that follow from recently arrived other flights. Second, a Greedy assignment method was introduced, that optimized the sub-problems from the rolling horizon optimization, by iteratively determining the unassigned flight with the highest regret factor and assigning to its best-scoring belts. Finally, the FCFS and the Greedy approach were combined to create a FCFS approach that incorporates future demand (FCFS+). Upon the arrival of a flight, instead of only looking at previously arrived flights, a selection of near-future flights is used to create a sub-problem. For each candidate belt of the arriving flight, the sub-problem is optimized in the Greedy manner, with the condition that the flight is placed on the candidate belt (so there is a schedule in which the arriving flight is on belt A, a schedule in which the arriving flight is on belt B, etcetera). The performance of all schedules are compared and the flight is assigned to the belt with the best resulting schedule.

Based on historical data, the FCFS approaches gave superior results to that of the current SA approach, both for the estimates used by the airport and an unbiased version of these estimates. Adding safety factors increased performance on the unbiased historical data by a comparable amount. Combining the FCFS approach and the safety factors however did not increase performance further than just using one of the improvements.

The implementation of MC overlap estimates also increased the performance of the SA algorithm on historical data by an equal amount to that of just adding safety times, or just using the FCFS approach. Using a combination of MC overlap estimates and adding safety times, the performance increase becomes larger. Combining the MC overlap estimates with the FCFS did not further increase performance, nor did a combination of all three approaches.

The Greedy approach was able to find reasonable solutions fast, but performed structurally worse than the SA approach. The mathematical model (MIP) solver was able to find optimal results for some problem instances, but as more flights would be considered to overlap, the problems quickly became too complicated for the time constricted MIP solver.

From the outcomes in Section 4.4.2 and 4.4.3, it also became clear that it is better to have less estimators for transportation time with unbiased estimates (using only the baggage class), than the currently used estimators (the baggage class and the apron) that give biased estimates.

In Chapter 5, an experimental design was introduced that enabled us to compare the performance of the mentioned solution approaches were compared under improved estimates. This was done because the airport expects that due to the introduction of more suitable estimators in the future, the quality of the estimators will increase.

In Chapter 6 we compared the results of the experimental design, with a focus on the impact of the levels of stochasticity on the solution approaches and on our three methods to mitigate the stochasticity: safety times, MC overlap estimates, and the use of FCFS solution approaches. The results showed that uncertainty in transportation time has the highest impact on the performance of all solution approaches, followed by arrival time uncertainty (only for non-FCFS approaches), and the uncertainty in on-belt duration. For experiments with transportation time uncertainty is 4 minutes or larger, the SA with MC overlap estimates always gave the best results. The FCFS+ approach with MC overlap estimates only outperforms the SA approach with MC overlap estimates when the arrival time uncertainty is high, while the other estimators have low uncertainties.

Solution approaches using basic overlap methods only perform better than, or similar to, the approaches using MC overlap estimates when the uncertainty in transportation time is low. It was found that it is much harder to visualize the MC overlap interactions, than the basic interactions. This makes it hard to show why a solution created using MC overlap estimates would outperform a solution created using basic overlap estimates, even though it generally does. Therefore, a trade-off must be made between the interpretability of a solution and the quality of a solution.

For basic overlap estimates, the safety time should at least be 4 minutes, but in case of high uncertainties, like in the current data, 6 minutes is generally better. Using no safety times has a large negative impact on the performance. For MC overlap estimates, a safety factor of 2 minutes is often sufficient and using a higher safety factor has a negative impact on the results.

7.2 – Recommendations

Although we have not done any experimentation with the rolling horizon optimization, we still found multiple problems with the rolling horizon approach that can easily be resolved. Next, we discuss the recommendations for changes in estimates, the values of safety times, and finally about the implementation of new solution approaches. We will not make recommendations on the use of MC overlap estimates or basic overlap estimates, as the trade-off between the interpretability is a managerial issue. We did however provide some insights on the matter in Section 6.4.

Rolling horizon optimization

1) Reduce the duration between data updates in the optimization. If instead of retrieving flight data every ten minutes, we do it every two minutes, the arrival time estimates should be of higher quality. In Section 2.2.4 we saw SA was able to find optimum results for a day of data in one minute of run time, so for the smaller problem instances this one minute should also be enough for the optimization part of the optimization round.
2) Reduce the planning horizon. There is too much uncertainty in arrival times to let a flight's belt assignment depend on a flight that is expected to arrive two hours later. Instead of creating a planning three hours in advance, for instance one hour should be good enough.

Changes in estimates

3) Improve the arrival time estimates in the system that provides our planning tool with the arrival time estimates, because somewhere between 40 minutes and 30 minutes prior to arrival, the quality of the arrival time estimate of a flight structurally worsens: The bias in the estimates jumps from -0.12 to +2.56 minutes. Presumably, this is due to the use of some different estimation method in the last phase of the flight.
4) Simplify the estimators for the transportation time to just using the flight's baggage class, and keep track the realized transport times over time, if the difference between the estimates and the realized times become too large, the estimates should be changed.

Basic overlap estimates versus Monte Carlo overlap estimates:

5) In case the airport wants to keep basic overlap estimates in order to be able to visualize the problem and the expected quality of its solutions in an easier way, the safety time of the on-belt time windows should be increased to 4-6 minutes in the current solution approach.
6) In case the airport wants to switch to MC overlap estimation and no better estimators become available, the SA approach remains superior to the other approaches. The safety factor should then be around 2 minutes.

Implementation of a new solution approach:

7) As our experimentation did not include the rolling horizon approach and it will take a lot of time to simulate the rolling horizon approach, we recommend to test the performance of alternative solution approaches by running them parallel to the current solution approach, without using the results of the new solution approach before a comparison between the performance has shown that the new approach performs better.

7.3 – Limitations

All findings in this research are based on the dataset that was used to calculate performance. We have compared the characteristics of the used week with the characteristics of another week of data, as shown in Appendix G, and the flight characteristics in this week were very similar. However, we could not compare the flight characteristics with a week within a peak season, as this data was not available. Therefore we made the recommendation to monitor the realized times of the estimators and adjust the estimates accordingly.

Second, we have not included the rolling horizon approach in the experimentation, as including these would result in very long runtimes of the simulation. The use of an SA approach implies that we need to use such rolling horizon approach. In Section 4.4.2, we saw that the performance difference between solving one problem instance using SA and the rolling horizon approach was small over one week of data, however there were some days for which this difference was relatively large. Therefore, we made the recommendation to test a new approach in the rolling horizon environment before implementing it.

We have not optimized the safety times. In Section 6.5 we have shown that these safety times have a larger impact on the basic overlap estimates than on the MC overlap estimates. Therefore, the solution approaches using basic overlap estimation methods are likely to benefit more from optimized safety times than the MC overlap solution approaches. Our comparison between the basic overlap estimation methods and the MC overlap estimation methods, without fully optimized safety times (we only assessed four safety time factors), might therefore not always show the difference between the best possible basic overlap estimation approach and MC overlap estimation approach. Optimizing these safety times would therefore be interesting in future research.

7.4 – Further research

As just mentioned, the safety times have not been optimized and only static safety time policies have been assessed. In future research, the usage of dynamic safety times could be explored to further increase performance**.** In Chapter 2, we saw for instance that for baggage class A, the standard deviation of the transportation time estimation was much lower than the standard deviation of the transportation time estimates for baggage class B flights. Performance could improve if baggage class B flights would then get a higher safety time factor than baggage class A flights. Especially for the basic overlap estimates these dynamic safety times might prove to be helpful, as for the MC overlap estimates the standard deviations of different estimators are explicitly included in the overlap estimation process.

If the usage of dynamic safety times is analyzed, an analysis of where to put these safety times is also needed. With large transportation uncertainty, flights may arrive at the belt much later than expected,

but also much earlier. In the current research it did not matter where the safety factor was added as all flights received the same safety factor.

Besides further optimizing the basic overlap estimates, we can also improve the MC overlap approach, by determining the optimal cut-off point in MC overlap estimation for flights to be considered overlapping. Currently the only flight combinations that have an expected overlap of at least 0.5 minutes are considered to overlap, but this threshold was not optimized. It might also be interesting to experiment with the outcome histogram of the MC overlap estimates. Instead of using the average of the Monte-Carlo overlap estimates, we could for instance we use the 75^{th} percentile of overlap estimates.

Furthermore, it might be interesting to experiment with the measures that prevent flights that are about to arrive, to change belt assignments. Currently, besides yet arrive flights, flights that are expected to arrive during the optimization run of the following round (so the first five minutes of the next optimization round) are disallowed to change belt assignments. In Section 2.1.6 we saw that currently 2.2% of flights that arrive during optimization round *n* are still assigned to another belt by the SA algorithm in this optimization round *n*, while in practice they will actually be assigned to the belt it was assigned to in optimization round *n*-1. Therefore, flights that have not yet arrived are assigned to a belt using false information of the flight's assignment during optimization round *n*. This is due to some flights not being expected within the first five minutes of the following optimization round, while in fact they arrive earlier than expected. As we did not experiment with rolling horizon optimization, we also did not assess the impact of changing this parameter.

Finally, further research on the relation between one-problem instance optimization and the rolling horizon optimization would be interesting, but before analyzing this relationship, it would be advised to first incorporate the recommendations about increasing the frequency of rolling horizon estimation, reducing the time horizon, preventing flights to change assignments during the optimization round in which they might arrive and remove the bias introduced by (presumably) changing arrival time estimation methods.

References

Alkhamis, T. M., Ahmed, M. A., & Tuan, V. K. (1999). Simulated annealing for discrete optimization with estimation. European Journal of Operational Research, 116(3), 530–544. https://doi.org/10.1016/s0377-2217(98)00112-x

Babić, O., Teodorović, D., & Tošić, V. (1984). Aircraft Stand Assignment to Minimize Walking. Journal of Transportation Engineering, 110(1), 55–66. https://doi.org/10.1061/(asce)0733-947x(1984)110:1(55)

Baker, K. & Trietsch, D. (2009). Principles of sequencing and scheduling. Hoboken, N.J: John Wiley.

Barnabani, M. (2015). An approximation to the convolution of gamma distributions. Communications in Statistics - Simulation and Computation, 46(1), 331–343. https://doi.org/10.1080/03610918.2014.963612

Beham, A., Wagner, S., & Affenzeller, M. (2018). Algorithm selection on generalized quadratic assignment problem landscapes. Proceedings of the Genetic and Evolutionary Computation Conference. GECCO '18: Genetic and Evolutionary Computation Conference. https://doi.org/10.1145/3205455.3205585

Bianchi, L., Dorigo, M., Gambardella, L. M., & Gutjahr, W. J. (2008). A survey on metaheuristics for stochastic combinatorial optimization. Natural Computing, 8(2), 239–287. https://doi.org/10.1007/s11047-008-9098-4

Bokhari, S. H. (1981). A Shortest Tree Algorithm for Optimal Assignments Across Space and Time in a Distributed Processor System. IEEE Transactions on Software Engineering, SE-7(6), 583–589. https://doi.org/10.1109/tse.1981.226469

Bolat, A. (2001). Models and a genetic algorithm for static aircraft-gate assignment problem. Journal of the Operational Research Society, 52(10), 1107–1120. https://doi.org/10.1057/palgrave.jors.2601190

Burkard, R., Amico, M. & Martello, S. (2009). Assignment problems. Philadelphia, Pa: Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104.

Chen, Y., Hao, J.-K., & Glover, F. (2016). An evolutionary path relinking approach for the quadratic multiple knapsack problem. Knowledge-Based Systems, 92, 23–34. https://doi.org/10.1016/j.knosys.2015.10.004

Chu, Holloway, Min-Tsung Lan, & Efe. (1980). Task Allocation in Distributed Data Processing. Computer, 13(11), 57–69. https://doi.org/10.1109/mc.1980.1653419

Chu, W. W. (1969). Optimal File Allocation in a Multiple Computer System. IEEE Transactions on Computers, C-18(10), 885–889. https://doi.org/10.1109/t-c.1969.222542

Cordeau, J.-F., Gaudioso, M., Laporte, G., & Moccia, L. (2006). A Memetic Heuristic for the Generalized Quadratic Assignment Problem. INFORMS Journal on Computing, 18(4), 433–443. https://doi.org/10.1287/ijoc.1040.0128

D.F. Votaw, A. Orden, The personnel assignment problem, Symposium on Linear Inequalities and Programmng, SCOOP 10, US Air Force, 1952, pp. 155–163.

Dahlhaus, E., Johnson, D. S., Papadimitriou, C. H., Seymour, P. D., & Yannakakis, M. (1992). The complexity of multiway cuts (extended abstract). Proceedings of the Twenty-Fourth Annual ACM

Symposium on Theory of Computing - STOC ’92. the twenty-fourth annual ACM symposium. https://doi.org/10.1145/129712.129736

Deng, W., Zhao, H., Yang, X., Li, D., Li, Y., & Liu, J. (2016). Research on a robust multi-objective optimization model of gate assignment for hub airport. Transportation Letters, 10(4), 229–241. https://doi.org/10.1080/19427867.2016.1252876

Easterfield, T. E. (1946). A Combinatorial Algorithm. Journal of the London Mathematical Society, s1-21(3), 219–226. https://doi.org/10.1112/jlms/s1-21.3.219

Edmonds, J., & Karp, R. M. (1972). Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems. Journal of the ACM (JACM), 19(2), 248–264. https://doi.org/10.1145/321694.321699

Fisher, M. L., Jaikumar, R., & Van Wassenhove, L. N. (1986). A Multiplier Adjustment Method for the Generalized Assignment Problem. Management Science, 32(9), 1095–1103. https://doi.org/10.1287/mnsc.32.9.1095

Frey, M., Kiermaier, F., & Kolisch, R. (2017). Optimizing Inbound Baggage Handling at Airports. Transportation Science, 51(4), 1210–1225. https://doi.org/10.1287/trsc.2016.0702

Gallo, G., Hammer, P. L., & Simeone, B. (1980). Quadratic knapsack problems. In Mathematical Programming Studies (pp. 132–149). Springer Berlin Heidelberg. https://doi.org/10.1007/bfb0120892

Gruler, A., Araújo, C. L. Q., Calvet, L., & Juan, A. A. (2017). Waste collection under uncertainty: a simheuristic based on variable neighbourhood search. European J. of Industrial Engineering, 11(2), 228. https://doi.org/10.1504/ejie.2017.083257

Gunawan, A., Ng, K. M., Poh, K. L., & Lau, H. C. (2014). Hybrid metaheuristics for solving the quadratic assignment problem and the generalized quadratic assignment problem. 2014 IEEE International Conference on Automation Science and Engineering (CASE). 2014 IEEE International Conference on Automation Science and Engineering (CASE). https://doi.org/10.1109/coase.2014.6899314

Haghani, Ali & Chen, Min-Ching, 1998, "Optimizing gate assignments at airport terminals," Transportation Research Part A: Policy and Practice, Elsevier, vol. 32(6), pages 437-454, August.

Hahn, P. M., Kim, B.-J., Guignard, M., Smith, J. M., & Zhu, Y.-R. (2007). An algorithm for the generalized quadratic assignment problem. Computational Optimization and Applications, 40(3), 351–372. https://doi.org/10.1007/s10589-007-9093-1

Heerkens, H., Winden, A. & Tjooitink. (2017). Solving Managerial Problems Systematically. Groningen. Noordhoff Uitgevers Noordhoff Uitgevers BV.

Hiley, A., & Julstrom, B. A. (2006). The quadratic multiple knapsack problem and three heuristic approaches to it. Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation - GECCO ’06. the 8th annual conference. https://doi.org/10.1145/1143997.1144096

Homem-De-Mello, T. (2003). Variable-sample methods for stochastic optimization. ACM Transactions on Modeling and Computer Simulation, 13(2), 108–133. https://doi.org/10.1145/858481.858483

Hung, M. S., & Fisk, J. C. (1978). An algorithm for 0-1 multiple-knapsack problems. Naval Research Logistics Quarterly, 25(3), 571–579. https://doi.org/10.1002/nav.3800250316

Juan, A. A., Barrios, B. B., Vallada, E., Riera, D., & Jorba, J. (2014). A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times. Simulation Modelling Practice and Theory, 46, 101–117. https://doi.org/10.1016/j.simpat.2014.02.005

Juan, A. A., Faulin, J., Grasman, S. E., Rabe, M., & Figueira, G. (2015). A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. Operations Research Perspectives, 2, 62–72. https://doi.org/10.1016/j.orp.2015.03.001

Kim, S. H., & Feron, E. (2011). Robust Gate Assignment. AIAA Guidance, Navigation, and Control Conference. Presented at the AIAA Guidance, Navigation, and Control Conference. https://doi.org/10.2514/6.2011-6382

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. Naval Research Logistics Quarterly, 2(1–2), 83–97. https://doi.org/10.1002/nav.3800020109

Lambrecht, M. ., Luyten, R., & Vander Eecken, J. (1985). Protective inventories and bottlenecks in production systems. European Journal of Operational Research, 22(3), 319–328. https://doi.org/10.1016/0377-2217(85)90251-6

Lee, C.-G., and Z. Ma (2004). The generalized quadratic assignment problem, Research Report, Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Ontario M5S 3G8, Canada.

Lim, W. L., Alias, M. A. S., & Haron, H. (2015). A hybrid metaheuristic for the generalized quadratic assignment problem. 2015 IEEE Student Conference on Research and Development (SCOReD). 2015 IEEE Student Conference on Research and Development (SCOReD). https://doi.org/10.1109/scored.2015.7449380

Magirou, V. F., & Milis, J. Z. (1989). An algorithm for the multiprocessor assignment problem. Operations Research Letters, 8(6), 351–356. https://doi.org/10.1016/0167-6377(89)90022-9

Mangoubi, R. S., & Mathaisel, D. F. X. (1985). Optimizing Gate Assignments at Airport Terminals. Transportation Science, 19(2), 173–188. https://doi.org/10.1287/trsc.19.2.173

Martello, S. & Toth, P. (1990). Knapsack problems : algorithms and computer implementations. Chichester New York: J. Wiley & Sons.

Mateus, G. R., Resende, M. G. C., & Silva, R. M. A. (2010). GRASP with path-relinking for the generalized quadratic assignment problem. Journal of Heuristics, 17(5), 527–565. https://doi.org/10.1007/s10732-010-9144-0

McKendall, A., & Li, C. (2016). A tabu search heuristic for a generalized quadratic assignment problem. Journal of Industrial and Production Engineering, 34(3), 221–231. https://doi.org/10.1080/21681015.2016.1253620

Morán-Mirabal, L. F., González-Velarde, J. L., Resende, M. G. C., & Silva, R. M. A. (2013). Randomized heuristics for handover minimization in mobility networks. Journal of Heuristics, 19(6), 845–880. https://doi.org/10.1007/s10732-013-9223-0

Nadarajah, S., & Kotz, S. (2008). Exact Distribution of the Max/Min of Two Gaussian Random Variables. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 16(2), 210–212. https://doi.org/10.1109/tvlsi.2007.912191

Pagès-Bernaus, A., Ramalhinho, H., Juan, A. A., & Calvet, L. (2017). Designing e-commerce supply chains: a stochastic facility-location approach. International Transactions in Operational Research, 26(2), 507–528. https://doi.org/10.1111/itor.12433

Price, C. C. (1981). The assignment of computational tasks among processors in a distributed system. Proceedings of the May 4-7, 1981, National Computer Conference on - AFIPS '81. Presented at the the May 4-7, 1981, national computer conference. https://doi.org/10.1145/1500412.1500453

Rodriguez-Molins, M., Salido, M. A., & Barber, F. (2014). Robust Scheduling for Berth Allocation and Quay Crane Assignment Problem. Mathematical Problems in Engineering, 2014, 1–17. https://doi.org/10.1155/2014/834927

Ronzani Borille, G. M., & Correia, A. R. (2013). A method for evaluating the level of service arrival components at airports. Journal of Air Transport Management, 27, 5–10. https://doi.org/10.1016/j.jairtraman.2012.10.008

Ross, G. T., & Soland, R. M. (1975). A branch and bound algorithm for the generalized assignment problem. Mathematical Programming, 8(1), 91–103. https://doi.org/10.1007/bf01580430

Saraç, T., & Sipahioglu, A. (2007). A Genetic Algorithm for the Quadratic Multiple Knapsack Problem. In Lecture Notes in Computer Science (pp. 490–498). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-75555-5_47

Silva, R. M. A., Resende, M. G. C., Pardalos, P. M., Mateus, G. R., & De Tomi, G. (2013). GRASP with Path-Relinking for Facility Layout. In Springer Proceedings in Mathematics & Statistics (pp. 175–190). Springer New York. https://doi.org/10.1007/978-1-4614-8588-9_11

Singh, A., & Baghel, A. S. (2007). A New Grouping Genetic Algorithm for the Quadratic Multiple Knapsack Problem. In Evolutionary Computation in Combinatorial Optimization (pp. 210–218). https://doi.org/10.1007/978-3-540-71615-0_19

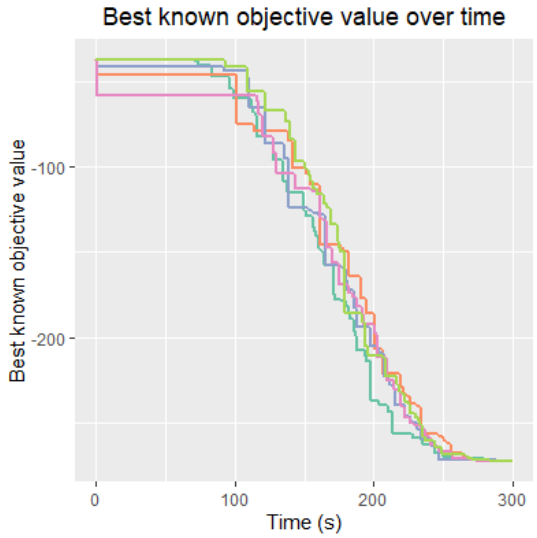
Sofianopoulou, S. (1990). Optimum Allocation of Processes in a Distributed Environment: A Process-to-Process Approach. Journal of the Operational Research Society, 41(4), 329–337. https://doi.org/10.1057/jors.1990.54

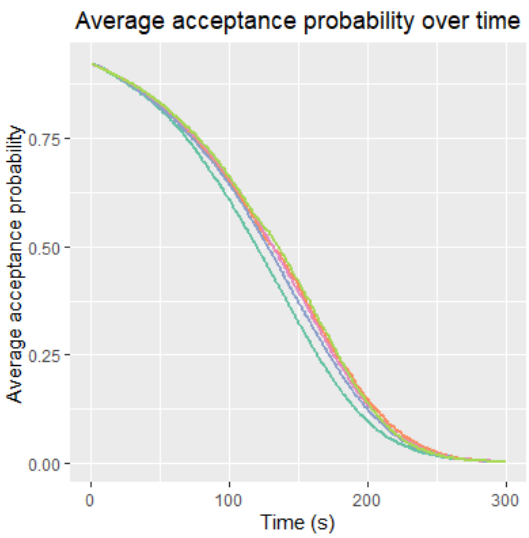
Sofianopoulou, S. (1992). Simulated annealing applied to the process allocation problem. European Journal of Operational Research, 60(3), 327–334. https://doi.org/10.1016/0377-2217(92)90084-m

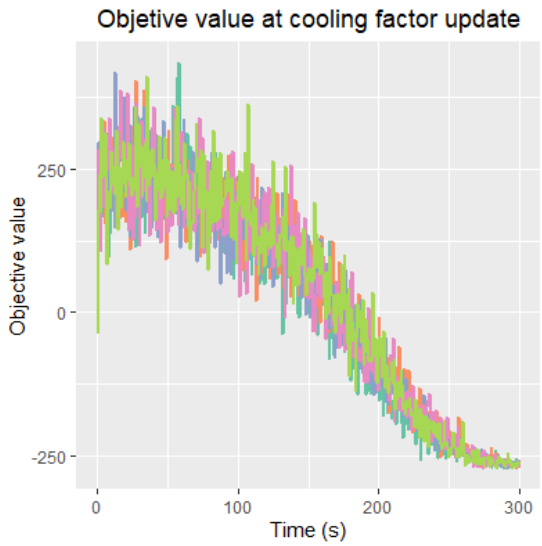
Sundar, S., & Singh, A. (2010). A Swarm Intelligence Approach to the Quadratic Multiple Knapsack Problem. In Neural Information Processing. Theory and Algorithms (pp. 626–633). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-17537-4_76

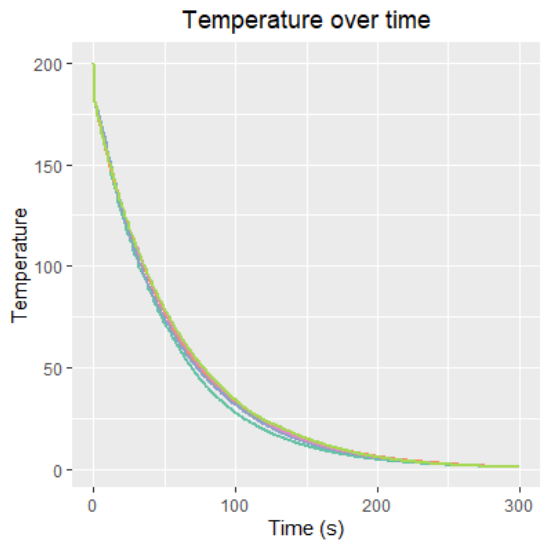
Vigo, D., & Maniezzo, V. (1997). A Genetic/Tabu Thresholding Hybrid Algorithm for the Process Allocation Problem. Journal of Heuristics, 3(2), 91–110. https://doi.org/10.1023/a:1009676913040

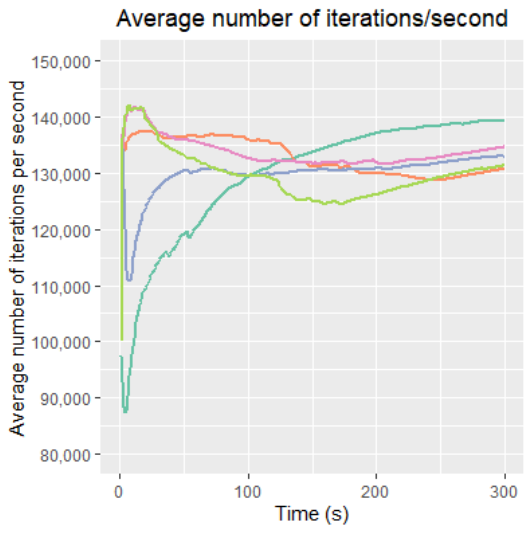
Appendix A: Simulated Annealing

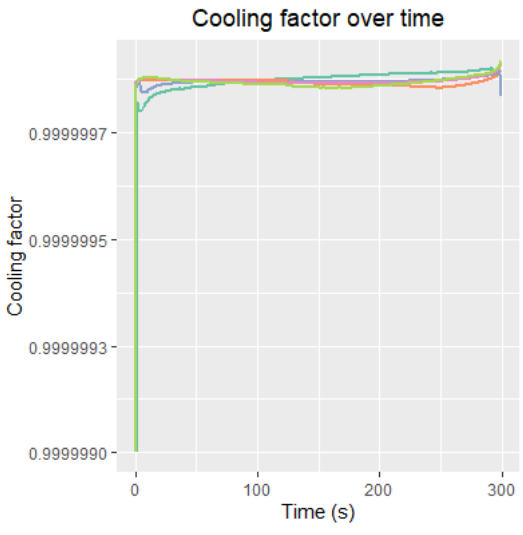








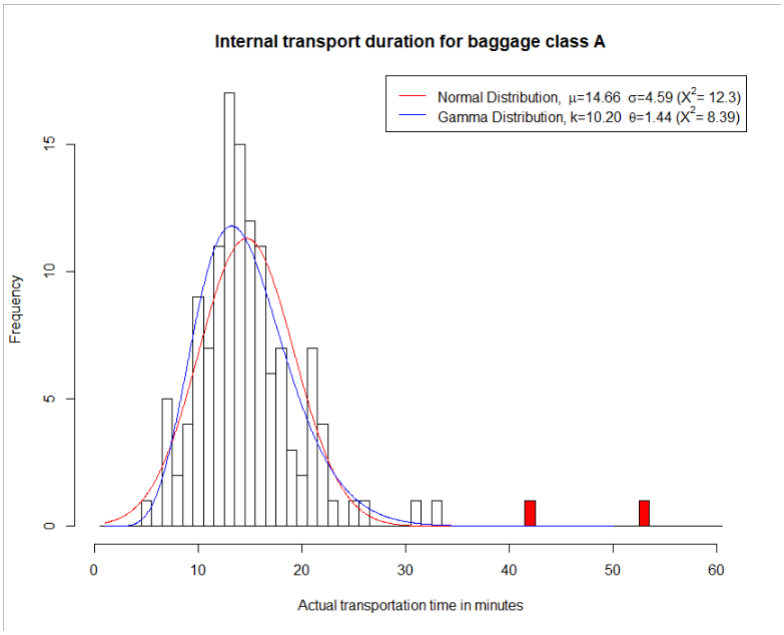




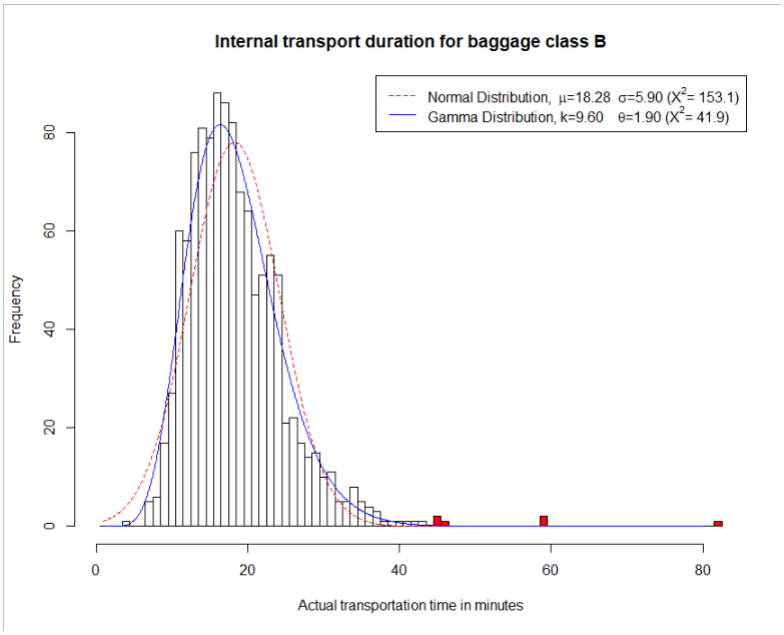
Appendix B: Chi Square tests input data

In the graphs, the bars in red are not incorporated in the parameter estimation of the hypothesized distributions (done using maximum likelihood), nor in the evaluation of the fit of these parameters. A solid line implies the Chi-Square value is below the test statistic and thus there is, at 95% significance, no significant difference between the presumed distribution and the data.

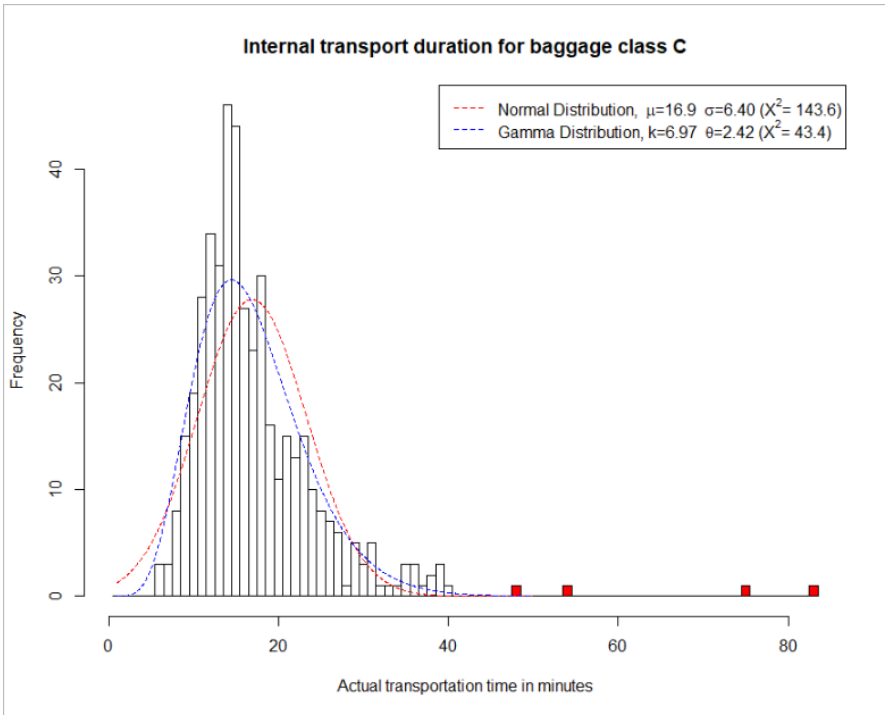
Transportation times for baggage class A:



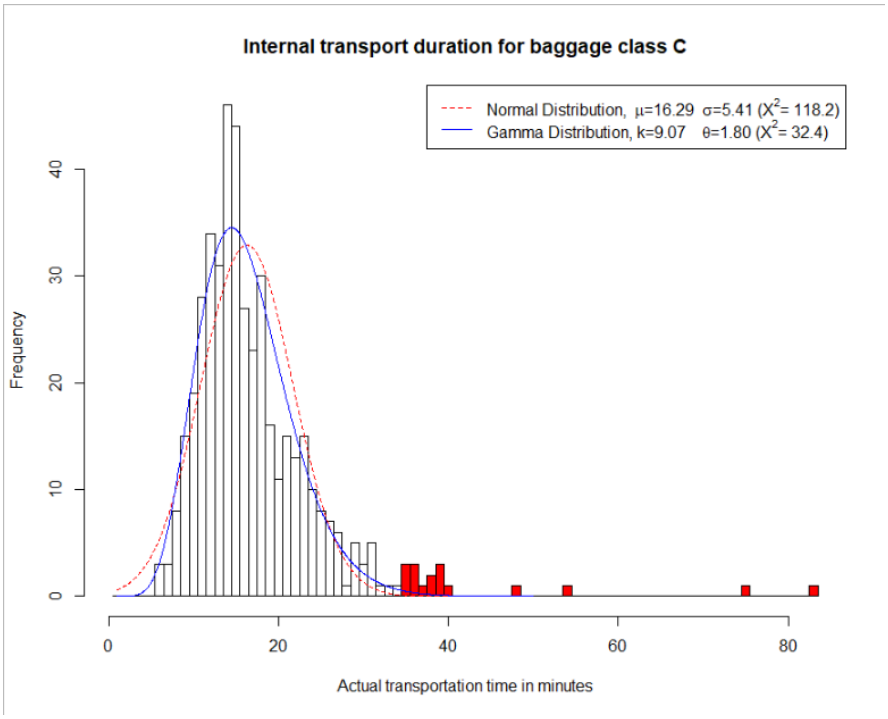
Transportation times for baggage class B:



Transportation times for baggage class C:



Transportation times for baggage class C with additional outliers:



Appendix C: Example of planning under expected and realized times

Figure 36 shows the optimized planning for a morning's worth of flights, based on the airport's time estimates. The objective value of the assignment is -328, a breakdown of the value is given in Table 43.

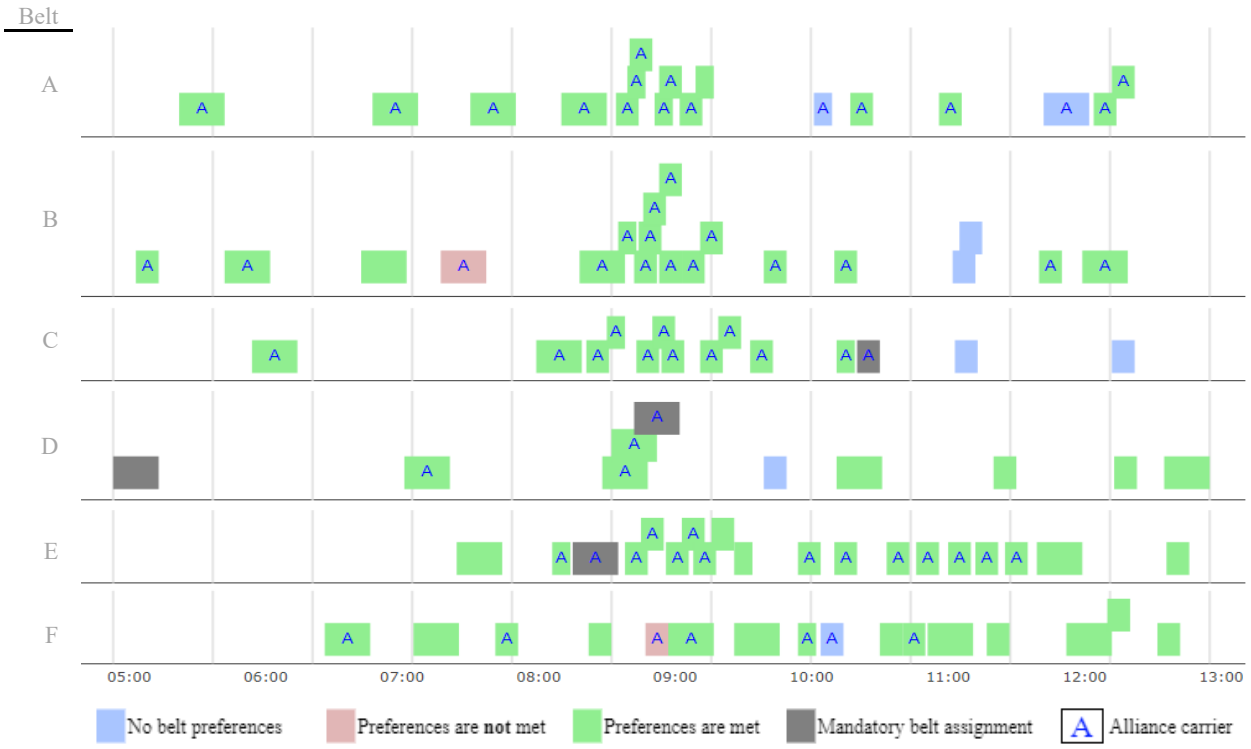


Figure 36: Expected planning with flight assignment based on each flight's start and end time estimation at $t_{arrival}$-1.

Figure 37 shows the same flight-to-belt assignments as Figure 36, but we now show the schedule based on realized arrival times and durations. The objective value for this planning is -232. In Table 43, which shows the breakdown of the objective values for the planning scenario's, we see that the difference between the objective values come from both additional overlapping minutes and from alliance flights that are not arriving at empty belts.

Table 43: Breakdown of objective values for different assignment scenario's

	Expected planning with assignments based on time estimates	Realized planning with assignments based on time estimates	Optimized planning with assignments based on the realized times
Alliance flight starting at an empty belt penalty	180	234	63
Preferred belt bonus	-648	-648	-640
Overlap minutes penalty	140	182	60
Total objective value	-328	-232	-517

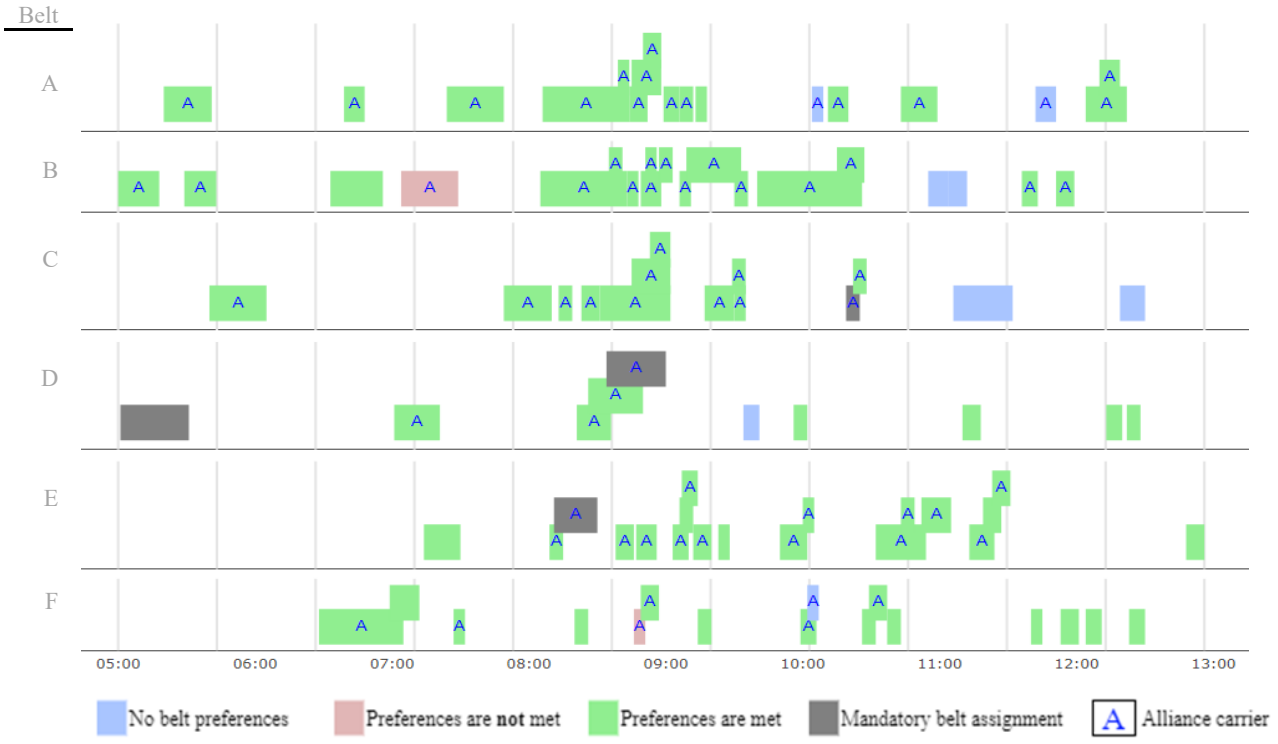


Figure 37: Realized planning with flight assignment based on each flight's start and end time estimation at $t_{arrival}$-1

Figure 38 shows a planning in which the assignment was optimized based on the realized belt occupation times. This planning reflects the situation in which our time estimates are perfect. The objective score for this planning is -517.

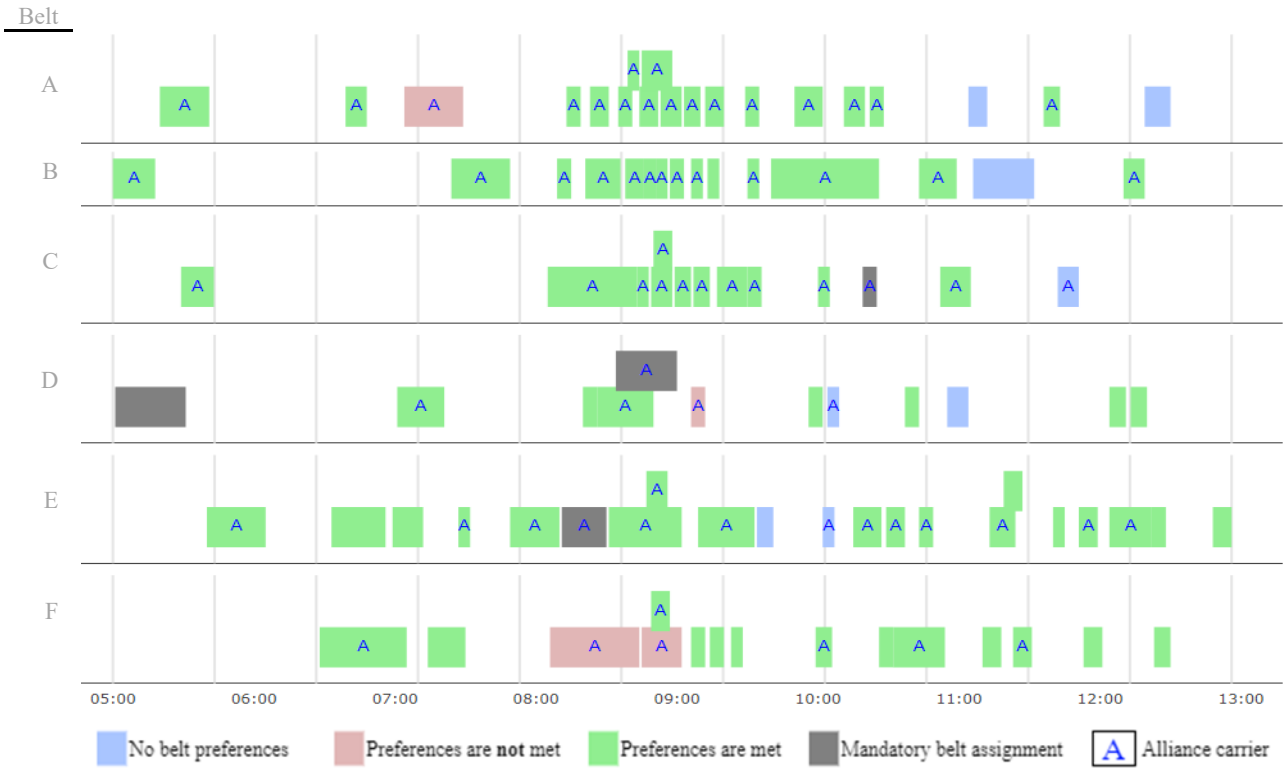


Figure 38: Realized planning with flight assignment based on the realized times.

Appendix D: Closed form expression for the overlap between two flights

When using the expectation of the estimated starting times and durations to calculate the expected overlap, the assumption is made that the expectancy of the overlap between two flights is equal to the overlap of the expected occupancies, which is not the case.

First, we will try to find a closed-form expression for the expected values of the overlap between each flight pair, and the probability that an arbitrary flight arrives on-belt before another arbitrary flight arrives and is still on the belt upon the arrival of the second flight. The latter is needed to determine the probability that a flight arrives at a non-empty belt, given an assignment.

Each flight has a start time and an end time for its time on the baggage retrieval belt, overlap between flight 1 and flight 2 two is defined by:

$$Overlap_{12} = Overlap_{21}$$
$$= \text{Max}(0; \text{Max}(start_1; start_2) - \text{Min}(start_1 + duration_1; start_2 + duration_2))$$

Nadarajah (2008), shows us that we can find the expectation and standard deviation of the maxima and minima of two random variables if those variables a normal distributed. We explore the use of the normal distribution, as the exact distribution of a convolution of independent gamma random variables does not admit a closed form (Barnabani, 2015). This makes it impossible to find a closed form expression on the time a flight leaves the belt, as this time would be a convolution of flight's arrival (a normal distributed value), its transportation time (a gamma distributed value) and its on-belt duration (also a gamma distributed value). A closed-form expression could potentially save a lot of computation time, so using a distribution with a worse fit might be worth it. This should be validated if the use of the normal distribution is continued.

For finding a closed form expression, assuming all times are normal distributed, we start off finding the expectation and standard deviation of the maxima and minima of two normal distributed random variables. Nadarajah (2008), shows us:

$$E[\max(start_1; start_2)] = \mu_{latestStart}$$
$$= \mu_{start1} * \Phi\left(\frac{\mu_{start1} - \mu_{start2}}{\theta}\right) + \mu_{start2} * \Phi\left(\frac{\mu_{start2} - \mu_{start1}}{\theta}\right) + \theta\phi\left(\frac{\mu_{start1} - \mu_{start2}}{\theta}\right)$$

$$E[(\max(start1; start2))^2]$$
$$= (\sigma_{start1}^2 + \mu_{start1}^2) * \Phi\left(\frac{\mu_{start1} - \mu_{start2}}{\theta}\right) + (\sigma_{start2}^2 + \mu_{start2}^2) * \Phi\left(\frac{\mu_{start2} - \mu_{start1}}{\theta}\right) + (\mu_{start1} + \mu_{start2})\theta\phi\left(\frac{\mu_{start1} - \mu_{start2}}{\theta}\right)$$

$$\sigma_{latestStart} = \sqrt{E[(\max(start1; start2))^2] - E[\max(start1; start2)]^2}$$

Where and $\phi(\cdot)$ and $\Phi(\cdot)$ are the probability density function and the cumulative distribution function of the standard normal distribution respectively, and $\theta = \sqrt{\sigma_{start1}^2 + \sigma_{start2}^2}$.,

For the values of the first completion of either of the flights, we find: (note: $end_1 = start_1 + duration_1$)

$$E[\min(end_1; end_2)] = \mu_{earliestEnd}$$
$$= \mu_{end1} * \Phi\left(\frac{\mu_{end2} - \mu_{end1}}{\theta}\right) + \mu_{end2} * \Phi\left(\frac{\mu_{end1} - \mu_{end2}}{\theta}\right) + \theta\phi\left(\frac{\mu_{end2} - \mu_{end1}}{\theta}\right)$$

$$E[(\min(end_1; end_1))^2]$$
$$= (\sigma_{end1}^2 + \mu_{end1}^2) * \Phi\left(\frac{\mu_{end2} - \mu_{end2}}{\theta}\right) + (\sigma_{end2}^2 + \mu_{end2}^2) * \Phi\left(\frac{\mu_{end1} - \mu_{end2}}{\theta}\right) + (\mu_{end1} + \mu_{end2})\theta\phi\left(\frac{\mu_{end2} - \mu_{end1}}{\theta}\right)$$

$$\sigma_{earliestEnd} = \sqrt{E[(\min(end1; end2))^2] - E[\min(end1; end)]^2}$$

With $\theta = \sqrt{\sigma_{end1}^2 + \sigma_{end2}^2}$.

By subtracting the expected latest start by the expected earliest finish of belt occupation, we find a value that tells us on average how much later the earliest end of the two flights is, than the latest start of the two flights. This however is not equal to the expected overlap between both flights, because the expected overlap between two flights is the expected earliest end, minus the expected latest start, given that the latest start is earlier than the earliest end times the probability that the latest start is earlier than the earliest end.

Therefore, we need to factor out the situation in which the latest start is later than the earliest end. We do that by finding the expected value of the normal distribution truncated at 0 as lower bound and determining the probability that the overlap will be less than 0. Multiplying the two values will give us the expected overlap between the two values. We find:

$$\mathrm{E}[EarliestEnd - LatestStart \mid LatestStart < EarliestEnd] * p(LatestStart < EarliestEnd)$$

$$= \left[\mu_{OverlapDuration} + \sigma_{OverlapDuration} * \frac{\phi\left(-\frac{\mu_{OverlapDuration}}{\sigma_{OverlapDuration}}\right)}{1 - \Phi\left(-\frac{\mu_{OverlapDuration}}{\sigma_{OverlapDuration}}\right)}\right] * \left[\Phi\left(\frac{\mu_{OverlapDuration}}{\sigma_{OverlapDuration}}\right)\right]$$

$$= \frac{\mu_{overlapDuration}}{\Phi\left(\frac{\mu_{overlapDuration}}{\sigma_{OverlapDuration}}\right)} + \sigma_{OverlapDuration} * \phi\left(-\frac{\mu_{OverlapDuration}}{\sigma_{OverlapDuration}}\right)$$

The parameters of the distribution to be truncated are:

$\mu_{OverlapDuration} = \mu_{earliestEnd} - \mu_{latestStart}$

$\sigma_{OverlapDuration} = \sqrt{\sigma^2_{latestStart} + \sigma^2_{earliestEnd} - 2\rho\sigma_{latestStart}\sigma_{earliestEnd}}$

$\rho\sigma_{latestStart}\sigma_{earliestEnd} = \mathrm{COV}\left(\max(start_1; start_2); \min(start_1 + duration_1; start_2 + duration_2)\right) =$

$\frac{\mathrm{E}[\max(start1; start2)] * E[\min(start_1 + duration_1; start_2 + duration_2)]}{\mathrm{E}[\max(start1; start2) * \min(start_1 + duration_1; start_2 + duration_2)]}$.

However, we cannot calculate the standard deviation of the distribution, because we cannot calculate the covariance term that is needed to calculate the $\sigma_{OverlapDuration}$. Therefore, a closed-form expression for the expected overlap between two flights is not possible. Not for gamma distributed values, and not for normal distributed values.

Appendix E: Assessing Neighbor solutions

```
CalculateChangeInObjective(NeighborSolution)

1. if(NeighborSolution.Mutation == swap){
2.  return(
     ChangeInPreferredBelt_Swap(NeighborSolution.flight1, NeighborSolution.flight2))
     ChangeInOverlap_Swap(NeighborSolution.flight1, NeighborSolution.flight2)+
     ChangeInAllianceEmptyStart_Swap(NeighborSolution.flight1, NeighborSolution.flight2)+
     ChangeInAllianceEmptyStart_Swap(NeighborSolution.flight2, NeighborSolution.flight1)+
3. }
4. else if(NeighborSolution.Mutation == move){
5.  return(
     ChangeInPreferredBelt_Move(NeighborSolution.flight, NeighborSolution.newBelt)
     ChangeInOverlap_Move(NeighborSolution.flight, NeighborSolution.newBelt)+
     ChangeInAllianceEmptyStart_Move(NeighborSolution.flight, NeighborSolution.newBelt)+
6. }
```

Figure 39: Calculate change in objective function (for both Monte Carlo and for basic overlap estimates)

```
CalculateChangeInObjective(NeighborSolution) // basic overlap estimation
1. if(NeighborSolution.Mutation == swap){
2.  return(
     Figure 43(NeighborSolution.flight1, NeighborSolution.flight2))
     Figure 45(NeighborSolution.flight1, NeighborSolution.flight2)+
     Figure 47(NeighborSolution.flight1, NeighborSolution.flight2)+
     Figure 47(NeighborSolution.flight2, NeighborSolution.flight1)+
3. }
4. else if(NeighborSolution.Mutation == move){
5.  return(
     Figure 42(NeighborSolution.flight, NeighborSolution.newBelt)
     Figure 44(NeighborSolution.flight, NeighborSolution.newBelt)+
     Figure 46(NeighborSolution.flight, NeighborSolution.newBelt)+
6. }
```

Figure 40: Figure overview for neighbor solution evaluation when using basic overlap estimates

```
CalculateChangeInObjective(NeighborSolution) // MC overlap estimation
1. if(NeighborSolution.Mutation == swap){
2.  return(
     Figure 43(NeighborSolution.flight1, NeighborSolution.flight2))
     Figure 45(NeighborSolution.flight1, NeighborSolution.flight2)+
     Figure 49(NeighborSolution.flight1, NeighborSolution.flight2)+
     Figure 49(NeighborSolution.flight2, NeighborSolution.flight1)+
3. }
4. else if(NeighborSolution.Mutation == move){
5.  return(
     Figure 42(NeighborSolution.flight, NeighborSolution.newBelt)
     Figure 44(NeighborSolution.flight, NeighborSolution.newBelt)+
     Figure 48(NeighborSolution.flight, NeighborSolution.newBelt)+
6. }
```

Figure 41: Figure overview for neighbor solution evaluation when using Monte Carlo overlap estimates

```
ChangeInPreferredBelt_Move(flight, newBelt)
1. preferredBeltValue ← 0
2. if(flight.PreferredBelts.Contains(flight.AssignedBelt)){ preferredBeltValue -= 1}
3. if(flight.PreferredBelts.Contains(newBelt)){ preferredBeltValue += 1}
4. return(preferredBeltValue* preferredBeltBonus)
```

Figure 42: Calculate change in preferred belt bonus for the move operator for Monte Carlo and basic overlap estimates

```
ChangeInPreferredBelt_Swap(flight1, flight2)
1. preferredBeltValue ← 0
2. if(flight1.PreferredBelts.Contains(flight1.AssignedBelt)){ preferredBeltValue -= 1}
3. if(flight1.PreferredBelts.Contains(flight2.AssignedBelt)){ preferredBeltValue += 1}
4. if(flight2.PreferredBelts.Contains(flight1.AssignedBelt)){ preferredBeltValue += 1}
5. if(flight2.PreferredBelts.Contains(flight2.AssignedBelt)){ preferredBeltValue -= 1}
6. return(preferredBeltValue * preferredBeltBonus)
`
```

Figure 43: Calculate change in preferred belt bonus for the swap operator for Monte Carlo and basic overlap estimates

```
1.  ChangeInOverlap_Move(flight, newBelt)
2.  oldBelt          ← flight.AssignedBelt
3.  reducedOverlap   ← 0
4.  increasedOverlap ← 0
5.  flightsOnOldBelt   ← remove(oldBelt.AssignedFlights, flight)
6.  flightsOnNewBelt   ← newBelt.AssignedFlights
7.  for every flightOnOldBelt part of flightsOnOldBelt {
8.    if(flightOnOldBelt.OverlapDurationDictionary[flight] > 0){
9.      reducedOverlap += flightOnOldBelt.OverlapDurationDictionary[flight]
10.   }
11. }
12. for every flightOnNewBelt part of flightsOnNewBelt {
13.   if(flightOnNewBelt.OverlapDictionary[flight] > 0){
14.     increasedOverlap += flightOnNewBelt.OverlapDurationDictionary[flight]
15.   }
16. }
17. return((increasedOverlap-ReducedOverlap) * penaltyPerMinuteOverlap)
```

Figure 44: Calculate change in overlap duration for the move operator for Monte Carlo and basic overlap estimates

```
ChangeInOverlap_Swap(flight1, flight2)
1.  belt1            ← flight1.AssignedBelt
2.  belt2            ← flight2.AssignedBelt
3.  reducedOverlap   ← 0
4.  increasedOverlap ← 0
5.  flightsOnBelt1   ← as.list(belt1.AssignedFlights() - flight1) // Exclude flight1
6.  flightsOnBelt2   ← as.list(belt2.AssignedFlights() - flight2) // Exclude flight2
7.  for every flightOnBelt1 part of flightsOnBelt1 {
8.    if(flightOnBelt1.OverlapDurationDictionary[flight1] > 0){
9.      reducedOverlap += flightOnBelt1.OverlapDurationDictionary[flight1]
10.   }
11.   if(flightOnOldBelt1.OverlapDictionary[flight2] > 0){
12.     increasedOverlap += flightOnBelt1.OverlapDurationDictionary[flight2]
13.   }
14. }
15. for every flightOnBelt2 part of flightsOnBelt2 {
16.   if(flightOnBelt2.OverlapDurationDictionary[flight2] > 0){
17.     reducedOverlap += flightOnBelt2.OverlapDurationDictionary[flight2]
18.   }
19.   if(flightOnOldBelt1.OverlapDictionary[flight1] > 0){
20.     increasedOverlap += flightOnBelt2.OverlapDurationDictionary[flight1]
21.   }
22. }
23. return((increasedOverlap-ReducedOverlap) * penaltyPerMinuteOverlap)
```

Figure 45: Calculate change in overlap duration for the swap operator for Monte Carlo and basic overlap estimates

```
ChangeInAllianceEmptyStart_Move(flight, newBelt) // basic overlap estimation
1.  oldBelt          ← flight.AssignedBelt
2.  allianceOverlaps ← 0
3.  for every allianceFlight part of flight.allianceFlightsBlockedByMe{
4.    if(allianceFlight.AssignedBelt == oldBelt){
5.      flightsBlockingTheAllianceFlightAssignedToSameBelt ← new list()
6.      for every blockingFlight part of allianceFlight.FlightsThatBlockMe{
7.        if(blockingFlight.AssignedBelt == oldBelt){
8.          flightsBlockingTheAllianceFlightAssignedToSameBelt.append(blockingFlight)
9.      } }
10.     if(flightsBlockingTheAllianceFlight.length == 1 AND
11.      flightsBlockingTheAllianceFlight[1] == flight){
12.       allianceOverlaps -= 1
13.   } }
14.   if(allianceFlight.AssignedBelt == newBelt){
15.     flightsBlockingTheAllianceFlightAssignedToSameBelt ← new list()
16.     for every blockingFlight part of allianceFlight.FlightsThatBlockMe{
17.       if(blockingFlight.AssignedBelt == newBelt){
18.         flightsBlockingTheAllianceFlightAssignedToSameBelt.append(blockingFlight)
19.     } }
20.     if(flightsBlockingTheAllianceFlight.length == 0{
21.       allianceOverlaps += 1
22. } } }
23. if(flight.isAlliance == FALSE){
24.   return(allianceOverlaps)
25. }
26. overlapOnOldBelt ← FALSE
27. overlapOnNewBelt ← FALSE
28. for every blockingFlight part of flight.FlightsThatBlockMe{
29.   if(!overlapOnOldBelt AND blockingFlight.AssignedBelt == oldBelt){
30.     overlapOnOldBelt == TRUE
31.   }
32.   if(!overlapOnOldBelt AND blockingFlight.AssignedBelt == newBelt){
33.     overlapOnNewBelt == TRUE
34.   }
35.   if(overlapOnOldBelt AND overlapOnNewBelt){
36.     break
37. } }
38. if(overlapOnOldBelt AND !overlapOnNewBelt){
39.   allianceOverlaps -= 1
40. }
39. if(!overlapOnOldBelt AND overlapOnNewBelt){
41.   allianceOverlaps += 1
42. }
43. return(allianceOverlaps * allianceOccupiedBeltPenalty)
```

Figure 46: Calculate change in empty start of alliance flights for the move operator for basic overlap estimates

```
ChangeInAllianceEmptyStart_Swap(flight1, flight2) // basic overlap estimation
1.  oldBelt          ← flight1.AssignedBelt
2.  newBelt          ← flight2.AssignedBelt
3.  allianceOverlaps ← 0
4.  for every allianceFlight part of flight1.allianceFlightsBlockedByMe{
5.    if(allianceFlight.AssignedBelt == oldBelt AND allianceFlight != flight2){
6.      flightsBlockingTheAllianceFlightAssignedToSameBelt ← new list()
7.      for every blockingFlight part of allianceFlight.FlightsThatBlockMe{
8.        if(blockingFlight.AssignedBelt == oldBelt){
9.          flightsBlockingTheAllianceFlightAssignedToSameBelt.append(blockingFlight)
10.     } }
11.     if(flightsBlockingTheAllianceFlight.length == 1 AND
12.      flightsBlockingTheAllianceFlight[1] == flight1){
13.       allianceOverlaps -= 1
14.   } }
15.   if(allianceFlight.AssignedBelt == newBelt){
16.     flightsBlockingTheAllianceFlightAssignedToSameBelt ← new list()
17.     for every blockingFlight part of allianceFlight.FlightsThatBlockMe{
18.       if(blockingFlight.AssignedBelt == newBelt AND blockingFlight != flight2){
19.         flightsBlockingTheAllianceFlightAssignedToSameBelt.append(blockingFlight)
20.     } }
21.     if(flightsBlockingTheAllianceFlight.length == 0{
22.       allianceOverlaps += 1
23. } } }
24. if(flight1.isAlliance == FALSE){
25.   return(allianceOverlaps* allianceOccupiedBeltPenalty)
26. }
27. overlapOnOldBelt ← FALSE
28. overlapOnNewBelt ← FALSE
29. for every blockingFlight part of flight1.FlightsThatBlockMe{
30.   if(blockingFlight != flight2 AND !overlapOnOldBelt AND
         blockingFlight.AssignedBelt == oldBelt)
31.     overlapOnOldBelt == TRUE
32.   }
33.   if(blockingFlight != flight2 AND !overlapOnOldBelt AND
         blockingFlight.AssignedBelt == newBelt)
34.     overlapOnNewBelt == TRUE
35.   }
36.   if(overlapOnOldBelt AND overlapOnNewBelt){
37.     break
38. } }
39. if(overlapOnOldBelt AND !overlapOnNewBelt){
40.   allianceOverlaps -= 1
41. }
42. if(!overlapOnOldBelt AND overlapOnNewBelt){
43.   allianceOverlaps += 1
44. }
45. return(allianceOverlaps * allianceOccupiedBeltPenalty)
```

Figure 47: Calculate change in empty start of alliance flights for the swap operator for basic overlap estimates

```
ChangeInAllianceEmptyStart_Move(flight, newBelt) // MC overlap estimation
1.  oldBelt            ← flight.AssignedBelt
2.  DeltaProbabilities ← 0
3.  for every allianceFlight part of flight.allianceFlightsBlockedByMe{
4.    if(allianceFlight.AssignedBelt == oldBelt){
5.      OldProbabilityOfFlightBlocked ← allianceFlight.ProbabilityOfBeingBlocked
6.     NewProbabilityOfFlightBlocked ← -1
7.     if(OldProbabilityOfBeingBlocked ==
           flight.flightsIBlockWithProbability[allianceFlight]){
8.       flightsBlockingTheAllianceFlightAssignedToSameBelt ← new list()
9.       for every blockingFlight part of allianceFlight.FlightsThatBlockMe{
10.        if(blockingFlight.AssignedBelt == oldBelt){
11.          flightsBlockingTheAllianceFlightAssignedToSameBelt.append(blockingFlight)
12.      } }
13.      NewProbabilityOfFlightBlocked ← 0
14.      if(flightsBlockingTheAllianceFlightAssignedToSameBelt.length > 1){
15.        for every remainingBlockFlight part of
              flightsBlockingTheAllianceFlightAssignedToSameBelt{
16.          if(remainingBlockFlight != flight){
17.            NewProbabilityOfFlightBlocked ← 1 - (1-NewProbabilityOfFlightBlocked) *
                 (1-remainingBlockFlight.allianceFlightsBlockedByMe[allianceFlight])
18.    } } } }
19.    else{
20.    NewProbabilityOfFlightBlocked ← OldProbabilityOfFlightBlocked /
          (1 - flight.allianceFlightsBlockedByMe[AllianceFlight]) + 1 -
          (1 / (1 - flight.allianceFlightsBlockedByMe[AllianceFlight]))
21.    }
22.    DeltaProbabilities += NewProbabilityOfFlightBlocked -
          OldProbabilityOfFlightBlocked
23.  }
24.  if(allianceFlight.AssignedBelt == newBelt)
25.    OldProbabilityOfFlightBlocked ← allianceFlight.ProbabilityOfBeingBlocked
26.    NewProbabilityOfFlightBlocked ← 1 - (1 - OldProbabilityOfFlightBlocked) *
          (1- flight.allianceFlightsBlockedByMe[allianceFlight])
27.    DeltaProbabilities += NewProbabilityOfFlightBlocked -
          OldProbabilityOfFlightBlocked
28. } }
29. if(!flight.IsAlliance){
30.   return(DeltaProbabilities * allianceOccupiedBeltPenalty)
31. }
32. FlightBlockedProbabilityOnOldBelt ← Flight.ProbabilityOfBeingBlocked
33. FlightBlockedProbabilityOnNewBelt ← 0
34. for every blockingFlight part of flight.flightsThatBlockMe{
35.   if(blockingFlight.AssignedBelt == NewBelt){
36.     FlightBlockedProbabilityOnNewBelt ← 1 -
          (1 - FlightBlockedProbabilityOnNewBelt)*
          (1 - Flight.ProbabilityOfBeingBlocked[blockingFlight])
37. } }
38. DeltaProbabilities += FlightBlockedProbabilityOnNewBelt -
      FlightBlockedProbabilityOnOldBelt
39. return(DeltaProbabilities * allianceOccupiedBeltPenalty)
```

Figure 48: Calculate change in empty start of alliance flights for the move operator for Monte Carlo overlap estimates

There are three types of estimates that may change when determining the impact on the probability of flights starting at empty belts. First, we have the flights that were previously blocked (or had a probability of being blocked) by the flight that is being moved, but due to the flight moving, they are no longer (or less) blocked. This impact is assessed in lines 4 through 23. Second, we have the flights on the new belt that will get an increased probability of being blocked. This impact is assessed in lines 24 through 28. Third, we have the impact on the flight that is being moved, which is discussed in lines 32 through 38.

As seen before, the probability of a flight (`FlightX`) being blocked after a new flight that is placed on the same belt (`FlightToAdd`) can be calculated in the following manner:

```
FlightX_Blocked_Probability_New = ( 1 – ( 1 - FlightX_Blocked_Probability_Old ) *
( 1 – FlightToAdd.ProbabilityIBlockFlight[flightX]))
```

But given we take away a flight that blocks `FlightX` with a certain probability, we can use some simple algebra to reverse the effect:

```
FlightX_Blocked_Probability_New = FlightX_Blocked_Probability_Old /
(1 - FlightToRemove.ProbabilityIBlockFlight[flightX]) + 1 – (1 /
(1 - FlightToRemove.ProbabilityIBlockFlight[flightX))
```

Which is used in line 20. However, there is an exception if a flight is taken away that has a blocking probability of 1. For instance, if Flight X is blocked by two flights: Flight A with p=0.4 and Flight B with p=1. Taking away flight B, would the set the new blocking probability to 0 using to the formula. Therefore, we check for this exception starting in line 7.

If the if-statement in line 7 returns true, this means that the alliance flight's probability of being blocked is equal to the probability that flight1 blocks the alliance flight. This means that either flight1 is the only flight that is still blocking the alliance flight, or that flight1's blocking probability of the alliance flight is equal to 1 (and thus removing flight1 we cannot subtract it's entire blocking probability). Therefore, we determine the number of flights that block the alliance flight that are still on the same belt (line 8-12). If this is only 1 (flight1), then after the removal of flight1, no more flights block the alliance flight and therefore the expected probability of the flight being blocked is 0 (line 13). If more possibly overlapping flights remain on the belt (line 14), then we calculate the new probability by looping over all remaining flights (lines 15-18).

In lines 24 through 28, the increase in blocking probabilities for alliance flights on the new belt is assessed.

If the flight itself is not an alliance flight, then it will not be penalized if it arrives at an occupied belt, therefore we are not interested in the increase or decrease of the probability of the flight being blocked, and we can return the values found thus far. If the flight that is being moved to another belt is an alliance

flight, then we retrieve its saved blockage probability and calculate the new probability in lines 34 through 37 and include this impact.

```
ChangeInAllianceEmptyStart_Swap(flight1, flight2) // MC overlap estimation
1.  oldBelt            ← flight1.AssignedBelt
2.  newBelt            ← flight2.AssignedBelt
3.  DeltaProbabilities ← 0
4.  flightsOverlappingWithBothFlights ←
       flight1.OverlappingFlights.Intersect(flight2.OverlappingFlights)
5.  impacteddByBothFlightsAndOnOldBelt ←
       oldBelt.AssignedFlights.Intersect(flightsOverlappingWithBothFlights)
6.  impactedByBothFlightsAndOnNewBelt ←
       newBelt.AssignedFlights.Intersect(flightsOverlappingWithBothFlights)
7.  for every allianceFlight part of flight1.allianceFlightsBlockedByMe{
8.    if(allianceFlight != flight2 AND allianceFlight.AssignedBelt == oldBelt AND
      !impacteddByBothFlightsAndOnNewBelt.contains(allianceFlight)){
9.      OldProbabilityOfFlightBlocked ← allianceFlight.ProbabilityOfBeingBlocked
10.     NewProbabilityOfFlightBlocked ← -1
11.     if(OldProbabilityOfBeingBlocked ==
           flight.flightsIBlockWithProbability[allianceFlight]){
12.       flightsBlockingTheAllianceFlightAssignedToSameBelt ← new list()
13.       for every blockingFlight part of allianceFlight.FlightsThatBlockMe{
14.         if(blockingFlight.AssignedBelt == oldBelt){
15.           flightsBlockingTheAllianceFlightAssignedToSameBelt.append(blockingFlight)
16.       } }
17.       NewProbabilityOfFlightBlocked ← 0
18.       if(flightsBlockingTheAllianceFlightAssignedToSameBelt.length > 1){
19.         for every remainingBlockFlight part of
              flightsBlockingTheAllianceFlightAssignedToSameBelt{
20.           if(remainingBlockFlight != flight1){
21.             NewProbabilityOfFlightBlocked ← 1 - (1-NewProbabilityOfFlightBlocked) *
                  (1-remainingBlockFlight.allianceFlightsBlockedByMe[allianceFlight])
22.     } } } }
23.     else{
24.     NewProbabilityOfFlightBlocked ← OldProbabilityOfFlightBlocked /
          (1 - Flight1.allianceFlightsBlockedByMe[AllianceFlight]) + 1 -
          (1 / (1 - Flight1.allianceFlightsBlockedByMe[AllianceFlight]))
25.     }
26.     if(impactedByBothFlightsAndOnOldBelt.Contains(AllianceFlight)){
27.       NewProbabilityOfFlightBlocked ← 1 - (1 - NewProbabilityOfFlightBlocked) *
             (1 - flight2.allianceFlightsBlockedByMe[AllianceFlight])
28.     }
29.    DeltaProbabilities += NewProbabilityOfFlightBlocked -
          OldProbabilityOfFlightBlocked
30.   }
28.   if(allianceFlight != flight2 AND allianceFlight.AssignedBelt == newBelt)
29.     OldProbabilityOfFlightBlocked ← allianceFlight.ProbabilityOfBeingBlocked
30.     NewProbabilityOfFlightBlocked ← 1 - (1 - OldProbabilityOfFlightBlocked) *
           (1- Flight1.allianceFlightsBlockedByMe[AllianceFlight])
32.     DeltaProbabilities += NewProbabilityOfFlightBlocked -
           OldProbabilityOfFlightBlocked
31. } }
. . .
```

```
. . .
32. if(!flight1.IsAlliance){
33.   return(DeltaProbabilities * allianceOccupiedBeltPenalty)
34. }
35. Flight1BlockedProbabilityOnOldBelt ← Flight1.ProbabilityOfBeingBlocked
36. Flight1BlockedProbabilityOnNewBelt ← 0
37. for every blockingFlight part of flight1.flightsThatBlockMe{
38.   if(blockingFlight != flight2 AND blockingFlight.AssignedBelt == NewBelt){
39.     Flight1BlockedProbabilityOnNewBelt ← 1 -
          (1 - Flight1BlockedProbabilityOnNewBelt)*
          (1 - Flight1.ProbabilityOfBeingBlocked[blockingFlight])
40. } }
41. DeltaProbabilities += Flight1BlockedProbabilityOnNewBelt -
      Flight1BlockedProbabilityOnOldBelt
42. return(DeltaProbabilities * allianceOccupiedBeltPenalty)
```

Figure 49: Calculate change in empty start of alliance flights for the swap operator for Monte Carlo overlap estimates

This method will only assess the change that the first input flight, being flight1, will create. Therefore, the method must be called twice, once with `ChangeInAllianceEmptyStart_Swap(flight1, flight2)` and once `ChangeInAllianceEmptyStart_Swap(flight2, flight1)`. The method behaves a lot like the move operator method, with the belt of `flight2` being the `newBelt`.

There are two exceptions due to which we are not able to use the move mutator twice, instead of creating a new swap mutator. We will explain these exceptions using the same example as in Section 4.3, shown in Figure 50.

First, if flight A and Flight B are on the same belt and we want to assess the impact swapping flight B and flight C, we first determining the impact of moving flight B to the belt of flight C. However, as flight C itself is moved to the belt of flight B, we ignore flight C. The first exception therefore is, is when determining all alliance flights that may be blocked by flight1, flight 2 should always be excluded. This can be seen in line 8, line 28 and line 38.

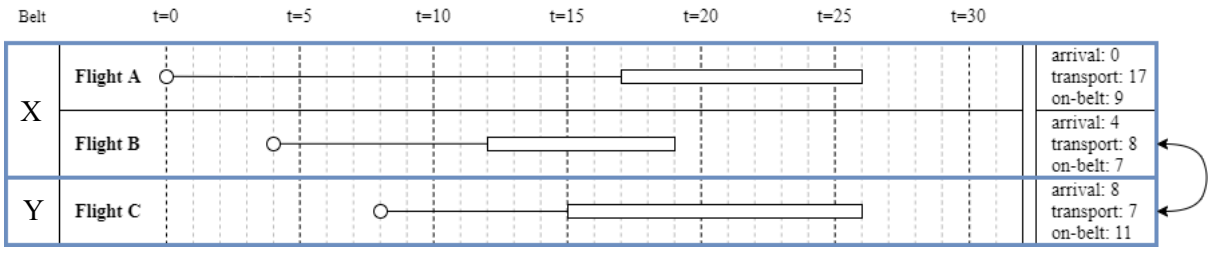


Figure 50: Example of assessing moving flights

The second problem that occurs for which we make an exception, is the change in the probability of flight A being blocked. Flight B has a 0.58 probability of blocking flight A, and flight C a probability of 0.72 (this follows from Table 16). If we would assess the impact of moving flight B to the belt of Flight C, Flight A would have a reduction probability in being blocked of 0.58. If we would assess the

impact of moving flight C to the belt of Flight B, then Flight A's probability would be (1 – (1 – 0.58) * (1 – 0.72) – 0.58 = 0.3024, resulting in a total change of 0.3024-0.58 = -0.2776 instead of 0.72-0.58 = 0.14 because Flight B's impact on Flight A is not excluded in the move statement.

To determine the change in probability correctly we make the following exception: each alliance flight that is impacted by both swapping flights should be processed in either `Swap(flight1, flight2)` or `Swap(flight2, flight1)`. In our method, we therefore exclude all alliance flights that are impacted by both swapping flights and are on the new belt of Flight1.

For our example this would mean that `Swap(FlightB, FlightC)` calculates the change in objective for flight B and for flight A (because flight A is impacted by both swapping flights, and is on the old belt of flightB), and `Swap(FlightC, FlightB)` only calculates the change in objective for flight C.

Appendix F: Number of MC samples

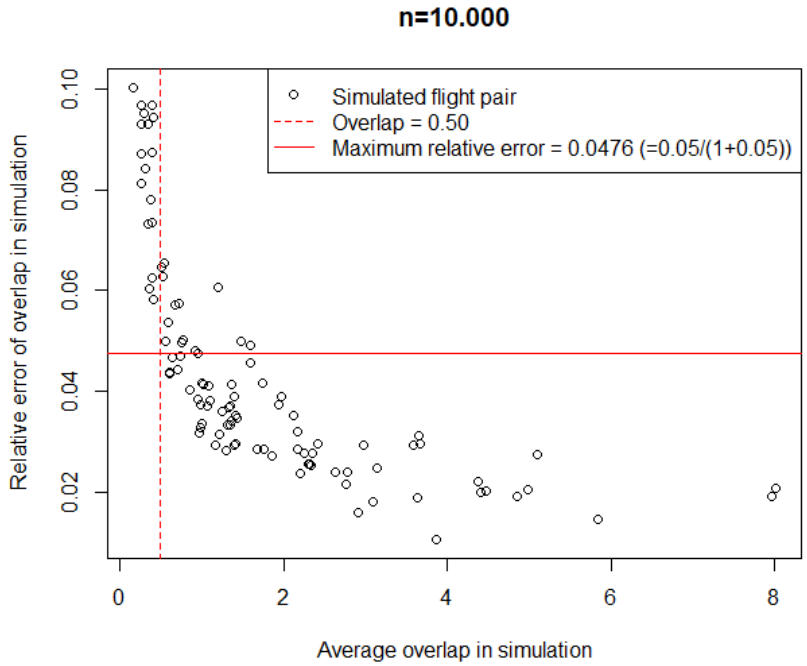


Figure 51:Relative error versus the average estimated overlap for 10.000 MC replications

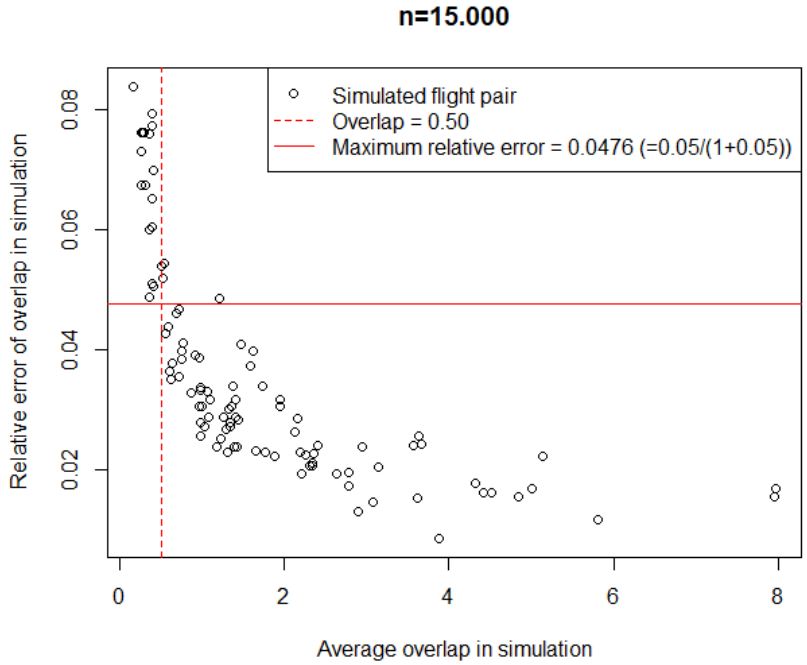


Figure 52:Relative error versus the average estimated overlap for 15.000 MC replications

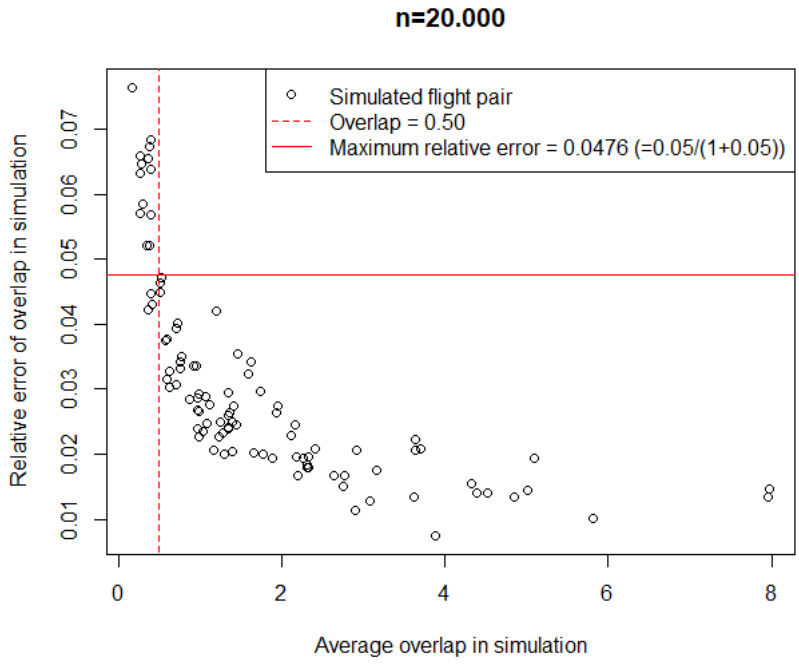


Figure 53:Relative error versus the average estimated overlap for 20.000 MC replications

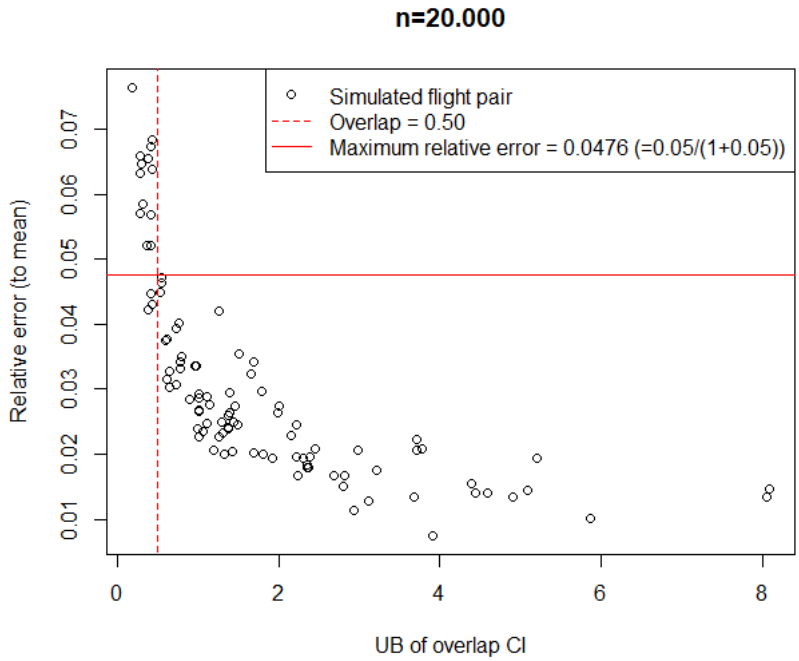


Figure 54: UB of overlap versus the error relative to the mean of the overlap, for 20.000 MC replications

Appendix G: Comparing sample data

Table 44: Uncertainties in flight timings for the analyzed dataset

Flight's baggage class	# flights	Transport time		On-Belt duration		Arrival time
	Total	Mean	St. Dev.	Mean	St. Dev.	St. Dev.
A	**~10%**	14.6	4.0	7.3	4.0	2.4
B	**~65%**	18.3	5.9	8.8	3.2	2.4
C	**~25%**	16.9	6.4	13.0	9.1	2.4

Table 45: Uncertainties in flight timings for another week of data (2 weeks later than the analyzed set)

Flight's baggage class	# flights	Transport time		On-Belt duration		Arrival time
	Total	Mean	St. Dev.	Mean	St. Dev.	St. Dev.
A	**~10%**	15.3	4.2	7.1	3.3	2.4
B	**~65%**	18.3	6.4	8.0	2.7	2.4
C	**~25%**	16.6	6.5	13.7	9.1	2.4

Table 46: Uncertainties in arrival time estimates prior to arrival for the analyzed dataset

Time	Mean overestimation of arrival time	Standard Deviation
$t_{arrival} - 1$	0.59	2.38
$t_{arrival} - 2$	2.29	3.97
$t_{arrival} - 3$	2.56	3.76
$t_{arrival} - 4$	-0.12	4.76
$t_{arrival} - 5$	-0.68	5.73
$t_{arrival} - 6$	-0.18	6.16
$t_{arrival} - 7$	0.33	7.30
$t_{arrival} - 8$	0.59	7.96
$t_{arrival} - 9$	0.84	8.52

Table 47: Uncertainties in arrival time estimates prior to arrival for another week of data (2 weeks later than the analyzed set)

Time	Mean overestimation of arrival time	Standard Deviation
$t_{arrival} - 1$	0.88	2.44
$t_{arrival} - 2$	2.26	4.27
$t_{arrival} - 3$	2.54	4.32
$t_{arrival} - 4$	0.05	5.02
$t_{arrival} - 5$	-1.14	6.12
$t_{arrival} - 6$	-0.84	7.17
$t_{arrival} - 7$	-0.18	8.42
$t_{arrival} - 8$	0.09	9.46
$t_{arrival} - 9$	0.41	10.17

Appendix H: Baseline results

In this appendix we will show the expected and realized performance of a solution. When the solution approaches use time estimates to assign belts to the flights, the solution approaches expect to achieve a certain performance based on the expected overlap between flights at the same belt, belt preferences, and the expected number of alliance flights starting at empty belts.

Determining the expected performance

The expected performance for the solution approaches using basic overlap estimates, are calculated using the time estimates and the objective function from Section 2.2.2:

$$\sum_{i\in F}\sum_{j\in F} O_{ij}t_{ij} \;+\; \beta\sum_{i\in F} s_i * V_i - \;\gamma\sum_{i\in F} Q_i, \qquad (1)$$

In which t_{ij} is the overlap of the expected belt occupation slots.

For the expected overlap of the MC overlap approaches, we use a different performance measurement:

$$\sum_{i\in F}\sum_{j\in F} O_{ij}t_{ij} \;+\; \beta\sum_{j\in F} s_j * (1-\prod_{i\in F}(1-B_{ij}*O_{ij})) - \;\gamma\sum_{i\in F} Q_i, \qquad (2)$$

In which t_{ij} is the expected overlap between flight *i* and flight *j*, following from the average overlap of the MC simulations as described in Section 4.3. B_{ij} is the probability that flight *i* is on the belt at the arrival of flight *j*, and thus blocking the empty belt arrival of flight *j*, calculated using the same MC simulations.

Even though the MIP solver uses another goal function to optimize its performance:

$$\min\sum_{i\in F}\sum_{j\in F} O_{ij}t_{ij} \;+\; \beta\sum_{j\in F} s_j * \sum_{i\in F} b_{ij} * O_{ij} - \;\gamma\sum_{i\in F} Q_i \qquad (3)$$

We will calculate the expected performance based on the goal function (2). The realized performance is always calculated using goal function (1).

Expected results for the unbiased estimates

		Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Week total
Basic overlap estimation	*MIP Solver*	**-1,673**	**-1,441**	**-1,575**	-1,832	**-1,661**	**-1,695**	**-1,575**	**-11,452**
	Avg SA	-1,673	**-1,441**	-1,574	-1,829	-1,660	-1,692	-1,574	-11,442
	FCFS	-1,534	-1,389	-1,441	-1,588	-1,559	-1,616	-1,448	-10,575
	Greedy	-1,627	-1,397	-1,552	-1,763	-1,598	-1,660	-1,551	-11,148
	FCFS + (1)	-1,551	-1,353	-1,465	-1,690	-1,549	-1,613	-1,421	-10,642
	FCFS + (2)	-1,559	-1,345	-1,472	-1,651	-1,581	-1,593	-1,465	-10,666
	SA rep1	-1,672	**-1,441**	-1,573	-1,827	-1,659	-1,692	**-1,575**	-11,439
	SA rep2	**-1,673**	**-1,441**	-1,573	**-1,833**	-1,660	-1,693	-1,571	-11,444
	SA rep3	**-1,673**	**-1,441**	-1,575	-1,827	**-1,661**	-1,691	-1,574	-11,442

	SA rep4	-1,672	**-1,441**	-1,574	-1,825	-1,660	-1,694	**-1,575**	-11,441
	SA rep5	**-1,673**	**-1,441**	-1,574	-1,832	-1,659	-1,689	-1,574	-11,442
Monte Carlo overlap estimates	*MIP Solver*	-1,315	-1,230	-1,342	-1,332	-1,273	-1,317	-1,229	-9,037
	Avg SA	-1,326	-1,227	-1,356	-1,388	-1,285	-1,335	-1,266	-9,181
	FCFS	-1,260	-1,169	-1,293	-1,314	-1,233	-1,252	-1,164	-8,685
	Greedy	-1,321	-1,211	-1,327	-1,373	-1,261	-1,335	-1,249	-9,076
	FCFS + (1)	**-1,418**	**-1,276**	**-1,421**	**-1,500**	**-1,446**	**-1,424**	-1,305	**-9,790**
	FCFS + (2)	-1,404	**-1,276**	-1,406	-1,377	-1,397	-1,372	**-1,369**	-9,601
	SA rep1	-1,330.0	-1,228.2	-1,357.0	-1,385.9	-1,282.3	-1,337.7	-1,263.6	-9,184.7
	SA rep2	-1,325.2	-1,228.7	-1,354.3	-1,394.0	-1,277.0	-1,339.0	-1,264.2	-9,182.4
	SA rep3	-1,322.1	-1,224.3	-1,359.4	-1,384.8	-1,284.2	-1,334.6	-1,267.6	-9,177.0
	SA rep4	-1,325.7	-1,226.8	-1,353.4	-1,387.3	-1,288.7	-1,332.1	-1,268.3	-9,182.3
	SA rep5	-1,326.2	-1,229.2	-1,354.3	-1,385.7	-1,290.8	-1,330.7	-1,263.9	-9,180.8

Realized results for the unbiased estimates

		Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Week total
Basic overlap estimation	*MIP Solver*	-977	-688	-897	-1,213	-1,026	-1,244	-1,033	-7,078
	Avg SA	-1,003	-782	-1,005	-1,163	-991	-1,230	-1,107	-7,280
	FCFS	-1,049	-825	-1,079	-1,191	-1,007	-1,348	-1,093	-7,592
	Greedy	-955	-613	-879	-1,073	-893	-1,197	-916	-6,526
	FCFS + (1)	-1,154	-906	-1,078	-1,224	-1,086	-1,411	-1,150	-8,009
	FCFS + (2)	-1,220	-909	-1,042	-1,249	-1,107	-1,355	-1,164	-8,046
	SA rep1	-930	-762	-982	-1,158	-1,045	-1,284	-1,125	-7,286
	SA rep2	-978	-750	-1,003	-1,187	-976	-1,098	-1,196	-7,188
	SA rep3	-1,033	-750	-1,004	-1,185	-1,013	-1,264	-1,035	-7,284
	SA rep4	-1,021	-869	-1,044	-1,061	-975	-1,248	-1,118	-7,336
	SA rep5	-1,051	-778	-992	-1,223	-946	-1,254	-1,061	-7,305
Monte Carlo overlap estimates	*MIP Solver*	-1,183	-935	-1,218	-1,233	-1,088	-1,266	-1,106	-8,029
	Avg SA	-1,128	-908	-1,187	-1,298	-1,092	-1,304	-1,143	-8,060
	FCFS	-1,102	-799	-1,129	-1,272	-1,118	-1,377	-1,175	-7,972
	Greedy	-1,160	-876	-1,178	-1,220	-1,022	-1,293	-1,156	-7,905
	FCFS + (1)	-1,186	-965	-1,143	-1,294	-1,057	-1,283	-1,129	-8,057
	FCFS + (2)	-1,096	-974	-1,151	-1,145	-1,138	-1,277	-1,145	-7,926
	SA rep1	-1,139	-921	-1,197	-1,317	-1,090	-1,270	-1,186	-8,120
	SA rep2	-1,175	-995	-1,244	-1,298	-1,136	-1,357	-1,082	-8,287
	SA rep3	-1,116	-789	-1,175	-1,253	-1,074	-1,281	-1,128	-7,816
	SA rep4	-1,079	-911	-1,177	-1,297	-1,086	-1,265	-1,199	-8,014
	SA rep5	-1,129	-926	-1,143	-1,325	-1,074	-1,345	-1,119	-8,061

Appendix I: All experimental outcomes

MIP, optimized using basic overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	-11,4	-10,5	-10,0	-9,6	-10,7	-10,3	-9,8	-9,4	-10,0	-9,8	-9,4	-9,1	-9,3	-9,2	-8,8	-8,4
	2	-9,8	-9,6	-9,3	-9,0	-9,8	-9,4	-9,2	-9,0	-9,4	-9,3	-8,8	-8,7	-8,9	-8,7	-8,5	-8,3
	4	-8,7	-8,6	-8,4	-8,1	-8,7	-8,4	-8,4	-8,1	-8,4	-8,2	-8,1	-8,0	-8,2	-8,1	-8,0	-7,8
	6	-8,0	-7,8	-7,6	-7,5	-7,9	-7,9	-7,8	-7,6	-7,7	-7,7	-7,6	-7,4	-7,7	-7,6	-7,3	-7,4
2	0	-11,4	-10,5	-10,0	-9,6	-10,6	-10,3	-9,6	-9,4	-9,9	-9,7	-9,3	-8,9	-9,3	-9,1	-8,7	-8,5
	2	-9,8	-9,6	-9,3	-8,8	-9,6	-9,4	-9,1	-9,0	-9,4	-9,2	-9,0	-8,5	-8,8	-8,7	-8,7	-8,3
	4	-8,6	-8,5	-8,4	-8,1	-8,6	-8,5	-8,1	-8,1	-8,4	-8,4	-8,1	-8,0	-8,2	-8,1	-8,0	-7,5
	6	-7,8	-7,9	-7,6	-7,4	-7,9	-7,8	-7,6	-7,6	-7,8	-7,7	-7,5	-7,2	-7,7	-7,5	-7,5	-7,4
4	0	-11,4	-10,6	-9,9	-9,5	-10,6	-10,2	-9,8	-9,5	-10,0	-9,6	-9,4	-9,0	-9,3	-9,1	-8,8	-8,5
	2	-9,9	-9,6	-9,2	-9,0	-9,7	-9,4	-9,2	-8,9	-9,4	-9,1	-8,9	-8,6	-8,9	-8,8	-8,6	-8,4
	4	-8,6	-8,6	-8,3	-8,2	-8,5	-8,5	-8,2	-8,0	-8,4	-8,3	-8,1	-8,0	-8,1	-8,0	-7,9	-7,7
	6	-7,9	-7,9	-7,7	-7,5	-7,8	-7,7	-7,6	-7,4	-7,8	-7,8	-7,5	-7,5	-7,6	-7,5	-7,5	-7,3
6	0	-11,4	-10,5	-10,0	-9,4	-10,6	-10,2	-9,7	-9,3	-9,9	-9,7	-9,2	-8,9	-9,2	-9,1	-8,9	-8,6
	2	-9,8	-9,6	-9,4	-9,1	-9,7	-9,5	-9,1	-8,8	-9,3	-9,2	-8,9	-8,4	-9,0	-8,7	-8,4	-8,2
	4	-8,7	-8,5	-8,3	-8,1	-8,6	-8,4	-8,3	-7,9	-8,5	-8,3	-8,1	-7,9	-8,3	-8,0	-7,9	-7,6
	6	-7,9	-7,9	-7,8	-7,6	-7,8	-7,7	-7,7	-7,5	-7,8	-7,7	-7,7	-7,2	-7,6	-7,6	-7,3	-7,3

MIP, optimized using Monte Carlo overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	-10,7	-10,2	-9,9	-9,8	-10,4	-10,1	-9,8	-9,5	-10,0	-9,7	-9,4	-9,2	-9,6	-9,5	-9,2	-9,1
	2	-10,0	-9,9	-9,6	-9,3	-9,9	-9,7	-9,6	-9,2	-9,6	-9,5	-9,3	-9,0	-9,4	-9,3	-9,1	-8,9
	4	-9,4	-9,3	-9,0	-8,7	-9,3	-9,3	-8,9	-8,7	-9,1	-9,1	-8,7	-8,7	-9,0	-8,8	-8,6	-8,6
	6	-8,9	-8,7	-8,7	-8,3	-8,8	-8,8	-8,4	-8,2	-8,7	-8,7	-8,5	-8,3	-8,5	-8,5	-8,2	-8,0
2	0	-10,8	-10,4	-10,2	-9,7	-10,5	-10,4	-10,0	-9,6	-10,2	-10,0	-9,6	-9,3	-9,8	-9,6	-9,3	-8,9
	2	-10,1	-10,1	-9,7	-9,3	-10,1	-9,9	-9,6	-9,2	-9,8	-9,6	-9,3	-9,1	-9,4	-9,3	-9,1	-8,8
	4	-9,5	-9,4	-9,0	-8,7	-9,4	-9,2	-9,1	-8,8	-9,2	-9,0	-8,7	-8,6	-9,0	-8,9	-8,7	-8,4
	6	-8,9	-8,7	-8,5	-8,4	-8,9	-8,7	-8,4	-8,2	-8,7	-8,7	-8,5	-8,3	-8,4	-8,4	-8,2	-8,2
4	0	-10,7	-10,3	-9,9	-9,5	-10,5	-10,2	-10,0	-9,4	-10,1	-9,9	-9,4	-9,1	-9,8	-9,4	-9,0	-8,8
	2	-10,2	-10,0	-9,5	-9,3	-10,0	-9,7	-9,4	-8,9	-9,8	-9,5	-9,3	-8,8	-9,4	-9,1	-8,9	-8,6
	4	-9,3	-9,2	-8,8	-8,6	-9,3	-9,1	-8,8	-8,4	-9,0	-9,0	-8,6	-8,4	-8,8	-8,6	-8,5	-8,2
	6	-8,6	-8,7	-8,4	-7,8	-8,7	-8,5	-8,1	-7,9	-8,5	-8,4	-8,2	-7,9	-8,4	-8,2	-7,8	-7,8
6	0	-10,3	-10,0	-9,6	-9,2	-10,2	-9,9	-9,4	-9,1	-9,9	-9,5	-9,2	-8,8	-9,3	-9,1	-8,8	-8,5
	2	-9,8	-9,5	-9,1	-8,8	-9,9	-9,5	-9,1	-8,7	-9,5	-9,2	-8,8	-8,4	-9,2	-8,8	-8,5	-8,2
	4	-9,2	-8,9	-8,5	-8,0	-9,1	-8,7	-8,6	-8,0	-9,0	-8,6	-8,1	-7,8	-8,7	-8,3	-8,1	-7,6
	6	-8,6	-8,3	-7,8	-7,7	-8,4	-8,3	-8,1	-7,8	-8,3	-7,9	-7,8	-7,2	-8,0	-7,9	-7,5	-7,0

SA, optimized using basic overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time 0 / σ On-belt 0	2	4	6	1 / 0	2	4	6	2 / 0	2	4	6	3 / 0	2	4	6
0	0	-11,4	-10,6	-9,9	-9,4	-10,6	-10,3	-9,8	-9,5	-9,9	-9,8	-9,2	-9,1	-9,3	-9,2	-8,9	-8,7
	2	-9,8	-9,6	-9,3	-9,0	-9,7	-9,4	-9,1	-8,8	-9,3	-9,1	-8,9	-8,6	-8,8	-8,8	-8,4	-8,3
	4	-8,7	-8,6	-8,3	-8,2	-8,7	-8,5	-8,3	-8,1	-8,5	-8,5	-8,3	-8,1	-8,2	-8,1	-7,9	-7,9
	6	-8,2	-7,9	-7,7	-7,6	-8,0	-7,9	-7,7	-7,6	-7,8	-7,9	-7,6	-7,4	-7,7	-7,7	-7,6	-7,3
2	0	-11,4	-10,7	-10,2	-9,8	-10,9	-10,5	-10,1	-9,7	-10,3	-10,1	-9,6	-9,3	-9,7	-9,5	-9,1	-9,0
	2	-10,2	-9,9	-9,5	-9,2	-10,0	-9,8	-9,6	-9,2	-9,7	-9,4	-9,2	-8,9	-9,2	-9,2	-8,9	-8,3
	4	-9,1	-9,1	-8,7	-8,4	-9,0	-8,9	-8,6	-8,3	-8,8	-8,8	-8,5	-8,3	-8,6	-8,5	-8,3	-8,0
	6	-8,3	-8,1	-8,0	-7,7	-8,2	-8,2	-8,0	-7,8	-8,2	-8,1	-7,9	-7,9	-7,9	-8,0	-7,8	-7,6
4	0	-11,3	-10,8	-10,3	-9,9	-10,9	-10,5	-10,1	-9,6	-10,4	-10,2	-9,8	-9,5	-9,8	-9,7	-9,4	-9,1
	2	-10,4	-10,2	-9,7	-9,4	-10,2	-10,0	-9,7	-9,4	-10,0	-9,8	-9,3	-9,2	-9,5	-9,4	-9,2	-8,9
	4	-9,3	-9,2	-8,8	-8,7	-9,3	-9,1	-8,8	-8,6	-9,0	-8,9	-8,8	-8,4	-8,9	-8,9	-8,5	-8,2
	6	-8,6	-8,4	-8,3	-8,0	-8,5	-8,4	-8,0	-7,9	-8,4	-8,2	-8,3	-8,0	-8,2	-8,1	-8,0	-8,1
6	0	-11,2	-10,7	-10,3	-9,9	-10,8	-10,5	-10,1	-9,8	-10,4	-10,1	-9,8	-9,4	-10,0	-9,8	-9,4	-9,1
	2	-10,3	-10,2	-9,8	-9,4	-10,2	-10,0	-9,7	-9,5	-10,0	-9,8	-9,4	-9,2	-9,5	-9,5	-9,3	-9,0
	4	-9,4	-9,2	-9,1	-8,8	-9,4	-9,3	-9,1	-8,9	-9,2	-9,1	-8,8	-8,6	-8,9	-8,9	-8,6	-8,4
	6	-8,6	-8,6	-8,4	-8,0	-8,6	-8,6	-8,3	-8,3	-8,6	-8,4	-8,2	-8,1	-8,2	-8,3	-8,1	-8,0

SA, optimized using Monte Carlo overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time 0 / σ On-belt 0	2	4	6	1 / 0	2	4	6	2 / 0	2	4	6	3 / 0	2	4	6
0	0	-11,3	-10,6	-10,4	-10,1	-10,8	-10,5	-10,1	-9,8	-10,3	-10,1	-9,8	-9,4	-9,8	-9,8	-9,5	-9,2
	2	-10,4	-10,1	-9,9	-9,5	-10,3	-10,0	-9,9	-9,4	-10,0	-9,8	-9,6	-9,2	-9,7	-9,5	-9,3	-9,1
	4	-9,6	-9,4	-9,2	-9,0	-9,6	-9,4	-9,2	-8,9	-9,4	-9,4	-9,0	-8,9	-9,1	-9,0	-8,9	-8,7
	6	-9,2	-8,9	-8,9	-8,5	-9,0	-8,8	-8,7	-8,5	-9,0	-8,8	-8,8	-8,4	-8,7	-8,7	-8,5	-8,2
2	0	-10,6	-10,5	-10,3	-9,8	-10,6	-10,3	-10,1	-9,7	-10,3	-10,1	-9,7	-9,4	-9,9	-9,7	-9,4	-9,2
	2	-10,3	-10,1	-9,8	-9,5	-10,2	-10,0	-9,7	-9,4	-10,0	-9,8	-9,5	-9,4	-9,7	-9,4	-9,3	-9,0
	4	-9,6	-9,7	-9,3	-9,0	-9,6	-9,4	-9,3	-9,0	-9,5	-9,3	-9,1	-8,9	-9,3	-9,2	-8,9	-8,7
	6	-9,1	-9,0	-8,8	-8,7	-9,1	-8,9	-8,9	-8,4	-8,9	-8,9	-8,7	-8,6	-8,8	-8,7	-8,6	-8,5
4	0	-10,2	-10,1	-9,9	-9,5	-10,1	-9,9	-9,8	-9,4	-10,0	-9,8	-9,6	-9,3	-9,8	-9,6	-9,4	-9,1
	2	-10,0	-9,9	-9,6	-9,5	-9,9	-9,8	-9,6	-9,2	-9,8	-9,6	-9,4	-9,2	-9,7	-9,4	-9,1	-9,0
	4	-9,6	-9,4	-9,2	-9,1	-9,5	-9,5	-9,3	-9,0	-9,4	-9,4	-9,1	-8,9	-9,2	-9,1	-8,8	-8,6
	6	-9,0	-9,0	-8,7	-8,5	-9,0	-8,9	-8,7	-8,5	-8,9	-8,7	-8,9	-8,4	-8,7	-8,6	-8,4	-8,4
6	0	-9,6	-9,7	-9,5	-9,4	-9,5	-9,6	-9,4	-9,2	-9,5	-9,4	-9,4	-9,3	-9,5	-9,3	-9,2	-9,0
	2	-9,6	-9,5	-9,3	-9,2	-9,6	-9,4	-9,4	-9,3	-9,6	-9,3	-9,3	-9,0	-9,3	-9,3	-9,2	-9,0
	4	-9,4	-9,4	-9,2	-8,9	-9,4	-9,3	-9,1	-8,8	-9,3	-9,2	-8,8	-8,7	-9,0	-9,0	-8,9	-8,5
	6	-8,9	-8,9	-8,7	-8,5	-8,9	-8,8	-8,8	-8,6	-8,8	-8,7	-8,5	-8,5	-8,6	-8,7	-8,6	-8,4

Greedy, optimized using basic overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	-11,1	-10,2	-9,6	-9,1	-10,3	-9,9	-9,4	-9,1	-9,6	-9,4	-8,9	-8,8	-8,9	-8,9	-8,4	-8,2
	2	-9,6	-9,4	-9,0	-8,7	-9,4	-9,2	-8,8	-8,5	-9,1	-8,9	-8,6	-8,3	-8,5	-8,4	-8,0	-7,8
	4	-8,2	-8,2	-7,9	-7,8	-8,4	-8,1	-7,8	-7,7	-8,1	-8,0	-7,9	-7,7	-7,9	-7,7	-7,6	-7,5
	6	-7,5	-7,5	-7,3	-7,1	-7,4	-7,5	-7,3	-7,2	-7,4	-7,3	-7,2	-7,0	-7,2	-7,2	-6,9	-6,8
2	0	-10,4	-10,1	-9,6	-9,1	-10,3	-10,0	-9,5	-9,1	-9,9	-9,7	-9,3	-8,8	-9,4	-9,0	-8,8	-8,6
	2	-9,8	-9,5	-9,2	-8,9	-9,7	-9,5	-9,1	-8,8	-9,4	-9,2	-8,9	-8,4	-8,9	-8,9	-8,5	-7,9
	4	-8,8	-8,7	-8,4	-7,9	-8,7	-8,6	-8,3	-8,0	-8,6	-8,4	-8,1	-8,0	-8,3	-8,1	-7,9	-7,7
	6	-7,9	-7,8	-7,7	-7,4	-8,0	-7,8	-7,7	-7,5	-7,7	-7,7	-7,5	-7,4	-7,5	-7,6	-7,3	-7,2
4	0	-9,9	-9,7	-9,4	-9,0	-9,9	-9,7	-9,2	-8,8	-9,7	-9,5	-9,1	-8,8	-9,3	-9,1	-8,8	-8,4
	2	-9,6	-9,5	-9,1	-8,8	-9,5	-9,3	-9,0	-8,8	-9,4	-9,2	-8,8	-8,6	-9,0	-8,9	-8,6	-8,3
	4	-8,8	-8,7	-8,3	-8,2	-8,8	-8,6	-8,2	-8,1	-8,6	-8,4	-8,1	-7,9	-8,4	-8,3	-8,0	-7,8
	6	-8,1	-8,1	-7,8	-7,6	-8,1	-8,0	-7,6	-7,6	-7,8	-7,8	-7,8	-7,5	-7,8	-7,7	-7,5	-7,5
6	0	-9,4	-9,4	-9,1	-8,7	-9,3	-9,2	-9,0	-8,5	-9,2	-9,1	-8,9	-8,3	-9,0	-9,0	-8,6	-8,3
	2	-9,3	-9,2	-8,8	-8,5	-9,3	-9,1	-8,8	-8,6	-9,1	-9,0	-8,7	-8,4	-8,8	-8,7	-8,5	-8,2
	4	-8,8	-8,6	-8,4	-8,2	-8,8	-8,7	-8,3	-8,1	-8,6	-8,5	-8,2	-7,9	-8,3	-8,2	-7,9	-7,7
	6	-7,9	-8,0	-7,7	-7,4	-7,9	-7,9	-7,7	-7,5	-7,9	-7,9	-7,6	-7,4	-7,7	-7,7	-7,5	-7,3

Greedy, optimized using Monte Carlo overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	-11,0	-10,4	-10,0	-9,7	-10,5	-10,2	-9,8	-9,5	-9,9	-9,7	-9,5	-9,2	-9,6	-9,6	-9,2	-9,0
	2	-10,1	-9,8	-9,7	-9,3	-10,0	-9,7	-9,5	-9,2	-9,7	-9,5	-9,3	-9,0	-9,4	-9,3	-9,2	-8,8
	4	-9,3	-9,2	-9,0	-8,7	-9,3	-9,2	-9,0	-8,8	-9,0	-9,1	-8,7	-8,5	-8,9	-8,8	-8,6	-8,5
	6	-8,8	-8,7	-8,6	-8,4	-8,7	-8,7	-8,6	-8,3	-8,7	-8,6	-8,5	-8,3	-8,4	-8,5	-8,3	-8,0
2	0	-10,3	-10,2	-9,9	-9,5	-10,3	-10,0	-9,7	-9,4	-9,9	-9,7	-9,4	-9,3	-9,5	-9,4	-9,1	-8,9
	2	-10,0	-9,8	-9,5	-9,3	-9,9	-9,7	-9,4	-9,1	-9,6	-9,4	-9,2	-9,1	-9,3	-9,2	-8,9	-8,6
	4	-9,3	-9,3	-8,9	-8,7	-9,3	-9,1	-9,0	-8,7	-9,2	-8,9	-8,8	-8,6	-8,9	-8,9	-8,7	-8,4
	6	-8,8	-8,7	-8,6	-8,4	-8,8	-8,7	-8,6	-8,3	-8,6	-8,7	-8,3	-8,5	-8,4	-8,4	-8,2	-8,3
4	0	-9,8	-9,7	-9,5	-9,3	-9,8	-9,6	-9,5	-9,2	-9,6	-9,5	-9,3	-9,0	-9,4	-9,2	-9,1	-8,8
	2	-9,6	-9,6	-9,3	-9,2	-9,5	-9,5	-9,3	-8,8	-9,5	-9,3	-9,1	-8,9	-9,2	-9,1	-8,8	-8,5
	4	-9,2	-9,0	-8,8	-8,7	-9,1	-9,1	-9,0	-8,6	-9,0	-9,0	-8,7	-8,4	-8,9	-8,8	-8,6	-8,4
	6	-8,6	-8,6	-8,4	-8,1	-8,7	-8,6	-8,3	-8,2	-8,5	-8,4	-8,5	-8,2	-8,5	-8,3	-8,1	-8,1
6	0	-9,3	-9,2	-9,1	-9,0	-9,2	-9,2	-8,9	-8,8	-9,2	-9,1	-9,0	-8,8	-9,0	-9,0	-8,9	-8,6
	2	-9,3	-9,2	-9,0	-8,9	-9,3	-9,2	-9,0	-8,8	-9,3	-9,1	-8,9	-8,7	-9,0	-9,0	-8,7	-8,6
	4	-9,0	-8,9	-8,7	-8,5	-9,0	-8,8	-8,5	-8,4	-8,9	-8,8	-8,5	-8,3	-8,6	-8,6	-8,3	-8,0
	6	-8,5	-8,4	-8,3	-8,2	-8,4	-8,4	-8,2	-8,1	-8,4	-8,3	-8,2	-8,0	-8,2	-8,2	-8,1	-7,9

FCFS, optimized using basic overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	-10,2	-9,8	-9,5	-9,2	-10,2	-9,8	-9,5	-9,1	-10,2	-9,8	-9,4	-9,1	-10,2	-9,8	-9,5	-9,1
	2	-9,4	-9,3	-9,0	-8,7	-9,4	-9,3	-9,0	-8,7	-9,4	-9,3	-8,9	-8,7	-9,4	-9,3	-8,9	-8,8
	4	-8,6	-8,6	-8,4	-8,1	-8,7	-8,6	-8,4	-8,1	-8,7	-8,6	-8,3	-8,1	-8,7	-8,5	-8,4	-8,2
	6	-8,1	-8,1	-7,9	-7,7	-8,2	-8,1	-7,9	-7,6	-8,2	-8,0	-7,8	-7,7	-8,2	-8,1	-7,8	-7,8
2	0	-9,6	-9,4	-9,1	-8,7	-9,6	-9,4	-9,1	-8,8	-9,6	-9,4	-9,0	-8,7	-9,6	-9,4	-9,0	-8,7
	2	-9,2	-9,1	-8,8	-8,5	-9,2	-9,1	-8,7	-8,5	-9,2	-9,0	-8,8	-8,5	-9,2	-9,1	-8,8	-8,6
	4	-8,4	-8,5	-8,1	-8,1	-8,5	-8,4	-8,1	-8,0	-8,6	-8,4	-8,1	-7,9	-8,6	-8,4	-8,2	-8,2
	6	-7,9	-8,0	-7,7	-7,5	-8,0	-8,0	-7,6	-7,5	-7,9	-7,7	-7,6	-7,6	-8,0	-7,9	-7,8	-7,5
4	0	-9,2	-9,1	-8,7	-8,6	-9,2	-9,1	-8,8	-8,5	-9,2	-9,1	-8,9	-8,5	-9,2	-9,1	-8,8	-8,7
	2	-8,9	-8,8	-8,6	-8,3	-8,9	-8,8	-8,6	-8,3	-9,0	-8,9	-8,4	-8,2	-8,9	-8,8	-8,5	-8,2
	4	-8,4	-8,3	-8,0	-7,8	-8,4	-8,3	-8,0	-7,9	-8,4	-8,1	-8,1	-7,9	-8,4	-8,3	-8,0	-7,8
	6	-7,8	-7,7	-7,6	-7,4	-7,7	-7,8	-7,6	-7,6	-7,8	-7,8	-7,6	-7,4	-7,9	-7,7	-7,7	-7,4
6	0	-8,8	-8,7	-8,5	-8,1	-8,8	-8,7	-8,5	-8,2	-8,8	-8,8	-8,5	-8,2	-8,8	-8,7	-8,6	-8,2
	2	-8,7	-8,6	-8,3	-8,0	-8,6	-8,5	-8,3	-8,2	-8,6	-8,5	-8,4	-8,0	-8,7	-8,5	-8,3	-8,0
	4	-8,1	-8,1	-7,8	-7,7	-8,1	-8,1	-7,9	-7,7	-8,1	-8,1	-7,7	-7,8	-8,1	-8,0	-7,8	-7,6
	6	-7,6	-7,5	-7,4	-7,1	-7,6	-7,6	-7,4	-7,2	-7,5	-7,5	-7,5	-7,2	-7,6	-7,5	-7,3	-7,1

FCFS, optimized using Monte Carlo overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	-10,3	-9,8	-9,6	-9,2	-10,3	-9,8	-9,6	-9,2	-10,3	-9,8	-9,5	-9,1	-10,3	-9,8	-9,7	-9,4
	2	-9,4	-9,3	-9,2	-8,9	-9,5	-9,3	-9,2	-8,9	-9,4	-9,3	-9,3	-8,9	-9,4	-9,4	-9,2	-9,0
	4	-8,9	-8,6	-8,6	-8,3	-8,9	-8,8	-8,5	-8,4	-8,8	-8,8	-8,4	-8,5	-8,9	-8,9	-8,6	-8,3
	6	-8,4	-8,4	-8,2	-8,0	-8,5	-8,3	-8,2	-8,0	-8,5	-8,5	-8,1	-8,0	-8,5	-8,3	-8,2	-8,0
2	0	-9,6	-9,4	-9,3	-8,9	-9,6	-9,4	-9,3	-9,0	-9,6	-9,4	-9,2	-9,0	-9,6	-9,4	-9,2	-9,0
	2	-9,3	-9,1	-9,0	-8,8	-9,3	-9,2	-8,8	-8,8	-9,3	-9,2	-8,9	-8,8	-9,3	-9,2	-8,8	-8,7
	4	-8,7	-8,8	-8,4	-8,3	-8,8	-8,7	-8,5	-8,3	-8,9	-8,6	-8,4	-8,3	-8,9	-8,8	-8,5	-8,3
	6	-8,2	-8,2	-8,1	-8,1	-8,3	-8,2	-8,1	-7,9	-8,3	-8,3	-8,1	-8,2	-8,3	-8,3	-8,0	-8,0
4	0	-9,2	-9,2	-8,9	-8,6	-9,2	-9,2	-9,0	-8,6	-9,2	-9,2	-8,9	-8,6	-9,2	-9,2	-9,0	-8,6
	2	-9,1	-9,0	-8,8	-8,6	-9,1	-9,0	-8,8	-8,4	-9,1	-9,0	-8,7	-8,5	-9,0	-8,9	-8,9	-8,4
	4	-8,7	-8,5	-8,4	-8,3	-8,7	-8,7	-8,4	-8,3	-8,7	-8,6	-8,3	-8,3	-8,6	-8,6	-8,4	-8,3
	6	-8,3	-8,2	-7,9	-7,6	-8,2	-8,2	-7,9	-7,8	-8,3	-8,1	-8,2	-7,9	-8,2	-8,1	-7,9	-8,0
6	0	-8,8	-8,8	-8,7	-8,6	-8,8	-8,9	-8,7	-8,5	-8,8	-8,9	-8,8	-8,5	-8,8	-8,8	-8,7	-8,5
	2	-8,8	-8,7	-8,4	-8,3	-8,8	-8,7	-8,6	-8,4	-8,7	-8,6	-8,5	-8,3	-8,7	-8,7	-8,5	-8,4
	4	-8,3	-8,3	-8,2	-8,0	-8,3	-8,4	-8,1	-7,9	-8,4	-8,3	-8,2	-7,9	-8,4	-8,2	-8,3	-8,0
	6	-8,0	-7,9	-7,8	-7,6	-8,1	-8,1	-7,8	-7,8	-8,1	-8,0	-7,9	-7,6	-7,9	-7,9	-7,9	-7,4

FCFS + future demand, optimized using basic overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time	0				1				2				3			
			σ On-belt				σ On-belt				σ On-belt				σ On-belt			
			0	2	4	6	0	2	4	6	0	2	4	6	0	2	4	6
0	0	0	-10,7	-10,1	-9,6	-9,3	-10,5	-10,0	-9,7	-9,3	-10,3	-9,8	-9,4	-9,2	-10,1	-9,8	-9,4	-9,2
		2	-9,7	-9,5	-9,2	-8,9	-9,7	-9,4	-9,1	-8,9	-9,5	-9,3	-9,1	-8,8	-9,4	-9,2	-8,8	-8,6
		4	-8,7	-8,7	-8,3	-8,3	-8,8	-8,6	-8,4	-8,2	-8,7	-8,7	-8,4	-8,3	-8,6	-8,5	-8,4	-8,1
		6	-8,3	-8,1	-8,1	-7,8	-8,3	-8,2	-8,0	-7,8	-8,2	-8,1	-8,0	-7,7	-8,0	-8,0	-8,0	-7,6
	2	0	-10,2	-9,9	-9,5	-9,2	-10,0	-9,7	-9,3	-9,1	-9,8	-9,6	-9,2	-8,9	-9,6	-9,5	-9,1	-8,8
		2	-9,7	-9,5	-9,2	-9,0	-9,5	-9,4	-9,2	-8,8	-9,4	-9,2	-9,0	-8,6	-9,3	-9,2	-8,8	-8,2
		4	-8,9	-8,9	-8,5	-8,2	-8,7	-8,6	-8,3	-8,0	-8,5	-8,5	-8,2	-8,2	-8,6	-8,5	-8,3	-8,1
		6	-8,2	-8,1	-8,1	-7,7	-8,1	-8,0	-7,9	-7,7	-8,1	-8,0	-7,9	-7,7	-7,9	-8,0	-7,7	-7,5
	4	0	-9,5	-9,4	-9,1	-8,8	-9,5	-9,4	-9,0	-8,7	-9,4	-9,2	-9,0	-8,7	-9,2	-9,1	-8,8	-8,5
		2	-9,4	-9,2	-8,9	-8,7	-9,2	-9,2	-8,8	-8,7	-9,1	-9,1	-8,7	-8,5	-9,0	-8,9	-8,7	-8,3
		4	-8,6	-8,6	-8,2	-8,1	-8,7	-8,6	-8,3	-8,1	-8,6	-8,5	-8,2	-7,9	-8,5	-8,6	-8,1	-7,9
		6	-8,2	-8,1	-7,9	-7,7	-8,0	-8,0	-7,7	-7,7	-7,9	-8,0	-7,7	-7,6	-7,8	-7,8	-7,7	-7,7
	6	0	-9,1	-9,0	-8,8	-8,5	-8,9	-8,8	-8,7	-8,3	-8,8	-8,8	-8,5	-8,1	-8,8	-8,7	-8,3	-8,1
		2	-9,0	-8,9	-8,7	-8,3	-8,8	-8,7	-8,5	-8,1	-8,8	-8,5	-8,4	-8,1	-8,6	-8,6	-8,3	-8,1
		4	-8,5	-8,5	-8,4	-8,1	-8,4	-8,4	-8,1	-7,9	-8,3	-8,3	-8,0	-7,7	-8,2	-8,2	-7,9	-7,8
		6	-7,9	-7,9	-7,7	-7,4	-7,9	-7,8	-7,7	-7,6	-7,7	-7,7	-7,5	-7,4	-7,6	-7,7	-7,5	-7,4

FCFS + future demand, optimized using Monte Carlo overlap estimates - Realized Performance (in 1000's)

Safety factor	σ Transport	σ Arrival time	0				1				2				3			
			σ On-belt				σ On-belt				σ On-belt				σ On-belt			
			0	2	4	6	0	2	4	6	0	2	4	6	0	2	4	6
0	0	0	-10,5	-10,0	-9,8	-9,5	-10,5	-9,9	-9,7	-9,4	-10,5	-9,9	-9,6	-9,4	-10,5	-9,9	-9,6	-9,3
		2	-9,7	-9,6	-9,3	-9,0	-9,8	-9,6	-9,5	-9,0	-9,7	-9,6	-9,3	-9,1	-9,7	-9,5	-9,4	-9,2
		4	-9,1	-9,1	-8,9	-8,6	-9,1	-9,1	-8,7	-8,6	-9,1	-9,1	-8,8	-8,6	-9,2	-9,1	-8,8	-8,6
		6	-8,6	-8,7	-8,5	-8,5	-8,7	-8,7	-8,5	-8,3	-8,7	-8,5	-8,5	-8,2	-8,6	-8,6	-8,4	-8,2
	2	0	-10,0	-9,9	-9,4	-9,4	-10,1	-9,9	-9,5	-9,3	-10,0	-9,9	-9,5	-9,1	-9,9	-9,8	-9,5	-9,1
		2	-9,7	-9,5	-9,3	-9,0	-9,6	-9,5	-9,3	-9,0	-9,7	-9,5	-9,3	-9,1	-9,6	-9,5	-9,3	-8,9
		4	-9,0	-9,0	-8,7	-8,6	-9,1	-9,0	-8,8	-8,6	-9,1	-9,0	-8,7	-8,5	-9,1	-8,9	-8,7	-8,6
		6	-8,7	-8,6	-8,4	-8,4	-8,5	-8,7	-8,5	-8,3	-8,6	-8,4	-8,3	-8,2	-8,6	-8,6	-8,4	-8,3
	4	0	-9,5	-9,5	-9,4	-8,8	-9,6	-9,5	-9,3	-9,0	-9,5	-9,5	-9,3	-8,9	-9,4	-9,5	-9,2	-9,0
		2	-9,2	-9,2	-9,0	-9,1	-9,4	-9,4	-9,1	-8,8	-9,5	-9,2	-9,0	-8,7	-9,4	-9,3	-9,0	-8,8
		4	-9,0	-8,9	-8,6	-8,6	-8,9	-8,8	-8,7	-8,5	-8,9	-8,8	-8,6	-8,5	-8,8	-8,9	-8,7	-8,3
		6	-8,5	-8,5	-8,4	-8,0	-8,7	-8,4	-8,2	-8,1	-8,5	-8,4	-8,3	-8,1	-8,5	-8,4	-8,2	-8,1
	6	0	-8,9	-9,0	-8,9	-8,6	-9,0	-9,1	-8,8	-8,6	-9,1	-9,1	-8,8	-8,7	-9,0	-9,1	-8,8	-8,6
		2	-8,8	-8,8	-8,6	-8,6	-8,9	-8,8	-8,7	-8,7	-8,9	-8,9	-8,7	-8,6	-9,0	-8,8	-8,8	-8,7
		4	-8,6	-8,6	-8,5	-8,4	-8,6	-8,6	-8,5	-8,3	-8,7	-8,6	-8,4	-8,3	-8,7	-8,7	-8,4	-8,3
		6	-8,4	-8,3	-8,2	-7,8	-8,3	-8,2	-8,0	-7,9	-8,3	-8,4	-8,1	-7,9	-8,3	-8,3	-8,0	-7,7

Appendix J: Best worst-case results

In Table 48 we show the solution approaches of which the worst replication (out of five) gave the best results. Entries in red mean that the solution approach or the overlap estimation has changed with respect to the best average replication results, shown in Table 31. There is little difference between the best average performing approach and the approach that result in the best worst-case.

Table 48: Solution approaches with the best worst replication for different levels of stochasticity

Arrival time			On-belt			
			0	2	4	6
0	Transport	0	MIP_Basic (0)	SA_Basic (4)	SA_MC (2)	SA_MC (0)
		2	SA_Basic (4)	SA_Basic (4)	SA_MC (0)	SA_MC (2)
		4	SA_MC (2)	SA_MC (2)	SA_MC (4)	SA_MC (4)
		6	SA_MC (2)	SA_MC (2)	SA_MC (0)	SA_MC (2)
1	Transport	0	SA_Basic (2)	SA_Basic (4)	SA_MC (0)	SA_Basic (6)
		2	SA_MC (2)	SA_MC (2)	SA_MC (0)	SA_MC (2)
		4	SA_MC (0)	SA_MC (4)	SA_MC (4)	SA_MC (2)
		6	SA_MC (0)	SA_MC (4)	SA_MC (6)	SA_MC (4)
2	Transport	0	FCFS+fut_MC (0)	SA_Basic (4)	SA_Basic (6)	SA_MC (0)
		2	SA_Basic (6)	SA_Basic (6)	SA_MC (0)	SA_MC (2)
		4	SA_MC (2)	SA_MC (4)	SA_MC (2)	SA_MC (4)
		6	SA_MC (2)	SA_MC (2)	SA_MC (4)	Greedy_MC (2)
3	Transport	0	FCFS+fut_MC (0)	FCFS+fut_MC (0)	FCFS_MC (0)	FCFS_MC (0)
		2	FCFS+fut_MC (0)	FCFS+fut_MC (0)	FCFS+fut_MC (0)	FCFS+fut_MC (0)
		4	SA_MC (4)	SA_MC (2)	SA_MC (2)	SA_MC (0)
		6	SA_MC (4)	SA_MC (2)	SA_MC (6)	SA_MC (2)

Solution approach	Overlap estimation
MIP	basic
SA	basic
SA	MC simulations
FCFS	MC simulations
FCFS+future dem.	MC simulations

Table 49: The highest (worst) replication score for the approach that created the best worst-case solution

Arrival time		Transport	On-belt 0	On-belt 2	On-belt 4	On-belt 6
	0	0	-11,398	-10,697	-10,186	-9,954
		2	-10,347	-10,085	-9,790	-9,334
		4	-9,547	-9,456	-9,105	-8,782
		6	-9,060	-8,850	-8,735	-8,414
	1	0	-10,844	-10,439	-9,976	-9,565
		2	-10,195	-9,962	-9,626	-9,234
		4	-9,479	-9,369	-9,080	-8,910
		6	-8,851	-8,800	-8,715	-8,444
	2	0	-10,524	-10,086	-9,667	-9,353
		2	-9,952	-9,681	-9,458	-9,289
		4	-9,409	-9,247	-8,941	-8,769
		6	-8,838	-8,768	-8,709	-8,328
	3	0	-10,504	-9,860	-9,464	-9,319
		2	-9,734	-9,449	-9,219	-9,173
		4	-9,184	-8,987	-8,895	-8,597
		6	-8,588	-8,520	-8,433	-8,258

Table 50: The gap between the average value of the best-average solutions and the worst value of the best-worst solutions

Arrival time		Transport	On-belt 0	On-belt 2	On-belt 4	On-belt 6
	0	0	0	135	195	171
		2	65	96	79	207
		4	102	194	193	313
		6	131	149	156	267
	1	0	58	84	169	205
		2	78	75	234	232
		4	148	155	189	114
		6	212	145	137	140
	2	0	73	94	156	172
		2	86	142	179	98
		4	82	157	118	118
		6	156	125	173	268
	3	0	85	104	191	51
		2	167	63	149	70
		4	144	194	53	71
		6	170	202	143	196

Appendix K: Linear Regression Models

Table 51: Linear Regression without a safety factor

		Intercept	Arrival	Transport	OnBelt	Arrival* Transport	Arrival* OnBelt	Transport* OnBelt	Adj. R^2
basic overlap estimation	MIP	-11,077	506	525	221	-65	-20	-25	0.98
	SA	-11,011	499	492	218	-62	-25	-20	0.98
	Greedy	-10,749	517	525	231	-64	-24	-23	0.97
	FCFS	-10,175	0	343	173	0	0	-17	0.99
	FCFS+	-10,491	117	381	182	-12	0	-20	0.98
MC overlap estimation	MIP	-10,636	302	281	145	-32	-13	-7	0.97
	SA	-11,117	354	329	172	-41	-13	-11	0.98
	Greedy	-10,787	337	318	166	-38	-13	-13	0.97
	FCFS	-10,139	0	290	143	0	0	-13	0.97
	FCFS+	-10,373	0	285	166	0	0	-19	0.98

Table 52: Linear Regression with a safety factor of two minutes

		Intersect	Arrival	Transport	OnBelt	Arrival* Transport	Arrival* OnBelt	Transport* OnBelt	Adj. R^2
basic overlap estimation	MIP	-11,019	499	519	220	-67	-20	-23	0.98
	SA	-11,264	448	483	223	-56	-20	-20	0.98
	Greedy	-10,659	354	429	214	-34	-15	-19	0.97
	FCFS	-9,687	0	281	143	0	0	-13	0.97
	FCFS+	-10,231	151	328	159	-14	0	-14	0.97
MC overlap estimation	MIP	-10,917	338	326	182	-33	-14	-14	0.98
	SA	-10,822	246	276	141	-28	0	-10	0.97
	Greedy	-10,529	288	270	148	-27	-11	-11	0.97
	FCFS	-9,667	0	218	106	0	0	-8	0.97
	FCFS+	-10,094	0	241	138	0	0	-14	0.97

Table 53: Linear Regression with a safety factor of four minutes

		Intersect	Arrival	Transport	OnBelt	Arrival* Transport	Arrival* OnBelt	Transport* OnBelt	Adj. R^2
basic overlap estimation	MIP	-11,044	511	521	221	-64	-25	-23	0.98
	SA	-11,291	399	445	210	-48	-19	-18	0.98
	Greedy	-10,,034	205	289	117	-20	0	0	0.94
	FCFS	-9,313	0	236	116	0	0	-7	0.97
	FCFS+	-9,718	83	249	134	0	0	-11	0.95
MC overlap estimation	MIP	-10,785	300	339	173	-35	0	-10	0.97
	SA	-10,383	162	206	115	-13	0	-6	0.94
	Greedy	-9,982	163	193	90	-15	0	0	0.94
	FCFS	-9,310	0	157	83	0	0	0	0.91
	FCFS+	-9,615	0	166	83	0	0	0	0.94

Table 54: Linear Regression with a safety factor of six minutes

		Intersect	Arrival	Transport	OnBelt	Arrival* Transport	Arrival* OnBelt	Transport* OnBelt	Adj. R^2
basic overlap estimation	MIP	-11,011	485	512	217	-62	-15	-23	0.98
	SA	-11,124	294	404	173	-36	0	-18	0.98
	Greedy	-9,580	116	211	111	0	0	0	0.92
	FCFS	-8,980	0	219	108	0	0	-8	0.94
	FCFS+	-9,241	109	189	112	0	0	-8	0.91
MC overlap estimation	MIP	-10,413	272	286	172	-19	0	0	0.97
	SA	-9,815	88	121	67	0	0	0	0.89
	Greedy	-9,506	92	141	76	0	0	0	0.90
	FCFS	-8,969	0	146	64	0	0	0	0.93
	FCFS+	-9,137	0	125	63	0	0	0	0.90

Appendix L: TPR and FDR

In Section 2.1.5. we have discussed the problem that in the current situation there are many flight pairs that are will overlap, but are not considered to overlap in the optimization phase due to their estimates. Besides, there was the problem that there are a lot of flight pairs that are considered to overlap, while in reality they will not end up overlapping. In this Section we will show and discuss the relationship between the stochasticity in the estimators, the overlap estimation approach, and the number of flight pairs correctly classified.

We will look at two statistics when making this comparison: The True Positive Ratio (TPR) and the False Discovery Ratio (FDR). The TPR is the ratio of flight pairs that were expected to overlap and ended up overlapping, to the total number of flights that ended up overlapping. A TPR of 1 means that all flight pairs that ended up overlapping, were expected to overlap. The FDR is the ratio of flight pairs that were expected to overlap and did not end up overlapping, to the total number of flight pairs that were expected to overlap. A FDR of 0 means that all flight pairs that were expected to overlap, ended up overlapping.

In Section 2.15. we saw that in the analyzed week of data, 5330 flight pairs were expected to overlap. Out of these 5330 flight pairs, only 2527 ended up overlapping. Besides these 2527 flight pairs that were expected to overlap and ended up overlapping, an additional 1716 flight pairs ended up overlapping while they were not expected to do so. For this case, we find:

$$TPR = \frac{2527}{2527 + 1716} = 0.60$$

$$FDR = \frac{5330 - 2527}{5330} = 0.53$$

The TPRs and FDRs of the basic overlap estimation method under the different levels of stochasticity are shown in Table 55 and Table 56, respectively. The TPRs and FDRs of the MC overlap estimation method are shown in Table 57 and Table 58.

Table 55: True Positive Rate of basic overlap estimation method for combinations of stochasticity and safety factor

Safety factor	σ Transport		σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	0	1,00	0,93	0,87	0,82	0,94	0,90	0,86	0,81	0,88	0,86	0,81	0,79	0,82	0,80	0,77	0,74
		2	0,87	0,85	0,81	0,78	0,85	0,83	0,79	0,77	0,82	0,80	0,77	0,74	0,78	0,76	0,73	0,70
		4	0,76	0,74	0,71	0,69	0,75	0,73	0,71	0,68	0,73	0,72	0,70	0,67	0,70	0,69	0,67	0,65
		6	0,68	0,66	0,65	0,62	0,67	0,66	0,64	0,62	0,65	0,65	0,63	0,60	0,63	0,63	0,61	0,59
	2	0	1,00	0,98	0,92	0,88	0,99	0,97	0,92	0,88	0,96	0,94	0,90	0,85	0,91	0,89	0,85	0,83
		2	0,95	0,93	0,89	0,85	0,94	0,92	0,88	0,84	0,91	0,89	0,86	0,82	0,87	0,85	0,83	0,78
		4	0,85	0,84	0,81	0,77	0,84	0,83	0,80	0,77	0,82	0,81	0,78	0,76	0,79	0,79	0,76	0,73
		6	0,77	0,75	0,74	0,69	0,76	0,75	0,73	0,70	0,75	0,74	0,72	0,69	0,72	0,72	0,70	0,67
	4	0	1,00	0,99	0,95	0,91	1,00	0,99	0,95	0,90	0,99	0,97	0,93	0,90	0,96	0,94	0,91	0,88
		2	0,99	0,97	0,93	0,90	0,98	0,96	0,93	0,89	0,96	0,95	0,91	0,88	0,93	0,92	0,89	0,86
		4	0,91	0,90	0,87	0,84	0,91	0,89	0,86	0,83	0,89	0,88	0,85	0,82	0,87	0,86	0,83	0,80
		6	0,84	0,83	0,80	0,78	0,83	0,82	0,79	0,77	0,81	0,81	0,79	0,76	0,80	0,79	0,77	0,75
	6	0	1,00	1,00	0,97	0,93	1,00	1,00	0,97	0,92	1,00	0,99	0,96	0,91	0,98	0,97	0,94	0,91
		2	1,00	0,99	0,96	0,92	0,99	0,98	0,95	0,92	0,98	0,97	0,94	0,91	0,96	0,95	0,93	0,89
		4	0,95	0,94	0,92	0,89	0,95	0,94	0,91	0,89	0,94	0,92	0,90	0,87	0,91	0,91	0,88	0,85
		6	0,88	0,88	0,86	0,82	0,88	0,87	0,85	0,82	0,87	0,86	0,84	0,82	0,85	0,84	0,83	0,80

Table 56: False Discovery Rate of basic overlap estimation method for combinations of stochasticity and safety factor

Safety factor	σ Transport		σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	0	0,00	0,07	0,13	0,17	0,06	0,10	0,14	0,19	0,12	0,14	0,18	0,22	0,18	0,19	0,23	0,26
		2	0,13	0,16	0,19	0,22	0,15	0,16	0,20	0,24	0,18	0,19	0,23	0,26	0,23	0,24	0,26	0,29
		4	0,24	0,26	0,28	0,31	0,25	0,27	0,29	0,32	0,27	0,28	0,30	0,33	0,30	0,31	0,33	0,35
		6	0,32	0,34	0,35	0,37	0,33	0,34	0,36	0,38	0,35	0,35	0,37	0,40	0,37	0,37	0,39	0,41
	2	0	0,18	0,20	0,25	0,28	0,19	0,21	0,25	0,28	0,21	0,23	0,26	0,29	0,25	0,27	0,30	0,32
		2	0,22	0,24	0,27	0,30	0,23	0,25	0,28	0,31	0,25	0,26	0,30	0,32	0,29	0,30	0,32	0,35
		4	0,30	0,31	0,34	0,37	0,31	0,32	0,35	0,37	0,33	0,34	0,36	0,38	0,35	0,35	0,37	0,40
		6	0,36	0,38	0,40	0,42	0,38	0,38	0,40	0,42	0,38	0,39	0,41	0,43	0,40	0,41	0,43	0,44
	4	0	0,30	0,31	0,34	0,36	0,30	0,31	0,33	0,37	0,31	0,32	0,35	0,38	0,33	0,34	0,36	0,38
		2	0,31	0,32	0,35	0,38	0,32	0,33	0,35	0,38	0,33	0,34	0,36	0,39	0,35	0,36	0,38	0,40
		4	0,37	0,37	0,39	0,42	0,37	0,38	0,39	0,42	0,38	0,39	0,41	0,42	0,40	0,40	0,42	0,44
		6	0,42	0,42	0,44	0,45	0,42	0,42	0,44	0,46	0,43	0,43	0,45	0,47	0,44	0,44	0,45	0,48
	6	0	0,39	0,39	0,41	0,44	0,39	0,39	0,41	0,44	0,39	0,40	0,41	0,44	0,40	0,41	0,42	0,44
		2	0,40	0,40	0,42	0,44	0,40	0,40	0,41	0,44	0,40	0,40	0,42	0,45	0,41	0,42	0,43	0,46
		4	0,42	0,43	0,44	0,46	0,43	0,43	0,45	0,47	0,43	0,44	0,45	0,47	0,45	0,45	0,46	0,48
		6	0,46	0,46	0,48	0,49	0,46	0,47	0,48	0,50	0,47	0,47	0,49	0,50	0,48	0,49	0,49	0,51

Table 57: True Positive Rate of MC overlap estimation methods for combinations of stochasticity and safety factor

Safety factor	σ Transport		σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	0	1,00	0,95	0,94	0,92	0,97	0,96	0,93	0,92	0,95	0,95	0,93	0,91	0,95	0,95	0,94	0,92
		2	0,96	0,95	0,94	0,91	0,95	0,95	0,94	0,91	0,95	0,95	0,94	0,92	0,95	0,95	0,94	0,92
		4	0,95	0,95	0,93	0,91	0,94	0,94	0,93	0,92	0,94	0,95	0,93	0,92	0,94	0,93	0,93	0,92
		6	0,94	0,93	0,93	0,91	0,94	0,93	0,92	0,91	0,93	0,93	0,92	0,92	0,93	0,93	0,92	0,91
	2	0	1,00	0,98	0,97	0,94	0,99	0,98	0,96	0,94	0,99	0,98	0,96	0,95	0,98	0,98	0,96	0,94
		2	0,99	0,98	0,96	0,94	0,98	0,98	0,96	0,94	0,98	0,97	0,96	0,95	0,97	0,97	0,96	0,94
		4	0,97	0,97	0,96	0,95	0,97	0,97	0,96	0,95	0,97	0,97	0,96	0,94	0,97	0,97	0,96	0,95
		6	0,96	0,97	0,95	0,95	0,96	0,96	0,96	0,94	0,97	0,96	0,96	0,95	0,96	0,96	0,96	0,95
	4	0	1,00	0,99	0,97	0,95	1,00	0,99	0,98	0,95	0,99	0,99	0,97	0,96	0,99	0,99	0,97	0,96
		2	0,99	0,99	0,97	0,96	0,99	0,99	0,97	0,95	0,99	0,99	0,98	0,96	0,99	0,99	0,97	0,95
		4	0,99	0,98	0,97	0,96	0,99	0,98	0,97	0,96	0,98	0,98	0,97	0,96	0,98	0,98	0,97	0,96
		6	0,98	0,98	0,97	0,95	0,98	0,98	0,97	0,96	0,98	0,98	0,97	0,96	0,98	0,97	0,97	0,96
	6	0	1,00	0,99	0,98	0,97	1,00	0,99	0,98	0,97	1,00	0,99	0,98	0,97	0,99	0,99	0,98	0,97
		2	1,00	0,99	0,98	0,97	0,99	0,99	0,98	0,97	0,99	0,99	0,98	0,97	0,99	0,99	0,98	0,97
		4	0,99	0,99	0,98	0,97	0,99	0,99	0,98	0,97	0,99	0,99	0,98	0,97	0,99	0,99	0,98	0,96
		6	0,99	0,99	0,98	0,97	0,99	0,99	0,98	0,97	0,99	0,99	0,98	0,97	0,99	0,99	0,98	0,97

Table 58: False Discovery Rate of MC overlap estimation methods for combination of stochasticity and safety factor

Safety factor	σ Transport		σ Arrival time 0 / σ On-belt 0	2	4	6	σ Arrival time 1 / σ On-belt 0	2	4	6	σ Arrival time 2 / σ On-belt 0	2	4	6	σ Arrival time 3 / σ On-belt 0	2	4	6
0	0	0	0,00	0,13	0,28	0,38	0,12	0,19	0,30	0,40	0,21	0,27	0,36	0,43	0,33	0,36	0,41	0,47
		2	0,21	0,27	0,36	0,43	0,23	0,28	0,37	0,43	0,30	0,34	0,41	0,46	0,38	0,40	0,44	0,50
		4	0,37	0,40	0,45	0,49	0,39	0,41	0,45	0,51	0,42	0,44	0,47	0,52	0,46	0,47	0,50	0,54
		6	0,47	0,48	0,51	0,54	0,48	0,49	0,52	0,55	0,49	0,51	0,53	0,56	0,52	0,52	0,55	0,58
	2	0	0,18	0,25	0,39	0,47	0,25	0,30	0,40	0,47	0,31	0,36	0,43	0,50	0,40	0,42	0,47	0,53
		2	0,31	0,36	0,44	0,50	0,32	0,37	0,44	0,51	0,38	0,40	0,47	0,53	0,44	0,46	0,50	0,55
		4	0,45	0,47	0,51	0,55	0,45	0,48	0,51	0,56	0,48	0,49	0,53	0,57	0,52	0,53	0,56	0,59
		6	0,52	0,54	0,57	0,60	0,53	0,54	0,57	0,60	0,54	0,55	0,58	0,61	0,57	0,57	0,59	0,62
	4	0	0,30	0,35	0,46	0,53	0,35	0,38	0,47	0,54	0,39	0,43	0,49	0,55	0,47	0,49	0,53	0,58
		2	0,39	0,43	0,50	0,56	0,40	0,43	0,51	0,56	0,45	0,47	0,52	0,57	0,50	0,52	0,55	0,59
		4	0,50	0,52	0,56	0,59	0,51	0,52	0,56	0,60	0,53	0,54	0,57	0,61	0,56	0,57	0,59	0,62
		6	0,57	0,58	0,60	0,63	0,57	0,58	0,60	0,64	0,59	0,59	0,62	0,64	0,60	0,61	0,63	0,65
	6	0	0,39	0,43	0,51	0,58	0,43	0,46	0,52	0,58	0,46	0,49	0,54	0,60	0,52	0,53	0,57	0,61
		2	0,46	0,50	0,54	0,60	0,47	0,50	0,55	0,60	0,51	0,52	0,57	0,61	0,54	0,56	0,59	0,63
		4	0,54	0,56	0,59	0,63	0,55	0,56	0,60	0,63	0,57	0,58	0,61	0,64	0,59	0,60	0,62	0,65
		6	0,60	0,61	0,63	0,67	0,61	0,62	0,64	0,67	0,62	0,62	0,65	0,67	0,63	0,64	0,66	0,68

From the tables it becomes clear that the MC overlap estimation method is better able to identify possible overlap between two flights. At the maximum assessed level of stochasticity without using safety times, still more than 90% of the overlapping flight pairs were identified correctly. If a safety time of 6 minutes is incorporated this grows to 96%. For the basic overlap estimates, this is 59% and 80%, respectively. The MC overlap estimation without safety factor is better able to identify overlapping flight pairs than the basic overlap estimation with a safety factor of six minutes.

On the other hand, the FDR for the maximum levels of stochasticity for MC overlap estimates are 0.58 and 0.68 without a safety time and with a safety time of 6 minutes, respectively. A FDR of 0.68 means that for every flight pair correctly identified as overlapping, an additional two flights pairs are wrongfully identified as overlapping. For the basic overlap estimates these FDRs are 0.4 and 0.51. And thus a lot less flights are wrongfully expected to overlap.