Optimization of Transfer Baggage Handling in a Major Transit Airport

Barth, Torben C.; Holm, Janus Timler; Larsen, Jakob Lindorff; Pisinger, David

Published in:
SN Operations Research Forum

Link to article, DOI:
10.1007/s43069-021-00058-z

Publication date:
2021

Document Version
Peer reviewed version

Optimization of transfer baggage handling in a major transit airport

December 20, 2021

Torben C. Barth[1,2] **Janus Timler Holm**[2] **Jakob Lindorff Larsen**[2] **David Pisinger**[2]

[1]*Information and Telecommunication Services, Fraport AG*
[2]*DTU Management, Technical University of Denmark*

Abstract

We consider the handling of baggage from passengers changing aircraft at an airport. The *transfer baggage problem* is to assign the bags from each arriving aircraft to an infeed area, from where a network of conveyor belts will bring them to the corresponding outbound flight. The main objective is to minimize the number of missed bags, but in order to reach this goal, we introduce some auxiliary objectives that minimize the transportation time of bags, while ensuring robustness and avoiding overload of the handling facilities.

We first present a static mixed integer programming model for the transfer baggage problem. However, the transfer baggage process is subject to uncertainty related to aircraft arrival time, transportation time from aircraft to baggage handling facility, and capacity use in the baggage handling system. In order to handle this uncertainty a stochastic model is developed, optimizing over a finite set of scenarios. Although the model in theory should lead to more balanced and stable decisions, it suffers of long solution times. Therefore, a semi-stochastic model is presented, where short-term decisions are stochastic, while long-term decisions are deterministic.

Computational experiments on real-life data from a major European hub airport are reported, demonstrating that each of the three models has its advantages. In particular the semi-stochastic model shows promising results both with respect to robustness and solution times.

Keywords: *Airport baggage handling*, *Optimization under uncertainty*, *Robust optimization*, *Stochastic programming*

1 Introduction

Baggage handling is an important process during the ground time of an aircraft at an airport. It runs from the arrival until the departure of the aircraft and is a time critical bottleneck during airport operations. Baggage handling consists of many subtasks and can be distinguished in three main processes. The first process is the *inbound baggage* handling which is concerned with the handling of the bags from passengers arriving at their final destination. The *transfer baggage* process deals with handling of bags from arriving passengers in transit to another flight. The

outbound baggage handling combines the baggage flow from transfer passengers and the local check-in desks, in delivering bags to departing aircraft.

The transfer baggage process is particularly important for the success of large hub airports with many transfer passengers, since the guaranteed connection time for passengers and the corresponding luggage is a competition factor between hub airports. Baggage handling quality is mainly measured as number of bags missing connecting transfers. For convenience we denote bags missing their connecting transfer as *missed bags*. Every missed bag implies loss of goodwill and can lead to monetary compensations to the airlines. Moreover, missed bags demand additional handling processes with the corresponding labor cost.

The transfer baggage process is significantly influenced by the choice of the *infeed station* in the airport infrastructure. The infeed stations are often grouped resources in handling halls in the basement of the terminal building. A group of infeed stations is referred to as an *infeed area* in the following. At the infrastructure level the bags are then delivered via the *baggage handling system* (BHS) to the handling facilities of the outbound flights. The BHS consists at most airports of a connected conveyor belt network.

The complexity of the transfer baggage process depends on the airport layout and the available capacity. For example the handling capacities at the infeed areas are limited and not equally distributed over the airport. The location of the aircraft parking positions, infeed areas and handling facilities of the outbound flights define the structure of the problem.

In this paper we study the operational transfer baggage process at Frankfurt Airport where more than half of the passengers are transfer passengers. In the considered setup, one dispatcher is responsible to control the transfer baggage process and to select infeed areas for handling the bags of the arriving aircraft. The dispatcher has to decide about several flights per minutes in rush hours. The choice of the infeed area needs to be decided before the arrival of the aircraft. Due to the high degree of uncertainty of the input data, e.g. position changes and delays, the decision should not be fixed before the final arrival of the aircraft. The whole setup and the best choices for all flights in a certain time interval can change within minutes. Therefore, the decision support tool needs to provide solutions quickly during daily operations.

Literature: For a general introduction to baggage handling at airports we refer to the comprehensive theses by Barth [4] and Frey [13]. Chapter 7 in the book [1] also gives a brief introduction to baggage handling in airports.

Only a few studies have considered *transfer baggage*. Barth et al. [5] present a general scheduling framework for the different baggage handling problems at airports. The aim of the framework is to find a solution which combines the different problems and provides the best overall solution in terms of efficiency and quality.

Some preliminary results on transfer baggage handling at Frankfurt Airport were presented in Barth [2] and Barth and Franz [6]. This paper presents the underlying mathematical model in details and shows important approaches to handle the uncertainty in the dynamic environment at airports.

A similar problem was studied for Munich Airport by Kiermaier and Kolisch [17], who developed a deterministic model for a full day. The problem is formulated as an assignment problem, where baggage vehicles are to assigned to infeed areas, subject to some capacity constraints on the infeed areas. A given infeed area may be overloaded, but only β overload periods within α

consecutive periods are allowed. The MIP model is solved with CPLEX, but running times are too big, so a genetic algorithm is developed that is able to solve the problem in 20 seconds with an optimality gap of 3-7%.

Clausen and Pisinger [10] develop a solution algorithm for the connected problem of handling critical transfer baggage. The problem is to plan the routes for the vehicles such that each bag is either delivered directly to the flight, or to the infeed area, respecting the time windows of each bag. The objective is to minimize the number of bags which do not catch the flight. The problem is formulated as a dynamic Vehicle Routing Problem (VRP) with time windows, and solved using a parameterized greedy algorithm with linear penalty functions. The algorithm is run each time a vehicle enters the dispatch hall, and calculates a set of bags for the next route.

The two case studies [9] and [16] present lessons learned during the development and introduction of optimization solutions for baggage handling problems at airports. In particular they illustrate the importance of respecting various real-world requirements in the developed solutions.

Contribution: The purpose of this paper is to introduce a general model for the control of transfer baggage handling in daily operations. The model deals with a rolling-horizon problem covering 150 minutes around the current time.

The baggage process has to deal with uncertainty related to aircraft arrival time, uncertainty in the transportation time from aircraft to baggage handling facility, and finally uncertainties in the available capacity of the infeed area. In order to handle these uncertainties a stochastic model is presented, optimizing over various scenarios for on-block time, transportation time, and capacity use of the BHS. Although the model in theory should lead to more balanced and stable decisions, it suffers of long solution times, and hence only a limited number of scenarios can be investigated in reasonable time. In order to overcome this problem, a semi-stochastic model is presented, where short-term decisions are stochastic, while long-term decisions are deterministic. Computational experiments on real-life data from Frankfurt Airport are reported, showing that each of the models has it advantages with respect to running time or solution quality. In particular, the semi-stochastic model shows promising results both with respect to running times and robustness. In this context, robustness means the ability to withstand variation in the uncertain data, without missing bags.

The semi-stochastic model appears to perform well for rolling-horizon optimization problems, where short-term decision are more important than the long-term decisions. The semi-stochastic model may be a good alternative to scenario reduction techniques [11, 12, 14] frequently used in stochastic programming.

Case study: To test and evaluate the developed models, we use datasets for a total of 9 days, from the daily operation at Frankfurt Airport. The model dealing with the trasfer baggage problem at Frankfurt Airport is solved approximately every 2nd minute. In this process, a data dump for each run is generated. The data dump holds information about the incoming/outgoing flights, expected parking positions etc. Much of the information is uncertain and changes continuously during the day. The data dump captures the exact information available at the time the model is run.

The nine data sets in Table 1 are based on data for the days 16/9-2013 and 9/1-2014 to 16/1-2014. The datasets represent a decent spread of complexity reflected by the number of bags

Name	Date	Weekday	Bags
D1	16/09-2013	Monday	46 028
D2	09/01-2014	Thursday	34 857
D3	10/01-2014	Friday	37 877
D4	11/01-2014	Saturday	38 826
D5	12/01-2014	Sunday	38 247
D6	13/01-2014	Monday	35 630
D7	14/01-2014	Tuesday	32 440
D8	15/01-2014	Wednesday	31 070
D9	16/01-2014	Thursday	30 730

Table 1: The nine provided instances, their reference name and number of transfer bags

and thereby assignments. For each day the models will be evaluated based on an entire day of operation. This is done by solving the rolling horizon model every 2nd minute, using the data that was available at the given time. This recreates how the model is used in Frankfurt Airport.

Outline: The paper is organized as follows: In Section 2 the problem of handling transfer baggage is introduced more formally, and important steps of the solution process are presented. Section 3 describes the mathematical model for the transfer baggage problem. In Section 4 the use of the model in the dynamic environment at an airport is explained. Furthermore, the uncertainty of the data and the modeling of the robustness are studied. Section 5 presents the stochastic model that better captures the uncertainty in arrival times. Finally, Section 6 describes the semi-stochastic model that combines the strength of the stochastic model, with the fast solution times of the deterministic model. Extensive computational experiments with all three models are presented in Section 7. The paper is concluded in Section 8 with a discussion of the achieved results. A list of all symbols can be found in Appendix A.

2 Transfer Baggage Problem

The transfer baggage handling process includes various activities between landing of an incoming flight and departure of the connecting flight, as illustrated in Figure 1. These activities form the basis of the studied problem.

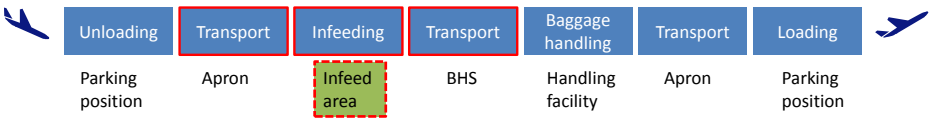

Figure 1: Process chain of the transfer baggage handling at an airport. The red marked activities are influenced by the choice of the infeed area.

After the arrival of the aircraft at the parking position, the unloading of the bags from the aircraft starts. In the next step the bags are transported with vehicles to the infeed area at the

airport infrastructure. This transportation happens in the part of the airport called the *apron*, the area where aircraft are parked and handled. Bags from transfer passengers and inbound passengers are separated during the unloading process and are transported to different infeed areas. It can happen that inbound and transfer bags are transported in one trip. In this case the vehicles will deliver the bags of one trip to a inbound and a transfer infeed area. At the infeed area, the bag-tags of the transfer bags are scanned and the destination of each bag is determined. The bags are then transported via the BHS to the handling facilities of the outbound flights. At the handling facilities of the outbound flights, the bags are loaded onto containers or dollies (for simplicity, we refer to both as *containers*). Afterwards, the bags are transported to the parking position of the departure aircraft. Finally, the bags are loaded into the aircraft. The process of loading the last bags into the aircraft has to be finished in due time before the expected departure time.

At Frankfurt Airport each truck can tow at most two containers. The transportation of paired containers is called a *trip* in the following. For each trip, an assignment to an infeed area has to be chosen. This assignment has a huge impact on the process time of the corresponding bags and therefore determines whether or not the individual bags will arrive at the transfer flight in time. Figure 2 illustrates the problem of choosing infeed areas for an arriving aircraft with a number of corresponding connecting flights.

The *Transfer Baggage Problem* (TBP) is the problem of assigning each trip containing transfer baggage to one of the available infeed areas. The overall goal is to use the existing capacity in the best possible way in order to optimize a number of different objective criteria that all aim at reducing the number of missed bags. It is assumed that a sufficient amount of vehicles is available, hence route planning of the trips can be ignored.

For every transfer bag, the process time depends on the position of the arriving aircraft, the selected infeed area and the handling facility from the outbound flight. Each transfer bag has different available time intervals for the connection process. It is therefore important to find a good assignment of the different trips to the handling facilities in such a way that as many bags as possible make their connection.

The infeed areas have limited capacities in respect to the number of handled bags per time interval. An important observation is that the bags are handled according to a first-in-first-out strategy. In practical terms, this means that it is not possible to prioritize critical trips or bags if the capacity limit is exceeded at an infeed area.

For each individual bag, the available transfer time is calculated as the difference between the estimated *on-block time* (EON)[1] of the inbound flight, and the estimated off-block time of the outbound flight. If the transfer time does not exceed the available time, the bag is *in time*, and the bag will normally catch the outbound flight. Otherwise the bag is categorized as *late* and may result in a lost bag.

Although the main objective is to minimize lost bags, we introduce four auxiliary objectives that help reaching the goal:

- Minimize the number of late bags.

[1]The status *on-block* is defined as official/real arrival at the parking position and *off-blocks* as official/real departure from parking position.

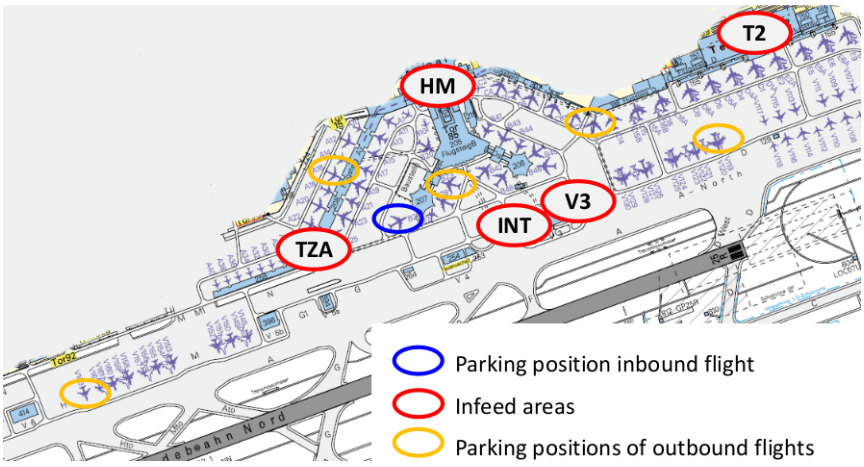

Figure 2: Example of an arriving aircraft (blue ring) with corresponding connecting flights (yellow rings) and the possible infeed areas (red rings).

- Minimize transportation time from the aircraft to infeed area at the apron.
- Minimize transportation time from infeed area to handling facility of the outbound flight through the BHS.
- Maximize capacity buffer for unforeseen events at the infeed areas.

We use a weighted linear combination of these auxiliary objectives as our goal function. The first three auxiliary objectives are somehow independent problems for each trip, and hence for each assignment of a trip to an infeed area, the contribution to the objectives can be pre-computed. The last objective depends on the combination of assignments of trips to infeed areas from different flights. To optimize this objective (in combination with the three previous) it is necessary to develop an optimization model covering all flights in the time horizon T. We use a one hour time horizon, since Barth [4] in a computational study showed that high quality solutions can be obtained with this time frame. The expected number of missed bags depend on the choice of infeed area, as this affects the transfer time of each bag. The calculation of the transfer time has to respect the whole transfer process chain for every single bag in a trip. Summarized, the choice of the infeed area (marked with red ovals in Figure 2) has an impact on the following processes:

- Transport time from the aircraft to the infeed area: If an area close to the parking position of the inbound aircraft is chosen, the distance and hence transport time is short.
- Waiting time at infeed area: If the infeed area is operating at its capacity limit, the bags must wait before they can be loaded into the BHS.
- Handling time at infeed area: The capacity or handling speed at each infeed area varies due to the preallocated heterogeneous resources. This influences the handling time of the bags.

- Transport time from infeed area to outbound handling facility: When the bags travel through the BHS, the transport time varies depending on the distance from the infeed area to the handling facility. Transportation on apron with a vehicle (first step of transportation) is generally faster than transportation through the BHS (second step), so a proper trade-off must be made.

The handling facility of the outbound flight is selected at least one hour, and frequently several hours, before the departure of the aircraft, however changes may appear, and the implied uncertainties are reflected in our data analysis.

2.1 General modeling

The mathematical model can be seen as a generalized assignment problem over a timespan. The timespan considered by the model is divided into time-steps based on a predefined step length. The set of all time-steps is denoted by T. This time discretization only influences the capacity use at the infeed areas.

The bags from an flight either arrive in containers or in bulk compartment, where the latter needs to be unloaded into dollies. There is no significant difference in handling times or process characteristics between container or bulk bags. For simplicity, we refer to both as *containers*. Each inbound flight holds information about its carried containers as well as information about how long time it will take before each of the containers is unloaded. The containers are grouped together, in groups of 1-2 containers, which is done according to some predefined rules depending on the aircraft type. These groupings are called *trips*. The *set of all trips* is denoted B. Thus a trip defines the containers that are transported together, but it is up to the model to choose which infeed area a trip is transported to. Other airports may use a different grouping, but this does not affect the general solution approach.

For each trip we generate possible *assignments* to the infeed areas I. In principle all trips can go to all infeed areas $i \in I$. However, if an infeed area i is closed in a given time interval or the capacity at the infeed area is insufficient, then no possible assignment will be generated between a trip and the given infeed area. The model then has to choose exactly one assignment for each trip.

The following sets are introduced: Set A, is the set of all feasible assignments of trips $b \in B$ from inbound aircraft to infeed areas $i \in I$. Set $A_b \subseteq A$ is the subset of feasible assignments for a specific trip $b \in B$. Set $A_i \subseteq A$ is the subset of feasible assignments for a specific infeed area $i \in I$. For each of the generated assignments a a number of parameters are calculated as follows.

For each assignment, the *total travel time* (in seconds) through the BHS is denoted u_a^{BHS}. This is the sum of transportation time in the BHS for all bags in assignment $a \in A$. The value is easily calculated by going through all bags in the assignment and extract the distance from the infeed area to the handling facilities of the connecting flights.

For a given bag in the assignment $a \in A$ the *buffer time* t^{buf} can be calculated as the available time minus the total time used to transfer the baggage from the inbound flight to the outbound flight (see the process chain in Figure 1).

Based on the buffer time (in minutes), the *missed probability* u^{missed} can be calculated. In a related study [7] in Frankfurt Airport, a detailed study was made of the likelihood that a baggage is missed as function of its delay. This study used several months of historic data to find that the missed probability can be approximated by the piecewise linear function:

$$u^m = \begin{cases} -0.003 \cdot t^{buf} + 0.09 & t^{buf} > 10.8 \\ -0.015 \cdot t^{buf} + 0.22 & \text{otherwise} \\ -0.035 \cdot t^{buf} + 0.35 & t^{buf} < 6.5 \end{cases} \tag{1}$$

$$u^{missed} = \max(0, \min(u^m, 1)) \tag{2}$$

The function is depicted in Figure 3. For a given assignment $a \in A$ of a trip to an infeed area, the overall missed probability u_a^{missed}, is the sum of missed probabilities u^{missed} for all associated bags. Hence u^{missed} denotes the expected number of missed bags if assignment $a \in A$ is selected.

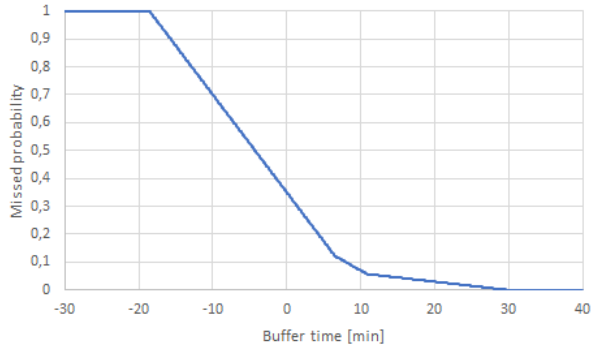

Figure 3: Single baggage *missed probability* dependency on buffer time

The *transportation time* (in seconds) from the inbound flights' parking position to the infeed area is denoted u_a^{drive} for assignment $a \in A$. The value can easily be extracted from a given time distance matrix. An infeed area close to the parking position, will result in smaller driving times.

The number of late bags, when choosing assignment $a \in A$, is denoted u_a^{late}. Formally, it is calculated as the number of bags in the assignment having negative buffer time.

The *capacity use* of each assignment is calculated as a percentage-wise usage of the available capacity in a given time-step. This is denoted by the parameter $u_{a,t}$ for assignment $a \in A$ and time-step $t \in T$. If an assignment means that the trip arrives late in a time-step, only the capacity remaining in the time-step can be used. The remaining bags are handled in the following time-step and so forth. This means that $u_{a,t}$ has to be between 0% and 100%. Values close to 100% are very unlikely, as the containers of a single assignment rarely hold enough bags to use all capacity of a time-step. Though single assignments cannot create overcapacity by themselves, overcapacity can happen when multiple assignments end up in the same infeed area in the same time-step.

3 Deterministic Model

Our first model is a deterministic model that resembles the model currently used in Fraport, excluding specific constraints and conditions for Frankfurt Airport. The model relies on time discretization and has a limited time horizon. Only flights that will land within this time horizon are considered in the optimization. We will use the deterministic model as baseline for our experiments.

The model includes the capacity restrictions at the infeed areas. Since the exact available capacity is hard to determine and the exact arrival times cannot be predicted, a strict limit does not express reality appropriately. The idea is to use soft constraints to avoid a high capacity use, hence saving a buffer for unforeseen events, e.g., break down of infeed stations or unexpected arriving bags. Furthermore, full capacity use over longer periods should be avoided to prevent break downs.

The capacity usage per time step is separated in several steps. Each step models an increasingly higher capacity usage which may then be increasingly penalized by the model. This basically allows us to penalize the capacity usage (and overcapacity usage) as a piece-wise linear function. The steps are defined by the following parameters: Parameter C_i^{des} is the desired maximum capacity use in area $i \in I$. A capacity usage no higher than this level is desired as it possess plenty of buffer capacity to cope with uncertainty. Parameter C_i^{tech} is the technical capacity in area $i \in I$, i.e. the highest capacity possible to maintain. All capacity used additional to this level is regarded as queuing baggage. Parameter C_i^{queue1} is the first degree of overcapacity in area $i \in I$. Overcapacity up to this level is slightly penalized as queuing baggage and will use capacity in the subsequent time period. Finally, parameter C_i^{queue2} is the second degree overcapacity in area $i \in I$. Capacity usage exceeding first degree over capacity and up to this level is additionally penalized as queuing baggage and will also use capacity in the subsequent time period.

Based on these levels the model is allowed to queue up a certain number of bags at the infeed area. However, this queue has a negative influence on the capacity in the subsequent time period. This reflects the physical situation where extraordinary many bags may be served as a result of extra hard working crew and/or left as queuing baggage. The following time is however used to handle the queue and to rest. In case the capacity use exceeds step 2, the excess is captured by a penalty variable which is heavily penalized.

The advantage of this formulation is that it reflects a certain amount of flexibility in the capacity and allows us to encourage the model to maintain some buffer capacity at the infeed areas and avoid using overcapacity.

The model seeks to optimize multiple criteria simultaneously. These are; minimizing transportation time, time spent on conveyor belts, queuing baggage, missed connections and *missed probability*, that all contribute to minimizing the number of missed bags. This is done by optimizing the weighted sum of the different criteria. Though this allows optimizing on many criteria simultaneously, it also gives rise to the challenge of balancing the many criteria against each other across very different units.

3.1 MIP model

In order to formulate the deterministic model for TBP, we define the variables, based on the sets introduces in Section 2.1: Let $x_a \in \{0,1\}$ denote whether assignment $a \in A$ is chosen. For each infeed area $i \in I$ and time-step $t \in T$ let $y_{i,t}$ denote the deterministic workload, $y_{i,t}^{des}$ denote the workload exceeding the desired capacity C_i^{des}, $y_{i,t}^{queue1}$ denote the workload exceeding the technical capacity up to C_i^{queue1}, $y_{i,t}^{queue2}$ denote the workload exceeding the technical capacity between C_i^{queue1} and C_i^{queue2}, and $y_{i,t}^{penalty}$ denote the workload exceeding C_i^{queue2}.

Since the objective function is a weighted sum of several criteria, we introduce the following weights: Let w^{late} be the weight for late bags, w^{missed} for connecting/missed probability, w^{drive} for apron transportation time, w^{BHS} for transportation through the BHS, w^{des} for capacity use, between desired and technical capacity, w^{queue1} for capacity use, between technical and queue1, w^{queue2} for capacity use, between queue1 and queue2, $w^{penalty}$ for capacity use above queue2. All these parameters are set by the airport to reflect their priorities.

Finally, we have the following parameters for each assignment $a \in A$. Let $u_{a,t}$ be the amount of capacity used in timeslot $t \in T$, u_a^{late} be the number of late bags, u_a^{missed} be the total connecting/missed probability of all bags, u_a^{drive} be the transportation time from the parking position to the infeed area (in seconds), and u_a^{BHS} be the total combined transportation time for each bag through the BHS (in seconds). These parameters can all be pre-calculated as a direct consequence of choosing assignment $a \in A$. The calculation of the parameters is based on the reported EON times for the incoming flights.

Furthermore, let $k_{i,t}$ be the number of bags infeed area $i \in I$ can service when working at technical capacity in time-step $t \in T$. If the infeed area is closed at time-step t then $k_{i,t} = 0$. To simplify notation let $T_i' = \{t \in T \mid k_{i,t} > 0\}$ be the set of time-steps where the baggage infeed area $i \in I$ is active.

$$\min \quad \sum_{a\in A}\left(w^{late}u_a^{late}+w^{missed}u_a^{missed}+w^{drive}u_a^{drive}+w^{BHS}u_a^{BHS}\right)x_a + \sum_{i\in I}\sum_{t\in T}k_{i,t}\left(w^{des}y_{i,t}^{des}+w^{queue1}y_{i,t}^{queue1}+w^{queue2}y_{i,t}^{queue2}+w^{penalty}y_{i,t}^{penalty}\right) \tag{3}$$

s.t.

$$\sum_{a\in A_b}x_a = 1 \qquad b\in B \tag{4}$$

$$\sum_{a\in A_i}u_{a,t}x_a+\frac{k_{i,t-1}}{k_{i,t}}\left(y_{i,t-1}^{queue1}+y_{i,t-1}^{queue2}+y_{i,t-1}^{penalty}\right) \leq C_i^{des}+y_{i,t}^{des}+y_{i,t}^{queue1}+y_{i,t}^{queue2}+y_{i,t}^{penalty} \qquad i\in I, t\in T_i' \tag{5}$$

$$C_i^{des}+y_{i,t}^{des}\leq C_i^{tech} \qquad i\in I, t\in T_i' \tag{6}$$

$$C_i^{tech}+y_{i,t}^{queue1}\leq C_i^{queue1} \qquad i\in I, t\in T_i' \tag{7}$$

$$C_i^{queue1}+y_{i,t}^{queue2}\leq C_i^{queue2} \qquad i\in I, t\in T_i' \tag{8}$$

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

$$y_{i,t}^{des}, y_{i,t}^{queue1}, y_{i,t}^{queue2}, y_{i,t}^{penalty}\geq 0 \qquad i\in I, t\in T_i' \tag{10}$$

The objective function includes both *late bags* and *missed probability*, since this is the way it appears in the current Frankfurt model. Although missed probability is the main objective, any late bag creates stress in the system, and it may become a missed bag.

The first constraint (4) ensures that every trip is assigned to an infeed area. Constraint (5) calculates the capacity use and ensures that stretched workload is carried over to next time period. The term $k_{i,t-1}/k_{i,t}$ is used to convert the load between time periods $t-1$ and t when productivity changes. Constraint (6) ensures that usage of buffer capacity is only possible between desired workload and technical workload capacity. Constraint (7) ensures that usage of queue1 capacity is only possible between technical workload and first degree overcapacity. Finally, constraint (8) ensures that usage of queue2 capacity is only possible between first degree overcapacity workload and second degree overcapacity workload. Constraints (9) and (10) define the domains of the variables.

The deterministic model has $O(A)$ binary and $O(IT)$ continuous variables. Furthermore it has $O(B+IT)$ constraints.

The model assigns each trip to exactly one infeed area. The different assignments will lead to various levels of each of the cost terms affecting the objective function, but it also affects the capacity use of the infeed area. The model keeps track of this capacity use at each infeed area in each time-step. Here, it takes into account workload from assigned trips and possible queuing work from the previous time-step. The different steps of the capacity use then affects both the objective function and the subsequent time-step.

If a trip contains only one transfer baggage container e.g. because of pairing with an inbound container, it is desired that the infeed area of the trip is either the same as the previous or the following trip. To model this, two new sets are introduced: Set $B'\subseteq B$ is the subset of trips where

only one container is used in the trip. Set $N_{b'} \subseteq B$ is the subset of neighboring trips for a trip with only one container $b' \in B$.

The sets are used in the following constraint ensuring that trips with only one container are transported to the same infeed area as either its previous or following trip.

$$\sum_{a \in A_i \cap A_{b'}} x_a \leq \sum_{b \in N_{b'}} \sum_{a \in A_i \cap A_b} x_a \qquad i \in I, b' \in B' \tag{11}$$

Constraint (11) is defined for each trip having only a single container. These trips can only be handled at an infeed area where one of their neighboring trips also is handled.

In Frankfurt Airport the TBP is currently solved using a similar simple deterministic MILP model. The model is solved continuously throughout the day approximately every 2nd minute. Each time it is solved, it only includes flights which are estimated to land within the time horizon of one hour. The underlying data is constantly updated. Each consecutive execution might therefore select different assignments for a given trip, but once the flight is on-block the chosen assignment for each of the flights trips is fixed, using capacity in the chosen infeed areas. Because the model is deterministic, it assumes that the available data is reliable. This is clearly wrong due to the many uncertainties i.e. arrival times, departure times, transportation times etc. The continuous re-optimization of the problem helps to minimize the effect of these uncertainties. It allows the final decision of the assignments to be postponed to the latest possible moment, which exploits that newer and more reliable data will be available closer to the time of an event.

A single instance of the model described in this section is denoted DET_{single}. The creation of a solution for an entire day of operation, by continuously solving the deterministic model every 2 minutes, is referred to as *DET*. This simulates the real life application of the model and makes up the base for evaluating its performance.

4 Uncertainties in Data

In order to improve the deterministic model it is necessary to take a number of uncertainties into account. From Frankfurt Airport, we have received various kind of data covering a month of operations at the airport. The data are grouped in three categories: updates on estimated arrival time, transportation time and position change. In the following we will handle these categories separately.

4.1 Updates on estimated arrival time

For each arriving aircraft, the *estimated on-block time* (EON) is the time when the aircraft is expected to be parked at the gate, with blocks under its tires to prevent it from moving, hence *on-block* (ONB). At this time the aircraft will be ready to unload passengers and baggage.

Prior to this time the aircraft regularly report updates of the EON. These updates reflect possible delays or time gain due to conditions at the departure airport, weather, air traffic or taxiing at Frankfurt Airport.

Frankfurt Airport has provided data for all updates on EON, within two hours before ONB, during a month. This adds up to a total of 163 150 updates distributed over 18 387 distinct flights. Because of imbalance, created by inconsistency between the way data have been extracted and the data processing, only updates up to 100 minutes before ONB are used. For each update we are given: INB_{ID} the id of the inbound flight, *ChangeTime* the time where the update (change in EON) is reported, EON_{new} the new EON, and finally *ONB* the actual on-block time.

4.1.1 General analysis

In Figure 4 the updates are plotted with $Difference = ONB - EON_{new}$ as function of $TimeBefore = EON_{new} - ChangeTime$. Note that a negative *Difference* indicates early arrival and a positive *Difference* indicate late arrival compared to the predictions. In addition we have added a linear regression line for the updates up to 60 minutes before ONB, where the trend seems to change. The line follows the function

$$f(x) = -0.1913x + 3.4179 \tag{12}$$

Here we see that updates around 40-60 minutes tend to be more cautious and predicts later ONB than the actual one. But as the updates get closer to the ONB they tend to be more and more optimistic, and end up predicting arrival a bit earlier than the actual one. Updates earlier than one hour before seems to deviate from this trend, being well distributed around 0 instead, however with a large variance. For the updates occurring within 30 minutes before actual ONB they have a tendency of drifting towards later arrival than expected, while the earlier arrivals have a more sharp border 10 minutes from the trend line. This makes sense because unexpected delays should be more common than early arrivals.

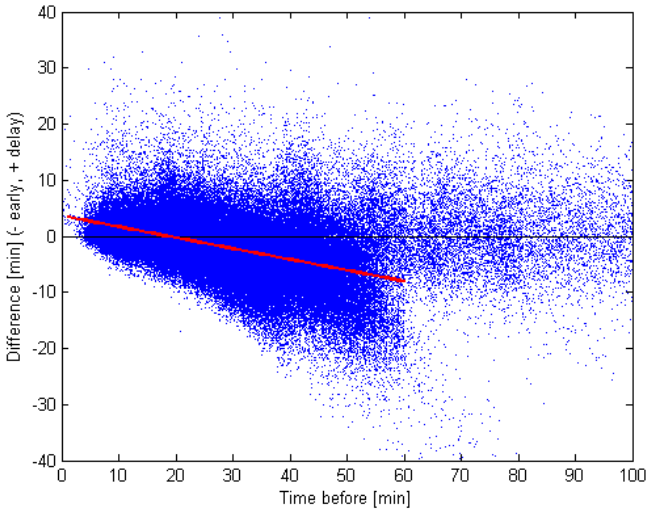

Figure 4: All updates, shown with a tendency line for updates up to 60 minutes before ONB

We investigate the distribution of the accuracy of the updates by plotting a histogram of the *Difference* for all updates. The distribution seems to follow the T-location-scale distribution

(TLS). This is based on the Student's t-distribution with a degree of freedom of ν. If y is a Student's t-distribution with a degree of freedom of ν, then the T-location-scale distribution is defined as $y\sigma+\mu$. TLS simulates a normal distribution transformed by the third parameter ν and it approaches the normal distribution when $\nu \to \infty$. For smaller values of ν it should be better at representing the larger tails and higher peak, keeping the bell shape of the normal distribution. A maximum likelihood fit of the T location scale distribution gives the following parameters:

$$\mu = -1.739, \qquad \sigma = 4.062, \qquad \nu = 4.552$$

In Figure 5 the T-location-scale distribution fit has been plotted in the histogram. It fits very well and definitely have a better representation of the tails and peak. We do see that the EONs are skewed a bit to the right, which is not captured by the symmetric fit. This explains why the fit is still a bit off. However we conclude that the T-location-scale distribution is a good approximation of the EONs.

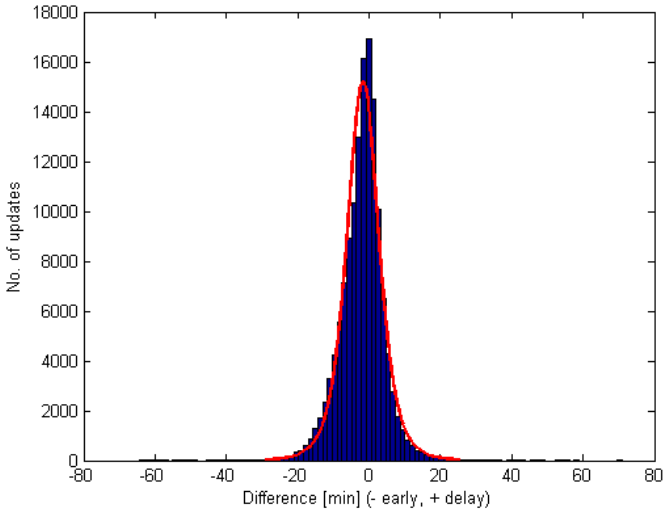

Figure 5: Histogram of *Difference* for all updates and fitted T-location-scale distribution

4.1.2 Time dependent distribution

To use the analyzed data in a simulation we aim to find the quality of a current EON for a flight at any given time before ONB. If we used the results from the previous section we would assume that all flights report an EON update constantly, but in reality this only happens 8.9 times on average during the last 2 hours before arrival. The consequence is that the EONs known to the airport are more or less outdated. For this analysis we will therefore look at the latest update for all distinct flights, and disregard the rest of the updates. If no update is available the flight is not considered.

Using this approach, we have examined the available EONs at the time points 0, 10, 20, 30, 40 and 60 minutes before the actual ONB. For these an analysis similar to the one performed

on all EONs is made. This shows a good fit with the T-location-scale distribution, for all time points. The related figures can be seen in [15].

From this we assume that the *Difference* of the available EON updates, at all times, can be approximated by a T-location-scale distribution. For the EONs available at each minute up to 100 minutes before ONB, we approximate a fit of the T-location-scale distribution. This results in Figures 6 and 7 where the changes in the distribution parameters can be seen over time. These graphs should be compared to Figure 8 and Figure 9. Here Figure 8 shows the number of distinct flights which have reported an EON as a function of time before each aircraft actually arrive. 100 minutes before actual arrival time almost no flights have reported an EON and many reports their first EON between 40 and 55 minutes before actual arrival. Figure 9 is showing the average age of the most recent EON updates of all flights which have yet reported an EON, again as a function of time before actual arrival. From 10 to 50 minutes before arrival the average age of the latest EON is around 5 minutes old. The graphs in this section tells us how reliable our EON's are when we they are reported, and we have identified the distribution that describes the variations.

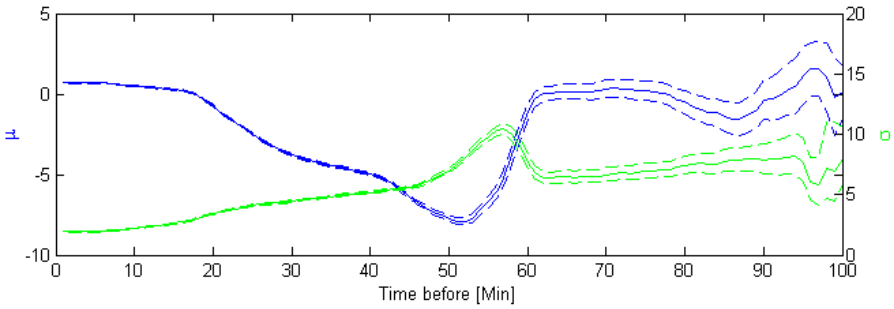

Figure 6: Parameters μ (blue line left axis) and σ (green line right axis) including confidential interval for T-location-scale distributions, based on *Difference* of available EONs at different times until ONB

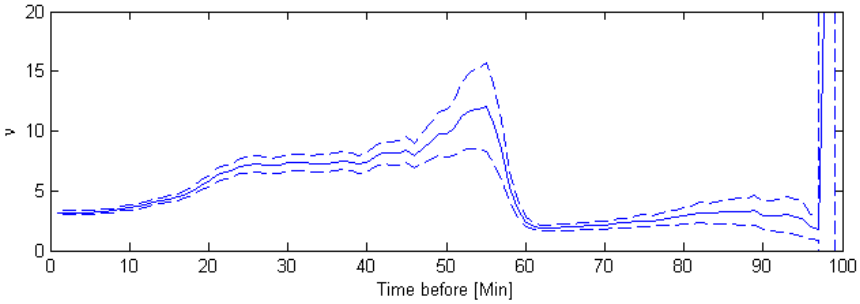

Figure 7: Parameter ν including confidential interval for T-location-scale distributions, based on *Difference* of available EONs at different times until ONB

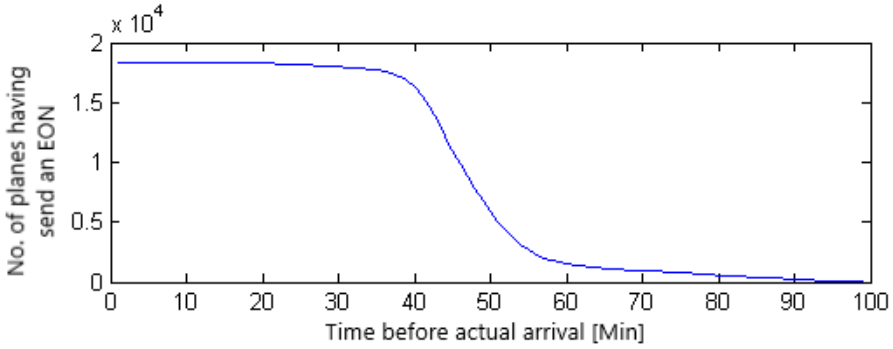

Figure 8: Number of flights where an EON have been reported as a function of the time until actual arrival

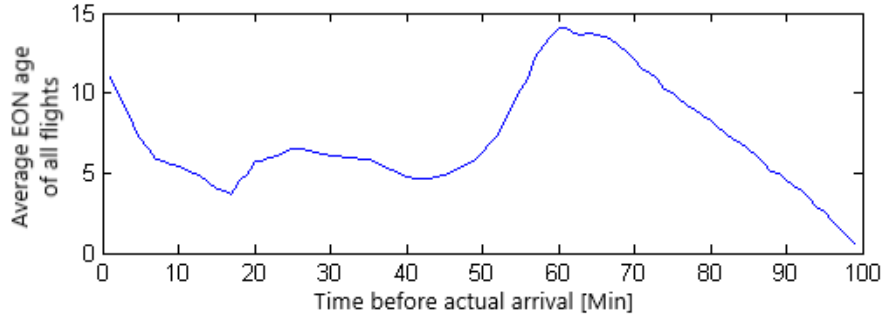

Figure 9: Average age of the latest received update for of all flights as a function of time until actual arrival time for each of those flights

4.2 Position changes

An analysis of the position changes was performed next. This investigates the probability of a flight having a change in the planned parking position assigned to it, prior to its landing. This did lead to various interesting findings, however when applying the found results we did not get any improvement in the solutions. For this reason, the corresponding analysis is not presented in this paper, but can be found in [15].

4.3 Transportation times

An analysis of transportation times from the inbound flight to the infeed area was also performed. This showed that the transportation times can be described with a log-normal distribution. The idea was to use this analysis to predict some of the uncertainties in the transportation times. However we ended up not using this analysis. This was decided because the transportation times cannot be confirmed from the data as the choice of assignment affects the transportation times. Thus we had no way of validating whether including uncertainties in transportation time helped

or not. The full analysis can be found in [15].

5 Stochastic formulation

An immediate way of using the data analysis is to extend the original model DET_{single} into a stochastic model. We use a simple single-stage optimization over scenarios as described in [8]. By simultaneously optimizing on multiple scenarios, the uncertainties in the data is taken into account by the model. The data analysis for EON updates is used to create the different scenarios, reflecting the uncertainties in the EONs. In this section we describe a stochastic version of the DET_{single} model. The stochastic model will be exposed to the changes in the ONB that will eventually happen. This should make it possible for the model to choose a better and more robust solution taking into account the future changes.

In order to extend the DET_{single} into a stochastic model a new set S is introduced, representing the individual scenarios s in the stochastic optimization.

The scenarios are created by choosing each inbound flight in the optimization that has not yet reached on-block. For each scenario, and for each of the inbound flights, the corresponding T-location-scale distribution is used to sample a simulated arrival time. Depending on the flights' time until EON, different distributions are picked. The distributions are picked with an accuracy of 1 minute. The simulated arrival times for two flights, with an estimated arrival time 1 minute apart, will therefore be drawn from two slightly different distributions. Hereby we utilize the detailed results from Section 4.1.2 of this article. The distribution consists of the parameters μ, σ and v. From these, a deviation in the flights arrival time is created for each scenario. A positive deviation makes up a delay, while a negative deviation simulates a flight arriving earlier than expected. The deviation then denotes the difference between the given EON and the simulated arrival time at the gate (ONB). When using the TLS distribution, the deviation is created according to Equation (13) where T is the standardized Student's t-distribution with parameter ν. The deviations now reflect the uncertainty of the EON of all inbound flights. We have not analyzed the scenarios generated in this way, nor have we done any selection or discrimination between them. The outcome relies completely on the random draws from the distributions. We are confident that the sample space of scenarios is sufficiently covered due to many individual flights and generation of new scenarios every 2 minutes.

$$scnDelay_s = \mu + \sigma T \qquad s \in S \tag{13}$$

Each flight now has different simulated ONB times in each of the scenarios. The deviation in arrival time affects the capacity usage in the baggage handling system, which will now be shifted accordingly. In each scenario and for each possible assignment, the affected bags may now result in a unique capacity usage among the time slots. Also *missed probability* related to each assignment will now be affected by the deviation in arrival time in each scenario. The capacity usage and the *missed probability* corresponding to each of the flights' assignments have to be calculated for each scenario. These values are now defined for each scenarios as:

- $\upsilon_{s,a,t}$, stochastic capacity use, amount of capacity used for scenario $s \in S$ in timeslot $t \in T$ of assignment $a \in A$.

- $\upsilon_{s,a}^{missed}$, total missed probability of all bags in assignment $a \in A$ for scenario $s \in S$.

These are calculated, for each scenario, using the same method as in Section 2 of this article. Only difference is that in each scenario we have shifted the simulated arrival time by the time sampled from the distribution.

The variables used for apron transportation times u_a^{drive} and BHS transportation times u_a^{BHS} are kept deterministic. However it would make sense to make these stochastic as well to reflect even more of the uncertainty in the transfer baggage system. The reason why we refrain from doing so is to enable a direct comparison with existing system DET_{single}. Since DET_{single} does not reflect any uncertainties in the transportation times, it would be an unequal comparison. The uncertainties in transportation would very likely make the model perform better in practice, but with the available data we would not be able to verify this. However, the *missed probability* $\upsilon_{s,a}^{missed}$ and the capacity use $\upsilon_{s,a,t}$ are made stochastic since the data does reflect uncertainties in the on-block times of the flights. The available data includes the correct final on-block times for all flights, thus it is possible for us to verify how well the prediction of the model would actually perform.

To make the model stochastic, new variables are defined to describe the stochastic capacity use. These represents buffer-, queue1-, queue2- and penalty-capacity usage for each scenario. The new stochastic variables $\gamma_{s,i,t}^{des}$, $\gamma_{s,i,t}^{queue1}$, $\gamma_{s,i,t}^{queue2}$ and $\gamma_{s,i,t}^{penalty}$ correspond to the variables $y_{i,t}^{des}$, $y_{i,t}^{queue1}$, $y_{i,t}^{queue2}$, and $y_{i,t}^{penalty}$, from DET_{single}. The new variables are simply adapted to stochastic capacity use by additionally spanning the scenarios $s \in S$.

The x variables defining the choice of assignments are kept the same. Hereby the model still needs to only do a single assignment for each trip. The model must choose the assignment that results in the average best solution over all scenarios. The stochastic model STO_{single} is now given as follows

$$\min \quad \sum_{a\in A}\left(w^{drive}u_a^{drive}+w^{BHS}u_a^{BHS}+\frac{1}{|S|}\sum_{s\in S}w^{missed}\upsilon_{s,a}^{missed}\right)x_a \\ +\sum_{i\in I}\sum_{t\in T}k_{i,t}\frac{1}{|S|}\sum_{s\in S}\left(w^{des}\gamma_{s,i,t}^{des}+w^{queue1}\gamma_{s,i,t}^{queue1}+w^{queue2}\gamma_{s,i,t}^{queue2}+w^{penalty}\gamma_{s,i,t}^{penalty}\right) \tag{14}$$

$$\text{s.t.} \quad \sum_{a\in A_b}x_a=1 \qquad b\in B \tag{15}$$

$$\sum_{a\in A_i\cap A_{b'}}x_a\le\sum_{b\in N_{b'}}\sum_{a\in A_i\cap A_b}x_a \qquad i\in I,b'\in B' \tag{16}$$

$$\sum_{a\in A_i}\upsilon_{s,a,t}x_a+\frac{k_{i,t-1}}{k_{i,t}}\left(\gamma_{s,i,t-1}^{queue1}+\gamma_{s,i,t-1}^{queue2}+\gamma_{s,i,t-1}^{penalty}\right) \\ \le C_i^{des}+\gamma_{s,i,t}^{des}+\gamma_{s,i,t}^{queue1}+\gamma_{s,i,t}^{queue2}+\gamma_{s,i,t}^{penalty} \qquad s\in S,i\in I,t\in T_i' \tag{17}$$

$$C_i^{des}+\gamma_{s,i,t}^{des}\le C_i^{tech} \qquad s\in S,i\in I,t\in T_i' \tag{18}$$

$$C_i^{tech}+\gamma_{s,i,t}^{queue1}\le C_i^{queue1} \qquad s\in S,i\in I,t\in T_i' \tag{19}$$

$$C_i^{queue1}+\gamma_{s,i,t}^{queue2}\le C_i^{queue2} \qquad s\in S,i\in I,t\in T_i' \tag{20}$$

$$x_a\in\{0,1\}, \qquad a\in A \tag{21}$$

$$\gamma_{s,i,t}^{des},\gamma_{s,i,t}^{queue1},\gamma_{s,i,t}^{queue2},\gamma_{s,i,t}^{penalty}\ge 0 \qquad s\in S,i\in I,t\in T_i' \tag{22}$$

The creation a solution for an entire day of operation using the stochastic model is referred to as *STO*. The most important changes to the model are in the objective function (14) that has been modified to express the average best choice of assignments. This is done by calculating the average assignment cost and capacity cost over all scenarios. Each scenario has probability $1/|S|$ since they represent evenly distributed outcomes.

The capacity constraints (17) to (20) are changed so that they are created for each scenario. The use of *buffer*, *queue1*, *queue2* and *penalty capacity* is constrained in each scenarios depending on the chosen assignments. Constraints (21) and (22) define the domains of the variables.

The stochastic model has $O(A)$ binary and $O(SIT)$ continuous variables. Furthermore it has $O(IB+SIT)$ constraints. Compared to the deterministic model, the stochastic model has significantly more constraints and linear variables when the number of scenarios S grows. However, the number of binary variables remains the same.

6 Semi-stochastic formulation

The complexity of the stochastic model STO_{single} model grows quickly with the number of scenarios. Our preliminary experiments showed that, within the given time limit of 2 minutes, only about 10 scenarios could be considered. We therefore present an improved stochastic model

combining the fast solution times of the deterministic model with the better solution quality of the stochastic model.

The idea is to focus on the most critical assignments in the stochastic model. These critical assignments are then handled in a stochastic way, while the rest are handled in a deterministic way. This semi-stochastic model will thus become a hybrid between the DET_{single} model and the STO_{single} model. This makes it possible to increase the number of scenarios while keeping the complexity of the model down.

The closer a flight is to ONB, the more critical its associated assignments are. Here we exploit that the method is rerun approximately every 2nd minute. In a subsequent solution step, the trips can be reassigned if their flight has not yet reached ONB. These re-assignable trips are therefore less critical. Consequently, we define the critical assignments to be the ones involving flight that are likely to reach ONB before the model is rerun. For this purpose we introduce the timespan Ω. All flights expected to reach ONB within Ω are regarded as critical. As time pass all flights will end up critical before the associated assignment are fixed.

The semi-stochastic model relies on identifying the critical assignments to handle in a stochastic way. Furthermore all time-steps affected by the stochastic assignments need to be identified. For this, the following subsets are introduced: Set $A^{STO} \subseteq A$ is the subset of assignments defining the stochastic assignments. Set $T_i^{STO} \subseteq T$ is the subset of time-steps defining the time-steps where a capacity use for stochastic assignment exist for area $i \in I$. Finally, set $T_i^{last} \subseteq T_i^{STO}$ is the subset of stochastic time-steps for area $i \in I$. The set defines the last time-step for each area included in T_i^{STO}.

Moreover, we define the binary coefficient $L_{i,t}$ to be 1 if and only if $t \in T_i^{last}$. We introduce the variable $y_{i,t} \geq 0$ to denote the deterministic capacity use for area $i \in I$ in time-step $t \in T'$. Further, to simplify notation let $T_i^* = \{t \in T_i^{STO} \mid k_{i,t} > 0\}$ be the set of time-steps where the baggage infeed area $i \in I$ is active and where capacity use exists for any stochastic assignment for that area. Reusing the definitions from DET_{single} and STO_{single} the semi-stochastic model $SEMI_{single}$ is given as

$$
\begin{aligned}
\min \quad & \sum_{a \in A \setminus A^{STO}} \left(w^{missed} u_a^{missed} + w^{drive} u_a^{drive} + w^{BHS} u_a^{BHS} \right) x_a \\
& + \sum_{i \in I} \sum_{t \in T} k_{i,t} \left(w^{des} y_{i,t}^{des} + w^{queue1} y_{i,t}^{queue1} + w^{queue2} y_{i,t}^{queue2} + w^{penalty} y_{i,t}^{penalty} \right) \\
& + \sum_{a \in A^{STO}} \left(w^{drive} u_a^{drive} + w^{BHS} u_a^{BHS} + \frac{1}{|S|} \sum_{s \in S} w^{missed} \upsilon_{s,a}^{missed} \right) x_a \\
& + \sum_{i \in I} \sum_{t \in T_i^{STO}} k_{i,t} \frac{1}{|S|} \sum_{s \in S} \left(w^{des} \gamma_{s,i,t}^{des} + w^{queue1} \gamma_{s,i,t}^{queue1} + w^{queue2} \gamma_{s,i,t}^{queue2} + w^{penalty} \gamma_{s,i,t}^{penalty} \right)
\end{aligned}
\tag{23}
$$

s.t.
$$\sum_{a \in A_b} x_a = 1 \qquad b \in B \tag{24}$$

$$\sum_{a \in A_i \cap A_{b'}} x_a \leq \sum_{b \in N_{b'}} \sum_{a \in A_i \cap A_b} x_a \qquad i \in I, b' \in B' \tag{25}$$

$$\sum_{a \in A_i \setminus A^{STO}} u_{a,t} x_a + \frac{k_{i,t-1}}{k_{i,t}} \left(y_{i,t-1}^{queue1} + y_{i,t-1}^{queue2} + y_{i,t-1}^{penalty} \right) + L_{i,t} \cdot \frac{k_{i,t-1}}{k_{i,t}} \frac{1}{|S|} \sum_{s \in S} \left(\gamma_{s,i,t-1}^{queue1} + \gamma_{s,i,t-1}^{queue2} + \gamma_{s,i,t-1}^{penalty} \right) \leq y_{i,t} \qquad i \in I, t \in T_i' \tag{26}$$

$$y_{i,t} \leq C_i^{des} + y_{i,t}^{des} + y_{i,t}^{queue1} + y_{i,t}^{queue2} + y_{i,t}^{penalty} \qquad i \in I, t \in T_i' \tag{27}$$

$$\sum_{a \in A_i \cap A^{STO}} v_{s,i,t} x_a + \frac{k_{i,t-1}}{k_{i,t}} \left(\gamma_{s,i,t-1}^{queue1} + \gamma_{s,i,t-1}^{queue2} + \gamma_{s,i,t-1}^{penalty} \right) + y_{i,t} \leq C_i^{des} + y_{i,t}^{des} + y_{i,t}^{queue1} + y_{i,t}^{queue2} + y_{i,t}^{penalty} + \gamma_{s,i,t}^{des} + \gamma_{s,i,t}^{queue1} + \gamma_{s,i,t}^{queue2} + \gamma_{s,i,t}^{penalty} \qquad s \in S, i \in I, t \in T_i^* \tag{28}$$

$$C_i^{des} + y_{i,t}^{des} \leq C_i^{tech} \qquad i \in I, t \in T_i' \tag{29}$$

$$C_i^{tech} + y_{i,t}^{queue1} \leq C_i^{queue1} \qquad i \in I, t \in T_i' \tag{30}$$

$$C_i^{queue1} + y_{i,t}^{queue2} \leq C_i^{queue2} \qquad i \in I, t \in T_i' \tag{31}$$

$$C_i^{des} + y_{i,t}^{des} + \gamma_{s,i,t}^{des} \leq C_i^{tech} \qquad s \in S, i \in I, t \in T_i^* \tag{32}$$

$$C_i^{tech} + y_{i,t}^{queue1} + \gamma_{s,i,t}^{queue1} \leq C_i^{queue1} \qquad s \in S, i \in I, t \in T_i^* \tag{33}$$

$$C_i^{queue1} + y_{i,t}^{queue2} + \gamma_{s,i,t}^{queue2} \leq C_i^{queue2} \qquad s \in S, i \in I, t \in T_i^* \tag{34}$$

$$x_a \in \{0,1\}, \qquad a \in A \tag{35}$$

$$y_{i,t}^{des}, y_{i,t}^{queue1}, y_{i,t}^{queue2}, y_{i,t}^{penalty} \geq 0 \qquad i \in I, t \in T_i' \tag{36}$$

$$\gamma_{s,i,t}^{des}, \gamma_{s,i,t}^{queue1}, \gamma_{s,i,t}^{queue2}, \gamma_{s,i,t}^{penalty} \geq 0 \qquad s \in S, i \in I, t \in T_i^* \tag{37}$$

The creation a solution for an entire day of operation using the semi-stochastic model is referred to as *SEMI*.

The $SEMI_{single}$ model uses the equations from DET_{single} to handle all the non-stochastic assignments. On top of this, the stochastic assignments are handled. The overall idea for the $SEMI_{single}$ is to use the DET_{single} as the backbone for the model. The constraints and variables from DET_{single} are used for all the non-stochastic assignments. The only difference here is that the capacity use constraint from the DET_{single}, constraint (5), is now represented by two constraints (26) and (27).

The objective function (23) combines the objective from DET_{single} and STO_{single}, calculating the cost for the deterministic assignments added with the average cost for the stochastic assignments.

Constraint (26) defines the deterministic capacity use $y_{i,t}$. Based on the chosen assignments and overcapacity carried over from previous time-step. When the time-steps in an area change from stochastic to deterministic, the indicator parameter $L_{i,t}$ ensures that the average of the stochastic capacity use is taken into account.

Constraint (27) links the deterministic capacity cost variables with the deterministic capacity use. Constraint (28) models the flow of the stochastic capacity which is influenced by the deterministic capacity use $y_{i,t}$. This constraint is defined for each scenario, but only for time-steps $t \in T_i^{STO}$ in area $i \in I$.

Constraints (29) to (31) ensure that the variables for the capacity use cannot exceed their predefined interval. Constraints (32) to (34) ensure that the combined deterministic and stochastic capacity use is within the interval. Constraints (35)–(37) define the domains of the variables.

The semi-stochastic model has $O(A)$ binary variables. Furthermore it has $O(SIT)$ "stochastic" continuous variables and $O(IT)$ "deterministic" continuous variables. The number of constraints is of magnitude $O(IB+SIT)$. Hence, compared to the stochastic model, we can limit the number of "expensive" stochastic variables.

6.1 Tuning

When solving the semi-stochastic model, both the number of scenarios and the timespan Ω needs to be defined. Ideally both of these should be tuned together, finding the best combination. As this is a very comprehensive task we have decided instead to use a fixed number of scenarios and afterwards perform a tuning on Ω. Preliminary tests showed that 100 scenarios were a reasonable number to solve within the time limit of 2 minutes. This is a 10 time increase compared to the number of scenarios that can be handled in *STO*. Using even more scenarios greatly increases the running time and complexity with more single TBPs reaching the time limit, and hence possibly reducing solution quality.

The Ω parameter controls how big a part of the problem should be considered stochastic. It has a large impact on the complexity of the model and is essential for the model to be solvable within a small time frame.

The tuning of Ω is done using instances D4, D6 and D8. These were chosen to reflect a good spread of the number of bags and thereby the level of workload. Instance D8 has a small number of bags (31070), D6 has medium number of bags (35630) and D4 has a large number of bags (38826).

Different Ω-values are tested for each of the instances ranging from 0 to 14 minutes. Each combination of instance and Ω-value is run five times to get an idea of the spreading of the solutions. The objective function and the running times can be seen in [15].

The best found solution in the tuning is identified and for each solution the percentage gap to the best solution is calculated. The gaps are illustrated in Figure 10 where higher gaps denote worse objective functions. The corresponding solution times are seen in Figure 11.

Despite large fluctuations the figures does show a tendency that an increase in Ω results in a better objective function. There is a small outlier when increasing Ω from 2 to 4 minutes, but we assume this is due to stochastic variation. the objective function deviates from this trend. This can likely be explained by stochastic variation. For the running times, the correlation is more

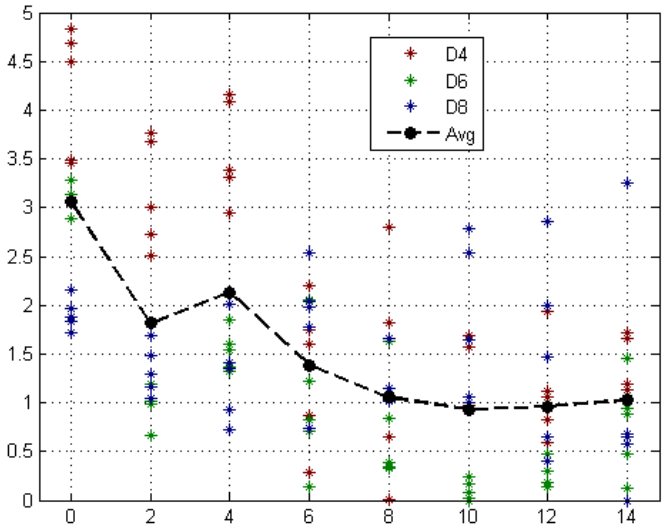

Figure 10: Objective value [% gap from best] for each of the three tested instances (D4,D6 and D8) and average objective value, as function of Ω [min]

consistent with time increasing when Ω is increased. When Ω is 8 minutes the objective function begins to flatten out, indicating that a further increase in Ω has no or a very small effect on the objective. Actually, a small worsening of the objective can be seen once Ω is increased above 10 minutes. This could be explained by the increase in running time caused by the larger values of Ω. Higher running times indirectly imply that more of the single executions reach the time limit of 120 seconds. Reaching this limit, results in non-optimal solutions worsening the overall solution for *SEMI*. Based on this analysis a timespan Ω of 10 minutes was chosen.

7 Computational Experiments

To evaluate the three presented models, we use the 9 datasets presented earlier in Table 1. Each dataset includes updated information for every 2nd minute over the cause of an entire day. This data enable us to solve the *DET*, *STO* and *SEMI* for each of the 9 days. This is done by running a model every 2nd minute using a rolling horizon and gradually fixing the assignments as the aircraft arrive. In the end the assignments for the entire day of operation will be fixed and the performance can be evaluated. The execution of each *single* model has a time limit of 120 seconds. The best found solution is used in case the time limit is reached before optimality is proven. In each *single* model the *time discretization* for the set T is set to 5 minutes. The *single* model considers a *time span* from 30 minutes before current time to 120 minutes after current time. All experiments have been run on a laptop with an i7 2.40 GHz processor and 8 GB RAM using CPLEX as solver.

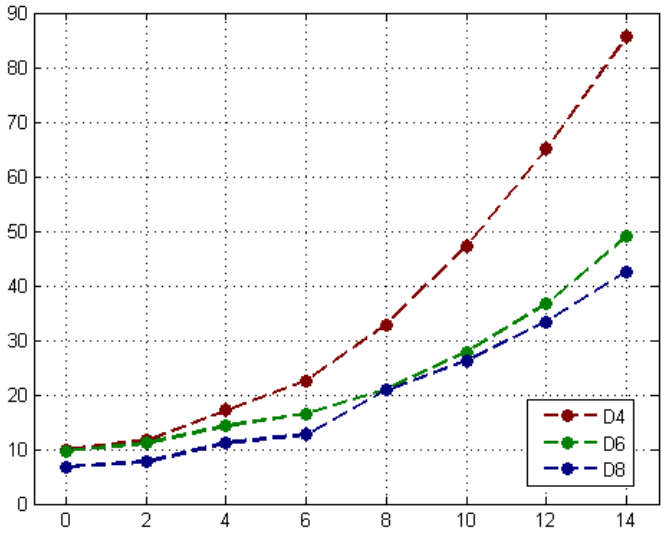

Figure 11: Average total running time [min] of *SEMI* for each of the three tested instance (D4,D6 and D8) as function of Ω [min]

7.1 Results, deterministic model

The deterministic model finds the best possible solution based on the available data disregarding any uncertainty. Most likely, the objective value of the solution will change later as the reality is affected by uncertainties. As data updates come in, the model is free to correct the solution the next time it is solved. However, when a flight is on-block the assignments of all its trips can no longer be changed as these operations need to be carried out immediately. In order to make a more correct evaluation of the solutions, we study how it performs in reality. Our approach is based on on the method used in [3]: All chosen assignments are evaluated after solving all TBPs, using DET_{single}, in an instance (during a day). Whenever assignments are locked because the flight is on-block, we save these locked assignments. At the end of the day, we take the final data for each flight and all locked assignments based on the saved ones. Based on these the objective function for the whole day is calculated, later referred to as *DET*. In practice this is done by formulating and solving a TBP covering the entire day with all assignments fixed.

For the objective function of the deterministic model (3) to (8), we use the weights currently used in the model at Frankfurt Airport, as reported in Table 2. The main objective is to minimize the number of missed bags, while the auxiliary objectives, have lower weights. Two separate terms in the objective function aim at reducing the number of missed bags: The number of late bags using weight w^{late}, and the *missed probability* using w^{missed}.

w^{BHS}	w^{drive}	w^{missed}	w^{late}	w^{des}	w^{queue1}	w^{queue2}	$w^{penalty}$
0.001	0.5	375	35	1.5	3	10	100

Table 2: Weights used in the objective function by Frankfurt Airport

For each instance, the TBP model is evaluated based on an entire day of operation. Table 3 shows the weighted terms of the objective function for a single instance, namely D1, summing up to the total objective function value of 470 347.

Capacity buffer	**Queue1**	**Queue2**	**Penalty**	**Transportation**		**Late baggage**	**Missed probability**
				BHS	drive		
5 708	1 838	4 148	55 537	27 431	173 956	16 157	185 572

Table 3: Weighted terms of the objective function for instance D1

To further analyze the capacity terms, we focus on the capacity use of the infeed area TZA, this area is chosen as it reflects the general use of the infeed areas throughout the day. The capacity use for TZA is shown in Figure 12, while results for the remaining infeed areas are reported in [15]. Each figure shows the percentage capacity use for each time-step throughout the day. The lines represent the restriction levels where respectively buffer, queue1, queue2 or penalty capacity becomes active. For TZA it is seen that the capacity use is clustered together in 4–5 clusters throughout the day. This probably reflects the peaks where most inbound flights are being handled at the airport. Inside each cluster a lot of fluctuation can be seen. Some time-steps have very little or no capacity use where others are close to technical capacity or even above. The model will never plan penalty capacity, because it is very expensive, hence when it occurs it indicates that an aircraft arrived early/late, shifting assignments from one time-step to another. The delays give rise to overcapacity, even though this was not foreseen when scheduling the trips to the infeed areas.

To get an idea of the quality of the solution, a lower bound on the objective function is calculated. This is done by running the model over the entire day with the final data for all flights. By already knowing the future, the model can choose the best possible assignments. The results are summarized in Table 4 showing the objective function, the lower bound, the gap between the objective and lower bound, the total execution time the and maximum time used for solving a single TBP.

Instance	Objective	Lower bound	Gap	Time	Max
D1	470 347	390 706	16.93%	11:07	41.0
D2	390 914	366 135	6.33%	04:54	1.1
D3	480 360	466 812	2.82%	04:32	0.6
D4	418 659	382 953	8.53%	04:55	0.8
D5	373 113	351 011	5.92%	04:29	1.9
D6	297 897	257 744	13.48%	04:23	1.2
D7	264 669	241 833	8.63%	04:49	0.6
D8	295 648	276 416	6.51%	05:05	10.7
D9	330 298	309 469	6.31%	04:27	0.8

Table 4: Results for the nine different instances, including total solution time (Time) in [mm:ss], and maximum solution time for solving DET_{single} (Max) in [s]

The deterministic model seems to find decent solutions to the TBP, and it is very fast to solve.

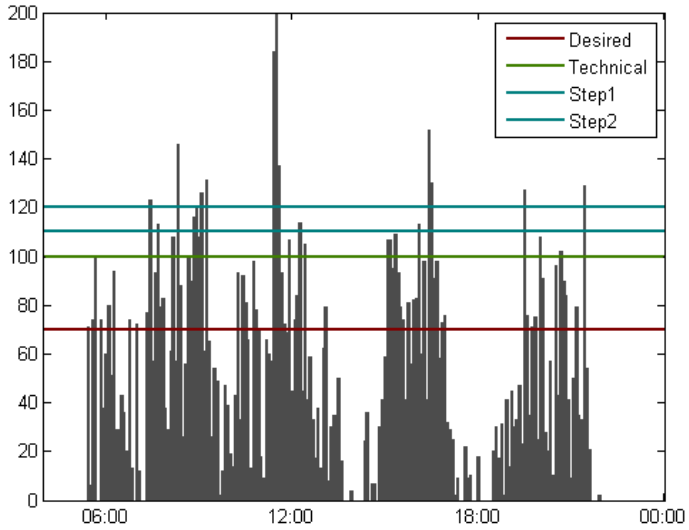

Figure 12: Capacity use [%] for each time-step throughout the day [hh:mm], together with the capacity restriction levels. Showed for infeed area TZA in instance D1

The main weakness of the model is the lack of ability to predict congestion, which illustrates a clear lack of robustness.

7.2 Results, stochastic model

The stochastic model *STO* has been tested on the same nine instances as *DET*. As expected the stochastic model could not be solved within a reasonable time when using a larger number of scenarios, hence only 10 scenarios are considered in our experiments. Despite the limited number of scenarios, some of the instances could not be solved to optimality. Therefore, each single STO_{single} was given a time limit of two minutes. This reflects the time available in the set-up at Frankfurt Airport.

The 10 scenarios were created by randomly sampling from the given TLS distribution. Due to the limited number of scenarios, the choice of scenarios can have a large effect on the solution. Consequently, each instance has been solved a total of five times. This should give a better evaluation of *STO*.

The complete set of test results can be found in [15]. The average stochastic objective value is reported in Table 5. Here it is presented together with the objective from *DET* and the best possible objective representing the lower bound. Table 6 shows the objective function when including the missed baggage term. Both tables also show the percentage difference from *DET* to *STO* and from *STO* to *LB*. Here positive values denote an improvement in the objective function.

From the tables it is seen that the *STO* model on average finds a better solution compared to *DET*. Eight out of the nine instances give a better objective function significantly reducing the gap towards *LB*. The improvements range from -1% to 8.5% with an average improvement of 3.78%. It can be seen that the inclusion of the late baggage term does not make any significant

Instance	*DET*	$Diff_1$	*STO*	$Diff_2$	*LB*
D1	454 190	8.18%	417 050	9.75%	376 394
D2	373 123	3.70%	359 322	2.96%	348 684
D3	453 244	-1.09%	458 192	3.85%	440 534
D4	400 696	3.16%	388 027	5.72%	365 815
D5	357 355	2.41%	348 734	3.58%	336 251
D6	290 018	8.67%	264 868	5.47%	250 373
D7	257 163	4.26%	246 209	4.52%	235 084
D8	284 505	2.61%	277 070	4.27%	265 248
D9	315 243	3.05%	305 619	3.52%	294 846

Table 5: Average objective function values of *DET* and *STO*. $Diff_1$ specifies the percentage difference between *DET* and *STO*. $Diff_2$ specifies the percentage difference between *STO* and *LB*.

Instance	*DET*	$Diff_1$	*STO*	$Diff_2$	*LB*
D1	470 347	7.97%	432 877	9.74%	390 706
D2	390 914	3.55%	377 033	2.89%	366 135
D3	480 360	-1.01%	485 196	3.79%	466 812
D4	418 659	3.01%	406 049	5.69%	382 953
D5	373 113	2.34%	364 394	3.67%	351 011
D6	297 897	8.48%	272 644	5.46%	257 744
D7	264 669	4.18%	253 617	4.65%	241 833
D8	295 648	2.51%	288 241	4.10%	276 416
D9	330 298	2.92%	320 666	3.49%	309 469

Table 6: Average objective function values of *DET* and *STO*, including missed baggage term. $Diff_1$ specifies the percentage difference between *DET* and *STO*. $Diff_2$ specifies the percentage difference between *STO* and *LB*.

difference.

The improvement can be translated into an equivalent number of saved bags, assuming that the weights in the objective function reflects the actual relation between the terms. This corresponds to the idea that an improvement in the capacity use should cause less bags to be delayed in the infeed process, resulting in fewer missed bags. One piece of missed baggage corresponds to a value of 375. In this case *STO* on average saves 36 bags daily.

Table 7 shows the average terms of the objective function when solving instance D8, when using model *STO*compared to model *DET*. Results for the other instances follow a similar trend, as can be seen in [15]. There is a clear tendency of capacity costs being largely decreased. This decrease is mainly due to reduced penalty capacity, but also the remaining capacity costs are reduced. Furthermore there is a trend towards a small increase in apron transportation. Probably, the longer transportation is necessary to ensure the better distribution of capacity use. With instance D8 representing the trend, clearly the improvements in eight out of nine instances are a result of a more robust solution. Hence *STO* is significantly better than *DET* at predicting the actual capacity use, ensuring that trips less often collide at the infeed areas.

The *missed probability* does not show any notable improvement. With most of the instances fluctuating very little corresponding to ± 1 missed baggage. Only D1 deviates from this trend by

saving up to 10 bags compared to *DET*. A significant improvement of *missed probability* in only one out of nine instances could be because *DET* already is handling this term very well.

For the *late bags* term we see a very small increase. This confirms that the term is redundant because of the *missed probability* term.

	Capacity buffer	**Queue1**	**Queue2**	**Penalty**	**Transportation**		**Missed probability**	**Late baggage**	**Objective (extended)**
					BHS	drive			
STO	2 745	569	683	3 171	17 882	127 775	124 247	11 170	288 241
DET	3 270	972	1 486	11 076	17 682	125 502	124 519	11 143	295 651

Table 7: Weighted terms for average *STO* and *DET* for instance D8

Instance D3 is examined further to identify why the *STO* solution here is worse than the *DET* solution. It is found that the main reason for the worse objectives is the penalty capacity. In D3 the *DET* solution has a very low penalty compared to the other instances. *STO* still tries to improve the capacity use by increasing apron transportation time. This leads to an increase in transportation cost while ending up with a larger penalty capacity anyway. A main factor is that there is not much penalty capacity in the *DET* solution to reduce. Here, *DET* might be lucky choosing a good assignment or in this instance, simply involving as few possible congestions. *STO* on the other hand might be unlucky if the 10 scenarios were a bad representation of the distribution in all 5 solutions. Alternatively, it could be due to the fact that critical flights unfortunately represented outliers of the distribution in the particular instance. Probably the explanation is a mixture of several of these factors.

The time consumption of all *STO* results are reported in [15]. The average time consumption for each instance is presented in Table 8. As seen, *STO* uses much more time for solving the instances. As for the model *DET*, instance D1 is again the most time consuming while D7 and D9 are very fast to solve. The maximum time consumption for a single TBP (Max) shows that all the instances except D9 have at least one run of STO_{single} where the time limit of 120 seconds is reached. This indicates that the optimal solution is not found. The Max times are larger than the limit due to the time usage for the initial simulation. However, for instance D1, the limit is on average exceeded by 8 seconds due to the presolve in GAMS that fails to finish within the time limit. It would be interesting to study how often *STO* reaches the time limit, and how severe it is. This have not been done and is only weakly indicated by the total time.

	D1	D2	D3	D4	D5	D6	D7	D8	D9
Total	04:44:36	01:32:52	01:15:32	02:11:20	01:08:26	01:14:38	00:18:37	01:03:38	00:10:34
Max	128.58	121.16	121.37	121.52	121.26	121.00	120.63	120.93	56.83

Table 8: Average time consumption for *STO*. Total [hh:mm:ss] denotes the total time consumption for solving an instance. Max [s] denotes the maximum time consumption for a STO_{single}

It can be seen that the stochastic model improves the solutions in all but one instance, despite the fact that only 10 scenarios are considered, and despite reaching the time limit for some instances. Note that the improvement comes at the cost of a large increase in computation time.

The improved solutions were mainly achieved due to an increase in robustness ensuring a much better capacity use at the infeed areas. Unfortunately, only a small improvement was observed for the *missed probability* term. This indicates that *missed probability* is already being handled quite well. The largest drawback of *STO* is the small number of scenarios that can be handled due to the tight time limit.

7.3 Results, semi-stochastic model

The model *SEMI* has been tested on the same nine instances as *STO* and *DET* using 100 scenarios and time span Ω of 10 minutes. Each of the nine instances is solved five times. The detailed results are reported in [15].

The average objective function values for *SEMI* can be seen in Table 9. Here they are compared with the average for *STO* and *LB*. The table also includes the percentage difference from *STO* to *SEMI* denoted by $Diff_1$ and the difference from *SEMI* to *LB* denoted by $Diff_2$.

Instance	*STO*	$Diff_1$	*SEMI*	$Diff_2$	*LB*
D1	417 050	1.51%	410 736	8.36%	376 394
D2	359 322	0.65%	356 980	2.32%	348 684
D3	458 192	-0.12%	458 747	3.97%	440 534
D4	388 027	1.41%	382 568	4.38%	365 815
D5	348 734	0.25%	347 855	3.34%	336 251
D6	264 868	2.09%	259 337	3.46%	250 373
D7	246 209	0.70%	244 494	3.85%	235 084
D8	277 070	0.61%	275 379	3.68%	265 248
D9	305 618	0.62%	303 727	2.92%	294 846

Table 9: Average objective function values of *STO* and *SEMI*. $Diff_1$ specifies the percentage difference between *STO* and *SEMI*. $Diff_2$ specifies the percentage difference between *SEMI* and *LB*.

The table shows that *SEMI* on average finds a better solution than *STO* in eight out of the nine instances. The only instance were *SEMI* performs worse is instance D3. The exact same instance showed worse performance of the *STO* compared to *DET*. Apparently *SEMI* was not able to improve what went wrong using *STO* for D3. Despite this, the percentage improvements for *SEMI* range from -0.12% to 2.09%, reducing the gap further towards the *LB*.

The improvement can be translated into an equivalent number of saved bags. Again using the assumption that the weights in the objective reflect the actual relation between the terms. The value of the improvement in *SEMI* corresponds to an average of 7 extra bags saved daily compared to *STO*. This means an improvement corresponding to 43 bags saved daily compared to *DET*. In relation to the improvement of around 20 bags found in [7] this is a very promising result.

For confidentiality reasons we cannot disclose how many bags were missed on the dates considered in our tests. But according to [18], around 8.83 out of 1000 bags were mishandled in 2012. With an average of 36 000 bags a day in the nine instances, this corresponds to 320 missed

bags per day. Here, 43 saved bags corresponds to a saving of 13.5%. Note that this is a very rough estimate, based on many assumptions.

Though the results are improved, *SEMI* did not manage to reduce the fluctuations in the results. Again the different solutions for each instance are fluctuating. This seems to be more related to the structure of the problem than the solution method and the number of scenarios.

	D1	D2	D3	D4	D5	D6	D7	D8	D9
Total	02:08:52	00:43:36	00:32:36	00:47:12	00:24:05	00:27:47	00:11:33	00:26:16	00:12:20
Max	122.50	124.64	122.71	122.58	122.41	122.12	18.44	121.85	26.03

Table 10: Average time consumptions for *SEMI*. Total [hh:mm:ss] denotes the total time consumption for solving an instance. Max [s] denotes the maximum time consumption for a single TBP

Table 10 shows the average time consumption for each instance. It can be seen that the maximum time usage of 120 seconds is reached in 7 of the instances. This is one fewer than for *STO*. Despite the significant increase in the number of scenarios, the total solution time is reduced by over 50 %. This also indicates that the time limit for single TBPs is met less often. Both this and the increased accuracy due to more scenarios should result in overall better solutions.

Despite the promising results, the change from *STO* to *SEMI* also introduces a possible shortcoming. Flights arriving long time before their EON may surpass the timespan Ω. The arrival time of these flights will be handled deterministic, and hence deviations can cause congestions. Such flights will definitely have a negative effect on the solution quality, however this will only happen rarely if Ω is chosen sufficiently large.

It can be concluded that using *SEMI* gives better and slightly more robust results than *STO*. This improvement can mainly be contributed to the increase in the number of scenarios which further increase the robustness. Due to the hybrid formulation, an increased number of scenarios can be solved, ensured by the fewer stochastic assignments. The reduction in the number of stochastic assignment apparently has a very limited negative effect on the solution quality. On top of the improvements the computation time is reduced by around 50 %.

8 Conclusion

In this paper the transfer baggage problem has been studied. The data analysis revealed that the schedule is influenced by many uncertainties.

The currently used deterministic solution method *DET* was formulated as a *MILP*. The model includes the handling and implementation of the relatively complex process chain of the transfer baggage. The performance of *DET* was tested on real life data and compared to a lower bound. From this it was observed that the capacity use deviated a lot from what was expected by the model. This demonstrated a big lack of robustness in the solution and showed that much can be gained by handling uncertainties.

The two developed improvement methods were both based on the modified *DET* model and simulations. In order to develop the simulation a data analysis of the updates on EONs were conducted. This showed that the inaccuracy in the estimated on-block times (EON) are following

a T-location-scale distribution. Also the inaccuracy in the EONs proved to be dependent on the expected time to on-block. The analysis resulted in a tool to simulate EONs, based on different criteria. With this tool the stochastic model (*STO*) was developed. This formulation experienced difficulties in form of large increases in the solution time, even for a small number of scenarios. But despite the fact that it was solved using only 10 scenarios, it managed to improve the solutions in 8 out of 9 instances. An average decrease in the objective function of 3.65% was observed. This corresponds to 36 missed bags or a decrease in the gap to the lower bound by 43%. The drawback of *STO* was identified to be the high complexity, heavily limiting the possible number of scenarios. To deal with this drawback a semi-stochastic model *SEMI* was developed as a hybrid between *DET* and *STO*. With only some of the flights modeled stochastically it was possible to increase the number of scenarios considered by a factor of 10. This lead to further improvement in of 8 out of 9 instances, while decreasing the solution time by 50%. On average the objective function was further improved by 0.8 percentage points, which corresponds to 43 missed bags. The gain mainly lies in the improved robustness regarding capacity use.

All solution methods were challenged by very fluctuating results. This made parameter tuning very difficult. The fluctuations were partly caused by the solution methods but also highly related to the uncertainties and the structure of the problem. Despite the fluctuations, the found improvements were definitely significant. All in all the project was successful in improving the planning of baggage handling operation at Frankfurt Airport.

On a more theoretical side, the proposed semi-stochastic framework may be a promising approach for solving a variety of stochastic programming problems using a rolling horizon. In such problems the most urgent decisions are more important than the later decisions, and hence a variable precision can be used. Due to the simplicity, and the promising results, the semi-stochastic framework may be an alternative to more classic scenario reduction techniques.

Conflict of interest

On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

[1] Norman Ashford, Pierre Coutu, and John Beasley. *Airport Operations*. McGraw-Hill, 2013.

[2] Torben Barth. A model for the transfer baggage problem at airports. In *International Annual Conference of the German Operations Research Society in Hannover*, page 149, 2012.

[3] Torben Barth. Optimal assignment of incoming flights to baggage carousels at airports. Technical Report Report 5, DTU Management Engineering, 2013.

[4] Torben Barth. *Optimization of Baggage Handling at Airports*. PhD thesis, Department of Management Engineering, Technical University of Denmark (DTU), 2013.

[5] Torben Barth, Frauke Böckmann, and Alexander Schäfer. Baggage scheduling at airports - a framework for efficiently allocating resources to flight events. In *6th Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA 2013)*, pages 599 – 602, 2013.

[6] Torben Barth and M. Franz. Gemischt-ganzzahlige Optimierung in Echtzeit am Beispiel der Steuerung des Transfergepäckumschlags am Frankfurter Flughafen (In German). In Dirk Mattfeld, editor, *Gemeinsame Fachtagung der Gesellschaft für Operations Research und der Deutschen Gesellschaft für System Dynamics*, pages 38–66. Braunschweig: Institut für Wirtschaftsinformatik, 2009, 2008.

[7] Torben Barth and David Pisinger. Short transfer baggage handling in an airport. In Torben Barth, editor, *PhD thesis: Optimization of Baggage Handling at Airports*, chapter 10, pages 125–141. Department of Management Engineering, Technical University of Denmark (DTU), 2013.

[8] John R. Birge and Fraois Louveaux. *Introduction to Stochastic Programming*. Springer, 2011.

[9] Frauke Böckmann, Alexander Schäfer, and Torben Barth. Optimization solutions at Frankfurt Airport. In *International Annual Conference of the German Operations Research Society in Hannover*, page 196, 2012.

[10] Tommy Clausen and David Pisinger. Dynamic routing of short transfer baggage. Technical Report 10, DTU Management Engineering, 2010.

[11] Jitka Dupačová, Giorgio Consigli, and Stein W. Wallace. Scenarios for multistage stochastic programs. *Annals of Operations Research*, pages 25–53, 2000.

[12] Jitka Dupačová, Nicole Gröwe-Kuska, and Werner Römisch. Scenario reduction in stochastic programming: An approach using probability metrics. *Mathematical Programming, Series A*, pages 493–511, 2003.

[13] Markus Frey. *Models and Methods for Optimizing Baggage Handling at Airports*. TUM-Bibliothek, 2015. PhD thesis.

[14] Holger Heitsch and Werner Römisch. Scenario reduction algorithms in stochastic programming. *Computational Optimization and Applications*, pages 187–206, 2003.

[15] Janus Timler Holm and Jakob Lindorff Larsen. Robust optimization of baggage handling in Frankfurt airport. Technical report, Technical University of Denmark (DTU), 2014.

[16] Kasper Dybkjaer Hounsgaard and Tor Fog Justesen. Implementing an improved chute allocation at Copenhagen airports. In *International Conference for Airport Operations Management in Munich*, pages 51 – 55, 2012.

[17] Ferdinand Kiermaier and Rainer Kolisch. Planning the transfer baggage handling at airports. In *International Conference for Airport Operations Management in Munich*, pages 23 – 26, 2012.

[18] SITA. Baggage report 2018. Technical report, www.sita.aero, 2018.

Appendix A: List of Symbols

In this section we give an overview of all sets, parameters and variables used in the various models.

Sets

A, set of feasible assignments of trips.

B, set of trips.

I, set of infeed areas.

$A_b \subseteq A$, subset of feasible assignments for a specific trip $b \in B$.

$A_i \subseteq A$, subset of feasible assignments for a specific infeed area $i \in I$.

$A^{STO} \subseteq A$, subset of assignments, defining the stochastic assignments.

$Bs \subseteq B$, subset of trips, where only one container is used in the trip.

$N_{b'} \subseteq B$, subset of trips, defining neighboring trips of $b' \in B$.

T, set of time-steps.

$T_i^{STO} \subseteq T$, subset of time-steps, defining the time-steps where a capacity use for stochastic assignment exists for area $i \in I$.

$T_i^{last} \subseteq T_i^{STO}$, subset of stochastic time-steps for area $i \in I$. Defines the last time-step for each area included in T_i^{STO}.

$T_i' = \{t \in T \mid k_{i,t} > 0\}$, set of time-steps where the baggage infeed area $i \in I$ is active.

$T_i^* = \{t \in T_i^{STO} \mid k_{i,t} > 0\}$, set of time-steps where the baggage infeed area $i \in I$ is active and where capacity use exists for any stochastic assignment for that area.

S, set of scenarios in the stochastic optimization.

Parameters

C_i^{des}, desired maximum capacity at infeed area $i \in I$.

C_i^{tech}, technical capacity at infeed area $i \in I$.

C_i^{queue1}, workload limit of extended capacity at infeed area $i \in I$. First step of overwork function.

C_i^{queue2}, workload limit of additionally extended capacity at infeed area $i \in I$. Second step of overwork function.

$u_{a,t}$, amount of capacity used in timeslot $t \in T$ of assignment $a \in A$.

u_a^{late}, number of late bags for assignment $a \in A$.

u_a^{missed}, total connecting/missed probability of all bags in assignment $a \in A$.

u_a^{drive}, transportation time from the parking position to the infeed area for assignment $a \in A$, in seconds.

u_a^{BHS}, total combined transportation time for each bag in assignment $a \in A$ through the baggage handling system (BHS), given in seconds.

$\upsilon_{s,a,t}$, stochastic capacity use, amount of capacity used for scenario $s \in S$ in timeslot $t \in T$ of assignment $a \in A$.

$\upsilon_{s,a}^{missed}$, total missed probability of all bags in assignment $a \in A$ for scenario $s \in S$.

w^{late}, weight of cost term for late bags.

w^{missed}, weight of cost term for connecting/missed probability.

w^{drive}, weight of cost term for apron transportation time.

w^{BHS}, weight of cost term for transportation through the BHS.

w^{des}, weight of cost term for capacity use, between desired and technical capacity.

w^{queue1}, weight of cost term for capacity use, between technical and queue1.

w^{queue2}, weight of cost term for capacity use, between queue1 and queue2.

$w^{penalty}$, weigh of cost term for capacity use above queue2.

$k_{i,t}$, number of bags infeed area $i \in I$ can service when working at technical capacity in time-step $t \in T$. Reflects the amount of workers assigned. If the infeed area is closed at a given time-step then $k_{i,t} = 0$.

$L_{i,t}$, indicator parameter for T_i^{last}, equal 1 for $t \in T_i^{last}$ and 0 otherwise.

Ω, timespan defining the stochastic assignments.

λ_0, initial increase rate.

τ, tolerance value.

Variables

$x_a \in \{0,1\}$, denotes whether assignment $a \in A$ is chosen.

$y_{i,t}^{des} \in \mathbb{R}_+$, use of capacity exceeding the desired capacity use C_i^{des} at infeed area $i \in I$ in time-step $t \in T$.

$y_{i,t}^{queue1} \in \mathbb{R}_+$, workload stretched above technical capacity up to C_i^{queue1} at infeed area $i \in I$ in time-step $t \in T$.

$y_{i,t}^{queue2} \in \mathbb{R}_+$, workload stretched above technical capacity between C_i^{queue1} and C_i^{queue2} at infeed area $i \in I$ in time-step $t \in T$.

$y_{i,t}^{penalty} \in \mathbb{R}_+$, ensures feasibility even if workload exceeds C_i^{queue2} at infeed area $i \in I$ in time-step $t \in T$.

$y_{i,t} \in \mathbb{R}_+$, denotes the deterministic capacity use for area $i \in I$ in time-step $t \in T$.

$\gamma^{des}_{s,i,t} \in \mathbb{R}_+$, like $y^{des}_{i,t}$, but used for stochastic capacity use, defined for each scenario $s \in S$.

$\gamma^{queue1}_{s,i,t} \in \mathbb{R}_+$, like $y^{queue1}_{i,t}$, but used for stochastic capacity use, defined for each scenario $s \in S$.

$\gamma^{queue2}_{s,i,t} \in \mathbb{R}_+$, like $y^{queue2}_{i,t}$, but used for stochastic capacity use, defined for each scenario $s \in S$.

$\gamma^{penalty}_{s,i,t} \in \mathbb{R}_+$, like $y^{penalty}_{i,t}$, but used for stochastic capacity use, defined for each scenario $s \in S$.

$q^{avg} \in \mathbb{R}_+$, average load of all infeed areas.

$q^{plus}_i \in \mathbb{R}_+$, positive deviation from average load at infeed area $i \in I$.

$q^{minus}_i \in \mathbb{R}_+$, negative deviation from average load at infeed area $i \in I$.

$q \in \mathbb{R}_+$, total deviation from average load at infeed area.