Operational Gate Allocation Using a Sliding Time Window

Master Thesis

J. A. Borghart

TU Delft

Department Control and Operations
Delft University of Technology
November 2019

Operational gate allocation using a sliding time window

Master Thesis

by

J.A. Borghart

in partial fulfillment of the requirements for the degree of

Master of Science

in Aerospace Engineering

at the Delft University of Technology,

to be defended publicly on Thursday December 13, 2019 at 14:30 PM.

Student number:	4162293	
Date:	November 29, 2019	
Thesis committee:	ir. P. C. Roling,	TU Delft, daily supervisor
	Prof. dr. R. Curran,	TU Delft, chair
	Dr. J. Ellerbroek,	TU Delft

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

TU Delft

Preface

This study is part of the MSc degree in Aerospace Engineering at Delft University of Technology. The project is a result of collaborative activities between Delft University of Technology and Amsterdam Airport Schiphol. The subject has been proposed by the author and has been formulated as a graduation assignment in collaboration with the Delft University University of Technology. The data required for the study was provided by Amsterdam Airport Schiphol.

This report is submitted to ir. Paul Roling (assistant professor at Delft University of Technology), Prof. dr. Richard (Ricky) Curran (professor at Delft University of Technology) and dr. ir. Joost Ellerbroek (assistant professor at Delft University of Technology). This thesis describes the methodology and results of a operational gate allocation model using a sliding time in conjunction with real world flight arrival times to dynamically compute the gate reassignment needed to mitigate arising gate conflicts ahead of flights arrival.

In this report, the process of achieving the following objective is covered:

"Investigate the feasibility and potential performance improvement that can be achieved by implementing a sliding time window in the operational gate allocation model"

I would like to mention my appreciation for the assistance and guidance given by the Air Transport and Operations group of Delft University of Technology. I would like to give special thanks to my daily supervisor ir. Paul Roling and Mihaela Mitici for providing me with valuable feedback at the green light meeting. Finally, I would like to thank the faculty of Aerospace Engineering of Delft University of Technology and Amsterdam Airport Schiphol, for providing me with the opportunity to participate in this innovative and challenging project. Lastly, I would like to thank a friend for her valuable support and help with structuring and spell checking the final version of this thesis.

J.A. Borghart
Delft, November 2019

Abstract

The steady growth of air traffic volumes and the subsequent increase of congestion at major airports require airports to increase their operational efficiency. Inefficient or the absence of gate reassignment models increase the possibility and occurrence of gate blockage and at the same time reduce the comfort of passengers as they often receive the notice that they have to wait for their assigned gate to become available.

This thesis describes a model to optimally reallocate flights to available gates on the actual day of operation for a large airport, such as Amsterdam Airport Schiphol (AMS). It takes into account the scheduled arrival time of a flight for the initial assignment, during daily operation the model updates the scheduled arrival time with the estimated arrival time as soon as the flight is enroute. The proposed model allows the user to do a trade-off between delaying, gate changing and remote handling of flights.

The sliding time window has shown to reduce the number of gate changes required during daily operation in order to mitigate gate conflicts. Validation of the model with data obtained from AAS showed that the approach is feasible for real world application and that it improves the operational efficiency of the airport.

Contents

List of Figures

List of Tables

1 Introduction

With the steady growth of air traffic volumes and the subsequent increase of congestion at major airports, delayed flights are inevitable and at the same time a massive expense for companies (CAPA, 2019) and an inconvenience for passengers. As a countermeasure, airports have started to expand their operation, increasing the number of runways and terminal buildings. Some airports, restricted by land availability, have been relocated to a new location further away from the city it serves. Modern airlines using these large airports are running a highly optimized operation where they keep their planes on the ground for the shortest time possible. This means that they operate on very tight schedules, often visiting multiple airports a day, for sometimes as short as 30 minutes. Small perturbations in these tight schedules mean that they often arrive early or late at their destination. This poses problems for airports as they have a high uncertainty when aircraft will be on the ground requiring a gate for passenger de-boarding and on-boarding. An early arriving aircraft can easily conflict with a departing aircraft from the same gate, forcing airport authorities to reschedule up to multiple flights to free a gate for this arriving aircraft.

Airports are struggling with their flight to gate assignment. As more and more flights are visiting the airport daily, airports are forced to use their available gates more effectively (McGeehan, 2018; Steiner et al., 2017; Topham, 2013). In order to do this new innovative solutions have to be found to optimize the gate usage in airport operations. Better aircraft to gate assignment will lead to higher operational efficiency and yield better on-time performances with a reduction of gate reassignments (Narciso and Piera, 2015).

Most of the major airports as we know them today started life as a plain field of grass, used by small cargo airplanes delivering mainly mail across the world. Throughout the 20^{th} century these airports gradually evolved into the massive passenger hubs they are today, serving hundreds of daily flights and handling millions of passengers a year. With the increase in size, the complexity of airports increased and more advanced methods were required to assist airport personnel with the flight to gate assignment. Airports were struggling to assign each aircraft to an available gate, while trying to maximize passenger conveniences and the operational efficiency of the airport at the same time: the Aircraft Gate Assignment Problem (AGAP) was born (Steuart, 1974).

The first quantitative study concerning the AGAP, proposed a method to determine the minimal number of gates needed at an airport (Steuart, 1974). This research approached the AGAP from an airport/airline perspective, depicting the minimal number of gates required to handle all scheduled flights. The AGAP can also be approached from a passenger perspective, aiming to increase the comfort of passengers by decreasing the distance they have to walk to their flight (Braaksma, 1977).
Three main stakeholders can be identified for the AGAP: airports, airlines and passengers (Deng et al., 2017). When optimizing the AGAP, the stakeholders are commonly divided into two groups: the passengers and the airport/airlines. Objectives are then formulated to cater to that group. Early research mainly focused on the first group (Babić Obrad et al., 1984; Mangoubi and Mathaisel, 1985). After the mid 90's, the research focus shifted mainly towards the second group (Cheng, 1997; Bolat, 1999). In the past decade, however, the focus shifted again and current research is now focused on optimizing for both groups concurrently (Ding et al., 2005; Dorndorf et al., 2007).

Early research aimed to increase passenger comfort by minimizing the walking distance. IATA [1] established guidelines for airports which set maximum walking distances of 300 and 650 meters for respectively unaided and aided walking (passenger conveyance system) (Pushkarev, 1975). Inside an airport, three flows of passengers can be identified: embarking, disembarking and connecting passengers. For each of these three flows, the walking distance can be minimized. At first only the embarking and disembarking flows where considered, yielding significant improvements over the previously more or less randomly assigned flight to gate assignment (Mangoubi, 1980). As the air traffic market grew, airlines started to commence a hub and spoke operation[2]. This lead to an increase of the transfer passenger flows, which were now surpassing the embarking and disembarking flows in size.

Proceeding research aimed to increase the operational efficiency for both the airline and airport by proposing new objectives for the AGAP. Operational efficiency of airlines and airports can be increased by utilizing gates more effectively. This could be done by either: minimizing the idle times of gates (e.g. time a gate is not used), minimizing the number of off-gate assignments (e.g. remotely handled aircraft), or using stochastic delays based on historical delay data to predict delays (Yan and Tang, 2007).

Modern research, however, is taking advantage of recent developments in computing power to formulate more complex models optimizing for multiple objectives simultaneously. The last 10 years, research has been focusing on developing new methods to create robust solutions for the AGAP. A solution is defined to be robust if it is able to absorb flight schedule perturbations, and thus preventing a gate change or remotely handled aircraft (Pesch et al., 2008). Proposed methods by research have been using two different approaches, either using statistical analysis to predict delayed flights and allowing them more space in the flight to gate assignment, or by employing buffers between subsequent flights (Şeker and Noyan, 2012). The downside of these approaches is that flight delays are hard to predict, as there are many external factors involved such as weather or mechanical disruptions at a flights origin. Both these approaches have proven to provide flight to gate assignments that exhibit robustness properties. However, they do so at the cost of operational efficiency as the gate idle times are increased for flights with a high delay probability (Castaing et al., 2016).

In this thesis a new approach is proposed by creating a reassignment framework which uses real-time flight arrival times computed from estimated flight times, in combination with a sliding time window to dynamically solve the AGAP during daily operation. A sliding time window is a shifting time horizon that is used during gate reassignment (Ferland and Fortin, 1989). In the current literature this has been applied with success to the scheduling of yard cranes in a harbor (Van Dijk, 2015). The sliding time window can be applied to the AGAP for the gate reassignment model where the initial optimal gate assignment will be updated based on the estimated arrival time, which can be determined with certainty as soon as a flight is enroute. The model should be feasible to real-world applications by allowing for fewer gate changes close to the actual arrival time of the aircraft. The proposed model in this thesis divides the AGAP into two parts. The first part concerns the initial gate assignment at the start of the operation. The second part considers the real-time operation. The first part of the AGAP is solved using different robust objectives aiming to provide a robust solution for the second phase. During operation the second part of the AGAP is solved by updating the arrival and departure times used during the first part, with the actual arrival and departure times that become available as flights depart from their origin. The proposed reassignment framework uses these new times to solve the AGAP for conflicts that have occurred due to perturbations in the flight schedule. The model will first be tested against a self-generated data set, the theoretical case. The best performing configuration of the model from the theoretical case will be validated using real-life flight data obtained from Amsterdam Airport Schiphol (AAS), the practical case.

1.1. Thesis outline

In chapter 2 an overview of the literature and research that has been published regarding the AGAP is presented, the chapter describes the different solution techniques that can be used to solve the AGAP and applications of a sliding time window. In chapter 3 an overview of the research framework, problem statement and the corresponding research questions that will be answered in this thesis are shown. In chapter 4 the data collection and the data analysis performed is described. Next, it describes the process of creating an

[1] International Air Transport Association

[2] A hub and spoke operation uses a central airport (hub) inside the route network of an airline at which passengers are transferring between flights to reach their final destination

aircraft to gate assignment framework,in further detail. In chapter 5 the outcomes of the theoretical case will be discussed. In chapter 6 the model outcomes for the practical case are discussed. In this case, the model is validated using real-world data obtained from AAS. Finally, chapter 7 will include a discussion, conclusion and future recommendations for further research.

2

Literature Review

Over the past 50 years a plethora of work has been published regarding the AGAP. In this section, a brief overview of the literature published regarding the AGAP are discussed. In section 2.1, the historical developments in research efforts for the AGAP will be placed on a timeline. In addition, it will briefly discuss the changes in optimization objectives regarding the AGAP during time. In this section the papers published before the year 2008 are discussed, the subsequent sections will consider later work. In section 2.2, different approaches to increase the robustness of solutions found for the AGAP are discussed. In section 2.3 the different strategies and proposed models to reassign flights to gates based on flight schedule perturbations are discussed. In section 2.4, the most commonly used MIP formulation for the AGAP and the solution strategies used to solve the proposed models are discussed. Finally section 2.6, discusses the use of a sliding time window in mixed integer programming.

2.1. Historical development of the AGAP

During the 1970's air traffic saw a steady growth in the number of passengers, doubling to roughly 650.000 passengers worldwide (Group, 2017). This sharp growth was the consequence of larger airplanes, the increased number of aircraft in operation and the adoption of jet liners which were able to carry more passengers over a greater distance. The increase in aircraft size meant that airports had to expand. During this time period the AGAP was first quantified by (Steuart, 1974), and a stochastic model was proposed to quantify the minimum number of gates required at an airport.

Two distinct stakeholders can be identified for the AGAP, each with their own main objective:

1. *Passengers*: the main objective for passengers is to increase their comfort by reducing the walking distance
2. *Airport and airlines*: the main objective for airports and airlines is to increase operational efficiency by reducing cost and/or increasing the effectiveness of equipment

In the literature there is a clear difference of interest between the main two stakeholders, in the beginning research was focusing on optimizing for passenger comfort. This shifted during the 1990's towards optimizing for operational efficiency. Approaching the year 2000, computing power had significantly increased allowing for multi-objective formulations of the AGAP. The following subsections will first provide a brief overview of the literature discussing the passenger comfort objective, followed by a brief overview of the literature discussing the operational efficiency objective.

2.1.1. Optimizing for passenger comfort

The main factor influencing the comfort of passengers regarding the AGAP is the distance they have to walk between flights. Pushkarev (1975) found that people feel comfortable walking up to 300 meters in facilities.

The IATA[1] took these numbers and defined guidelines for airport designers regarding the maximum distances passengers should walk inside an airport. They prescribe that a passenger should not walk more than 300 meters unaided, and no more than 650 meters aided (by implementing passenger conveyance systems) (Airport Cooperative Research Program, 2012). Airport designers use these guidelines during the design process of the airports terminal and gate layout. However, in practice the actual distances passengers have to walk depend on the assignment of flights to the available gates.
In order to optimize for passenger comfort, one has to consider the three major passenger flows present in an airport; embarking, disembarking and transfer passengers. Early models considered only the first two, as the inclusion of transfer passengers leads to a harder to solve Quadratic Assignment Problem (QAP) (Obata, 1979). The problem of assigning aircraft to gates is relatively simply to formulate, as there are only two constraints required for the most basic form:

1. Each flight should be assigned to a gate
2. Each gate should be assigned only one flight at a time, with no overlap in gate occupancy time

An early study by Braaksma (1977), was the first to apply optimization to the gate assignment. Arriving, departing and transferring flows for both international and domestic passengers were studied. The results showed that the current gate assignment for Toronto Airport could be improved such that the average walking distance for each passenger was reduced by up to 57%. While these results were promising, in order to solve the problem systematically it was formulated as a linear problem. The aim was to minimize the sum of the total walking distance for the passengers. Babić Obrad et al. (1984) created a model to solve this problem and found that the optimum solution for this model distributed the walking times more evenly over the passengers and thus reduced the average walking distance. Similar research by Mangoubi and Mathaisel (1985) aimed to reduce the computing time. The study proposed an heuristic[2] which assigns the most crowded aircraft to the gate with the shortest walking distance, reducing computing cost by a factor 40. Since a heuristic was employed, an optimum solution was not guaranteed. The found solution yielded a 27% reduction in walking distance compared to the 32% achieved by the optimum solution.

In the research mentioned above, two distinct flows were modeled: embarking and disembarking passengers. This means that for each flight to gate assignment, there is an associated walking distance all passengers have to walk. However, in practice, the larger airports are serving as a hub for a major airline. Here the majority of passengers flying this airline are transferring to another flight. Inserting the transfer passenger flow into the AGAP leads to a harder to solve QAP, where there is a set of n facilities (aircraft) and a set of n locations (gates). For each pair of locations there is an associated walking distance and for each pair of facilities there is a flow (transfer passengers). The goal is to minimize the sum of the walking distance between each pair of facilities multiplied with the actual flow of passengers. A model formulating the AGAP as a QAP was first presented by Haghani and Chen (1998). Reformulating the model into a QAP lead to the problem being NP-hard[3]. In order to solve such a model, an efficient heuristic is required as the exact solution takes a long time to compute (Obata, 1979). In addition to the inclusion of the transfer passengers, Haghani and Chen (1998) were the first to introduce a time constraint were each flight was split into blocks of 15 minutes. A flight can occupy multiple blocks, indicating their presence at a gate and blocking that gate for other flights. The proposed heuristic tries to assign two consecutive flights with a large flow of passengers to the same gate since there is no overlap in gate occupancy time.

Previously described models considered the allocation of a flight to a gate to be instantaneous. This means that a flight is assigned to a gate as soon as it arrives at the airport. However, in real-life a flight can be delayed. Zhu et al. (2003); Lim et al. (2005) proposed a method where flights can be delayed, at a cost. This presents the opportunity to assign each flight a time-window in which the flight must be at a gate. This time is defined as the flight specific gate occupancy time. With this approach a more optimum gate assignment with a minimum walking distance for the passengers and associated delay cost can be reached.

[1] International Air Transport Association
[2] An heuristic is an approach used in problem solving using a practical or logical method to find a solution that is not guaranteed to be optimal. But is sufficiently acurate for the usecase.
[3] NP-hard means that the problem is not solvable in polynomial time

2.1.2. Optimizing for operational efficiency by minimizing the off-gate assignment

Optimizing the AGAP to increase the operational efficiency has the interest of both airlines and airports. Increasing the operational efficiency of an airport can have multiple effects on the operation depending on the objective. A commonly used objective in early research was to increase the efficiency by minimizing the number of off-gate assignments. Minimizing the number of off-gate assignments or remote handled aircraft, increases the operational efficiency by reducing the number of towing actions required for an airport, thus lowering the cost to handle an airplane. An added benefit is that it increases the comfort for the passengers as they are not required to take a bus over the apron to reach their aircraft. Early research describes the successful implementation of a linear model to the AirFrance Terminal at Paris Charles de Gaulle airport. The published research, however, lacks details about the implementation (Hamon and Weissert, 1980). A similar study by Vanderstraeten and Bergeron (1988) describes a linear model with binary variables. The model creates events for each flight, an event prescribes the embarking or disembarking of an aircraft in time. Each event prescribes whether it is possible to tow the aircraft away from the terminal to a remote parking position if the on-ground time is sufficient between both events. The model has been tested on data obtained from Air Canada's terminal at Toronto Pearson International Airport. The terminal has 27 gates and handled roughly 150 flights, resulting in 300 events. While it is physically impossible to assigns all the events to a gate, the model was solved to near optimality by using a heuristic. It was found that the proposed model was able to reduce the number of off-gate handled events from 50 down to 10.

More recent research has attempted to improve the solution technique by proposing a hybrid approach allowing for the model to be solved to optimality much quicker. The proposed method by Ding et al. (2004, 2005) splits the AGAP into two distinct stages. In the first stage the flights are assigned to gates using a simple greedy heuristic. The found solution is then further solved via an algorithm to obtain an exact and optimal solution. The heuristic sorts all the flights according to their departure time. Each flight is then assigned to an available gate with the latest arrival time. If such a gate does not exist, then the flight is assigned to a remote parking spot. The found solution by the heuristic is then optimized to optimality by an hybrid approach combining Simulated Annealing (SA) and Tabu Search (TS). It was found that SA could reduce the objective function greatly in a short amount of time after which it stagnates. TS on the other hand was able to reduce the objective steadily but slowly. Therefore a hybrid approach was proposed which uses the high initial effectiveness of SA combined with the steady improvement over time achieved by TS. The approach has been tested fictional data using a generic airport layout and computer generated time schedules for flights and passenger flows.

2.2. Robust solutions for the Aircraft to Gate Assignment problem

Robustness for the AGAP is defined as the ability of a found solution to absorb disturbances in the flight schedule with minimal to no rescheduling in the flight to gate assignment (Pesch et al., 2008). In Figure 2.1, an example of a robust and non robust gate assignment is given. Robustness for the AGAP is desired by both airlines and airports, as a more robust assignment sees fewer gate re-assignments due to aircraft delays. In the previously discussed literature, the timetables are assumed to be fixed and aircraft operate accordingly. In practice however, aircraft are susceptible to both arrival and departure delays. This can have multiple reasons and can both be positive or negative for the passengers, arriving early or late respectively. For airports, both delays are negative as it requires them to reschedule the flight if, due to the delay, it coincides with another flight assigned to the same gate. Often these mutations to the flight schedule are last minute and put a lot of time pressure on the airport operation staff. Airports are therefore highly motivated to address this issue by finding robust solutions for the AGAP. In literature there have been multiple approaches proposed to achieve these robust solutions.

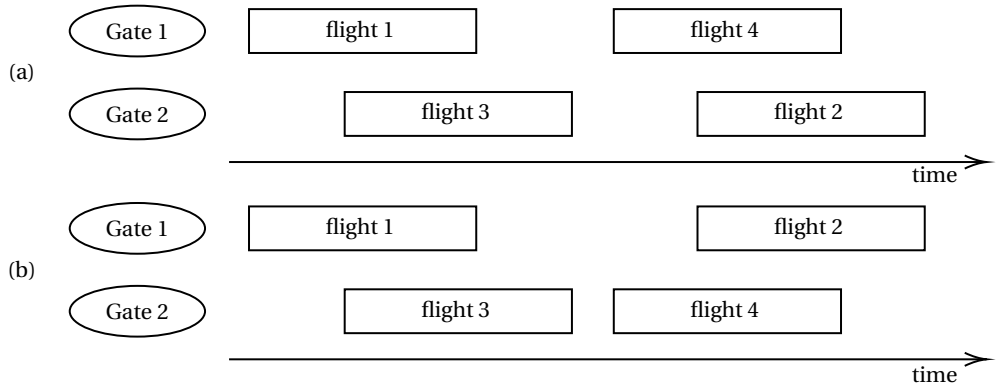

Figure 2.1: Example of a robust solution (a) and a non-robust solution (b) for the AGAP

2.2.1. Stochastic flight delays and historical data

The first proposed approach aiming to achieve a robust solution for the AGAP used stochastic flight delays and historical data. During real-time operations the flight schedule is subjected to stochastic and semi-stochastic delays. Some flights are more prone to delays than other flights and often historical data is used to create representative, flight specific stochastic delays.
An early proposed model by Yan and Tang (2007), used pure stochastic delays in a framework to create a robust solution for the AGAP. The model consists of a planning stage which uses a flight schedule on which n random flight delay scenarios are applied. Then the AGAP is solved and this solution is updated using a reassignment model which applies the delay scenarios to the flight schedule. An objective value is found which is compared to previous runs. Based on this result the weights for the initial model are altered and the process is repeated. The model was tested on a real flight schedule for Taiwan Taoyuan International Airport, spanning 172 flights and 29 active gates. Current practice at the airport is to create an optimal gate assignment the day before operation. Then during the day, flights are reassigned by hand if gate conflicts occur due to flights schedule perturbations. The advantage of the proposed model is that during the initial assignment it already incorporates stochastic flight delays, leading to a more robust initial solution. The model was found to yield between 40% and 60% improvement on average for passenger waiting times (Yan and Tang, 2007).

2.2.2. Buffer times between flights

The second proposed approach aiming to create a robust solution used the idle time or buffer time of gates between consecutive flights. Buffer times between consecutive flights assigned to the same gate are used in order for the flight to gate assigment to absorb a delayed flight instead of reassigning the subsequent flight to another gate. An early approach taken, was to distribute the idle time evenly over the gates. The idle time is defined as the time in between flights when the gate is not in use. Minimizing the idle times of gates or maximizing the gate usage will increase the utilization of resources and thus the operational efficiency. Bolat (1996, 1999), proposed a method in which idle time periods are distributed evenly across the gates. In case of disruption, the assignment is then flexible enough to absorb flight delays at the gate instead of having to reassign the delayed flight to a new gate or remote area. An important assumption in this paper is that the possibility of a scheduled flight to be delayed is independent from the gate assignment. The possibility for a flight to be delayed is thus independent and identically distributed. From this assumption it follows that a gate assignment with evenly distributed idle times will exhibit robust properties. When applied to artificial and actual data obtained from Terminal 2 at Riyadh's King Khaled International Airport, spanning 8 gates and accommodating roughly 75 daily flights, the model on average reduced the number of remotely served aircraft from 72.03% to 54.28%. A drawback of the above described method is that, depending on the flight schedule, the idle times are not evenly distributed between the flights. Some gates see a very long idle time at the start of the day and then a couple of tightly scheduled flights. This issue was solved by reformulating the objective function and minimizing for the variance of idle times (Bolat, 2000). This approach is minimizing the total squared buffer time, which is as a consequence heavily penalizing larger buffers. An improvement was suggested by Diepen et al. (2012) by changing the objective or cost function to a form closer resembling the choices an airport gate scheduler could make. The new cost function penalizes assignments with

a large conflict probability (e.g. small buffer) and favor assignments with a small conflict probability (e.g. large buffer). This improved model has been verified with data obtained from Amsterdam Airport Schiphol , spanning 120 gates and accommodating 600 flights.

Airports typically have to balance the conflicting requirements of all stakeholders in the AGAP, therefore multi-objective formulations have been developed. A clique partitioning model (CPM) was proposed by Pesch et al. (2008). This model uses a reference flight schedule to allow recurring flights to be assigned to the same gate. Dorndorf et al. (2012) improved this CPM by implementing a new heuristic allowing the model to be solved for multiple time periods. The main objectives of this model are:

1. Maximize the assignment preference score: e.g. each possible assignment is scored based on operator and airport preference
2. Minimize the number of remote assigned flights
3. Minimize the number of towing operations: e.g. aircraft that need to be towed away from the gate between disembarking and embarking
4. Minimize the deviation from the reference schedule

The robustness measure for the above objective is the amount of buffer time between flights. If the buffer time is below a set number it is penalized, there is no penalty for buffers above this set number.

2.2.3. Optimizing the number of stands and gate strategies

Finally, airports can optimize their operational efficiency and thus robustness of their flight to gate assignment by determining the minimum number of stands required. Narciso and Piera (2015) extended the first study by Steuart (1974) by determining the minimum number of gates required at an airport to handle in- and outgoing traffic. In order to do this, two different strategies regarding gate usage can be identified:

1. Non-Preemptive: a gate is assigned to an arriving flight and is only released when the flight leaves the gate. This strategy does not allow for rescheduling of flights
2. Preemptive: a gate is assigned to an arriving flight and can be reassigned if the flight has a delay. This strategy allows for rescheduling of flights in case of delays.

Both of these strategies have been tested using simulations to determine the optimal strategy for handling flight delays. For small delays (0 - 10 minutes) the non-preemptive strategy requires less gates. For larger delays (10 - 20 minutes) the preemptive strategy requires less gates (Narciso and Piera, 2015).

2.3. Gate re-assignment models

In modern aviation, gate re-assignments are inevitable as congestion increases due to an increasing number of flight movements. The in the previous section discussed approaches for creating a robust initial assignment are proven to work well for small deviations to the flight schedule. However, if deviations are sufficiently larger or not in line with historical data, the airport authorities are forced to perform gate re-assignments to resolve gate conflicts as the flight to gate assignment is unable to absorb the delay. A gate re-assignment is the act of changing the gate to which the aircraft was previously assigned. Compared to the initial gate assignment model, used to create the gate assignment for the start of the day, there are five additional factors that need to be considered in order to define gate re-assignment models(Tang, 2011):

1. Flight delay is important as a flight can be held waiting on the apron for the gate to become available.
2. A flight can complete boarding ahead of schedule, allowing the gate to be re-used.
3. Due to delay transfer passengers might risk missing their connection if no proper re-assignment is found in time.
4. Solution time should be short as the operation requires a solution minutes after the disruption occurred.

5. Minimal deviation from a reference schedule is required as airport services require preparation time before an aircraft arrives.

Flight delays are difficult to predict due to their stochastic nature and depending on the location of the airport, they can occur frequent. Tang et al. (2010) researched flight delays at Taiwan Taoyuan International Airport (TTY) and concluded that on a day of operation as many as 80% of the flights experienced a delay between 1 and 20 minutes. Flights at TTY are manually re-assigned by airport personnel, a time-consuming operation which is prone to human errors. It is therefore required to solve the problem real-time in a systematical manner using automation, reducing the number of human errors. Tang et al. (2010) proposed a model that re-assigns aircraft based on gate availability and conflicts arising due to delayed aircraft. If an aircraft is delayed and can therefore not be assigned to its initial gate, it is re-assigned to a new gate. The amount of minutes a flight is delayed is defined as the time inconsistency. As the aircraft is assigned to a new gate, the space inconsistency is set to be 30 (fixed number determined with TTY staff). The objective function of the model aims to minimize the total sum of both the time and space inconsistency.

The model has been tested with data obtained from TTY airport, the data set spanned 23 gates, accommodating 171 flights. It was found that the solution value for the objective function of the model was 14% lower compared to the manual approach (390 vs 420). This indicated that the model out performed the manual method. In addition, it is interesting to note that the model preferred to slightly delay aircraft and have them wait on the apron for the original assigned gate to become available. This is preventing a "domino effect" of delays which can occur if an early arriving aircraft is re-assigned to a new gate. The model assigned 8 flights to a remote gate, 13 flights to alternative gates and delayed 2 flights. The manual approach assigned 5 flights to a remote gate, 8 flights to an alternative gate and delayed 7 flights.
The model was developed further for TTY airport to be able to re-assign flights during periods of gate shortages and stochastic delays (Tang, 2011).

The papers discussed above, do incorporate transferring passenger flows between aircraft but they do not optimize for the convenience of these passengers. At major hub airports, up to 25% of the total passenger can belong to the transferring passengers flow (Chung et al., 2017). It is therefore of importance that these flows are included into the optimization objective since passengers that miss their connection are costly for the airline. Zhang and Klabjan (2017), proposed a model which reformulates the model to a multi-objective, multi-commodity flow network. The four objectives of the model are to minimize; total flight delay, number of gate-reassignments, total passenger transfer distance and the number of missed connections. In order to optimize for these objectives and to resolve the gate conflicts, the model has the following four options to choose from when searching for a solution:

1. Re-assign an aircraft to another gate or remote parking area
2. Hold the aircraft on the apron for it to wait for the assigned gate to become available
3. Delay a departing flight in order to wait for the transfer passengers
4. Tow the aircraft away from the gate between embarking and disembarking

The model has been tested on a data set obtained from the hub airport of a major US carrier spanning 3 days of operations with roughly 50 gates, accommodating 1075 daily flights. Results showed an improved solution over the current method. Run-time on the 3 hour time window scenario was fast, below 20 seconds, thus making the model suitable for real-world cases.

2.4. Solution techniques

With the rise of the computer and the subsequent increase in computing power, it became possible to solve large and complex problems within reasonable time with the help of a computer. For the AGAP this meant that it became possible to create increasingly complex models that accurately represent situations present at real world airports. For the early research this meant that the statistical work performed by Steuart (1974) could be implemented as a linear model and solved by the computer. Babić Obrad et al. (1984), was the first to formulate the AGAP as a Mixed Integer Linear Programming (MILP) and used a computer to solve the model. Linear programming is a method to solve mathematical models, where it is required to find a minimum or a maximum for a defined objective metric. The requirements for the model are given by linear

relationships. The objective which is either minimized or maximized is linear and subjected to equality or inequality constraints. Depending on the type of constraints the category of the model is changed. Mixed Integer Linear Programming or MILP contain only linear constraints, Mixed Integer Quadratic Programming or MIQP has a quadratic objective but still linear constraints. In literature this is also referred to as a Quadratic assignment problem or QAP. Mixed Integer Quadratic Constrained Programming or MIQCP is similar to the MIQP but now the constraints are also quadratic. The described formulations for linear programming are commonly referred to as Mixed Integer Programming or MIP.

The MIP formulation is, even today, the most commonly used formulation for the AGAP (Bouras et al., 2014) due to the minimization or maximization objectives and the linear or quadratic formulation of the constraints. The following section will further elaborate on the MIP formulations and the solution techniques used for the AGAP.

2.4.1. MIP formulation

Over the past 50 years, a lot of research has been conducted regarding the AGAP. Dorndorf et al. (2007) and Bouras et al. (2014) created an overview of the contributions made towards the AGAP. Both these review papers show that the majority of models are formulated by Mixed Integer Programming (MIP). The objective (single objective optimization) or objectives (multi objective optimization) can then either be Linear (MILP) or Quadratic (QAP). Models that include flows between the to be assigned objects such as transferring passengers are by definition quadratic. Solving these models is done by using mathematical optimization software. Currently, the largest commercially available solvers on the market are Gurobi[4] and CPLEX[5]. It is difficult to compare the solvers as they can perform very different depending on the models used. Meindl and Templ (2012) provides a performance overview of a large set of solvers for a variety of models. The results conclude that Gurobi and CPLEX are among the fastest solvers available on the market.
Both Gurobi and CPLEX use a version of the Branch and Bound algorithm to solve MIP problems (Land and Doig, 1960). This algorithm creates a search tree of nodes. Each node is representing a new MIP problem obtained from the main MIP by using integer relaxation. The solver then visits each node in the three from top to bottom. If the lower node is not providing a better solution than the node above, the lower part of the branch is discarded. For problems spanning many decision variables this can lead to very large search trees containing thousands of nodes. Exploring all these nodes, as solvers such as Gurobi and CPLEX do to find the optimal solution, can take a long time depending on the size of the problem and the convergence of the problem. Recent research has therefore been focusing on heuristics (Guépet et al., 2015). A heuristic is defined as a method or algorithm used in problem solving to find a solution that is not optimal but close to optimal. The found near optimal solution from the heuristic can be used to greatly reduce the number of nodes in the search three and thus accelerate the process of finding an optimal solution. An example of a tree of nodes can be seen in Figure 2.2[6]

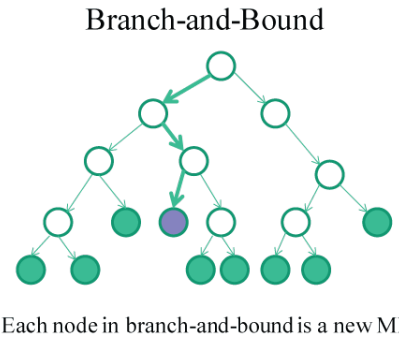

Figure 2.2: Example of branch and bound algorithm tree of nodes

Heuristics can either be applicable to generic MIP problems or can be problem specific. General purpose solvers such as Gurobi and CPLEX have their own set of heuristics that are used to presolve the problem.

[4]https://www.gurobi.com
[5]https://www.ibm.com/analytics/cplex-optimizer
[6]Retrieved from: https://www.gurobi.com/resource/mip-basics

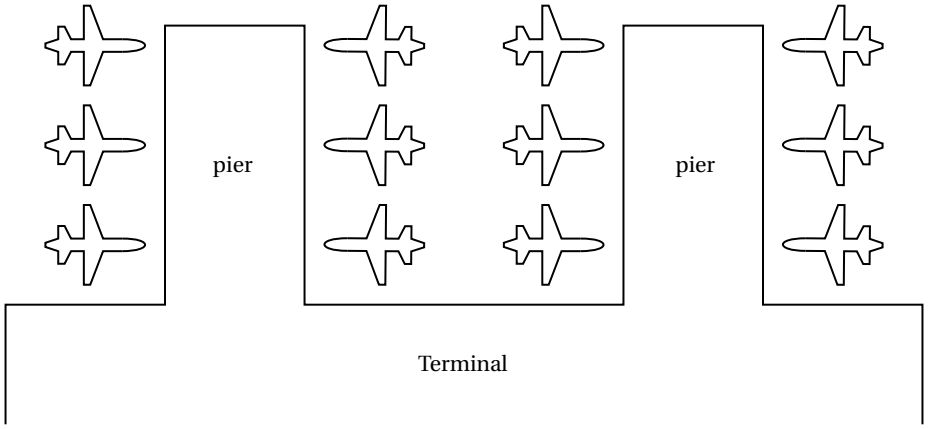

Figure 2.3: Example of a terminal and finger layout

However, in some cases problem specific heuristics can greatly outperform the more generic heuristics for MIP problems implemented in solvers. The above cited review papers provide an overview of the by literature proposed heuristics specific to the corresponding model for solving the AGAP.

The majority of literature formulates the AGAP as a MIP due to the minimization or maximization nature. However, there have been multiple publications that explored other formulations for the AGAP. These formulations include, but are not limited to, Evolutionary Algorithms (Deng et al., 2017; Dell'Orco et al., 2017). Although interesting, these algorithms are out of scope for this research.

2.5. Generation of realistic flight schedules and delays

Depending on the nature of the research it is not always feasible to use a real world data set. Often it is decided that by using an theoretical data set the improvements of the proposed model can be demonstrated more clearly. In other cases there simply is no real world data set available to the researchers. Research that is using a theoretical data set follows a similar approach to generate the data set. First an artificial airport is defined. Often these use the terminal and finger layout as seen in Figure 2.3. A benefit of this popular layout, which is commonly found at major airport hubs, is that it reduces walking distances for connecting passengers compared to other layouts (Belobaba et al., 2015). It should be noted that none of the papers mentioned in this section aim to create a very accurate time schedule as the main purpose of the generated data set is to demonstrate that the proposed model improves current practices. Methods used to generate the time schedules are therefore kept simple. Discrete uniform distributions are used to determine the time between flights, number of passengers and for the different flows considered by the model (Bihr, 1990; Ding et al., 2004; Lim et al., 2005). Models that aim to improve the gate reassignment models used in the AGAP extend these generated time schedules with a delay component. Genç et al. (2012) used a normal distribution from which the delays were taken for each flight in the flight schedule. Şeker and Noyan (2012) used a skewed triangular distribution for which is argued that it is better at representing a real wold situation where flights are more likely to arrive late than arrive early. In addition, it allows for a larger positive delay than negative delay, where a negative delay is defined as a flight arriving ahead of schedule and a positive delay is defined as a flight arriving behind schedule.

The literature above assumes that the delay of aircraft is fixed and independent. This means that there is no relation between two delayed aircraft. Dijk et al. (2018), states that delays are not independent and that, in order to generate a representative time table there should be a relation modeled between aircraft. Often aircraft delay is caused by congestion at the departure airport. As a consequence all the flights originating from that airport will have a higher probability to arrive late. Dijk et al. (2018) proposes a method in which the probability for a flight to be delayed changes based on the delay of flights that have arrived earlier that day from the same origin airport.

2.6. Sliding time window

Sliding time window or rolling time horizon is an algorithm used in linear programming to speed up the process of finding a good solution by splitting the problem into smaller sub problems. Instead of having the model solve for the complete time horizon, the horizon is split up into smaller, overlapping parts which can be solved independently. The time horizons are overlapping to mitigate the compatibility problems that arise at the end of each time window. At the time of writing, there has not been any research published that is applying this concept to the AGAP. Research performed in other fields such as vehicle scheduling, has successfully applied the concept of sliding time windows to large scale problems (Cordeau et al., 2015). Compared to the AGAP, vehicle scheduling problems involve uncertainties in arrival and departure times of vehicles. In case of perturbations in the vehicle schedules, a new solution to the vehicle assignment is required in a limited amount of time.

Proposed models to schedule vehicles have long been using sliding time windows to reduce the size and complexity of the problem, while simultaneously increasing the solution speed. Early research focused on the scheduling of school buses. Ferland and Fortin (1989) proposed two different algorithms to solve the assignment of school buses. The first used a fixed time window with fixed starting times for the buses and the second used a sliding time window with flexible starting times for the buses. Both approaches led to improved solutions when compared to the traditional sequential assignment. The sliding time window proved to be more effective on average than the fixed time window.

A problem very similar to the AGAP is the scheduling of yard cranes in shipping yards. In a shipping yard, a set of cranes is used to on and off load containers from a ship. Van Dijk (2015) proposed different approaches to solve this problem of assigning yard cranes. The first approach, assigned specific parts of the yard to individual cranes. The working areas of the cranes are non-overlapping. The approach proved to be more effective than the second approach, in which the working zones of the cranes were overlapping.

The work described above has a close relation to the AGAP. At Amsterdam Airport Schiphol, currently the gate assignments are made the day before operation. Often these schedules are based on the seasonal airline schedules (Prent, 2019). Then during the operation this daily schedule is altered to mitigate any occurred gate conflict due to flight schedule perturbations. As discussed in section 2.2, a lot of research has been focusing on finding more robust solutions to the AGAP. Unfortunately, these robust solutions are weak in successfully absorbing uncommon or extreme delays as they often rely on historical flight delay information for their stochastic delay models or employ fixed buffer sizes between subsequently assigned flights to the same gate.

The proposed framework in this thesis, will implement a sliding time window to dynamically adapt to perturbations in the flight schedule. The sliding time window will be implemented for the reassignment phase of the framework. Modern jet airliners are equipped with a vast number of sensors and telecommunication equipment, one of these telecommunication devices is the ADS-B[7] system, it is used to communicate the current position of the aircraft in real time. The proposed framework tracks all the scheduled aircraft via the ADS-B system to precisely track when a scheduled aircraft takes off from its origin. Based on its flight path an accurate estimate with a deviation of 5 to 10 minutes can be made about the arrival time. These accurate and real time updating arrival times will be used by the sliding time window in the operational phase of the framework to solve for gate conflicts ahead of time. It is estimated that this approach will lead to a robust assignment for the AGAP as gate conflicts will be resolved up to hours before the conflicting flights arrive at the airport.

[7] Automatic Dependent Surveillance-Broadcast, is a cooperative surveillance system for aircraft and air traffic where the aircraft periodically sends out its current position, heading and velocity. A network of ground stations captures and aggregate these messages. The system can replace the secondary surveillance radar

3

Research definition

This chapter will describe the research definition for the model proposed to solve the AGAP. In the first section, the problem definition is formulated which follows from the literature review. The subsequent section, discusses the research framework, formulates the set of main research questions and derives the sub-research questions. The chapter is concluded with a motivation for the relevance of this study.

3.1. Problem definition

Following from chapter 2, a considerable amount of research has been published regarding the AGAP. The majority of the early proposed models in research have been focusing on solving the AGAP in a hypothetical environment, ignoring stochastic delays present in real world operation. The solutions found by these models are mostly non-robust as they aim to increase operational efficiency and/or passenger comfort, but they are not optimizing the flight to gate assignment to absorb flight that experience a flight schedule perturbation. However, a shift in research focus has been observed in the literature and more recent models are moving away from these traditional objectives and are focusing on providing robust solutions for the AGAP, either by changing the objective or formulating multi-objective models. Different approaches are proposed such as introducing a buffer between consecutive flights or using historical arrival trends to predict aircraft delays. In practice, these approaches are not as robust as theory suggests, since flight delays are difficult to predict because they are very dependent on random events such as weather, equipment failures and scheduling conflicts. The problem is that flight delays are unavoidable and very difficult to accurately predict in advance. The increasing demand in air travel and the restricted space most modern airports have, require a more efficient method to assign flights to gates dynamically in a real world environment.

3.2. Research framework

In this thesis, a new approach is proposed for the AGAP. In addition to an initial robust gate assignment for the start of the operation, it is proposed to implement a dynamically shifting sliding time window in the operational phase. During the operational phase, in which the gate reassignment is handled, the proposed model will use a sliding time window and real-time flight arrival times, computed from estimated flight times. These computed estimated arrival times are used in combination with the sliding time window to dynamically solve the AGAP in real time, swiftly reacting after the flights has departed from the origin and ahead of flight arrival to resolve gate conflicts that occur due to perturbations in the flight schedules. The proposed approach should be feasible for real world applications, allowing for fewer gate changes closer to the flights arrival time.

In order to achieve the above stated research objective, the following main research questions and sub questions were formulated:

1. *What is the advantage of using a sliding time window to dynamically update the initial optimal gate assignment solution to create a robust solution?*

   (a) What is the effect of statistically unexpected schedule changes in current robust models?

   (b) What is the effect of varying the sliding time window size?

2. *How does the sliding time window implementation perform in comparison to current robust models considering gate blockage as a metric?*

   (a) Does the sliding time window approach outperform manual re-assignment methods?

   (b) Does the sliding time window approach see an increase in performance if the initial optimal flight to gate assignment is robust?

3. *Which solution method, heuristic or optimal, provides the best run time performance for solving the re-assignment model?*

   (a) Which heuristic method is preferred when considering speed and finding a feasible solution?

3.3. relevance of the study

To be able to answer this set of questions, a new and innovative approach to the AGAP is used which is able to dynamically reassign gates based on the current and real time updated delay of the aircraft. It is expected that this innovative approach will yield to less manual interference in gate re-assignment process in comparison to current practices. In addition will it allow airports to increase their operational efficiency by using the available gates more effectively. At the same time will it increase passenger comfort by reducing the number of late notice gate changes. It is of importance for airports to increase their operational efficiency and thus their capacity without physically expanding the airport as the demand for air travel will continue to increase in the future. Most airports are physically restricted, which means that they are unable to expand their current operation without moving to an entire new location. Using a more effective gate reassignment model will increase the operational efficiency while at the same time increasing the passenger satisfaction levels by reducing passenger confusion, as there will be fewer last minute gate changes.

4

Methodology

The underlying chapter will discuss the process of creating a robust aircraft to gate assignment and reassignment framework. The framework of the model consists of three distinct phases. In phase one, fictional flight schedules and delays are generated. In phase two, the initial assignment of flights to gates is performed according to a set of objective metrics. Phase three, describes the gate re-assignment process in which flights are reassigned to different gates in case of gate conflicts due to flight delays. The framework will be tested using two different cases. The theoretical case is used to determine the optimal objective metric for phase 2. A theoratical case using artificial airport and flight schedules and a practical case using data real world data obtained from Amsterdam Airport Schiphol The second case is a practical case in which the chosen objective metric from the first case will be implemented and the model will be tested using data obtained from AAS. For this case the framework will be used without the first phase since it uses the actual flight schedules from AAS. For the creation of the framework, two different data sources have been used in the first phase. This chapter starts with a brief summary of the software used and will then describe in the following section the used data sources and the content of the data. The subsequent sections will describe the different phases of the framework.

4.1. Software

Software is an essential part of this research. In this section, the chosen software and associated packages (if applicable) are discussed. The main software component for the research is the linear solver. Following from section 2.4, CPLEX and Gurobi are among the most popular solvers and are also the fastest available on the market. Both companies behind the software offer their full fledged product for free to the academic community via academic licensing. These software packages have their own tools to create and run models. It is however, more convenient to use a general purpose programming language to interface with the solver directly. The added benefit of this is that it does not require learning a new programming language. Familiarity and positive, previous experiences of the author led to opting for the Python programming language. Both previously mentioned solvers provide an API[1] for Python. In this thesis, the Gurobi solver was used since it was easiest to obtain a license for and the website provides clear documentation and examples for Python interaction with the solver. The latest version of Gurobi was used at the time of research which was version 8.1.0. This version of Gurobi dictated the Python version used. The latest compatible version of Python was 3.6, so for the research version 3.6.8 was obtained from the Anaconda Python distribution. Anaconda is a free and open-source Python distribution which aims to simplify package management and dependencies for the scientific computing purpose. The benefit of this is that Gurobi provides its Python API module via an Anaconda package which made installation very easy.

Aside from the Gurobi module, the following additional Python modules have been used throughout this research:

– Geopy v1.20.0, used to compute the distance between a set of Longitude and Latitude coordinates;

[1] Aplication Programming Interface

- Matplotlib v3.0.2 for visualizing the results from the framework and model;
- Pandas v0.24.2, mainly for importing data sets and handling simulation results;
- Scikit-learn v0.21.3, for creating and fitting distributions for phase 1 of the framework;
- Seaborn v0.9.0, for simplifying the plotting process of Matplotlib;

4.2. Data Sources

Data used in this research was obtained from a non-public data set owned by the Schiphol Airport Group. This data set was provided for this research by an expert (service owner gate allocation at AAS) on August 4, 2019. The data set spanned from January 1, 2019 until August 31, 2019 and contained all arriving and departing flights to and from AAS. This data provided all the relevant information for each flight such as turnaround times, aircraft type, origin, destination, currently assigned and previously assigned gates. The data set was split into three tables. The first table contained 149,807 unique arriving aircraft. The second table contained 160,288 unique departing aircraft. Table three contained all arriving and departing aircraft as found in table one and two, but in addition it contained all mutations that have been made to the records from the other tables resulting in 422,643 non-unique arriving and departing aircraft.
Publicly available airport data was obtained from the OpenFlights website[2] on August 4, 2019. This data set contained detailed location information (latitude, longitude, region, country and city) for 7,698 unique airports. Merging this data set with the set obtained from AAS made it possible to determine flight time estimates for each flight arriving and departing from AAS.

Table 4.1: Columns present in data sets obtained from AAS after removal of non-relevant columns

Data column	**Description Arrival**	**Description Departure**
ID	unique id	
*ARRDEP	A for arrival	D for departure
*SDATE	date	
*SDATETIME	scheduled on blocks time	scheduled off blocks time
*FLTNR	flight number of arriving aircraft	flight number of departing aircraft
CONFLTNR	flight number for departure	flight number for arrival
*ACREG	aircraft registration	
*ACTYP	aircraft type according to IATA	
*HANDLER PAX	handler used for deplaning	handler used for enplaning
AIRLINE	airline	
RUNWAY	runway used for landing	runway used for take-off
*RAMP	aircraft parking spot	
*GATE	gate used by passengers	
ELDT	estimated landing time	N/A
EIBT	estimated in blocks time (time at gate)	N/A
ALDT	actual landing time	N/A
*AIBT	actual in blocks time	N/A
*AOBT	N/A	actual off blocks time (departure at gate)
EOBT	N/A	estimated off blocks time
*ADEP	airport of departure	N/A
ADES	N/A	airport of destination
MODIFICATIONS	modifications since initial record	
FLTDATA_MODIFIED	date record was added	

The obtained data from AAS contained three tables, each row in the table contained the information listed and explained in Table 4.1. This table shows that the last column (FLTDATA_MODIFIED) of each entry stores a timestamp indicating when that row has been modified. In practice however, rows are not modified but new rows are added for the flights where there has been a change. The second-last column (MODIFICATIONS), stores all the names of columns that have changed as a string, since the first occurrence in time of the combination "SDATE" and "FLTNR". An example of this can be seen in Table 4.2.

[2] https://openflights.org/data.html

Before the received data could be used for the analysis, it first had to be processed into a manageable form. The result of the processing will be two tables of data or data sets. The data set (Commercial Aircraft Visits, CAV-data set) will contain all the commercial flights that have visited AAS and the second data set will contain all the gate changes that have been imposed on those commercial flights (Commercial Aircraft Gate Changes, CAGC-data set).

The CAV data set has been created from the first two tables obtained from AAS and the airport data taken from OpenFlights. The data from AAS was first filtered to remove all the rows that missed data for import columns. All the columns that are required hold data are marked with an * (asterisk) in Table 4.1. After filtering, both tables could be joined following two distinct approaches. The columns that are used to join the two tables, determine the usage of the CAV data set for this research. The goal of joining the two tables is to determine the amount of time each individual aircraft has been on the ground at the airport during a visit. And to calculate the deviations from its scheduled arrival and departure time.

The two tables can be combined by matching the "FLTNR" column from the arrival table with the "CONFLTNR" column from the departure table and vice versa. However, this results in an unusable data set as the outbound flight number can change after the flight has already landed. The results is that the "CONFLTNR" in the arrival table is not updated and thus not reflect the actual situation. It was therefore chosen to merge both tables by matching the "ACREG" and "SDATETIME" columns from both tables. For each record in the first table, the first matching "ACREG" in time after "SDATETIME" of the first table was selected, but only if they were no more than 23.5 hours apart. This was done to remove aircraft from the data set that went out of operation for maintenance. Flight numbers are the same each day and to remove the possibility of selecting the flight from the following day, a time horizon of less than 24 hours was chosen. The resulting CAV data set has all the relevant data to determine the scheduled and actual time an aircraft has been on the ground at the airport. In the next step, columns were added to the data set which hold the computed deviations in minutes from the scheduled arrival and departure time, "ARR_DELAY" and "DEP_DELAY" respectively, see Equation 4.1 and Equation 4.2. An early arriving or departing aircraft has a negative sign and a late arriving or departing aircraft has positive sign.

$$\text{arrival\_delay} = \text{AIBT} - \text{SDATETIME} \tag{4.1}$$

$$\text{departure\_delay} = \text{AOBT} - \text{SDATETIME} \tag{4.2}$$

The CAGC data set has been created by joining the CAV data set with the third table from the data obtained from AAS. Each row in the CAV data set represents a visiting aircraft with an arrival time and a departure time. In Table 4.2, an example snippet is shown. The "FLTDATA" column is sorted so that the last addition to the table is at the top. The "CDMFLTSTATE" column reflects the state of the flight at the time the record was added to the table, "SCH" = scheduled, "AIR" = airborne and "TAX" = taxiing. So for the snippet in Table 4.2, the flight was scheduled for ramp A41, then while it was enroute to AAS, the ramp was changed to A54 and after landing at AAS, it received a final ramp change while taxiing to A44.
For both the arriving and departing part of each row in the CAV data set, the "SDATE" and "FLTNR" were joined on the same columns from the third table obtained from AAS. The rows from the third table that matched with the row from the CAV data set are saved to a new table in order to create the CAGC data set. For example, a flight from the CAV data set has 3 gate changes for the arrival part and zero gate changes for the departure part, matching this flight with the third table results in three and one rows for respectively the arrival and departure part. For this specific combination of arrival "SDATE"-"FLTNR" and departure "SDATE"-"FLTNR" there will be 4 rows in the CAGC data set.

Table 4.2: Example snippet taken from table 3 of the AAS data set to illustrate how changing data is handled and stored

ID	ARR DEP	SDATE	FLTNR	ACREG	CDM FLT STATE	RUNWAY	RAMP	ADEP	MODIFICATIONS MODIFIED	FLTDATA
83441879	A	05-01-19	AF1030	FGRZK	TAX	6	A44	LFRN	RAMP;	11:34
83440585	A	05-01-19	AF1030		AIR	36R	A54	LFRN	RAMP;	11:00
83422784	A	05-01-19	AF1030		SCH		A41		CDMFLTSTATE; RAMP;GATE	20:13

4.3. Data Analysis

The main purpose of the data analysis was to understand how delays developed during daily airport operations. These findings are used to calibrate a model which is used to generate representative timetables and delays used for the first phase of the framework, scenario generation as discussed in subsection 4.4.1. The analysis of the data has been performed based on the following questions:

1. Are perturbations in the arrival time dependent on the origin of the aircraft, i.e. are flights arriving from some origins more prone to delays than others?
2. Are certain flights more prone to delays than other flights and do flights have a systematic delay?
3. Are delayed flights connected, i.e. does the delay of other aircraft cause a snowball effect at the airport where other flights will be delayed as a consequence of a delayed flight?

These questions will determine whether there are trends that may influence the delay model, and thus should be taken into account while defining the flight delay model. The goal of this work is not to create a delay model as realistic as possible but rather to create a model that is capable of creating representative timetables that accurately represent global trends visible in the data obtained from AAS.

The first question determines if flights arriving from a certain airport, country or region are more prone to delays than other flights. This question is approached by using a method similar to the one used by Buitendijk (2014). In order to determine the relation stated in the question, the delay data was clustered based on the arrival delay and origin of a flight. Since each flight number occurred multiple times in the data set, the predominant cluster for each flight number could be determined. Then for each of the clusters a delay distribution can be fitted, for each flight number that has this cluster as their predominant cluster this distribution can be used to generate random delays. Following this approach, ensures that the delay model produces delays which mimic the real world. In addition would the delay model exhibit the effect that flights coming from a certain origin are more prone to delays than others.
The clustering has been performed on the CAV data set, using the K-means algorithm seen in Algorithm 1. From the CAV data set, the "ADEP_lat", "ADEP_lon" and "arr_delay" columns were filtered out. Each of these columns were normalized to have a zero mean and a variance of 1^2, doing so ensured that each of the columns had an equal weight for the distance metric (Euclidean distance) used by the algorithm. Determining the optimal number of clusters was done by using the elbow method proposed by Peña (2018). The Elbow plot for clustering the CAV data set is seen in Figure 4.2. The plot is roughly representing a humans arm and at the location of the "elbow" indicates the optimal number of clusters, six in this case. Visualizing the cluster gives an overview of how the algorithm has performed. For this the latitude and longitude columns were reverted back to decimal degrees, the result is visualized in Figure 4.1. This picture shows that all the flights are grouped by geographical region. Table 4.3, lists the top 3 countries in each cluster. Studying the countries found inside the individual clusters reveals that the "ADEP_lat" and "ADEP_lon" variables are despite the normalization dominant over the "arr_delay" variable. The wide range of departure airports for flights arriving at AAS prevents the clustering algorithm from grouping flights based on historical delay information with certainty. Fitting a delay distribution over the flights in each of the found clusters leads to very similar distributions with very low fitting scores. It is expected that adding sub-clusters to each of the found clusters will improve the accuracy of the fitted distributions as it lowers the dominance of the "ADEP_lat" and "ADEP_lon" variables since their variance is smaller. Sub-clustering can also be done by adding extra variables to the algorithm such as airline and time of departure. However, the large number of resulting clusters makes this an infeasible approach. The goal of the question is to determine whether certain origins are more prone to delays than others. The above analysis concludes that for this research the potential effect of the origin on flight delay is not significant enough to be relevant for the delay model used in this research.

Table 4.3: Found clusters and their dominant countries, total observations: 138061

cluster	observations	top 3 dominant countries
0	40316	Spain (10022), Italy (9118), France (4324)
1	6976	China (1896), UAE (1306), India (591)
2	9426	USA (7593), Canada (1265), Mexico (454)
3	76520	UK (26580), Germany (12863), France (5419)
4	2018	Kenya (714), South Africa (396), Uganda (199)
5	2805	Netherlands Antilles (728), Brazil (506), Puerto Rico (252)

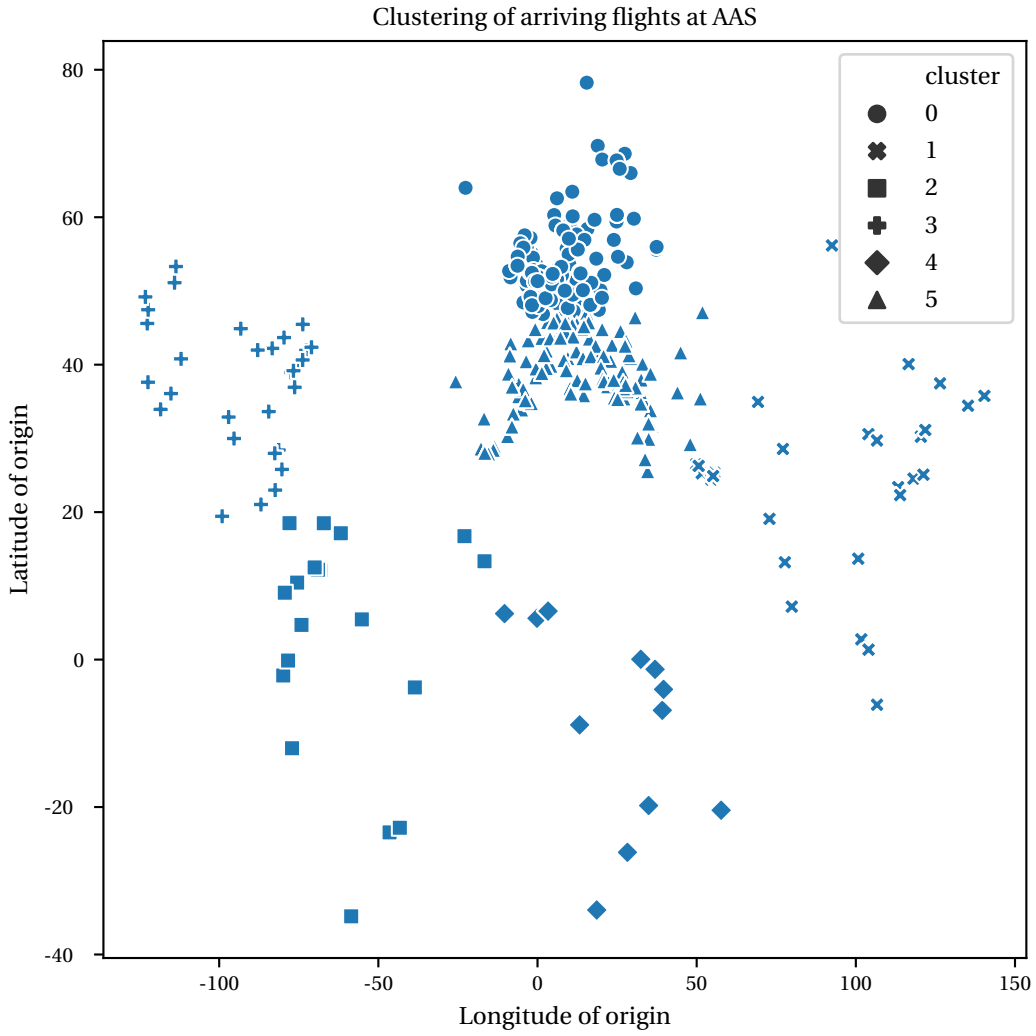

Figure 4.1: Visualization of the clusters found for the arriving flights in the CAV dataset

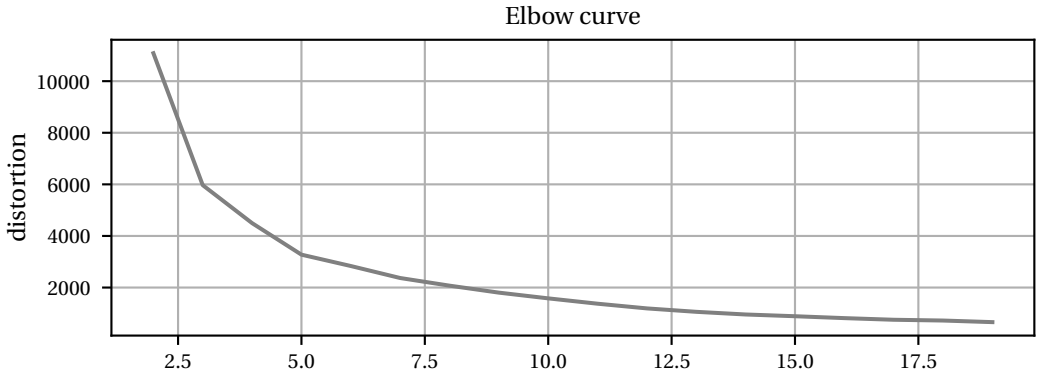

Figure 4.2: Elbow plot used to determine the number of clusters for the K-Means algorithm

Algorithm 1 K-Means algorithm

input: F, k **where** $F = setoflights, k = \#clusters$
output: k number of clusters
require: $F \neq \{\}$ and $k > 0$
procedure GENERATION OF CLUSTERS
  initialize k random centroids
  repeat
    for all $instancesi$ in F **do**
      $shortestdistance \leftarrow 0$
      $membership \leftarrow None$
      for all $centroid$ **do**
        $dist \leftarrow$ Distance($centroid$)
        if $dist < shortestdistance$ **then**
          $membership \leftarrow cluster$
          $shortestdistance \leftarrow dist$
        end if
      end for
    end for
    RecalculateCentroids($centroids$)
  until convergence
end procedure

The second question, explores the possibility that certain flight numbers are more prone to being delayed than others. This question is raised by the increased congestion at major airports and the tight schedules used by airlines. It explores the possibility of trends arising from this i.e. the morning flight from airport XYZ of airline A is always on-time while the afternoon flight of airline A is always late (Dijk et al., 2018). In order to provide an answer for this question, the CAV data set was used. For each inbound flight number the median, lower quartile and upper quartile were computed for the arrival delay. These results are visualized for a few flights using a boxplot in Figure 4.3. From this plot it can be seen that for some flights the lower and upper quartile are further spaced than for other flights. This implies that some flights achieve a better on-time performance and experience less variance in the arrival time. The median can be positive or negative. A negative median means that the flight, in general, arrives before its scheduled arrival time. A positive median implies that the flight, in general, arrives after its scheduled arrival time. The boxplot shows that the variance in arrival time for flight "KL0588" is much smaller than for flight "KL0898". "KL0898" and "KL0876" are both originating from Asia, however the on-time performance for the first is significantly better and with a smaller variance. The boxplot provides closure for the second question: some flights are more prone to delays than other flights. The variance can be caused by variety of circumstances outside the scope of this research, but include factors such as, previously accumulated delay for the aircraft from previous flights and congestion at the airport of departure, resulting in a delayed departure.

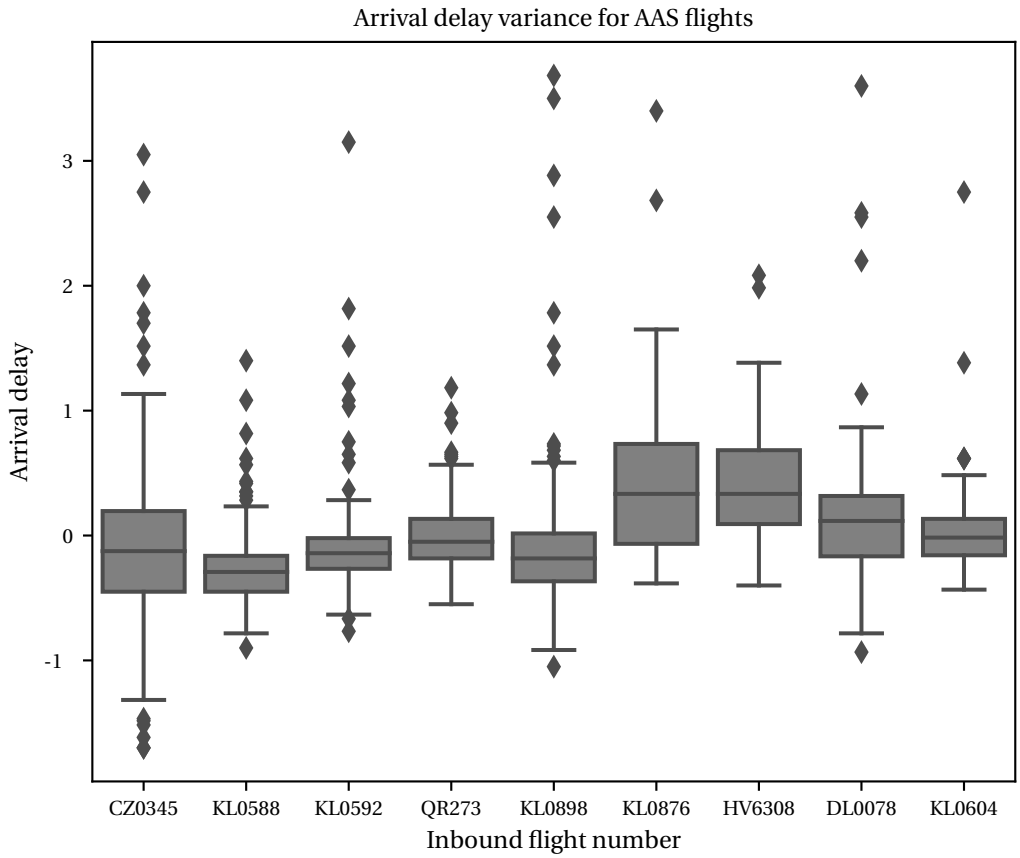

Figure 4.3: Visualization of the clusters found for the arriving flights in the CAV data set
Origin of flights from left, PEK, LOS, JNB, DOH, PEK, BKK, BEY, LAX, LAX

The third question, concerns the relation between individual aircraft delay and the effect of this on the rest of the airport. Heavy weather or union strikes at the airport or in neighboring countries are known to effect other airports in the vicinity directly or indirectly. The question explores if this effect is visible at a smaller scale where a single aircraft delay causes a snowballing effect at the airport. This is a complex question since aircraft delay is dependent on many different factors. One of these factors is the size of the aircraft, large wide-body aircraft require more time for the turnaround than smaller narrow-body aircraft. Narrow-body aircraft can be turned around in less than 30 minutes, airlines use these small turnaround times to their advantage and schedule their narrow-body aircraft for multiple flights a day. As a result, narrow-body aircraft can accumulate delay during their first few flights of the day which will have a snowballing effect on later flights in the schedule of that aircraft. This effect is visualized by plotting the arrival delay for all the flights that operate a given route. In Figure 4.4, this has been done for all airlines that fly from LHR to AMS on three different days. Appendix B, shows the plots for five most flown routes to AMS.

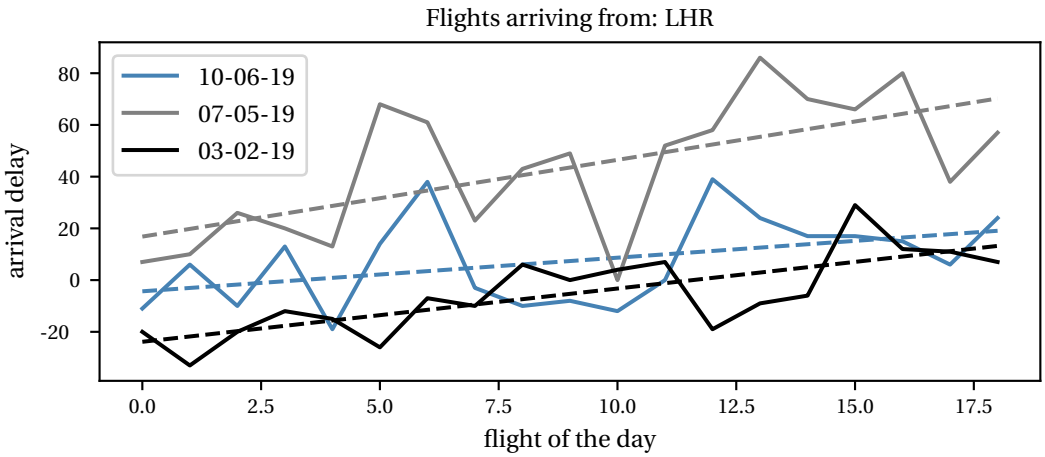

Figure 4.4: Delay propagation of flights arriving from LHR

The figure shows that, for all three days, the LHR to AMS route accumulates delay as the day progresses. Such propagating delays, where aircraft are stacking delays as the day progresses are not beneficial for the airlines. To counteract this effect, airlines can employ different techniques. An aircraft that is due to arrive but is behind schedule will, with the tight turnarounds see a delayed departure for its subsequent flight. Airlines are using different techniques to mitigate this effect, they can for example insert a "Firebreak" mid-day in the aircraft's flight schedule. A "Firebreak" is a longer than strictly necessary turnaround in the flight schedule of an aircraft. This creates a buffer in the schedule which can absorb accumulated delay from earlier that day. Another option which airlines have available at their home base, is to swap airplanes between routes or use an unassigned aircraft to start the flight without previously accumulated delay. Both these options are visualized in Figure 4.5. This figure shows that for a small delay accumulation the "Firebreak" suffices. For longer delays, a swap of aircraft is required to ensure future flights arrive and depart according to schedule. The figures shown in Appendix B, combined with the fact that narrow-body aircraft are flying on tight schedules lead to the conclusion that narrow-body aircraft are prone to accumulating delay, resulting in a snowball effect in their flight schedule. For larger wide-body aircraft this effect is not observed as they do not operate with scheduled tight turnarounds. They also operate far less daily flights and are constantly in different time zones. Narrow-body aircraft fly multiple flights a day in and out of their home base. Most airport are closed in the middle of the night for a short period of time. At this time no flights are allowed to depart. As a consequence of this, there is a brief period in which narrow-body flights have landed but can not take off again because the airport is closed. In other words, there are always wide-body aircraft airborne while the narrow-body fleet is mostly grounded during the night.

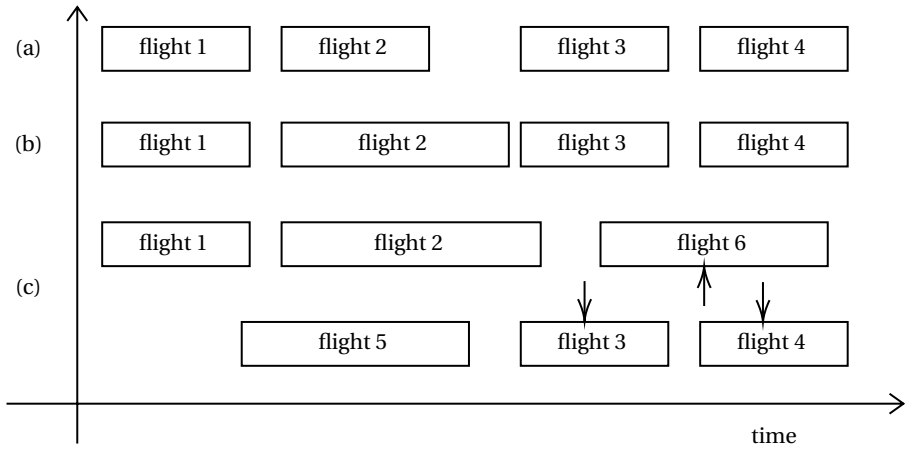

Figure 4.5: (a) reference schedule for AC1, (b) "Firebreak" used by AC1 to absorb the delay from flight 2, (c) swapping of flights between aircraft to mitigate delay

4.3.1. Conclusion

The scope of the data analysis was to identify trends in the delay data. These trends can then be modeled into the framework to create realistic random timetables and associated delays. Results of the data analysis showed that aircraft delay is not related to the origin of the flight. Certain airline routes achieve better on-time performances with a smaller variance in arrival delay than other routes. The cause of this effect is hard to determine as it depends on many different factors such as airline operation, aircraft schedule, aircraft type and congestion at the airport. The results of the analysis also showed that for narrow-body aircraft there is a delay trend visible as the day progresses. During the day, narrow-body aircraft accumulate delays from earlier flights that day due to tight scheduling of the aircraft by the airline. Towards the end of the day, the probability that a narrow-body flight will arrive later than its scheduled time is higher than at the start of the day. For wide-body aircraft this trend is not so pronounced as they often have longer turnaround times and execute far less flights a day (2 vs up to 5 for a narrow-body aircraft).
The analysis of the three questions showed that there are two trends that need to be implemented in the delay model to accurately represent the arrival variance observed at AAS. Certain flight numbers are more prone to being delayed than others and narrow-body aircraft accumulate delay as the day progresses.

4.4. Aircraft to gate assignment framework

The flight to gate assignment framework consists of three different phases, the first phase is only used for the theoretical case and the latter two phases are used for both the theoretical and practical case in which the framework will be validated. An overview of the framework with the distinct phases is shown in Figure 4.6. The advantage of using representative fictional data for the theoretical case of the framework is that it allows for greater control over the input to the framework and thus provides the opportunity to feed the framework with hard to solve flight schedules and delays. The framework is modeled after the pier-finger layout found at AAS. Data from the operation of this airport is used in the validation process. The framework is implemented according to object-oriented programming where each flight schedule is represented as a class, all the methods that interact with the flight schedule are implemented inside the class.

This section will describe the three phases of the framework, in subsection 4.4.1, the first phase of the framework concerning the scenario generation will be discussed. In subsection 4.4.2, the two different objectives for the tactical planning phase will be discussed. The section is concluded with subsection 4.4.3, which describes the third phase of the framework, the operational phase determining the gate reassignments. In this section, the implementation of a sliding time window into the flight to gate assignment framework is discussed.

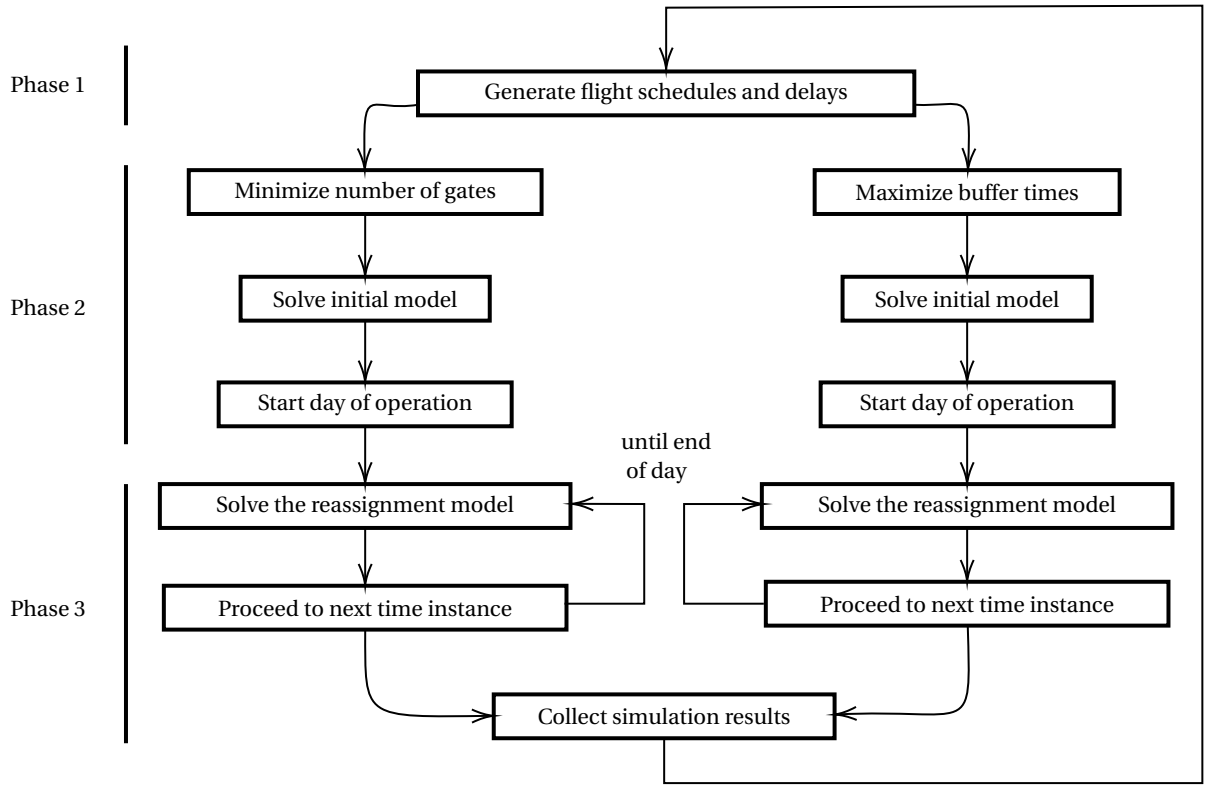

Figure 4.6: Simulation flow diagram for the framework with the three phases

4.4.1. Phase 1 - Scenario generation: Initial generation of scenarios

The first phase of the framework is only used for the theoretical case of this research. The main goal of this first phase is to generated realistic scenarios in which gate conflicts are guaranteed to occur as aircraft deviate from their scheduled arrival time. The scenario generation process is two-fold. First a flight schedule is generated, then delays are imposed on the flights in the flight schedule. The delays are based on the data analysis to accurately reflect real world delays found at AAS.

The framework has been written in Python, the advantage of using this widely adopted general purpose programming language is the large availability of packages that reduce the programming workload for the user. The framework is written in Object Oriented Programming (OOP) paradigm. This is a programming style based on the concepts of objects which represent data and all the methods (functions) that can manipulate the data of the object. Objects are instances of classes. The flight schedule used in the framework is represented by the "FlightSchedule"-class. An example of the data stored inside the class object can be seen in Table 4.4.

Table 4.4: Data field for each flight inside the "FlightSchedule"-class

flight_arr	scheduled arrival time of flight
flight_dep	scheduled departure time of flight
gate	gate the flight has been assigned to
pax	number of passenger on board
flight_time	time aircraft has to fly to reach the airport
ATD	Actual time of departure from origin
STD	Scheduled time of departure from origin
req_turnaround	minimum number of minutes flight needs to turnaround
aircraft_type	narrow or wide-body aircraft

Each time this class is initialized, a flight schedule is automatically generated. Before the class can be initialized, the following parameters are required to be set by the user:

- number of gates (integer)
- number of flights (integer)
- time of operation (start of day, end of day)
- minimum turnaround time (narrow-body, wide-body)
- distribution of wide and narrow-body aircraft (% wide-body, % narrow-body)
- seed to reproduce the same result (integer)

With these parameters set, the model generates a single flight schedule for the specified day, following the procedures described by the pseudo code in Algorithm 2. The algorithm fills all the gates sequentially with flights, it starts the day at the defined start of day, it then applies a random variance of plus or minus 30 minutes to prevent all the flights arriving at exactly the start of the day. Then it determines the time the flight will be at the gate using a random distribution, combined with the arrival time, this determine the departure time of the flight. The algorithm then determines the idle time of the gate which determines the arrival time of the next flight. This process continues until the algorithm has reached the end of the day. The algorithm then moves to the next gate and starts the whole process again until all the gates have flights. The algorithm is either constrained by the number of gates, or the number of flights, whichever is reached first. This means that if the algorithm receives 140 flights and two gates it will stop when it has generated enough flights to occupy two gates. The main motivation for constraining the flight schedule generation process this way, is that the generated flight schedules are guaranteed to be solvable if the target airport has at least a similar number of gates. The algorithm uses different configurable distributions to generate the required information for each flight:

- **Passengers on board:** Discrete uniform distribution, upper and lower bounds depend on the size of the aircraft and are varying so that the load factor of the flight ranges between 50% and 95%;
- **Turnaround time:** Discrete uniform distribution, upper and lower bounds depend on the aircraft type, wide-body flights have a higher lower and upper bound than narrow-body flights;
- **Time between flights:** Absolute random normal distribution which returns the gate idle time, during a configurable time it generates longer idle times to simulate a quiet period between incoming and outgoing waves of aircraft;
- **Flight time:** Discrete uniform distribution, upper and lower bounds depend on the aircraft type, wide-body flights have a higher lower and upper bound than narrow-body flights.

Example shapes for these distributions are shown in Figure 4.7.

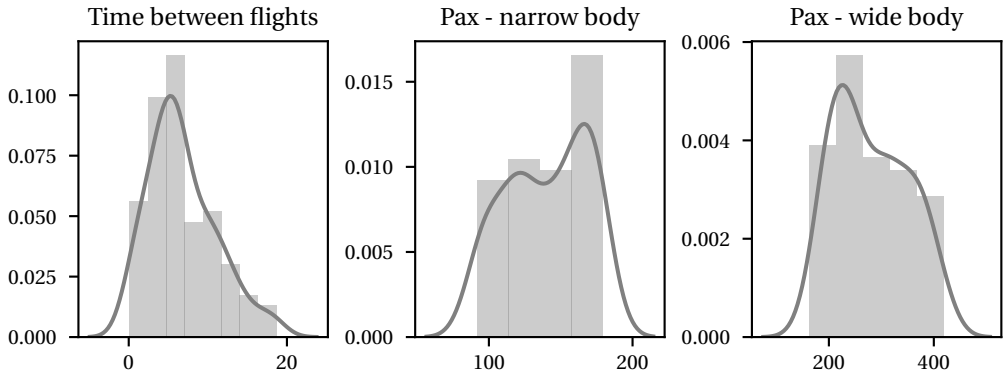

Figure 4.7: Distribution examples used for the scenario generation, from left, absolute normal distribution [scale=6, loc=6], discrete uniform [90, 180] and discrete uniform [160, 420]

The data analysis showed that there is no distinct relation between the origin of a flight and average delay. Instead, the delay depends on the individual schedules of each aircraft. The tighter these schedules, the higher the probability that delay will accumulate as the day progresses. This effect was predominantly visible

Algorithm 2 Initialization of flight schedule

```
1: procedure FLIGHTSCHEDULE
2:     current time = start of day + randomInt[-30, 30]                ▷ in minutes
3:     for i in gates do                                               ▷ loop through all the gates
4:         while not end of day and not last flight do
5:             flight arrival = current time
6:             flight departure = current time + rnd(time at gate)
7:             current time = flight departure + rnd(gate idle)
8:             schedule ← flight                                       ▷ save flight to schedule
9:         end while
10:    end for
11:    return flightschedule   ▷ return flight schedule for the day with all generated flights
12: end procedure
```

for narrow-body aircraft as they execute more flights per day when compared to wide-body aircraft. The arrival variance showed that flights arrive up to roughly 30 minutes ahead of the scheduled arrival time on a regular basis and in a best case scenario up to about 1 hour earlier. Flights are commonly delayed up to 30 minutes, with outliers easily ranking up a couple of hours in delay. The effect of accumulated delay is clearly visible in the AAS data and is therefore modeled into the framework. In order to create representative delays, individual aircraft delay was picked from a fitted random distribution.

The data analysis showed that the delay variance for narrow-body and wide-body aircraft differ. The CAV-data set was therefore split into a CAV-narrow and a CAV-wide data set for respectively, narrow-body and wide-body aircraft. Splitting was done based on the "ACTYPE" column that lists the ICAO[3] aircraft abbreviation which is unique for each type and model of aircraft on the market. The distitinction between wide-body and narrow-body aircraft was made based on the number of aisles present in the aircraft. Narrow-body aircraft are using a single aisle configuration for the cabin where wide-body aircraft are using more than one aisle inside the cabin.

In order to fit a large collection of different distributions to the data set, the Scikit-learn package for Python was used. This popular open-source machine learning library features the most commonly used machine learning algorithms ordered in different modules. For this part of of the project the "stats" module was used. This module contains over 80 popular distributions[4]. Choosing the right distribution is a meticulous process as the quality of the selected distribution determines how well the results drawn from the distribution will mimic the real world situation. A systematic approach was taken to select the best matching distribution for the AAS data set, this process is described by Algorithm 3.

In order to reduce the computational effort, a pre-selection of distributions for which a fit was computed was made. The preselection was done by plotting the delay distribution for a couple of random days from the data set. The general shape of the plotted Probability Density Function (PDF) was compared with the distributions found in Scikit-learn. An example of a distribution plot for such a random day is shown in Figure 4.8. A quick glance at the plot might lead to believe that the plotted distributions resembles that of a normal distribution. However, the PDF for a normal distribution is symmetrical and for the arrival delay it is asymmetrical. The right side of the PDF slopes more gradually compared to the left side. This can be explained with the results from the data analysis, the negative delay has narrower bandwidth than the positive delay. When comparing the shapes of the PDF's found for the arrival delay of random days from the data set, most of the distributions from the Scikit-learn package were discarded since the general shape of the PDF's did not match. The following distributions remained: Burr Type III (scipy.stats.burr), Exponentially modified normal (scipy.stats.exponnorm), F (scipy.stats.f), Fisk (scipy.stats.fisk), generalized extreme value (scipy.stats.genextreme), right-skewed Gumbel (scipy.stats.gumber_r), log-Laplace (scipy.stats.loglaplace) and a normal distribution (stats.scipy.norm).

With the potential distributions selected, the distribution fitting algorithm can be run for both the CAV-narrow and CAV-wide data set. The fitted distributions for both the CAV-narrow as well as the CAV-wide can

[3] International Civil Aviation Organization
[4] https://docs.scipy.org/doc/scipy/reference/stats.html

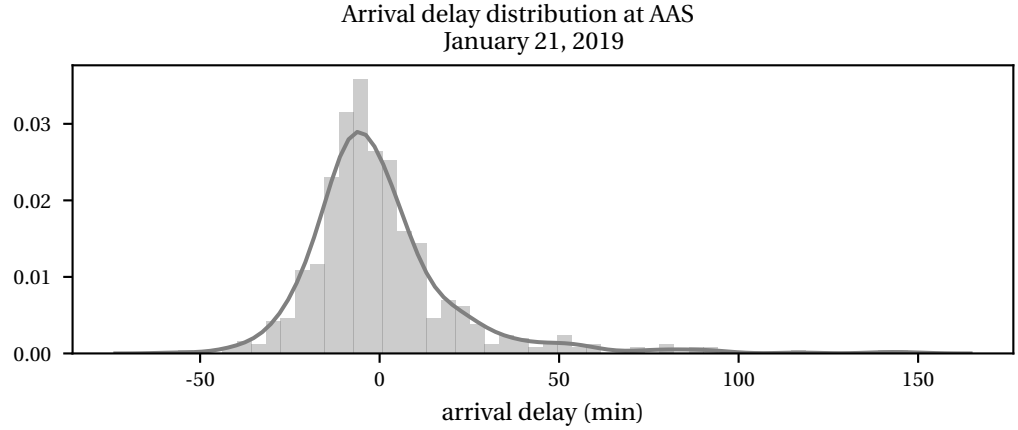

Figure 4.8: Delay distribution for AAS on January 21, 2019

be seen in Figure 4.9 and Figure 4.10 respectively. These plots both show that the Exponnorm (exponentially modified Gaussian) distribution achieved the best fit for both the narrow-body and the wide-body flights. This distribution is also known as a skewed normal distribution. Depending on the shape parameters, the distribution can vary in shape from an exponential to a normal distribution. The probability density function is given by the standard form:

$$f(x;\mu,\sigma,\lambda) = \frac{\lambda}{2} e^{\frac{\lambda}{2}(2\mu+\lambda\sigma^2-2x)} \operatorname{erfc}\left(\frac{\mu+\lambda\sigma^2-x}{\sqrt{2}\sigma}\right) \tag{4.3}$$

Where *erfc* is the complementary error function which is defined as:

$$\operatorname{erfc}(x) = 1 - \operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2}\, dt \tag{4.4}$$

The PDF from Equation 4.3 can be rewritten to the parameter form as found by the Scikit-learn implementation, here $loc = \mu$, $scale = \sigma$ and $K = 1/\sigma\lambda$. Substitution leads to the following equation:

$$f(x;loc,scale,K) = \frac{1}{2k * scale} e^{\frac{1}{2k*scale}(2*loc+\frac{scale^2}{k*scale}-2x)} \operatorname{erfc}\left(\frac{\mu+\frac{scale^2}{k*scale}-x}{\sqrt{2}*scale}\right) \tag{4.5}$$

For the narrow-body aircraft the following parameters were found for the distribution:

$$K = 2.43 \quad loc = -15.05 \quad scale = 10.51$$

And for the wide-body aircraft the following parameters were found for the distribution:

$$K = 1.83 \quad loc = -23.23 \quad scale = 12.40$$

The above parameters show that in comparison, wide-body aircraft see a larger variance in actual arrival time than the narrow-body aircraft. The wide-body aircraft are, potentially, arriving earlier as well as later than narrow-body aircraft. In the above equations, this is reflected by the shape parameter, a large shape parameter indicates a more stretched distribution. The loc-parameter indicates where the top of the PDF is located, the values drawn from the distribution will be centered around this value, for both aircraft types, this means that flights arrive slightly ahead of time in general. In addition, both the narrow- and wide-body aircraft tend to be arriving ahead of their scheduled arrival time, indicated by the negative loc parameter for both fitted distributions, which indicates where the top of PDF from the distribution is located.

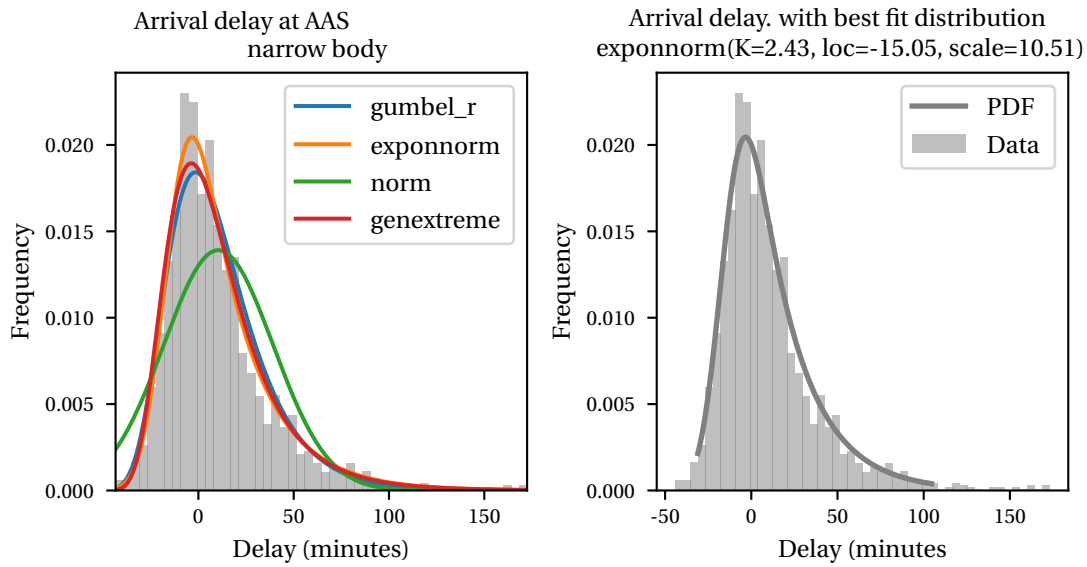

Figure 4.9: Fitting delay for AAS for the narrow-body fleet visiting at AAS

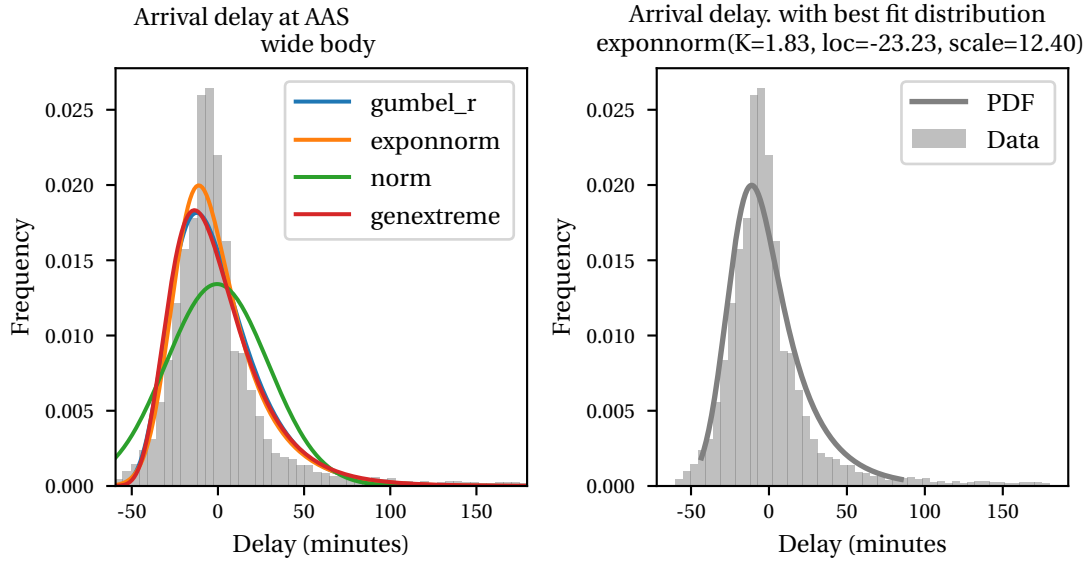

Figure 4.10: Fitting delay for AAS for the wide-body fleet visiting at AAS

The data analysis concludes that narrow-body aircraft are more prone to positive delay as the day progresses. This effect is implemented in the framework by shifting the found distribution for narrow-body aircraft slightly to the right during the day. The loc parameter for the narrow-body distribution equals -15.05 for the first narrow-body flight at the gate, then each time a narrow-body aircraft arrives at that gate, the distribution is shifted to the right by adding 0.25 to the loc parameter. The goal of this is to set an upward trend that represents the observed trend found at AAS. Figure 4.11, shows the delay trend for three days as obtained from the fitted distribution for the narrow-body aircraft. Figure 4.12, shows the delay trend for three days as obtained from the shifting distribution for the narrow-body aircraft. The indicated, dotted, trend line shows a similar behavior as observed from the plots shown in Appendix B.

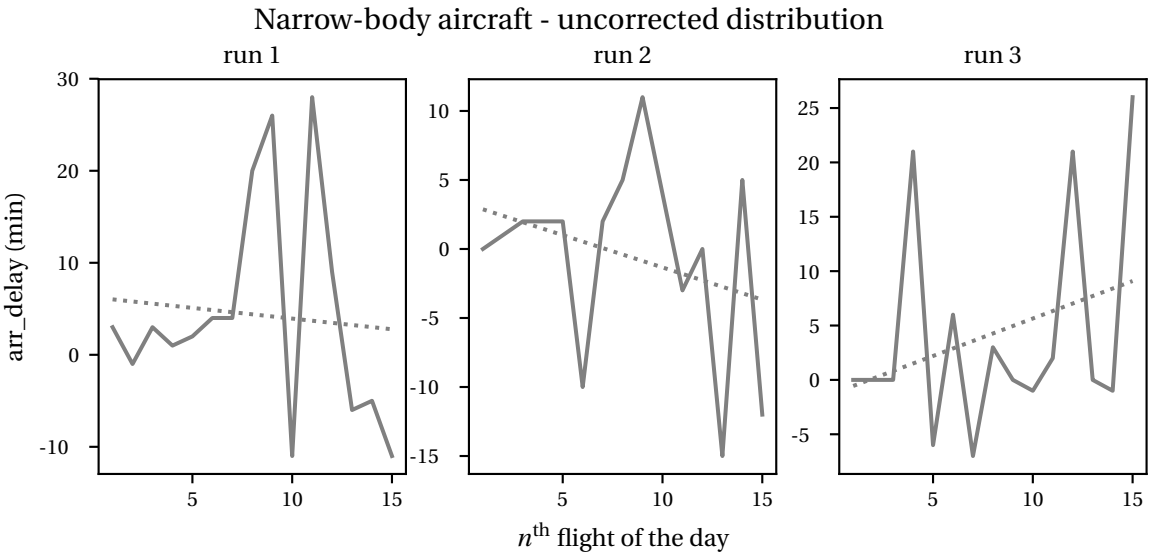

Figure 4.11: Three runs on the uncorrected distribution to indicate the absence of the correct delay trend as observed at AAS, dotted line represents the trend line.

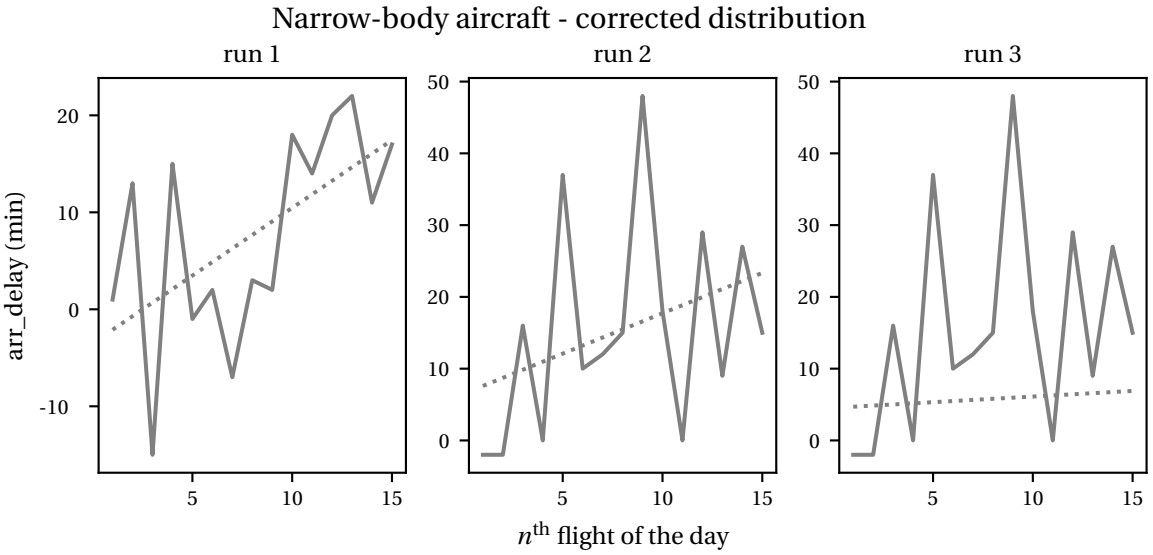

Figure 4.12: Three runs on the corrected distribution to represent the delay trend found for narrow-body aircraft at AAS, dotted line represents the trend line.

For the theoretical case of the framework, an fictional airport is used. The layout of this airport is based on the most commonly found layout, the pier and finger layout, shown in Figure 4.13. The figure shows that the fictional airport has two piers with either 9 or 10 gates. This layout has been inspired on the layout found at AAS. The gate restrictions are slightly relaxed as compared to a real airport such as AAS. For the fictional airport, all the gates can handle both wide and narrow-body aircraft. At real airports, such as AAS, often each gates have restrictions on the size of aircraft that can be handled at the gate. The relaxation is used to rule out gate conflicts that arise from gate and flight compatibility conflicts due to gate size constraints.

The generated flight schedules contain more flights than piers at AAS typically have given their number of gates. This is done to create a challenge for the model. The delays are representative for the operation observed at AAS and an example flight schedule can be seen in Figure 4.14. This concludes phase 1, the scenarios have now been generated and will be used in phase 2.

Algorithm 3 Process to find the correct delay distributions for the first phase of the framework

input: S, D **where** $S = CAV - data\,pier\,E\,and,\ F, D = set\,of\,distributions$
output: best distribution and parameters for each day
require: $S \neq \{\}$
procedure FIT DISTRIBUTIONS FOR EACH DAY IN S
  $results$ ▷ table for best fitting distribution and parameters per day
  for each day in S **do**
    $best\_dist \leftarrow None$
    for each $dist$ in D **do**
      $result = dist$.fit(arr_delay of day)
      fit = SSE($result$) ▷ Error of Sum Squares
      if $fit < best\_dist$ **then**
        $best\_dist \leftarrow dist$
      end if
    end for
    $results += best\_dis + best_d is\_parms$ ▷ store best distribution and parameters in table
  end for
  sum($results$) ▷ sum results to get most popular distribution
end procedure

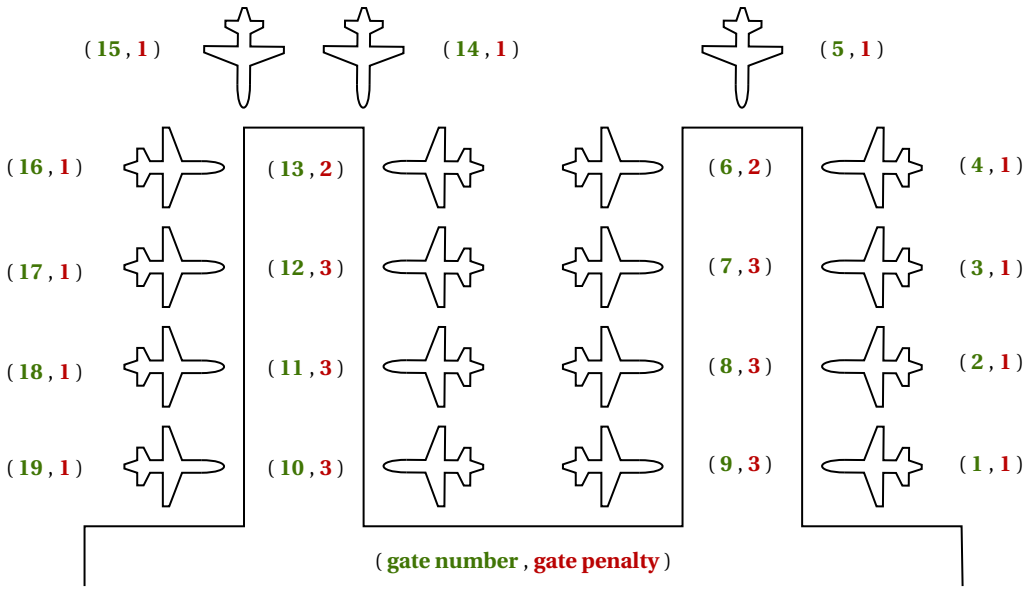

Figure 4.13: Layout of the fictional airport used in the first case of the framework

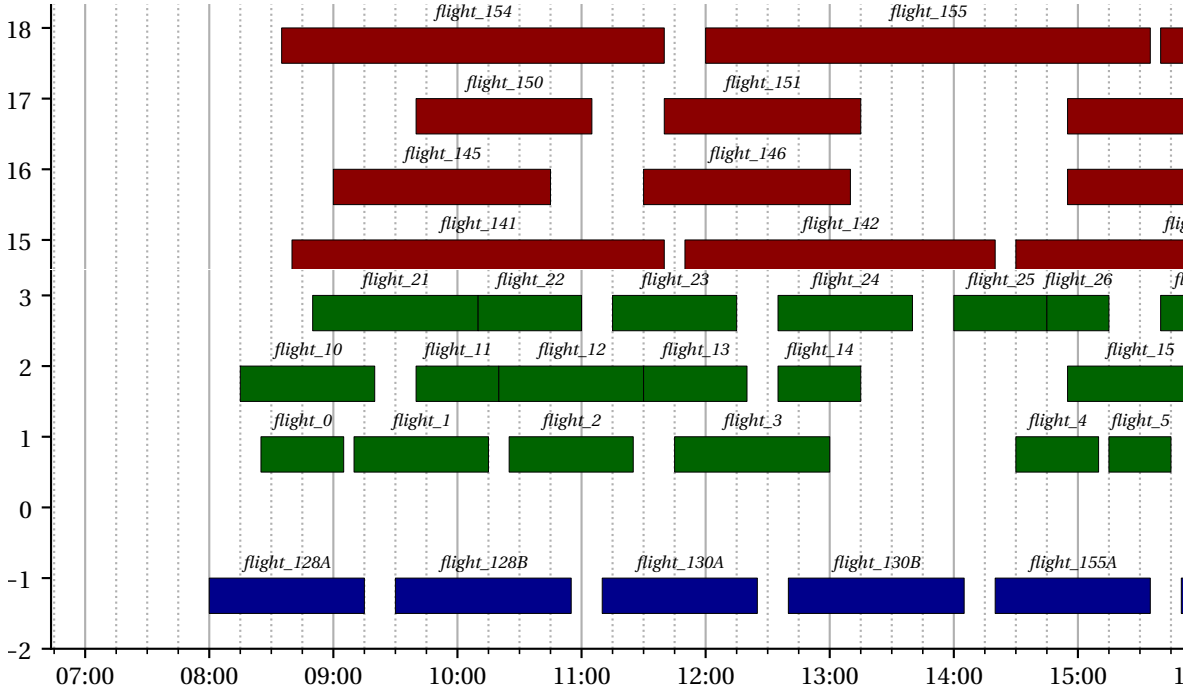

Figure 4.14: Example snippet from a generated flight schedule, blue indicates flights that are split due to a significantly long layovers, green and red are indicating respectively narrow- and wide-body flights

4.4.2. Phase 2 - Tactical planning phase: start of day assignment

The main goal of the tactical planning phase, is to find a solution for the AGAP that can be communicated by the airport authorities to all the relevant stakeholders. The stakeholders can use this initial gate assignment to plan their work schedules and the airlines can communicate to the passengers via which gate they have to board their flight. Ideally the initial gate assignment should be robust so that fewer gate reassignments are necessary closer to the flights arrival. A new objective is proposed which minimizes the number of gates used. In addition, an objective metric commonly used in literature has been chosen: maximizing the buffer times between consecutive flights. These objectives will define which of the objectives will be used for the practical implementation of the model i.e. the AAS data. The implementation of both these objectives will be discussed in this subsection. For the tactical planning model, the model has several options that can be used to resolve gate conflicts:

- If the turnaround time is sufficiently long, the flight can be split into a disembarking, parking and embarking part. For the parking part the aircraft can be towed away from the gate;
- Flights can be delayed on the apron to wait for a gate to become available;
- Flights can be assigned remotely for the cases where there is no gate available.

The above options have been listed in order of favor, for this a balance is made between the passenger comfort and the convenience for the airport. The first option, where the aircraft is towed to a remote parking location between disembarking and embarking is favorable for the passengers as they will not notice any difference compared to a flight that stayed at the gate between embarking and disembarking, as shown by Figure 4.15. For the airport authorities there is a slight cost involved for the towing procedure, it is however more convenient than shuttling the passengers to the remote parking location for disembarking or embarking. For the second option, the flight is hold on the tarmac for set maximum number of minutes to wait for the assigned gate to become available. This is favorable for the airport since there is no rescheduling required to resolve the gate conflict. For passengers this is small inconvenience as they have to sit in the airplane for a few extra minutes. However, they are most probably sooner inside the terminal building than when the aircraft was assigned to a remote parking location as per the third option. Assigning the flight remotely is a last resort for the airport authorities as it greatly reduces passenger comfort and involves a lot of extra arrangements on the ground since all the passengers need to be transferred between the terminal and aircraft.

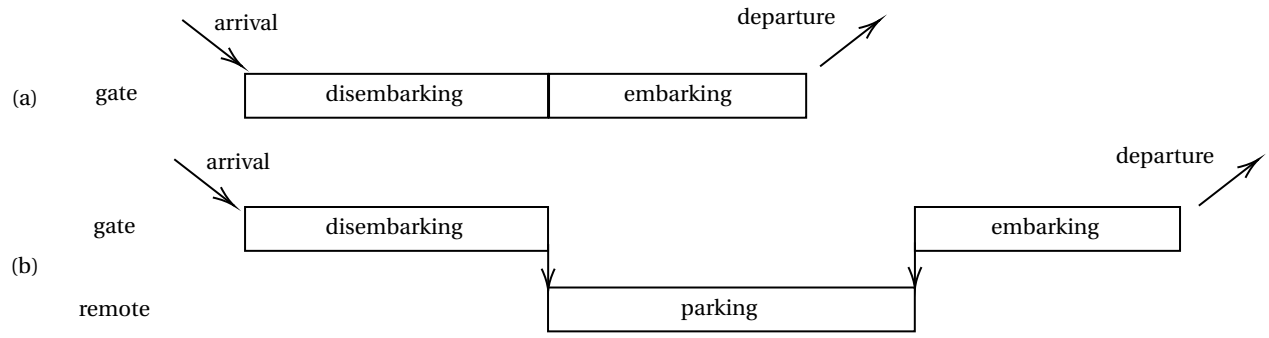

Figure 4.15: (a) flight that cannot be split due to a short turnaround time is handled at the same gate, (b) flight with a sufficiently long turnaround can be split, between disembarking and embarking, the flight is parked at a remote location. The disembarking and embarking can be handled at different gates

The sets that are used for the tactical planning phase are summarized below:

$$F = \text{set of flights} \tag{4.6}$$

$$C_i = \text{subset of F with flights that conflict with flight } i \tag{4.7}$$

$$F_c = \text{subset of } F\text{containing flights that can be split for embarking and disembarking} G = \text{set of gates} \tag{4.8}$$

$$G_{ci} = \text{subset of } G \text{ containing all compatible gates for flight } i \tag{4.9}$$

$$\tag{4.10}$$

In order to define the sets above, several parameters are required to be set by the user:

- **Maximum buffer time:** Maximum time between two consecutive flights assigned to the same gate, the model distributes the buffer time equally over all the flights in the model
- **Maximum delay time:** Maximum number of minutes a flight can be hold waiting on the apron before a assigned gate becomes available
- **Minimum turnaround time:** Minimum number of minutes a flight has to stand at a gate to perform a complete turnaround
- **Minimum time to tow:** Minimal number of minutes a specific aircraft type has to be at the airport to justify towing it away to a remote parking location between disembarking and embarking passengers
- **Average flight time:** Average number of minutes it takes a flight to fly to the airport, this time is used to compute the estimated arrival time
- **Maximum delay absorb:** The maximum number of minutes a flight can be delayed before the departure time needs to be delayed to ensure sufficient time at the gate to perform a turnaround

The subset C_i contains all the flights that are conflicting with flight$_i$. A flight is defined to be conflicting with flight$_i$ if its arrival time is past that of flight$_i$ and its arrival time is before the departure time of flight$_i$ + max_delay_i. This definition of conflicting flights is illustrated in Figure 4.16.

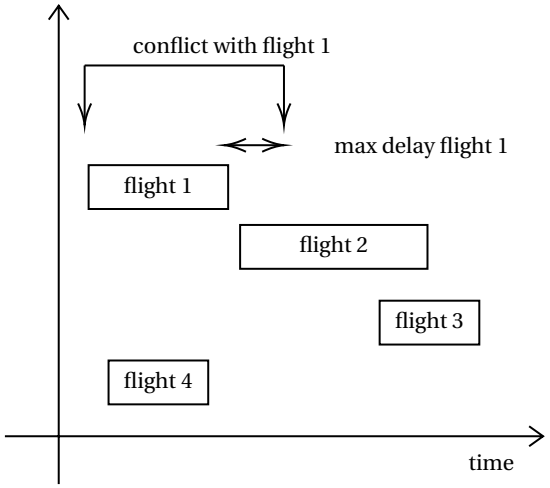

Figure 4.16: In this example, flight 2 and flight 4 are conflicting with flight 1. If the max delay of flight 1 is defined to be zero then flight 2 is no longer in conflict with flight 1

The model to solve the AGAP in this thesis is formulated as a Mixed-Integer Linear Problem (MILP), the formulation has been influenced by the proposed model from Şeker and Noyan (2012). The goal of the model is to assign every flight to a compatible stand while minimizing the delay for passengers and the number of remote assigned aircraft. This can be deducted to the following decision variables which are used in the subsequent discussed models:

$$x_{i,j} = \text{binary variable, 1 if flight } i \text{ is assigned to gate } j\text{, and 0 if not} \tag{4.11}$$
$$a_i = \text{integer variable, minutes that flight } i \text{ has to wait for the gate} \tag{4.12}$$
$$t_i = \text{integer variable, minutes that flight } i \text{ can shorten the turnaround time to depart on time} \tag{4.13}$$
$$r_i = \text{binary variable, 1 if flight } i \text{ is assigned to a remote parking, and 0 if not} \tag{4.14}$$
$$b_i = \textit{integer variable, minutes that flight } i \text{ has as a buffer before the sheduled arrival time} \tag{4.15}$$

Minimize the number of gates used
Minimizing the number of gates used is a new objective proposed in this research to investigate if the sliding time window implementation will benefit from the extra availability of gates. When assigning the flights to as few gates as possible, it is expected that there will be "overflow" gates available which will have no or very few initial assigned flights. These gates can then be assigned to conflicting flights during the third phase of the framework, the reassignment.

The objective metric is as follows:

$$\min : \sum_{i \in F} \sum_{j \in G_{ci}} \boldsymbol{X}_i * x_{i,j} + \boldsymbol{A}_i * a_i + \boldsymbol{R}_i * r_i + t_i \tag{4.16}$$

Subject to:

$$\sum_{j \in G_{ci}} x_{i,j} + r_i = 1, \quad \forall i \in F \tag{4.17}$$
$$a_k - a_i + t_i - DA_{i,k} - 1440x_{i,j} - 1440x_{k,j} \geqq -2880, \quad \forall i \in F, \quad \forall k \in C_i, \quad \forall j \in G_{ci,ck} \tag{4.18}$$
$$t_i \leqslant a_i \quad \forall i \in F \tag{4.19}$$
$$a_i \in [0,\ max\_delay] \quad t_i \in [0, max\_delay\_absorb] \tag{4.20}$$
$$DA_{i,k} = \text{minutes between departure flight } i \text{ and arrival flight } k \tag{4.21}$$

In which $\boldsymbol{X}$, $\boldsymbol{A}$ and $\boldsymbol{R}$ are constants used to improve the convergence of the model by ensuring that every decision variable has a unique value. They are computed using the following functions:

$$\mathbf{X}_i = \left(\frac{normalized\_pax_i}{3} + 2 * \frac{normalized\_gate_i}{3} + 1 \right) * 1500 \tag{4.22}$$

$$\boldsymbol{A}_i = (normalized\_pax + 1) * 500 + 6000 \tag{4.23}$$

$$\boldsymbol{R}_i = (normalized\_pax_i + 1) * 500000 \tag{4.24}$$

with:

$$normalized\_pax_i = \frac{pax_i - max\_pax}{max\_pax - min\_pax} \tag{4.25}$$

$$normalized\_gate_i = \frac{max\_turnaround - (dep_i - arr_i - req\_turnaround)}{max\_turnaround - min\_turnaround} \tag{4.26}$$

leading to the following ranges for $\boldsymbol{X}$, $\boldsymbol{B}$, $\boldsymbol{A}$ and $\boldsymbol{R}$:

$$\boldsymbol{X}_i \in [1500,\ 3000] \quad \boldsymbol{A}_i \in [6500, 7000] \quad \boldsymbol{R}_i \in [500000, 1000000] \tag{4.27}$$

The constants in the above equations have been chosen for the case where *max_delay* was set to 20 minutes and *max_buffer* was set to 15 minutes. Combining the constants with these maximum values ensures that the model prioritizes all the variables in the correct sequence. First the model tries to assign all the flights to a gate. Here the model tries to assign all the flights to the gates with lowest gate penalty score. Distinction between two flights for the same gate is made by counting the number of passengers onboard. Flights with more passengers are favored over those with less passengers. In the case of gate conflicts, the model first tries to delay flights. If that does not resolve the gate conflicts then the model will assign the flight with the fewest passengers from the conflicting flights to a remote stand.
The individual functions of $\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{R}$ have been defined such that minimizing the objective function yields to better real world solutions. $\boldsymbol{X}$, is defined such that the model favors assigning flights to gates with a lower gate penalty score. These scores can be set by either the airline or by the airport authorities. For this case they have been set to mimic how an airport assigns scores to each gate. An overview of the airport layout used can be seen in Figure 4.13. The figure shows that the outer gates have a lower gate penalty score than the inner gates. This is done because the outer gates are easier to access for aircraft since there is no risk of colliding with an aircraft simultaneously reversing from the opposite gate. $\boldsymbol{A}$, is defined such that the model first delays flights with less passengers in favor of flights with more passengers on board. This to minimize the impact on passenger comfort. A similar approach has been taken for $\boldsymbol{R}$, which represents the remote assignment of an aircraft. Here, there is a very high penalty added so that the model first considers slightly delaying a few flights before it resorts to remotely handling an aircraft.
$DA_{i,k}$, indicates the overlap between two consecutive flights. If they overlap than $DA_{i,k}$ is negative and vice versa for flights that are not overlapping if there is no delay.

Constraint 4.17, assigns every flight to either a gate or a remote parking position. Constraint 4.18, ensures that overlapping flights are not assigned to the same gate simultaneously. Delaying a flight can result in flights that are not overlapping anymore and thus the pair can be assigned to the same gate. The values have been defined so that even for flights with a very long turnaround time at the airport, there are no overlapping flights. The maximum overlap is set to be one full day, $60 \times 24 = 1440$. For application to cases where the turnaround time exceeds a full days these numbers need to be scaled accordingly. Constraint 4.19, ensures that the amount of delay absorption is not greater than the assigned delay for that flight. Removing this constraint will result in overlapping flights being assigned to the same gate.

Maximize the buffer time between flights
Maximizing the buffer time between consecutive flights is the second objective and has been well discussed in literature. It has proven to be capable of producing schedules that exhibit robustness properties (Bolat, 2000; Dorndorf et al., 2012). The model formulation in this chapter has been influenced by these models. For this research it was desired to have a model which distributes the flights evenly over the gates and tries to maintain a constant buffer of 15 minutes between all the flights. Cheung (2018), proposed a method which is able to predict the flight arrival time with an accuracy of 10 minutes. This means that as soon as the aircraft departs, the model can predict at what time the flight will arrive at its destination. The presence of a 15 minutes buffer

between flights will thus ensure that the found schedule is able to absorb these small perturbations in the arrival times and thus provide a robust solution.

The proposed model maximizes the buffer time between consecutive flights, while minimizing the delay of individual aircraft and the number of remotely handled aircraft. The model discussed in the previous section is used as a starting point and extended with the following decision variable. The newly added decision variable describes the amount of minutes the gate is free before the arrival of the flight:

$$b_i = \text{integer variable, amount of buffer minutes before flight } i \text{ arrives} \tag{4.28}$$

The objective metric is formulated as:

$$\min : \sum_{i \in F} \sum_{j \in G_{ci}} \boldsymbol{X}_i * x_{i,j} - \boldsymbol{B}_i * b_i + \boldsymbol{A}_i * a_i + \boldsymbol{R}_i * r_i + t_i \tag{4.29}$$

Subject to:

$$\sum_{j \in G_{ci}} x_{i,j} + r_i = 1, \quad \forall i \in F \tag{4.30}$$

$$a_k - a_i + t_i - DA_{i,k} - b_i - 1440x_{i,j} - 1440x_{k,j} \geqq -2880, \quad \forall i \in F, \quad \forall k \in C_i, \quad \forall j \in G_{ci,ck} \tag{4.31}$$

$$t_i \leqslant a_i \quad \forall i \in F \tag{4.32}$$

$$b_i \in [0,\ max\_buffer] \quad a_i \in [0,\ max\_delay] \quad t_i \in [0, max\_delay\_absorb] \tag{4.33}$$

$$DA_{i,k} = \text{minutes between departure flight } i \text{ and arrival flight } k \tag{4.34}$$

In which $\boldsymbol{X}$, $\boldsymbol{B}$, $\boldsymbol{A}$ and $\boldsymbol{R}$ are used to improve the convergence of the model by ensuring that every decision variable has a unique value. They are computed using the following functions:

$$\boldsymbol{X_i} = 15000, \quad \forall i \in G_{ci} \tag{4.35}$$

$$\boldsymbol{B}_i = \left(1 + \frac{normalized\_pax_i}{3} + 2 * \frac{normalized\_gate_i}{3}\right) * -500 \tag{4.36}$$

$$\boldsymbol{A}_i = (normalized\_pax + 1) * 500 + 15000 \tag{4.37}$$

$$\boldsymbol{R}_i = (normalized\_pax_i + 1) * 500000 \tag{4.38}$$

with:

$$normalized\_pax_i = \frac{pax_i - max\_pax}{max\_pax - min\_pax} \tag{4.39}$$

$$normalized\_gate_i = \frac{max\_turnaround - (dep_i - arr_i - req\_turnaround)}{max\_turnaround - min\_turnaround} \tag{4.40}$$

leading to the following ranges for $\boldsymbol{X}$, $\boldsymbol{B}$, $\boldsymbol{A}$ and $\boldsymbol{R}$:

$$\boldsymbol{X}_i \in [1500,\ 4500] \quad \boldsymbol{B}_i \in [-250,\ -1000] \quad \boldsymbol{A}_i \in [6500, 7000] \quad \boldsymbol{R}_i \in [500000, 1000000] \tag{4.41}$$

The constants in the above equations have been chosen for the case where *max_delay* was set to 20 minutes and *max_buffer* was set to 15 minutes. Combining the constants with these maximum values ensures that the model prioritizes all the variables in the correct sequence. First the model tries to assign all the flights to a gate, then it will shuffle the flights around to maximizes the buffer time between consecutive assigned flights. In case of gate conflicts the model will first try to delay flights. If that does not resolve the gate conflict it will assign the conflicting flight with the least amount of passengers to a remote position. This is the desired behavior since the changes will affect the least amount of passengers.
The individual functions of $\boldsymbol{X}$, $\boldsymbol{B}$, $\boldsymbol{A}$ and $\boldsymbol{R}$ have been defined such that minimizing the objective function yields to better real world solutions. $\boldsymbol{X}$, is defined to be constant for every flight, this means that there is no

preference for a flight to be assigned to a certain gate. The value is set to 15000, so that the objective function will always remain positive if b_i is set to max 15 minutes. Then $\boldsymbol{B}$ is a mixture of airport preference and passenger comfort, it will tries to increase the buffer for flights with a tight turnaround e.g. where the minimum required turnaround time is equal to the scheduled turnaround time. A small factor is included which biases the buffer time towards flights with more passengers so that for the initial assignment flights with more passengers are scheduled more robust. $\boldsymbol{A}$, is defined such that it delays flights with less passengers in favor of flights with more passengers. This to minimize the impact on passenger comfort. A similar approach has been taken for $\boldsymbol{R}$, which represents the remote assignment of an aircraft. Here, a very high penalty is added so that the model first considers slightly delaying a few flights before it resorts to assigning a flight to a remote parking position for remote handling of the aircraft.
$DA_{i,k}$, indicates the overlap between two consecutive flights. If they overlap than $DA_{i,k}$ is negative and vice versa for flights that are not overlapping if there is no delay.

Constraint (4.30), ensures that every flight is assigned to a gate or a remote parking position. Constraint (4.31), ensures that no overlapping flights are assigned to the same gate simultaneously. Delaying a flight can result in flights that are not overlapping anymore and thus the pair can be assigned to the same gate. The values have been defined so that even for flights with a very long turnaround time at the airport, there are no overlapping flights. The maximum overlap is set to be one full day, $60 \times 24 = 1440$. For application to cases where the turnaround time exceeds a full days these numbers need to be scaled accordingly. Constraint (4.32), ensures that the amount of delay absorption is not greater than the assigned delay for that flight. Removing this constraint results in overlapping flights being assigned to the same gate.

In phase 2, the flights of the generated flight schedule have been assigned to gates by one of the above described objectives. These models have been solved yielding an optimal initial gate assignment. This gate assignment is then used in phase 3 where the sliding time window is used to resolve gate conflicts in real time during daily operation.

4.4.3. Phase 3 - Operational phase: sliding time window implementation

The operational phase is used during daily operation. In this phase, the framework is run at a set time interval, for each time interval the framework computes the gate reassignments that are required to mitigate gate conflicts. These gate conflicts arise due to aircraft deviating from their scheduled arrival and departure time. In daily operation, the planning horizon of the reassignment model shifts in time, i.e. the sliding time window. Each time the reassignment model runs, the following steps are taken in chronological order:

1. Update the arrival times in the time schedule
2. Initialize the reassignment model based on the updated flight schedule
3. Solve the model and write the result to the timetable

In the first step, the framework updates the flight schedule. As soon as a flight has departed from its origin, the Actual Time of Departure (ATD) is stored in the FlightSchedule-class. Using the average flight time, the model can compute Actual Time of Arrival (ATD) for the flight. This ATD is stored in the FlightSchedule-class as the new arrival time and based on this time, the framework will determine the optimal flight to gate assignment. As all the departed flights the flight schedule have been updated for the given time window, the framework proceeds to the second step in which the model will be initialized. The model is initialized based on the previously found flight to gate assignment. As the model is initialized, the framework will move to the next step. In step three, the model is solved and the assigned gates will be written to the FlightSchedule-class to be used for the next time window.

The model used for the gate reassignment has the goal to resolve all the gate conflicts that are induced by flight schedule perturbations while minimizing the inconvenience for the passengers and costs for airlines and the airport. In order to do this the model has the following options, listed in order of degrading preference:

- Reassign the flight to an empty gate;
- Split flights into disembarking, remote parking and embarking parts;
- Delay a flight on the apron to wait for the assigned gate to become available;

– Assign the flight to a remote stand for handling.

The reassignment of flights is based on the found gate assignment from the previous iteration. The model is optimizing for a minimal number of gate changes and takes the flight time left for each flight into account during reassignment. Based on the remaining flight time left, there are several restrictions to change the assigned gate of a flight:

– 1h before arrival: no gate changes are allowed;

– 1-4hrs before arrival: no pier change is allowed;

– 4+hrs before arrival: all compatible gate changes are allowed.

These numbers have been discussed with an expert from AAS. It was concluded that with these numbers the impact for the passengers is reduced to a minimum while it still gives the model enough options to change flights in order to resolve gate conflicts. During the 1 to 4 hours before arrival of the flight, most passengers are on their way to the airport and are proceeding through check-in and security. Here they will receive their boarding card on which the current gate for their flight is printed. In this time period the gate of their flight might still change. However, since the pier is not allowed to change anymore, the passengers will still be able to make their flight as they are already in the correct area of the airport. In order to assist the passengers even further, the model will first consider the neighboring gates before assigning the flight to further away gates at the same pier.

Similar to the model proposed for the second phase of the framework, a flight can also be split into a disembarking part, a remote parking part and an embarking part. Here the first and last part will be assigned to the same or different gates depending on the length of the turnaround and gate availability. The guidelines regarding splitting flights have been inspired by the RASAS[5] from AAS (Prent, 2019). The eligibility of a flight for, depends on different factors such as the aircraft size and scheduled time at the airport. In this research a distinction is made between narrow-body and wide-body aircraft. The rules for splitting a flight are assuming these two categories. For a wide-body aircraft, the turnaround time needs to be above 210 minutes. The disembarking part is then assigned to a gate for 75 minutes and the embarking part will be assigned to a gate for 85 minutes. For a narrow-body aircraft these numbers are smaller: 170, 55 and 65 minutes for respectively the turnaround time, disembarking and embarking of passengers.

Similar to the model proposed for the second phase, flights can be delayed on the apron for up to 20 minutes and wait for their assigned gate to become available. In the case where none of the above options lead to a mitigation for the gate conflict, the conflicting flight with the least amount of passengers will be assigned to a remote parking position.

Summarizing the above leads to the following objective metric:

$$\min : \sum_{i \in F} \sum_{j \in G_{ci}} \boldsymbol{X}_i * x_{i,j} + \boldsymbol{A}_i * a_i + \boldsymbol{R}_i * r_i + t_i \tag{4.42}$$

The reassignment model is subjected to the same constraints found in the model optimizing for minimal gate usage of the second phase of the framework. The sole difference between the two are the functions used to determine $\boldsymbol{X}$, $\boldsymbol{B}$, $\boldsymbol{A}$ and $\boldsymbol{R}$.

$\boldsymbol{X}$, depends as previously stated on the distance the flight is from the airport. The pseudo code in Algorithm 4, explains how these different values for $\boldsymbol{X}$ are computed based on the remaining flight time. The underlying algorithm shows that the model aims to assign as few flights as possible to a remote stand. The weights of the different values of $\boldsymbol{X}$ are purposely chosen to be extreme so that the model is converging fast which is beneficial for real world application.

[5]Regulation Aircraft Strand Allocation Schiphol

Algorithm 4 Process to find the correct delay distributions for the first phase of the framework

input: F_i, T_i **where** $F_i =$ flight, $T_i =$ remaining flight time of F_i
output: $X_{i,j}$ for flight F_i
require: $F_i, T_i \neq \{\}$
procedure DETERMINE $X_{i,j}$ FOR FLIGHT F_i WITH REMAINING FLIGHT TIME T_i
  $P = 2,000,000$ ▷ sufficiently large value for p as penalty for invalid options
  for i **in** F **do**
    $C \leftarrow$ current gate of F_i
    for j **in** G **do**
      if $F_i \leqslant 60$ **then**
        $X_{i,j} \leftarrow 1$ **if** $X_{i,j} = C$
        $X_{i,j} \leftarrow P$
      end if
      if $C =$ remote **then**
        $X_{i,j} \leftarrow 1$ ▷ favor every gate if currently remote assigned
      end if
      if $60 < F_i \leqslant 240$ **then**
        $X_{i,j} \leftarrow 1$ **if** $X_{i,j} = C$
        $X_{i,j} \leftarrow pax_i * 1000$ **if** $X_{i,j} =$ neighbor of C
        $X_{i,j} \leftarrow pax_i * 1000 + 1000$ **if** $X_{i,j} =$ at same pier as C
        $X_{i,j} \leftarrow pax_i * 4000 + 2000$ **if** $X_{i,j} =$ other pier than C
      end if
      if $F_i > 240$ **then**
        $X_{i,j} \leftarrow 1$ **if** $X_{i,j} = C$
        $X_{i,j} \leftarrow 1000$ **if** $X_{i,j} \neq C$
      end if
    end for
  end for
end procedure

In phase 3, the gate reassignment model with the implemented sliding time window uses the initial gate assignment from phase 2. As the daily operation starts, the reassignment model is run at a set time interval. Gate conflicts are resolved for all flights in the FlightSchedule-class and the gate changes are stored in the flight schedule to be used for the next time interval.

4.5. Validating the aircraft to gate assignment framework with AAS

The above described framework will be validated with the data obtained from AAS. For this, the framework will use actual flight schedules obtained for a day of operation from a subset of piers at AAS. The subset of piers is dependent on the results obtained from theoretical case. The main driving factor is the distribution of narrow-body and wide-body aircraft in the time schedule. For the theoretical case, the distribution between both aircraft types is varied in order to determine for which distribution of aircraft types the performance of the framework is optimum. The piers at AAS are distributed between the Schengen and non-Schengen, flights originating from outside the Schengen area[6] need to arrive at non-Schengen piers to allow for the passengers to pass to immigration before transferring to a connecting Shengen flight or exiting the airport. Some piers at Schiphol are able to handle both type of flights since they have a separated de-boarding corridor leading straight through immigration if that is required for a flight. For flights that do not require immigration, the corridor can be closed and passengers enter the terminal area directly from the gate. Gates with this option are called swing gates. In Table 4.5, an overview of the piers at AAS together with their border region is given.

[6]Originated from a European agreement between member and non-member states of the European Union which abolished border checkpoints between signed countries. The agreement was signed in 1985 near the town of Schengen in Luxembourg

Table 4.5: Different piers at AAS and their border status

	Pier						
	B	C	D	E	F	G	H/M
Schengen	Yes	Yes	Yes	No	No	No	Yes
non-Schengen	No	No	Yes	Yes	Yes	Yes	Yes
Swing gate	No	No	Yes	No	No	No	Yes

Depending on the performance of the model in the theoretical case, either it is chosen to use the Schengen or the non-Schengen piers for the validation process. Using the Schengen piers would represent a flight schedule compromising mainly of narrow-body aircraft and using the non-Schengen piers would represent a flight schedule with mainly wide-body aircraft. In chapter 6, the validation process is described into more detail as it is dependent on the results presented in chapter 5.

The data set obtained from AAS spanned 6 months of operation, the first 26 weeks of the year 2019. During this period there are two national holiday periods, February 16th till March 6th and April 27th till May 5th. During this period it is expected that there is more chaos at the airport since there are more seasonal flights, increasing congestion. Selecting days from these busy periods together with days outside the holiday period, verifies if the framework is outperforming the current situation. Currently there are extra gate changes during these busy periods as shown in figure Figure 4.17. The figure shows a trend of increasing number of average weekly flights as summer approaches. The effects of the holiday periods are visible at week 9, 8, 17 and 18, in these weeks the average number of scheduled flights increases. The large peak at week 11 is caused by a strike organized by the trade union for airport staff (Schiphol, 2019). Other peaks and dips in the graph are caused by weather disruptions such as dense fog and snow.

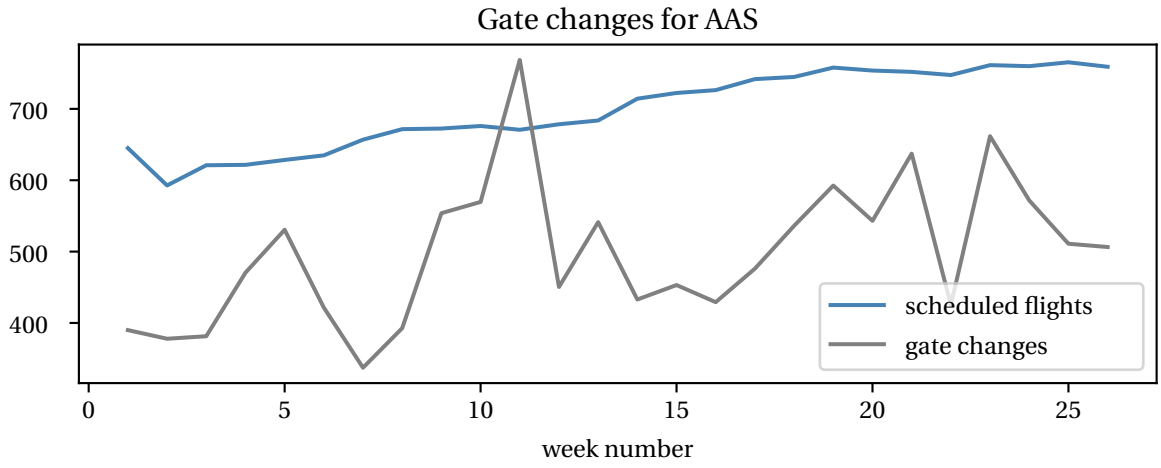

Figure 4.17: Average number of daily flights per week and the average number of daily gate changes per week at AAS

5

Theoretical case - Implementation and simulation results

In this chapter, the flight to gate assignment framework is applied to the theoretical case. The theoretical case uses generated data which means that all the three phases of the framework are used. The theoretical case is used to determine which of the two objectives from the second phase of the framework are the most effective for the third phase of the framework, reassignment phase. The most effective objective is determined based on several KPI's that have been specified in order to compare the found solutions or flight to gate assignments. The simulation starts with the definition of three distinct scenarios. The difference between each scenario, is the distribution of wide- and narrow-body aircraft in the generated flight schedule. For each of the three scenarios, hundred different runs have been performed of the model, each run with distinct perturbations in the flight schedule. The runs and scenarios are similar for both the objectives used in phase 2 of the framework. In this chapter, the minimize gate usage objective and maximize buffer time objectives are referred to as *mingate* and *maxbuf* respectively.

This section will start with an overview of the three scenarios, a brief description explaining the similarities and differences. Next, the results of the runs will be discussed and how the perturbations in the flight schedule are affecting the runs performed for each scenarios. Then, the results from the simulation will be discussed and the most effective objective for each scenario will be determined based on a set of KPI's.

5.1. Scenario generation

Three different scenarios have been defined, the distribution of wide- and narrow-body aircraft in the flight schedule differ between the scenarios. The first scenario has a total of 116 flights of which 79 and 37 are respectively wide- and narrow-body aircraft. The second scenario has a total of 128 flights of which 66 and 62 are respectively wide- and narrow-body aircraft. The last scenario has a total of 151 flights of which 43 and 108 are respectively wide- and narrow-body aircraft. The total number of flights differ from scenario to scenario since narrow-body aircraft require a shorter turnaround time than wide-body aircraft. Thus more flights can fit within the flight schedule operational bounds of the day. The variation in aircraft type has been chosen to identify for which type of terminal operation the proposed flight to gate assignment framework performs best. Terminals that handle predominately wide-body aircraft or long haul flights as simulated by scenario 1. Can know far in advance that a flight will arrive early or late due to the long flight time. Scenario 3 on the other hand, simulates a terminal that predominantly handles narrow-body aircraft or short-haul flights. Due to their shorter flight time it less far known in advance that a flight will arrive late or early.
The number of gates and thus the layout of the fictional airport remains unchanged between the three scenarios. The flight schedule for each scenario is visualized using a Gantt chart and shown in Appendix A.

5.2. Effect of varying sliding time window size

The size of the sliding time window indirectly dictates the amount of flights for which the scheduled arrival times are updated with the actual arrival time. A sliding time window that is too narrow reduces the number of updated flights and thus possibly increase the number of gate changes. In this section, the effects of varying the sliding time window is studied. The sliding time windows size is investigated for the first and third scenario, the distribution between wide and narrow-body aircraft is respectively 70%-30% and 30%-70%. For both scenarios, 20 runs have been performed for each size of sliding time window. The window size is varied between 15 and 45 minutes with a 15 minute step size. It is estimated that the effect of varying the sliding time window size depends on the distribution between wide- and narrow-body aircraft. For a time schedule containing predominantly narrow-body aircraft it is estimated that a narrow time window will be beneficial since flights will be updated sooner with their actual arrival time. Due to the short flight times the reassignment model has more reassignment options available for each flight if the sliding time window is narrow. For wide-body aircraft, it is estimated that a wider sliding time window is beneficial as this allows for more flights to be updated with their actual arrival time. For wide-body aircraft, flight time is in general longer compared to narrow-body aircraft, thus there are no immediate penalties for the reassignment model if the time window size is wider and there are thus less frequent updates of the flight to gate assignment.
The result for both scenarios are shown in Table 5.1. The table shows that for timetables with predominantly wide-body aircraft, a sliding time window size of 45 minutes yields the best performance. For time schedules with predominately narrow-body aircraft, it more effective to use a shorter sliding time window size of 15 minutes.

Table 5.1: Gate changes for each scenario for different sliding time window sizes

	sliding time window size		
	15 min	30 min	45 min
Scenario 1	55	55	48
Scenario 2	54	52	53
Scenario 3	55	63	63

For this report, the time window size is set to 15, 30, 45 minutes for respectively scenario 1, 2 and 3.

5.3. Simulation results

The simulation spans three distinct scenarios which differ on the distribution of wide- and narrow-body aircraft. For each scenario there is one flight schedule and for every flight schedule, 100 different delay simulations have been generated. A total of 600 runs have been performed to determine the initial objective, per simulation, that yields the most robust flight to gate assignment. The first objective aims to minimize the number of gates used during the operation. The idea behind this objective is that the resulting flight to gate assignment has one or multiple gates with very few flights assigned to them. If during the daily operation a perturbation occurs in the flight schedule, the reassignment model has empty gates available to reassign the conflicting flight to, yielding robustness. The initial flight to gate assignment for the second scenario, optimized for the minimal usage of gates is shown in Figure 5.2. The Gannt-chart shows that gates with a high gate penalty score such as gate 7,8,9,10,11 and 12 are assigned less flights than gates with a low gate penalty score such as gate 1,2,3,4,5,14,15,16,17,18 and 19.
The second objective, aims to maximize the number of minutes between two consecutive flights assigned to the same gate. The idea behind this objective is that during daily operation, slight perturbations in the flight schedule can be absorbed by the flight to gate assignment rather than have flights conflicting with each other and thus requiring a gate reassignment. The initial flight to gate assignment for the second scenario, optimized for the maximization of buffer time between consecutive flights is shown in Figure 5.1. The Gannt-chart shows that there is a, where possible, constant separation of at least 15 minutes between flights assigned to the same gate. In both the figures, the red and green color indicate wide- and narrow-body aircraft respectively.

Interpreting the result of the runs is performed by defining as set of KPI's and performance graphs that will assist in determining which of the initial assignment objectives is best performing for each scenario. The

KPI's have been chosen to reflect both the passenger comfort as well as operational efficiency:

- $\sum$ passengers × delay
- Delayed narrow-body flights
- Delayed wide-body flights
- Number of gate changes
- Remote handled aircraft
- Number of hours before arrival a flight is reassigned

Performance graphs and tables indicate how the reassignment model is performing over time and if the initial flight to gate assignment exhibits robustness properties. The number of gate changes, combined with the hours before a flights arrival the gate has been changed, provide a robustness measure. Initial flight to gate assignments which see fewer gate reassignments and if they occur have them earlier during the simulation runs, are said to be robust. In addition, the number of remote assigned aircraft indicate the ability of the initial flight to gate assignment to solve for the delays in such manner that all the flights can be handled at gates. In some cases due to excessive flight schedule perturbations this is inevitable and thus it is important to investigate if both initial objectives for a specific delay scenario assign aircraft to a remote handling position.

The time before a flights x-hour before arrival plotted to visualize the robustness of the initial schedule. Gate changes far before the flight arrives are less troublesome for the airline and airport than short notice gate changes. The number of flights that have been updated in each time frame have been plotted against the gate changes for that time frame to visualize the robustness of the initial schedule. A more robust initial assignment sees fewer gate reassignments as actual flight arrival times are updated via the sliding time window. For each of the scenarios, the number of gate changes for each run have been visualized to determine relative performance between both initial objectives.

Figure 5.1: Initial gate assignment where the buffer between flights is maximized for scenario two. Where green and red indicate respectively; narrow-body, wide-body flights. Blue represents narrow- or wide-body flights that have been split

Figure 5.2: Initial gate assignment where the minimal number of gates is used for scenario two. Where green and red indicate respectively; narrow-body, wide-body flights. Blue represents narrow- or wide-body flights that have been split

5.3.1. Scenario 1, 70% wide-body - 30% narrow-body

The first scenario, simulates that the majority of airplanes visiting the fictional airport are wide-body aircraft, the flight schedule used in the scenario is visualized in Appendix A. The flight schedule has been solved by the Gurobi solver for both objectives from the second phase of the framework. Solution time for both the objectives was fast being under five minutes which is sufficient for practical use. The found, initial flight to gate assignments from the second phase is used for the third phase of the framework, the reassignment model. In this phase, 100 runs of the reassignment model are performed on each of the found initial flight to gate assignments. Each each run has unique flight schedule perturbations. In this section, the 200 runs performed to investigate the performance of the initial flight to gate assignments are discussed.

The size for the sliding time window was determined in page 41 and set to 45 minutes. This means that every 45 minutes, the reassignment model is run and updates the arrival time for all the flights that have departed their origin in the last 45 minutes. The run time for the gate reassignment model from phase 2 ranged between 2 minutes and a few seconds. The run time mainly depends on the number of flights that are updated and the following gate conflicts, an increasing number of gate conflicts leads to a longer run time for the model. As the day progresses, run time of the model steadily decreased. This is caused by the fact that the number of flights that can be changed are decreasing since departed aircraft can no longer be assigned to a different gate, this reduces complexity and thus run time.

Figure 5.3, shows a box plot that describes the number of gate changes for each scenario. The figure shows that the objective of maximizing the buffer time between consecutive flights yields less gate changes than when the objective is to minimize the number of gates used.

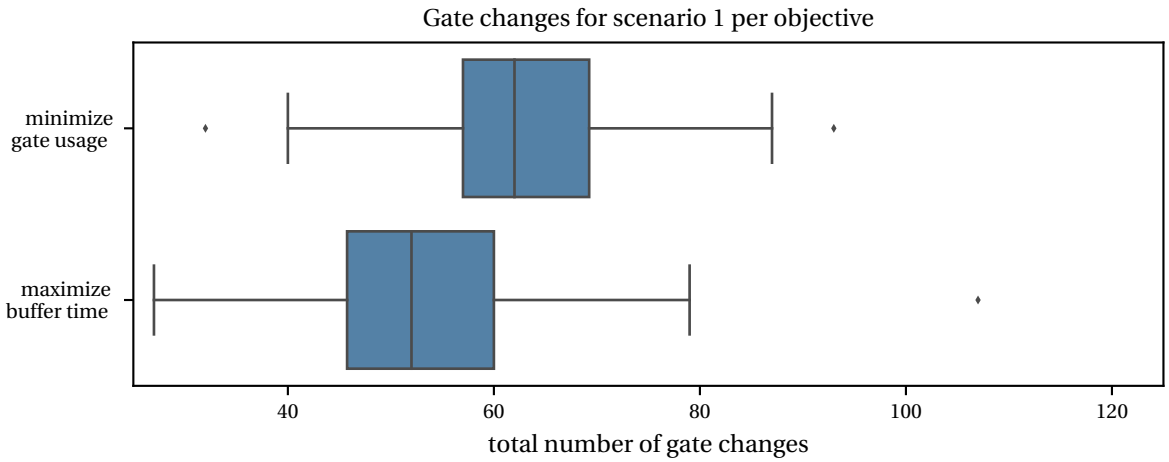

Figure 5.3: Box plot indicating the total number of gate changes per objective

The total number of gate changes only provides a partial indication about the robustness of the flight to gate assignment. In order to precisely determine which initial assignment is more robust, it is required to study at which point in time the gate change for a certain flight occurred. The closer this time is to the arrival time of a flight, the less desired a gate change is. In page 44, a box plot is used to visualize the aggregated result of all the runs performed. The median of the box plot shows that the median for the *maxbuf*-objective is lower than that for *mingate*, this means that the *maxbuf*-objective is resulting in a more robust initial flight to gate assignment.

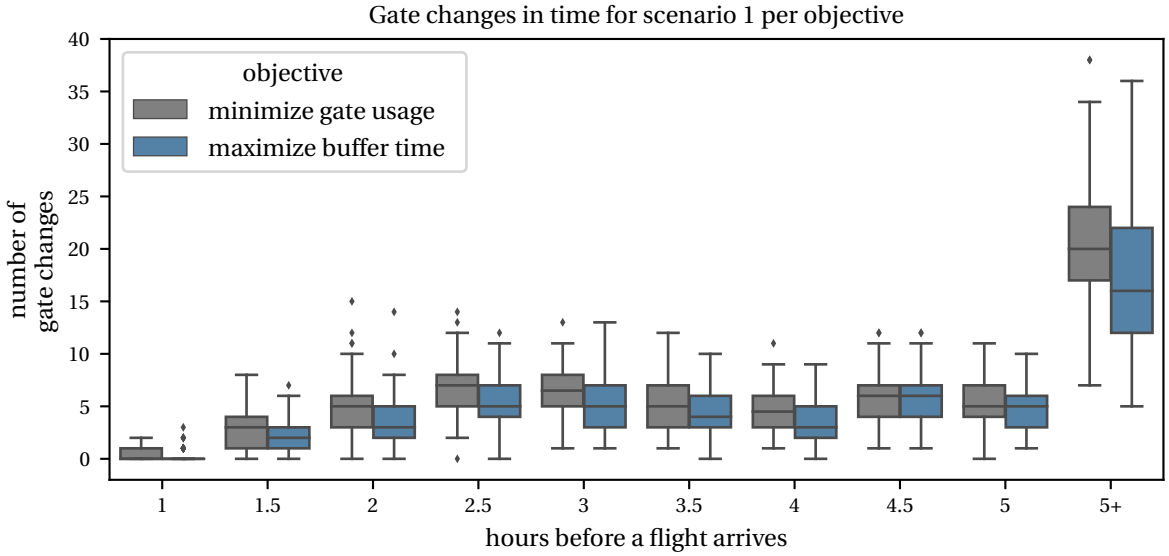

Figure 5.4: Box plot indicating the point in time at which gate changes occurred aggregated in hours before a flight arrives at the airport and grouped by objective

Robustness of the initial flight to gate assignment is beneficial for the passengers but it is more of interest to the airlines and airports as it aims to improve operational efficiency. From a passenger perspective, the number of delayed flights, remote handled flights and the number of delayed minutes are of greater interest as they reduce the comfort of the passengers. In page 45, a box plot is used to visualize the number of delayed flights. The figure shows that the *maxbuf*-objective sees on average less delayed flights per run and thus a higher passenger comfort level. The size of the whiskers and box for the *maxbuf*-objective furthermore indicate that the results from the *maxbuf*-objective are more stable since the variance in the solution is smaller than the variance seen with *mingate*-objective.

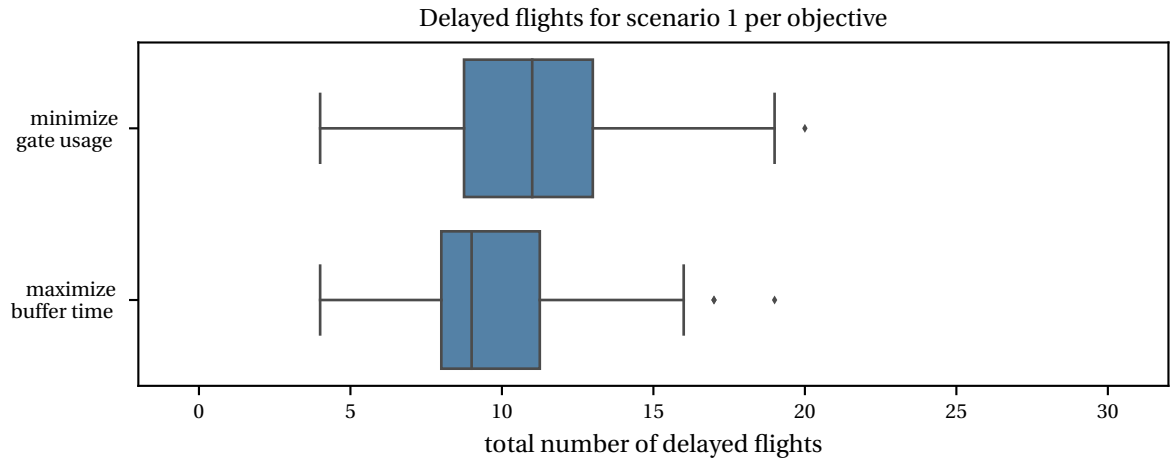

Figure 5.5: Box plot indicating the total number of delayed flights per objective

Apart from the operational efficiency, the reassignment model aims to minimize the inconvenience for the passengers and thus increase the passenger comfort. This is measured in more detail by computing af few KPI's. For all the KPI's a differentiation between narrow- and wide-body aircraft has been made. This to investigate if certain objectives are beneficial for a certain type of aircraft. The first two KPI's are computed by summing the arrival and departure delay for all the flights in a run. The last KPI is computed by multiplying the the delay of each delayed flight with the number of passengers on that flight. The reassignment model described in chapter 4, prioritizes delaying of flights with less passengers. For all the above mentioned KPI's, the mean and standard deviation have been determined over the runs in the simulation. The mean or average indicate the average delay while the standard deviation indicates how robust the schedule is, as it is desirable to have a low mean with a low standard deviation.
In page 45, the above mentioned KPI's are presented, the table shows that, on average, narrow-body flights experience more delay than wide-body aircraft. This is expected since 70% of the flights in the flight schedule are wide-body aircraft. Comparing the standard deviations fro the arrival and departure delay for both objectives shows that the *maxbuf*-objective produces marginally better results regarding flight delays. Comparing the summed passenger delay confirms that the *maxbuf*-objective outperforms the *mingate*-objective in terms of passenger comfort and robustness of the initial flight schedule. This concludes that the *maxbuf*-objective yields on average the highest operation efficiency and passenger comfort for time schedules where the majority of flights are served by wide-body, long-haul aircraft.

Table 5.2: Results for scenario 1, overview of the KPI's used to compare the performance of the reassignment model for both initial flight to gate assignment objectives

Scenario 1 - 70% wide-body, 30% narrow-body					
KPI		*mingate*: Minimize gate usage		*maxbuf*: Maximize buffer time	
		narrow-body	wide-body	narrow-body	wide-body
Arrival delay	mean	37.8	28.2	37.9	22.5
∑ run in (min)	std	18.7	15.8	18.0	14.9
Departure delay	mean	5.1	3.5	4.7	4.3
∑ run in (min)	std	7.0	4.7	6.1	6.7
passenger delay	mean	4424.0	5111.0	4289.0	4171.2
∑ pax × delay (min)	std	2191.1	2859.6	2132.1	2688.1

5.3.2. Scenario 2, 50% wide-body - 50% narrow-body

The second scenario, simulates that the airplanes visiting the fictional airport are distributed evenly between narrow- and wide-body aircraft, the flight schedule used in the scenario is visualized in Appendix A. The flight

schedule has been solved by the Gurobi solver for both objectives from the second phase of the framework. Solution time for both the objectives was fast being under eight minutes which is sufficient for practical use. The found, initial flight to gate assignments from the second phase is used for the third phase of the framework, the reassignment model. In this phase, 100 runs of the reassignment model are performed on each of the found initial flight to gate assignments. Each each run has unique flight schedule perturbations. In this section, the 200 runs performed to investigate the performance of the initial flight to gate assignments are discussed.

The size for the sliding time window was determined in page 41 and set to 30 minutes. This means that every 30 minutes, the reassignment model is run and updates the arrival time for all the flights that have departed their origin in the last 30 minutes. The run time for the gate reassignment model from phase 2 ranged between 3 minutes and a few seconds. The run time mainly depends on the number of flights that are updated and the following gate conflicts, an increasing number of gate conflicts leads to a longer run time for the model. As the day progresses, run time of the model steadily decreased. This is caused by the fact that the number of flights that can be changed are decreasing since departed aircraft can no longer be assigned to a different gate, this reduces complexity and thus run time.

Figure 5.6, shows a box plot that describes the number of gate changes for each scenario. The figure shows that the objective of maximizing the buffer time between consecutive flights yields less gate changes than when the objective is to minimize the number of gates used.

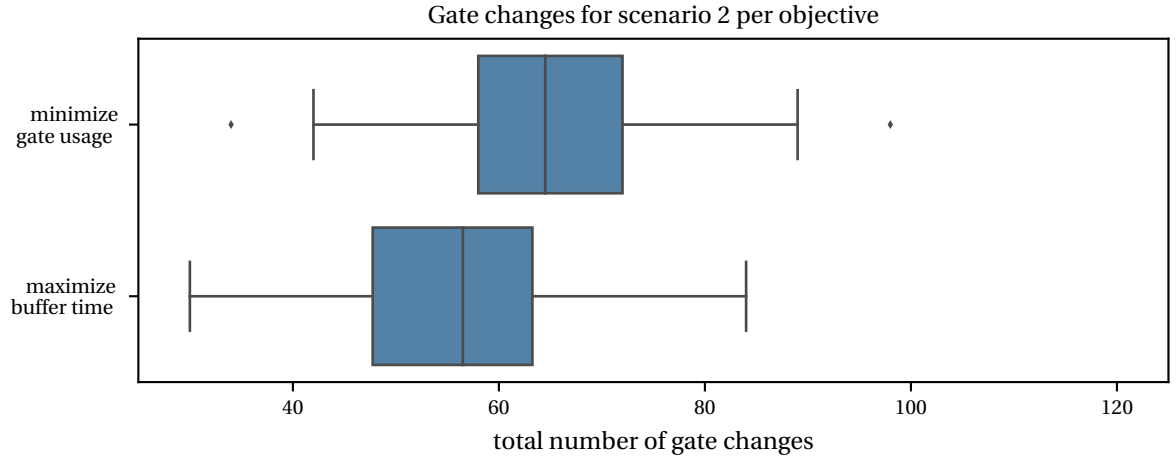

Figure 5.6: Box plot indicating the total number of gate changes per objective

The total number of gate changes only provides a partial indication about the robustness of the flight to gate assignment. In order to precisely determine which initial assignment is more robust, it is required to study at which point in time the gate change for a certain flight occurred. The closer this time is to the arrival time of a flight, the less desired a gate change is. In page 47, a box plot is used to visualize the aggregated result of all the runs performed. The median of the box plot shows that the median for the *maxbuf*-objective is lower than that for the *mingate*-objective, this means that the *maxbuf*-objective is resulting in a more robust initial flight to gate assignment.

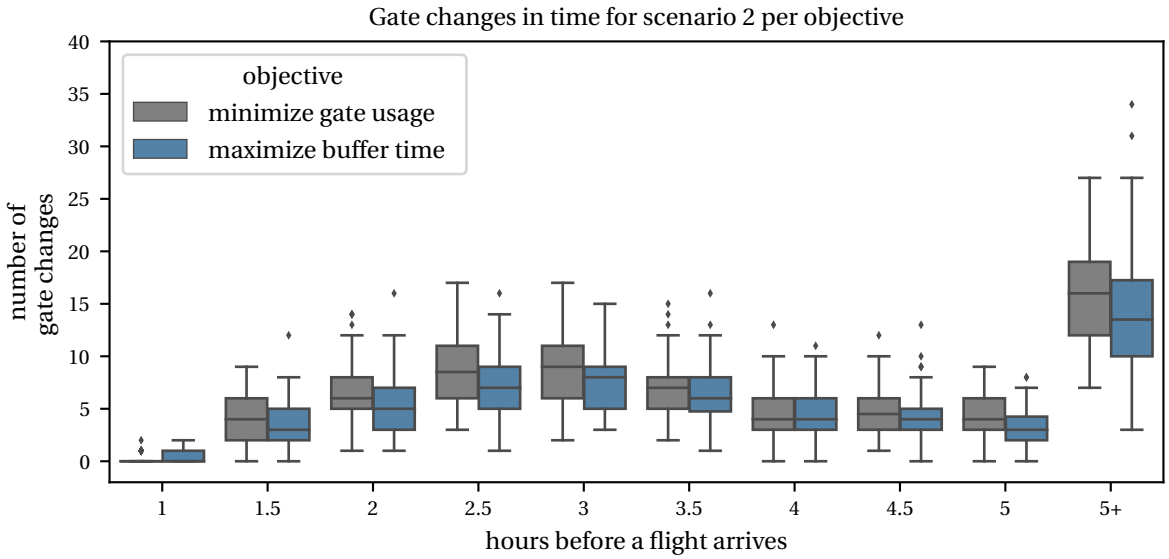

Figure 5.7: Box plot indicating the point in time at which gate changes occurred aggregated in hours before a flight arrives at the airport and grouped by objective

Robustness of the initial flight to gate assignment is beneficial for the passengers but it is more of interest to the airlines and airports as it aims to improve operational efficiency. From a passenger perspective, the number of delayed flights, remote handled flights and the number of delayed minutes are of greater interest as they reduce the comfort of the passengers. In page 47, a box plot is used to visualize the number of delayed flights. The figure shows that the *maxbuf*-objective sees on average less delayed flights per run and thus a higher passenger comfort level. The size of the whiskers and box for the *maxbuf*-objective furthermore indicate that the results from this objective are more stable since the variance in the solution is smaller than the variance seen with the *mingate*-objective.

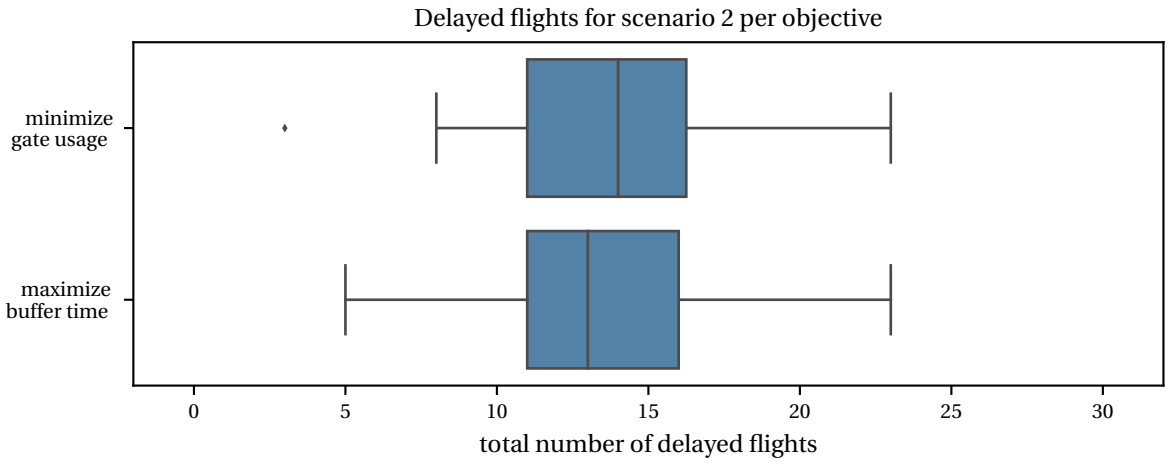

Figure 5.8: Box plot indicating the total number of delayed flights per objective

Apart from the operational efficiency, the reassignment model aims to minimize the inconvenience for the passengers and thus increase the passenger comfort. This is measured in more detail by computing af few KPI's. For all the KPI's a differentiation between narrow- and wide-body aircraft has been made. This to investigate if certain objectives are beneficial for a certain type of aircraft. The first two KPI's are computed by summing the arrival and departure delay for all the flights in a run. The last KPI is computed by multiplying the the delay of each delayed flight with the number of passengers on that flight. The reassignment model described in chapter 4, prioritizes delaying of flights with less passengers. For all the above mentioned KPI's,

the mean and standard deviation have been determined over the runs in the simulation. The mean or average indicate the average delay while the standard deviation indicates how robust the schedule is, as it is desirable to have a low mean with a low standard deviation.
In page 48, the above mentioned KPI's are presented, the table shows that, on average, narrow-body flights experience more delay than wide-body aircraft. This is expected since 70% of the flights in the flight schedule are wide-body aircraft. Comparing the standard deviations fro the arrival and departure delay for both objectives shows that the *maxbuf*-objective produces marginally better results regarding flight delays. Comparing the summed passenger delay confirms that the *maxbuf*-objective outperforms the *mingate*-objective in terms of passenger comfort and robustness of the initial flight schedule. This concludes that the *maxbuf*-objective yields on average the highest operation efficiency and passenger comfort for time schedules where the majority of flights are served by wide-body, long-haul aircraft.

Table 5.3: Results for scenario 2, overview of the KPI's used to compare the performance of the reassignment model for both initial flight to gate assignment objectives

Scenario 2 - 50% wide-body, 50% narrow-body					
KPI		*mingate*: Minimize gate usage		*maxbuf*: Maximize buffer time	
		narrow-body	wide-body	narrow-body	wide-body
Arrival delay	mean	67.1	24.4	55.4	26.8
$\sum$ run in (min)	std	25.5	16.0	23.0	15.4
Departure delay	mean	11.5	5.5	10.0	5.9
$\sum$ run in (min)	std	12.2	6.4	10.0	7.4
passenger delay	mean	8779.6	4927.0	7143.4	5437.8
$\sum$ pax × delay	std	3253.9	3343.5	3001.8	3060.6

5.3.3. Scenario 3, 30% wide-body - 70% narrow-body

The third scenario, simulates that the majority of airplanes visiting the fictional airport are narrow-body aircraft, the flight schedule used in the scenario is visualized in Appendix A. The flight schedule has been solved by the Gurobi solver for both objectives from the second phase of the framework. Solution time for both the objectives was reasonable being under twelve minutes which is sufficient for practical use. The found, initial flight to gate assignments from the second phase is used for the third phase of the framework, the reassignment model. In this phase, 100 runs of the reassignment model are performed on each of the found initial flight to gate assignments. Each each run has unique flight schedule perturbations. In this section, the 200 runs performed to investigate the performance of the initial flight to gate assignments are discussed.

The size for the sliding time window was determined in page 41 and set to 15 minutes. This means that every 15 minutes, the reassignment model is run and updates the arrival time for all the flights that have departed their origin in the last 15 minutes. The run time for the gate reassignment model from phase 2 ranged between 5 minutes and a few seconds. The run time mainly depends on the number of flights that are updated and the following gate conflicts, an increasing number of gate conflicts leads to a longer run time for the model. As the day progresses, run time of the model steadily decreased. This is caused by the fact that the number of flights that can be changed are decreasing since departed aircraft can no longer be assigned to a different gate, this reduces complexity and thus run time.

Figure 5.9, shows a box plot that describes the number of gate changes for each scenario. The figure shows that the objective of maximizing the buffer time between consecutive flights yields less gate changes than when the objective is to minimize the number of gates used.

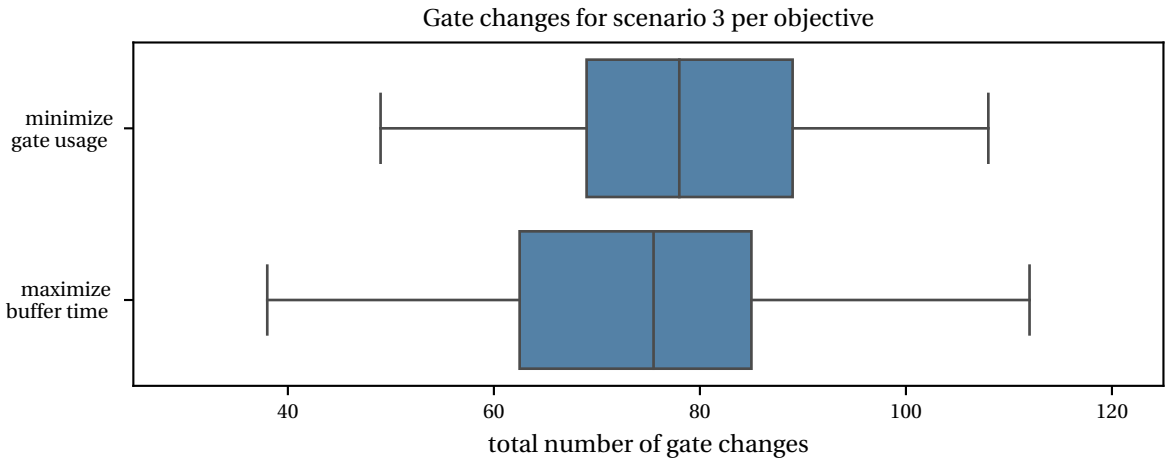

Figure 5.9: Box plot indicating the total number of gate changes per objective

The total number of gate changes only provides a partial indication about the robustness of the flight to gate assignment. In order to precisely determine which initial assignment is more robust, it is required to study at which point in time the gate change for a certain flight occurred. The closer this time is to the arrival time of a flight, the less desired a gate change is. In page 49, a box plot is used to visualize the aggregated result of all the runs performed. The median of the box plot shows that the median for the *maxbuf*-objective is lower than that for the *mingate*-objective, this means that the the *maxbuf*-objective is resulting in a more robust initial flight to gate assignment.

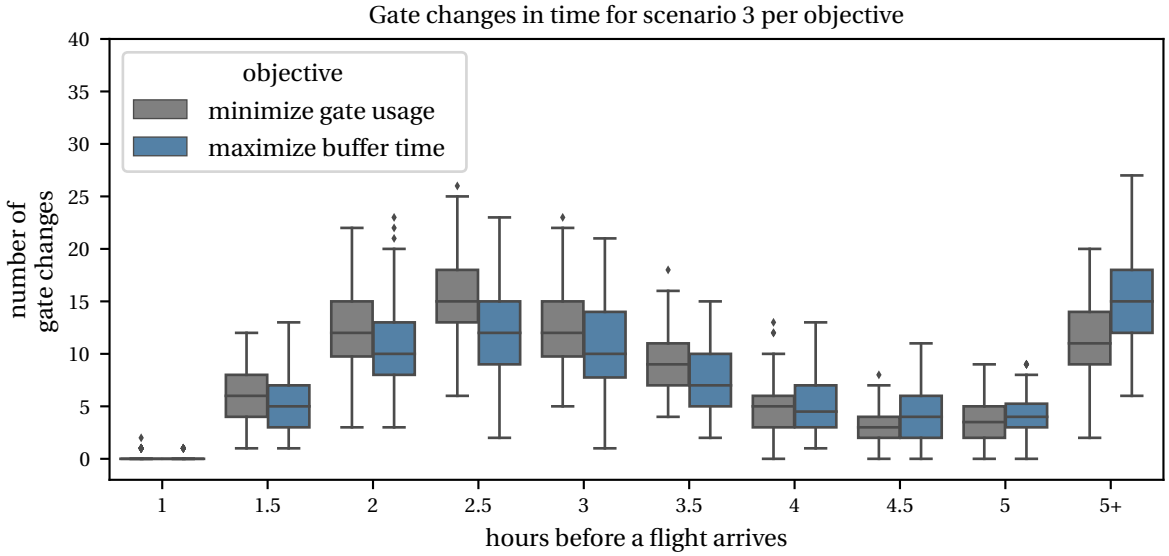

Figure 5.10: Box plot indicating the point in time at which gate changes occurred aggregated in hours before a flight arrives at the airport and grouped by objective

Robustness of the initial flight to gate assignment is beneficial for the passengers but it is more of interest to the airlines and airports as it aims to improve operational efficiency. From a passenger perspective, the number of delayed flights, remote handled flights and the number of delayed minutes are of greater interest as they reduce the comfort of the passengers. In page 50, a box plot is used to visualize the number of delayed flights. The figure shows that the *maxbuf*-objective sees on average less delayed flights per run and thus a higher passenger comfort level. The size of the whiskers and box for the *maxbuf*-objective furthermore indicate that the results from this objective are more stable since the variance in the solution is smaller than the variance seen with the *mingate*-objective.

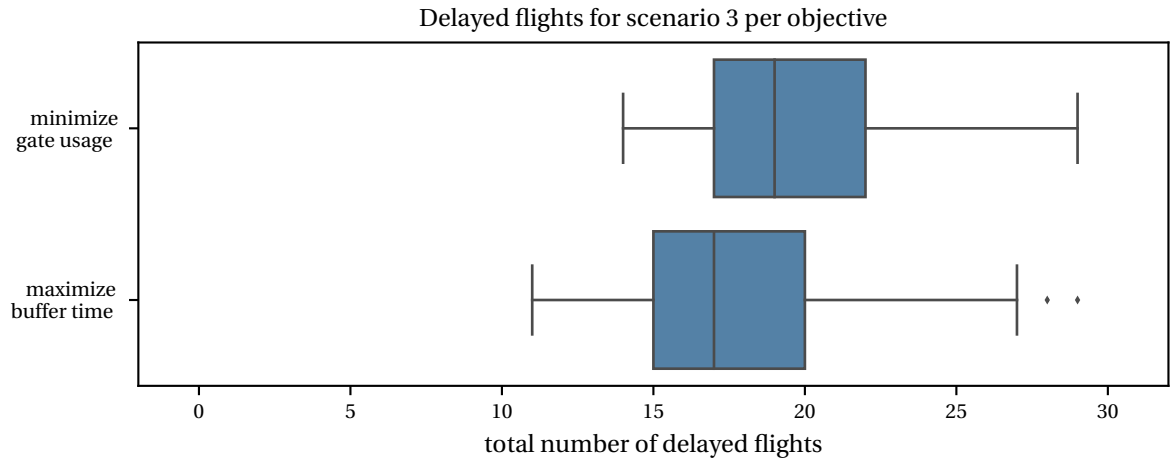

Figure 5.11: Box plot indicating the total number of delayed flights per objective

Apart from the operational efficiency, the reassignment model aims to minimize the inconvenience for the passengers and thus increase the passenger comfort. This is measured in more detail by computing af few KPI's. For all the KPI's a differentiation between narrow- and wide-body aircraft has been made. This to investigate if certain objectives are beneficial for a certain type of aircraft. The first two KPI's are computed by summing the arrival and departure delay for all the flights in a run. The last KPI is computed by multiplying the the delay of each delayed flight with the number of passengers on that flight. The reassignment model described in chapter 4, prioritizes delaying of flights with less passengers. For all the above mentioned KPI's, the mean and standard deviation have been determined over the runs in the simulation. The mean or average indicate the average delay while the standard deviation indicates how robust the schedule is, as it is desirable to have a low mean with a low standard deviation.
In page 50, the above mentioned KPI's are presented, the table shows that, on average, narrow-body flights experience more delay than wide-body aircraft. This is expected since 70% of the flights in the flight schedule are wide-body aircraft. Comparing the standard deviations fro the arrival and departure delay for both objectives shows that the *maxbuf*-objective produces marginally better results regarding flight delays. Comparing the summed passenger delay confirms that the *maxbuf*-objective outperforms the *mingate*-objective in terms of passenger comfort and robustness of the initial flight schedule. This concludes that the *maxbuf*-objective yields on average the highest operation efficiency and passenger comfort for time schedules where the majority of flights are served by wide-body, long-haul aircraft.

Table 5.4: Results for scenario 3, overview of the KPI's used to compare the performance of the reassignment model for both initial flight to gate assignment objectives

Scenario 3 - 30% wide-body, 70% narrow-body					
KPI		*mingate*: Minimize gate usage		*maxbuf*: Maximize buffer time	
		narrow-body	wide-body	narrow-body	wide-body
Arrival delay	mean	67.1	24.4	55.4	26.8
∑ run in (min)	std	25.5	16.0	23.0	15.4
Departure delay	mean	11.5	5.5	10.0	5.9
∑ run in (min)	std	12.2	6.4	10.0	7.4
passenger delay	mean	8779.6	4927.0	7143.4	5437.8
∑ pax × delay	std	3253.9	3343.5	3001.8	3060.6

5.4. Phase 2 - Objective performance comparison

Comparing the results obtained from the three scenarios for both objectives, shows that the *maxbuf*-objective, maximizing buffer time between consecutive assigned flights, outperforms *mingate*, minimizing the gate us-

age, for all the tested scenarios based on the number of gate changes. The box plots visualized in page 44, page 46, show that for the first and second scenario the number of gate changes increase evenly for both objectives. The plot for the third scenario shown in page 49, shows that the median for both objectives is very close which indicate that the number of gate changes and thus robustness similar is for both objectives. The variance for the *maxbuf*-objective, indicated by the whiskers, is larger which means that this objective is less suitable for flight schedules where the majority of flights are wide-body aircraft as the outcome is unpredictable. The absolute number of gate changes increase as the scenarios contain a higher percentage of narrow-body aircraft. However, the relative number of gate changes remain constant for both objectives across all the scenarios as shown in Table 5.5.
The box plots visualized in page 44, page 47 and page 49, show a trend that when the percentage of narrow-body aircraft in the scenario is increased, the number of gate changes between 1.5 and 3.5 hours before a flights arrival increase. A larger percentage of the gate changes are known less far in advance of a flights arrival, this effect is expected since narrow-body aircraft have an on average shorter flight time than wide-body aircraft.
Comparing the delay box plots for the scenarios shown in Figure 5.5, Figure 5.8 and Figure 5.11, show a similar trend as with the box plots indicating the number of gate changes per scenario. The variance of the results increases after the first scenario, the relative number of delayed flights remain stable as shown in Table 5.5.

Table 5.5: Overview of results from all the scenarios

		mean gate changes		mean delayed flights	
	total flights	*mingate*	*maxbuf*	*mingate*	*maxbuf*
scenario 1	116	63 (54%)	53 (46%)	11 (9.5%)	10 (8.6%)
scenario 2	128	65 (51%)	56 (44%)	14 (10.9%)	14 (10.7%)
scenario 3	151	79 (52%)	75 (50%)	20 (13.2%)	18 (11.9%)

The performance of the *mingate*-objective falls short in comparison to that of the *maxbuf*-objective. This can be explained by close examination of the scenarios used together with the delay distributions used. The scenarios displayed in Appendix A, show that the time between subsequent flights is relatively short and that, especially wide-body, flights have quite significant turnaround times making it difficult for the model to schedule the flight-to-gate assignment such that it has a gate without flights assigned. All the scenarios, represent a busy airport operation, where all the gate utilization is high. The *maxbuf*-objective, is still able to maintain at least a 10 minute buffer between most subsequent flights assigned to the same gate. This leads to the conclusion that the scenarios are not completely suitable for testing the *mingate*-objective, as the idea behind this objective is that there are spare gates available in the found flight to gate assignment.
The delay distribution shown in Figure 4.9 and Figure 4.10, show that the majority of flights deviate between +10 and -10 minutes from their scheduled arrival time. The found initial flight to gate assignments for each scenario are shown in Appendix A, examination of the flight to gate assignments confirms that the consecutive assigned flights computed using the *maxbuf*-objective are consistently spaced by, on average, 10 minutes. An examination of the flight to gate assignments computed using *mingate*, show that there are some very tight scheduled gates and some gates with very few flights. There are however no empty gates in none of the scenarios. Given that the majority of flights have a slight deviation confirms the conclusion that the proposed scenarios are best suited for the *maxbuf*-objective. It is expected that the *mingate*-objective will perform better in scenarios with less tight flight schedules so that the found flight to gate assignment contains at least a spare gate.

5.5. Conclusion

Concluding the theoretical case leads to the following recommendations for the practical case as discussed in chapter 6.
In the theoretical case, the flight-to-gate assignment framework has been applied to different scenarios where the distributions between narrow- and wide-body aircraft is varied. For each of these scenarios, a random delay model mimicking the delays experienced by flights visiting AAS has been applied a hundred times. The goal of this approach was to create representative scenarios of which the findings can be used in the applica-

tion of the framework in the practical case. Analyzing the results of the theoretical case, it is concluded that the *maxbuf*-objective, which maximizes the buffer time between consecutive assigned flights outperforms Objective1, which minimizes the gate usages.

The optimal size of the sliding time window is dependent on the distribution between narrow- and wide-body aircraft. If the majority of flights is operated by narrow-body aircraft, as per scenario 1, then a narrow sliding time window of 15 minutes is optimal. Increasing the sliding time window for narrow-body aircraft limits the options the model has for reassignment as flight time is generally short for narrow-body aircraft (<4 hours). Narrow-body aircraft, in general, spend minimal time at the gate as airlines optimize their schedules to execute as many flights per day as possible. The consequence of these highly utilized aircraft is that the time spend at the gate is just enough to perform the turn-around procedure, de-boarding and on-boarding of the aircraft. Another effect visible of this approach taken by airlines, is that narrow-body aircraft are prone to accumulate delay over a day of operation. This effect has been modeled into the delay model used for the theoretical case discussed in this chapter.

For flight schedules where the majority of flights is executed by wide-body aircraft, a long sliding time window is advantageous. A wide sliding time window of 45 minutes increases the number of flights that have their scheduled arrival time updated with the actual arrival time based on the departure time at their origin. Flight time for wide-body aircraft is in general longer (>7 hours) since they serve destination further away, compared to narrow-body aircraft. Even though the reassignment model is run less frequent, the model has still ample options for reschedule due to the long flight time. The reassignment model has all the possible option for reassignment available if the arriving flight is more than 4 hours away. If the size of the sliding time window is decreased, the number of flights that are updated are reduced. The effect of this is that there are relatively more gate reassignments as the subsequent flight at the same gate are not updated simultaneously. As a consequence, there are a lot of gate conflicts and thus gate reassignments to either gates or remote parking locations. In the next run of the reassignment model, as the subsequent flight is updated, often flights are reassigned back to their original gate or from the remote parking locations to a gate. Increasing the size of the sliding time window reduces this ping-pong effect of flights between gates and remote parking positions.

From the theoretical case it is found that the maximize buffer time objective is outperforming the minimize gate usage objective for all scenarios. The optimal size of the sliding time window is dependent on the distribution between narrow- and wide-body aircraft in the flight schedule.

6

Practical case - model validation with AAS data

In this chapter the practical case of the framework is discussed. The practical case spans the application of the model to real world flight schedules obtained from AAS. In the previous section it was concluded that the objective which maximize the buffer time between consecutive flights outperforms the objective that minimizes the number of gates used. Furthermore, did the sliding time window size depend on the distribution between narrow- and wide-body aircraft in the flight schedule. For the practical case different flight schedules from AAS are used to asses the performance of the proposed reassignment framework. The validation of the framework will be executed with and without the second phase of the framework. During the validation gates will be removed or added to the model to investigate the minimal number of gates required to achieve a set performance minimum in terms of gate reassignments and remotely handled aircraft.
The chapter will start with a definition of the flight schedules used as the number of flight differ from day to day. The subsequent section will discuss the modification done to the framework to handle the data obtained from AAS. The chapter will conclude with the results and a conclusion.

6.1. Selecting flight schedules from the AAS data set

The data set obtained from AAS describing the gate changes, spans 181 days of operation. For each individual day there is a set of flights recurring flights; monthly, weekly, daily or a combination. On top of this set of recurring flights, there is set of non-recurring flights that do not operate periodically but are scheduled as extra flights, either seasonal or as temporary extra capacity to certain destinations.
At AAS there are two border statuses enforced for arriving and departing flights, Schengen and non-Schengen. The border status for a gate depends on the flight assigned to the gate. If a flight is arriving from outside the European Union it needs to arrive at a non-Schengen pier and vice versa. In Table 6.1, an overview of the different piers a AAS and there corresponding border status is given.
Appendix A, concluded that the model performed best on flights with a relatively long flight time as this gave the reassignment framework ample options to resolve gate conflicts. Flights with a long flight time are generally speaking the wide-body aircraft. The majority of flights arriving at AAS are narrow-body aircraft predominantly arriving from the Schengen area. For the validation of the framework it is therefore decided to validate the working principles of the framework on a subset of piers from AAS, serving the non-Schengen area.

Table 6.1: Different piers at AAS and their border status

	Pier						
	B	C	D	E	F	G	H/M
Schengen	Yes	Yes	Yes	No	No	No	Yes
non-Schengen	No	No	Yes	Yes	Yes	Yes	Yes
Swing gate	No	No	Yes	No	No	No	Yes

Table 6.1, shows that pier; D, E, F, G and H/M can handle the non-Schengen flights. However, pier D and H/M are so called swing gates, this means that they can handle flight arriving from both inside and outside the Schengen area. H and M piers are used for low cost airlines and thus handle mainly narrow-body aircraft, they are excluded from the subset. The D pier handles a mixed set of flights originating and departing to/from destinations inside and outside the Schengen area. Being an older pier, and with the increase in aircraft size, the D-pier was modified according to the MARS concept[1]. Due the implementation of the MARS concept at the D-pier, it uses a lot of complicated gate constraints and as a consequence it is handling a significant number of narrow-body aircraft. For the validation it is therefore decided to focus on the E, F and G pier of AAS. Filtering out the flights initially assigned to these piers, results in 21,772 flights of which 27% is narrow-body and 73% is wide-body.

The E-pier has 14 gates and 3 remote parking positions, the F pier has 7 gates and no remote parking positions, the G pier has 8 gates and 3 remote parking positions. Even though pier F and G seemingly have there own dedicated remote parking positions, this is not how these are used within AAS operation. Remote parking positions are always served by bus which transport the passengers back and forth between the aircraft and the terminal building. Remote parking positions are therefore not related to the associated pier border status. AAS has used the same letter to indicate the nearest pier building to a set of remote parking positions. An overview of the gate plan for AAS is shown in Figure 6.1

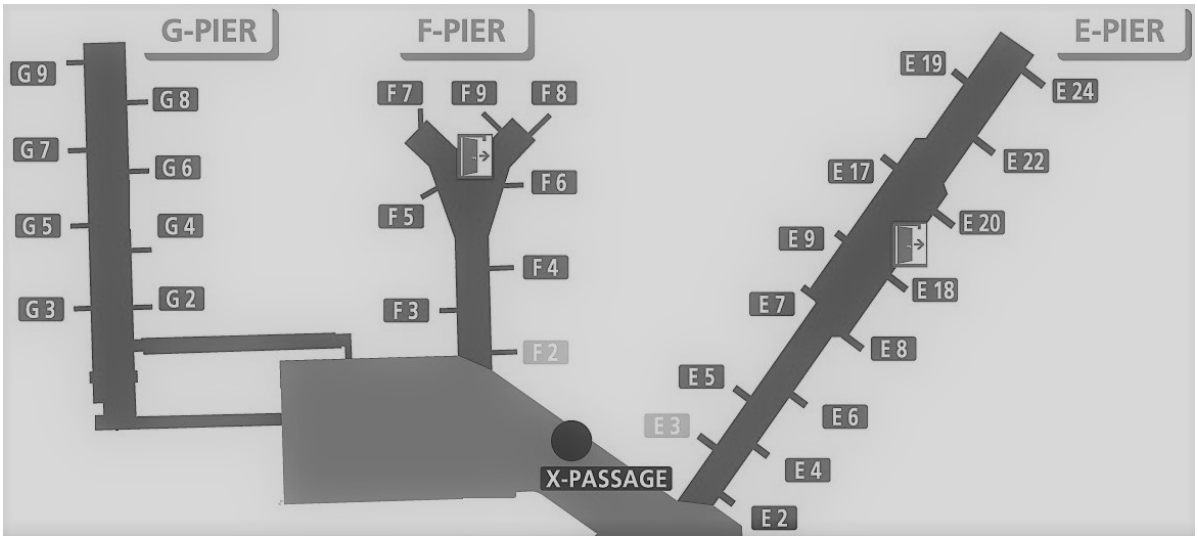

Figure 6.1: Overview of the gate plan for the E, F and G piers found at AAS

Analysis of the 181 days of operation is performed in order to create a subset of days that provide a representative challenge for the model. Figure 6.2, shows the average number of weekly flights scheduled for the E, F and G pier, together with the average number of weekly gate changes. For each day, the flights that are scheduled for the E, F or G pier at the start of the day are stored. During the day, AAS re-assigns flights and gates if gate conflicts occur. Only the gate changes that have been imposed on the flights initially scheduled to the E, F or G pier are counted. This is done to isolate the the subset of flights from the rest of the AAS operation.

[1] MARS, Multiple Apron Ramp System. Is used to exactly identify the space available at each gate and the space each aircraft type requires. If a gate can for example fit a large aircraft it can also be used to handle two small aircraft simultaniously. The MARS concept also applies to neighbouring gates, a small aircarft at one gate, leaves extra room for a larger than normal jet at the neighbouring gate.

During the validation process a number of gates can be removed or added from/to the framework to access the performance and capacity of the current operation.

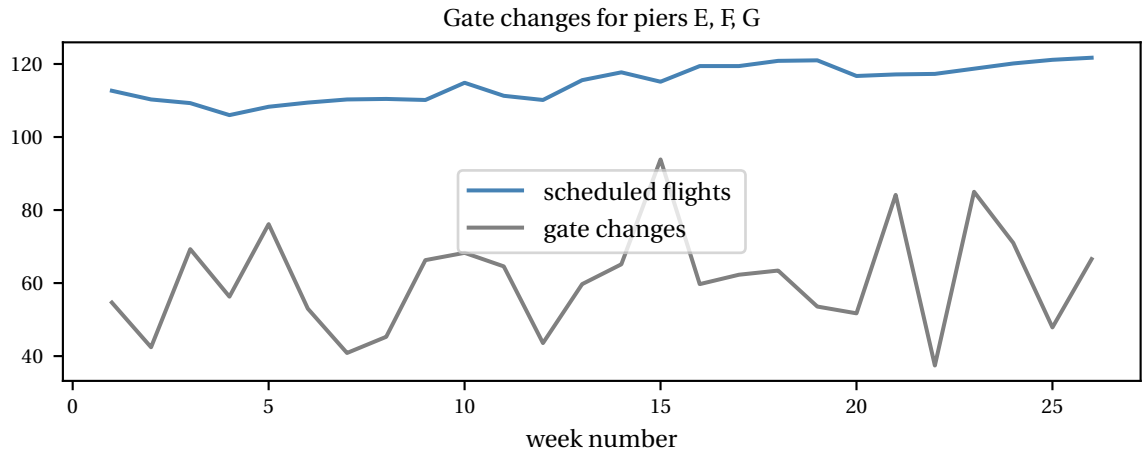

Figure 6.2: Average number of flights assigned to pier E, F and G at AAS per week. For all of the assigned flights, the number of gate changes are visualized in grey

The figure shows that as summer approaches, the number of scheduled flights slightly increases. The average number of weekly gate changes stays stable. However, towards the summer months there are more peaks, indicating that flights experienced more delays, resulting in gate changes or that there were more flights in certain time slots causing congestion at the airport.
On average, the E, F and G pier handle 120 daily flights and process 64 gate changes, some flights receive multiple gate changes while other flights receive no gate changes, on average about 53% of the flights have to be rescheduled. For the validation framework it is of interest to select days with a larger percentage of gate changes than the average. This to study the effect of the proposed model has on the daily operation at AAS.
Table 6.2, lists the busiest days in terms of relative gate changes, the gate changes are given as a percentage of the number of scheduled flights for that day for the E, F and G pier. E.g. if there are 125 flights scheduled for a day, and during the day there have been 50 gate changes to these initial 125 flights than it is said that on average 40% of the scheduled flights gate changed. In this example it is possible that flights receive multiple gate changes, so in practice less than 50 unique flights received a gate change.

Table 6.2: Top 5 busiest days at the E, F and G pier in terms of percentage of gate changes and if applicable the circumstance that contributed to large percentage of gate changes

Day	Initially scheduled flights	gate changed	% gate changed	Circumstance
08-06-2019	103	128	124.3 %	strong winds
10-04-2019	113	138	122.1 %	n/a
09-06-2019	119	124	104.2 %	end of holiday weekend
29-04-2019	125	121	96.8 %	power failure
10-03-2019	102	93	91.2 %	strong winds

The table shows that it is very common that on busy days there are external factors at play that cause disruptions in the flight schedule and thus lead to an above average number of gate reassignments.

Both days with strong winds from Table 6.2, provide an interesting case for the model. Strong winds can limit the usage of the runway system and thus reduce the capacity, increasing the time to land. The strong winds increase the workload of the pilots during approach and thus increase the chances of a go around, delaying the flight. As a result, there will be a high chance of correlated delay as flights have to wait for each other.
In addition it is of interest to see how the proposed model is performing in cases of congestion. April 22nd and June 30th, are respectively the busiest and 2nd busiest days in terms of initial assigned flights. With respectively 125 and 120 initially assigned flights to to piers E, F and G. In the data set there are also quiet days,

in respect to the number of initial assigned flights and lowest percentage of flights gate changed, respectively January 9th and January 11th.
Lastly it is chosen to include April 24th into the data set as it has a below average number of gate changes were, on average, 38% of the flights received a gate change. An overview of all the selected days and the relevant statistics for these days is shown in Table 6.3

Table 6.3: Overview of the days from the AAS data set used for the validation process of the framework and the reason for inclusion in the validation set

Day	Initially scheduled flights	Scheduled flights gate changed	% Scheduled flights gate changed	Selection reason
10-03-2019	102	93	91.2%	Strong winds
08-06-2019	103	128	124.3%	Strong winds
22-04-2019	125	79	63.2%	Most assigned flights
30-06-2019	120	75	62.5%	2nd most assigned flights
24-04-2019	116	44	37.9%	below average number of gate changes
09-01-2019	93	21	22.6%	least assigned flights
11-01-2019	98	19	19.4%	lowest % of gate changes

6.2. Adaptions to the framework

The framework as described in chapter 4, has been designed with the intention of using artificial flight schedules. For the validation purposes, the framework needs to be altered so that it works with the external flight schedules obtained from AAS and to ensure that the model satisfies the gate compatibility constraints imposed by AAS.

In the chapter 4 and Figure 4.6, the simulation flow of the framework for the theoretical case is discussed. Figure 6.3, shows the updated simulation as used for the practical case or validation process. The changes in the flow occur in the second phase of the Framework. The initial objectives which optimize the flight to gate assignment for either minimal gates used or maximizing the buffer time between consecutive assigned flights have been removed from the framework. The initial assignment performed by AAS is assumed to be optimal. AAS publishes a document in which all the gate assignment rules are listed, however, certain airlines are allowed to express gate or pier preferences. These preferences and other are not summarized in the gate assignment rules (Prent, 2019). It is therefore assumed that initial assignment used by AAS is sub-optimal and that solving the initial assignment for AAS to an optimum with the proposed objectives would yield infeasible solutions for practical use.

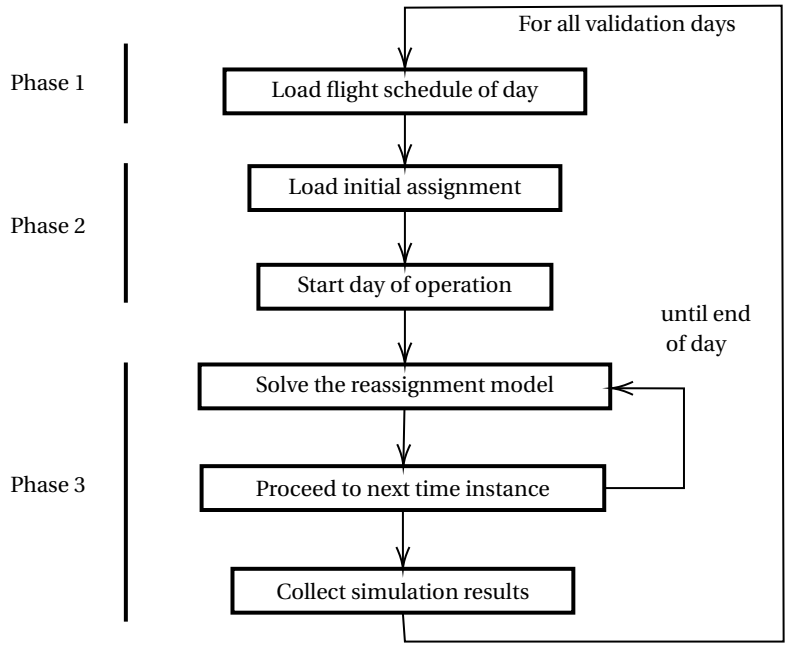

Figure 6.3: Simulation flow diagram for the framework with the three phases

For the third phase of the framework, the actual time of departure from the airport of origin is required. In the theoretical case, this is a random selected number based on the type of aircraft, wide-body or narrow-body. For the practical case, the AAS data set is used and includes the origin and destination for every visiting aircraft. With the coordinates of the origin and of AAS, the distance is computed using the "great-circle distance"-method. (Dobruszkes, 2019), showed that this method provides the most optimum flight distance, but that it is seldom the actual distance flown. More often, flights are flying greater distances due to geopolitical situations, technical restrictions or natural barriers. Using the great circle distance, requires a small correction for these errors as discussed in the next paragraph. Equation 6.1, shows the great circle formulation (Haversine) used to compute the distance between two coordinates.

$$d = 2r \arcsin\left(\sqrt{\sin^2\left(\frac{\varphi_2-\varphi_1}{2}\right) + \cos(\varphi_1)\cos(\varphi_2)\sin^2\left(\frac{\lambda_2-\lambda_1}{2}\right)}\right) \tag{6.1}$$

where:

φ_1, φ_2 latitude of point 1 and point 2 (rad)
λ_1, λ_2 longitude of point 1 and point 2 (rad)

There are more accurate formulas to compute the the distance between two earth coordinates such as the Vincenty formula. Such accuracy is not required for the purpose in this research. The computed flight distances are converted to estimated flight times by dividing it with the average cruising speed of an aircraft, $850 km/h$. A small correction is performed to compensate for the fact that aircraft are not actually flying the great circle distance. In addition a correction is applied for the landing and departure phase of a flight, during these phases the aircraft travels at a lower speed. The total flight time for each route is compensated by adding 20 minutes on top of the computed flight duration using the great circle distance. The addition is most critical for the shortest flights in data set, London to Amsterdam, the flight time for this route is roughly an hour. This route has been used to determine the correction that needed to be applied to the computed great circle distances.

6.3. Validation methodology

This section describes how the validation process was performed and which assumptions have been made to perform the validation. The data provided by AAS contains all the gate changes that have been performed

by the currently used flight to gate assignment framework. The data provides no explanation why a certain gate change has been performed. Following from chapter 5, gate changes can occur for a variety of different reasons. The data provided and the scope of the validation do not require to exactly know the circumstance behind a gate change. The main focus of the validation, is to study the effectiveness of the reassignment framework proposed in subsection 4.4.3 and how this framework could improve the operation efficiency of the airport. The novelty of this framework is that it uses real time data to determine gate conflicts ahead of time and mitigate potential gate conflicts before a flight arrives.

The reassignment framework has been applied to the days listed in Table 6.3. For each of these days, the initial flight to gate assignment has been used as a starting point for the reassignment model. Then the day of operation is simulated, where at each interval of the sliding time window, the scheduled arrival time of each flight is updated by using its actual arrival time as observed at AAS. At each interval, the reassignment model is run to determine the reassignments needed to mitigate potential gate conflicts. At the end of the day, the total number of reassignments performed by the reassignment model are compared to the actual number of reassignments performed by AAS. Only the flights initially assigned by AAS to the E, F and G pier are used in the validation. The number of gate reassignments performed by AAS are therefore computed only for these flights. For example, if AAS reassigns a flight from gate E10 to D5, then from D5 to D6 it is counted as 2 gate reassignments. The total number of gate changes using this approach are shown in Table 6.3 for each day in the validation set.

Using this approach, the reassignment model is run as if it is an ideal world where the arrival time of flight is updated only once and where there are no unforeseen congestion at the airport. Therefore there are multiple runs performed for each day in the validation set, for each subsequent run, the gate that has the least number of initial assigned flights is removed from the model. The goal of this approach is to gather insight in the capacity that is needed to handle the flights of a given day.

A number of assumptions and simplifications have been discussed in this section for the validation which are listed below:

1. Only the flights initially assigned to the E, F and G pier are used for the validation
2. All the gate changes for these flights are counted, also if this means that they are no longer assigned to the E, F and G pier
3. The great circle distance computation, together with an average cruising speed of 850km/h and a correction of 20 minutes is used to determine the flight time left for each aircraft visiting AAS
4. The initial flight to gate assignment determined by AAS is used as the starting point for the proposed new flight to gate assignment model
5. All the gate changes in the AAS data are considered to be a required gate change to resolve a gate conflict

6.4. Results

In total, 7 days from the data set obtained from AAS have been selected to be used for the validation of the proposed real time flight to gate reassignment model. For each of the days, three runs have been performed and thus up to three gates have been removed from the model. For each run, the number of gate changes and remote handled flights have been used as a KPI to determine the operation efficiency and to indicate the amount of capacity needed to handle all the to the E, F and G pier assigned flights of that day. In Figure 6.4, the results for the runs are shown. Run 0 indicates the first run in which none of the gates are removed, starting with run 1, a gate is removed for each subsequent run, for run 3, three gates have been removed from the model. The figure also indicates the number of gate changes Schiphol processed during the day as a reference number. The figure shows that during chaotic days such as 08-06-2019, there are a lot of gate changes processed by Schiphol while the reassignment model has far less gate changes. The reason for this discrepancy is that the model is using data that is updated only once, this while on such chaotic days, some flight can receive upwards to 6 gate changes. On 08-06-2019 AAS was experiencing a lot of delays due to heavy winds in the western part of Europe, as a result a lot of flights missed their arrival and/or departure window leading to congestion and consequently a lot of gate changes. The absolute number of visiting aircraft is low and thus the reassignment model performs good with a low number of gate changes. Selecting a busy

day with a lot of initial assigned aircraft such as 22-04-2019, it is seen in Figure 6.4, that the number of gate changes per run do not vary drastically as the number of gates is decreased.

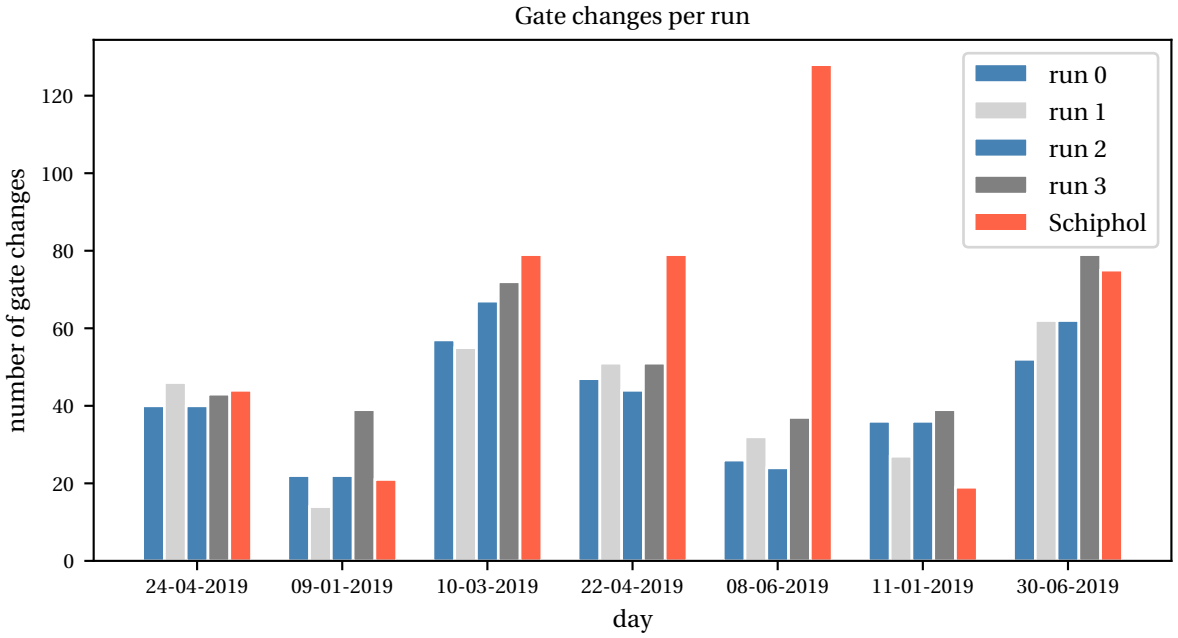

Figure 6.4: Gate changes per day of the validation data using the reassignment model, each run a gate is removed from the reassignment model

This indicates that the flight schedule for that day is not very tight as there is room for more flights at the gates. However, Table 6.4 shows that the delay for narrow- and wide-body passengers significantly increases as the number of gates decreases. The number of remote assigned aircraft is not increasing, which means that, while there are a lot flights, they are not arriving all at the same time but are spread out evenly over the entire day. The reassignment model only resorts to assigning a flight remotely if the gate conflict cannot be solved by slightly delaying one of the conflicting aircraft.

A close examination of the results shown in Table 6.4, indicate that for all the days, the number of remote handled flights remain constant over all the the runs. This implies that if a gate is removed, the flights assigned to that gate can be handled by other gates and do not have to be handled remotely. The last two columns of the table, indicating the passenger delay for narrow- and wide-body aircraft show that the removal of gates is not without a cost. The more gates that are removed from the model, the higher the summed number of passenger delay minutes is for that day. This indicates that the initial schedule for the start of the day from AAS has a significant buffer time between consecutive flights as the flights from removed gates can be handled on other gates with minimal induced delay for subsequent flights. Combined with the number of gate changes leads to the conclusion that some consecutive assigned flights are tightly scheduled while other consecutive assigned flights have ample buffer time between them.

Table 6.4: Result from all the runs of the validation data set

Day	Run	Total gate changes	Remote handled flights	Delayed flights	∑Narrow-body pax delay (min)	∑Wide-body pax delay (min)
24-04-2019	0	40	35	12	1045	15245
	1	46	36	9	1045	15150
	2	40	38	9	1045	11420
	3	43	39	10	1045	10975
09-01-2019	0	22	0	0	0	0
	1	14	0	0	0	0
	2	22	0	0	0	0
	3	39	0	2	0	710
10-03-2019	0	57	10	1	200	0
	1	55	10	4	200	2685
	2	67	10	2	200	4275
	3	72	10	4	200	6040
22-04-2019	0	47	3	3	0	1065
	1	51	3	4	0	1665
	2	44	3	4	320	1520
	3	51	4	4	0	2015
08-06-2019	0	26	33	0	0	0
	1	32	33	1	0	3040
	2	24	35	0	0	0
	3	37	35	2	0	705
11-01-2019	0	36	0	1	1610	0
	1	27	0	0	0	0
	2	36	0	2	640	460
	3	39	0	3	2765	305
03-06-2019	0	52	0	4	1225	700
	1	62	0	5	1600	1065
	2	62	0	8	405	5375
	3	79	0	7	495	4250

6.5. Conclusion

Analysis of the obtained validation results indicate that the proposed flight to gate reassignment model implementing the sliding time window yields better performance in terms of gate reassignments compared to the current approach taken by AAS. Currently the reassignment model in use by AAS relies on the aircraft handlers to update the arrival and departure times of the aircraft visiting AAS. As a consequence, some handlers are swift with updating the arrival time of their flights, while others are late, leading numerous last minute gate changes which reduce passenger comfort and decrease the operational efficiency. The proposed gate reassignment framework shows that the operation efficiency of the airport can be increased, by adopting a gate reassignment framework that implements a sliding time window to determine and mitigate gate conflicts ahead of flights arrival based on the expected flight time left for each arriving flight to the airport. The graphs shown in Figure 6.4 combined with the data from Table 6.4, show that AAS is not yet operating at it maximum capacity since gates can be removed from operation without penalties. A more efficient flight to gate reassignment model increases the operational efficiency and passenger comfort.

7

Conclusion and Recommendations

In this research, the AGAP was investigated and solved using a sliding time window for the operation phase of airport operations. The sliding time window is used to determine the required gate reassignments in real time to mitigate gate conflicts that occur during daily operation as a consequence of flight schedule perturbations. This research is the first to integrate real time flight information into a flight to gate reassignment model.

7.1. Conclusion

In order to investigate the effectiveness of this newly proposed addition to solving to the AGAP, the research was split into a theoretical case and a practical case. The theoretical case was used to determine an optimal initial flight to gate assignment which is used as a starting point by the gate reassignment model. During operation, the reassignment model is run at predefined intervals at which it updates the arrival time of flights based on the flight time left for all the flights scheduled to visit the airport. As the flight arrival times have been updated, the reassignment model is run to determine the gate conflicts and to mitigate them ahead of a flights arrival. The theoretical case uses a fictional airport modeled after the terminal layout found at AAS with fictional flight schedules and delays to stress the model with known hard to solve scenarios. The results of the theoretical case are used in the practical case where real world data obtained from AAS is used to study the performance of the proposed gate reassignment model.

The proposed model is divided into three distinct phases, during the first phase, the flight schedules are generated. Three different flight schedules have been generated for three distinct scenarios where each scenario is representing a different airport operation, This to study the ideal airport operation to implement the sliding time window approach into the gate reassignment model. The three scenarios differ on the total number of flights and the distribution of flights between narrow- and wide-body aircraft. Operational data from AAS showed that there is a clear distinction between a visiting narrow- or wide-body aircraft, the first type is, ideally, on the ground for the shortest time possible resulting in a short (30 minute) turnaround time. In addition to the short turnaround time, this type of aircraft also operates on the short haul routes, which sporadically exceed 5 hours of flight time. In comparison to wide-body aircraft, this gives the gate reassignment model less time to react on perturbations in the arrival time of flights. Wide-body aircraft execute far less daily flights due to the longer average flight time. This is beneficial for the gate reassignment model as it can utilize multiple iterations of the sliding to determine the gate reassignments without penalties. For each gate reassignment, there is a penalty involved depending on the number of hours before a flights arrives. The further the aircraft is away from the airport, the more gates the gate reassignment model has available to assign that flight to. The scenarios have been defined 70%-30%, 50%-50% and 30%-70% for respectively wide- and narrow-body aircraft.

The second phase of the framework, spans the initial flight to gate assignment which is communicated by the airport authorities at the start of the day to all the relevant stakeholders. The main goal for this assignment is to be robust and thus able to absorb flight schedule perturbations instead of relying on gate reassignments

to mitigate occurring gate conflicts. There is an abundance of research performed concerning the creation of robust initial flight to gate assignments. For this research a popular, known, robust objective has been selected for the initial flight to gate assignment model. This objective aims to increase the buffer time between consecutive assigned flights such that there is at least 15 minutes between each flight assigned to the same gate. The idea behind this approach is that, if a flight is delayed slightly this does not affect the successive flight at the same gate as there is sufficient time to absorb the delay. In this research a new objective is proposed which aims to assign all the flight to a minimal number of gates possible. The idea is that there will be so called "overflow" gates in the initial flight to gate assignment to which during the operation, in case of gate conflicts, the conflicting flight can be assigned to. Each of the objectives have been tested on the three scenarios using 100 different delay scenarios per run. It was found that while the scenarios are a good representations for busy airports with a lot of flights, they are too busy for the proposed objective which minimizes for gate usage as the flights are too tightly scheduled in the flight schedule to create an unassigned overflow gate. The objective that maximizes the buffer time between subsequent assigned flights is able to realize a buffer between the majority of flights.

The third phase of the framework, spans the flight to gate reassignment model with the implemented, configurable sliding time window interval. Analysis performed on the theoretical case showed that for operations where the majority of flights is operated by wide-body aircraft, a wider 45 minutes time window yields the lowest number of gate reassignments. For operations where the majority of flights is operated with narrow-body aircraft, a narrower time window of 15 minutes yielded the lowest number of gate reassignments. This behavior is explained by considering that the average flight time for narrow-body aircraft is much shorter compared to wide-body aircraft. The gate reassignment model is increasingly limited in its options for gate reassignment as the arriving aircraft is closer to the airport. For flights that are 4 hours or more away from the airport, all the compatible gate changes are allowed, for flights up to an hour away no more gate changes are allowed. The risk of running the reassignment model too frequent, is that the model updated only a few flights, possibly leading to a lot of gate conflicts, this leads to many gate reassignments. At the next iteration, subsequent flights of the previously updated flights are updated which can result in flights being moved back to their originally assigned gate since the gate conflict is no longer present. Analysis performed on the results from the theoretical case showed that the gate reassignment strategy with the sliding time window achieves the best results in terms of relative number of gate changes on operations where the majority of flights are operated by wide-body aircraft.

The conclusions from the theoretical case have been used to create a validation data set out of the data obtained from AAS. The data from AAS spanned 6 months of operation, starting on January 1st, 2019. The data contained all the aircraft visiting the airport during this time span. The airport is split into two border zones, Schengen and non-Schengen for respectively flights within in the European border region and flights to or from the outside of the European border region. An exhaustive data analysis of the AAS data showed that the non-Schengen piers are handling predominantly wide-body aircraft, these piers include the E, F and G pier. From the data set, a wide range of days where selected, each with unique characteristics. Data from each of the days have been used to run the gate reassignment model. Results showed that it is difficult to compare the number of gate reassignment from the reassignment model directly with the number of gate reassignments performed by AAS. Reasons for the gate changes performed by AAS are unknown and the gate reassignment model has ideal data, in the sense that the flights arrival times are only changed once. However, the results show that the airport is currently not operating at is maximum capacity since the removal of gates does not lead to more remotely handled aircraft and only to slightly more delayed flights.

It is concluded that the proposed gate reassignment model with the sliding time window is capable of providing a more efficient solution for the airport to reduce the number of gate changes and remote handled aircraft. The tested real world operational days from the validation data set, show that major disruptions caused by external factors such as wind, will still result in disruptive situations at the airport. During such events, a lot of aircraft need to be reassigned or handled remotely due to congestion at the airport which is increased due to a reduced runway capacity.

The results drawn from this research can be related back to the main research questions and sub-research questions that were introduced in chapter 5: **text**

1. **What is the advantage of using a sliding time window to dynamically update the initial optimal gate assignment solution to create a robust solution?**
   The advantage of using a sliding time for the gate reassignment model is predominantly visible for

piers where the majority of flights handled are operated by wide-body aircraft. The plots shown and discussed in section 6.4, show that as the relative number of narrow-body aircraft is increased in the flight schedule, the number of gate changes closer to the flights arrival time increases. The advantage of the sliding time window is that the gate conflicts are know further in advance which leads to a more robust airport operation as there less last minute gate changes compared to the current practice or more reactive gate reassignment models that act when gate blockage occurs.

(a) *What is the effect of statistically unexpected schedule changes in current robust models?*
Current robust models are mainly focusing on creating an initial robust flight to gate assignment for the start of the day. During the day, the initial flight to gate assignment is said to be robust if it can absorb flight time arrival variances at the gate the flight is assigned to, rather than to need to reassign flights to new gates. In literature, optimizing for buffer time between consecutive assigned flights is often used to compute a robust flight to gate assignments. A major setback of this approach is illustrated by Figure 4.9 and Figure 4.10, which show that the arrival time variance of flights exceeds the commonly used 10 to 15 minute buffer time. This means that the flights individual gates are not able to absorb the delay and flights have to be reassigned to other, available, gates. The proposed gate reassignment framework solves this issue as it is able to mitigate gate conflicts ahead of a flights arrival when gate reassignments are less costly for the airport and less confusing for passengers.

(b) *What is the effect of varying the sliding time window size?*
The size of the sliding time window, indirectly defines the number of flights for which the scheduled arrival time is updated with the actual or estimated flight arrival time based on the time left to fly for that flight. Analysis performed in section 5.2, concluded that the ideal size of the sliding time window is depended on the distribution of narrow- and wide-body aircraft in the flight schedule. If there are predominantly wide-body aircraft, a wide sliding time window of 45 minutes resulted in the least amount of gate reassignments. For flight schedules with predominantly narrow-body aircraft, a narrow window size of 15 minutes yielded the best performance.

2. **How does the sliding time window approach perform in comparison to current robust models considering gate blockage as a metric?**
The main advantage of the sliding time window implementation is that it mitigates gate conflicts ahead of a flights arrival. The penalty of gate reassignments decreases the further away they are initiated from a flights arrival time. Compared to current robust modes, the sliding time window reassignment model is able to eliminate gate blockage.

   (a) *Does the sliding time window approach outperform manual re-assignment methods?*
   Although the research did not consider a direct comparison between manual re-assignment methods, it is expected that the sliding time window approach is able to outperform or at least match the performance of manual gate reassignment methods. An advantage of the proposed sliding time window model is that it allows for tuning of specific variables to change the gate reassignment behavior.

   (b) *Does the sliding time window approach see an increase in performance if the initial optimal flight to gate assignment is robust?*
   The initial flight to gate assignment is said to be robust if it is able to absorb flight schedule perturbations and that deviations in the arrival time of a flight does not lead to gate conflicts. The parameters of the gate reassignment model have not been altered between the different initial objectives from the theoretical case, so the results are directly comparable. The results show, that the objective which maximizes the buffer time between consecutive assigned flights leads to less gate reassignments during simulated operations. It is concluded that the performance of the sliding time window is dependent on the optimal initial flight to gate assignment and that different initial objectives can increase the performance of the sliding time window model.

3. **Which solution method, heuristic or optimal, provides the best run time performance for solving the re-assignment model?**
The run time for all the models in the framework was sufficiently fast for their intended applications. Run time is most critical for the reassignment model as it application requires frequent solutions for the AGAP. Depending on the time of day the run time varies, near the end of the day the run time is neg-

ligible since most flights have already left the airport and thus gate changes are not possible anymore. The run time for the reassignment model at the start of the operation was in the range of a couple of minutes which is still adequately quick for practical application since the interval at which the model is run was set at 15 minutes. The reassignment model does not employ a real heuristic to improve the performance. But at at the start of each run, it uses the found flight to gate assignment from the previous run as a starting point, which decreases the run time of the model. If this gate assignment leads to no gate conflicts due to flight arrival times being updated , then the model returns this gate assignment. Each gate change has a cost associated to reduce the number of gate changes to a minimum.

(a) *Which heuristic method is preferred when considering speed and finding a feasible solution?* Given the already low run times for both the initial and the reassignment model, there was no motivation to implement a heuristic in order to improve the run time of the models.

7.2. Recommendations for future work

The current study focuses on implementing a sliding time window to solve the AGAP in real time for the operational phase of an airport. The study first explored the possibilities and implications that arise when a MILP model is required to run within a time constraint. The time constraint is enforced to ensure that a solution is found before an assigned flight experiences gate blockage. At the start of the research, the focus lay on defining a framework which would allow an initial optimal gate solution to be resolved in real time to mitigate gate conflicts. For future research it is recommended to combine the method and results of this study to create a reassignment model for a real world airport. For a real world airport, real time data about flight arrival times can be streamed from a sources such as "*Flightradar24*" and "*Flightaware*". This leads to a more thorough validation as the reassignment model can be run next to the current reassignment model of an airport. The results from both models can be compared directly leading to a more accurate performance improvement metrics for the sliding time window approach.

For the validation case, it is assumed that the initial flight to gate assignment computed by AAS is optimal for practical usage. However, the scenarios discussed for theoretical case showed that the performance of the reassignment model with the sliding time window is dependent on the initial flight to gate assignment. In future work, the focus could shifted to find new objectives for the initial flight to gate assignment that would be beneficial for the real time flight to gate reassignment model. The reassignment model relies on the estimated arrival time for flights, future studies can focus on the effect for the reassignment framework if flights with similar flight time are assigned to the same gate.

The framework discussed in this research only considers the arrival delay of aircraft, in cases where the arrival delay is so severe that the minimal turn around time is not met, the departure of the flight is delayed. However, in real world operations flights can also be delayed in their departure due to for example, late arriving passengers, technical problems or congestion at the airport of destination. These effects are not considered for this research, future research can study the effects and the severity departure delays to determine if they should be taken into account for the gate reassignment model.

Finally it is recommended to further study the feasibility of real world applications for a real time flight to gate assignment with the relevant stakeholders in airport operations.

A

Scenarios

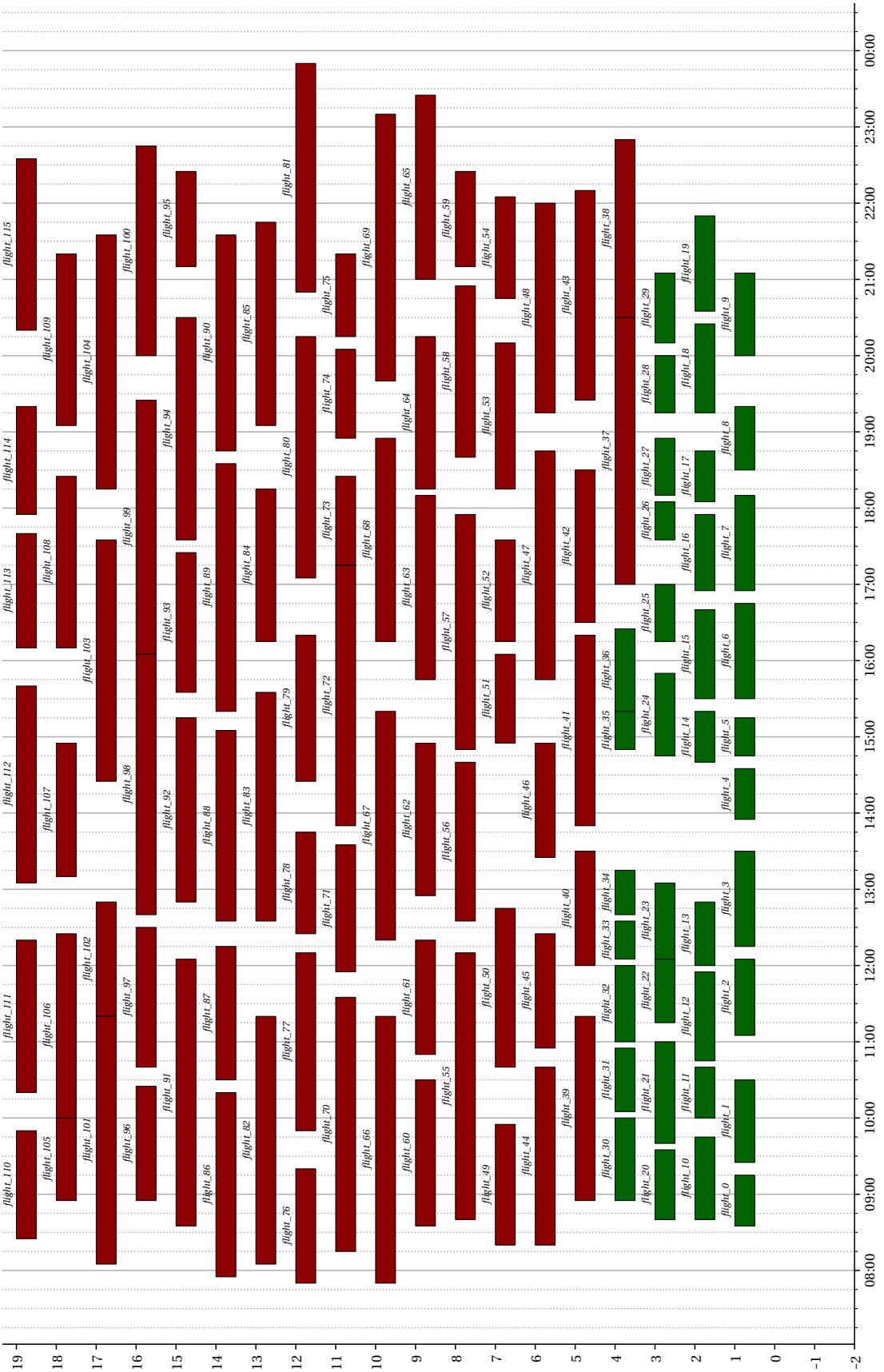

Figure A.1: Flight schedule used for the first scenario, with 70% wide-body and 30% narrow-body flights. Narrow- and wide-body flight are indicated by respectively green and red

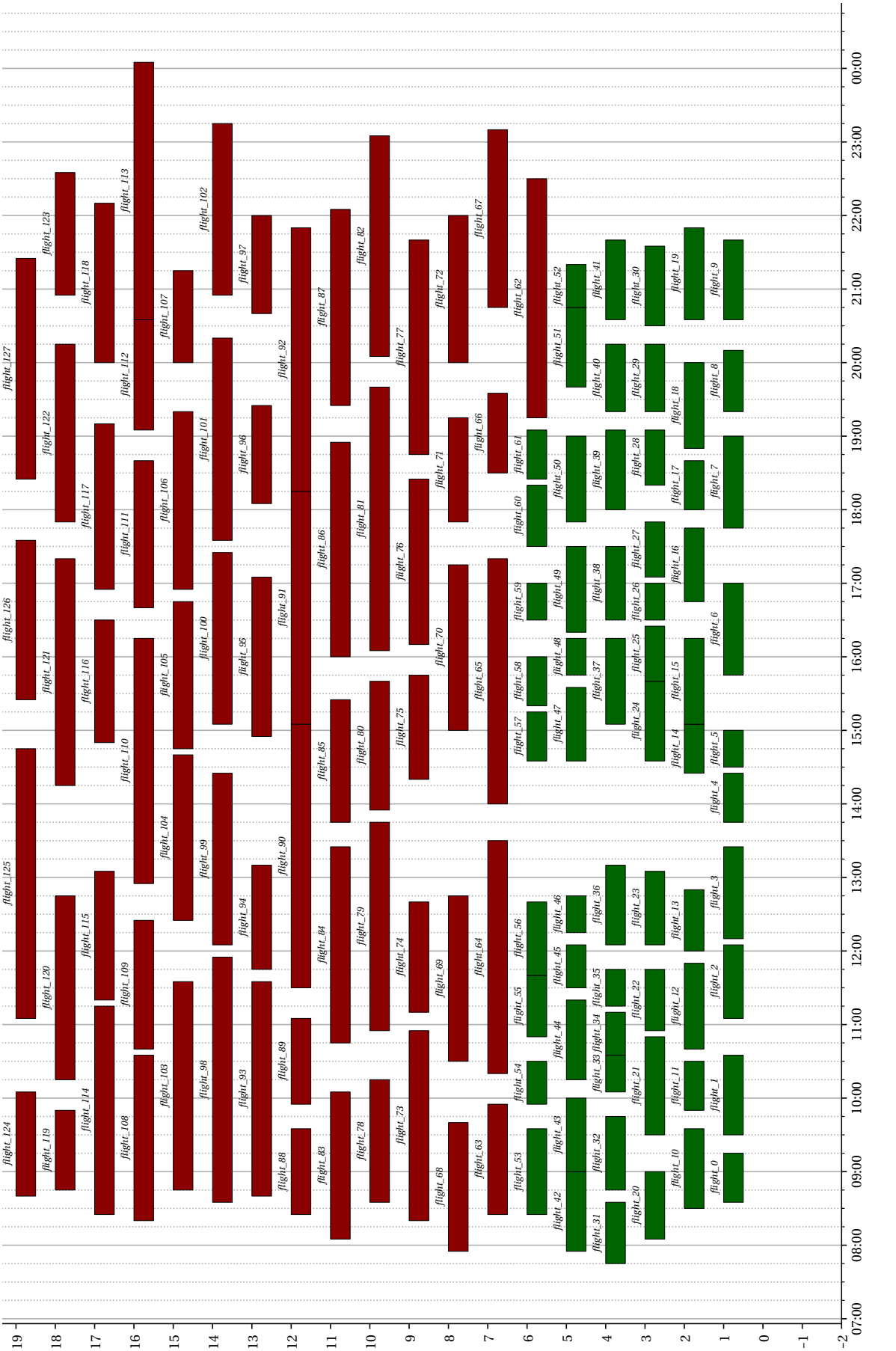

Figure A.2: Flight schedule used for the second scenario, with 50% wide-body and 50% narrow-body flights. Narrow- and wide-body flight are indicated by respectively green and red

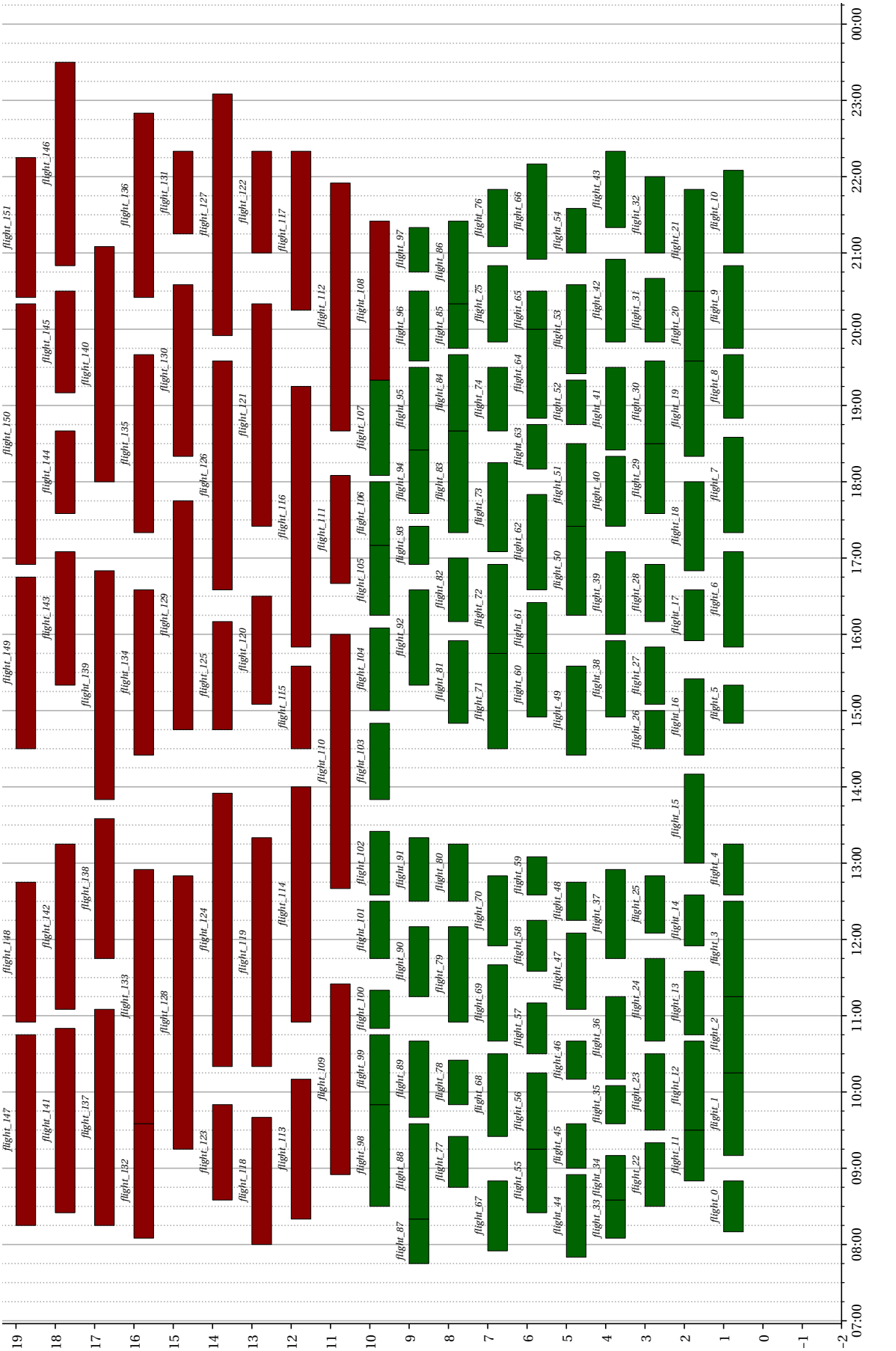

Figure A.3: Flight schedule used for the third scenario, with 30% wide-body and 70% narrow-body flights. Narrow- and wide-body flight are indicated by respectively green and red

B

Narrow-body delay propagation during operation

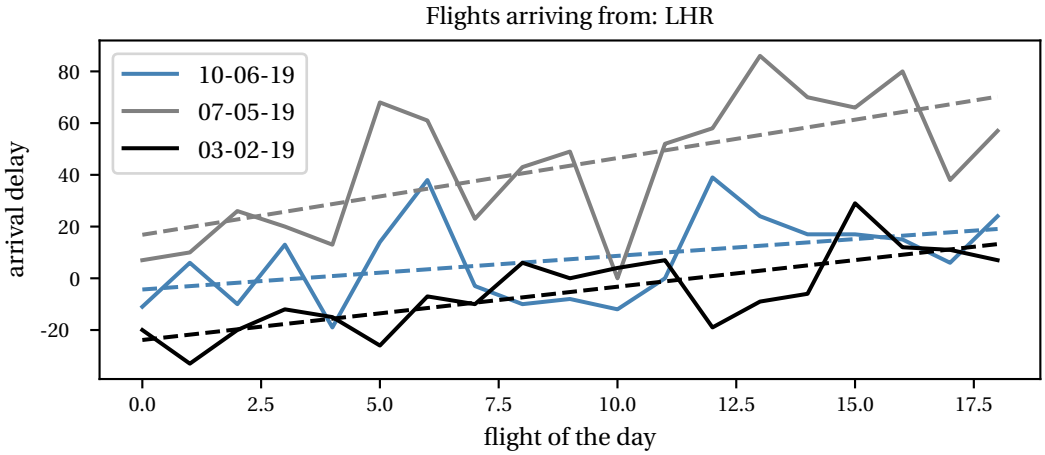

Figure B.1: Delay propagation of flights arriving from LHR

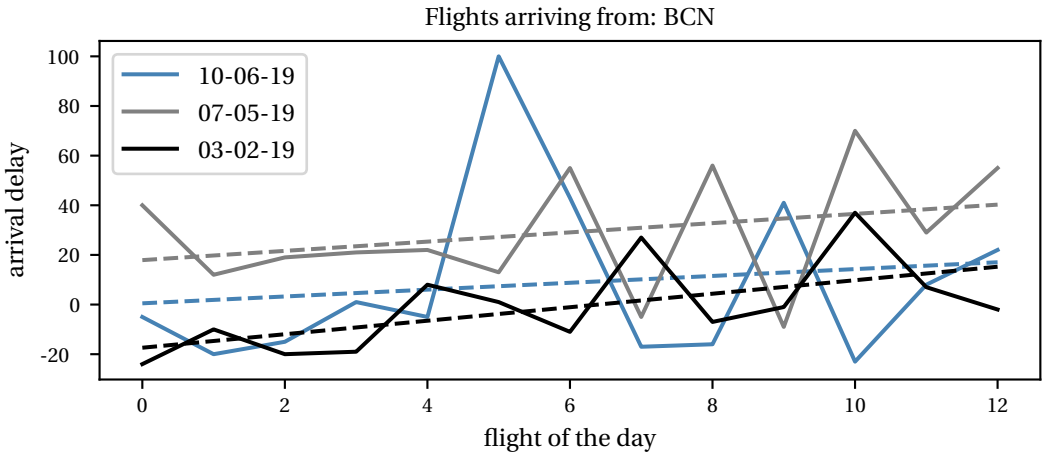

Figure B.2: Delay propagation of flights arriving from BCN

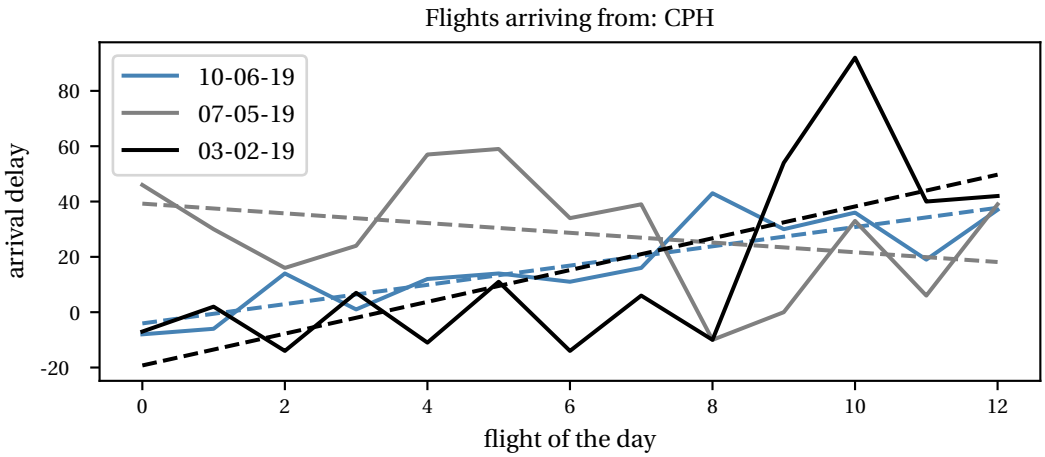

Figure B.3: Delay propagation of flights arriving from CPH

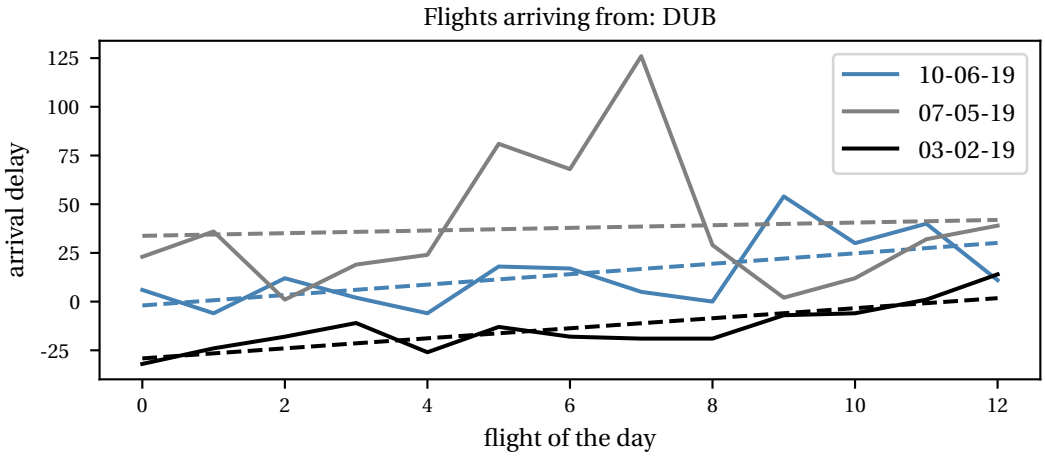

Figure B.4: Delay propagation of flights arriving from DUB

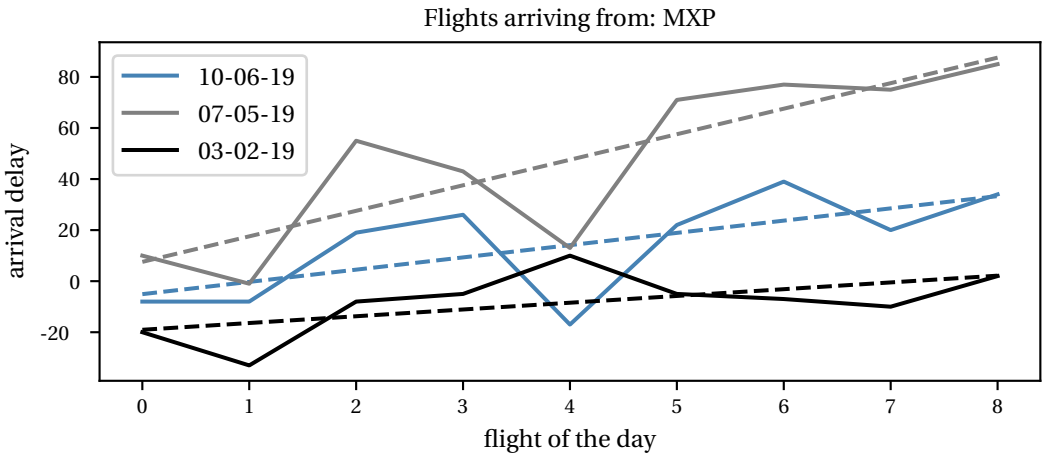

Figure B.5: Delay propagation of flights arriving from MXP

C

Initial flight to gate assignments per objective and scenario

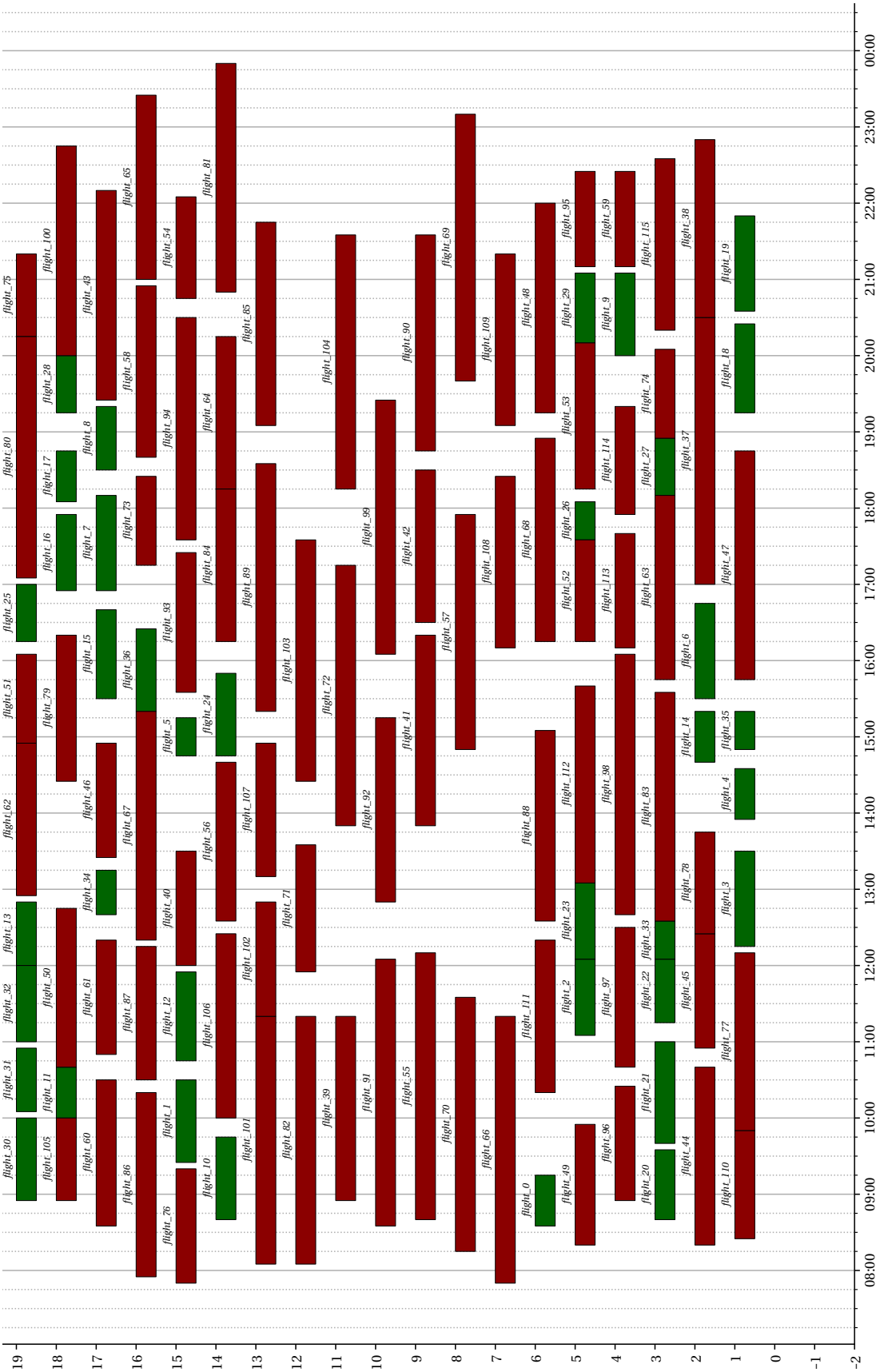

Figure C.1: Initial flight to gate assignment for scenario 1, with 70% wide-body and 30% narrow-body flights, minimizing the number of gates used

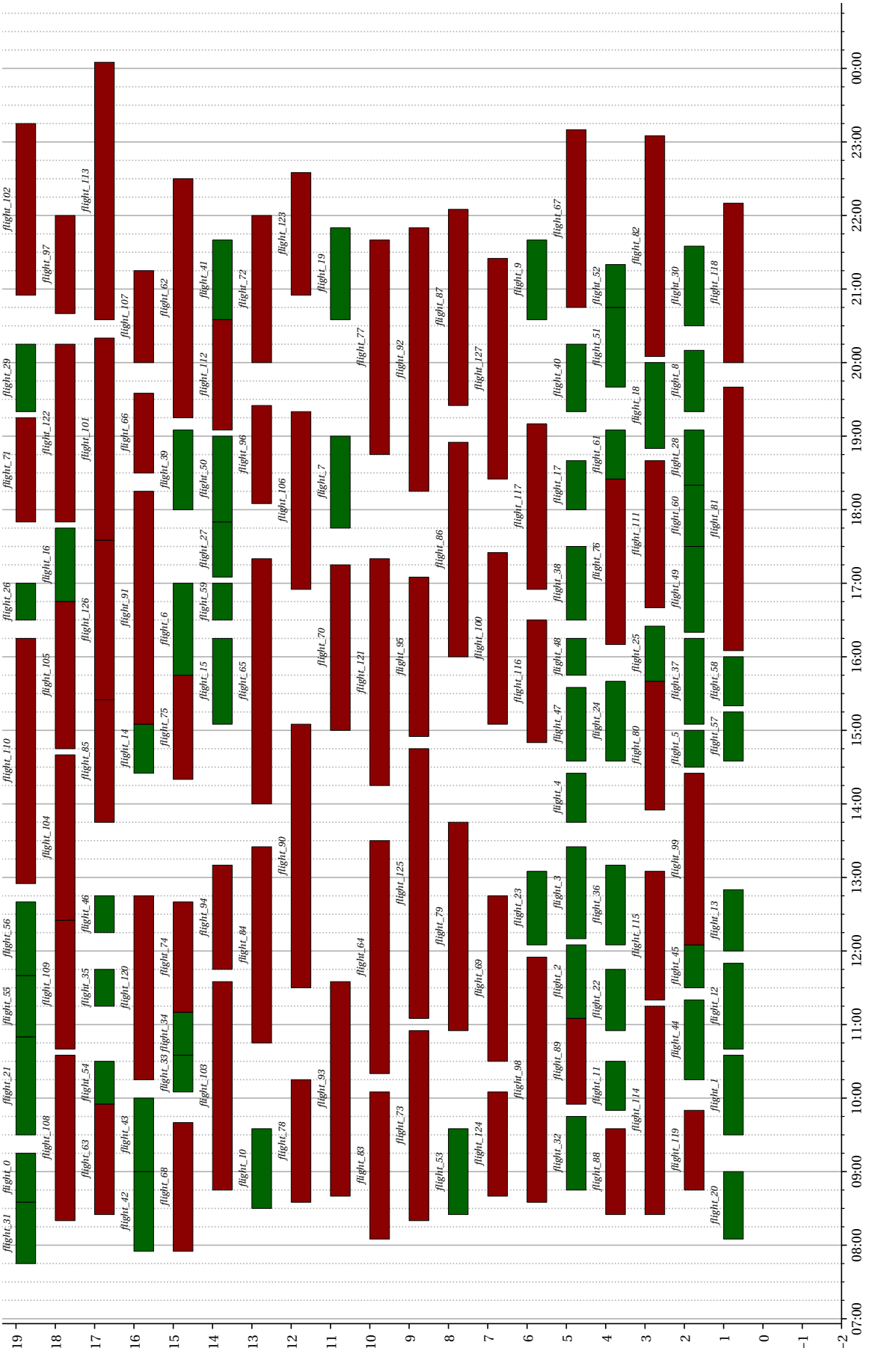

Figure C.2: Initial flight to gate assignment for scenario 2, with 50% wide-body and 50% narrow-body flights, minimizing the number of gates used

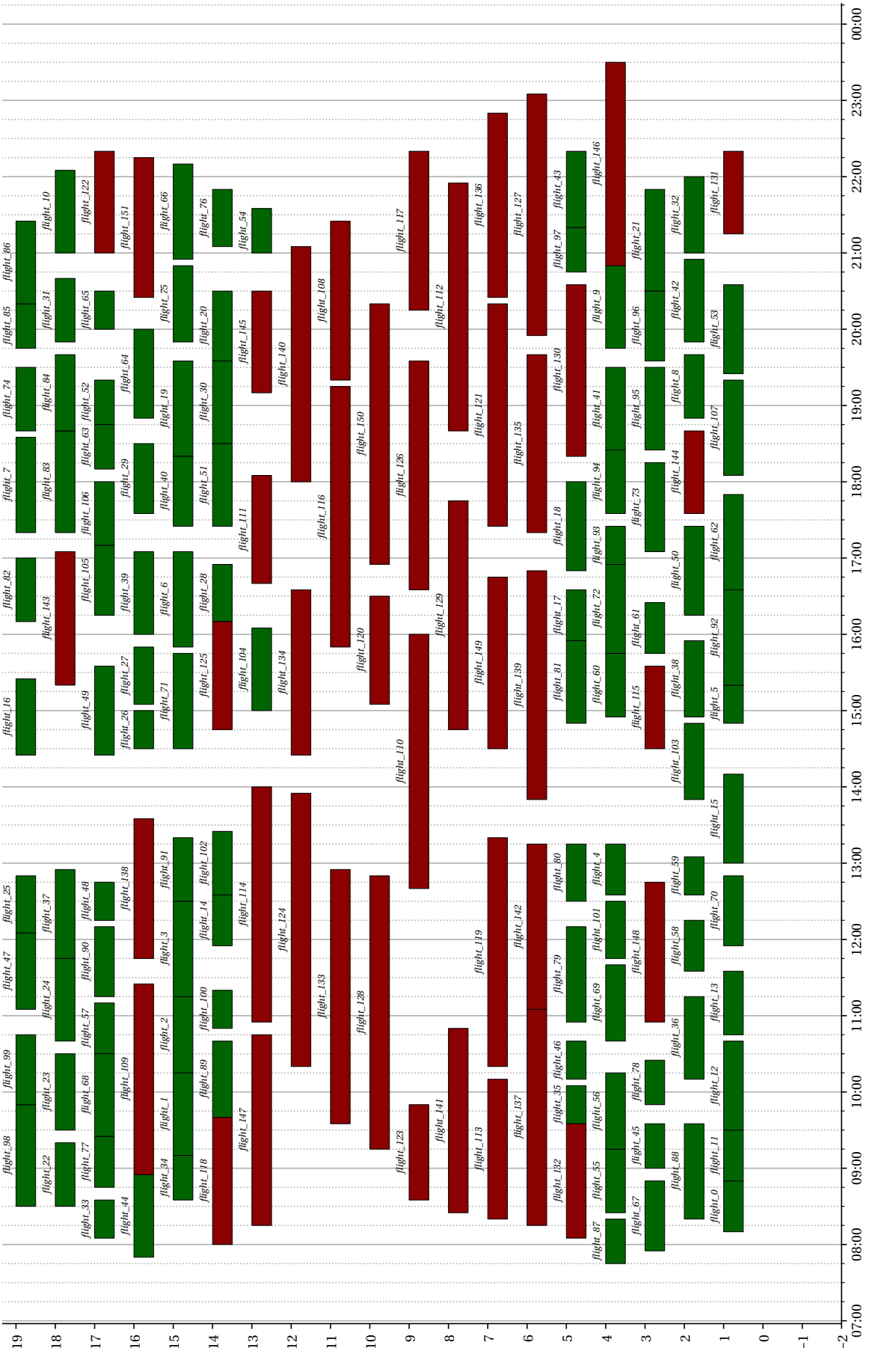

Figure C.3: Initial flight to gate assignment for scenario 3, with 70% wide-body and 30% narrow-body flights, minimizing the number of gates used

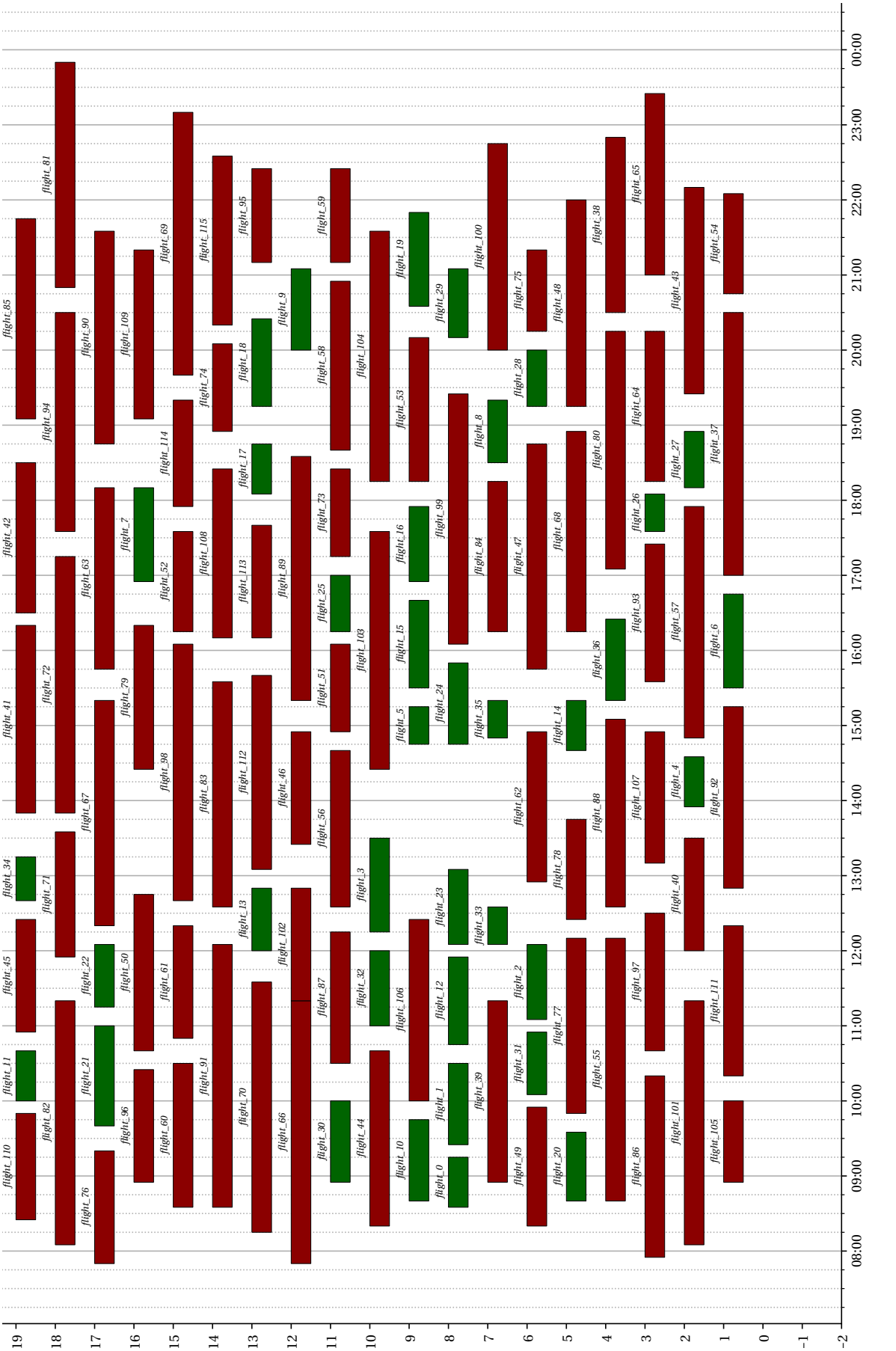

Figure C.4: Initial flight to gate assignment for scenario 1, with 70% wide-body and 30% narrow-body flights, maximizing buffer time between consecutive flights

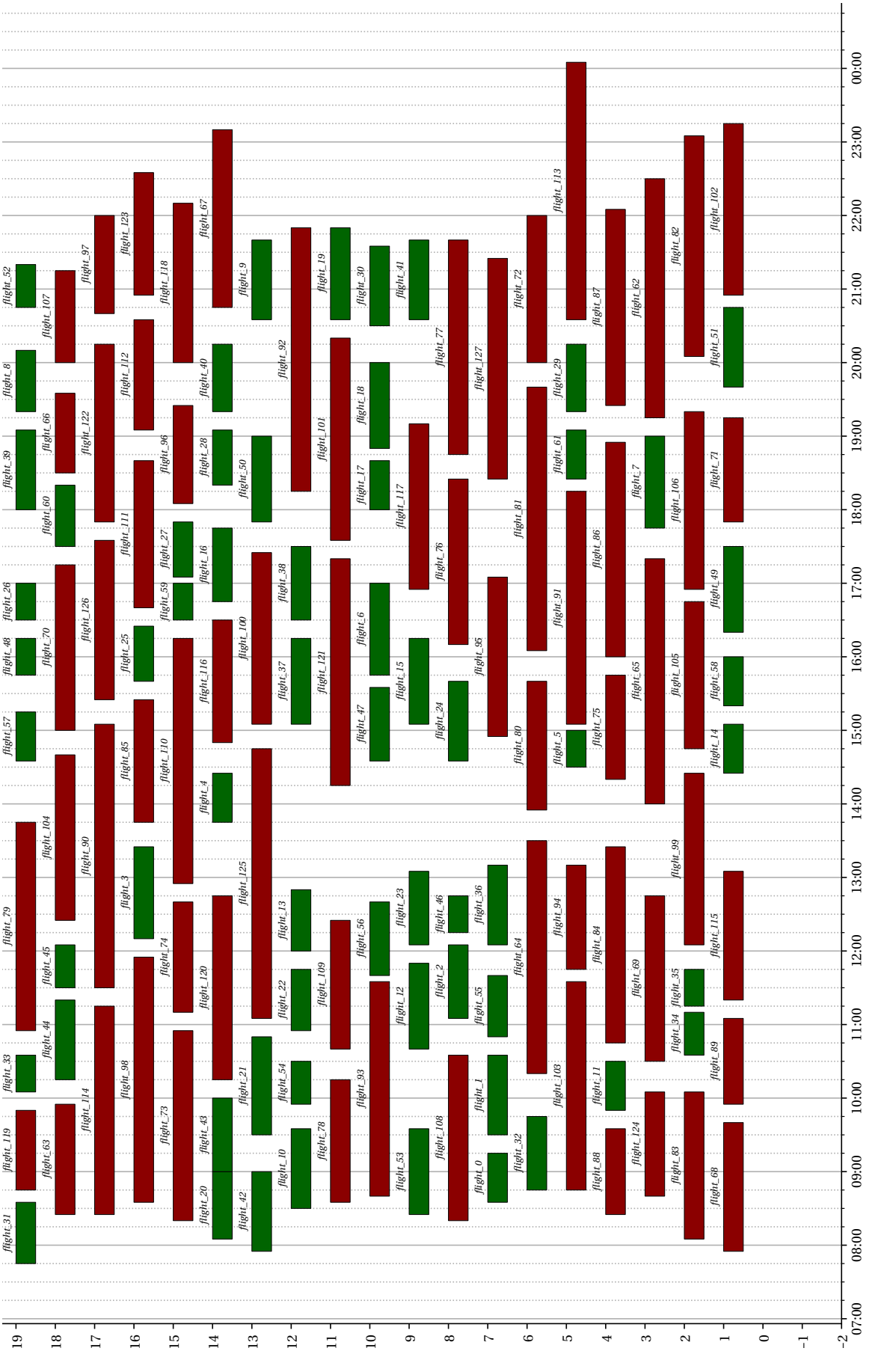

Figure C.5: Initial flight to gate assignment for scenario 2, with 50% wide-body and 50% narrow-body flights, maximizing buffer time between consecutive flights

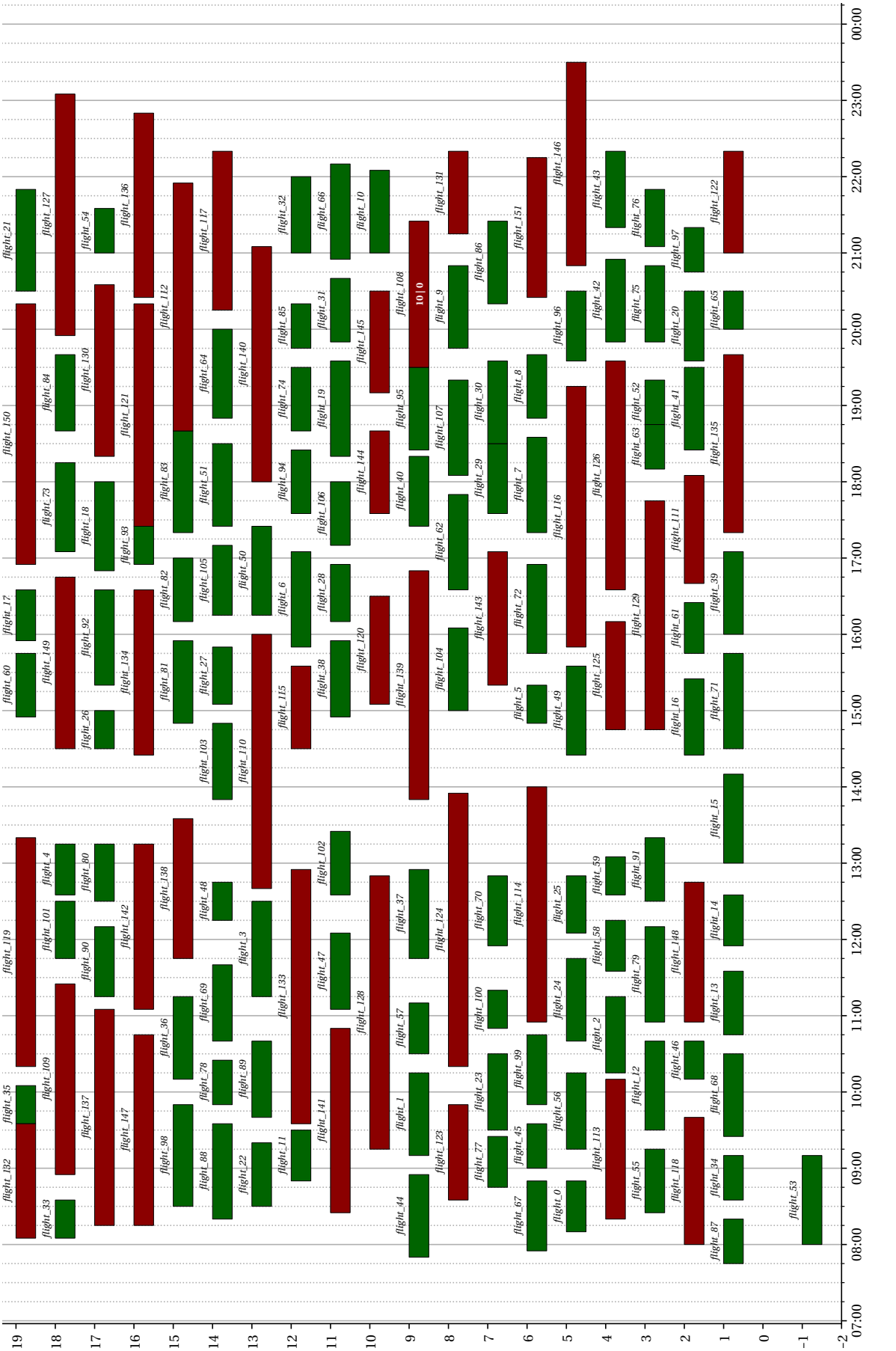

Figure C.6: Initial flight to gate assignment for scenario 3, with 70% wide-body and 30% narrow-body flights, maximizing buffer time between consecutive flights

Bibliography

TransSolutions Airport Cooperative Research Program. *Airport Passenger Conveyance Systems Planning Guidebook*. Transportation Research Board, 2012. ISBN 978-0-309-21412-4. Google-Books-ID: jQRLsXF1scgC.

Babić Obrad, Teodorović Dušan, and Tošić Vojin. Aircraft Stand Assignment to Minimize Walking. *Journal of Transportation Engineering*, 110(1):55–66, January 1984. doi: 10.1061/(ASCE)0733-947X(1984)110:1(55). URL https://ascelibrary.org/doi/10.1061/%28ASCE%290733-947X%281984%29110%3A1%2855%29.

Peter Belobaba, Amedeo Odoni, and Cynthia Barnhart. *The Global Airline Industry*. Wiley, Southern Gate, Chichester, West Sussex, UK, 2. edition, September 2015. ISBN 978-1-118-88117-0.

Richard A. Bihr. A conceptual solution to the aircraft gate assignment problem using 0, 1 linear programming. *Computers & Industrial Engineering*, 19(1):280–284, January 1990. ISSN 0360-8352. doi: 10.1016/0360-8352(90)90122-3. URL http://www.sciencedirect.com/science/article/pii/0360835290901223.

A. Bolat. Assigning arriving flights at an airport to the available gates. *Journal of the Operational Research Society*, 50(1):23–34, January 1999. ISSN 0160-5682. doi: 10.1057/palgrave.jors.2600655. URL https://doi.org/10.1057/palgrave.jors.2600655.

Ahmet Bolat. Procedures for aircraft-gate assignment. *Mathematical & Computational Applications*, 1:9–14, June 1996. doi: 10.3390/mca1010009.

Ahmet Bolat. Procedures for providing robust gate assignments for arriving aircrafts. *European Journal of Operational Research*, 120(1):63–80, January 2000. ISSN 0377-2217. doi: 10.1016/S0377-2217(98)00375-0. URL http://www.sciencedirect.com/science/article/pii/S0377221798003750.

Abdelghani Bouras, Mageed A. Ghaleb, Umar S. Suryahatmaja, and Ahmed M. Salem. The Airport Gate Assignment Problem: A Survey, 2014. URL https://www.hindawi.com/journals/tswj/2014/923859/.

J. P. Braaksma. REDUCING WALKING DISTANCES AT EXISTING AIRPORTS. *AIRPORT FORUM*, 7, August 1977. URL https://trid.trb.org/view/79133.

Dennis Buitendijk. *First order stability, Flexible gate scheduling*. PhD thesis, Delft University of Technology, Delft, August 2014.

CAPA. News for Airlines, Airports and the Aviation Industry | CAPA, January 2019. URL https://centreforaviation.com/news/heathrow-and-gatwick-delays-cost-a-combined-gbp500m-for-passengers-in-

Jeremy Castaing, Ishan Mukherjee, Amy Cohn, Lonny Hurwitz, Ann Nguyen, and Johan J. Müller. Reducing airport gate blockage in passenger aviation: Models and analysis. *Computers & Operations Research*, 65:189–199, January 2016. ISSN 0305-0548. doi: 10.1016/j.cor.2014.02.011. URL http://www.sciencedirect.com/science/article/pii/S030505481400046X.

Yu Cheng. A knowledge-based airport gate assignment system integrated with mathematical programming. *Computers & Industrial Engineering*, 32(4):837–852, September 1997. ISSN 0360-8352. doi: 10.1016/S0360-8352(97)00001-6. URL http://www.sciencedirect.com/science/article/pii/S0360835297000016.

Jacob C. H. Cheung. Flight planning: node-based trajectory prediction and turbulence avoidance. *Meteorological Applications*, 25(1):78–85, 2018. ISSN 1469-8080. doi: 10.1002/met.1671. URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/met.1671.

Tae Won Chung, Yong Joo Lee, and Hyun Mi Jang. A Comparative Analysis of Three Major Transfer Airports in Northeast Asia Focusing on Incheon International Airport Using a Conjoint Analysis. *The Asian Journal of Shipping and Logistics*, 33(4):237–244, December 2017. ISSN 2092-5212. doi: 10.1016/j.ajsl.2017.12.007. URL http://www.sciencedirect.com/science/article/pii/S2092521217300639.

Jean-François Cordeau, Mauro Dell'Amico, Simone Falavigna, and Manuel Iori. A rolling horizon algorithm for auto-carrier transportation. *Transportation Research Part B: Methodological*, 76:68–80, June 2015. ISSN 0191-2615. doi: 10.1016/j.trb.2015.02.009. URL http://www.sciencedirect.com/science/article/pii/S0191261515000375.

Mauro Dell'Orco, Mario Marinelli, and Maria Giovanna Altieri. Solving the gate assignment problem through the Fuzzy Bee Colony Optimization. *Transportation Research Part C: Emerging Technologies*, 80:424–438, July 2017. ISSN 0968-090X. doi: 10.1016/j.trc.2017.03.019. URL http://www.sciencedirect.com/science/article/pii/S0968090X17301043.

Wu Deng, Bo Li, and Huimin Zhao. Study on an Airport Gate Reassignment Method and Its Application. *Symmetry*, 9(11):258, November 2017. ISSN 2073-8994. doi: 10.3390/sym9110258. URL http://www.mdpi.com/2073-8994/9/11/258.

G. Diepen, J. M. van den Akker, J. A. Hoogeveen, and J. W. Smeltink. Finding a robust assignment of flights to gates at Amsterdam Airport Schiphol. *Journal of Scheduling*, 15(6):703–715, December 2012. ISSN 1099-1425. doi: 10.1007/s10951-012-0292-y. URL https://doi.org/10.1007/s10951-012-0292-y.

Bert Dijk, Bruno F. Santos, and Joao P. Pita. The recoverable robust stand allocation problem: a GRU airport case study. *OR Spectrum*, July 2018. ISSN 1436-6304. doi: 10.1007/s00291-018-0525-3. URL https://doi.org/10.1007/s00291-018-0525-3.

H. Ding, A. Lim, B. Rodrigues, and Y. Zhu. Aircraft and gate scheduling optimization at airports. In *37th Annual Hawaii International Conference on System Sciences, 2004. Proceedings of the*, pages 8 pp.–, January 2004. doi: 10.1109/HICSS.2004.1265219.

H. Ding, A. Lim, B. Rodrigues, and Y. Zhu. The over-constrained airport gate assignment problem. *Computers & Operations Research*, 32(7):1867–1880, July 2005. ISSN 0305-0548. doi: 10.1016/j.cor.2003.12.003. URL http://www.sciencedirect.com/science/article/pii/S0305054803003836.

Frédéric Dobruszkes. Why do planes not fly the shortest routes? A review. *Applied Geography*, 109:102033, August 2019. ISSN 0143-6228. doi: 10.1016/j.apgeog.2019.06.001. URL http://www.sciencedirect.com/science/article/pii/S014362281930089X.

Ulrich Dorndorf, Andreas Drexl, Yury Nikulin, and Erwin Pesch. Flight gate scheduling: State-of-the-art and recent developments. *Omega*, 35(3):326–334, June 2007. ISSN 0305-0483. doi: 10.1016/j.omega.2005.07.001. URL http://www.sciencedirect.com/science/article/pii/S0305048305000939.

Ulrich Dorndorf, Florian Jaehn, and Erwin Pesch. Flight gate scheduling with respect to a reference schedule. *Annals of Operations Research*, 194(1):177–187, April 2012. ISSN 1572-9338. doi: 10.1007/s10479-010-0809-8. URL https://doi.org/10.1007/s10479-010-0809-8.

Jacques A. Ferland and Luc Fortin. Vehicles scheduling with sliding time windows. *European Journal of Operational Research*, 38(2):213–226, January 1989. ISSN 03772217. doi: 10.1016/0377-2217(89)90106-9. URL http://linkinghub.elsevier.com/retrieve/pii/0377221789901069.

Hakkı Murat Genç, Osman Kaan Erol, İbrahim Eksin, Mehmet Fatih Berber, and Binnur Onaran Güleryüz. A stochastic neighborhood search approach for airport gate assignment problem. *Expert Systems with Applications*, 39(1):316–327, January 2012. ISSN 0957-4174. doi: 10.1016/j.eswa.2011.07.021. URL http://www.sciencedirect.com/science/article/pii/S0957417411009833.

World Bank Group. Air transport, passengers carried | Data, 2017. URL https://data.worldbank.org/indicator/is.air.psgr?end=2017&start=1970.

J. Guépet, R. Acuna-Agost, O. Briant, and J. P. Gayon. Exact and heuristic approaches to the airport stand allocation problem. *European Journal of Operational Research*, 246(2):597–608, October 2015. ISSN 0377-2217. doi: 10.1016/j.ejor.2015.04.040. URL http://www.sciencedirect.com/science/article/pii/S0377221715003331.

Ali Haghani and Min-Ching Chen. Optimizing gate assignments at airport terminals. *Transportation Research Part A: Policy and Practice*, 32(6):437–454, August 1998. ISSN 0965-8564. doi: 10.1016/S0965-8564(98)00005-6. URL http://www.sciencedirect.com/science/article/pii/S0965856498000056.

J.P. Hamon and F. Weissert. Automatic assignment of aircraft to parking positions: formulation and application to CDG Terminal 2. *Procceedings of the 20th AGIFORS Symposium, New Delhi, India*, September 1980.

A. H. Land and A. G. Doig. An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, 28(3):497–520, 1960. ISSN 0012-9682. doi: 10.2307/1910129. URL https://www.jstor.org/stable/1910129.

A. Lim, B. Rodrigues, and Y. Zhu. Airport Gate Scheduling with Time Windows. *Artificial Intelligence Review*, 24(1):5–31, September 2005. ISSN 1573-7462. doi: 10.1007/s10462-004-7190-4. URL https://doi.org/10.1007/s10462-004-7190-4.

R. S. Mangoubi and Dennis F. X. Mathaisel. Optimizing Gate Assignments at Airport Terminals. *Transportation Science*, 19(2):173–188, May 1985. ISSN 0041-1655. doi: 10.1287/trsc.19.2.173. URL https://pubsonline.informs.org/doi/abs/10.1287/trsc.19.2.173.

Rami Mangoubi. A linear programming solution to the gate assignment problem at airport terminals. Technical Report, Cambridge, Mass. : Massachusetts Institute of Technology, Flight Transportation Laboratory, [1980], 1980. URL http://dspace.mit.edu/handle/1721.1/67926.

Patrick McGeehan. Cuomo's $13 Billion Solution to the Mess That Is J.F.K. Airport. *The New York Times*, October 2018. ISSN 0362-4331. URL https://www.nytimes.com/2018/10/04/nyregion/jfk-airport-cuomo.html.

Bernhard Meindl and Matthias Templ. Analysis of commercial and free and open source solvers for linear optimization problems. *March, 2012*, page 14, March 2012.

Mercedes E. Narciso and Miquel A. Piera. Robust gate assignment procedures from an airport management perspective. *Omega*, 50:82–95, January 2015. ISSN 0305-0483. doi: 10.1016/j.omega.2014.06.003. URL http://www.sciencedirect.com/science/article/pii/S0305048314000760.

Takashi Obata. *The Quadratic Assignment Problem: Evaluation of Exact and Heuristic Algorithms*. Rensselaer Polytechnic Institute, 1979. Google-Books-ID: wbbYHAAACAAJ.

Erwin Pesch, Ulrich Dorndorf, and Florian Jaehn. Flight Gate Allocation : Modells , Methods and Robust Solutions, 2008. URL /paper/Flight-Gate-Allocation-%3A-Modells-%2C-Methods-and-Pesch-Dorndorf/21a14ed4019416c37d0a0cf33b3fc00e6e29b996.

Marian Peña. Robust clustering methodology for multi-frequency acoustic data: A review of standardization, initialization and cluster geometry. *Fisheries Research*, 200:49–60, April 2018. ISSN 0165-7836. doi: 10.1016/j.fishres.2017.12.013. URL http://www.sciencedirect.com/science/article/pii/S0165783617303557.

Simon Prent. *Regulation Aircraft Stand Allocation Schiphol*. Royal Schiphol Group, January 2019.

Boris Pushkarev. Urban space for pedestrians : a report or the regional plan association /, 1975.

Schiphol. Schiphol | Gevolgen van aangekondigde staking op maandag 18 maart, April 2019. URL https://www.schiphol.nl/nl/berichten/gevolgen-van-aangekondigde-staking-op-maandag-18-maart-2019.

Rupert Steiner, Jon Henley, Jamie Grierson, and Kim Willsher. Passengers facing delays and long queues at some European airports. *The Guardian*, August 2017. ISSN 0261-3077. URL https://www.theguardian.com/world/2017/aug/01/passengers-facing-four-hour-security-queues-at-some-european-airports.

Gerald N. Steuart. Gate Position Requirements at Metropolitan Airports. *Transportation Science*, 8(2):169–189, May 1974. ISSN 0041-1655. doi: 10.1287/trsc.8.2.169. URL https://pubsonline.informs.org/doi/abs/10.1287/trsc.8.2.169.

C. Tang. A Gate Reassignment Model for the Taiwan Taoyuan Airport Under Temporary Gate Shortages and Stochastic Flight Delays. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 41(4):637–650, July 2011. ISSN 1083-4427. doi: 10.1109/TSMCA.2010.2089512.

Ching-Hui Tang, Shangyao Yan, and Yu-Zhou Hou. A gate reassignment framework for real time flight delays. *4OR*, 8(3):299–318, October 2010. ISSN 1619-4500, 1614-2411. doi: 10.1007/s10288-009-0112-1. URL http://link.springer.com/10.1007/s10288-009-0112-1.

Gwyn Topham. Heathrow airport too expensive and overcrowded, say aviation chiefs. *The Guardian*, June 2013. ISSN 0261-3077. URL https://www.theguardian.com/travel/2013/jun/03/heathrow-airport-expensive-overcrowded-aviation.

S. M. Van Dijk. *Decomposition methods and rolling horizon approach for the yard crane scheduling problem*. PhD thesis, Delft University of Technology, Delft, 2015. URL http://resolver.tudelft.nl/uuid:2f22057e-3676-4103-8f22-021a2a4e01ba.

Godelieve Vanderstraeten and Michel Bergeron. Automatic assignment of aircraft to gates at a terminal. *Computers & Industrial Engineering*, 14(1):15–25, January 1988. ISSN 0360-8352. doi: 10.1016/0360-8352(88)90034-4. URL http://www.sciencedirect.com/science/article/pii/0360835288900344.

Shangyao Yan and Ching-Hui Tang. A heuristic approach for airport gate assignments for stochastic flight delays. *European Journal of Operational Research*, 180(2):547–567, July 2007. ISSN 0377-2217. doi: 10.1016/j.ejor.2006.05.002. URL http://www.sciencedirect.com/science/article/pii/S0377221706003316.

Dong Zhang and Diego Klabjan. Optimization for gate re-assignment. *Transportation Research Part B: Methodological*, 95:260–284, January 2017. ISSN 0191-2615. doi: 10.1016/j.trb.2016.11.006. URL http://www.sciencedirect.com/science/article/pii/S019126151630594X.

Yi Zhu, A. Lim, and B. Rodrigues. Aircraft and gate scheduling with time windows. In *Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence*, pages 189–193, November 2003. doi: 10.1109/TAI.2003.1250189.

Merve Şeker and Nilay Noyan. Stochastic optimization models for the airport gate assignment problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(2):438–459, March 2012. ISSN 1366-5545. doi: 10.1016/j.tre.2011.10.008. URL http://www.sciencedirect.com/science/article/pii/S1366554511001335.