TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

Mixed Integer Programming to Optimize Vienna's Airport Rolling Processes

Master's Thesis

to achieve the university degree of

Diplom-Ingenieur

submitted to

Vienna University of Technology

Institute of Statistics and Mathematical Methods in Economics
Research Unit: ORCOS

Author: Clemens Donà, BSc
Supervisor: Ao. Univ.Prof. Dipl.-Ing. Dr.techn. Gernot Tragler

Vienna, September 2017

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used.

Date	Signature

Zusammenfassung

Prozessplanungsprobleme haben in den letzten Jahren des vorigen Jahrhunderts einen großen Bedeutungsgewinn als Anwendungsfeld der gemischt-ganzzahligen linearen Optimierung (MILP) erfahren, was dem Aufkommen effizienter Lösungsmethoden auch für mittlere und größere Probleminstanzen geschuldet ist. In dieser Arbeit wird ein Modell formuliert, das folgerichtig aus dem Bereich des *job scheduling* kommt, um die Rollprozesse für landende und abfliegende Flugzeuge zu beschreiben. Der Anspruch dieser Thesis ist zweifacher Natur:

Einerseits wird die Wahl des Modells zunächst durch eine detaillierte Beschreibung der *airside* Operationen eines Flugzeugs im Flughafenbereich motiviert. Das Ziel der Optimierung ist jenes, die Gesamtrollzeit für einen gegebenen Flugplan zu minimieren, um die Effizienz dieser Operation zu steigern und damit die negativen ökonomischen wie ökologischen Auswirkungen eines zu lang laufenden Prozesses zu reduzieren. Das Modell wird mit Daten des Flughafens Wien-Schwechat (VIE) kalibriert, sodass eine Abschätzung über die potentielle Verbesserung des Rollprozesses ebendort gegeben werden kann.

Der zweite Schwerpunkt dieser Diplomarbeit liegt auf der Theorie der gemischt-ganzzahligen Optimierung. Dabei wird zunächst dargelegt, wie Probleme über Polyedern formuliert werden können und wie daraus verschiedene Arten von Relaxierungen und die mächtige Methode der *cutting planes* als Versuche der Annäherung an eine ideale Formulierung verstanden werden können. Die präsentierten Methoden münden schließlich im Branch-and-Cut-Verfahren, welches das Beste der beiden Welten vereint, sodass der Abschluss des theoretischen Teils gleichzeitig eine Beschreibung eines der heute meist verwendeten Lösungsverfahren darstellt.

Abstract

Process scheduling problems have become a considerable field of application for mixed integer linear programs (MILPs) with the raise of powerful methods to solve medium-large-scale models in a satisfying amount of time in the last decade of the previous century. In this work we propose a model that is inspired from the area of job scheduling to depict the rolling process for landing and departing aircrafts. Therefore, the claim of this thesis is twofold:

On the one hand, it is aimed to motivate carefully the choice of the model by describing the airside operations of an aircraft in the airport perimeter. The objective of the optimization is to reduce the overall rolling time for a given flight schedule in order to improve efficiency and reduce the negative economic and environmental consequences of an unnecessarily extended process. The model is fitted with data from the Vienna International Airport (VIE) and does provide an estimate on the potential margin of improvement in the rolling process.

On the other hand, the second focus of this thesis is on the theory of MILPs that lie in the class of NP-hard problems. It will be presented how problems can be formulated using polyhedra, motivating the introduction of different kinds of relaxations and the powerful cutting plane toolbox as attempts to approach the ideal formulation. The outlined methods culminate in the Branch-and-Cut algorithm combining the best of the two worlds, so that the conclusion of the theoretical part represents also a description of one of nowadays most used MILP solution procedures.

Contents

1 Introduction

Airports play a crucial role in the infrastructure of an economic area. In a study conducted by Eurocontrol in 2013 ([13]), the most likely scenario for Austria is calculated to see an average growth in flight numbers of 2.1% per year. In an absolute level, the projection assumes that 1.81 million compared to the 1.13 million flights in the year prior to the study's publication will be operated in Austria in 2035 (and thus, about 1.6 times as many). Nevertheless, in the European Union, although the predicted demand increase coincides approximately with the Austrian estimate, plans to enhance airport capacity are estimated with only 17% by 2035.

Before coming to examine the implications of this growth estimate and the possible strategies to react (where this thesis hopes to be a little contribution) it is useful to spend a few lines on the conception of the performed forecast since it might give the reader a feeling for the important factors in aviation that we will encounter throughout this report. The fundamental reasons to operate flights will be very likely the same in 20 years as they are today, namely to move people and goods in a safe, efficient, cost-effective and as environmentally friendly way as possible from one place to another. Aviation will have its main functions still in catalyzing various economic branches such as tourism, business and the manufacturing industry as it will provide connectivity in a world that urges for that. In order to arrive at the quantitative statement presented above, what remains to be done is to figure out the factors that might change dramatically in the future:

Regulations from public authorities are expected to play a crucial role, and here the regulations regarding the environment will have an outstanding position. In this context, the verdict of the Austrian Federal Administrative Court from February 2017 ([24]) can be seen, which forbids the construction of a third runway at Vienna's airport arguing that the public interest to be protected from the negative consequences of climate change - particularly the high CO_2 pollution - has to be judged higher than the positive influence on the industrial location and the labor market. It is further argued that Austria has committed itself in international treaties to reduce greenhouse gas emissions and that the airport has not taken own measures of sufficient degree to contribute to this target.

Taking a glance at the supply side, the main driver here is the cost factor of providing capacity for air traffic. While the vast majority of global airports is under state ownership or part of the public sector (see [20], page 21), more and more airports are denationalized. In our example, the shareholder structure of the Vienna Airport exhibits that 50% are privatized (see [3], page 18), while another 40% are in the hands of the two neighboring federal states. Especially for the latter it is likely that it will be more difficult in the next years due to the debts incurred in the economic crisis from 2008 onwards to invest substantially in new infrastructure to raise capacity, which typically requires a massive knock-on financing and pays off with continuous cash-flows over a long horizon. In any case, be it for the public sector's restricted budget constraints or the private shareholders' wish to be as cost-effective as possible, we can record the importance of using optimally the already available resources. The focus of this thesis shall be exactly on this point, namely we will examine if there is, taken current conditions for given, still potential to operate more effectively and thus satisfying the demand better. Before we come back to the precise description of the task in Chapter 2, we will first draw the reader's attention to some remarks to which we will refer in consequence and with which we hope to profound the reader's understanding of the problem.

To end the description of possible drivers of future demand, another reason mentioned in [13] is that different costumer segments will exhibit diverging propensities to fly: Retirees will be older and in the estimated scenario less wealthy (due to decreasing pension payments), while for business trips an increase is predicted. This segmentation of costumers is indeed of high importance for the matter of capacity, since a decline in one group and an incline in another one might leave the total number of passengers unchanged, but not the level of capacity needed: obviously, businesspersons travel at different times than tourists do and so there could be seen a rising demand for capacity on Monday mornings and a decreasing one on Tuesday middays, where price sensitive clients would travel, but the excessive demand cannot be shifted to the off-peak periods.

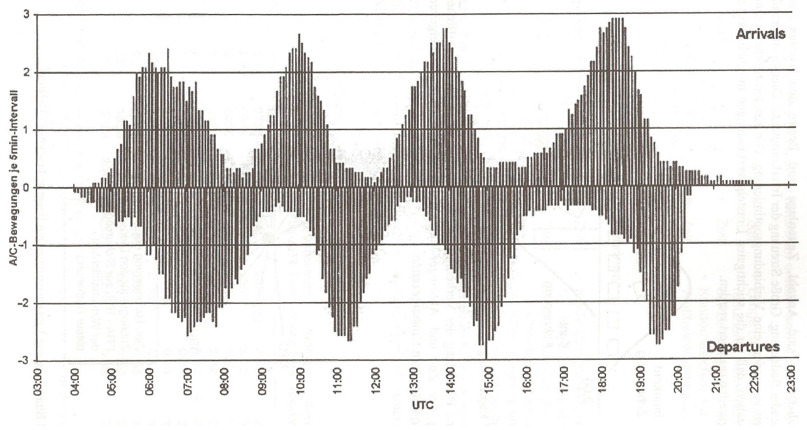

Figure 1.1: Flight movements in Frankfurt over a day

As we can read in [19] (page 241), as all other transport infrastructure sides, airports are planned so they can satisfy the demand at peak periods. The characteristic periods of interest for an airport are the definitions of a peak day and a peak hour. As it can be seen in the Figures 1.1 and 1.2, the distributions of departures and landings throughout a day and a year are not constant but are subject to fluctuations. It can be thus concluded that in the off-peak periods the demand can be easily met with the infrastructure at hand and that the focus shall be put on those moments in which the demand is critically high. This approach of determining peak periods is obviously a backward looking one that uses real data, but it can be also verified in a methodological way. For that purpose, we are using the method suggested in [19] and can thus, for any given point of time, state whether it is a peak day and/or hour. It shall only be mentioned that there is a quite wide range of possible approaches in the literature for methods to define peak periods (an interesting read could be [28]).

The day with the most passengers in the history of Vienna's Airport so far has been the 16th of September 2016, where 89.361 passengers have been served at the airport whereas there were 23.352.016 people in the whole year of 2016. The quantity that we have a look at is the number of flight movements, where

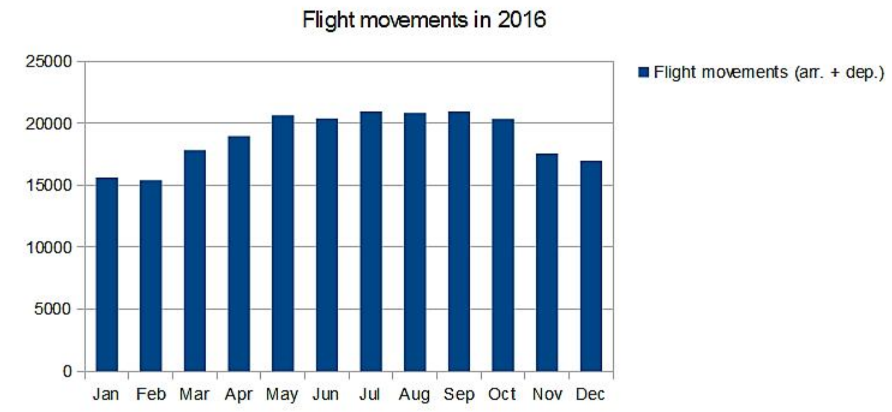

Figure 1.2: Flight movements in Vienna in 2016

there have been operated in total 226.395 for 2016 and a high three-digit number (that is known to the author, but for confidentiality reasons cannot be stated here) on the 16th of September ([3],[5]). Thus the percentage of all flight movements which were operated on the day of interest has been significantly over the threshold that is needed so the day can be defined as a peak day according to [19], where a percentage of $0,344\%$ or higher is required. Following the pattern seen in Figure 1.1 we consider the period from 6am to 9am, which is characteristically the first strong wave of flights in a day and accounts for one fifth of flight movements on the 16th of September and is thus indeed a sequence of peak hours.

Coming back to the matter of capacity constraints and the need to use the given infrastructure to an optimal extent, we will in this work tackle the question, which policy to implement so that flight movements are handled most effectively. This means, in more concrete words, that for the busiest day in the history of Vienna's Airport we will perform an analysis on how an airplane shall be directed from the moment it puts its wheels on the runway until it leaves the ground again for departure, i.e., as long as it is in the area of competence of the responsible airport. Here we shall be very precise: the rolling time for a single aircraft is minimized simply by taking the shortest path; the minimum rolling time for all aircrafts in a given period - and thus the optimization criterion in this thesis - amounts in scheduling their routes such that a global optimum is reached.

After having given the reader an introduction to the specific aspect of airport operations management we are examining in this work, we would like to present a short outline to the following chapters. In Chapter 2, after describing the processes during the stay at an airport and the particularities for Vienna-Schwechat, we will embed the here treated aspect shortly in the previous applications of Operations Research performed in the processes of our interest. With these preliminary steps we will be able in Chapter 3 to provide a formulation of the model as a mixed-integer linear program (MILP), providing the reader also with some parallels to other application fields. In Chapter 4, the mathematical foundations for the method used to solve the MILP (Chapter 5) will be presented enabling thus the reader to look behind the derived solution and understand what the solver does. After a sensitivity analysis to analyze the robustness of the derived results in Section 5.2, some final remarks and a compact outlook will conclude the thesis.

2 Airport operations

2.1 Processes at aircraft handling

The facilities of an airport are distinguished between *air side* and *land side*, where the first refers to all the technical facilities which secure the conducting of air traffic (i.e., where the focus lies on the aircraft) and the latter to the infrastructure that is targeted on attending the passengers (see [20], page 204). In this thesis, where we want to minimize the time window between the touch down and the arrival at the final parking position (and the other way around in the departure process), we will have to examine exclusively the air side facilities. The next paragraph follows the presentation in [20] (chapter 8) and [19] (chapters 10-12): In order to allow for a chronological description from the moment of the touchdown to the arrival at the parking position, a schematic depiction as in Figure 2.1 may be useful.

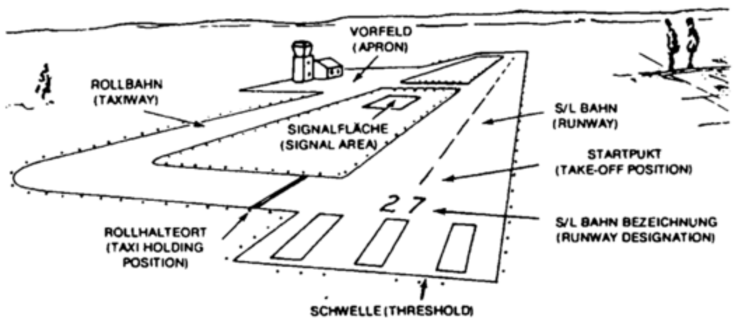
VORFELD (APRON)
ROLLBAHN (TAXIWAY)
SIGNALFLÄCHE (SIGNAL AREA)
S/L BAHN (RUNWAY)
STARTPUKT (TAKE-OFF POSITION)
2 7
S/L BAHN BEZEICHNUNG (RUNWAY DESIGNATION)
ROLLHALTEORT (TAXI HOLDING POSITION)
SCHWELLE (THRESHOLD)

Figure 2.1: Operation areas of an airport (simplified), [20] (page 204)

The point where an airplane has the first physical contact with the airport area is the *runway.* A runway at an airport is indicated according to its geographical orientation (which, for its part, is given by the prevailing direction of wind): runway 16/34 at Vienna's airport means that it extends along the direction 160 degrees/340 degrees, so approximately South-Southeast to North-Northwest ($0 = 360$ degrees coincides with North). Since an airplane lands and takes off always against the wind (i.e., if the wind blows from South-Southeast, an arriving plane lands at the North-Northwest end of the runway and its compass exhibits 160 degrees), we say that 16 is the *runway in use* as in the example just given. As it has been discussed briefly in Chapter 1, the Viennese airport has two runways at its disposal. It is furthermore crucial for the capacity and handling of departing and arriving flights how the layout of the runways looks like, and in Vienna they are intersecting each other in the extension, which results in a real capacity equal to 1.6 runways only ([4]). Additional factors are the length and width of runways so that (big and heavy) long-haul aircrafts can maneuver there without any restrictions. With lengths of 3.600 and 3.500 meters, respectively, and widths of 45 meters each, the runways in Vienna are suitable for all sizes ([2]). What remains to be specified for our purpose is to state when and for which processes an airplane is on the runway: Once the tower gave the landing clearance, the airplane is allowed to land in the touchdown zone (a particularly marked zone) of the runway in use and rolls on the runway while reducing its speed until the tower - that is in this area the control authority - gives the allowance to turn into a *taxi way.* On the other hand, a plane in departure rolls to the runway and is required to stop at the *take-off position*, until the tower gives the clearance to the captain to take-off. Afterwards, the airplane is allowed to accelerate and leaves the airport area.

In order to allow for a correct execution of airplane traffic within the airport area, a system of taxiways, the numbers and dimensions of which depend on the importance and size of the airport, is needed. Such a system shall assure that there exists a connection between the different zones of an airport that is as direct as possible. This is required mainly for two reasons, namely the reduction of rolling times (which also will be the minimization criterion in this thesis, see Chapter 3) and the augmentation of capacity. The suitable conception of the taxiways can thus be very differing for different airports, and can reach from one single road connecting the runway with the parking area and a turnaround point at that end of the runway where the taxiway does not inter-

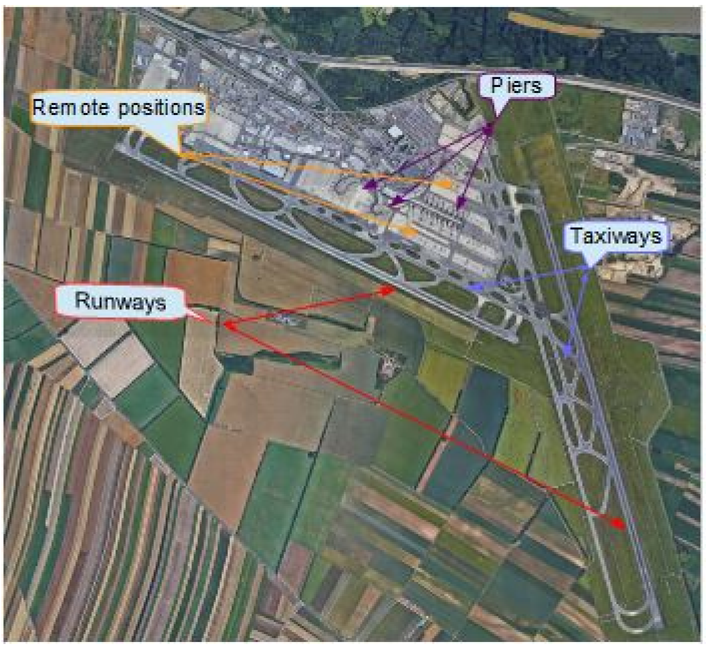

Figure 2.2: Aerial photograph of Vienna's airport

sect (so an airplane can take off and land in both directions) to a very complex system of taxiways with one-way routes within which parallel, interconnected ways exist. At the Airport Vienna-Schwechat, the given situation is the latter one (see Figure 2.2). The speed with which airplanes drive along the taxiways depends on the part of the area and can vary between 10-30 km/h and 50-60 km/h. This spread is due to the different processes that are all performed on what trades under the name taxiway. Around the parking positions of the airplanes, the rolling operations are performed at a very limited speed, while on runways peaks over 250 km/h are reached. Proceeding chronologically as before, a landing airplane turns off the runway and comes in to the taxiway once the speed has been reduced so much that the plane can take the turn; in order to minimize the time a plane has to remain on the runway while at the same time blocking it for the following aircraft, many airports have built so-called *high speed taxiways* which confine a small angle with the runway so it permits to enter at a higher speed; in Vienna, there exist 5 and 7 such high speed highways from the two runways, respectively ([2]). In a subsequent step, the airplane follows its way through the network of taxiways to the *apron.* In the departure procedure, the pilot steers the airplane in accordance with the commands from the tower to the *taxi-holding-position* (see Figure 2.1), which is the position that must not be crossed without an allowance from tower. Shall this position be occupied by previous machines, then the near holding bays are used by the waiting planes that prevent them from blocking the regular traffic on the taxiways.

The third area at an airport is the previously mentioned apron which consists of the apron-taxiways (used by the planes with a speed not superior to 20 km/h), parking and maintenance areas for airplanes and apron-service-roads for service cars and buses from the airport (that shall interfere as little as possible with the spaces where airplanes circulate or park). An incoming plane thus enters the apron at an interface between the taxiway and the apron-taxiway and is from then on under the responsibility of the airport operator. With his or her guidance, the plane reaches its final parking position and at this very moment, the rolling time for incoming machines is concluded. In Vienna, the airport has at its disposal 99 aircraft positions from which 36 allow gate parking ([3]). These are the spots where passengers reach the airplane from the waiting areas inside the airport building via extendible boarding bridges, while if the aircraft is assigned to a remote position, a gangway allows passenger to enter or exit the plane to which they are carried by an apron bus. Hence as long as an aircraft remains at its parking position, it is playing the role as connecting piece between the air and land side of the airport. As a result, one main driver for positioning a plane at a certain parking lot is the passenger handling, i.e., if it is known that a certain incoming flight serves mainly as a stopover for a connecting flight, then it shall be assured that the parking positions are scheduled in such a way that the travelers face a short walking distance inside the airport complex only. A second reason is that differing positions are granted at diverse prices (namely, a position at a pier is more expensive than one at the apron; see [23], page 10, for the price regime in Vienna) and thus low-cost airlines are likely to show other demand patterns for parking lots than expensive airlines. While these arguments arise from economic necessities, there exists as well a technical constraint: whether or not an aircraft can be assigned to a given position depends principally on the dimension of it, on the space it needs to maneuver and on the organization of means of service (luggage transport, cleaning staff, refueling kerosene) for the airplane while it is parked. In this time, passengers leave the aircraft, all the services necessary are performed and at the end the outgoing travelers enter the machine - using a technical term, we speak of the *turnaround time*. Every aircraft type has a minimum turnaround time (e.g., 38 minutes for a Boeing B737-300), but the actual time depends also on the services performed at the airport ([17]). Once the aircraft is ready to leave its parking position, it either does start by itself if it had been parked at a remote position or, if it had been *nose-in* (i.e., with the front part towards a gate), it is *pushed back* by a special vehicle until it can start its engines and leave the apron subsequently.

2.2 Literature review on optimizing air side processes

In the previous Section 2.1, the reader became acquainted with the core processes of air side operations. The following section has as a target to present a short review on research done in optimization of air side operations and embedding this thesis in that field pointing out at the one hand possible new insights and on the other hand starting points for further work (see Chapter 6).

A major part of OR work with focus on optimization deals with the effective use of runway systems. Facing the situation of having a fixed number n of aircrafts on the sky and figuring out a landing sequence in order to minimize the landing time of the last plane can be seen as a Hamiltonian path problem with n nodes ([9], page 384). If the situation is allowed to be dynamic, there are not always the same aircrafts to be stacked, but new ones to arrive with a landing request and other ones exiting the pool since they have already landed. The first work to the author's knowledge tackling this situation is a Ph.D. dissertation by Roger Dear ([12]) finding out that the first-come, first-serve (FCFS) discipline is inefficient in a dynamic environment and furthermore exhibits undesirable properties. Consequently, the CPS (Constrained Position Shifting) methodology is proposed to eliminate these problematics by prohibiting an incoming airplane from being shifted more than a given number from the position it would get under an FCFS regime. With this approach, the runway throughput rate can be increased (more specifically, it shows particular success in peak periods) as it treats the individual planes equitably in contrast with FCFS. Roger Dear furthermore extends his research to heterogeneous aircraft types and takes into account landings mixed with departures where he finds even greater improvements. After this preliminary work, there have been some publications building on it and aiming at a more realistic version of the sequencing issue facing thus higher complexity. Here shall be shortly discussed [10] because as in our report it performs a case study at an airport, namely Roma-Fiumicino. The authors combine a network and a scheduling model in which the inbound and outbound traffic flows in the so-called Terminal Maneuvring Area (which is the airspace in close proximity to an airport) are coordinated by scheduling procedures. In other words, the flow of aircrafts is not exogenously taken but is scheduled to optimality. The drawback of the more realistic representation of traffic flows is that real-time scheduling requires very fast algorithmic tools that are capable of dealing with big data sets. It is solved by providing a model based on discretizing the airspace so to capture the evolution of flows in the Terminal Maneuvring Area together with a set of algorithms optimizing the schedule for various objective functions.

Somewhat closer to the topic of this thesis is a report ([26]) that proposes a model to reduce congestion from departing airplanes within the airport area in periods when the capacity of the air side infrastructure reaches its limit. It is thus opportune to present in a few words that model and to mention its advantage and drawback compared to the model presented later in Chapter 3. There are two servers for queues, namely the terminal system as the natural "beginning" of the departure process and the runway(s) as its "end". The stochastic distribution of leaving the terminal and arriving at the runway is derived from the available data about the empiric taxi-times. The authors do thus aim at providing a model of an airport departure process that is validated with operational data from the 10 main US airlines departing from the airport for a whole year and making use of this model to quantify the effects of departure process control. While the scope is a very similar one to this thesis (with the difference that there the focus lies on the departure process exclusively), the methodology chosen in [26] claims to be less costly in obtaining statistically significant validation data for different airport configurations (e.g., think of changing wind directions that change the departure direction and thus have a high impact on the control process) and to carry out exhaustive validation from this data. An advantage of the model of this thesis instead is its highly detailed depiction of the airport layout that allows to test procedural changes easily (e.g., incorporating the meanwhile closure of a parking position) and can - for the configuration examined - determine the optimal policy. In [26] the term “microscopic” is introduced for such models which is a very accurate wording in the author’s opinion.

The reader who wants to shape his or her expertise in the OR work of the land side of airports - that, as we have seen in Section 2.1, can have influence on the air side - shall be referred examplarily to part IV of [22].

2.3 Parameters for VIA

We shall now come to the specification of Vienna airport’s particularities. Based on the general description in Section 2.1, where we have discussed the processes that are carried out by an aircraft and the restrictions that it might possibly face, the data presented in the following paragraph can be understood as an indication of parameters at the airport Wien-Schwechat. It has to be stated that the Vienna International Airport and the author have agreed on confidentiality, so that what is outlined here is in accordance with the agreement, aiming still to give the reader a feeling for the parameters and explaining how they have been established.

- Runways: 2
- Parking positions: 99, of that 36 gate positions
- Average taxi times: see Table 2.1

Runway	**Time to apron positions**	**Time to pier positions**
11	4 mins	5 mins
16	7 mins	10 mins
29	3 mins	4 mins
34	5 mins	7 mins

Table 2.1: Average taxi times at Vienna International Airport [2]

For the model, a more accurate matrix with the long-term average rolling times (in- and outbound) from each runway to each position for the different aircraft types is used, but the indications in Table 2.1 should provide a good impression.

- Flight schedule: The flight schedule for the 16th of September 2016 was made available by the airport authorities. From this, we derive easily the arrival and departure times of every single aircraft. One shall be aware that in this thesis we work with the scheduled times, since when the position assignment was made, the actual times were not known yet. Another piece of information from the schedule is the runway in use, since it is recorded for each aircraft the runway where it landed or from which it departed. Furthermore, it is recorded which aircraft type was used to operate a flight, which is important to know since certain types can for technical reasons only be assigned to certain positions.
- Turnaround times: As discussed above, the minimum turnaround time depends on the aircraft type as on the operations performed during parking. Therefore, it would be necessary to know the characteristics of each aircraft as specified by the manufacturer as well as the operations performed by the airport stuff. Nevertheless, the formulation given in Chapter 3 requires a specification of the minimum turnaround time only in order to make sure that pathological outcomes (e.g.: the aircraft leaves a position before it arrives there) are avoided, although this would never happen since it would imply a rolling time that is far longer than any reasonable outcome and thus all the opposite of a minimization. Therefore, although the usage of the term "turnaround time" might be nonchalant here, we will set the parameter equal to the minimum turnaround time an aircraft had on the day of interest in case it did an inbound and outbound trip effectively.
- Interdependence of positions: Additionally, there has been incorporated a list of positions' interdependences, which means nothing else than "if position x is occupied, position y is not available for parking". This arises due to guidelines of minimum distances between two aircrafts and will show to be a parallelism to a very basic requirement.

3 Model

3.1 Model formulation

The model presented in this chapter is formulated as a mixed integer linear program (MILP), so a model that contains in addition to integer variables also continuous ones. While the theory of MILPs and the ways to solve these problems will be discussed in a detailed way in Chapter 4, it is indispensable not to describe in a few words the area of application to which this model can be attributed. Indeed one important branch of MILPs in practice is constituted by scheduling problems. A very classical model has been formulated for example in [8] (pp. 22) and shall be - as a foundation - explained here:

Two sets of m machines and n products are given. For each product j we suppose that we know the times $p_{j,k}$ that a product j spends on machine k. The task is to determine an optimal schedule such that the overall time (i.e., the time needed for all products to be completed) is minimized. Two important assumptions are made and will be met in our model again. Firstly, two or more products can't be processed simultaneously on the same machine (so, if once occupied by one product, the machine is ready to accept another product only when it has completed the current job) and secondly, a process once started is not allowed to be interrupted. In the basic version it is furthermore assumed for reasons of simplicity that each product has to be processed on all machines in the same order indicated by the indices of the machines. As stated above, the model returns the minimum throughput time for a given number of products, and thus determines when a product shall start being processed on the machines.

The analogy between this baseline and our model can be made by considering the products as aircrafts that need to pass through processes, which previously had been the machines. While a model as the baseline one would be equivalent to minimize the usage time of the airport (which could be the equivalent to a whole factory in the baseline image), in our formulation we take the starting times of the processes as given and known as are the processes themselves, but what we want to determine is the assignment to one position out of many possible *within* a process step, where all the positions fulfill the aim of the process "parking" but they exhibit different properties that allow us to compare two assignments with each other and thus search for an optimal one.

In the following paragraph we give the MILP formulation of this thesis' model:

Sets

- $A = \{a_1, \dots, a_n\}$: aircrafts that are recorded to land and/or depart during the period of observation
- $Y = \{y_1, \dots, y_l\}$: type of the aircraft a, for example Airbus A320
- $R = \{R11, R16, R29, R34\}$: runways at the Airport Wien-Schwechat
- $P = \{p_1, \dots, p_m\}$: parking positions at the Airport Wien-Schwechat
- $K = \{1, 2, 3\}$: operations, where: $\begin{cases} 1, & \text{landing} \\ 2, & \text{arrival to gate} \\ 3, & \text{departure from gate} \end{cases}$
- $N \subset A$: flights that are in- and outbound within the period of observation
- $I \subset A$: flights that are only inbound within the period of observation
- $O \subset A$: flights that are only outbound within the period of observation

Parameters

- $d\_i_{y,r,p} \in \mathbb{R}^+$: rolling time for an aircraft of type $y \in Y$ from touchdown at runway in use $r \in R$ to position $p \in P$
- $d\_o_{y,r,p} \in \mathbb{R}^+$: rolling time for an aircraft of type $y \in Y$ from position $p \in P$ to the take-off at runway in use $r \in R$

- $u\_i_{a,r} \in \{0,1\} : \begin{cases} 1, & \text{runway } r \in R \text{ is in use for inbound flight } a \in N \cup I \\ 0, & \text{otherwise} \end{cases}$
- $u\_o_{a,r} \in \{0,1\} : \begin{cases} 1, & \text{runway } r \in R \text{ is in use for outbound flight } a \in N \cup O \\ 0, & \text{otherwise} \end{cases}$
- $typ_{y,a} \in \{0,1\} : \begin{cases} 1, & \text{aircraft } a \in A \text{ is of type } y \in Y \\ 0, & \text{otherwise} \end{cases}$
- $pos_{p,y} \in \{0,1\} : \begin{cases} 1, & \text{position } p \in P \text{ is feasible for type } y \in Y \\ 0, & \text{otherwise} \end{cases}$
- $t_{a,1} \in \mathbb{R}^+$: starting time of operation 1 for airplane $a \in A$, i.e., time of touchdown
- $t_{a,3} \in \mathbb{R}^+$: starting time of operation 3 for airplane $a \in A$, i.e., time of leaving the position for departure
- $cat_a \in \mathbb{R}^+$: minimum turnaround time for an aircraft $a \in A$, i.e., the minimum time interval required between operations 2 and 3
- C: high constant; feasible choice, e.g., can be $C = \max_{a \in A} t_{a,3}$
- $\tilde{C}$: constant; feasible choice, e.g., can be $\tilde{C} = 3$
- ϵ: positive, but small constant; it shall be selected so that its first non-zero entry appears at a position that is lower than the measuring accuracy of the starting time of the operations (e.g., for times of format XXX.X sec, $\epsilon = 0.01$ would be a feasible choice)

Variables

- $x_{p,a} \in \{0,1\} : \begin{cases} 1, & \text{if aircraft } a \in A \text{ is assigned to position } p \in P \\ 0, & \text{otherwise} \end{cases}$
- $t_{a,2} \in \mathbb{R}^+$: starting time of operation 2 for airplane $a \in A$, i.e., time of arriving at parking position

- $y_{a,b} \in \{0,1\} : \begin{cases} 1, & \text{if } t_{a,2} \leq t_{b,3}, \text{ where } a,b \in A, a \neq b \\ 0, & \text{otherwise} \end{cases}$
- $z_{a,b} \in \{0,1\} : \begin{cases} 1, & \text{if } t_{b,2} \leq t_{a,2}, \text{ where } a,b \in A, a \neq b \\ 0, & \text{otherwise} \end{cases}$
- $w_{a,b} \in \{0,1\} : \begin{cases} 1, & \text{if } t_{b,2} \leq t_{a,2} \leq t_{b,3}, \text{ where } a,b \in A, a \neq b \\ 0, & \text{otherwise} \end{cases}$

Model

$$\min_{x_{.,.},t_{.,2}} \sum_{r\in R}\sum_{y\in Y}\sum_{p\in P}(d\_o_{y,r,p} * (\sum_{m_1\in N\cup O} u\_o_{m_1,r} * typ_{y,m_1} * x_{p,m_1})) + \sum_{m_2\in N\cup I}(t_{m_2,2} - t_{m_2,1}) \tag{3.1}$$

s.t.

$$\sum_{p\in P} x_{p,a} = 1 \quad \forall\, a \in A \tag{3.2}$$

$$t_{a,2} \geq t_{a,1} + \sum_{r\in R}\sum_{y\in Y}\sum_{p\in P} u\_i_{a,r} * d\_i_{y,r,p} * typ_{y,a} * x_{p,a} \quad \forall\, a \in N \cup I \tag{3.3}$$

$$t_{a,2} + cat_a \leq t_{a,3} \quad \forall\, a \in N \cup I \tag{3.4}$$

$$t_{a,2} = t_{a,1} \quad \forall\, a \in O \tag{3.5}$$

$$t_{a,2} \leq t_{b,3} + C * (1 - y_{a,b}) \quad \forall\, a,b \in A, a \neq b \tag{3.6}$$

$$t_{a,2} + \epsilon \geq t_{b,3} - C * y_{a,b} \quad \forall\, a,b \in A, a \neq b \tag{3.7}$$

$$t_{b,2} \leq t_{a,2} + C * (1 - z_{a,b}) \quad \forall\, a,b \in A, a \neq b \tag{3.8}$$

$$t_{b,2} + \epsilon \geq t_{a,2} - C * z_{a,b} \quad \forall\, a,b \in A, a \neq b \tag{3.9}$$

$$0 \leq y_{a,b} + z_{a,b} - 2 * w_{a,b} \quad \forall\, a,b \in A, a \neq b \tag{3.10}$$

$$y_{a,b} + z_{a,b} - 2 * w_{a,b} \leq 1 \quad \forall\, a,b \in A, a \neq b \tag{3.11}$$

$$w_{a,b} * x_{p,b} \leq 1 - x_{p,a} \quad \forall\, p \in P, \forall\, a,b \in A, a \neq b \tag{3.12}$$

$$t_{a,2} \geq 0 \quad \forall\, a \in A \tag{3.13}$$

$$x_{p,a} \in \{0,1\} \quad \forall\ p \in P, a \in A \tag{3.14}$$

$$y_{a,b}, z_{a,b}, w_{a,b} \in \{0,1\} \quad \forall\ a,b \in A, a \neq b \tag{3.15}$$

One shall be aware of the fact that equations as of form (3.12) can be linearized as follows:

$$w_{a,b} - \tilde{C} * (1 - x_{p,b}) \leq 1 - x_{p,a} \quad \forall\ p \in P, \forall\ a,b \in A, a \neq b \tag{$\widetilde{3.12}$}$$

This might seem to be a little variation only, but indeed it is of crucial importance: We achieve to transform a mixed integer non-linear program (MINLP) into an MILP and thus can refer to all the results available in this branch rather than entering in the far less explored world of MINLPs.

3.2 Model explanation

The objective function we formulated in (3.1) is the minimization over the rolling times, where we have to observe that aircrafts which do the "normal" round trip (inbound and outbound) have to cover a rolling distance between operations 1 and 2 and another one between the end of 2 and 3 while only incoming (or outgoing) aircrafts have only one, namely the first (the second), distance to cover. One shall notice the difference between the in- and outbound process in the objective function: While it is known when an aircraft touches down, it can only take the direct way to a position if it is free, and indeed with the formulation given it is taken into account that there might arise moments in which all feasible positions are occupied and thus the term in the second sum is strictly bigger than the parameter $d\_i_{y,r,p}$ for y, r, p known. For the outbound process instead the assumption is made that the rolling time is given by the parameter value $d\_o_{y,r,p}$, which might surprise at the first sight, but is justified given that the parameters are average values for a long observation period. In accounting for the parameters $u\_o_{a,r}$ and $typ_{y,a}$ it is taken into consideration which is the runway a given aircraft will reach for to take off and which is the aircraft's type.

Proceeding with the constraints of the model, (3.2) states simply that each aircraft has to be assigned to exactly one parking position. In (3.3)-(3.5) it is expressed the definition of the starting time of operation 2, which encodes the arrival at the position. It results on the one hand from adding to the touchdown time the rolling time to the position that is decided to be the parking one. Clearly, operation 2 cannot be started before that moment. On the other hand it must start soon enough so that, when adding the minimum turnaround time, the sum does still not exceed the ending time of operation 2 which is simply given by $t_{a,3}$. The previous consideration is valid only for aircrafts that actually go through the inbound process, so all aircrafts $\in I$ and $\in N$. For aircrafts that are only departing, $t_{a,2}$ is set equal to $t_{a,1}$.

The equations (3.6) - (3.12) form a block of matching constraints that is worth to be examined closer: First of all, in its very nature the variables $y_{a,b}$ and $z_{a,b}$ are auxiliary variables which indicate if an aircraft a does arrive at a parking position while another aircraft b has parked already and not left the position yet at this very moment (if and only if both $y_{a,b}$ and $z_{a,b}$ take the value 1). We shall spend a few words here since binary variables are predestined to formulate logical conditions: According to its definition, $y_{a,b}$ (for a, b fixed) shall be equal to 1 if and only if $t_{a,2} \leq t_{b,3}$. In (3.6) the requirement $y_{a,b} = 1 \Rightarrow t_{a,2} \leq t_{b,3}$ is reflected, since $y_{a,b} = 1$ deletes the additive term on the right-hand side and forces $t_{a,2} \leq t_{b,3}$ to hold. The other direction needs to be assured as well, and therefore we notice the following equivalence:

$[y_{a,b} = 1 \Leftarrow t_{a,2} \leq t_{b,3}] \Leftrightarrow [\neg(y_{a,b} = 1) \Rightarrow \neg(t_{a,2} \leq t_{b,3})] \Leftrightarrow [y_{a,b} = 0 \Rightarrow t_{a,2} > t_{b,3}]$

The only correction left is that we need to get rid of the strict inequality and do that by adding a constant small enough (ϵ) to the left-hand side and by replacing by a not-strict inequality sign.

Since these sets of variables y and z are defined for each pair of two diverse aircrafts, it can be determined for every pair if there takes place a temporal overlapping at the parking positions (precisely, this is done in (3.10) and (3.11)). The preceding preparation work amounts in (3.12), as the overlapping indicator $w_{a,b}$ is connected with the scheduling variables $x_{a,p}$ and $x_{b,p}$. If there is no temporal overlapping of aircrafts a and b, all choices are possible (aircrafts a *and* b can park at position p, only a *or* only b or neither of them). If aircraft a arrives at the parking area of the apron while b is parked, it must be prevented from heading towards the position occupied by b. So, if $w_{a,b}$ takes the value 1, there are still all options available but one: a *or* b can park at position p or neither of them does, thus $\neg(x_{a,p} = 1 \wedge x_{b,p} = 1)$. It can be seen that (3.12) delivers what is desired, but on the left-hand side two variables are multiplied and so the linearity of the model gets lost. Thus the linear reformulation $(\widetilde{3.12})$ is chosen, which again does only function as a restriction for $x_{a,p}$ and $x_{b,p}$ if $w_{a,b}$ is equal to 1 and if so, then it excludes parking at the same position (therefore the term reformulation is justified, since the logical properties of (3.12) and $(\widetilde{3.12})$ are the same).

The constraints (3.13)-(3.15) are nothing else than the statement to which numerical range the variables belong to.

With this, the basic foundation of a minimization model for rolling times is concluded. For the many more constraints that are airport specific we want to just give a very short comment so the reader understands that once understood the basic concept, further constraints are easy to model: If the occupation of a certain position forces the neighboring positions to be free during the whole period, the parallel to the basic requirement “if one position is occupied by one aircraft, it cannot be available for any other aircraft in that period” is obvious - instead of preventing all other aircrafts for a certain amount of time to park at the occupied position, it is simply forbidden in addition to park at the neighbor positions.

4 Algorithms to solve MILPs

Nowadays there exists a wide range of commercial and free solvers to deal with optimization problems. The aim of this thesis is to not only formulate and solve a practical problem, but also to provide an insight to what happens "behind the scenes", i.e., the mathematical foundations on which the methods of popular available solvers are based. The following chapter thus has as a target to present the world of mixed integer linear programming from the formulation of problems over various methods to their combination into an algorithm to solve them.

It shall be said that we are not going to retrace the manual of a specific solver, but rather present a framework for mixed integer linear programs with which it becomes clear how a solver works and does generate the output (=solution) for a given input (=model).

The presentation is inspired by [29], but refers to other works whenever explicitly mentioned for more detailed explanation. Also, a very good introduction is offered by [1] where one later finds the particularities of the SCIP solver (i.e., which branching strategy is used, which heuristics for upper bounds, etc.).

4.1 Formulation

Let us suppose c and g to be rational row vectors of length n and m respectively, $A \in \mathbb{Q}^{q \times n}$ and $B \in \mathbb{Q}^{q \times m}$ two matrices and h a rational column vector with the appropriate dimension q.

Definition 4.1.1. *A mixed integer linear program (MILP) is an optimization problem of the form*

$$\begin{aligned} &\min_{x,y} c * x + g * y \\ A * x + B &* y \leq h \\ x \in \mathbb{R}^n_{\geq 0}&, y \in \mathbb{Z}^m_{\geq 0} \end{aligned} \tag{4.1}$$

In our model from Section 3.1 we face the additional requirement that $y \in \{0,1\}$, which is just a special case of Definition 4.1.1. While such problems without the continuous part (i.e., $c = 0$, $A = 0$) are referred to as 0-1 ILPs in the literature, for models as ours there is no special name. One might note that the inequalities in (4.1) do not mean a restriction, since any equality can be written as two inequalities.

We shall now state more precisely what is meant by the formulation of a MILP.

Definition 4.1.2. *Let $P \subseteq \mathbb{R}^n$. If P can be described by a finite set of linear constraints, i.e., $P = \{x \in \mathbb{R}^n : A * x \leq b\}$, we say P is a polyhedron.*

Geometrically, we can thus think of a polyhedron as the intersection of finitely many closed halfspaces (as illustrated examplarily in Figure 4.1).

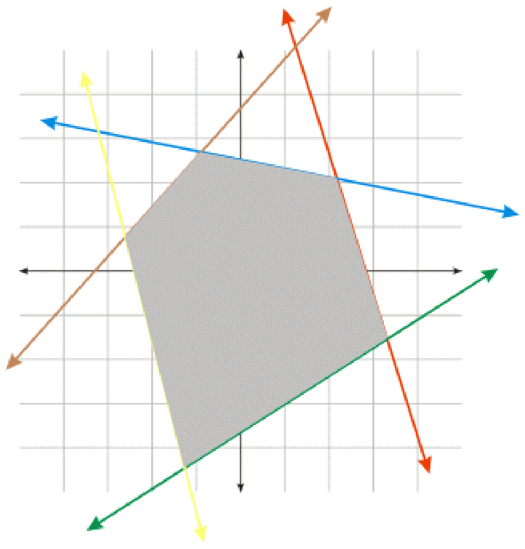

Figure 4.1: A polyhedron in $\mathbb{R}^2$ is an intersection of halfspaces

Definition 4.1.3. *A polyhedron $P \subseteq \mathbb{R}^{n+m}$ is called formulation of a set $X \subseteq \mathbb{R}^n \times \mathbb{Z}^m$ if and only if $P \cap (\mathbb{R}^n \times \mathbb{Z}^m) = X$.*

Example 4.1.4. *Let the pure integer set $X \subseteq \mathbb{Z}^2$ be given as*

$$X = \{(1,1),(2,1),(3,1),(1,2),(2,2),(3,2),(2,3)\}.$$

In Figure 4.2 we see three different possible formulations of X ($P_1 \cap \mathbb{Z}^2 = P_2 \cap \mathbb{Z}^2 = P_3 \cap \mathbb{Z}^2 = X$). Note that there exists clearly an infinite number of formulations. If we have to solve a linear optimization problem over the set X and suppose we do not know anything about Integer Programming yet, one could come up with the idea to solve the problem over the polyhedron P_i and thus have at hand the tools of Linear Programming. The "smaller" the region P_i that has to be explored is, the "closer" we will intuitively stay to the integer solution.

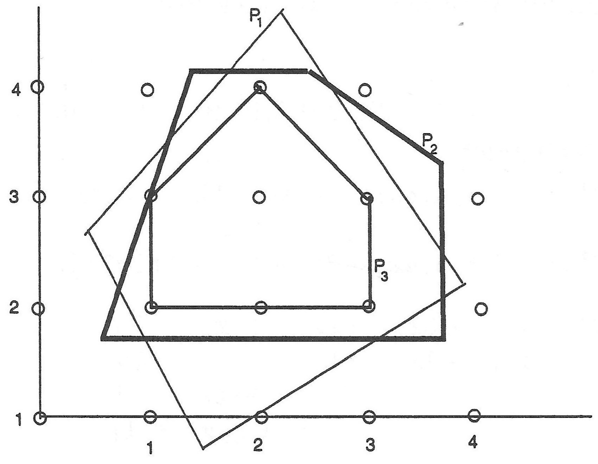

Figure 4.2: Different formulations for the region X from Example 4.1.4

We shall formulate in a stringent mathematical way what has just been outlined.

Definition 4.1.5. *A set $X \subseteq \mathbb{R}^n$ is called convex if $\forall\ x_1, x_2 \in X, \forall\ t \in (0,1)$ it holds that $t * x_1 + (1-t) * x_2 \in X$. In other words, for every two points of the set also their connecting line is fully contained in X.*

Definition 4.1.6. *For a set $X \subseteq \mathbb{R}^n$ the convex hull of X or conv(X) is defined as the intersection of all convex sets containing X, i.e., conv(X):=* $\bigcap\limits_{X \subseteq K \subseteq \mathbb{R}^n} K$ *for convex K.*

Proposition 4.1.7. *conv(X) is a convex subset of $\mathbb{R}^n$.*

Proof. Let $\mathcal{K}$ be a family of convex sets $\subset \mathbb{R}^n$ such that $\bigcap \mathcal{K} := \operatorname{conv}(X)$ holds. For two elements x_1, $x_2 \in \bigcap \mathcal{K}$ it holds that x_1, $x_2 \in K \ \forall K \in \mathcal{K}$. Since every K is convex by definition, it holds that $\forall K \ \forall t \in (0,1) : t*x_1+(1-t)*x_2 \in K$. Thus, $t * x_1 + (1-t) * x_2 \in \bigcap \mathcal{K}$ and therefore $\bigcap \mathcal{K}$ is convex. □

Theorem 4.1.8. *For a set X with the formulation P let P_I :=conv(X). Then P_I is a polyhedron (in $\mathbb{R}^n$).*

Proof. [25], pp.360 □

Definition 4.1.9. *Let K be a convex set. A point $x \in K$ is said to be extreme if $x = t * x_1 + (1-t) * x_2, x_1$ and $x_2 \in K$ implies $x = x_1 = x_2 \ \forall t \in (0,1)$.*

Theorem 4.1.10. *All extreme points of conv(X) lie in X.*

Proof. Suppose $x \in \operatorname{conv}(X)$ to be an extreme point. According to the definition of the convex hull in [29] which is equivalent to our definition 4.1.6 we can represent x as $x = \sum_{i=1}^m t_i * x_i, \sum_{i=1}^m t_i = 1, t_i \geq 0 \ \forall i = 1, \ldots, m$ with $\{x_1, \ldots, x_m\}$ being a finite subset of X. Since x is extreme, it must follow that $t_i = 1$ for one i (and thus 0 for all the others). Subsequently, we get $x = x_i$ and since $x_i \in X$ we can conclude that x lies in X. □

The reader shall notice that in the literature the results 4.1.8 and 4.1.10 are often referred to as *Meyer's theorem* (see, e.g., [11]).

We have been doing a lot of preparation work so far, but now we are at the point at which we can state the main insight of this section: Knowing from linear programming that the optimal solution over a polyhedron is - if there is any - attained at an extreme point, we can now replace the MILP (4.1) (with the feasible region denoted here $X \subseteq \mathbb{R}^n \times \mathbb{Z}^m$) by a linear program (LP)

$$\begin{aligned} &\min_{x,y}\; c * x + g * y \\ &(x, y) \in \operatorname{conv}(X) \end{aligned} \tag{4.2}$$

At this point, we might wonder why at all shall we further explore MILPs, if we have just seen that they can be equivalently posed as LPs. Indeed, the above stated formulation is in this sense the *ideal* one:

Definition 4.1.11. *For a mixed integer set $X \subseteq \mathbb{R}^n \times \mathbb{Z}^m$ we call the polyhedron $P \subseteq \mathbb{R}^{n+m}$ the ideal formulation if and only if $P =$ conv(X).*

The problem we face in real-world applications is that in general, finding conv(X) is very difficult. In the following sections we will see how conv(X) can still be approached sufficiently well so MILPs can be efficiently solved. In order to have a criterion at hand to say what a "good" formulation is, we need a last definition here:

Definition 4.1.12. *For a mixed integer set $X \subseteq \mathbb{R}^n \times \mathbb{Z}^m$ we say a formulation P_1 is better than a formulation P_2 if and only if $P_1 \subset P_2$.*

Example 4.1.13. *In Figure 4.2 the formulations P_2 and P_1 are both worse than P_3, while neither of them is better than the other. P_3 is also the ideal formulation of the feasible region.*

4.2 Relaxations

If it is - apart from some special cases - not possible to determine the convex hull of the feasible region, the main question is how can a solution for a problem as in Definition 4.1.1 be found. A first intuitive guess could be that we may look for a lower bound and an upper bound for the objective value which coincide, because then we would have solved the original problem.

For $z = \min_{x \in \mathbb{R}^n_{\geq 0}, y \in \mathbb{Z}^m_{\geq 0}} \{c * x + g * y : A * x + B * y \leq h\}$ we need thus to find an upper bound $\overline{z} \geq z$ and a lower bound $\underline{z} \leq z$ that fulfill $\underline{z} = \overline{z}$. Therefore, to solve such problems practically, algorithms which generate an increasing (decreasing) sequence of lower (upper) bounds and have a small constant ϵ for the termination criterion $|\overline{z_{n_1}} - \underline{z_{n_2}}| \leq \epsilon$ can be used.

Regarding the monotonously decreasing sequence of upper bounds $(\overline{z_n})_{n\in\mathbb{N}}$, the point is to find feasible solutions to the minimization problem that approach the optimal solution step after step of the algorithm. In a problem as posed in Example 4.1.4 one can find a feasible solution in a straightforward way. For other MILPs, finding even a feasible solution can be very difficult. We shall come back shortly to describe a heuristic approach to determine an upper bound at the end of this section.

Initially we will address the issue of finding lower bounds. Roughly speaking, the aim is to return the solution of a minimization problem that has a bigger feasible region than the original one. We will see in the following that indeed such a new problem will provide us with a lower bound.

Definition 4.2.1. *A problem $min_{x,y}\{\tilde{f}(x,y) : (x,y) \in \tilde{X} \subseteq \mathbb{R}^{n+m}\}$ is called a relaxation of the original problem $min_{x,y}\{f(x,y) : (x,y) \in X \subseteq \mathbb{R}^{n+m}\}$ if and only if $X \subseteq \tilde{X}$ and $\tilde{f}(x,y) \leq f(x,y)$ $\forall (x,y) \in X$.*

Proposition 4.2.2. *Let $\tilde{z} = min_{x,y}\{c * x + g * y : (x,y) \in \tilde{X} \subseteq \mathbb{R}^{n+m}\}$ and $z = min_{x,y}\{c * x + g * y : (x,y) \in X \subseteq \mathbb{R}^{n+m}\}$. Then $\tilde{z} \leq z$.*

Proof. Denote by $(x^\star, y^\star)$ the solution of the original problem. Clearly, $(x^\star, y^\star)$ is a feasible solution to the relaxation since $X \subseteq \tilde{X}$. Therefore, $\tilde{z} \leq c * x^\star + g * y^\star = z$. □

Proposition 4.2.3. *Let $(\tilde{x}^\star, \tilde{y}^\star)$ be the optimal solution of the relaxation. If $(\tilde{x}^\star, \tilde{y}^\star) \in X$, then it is also the optimal solution for the original problem.*

Proof. $(\tilde{x}^\star, \tilde{y}^\star) \in X$ means that it is feasible for the original problem. Since it is optimal over the relaxed region $\tilde{X}$, the optimality for X follows instantaneously. □

We will now discuss three possible relaxations to derive lower bounds, in which we will also see that drawbacks are encountered if the relaxation is chosen in a too naive way.

Definition 4.2.4. *A relaxation by elimination is the result of eliminating at least one constraint from the original problem.*

This is probably the most straightforward way to relax a MILP, but as it might be expected - since the aim is to drop the difficult constraints, which comes at a price - it does yield very weak lower bounds in practice only. Therefore, more sophisticated ways of relaxations are required.

Definition 4.2.5. *For a MILP as in* (4.1) *the linear relaxation is defined as*

$$\min\{c * x + g * y : A * x + B * y \leq h, (x, y) \in \mathbb{R}^{n+m}_{\geq 0}\}$$

Proposition 4.2.6. *The linear relaxation is indeed a relaxation as defined in 4.2.1.*

Proof. Denote the region defined by the inequalities $A * x + B * y \leq h$ by P. Then we can write the MILP as $\min\{c * x + g * y : (x, y) \in P \cap (\mathbb{R}^n_{\geq 0} \times \mathbb{Z}^m_{\geq 0})\}$. The problem of the linear relaxation can then be written as $\min\{c * x + g * y : (x, y) \in P\}$, and since $P \cap (\mathbb{R}^n_{\geq 0} \times \mathbb{Z}^m_{\geq 0}) \subseteq P$, it is a relaxation as in Definition 4.2.1. □

As we have argued before, it is desirable to derive lower bounds that do not deviate too much from the actual solution ("the tighter the better"). It is thus reasonable to ask whether there are linear relaxations that are better than others. Indeed this is the case, as the following proposition shows.

Proposition 4.2.7. *For the MILP of* (4.1) *suppose we have two formulations* P_1 *and* $P_2 \in \mathbb{R}^{n+m}$, $P_2 \subseteq P_1$ *given. It holds that* $z_1 \leq z_2$ *with* $z_i := \min\{c * x + g * y : (x, y) \in P_i\}$.

Proof. The proof follows immediately from Proposition 4.2.2 if we identify P_1 with $\tilde{X}$ and P_2 with X. □

The linear relaxation has a further advantage with respect to other kinds of relaxations: In special cases, solutions of the linear relaxation permit conclusions on the solution of the underlying MILP.

Proposition 4.2.8.

- *If the linear relaxation of a MILP is infeasible, the MILP does not have a solution.*
- *For an optimal solution $(x^\star, y^\star)$ of the linear relaxation it holds that it is optimal for the MILP if $(x^\star, y^\star)$ is feasible for it.*

Proof.

- Infeasibility of a problem means that the feasible region is empty. If $\tilde{X} = \emptyset$, so is X.
- It follows from the linearity of the objective function and Proposition 4.2.2.

□

Now we examine a last kind of relaxation that generalizes the approach of the simple elimination in a more elaborated way. Suppose for that purpose that the constraints of a MILP can be divided into "easy" and "difficult" ones. In our case, constraints are "easy" if the problem with only these constraints can be solved easily.

Definition 4.2.9. *Suppose we have again the MILP of* (4.1) *which we can divide as follows:*

$$\min\{c * x + g * y : A_1 * x + B_1 * y \leq h_1, A_2 * x + B_2 * y \leq h_2, x \in \mathbb{R}^n_{\geq 0}, y \in \mathbb{Z}^m_{\geq 0}\}$$

with the q_1 constraints $A_1 * x + B_1 * y \leq h_1$ being the "easy" and $A_2 * x + B_2 * y \leq h_2$ the q_2 many "hard" ones ($q_1 + q_2 = q$). Then the problem

$$z(\lambda) = \min\{c * x + g * y - \lambda * (h_2 - A_2 * x - B_2 * y) : (x, y) \in \tilde{X}\} \qquad (4.3)$$

with $\lambda = (\lambda_1, \ldots, \lambda_{q_2}) \geq 0$ and $\tilde{X} := A_1 * x + B_1 * y \leq h_1 (\subseteq \mathbb{R}^n_{\geq 0} \times \mathbb{Z}^m_{\geq 0})$ is called Lagrangian relaxation.

Proposition 4.2.10. *A Lagrangian relaxation is a relaxation as defined in 4.2.1.*

Proof. Since the feasible region of the Lagrangian relaxation is obtained from the original one by eliminating the difficult constraints, it is strictly greater. As for the two objective functions, over the whole region $X := A*x+B*y \leq h$ it holds that $c*x+g*y-\lambda*(h_2-A_2*x-B_2*y) \leq c*x+g*y \;\; \forall (x,y) \in X$ since the vector λ has only non-negative entries and surely in $A_2*x-B_2*y \leq h_2$ by definition of X. Thus the second term of the left-hand side of the inequality is always non-negative. □

In Definition 4.2.9 we have seen that the solution of the Lagrangian relaxation depends on the vector λ. Since the aim is to get a lower bound that is as tight as possible, what we are actually looking for is the value of λ so that $z(\lambda)$ is maximized.

Definition 4.2.11. *The optimization problem $L = max_{\lambda \geq 0}\{z(\lambda)\}$ is called Lagrangian Dual Problem.*

Now the general idea of the Lagrangian dual should be clear: The constraints are divided into "easy" and "hard" ones, the Lagrangian relaxation is set up and at the final step the one value of λ is determined that yields the tightest lower bound. There arise three questions so far open from this procedure:

(i) How strong is the Lagrangian relaxation at all?

(ii) How can the Lagrangian Dual Problem be solved in practice?

(iii) What are the "difficult" constraints to be added as a penalty term to the objective function?

In order to address (i) and (ii), we first need a theorem from which the answers follow.

Theorem 4.2.12. *$L = min\{c*x+g*y : A_2*x-B_2*y \leq h_2, (x,y) \in conv(\tilde{X})\}$ with $\tilde{X}$ again defined as in* (4.3)*.*

Proof. [21], pp.327 □

This theorem is of crucial importance: It tells us that the Lagrangian Dual Problem can be solved by solving the problem in 4.2.12. But if we take a close look at this theorem, the formulation given there is nothing else than an LP. So it provides us an answer for question (ii): One way (of severals) to solve the Lagrangian Dual is given by searching for the solution of a linear problem. As a consequence of this theorem (for the detailed derivation take a look at [21], pp.328) we get that the lower bound returned by the Lagrangian relaxation is at least as tight as the one from the linear relaxation and thus answers (i).

For point (iii) the decision will be a trade-off between the tightness of the resulting lower bound (this information again is contained in the powerful result 4.2.12), the simplicity of solving the Lagrangian relaxation (problem specific) and the Lagrangian Dual (depends on the number of dual variables λ_i).

Example 4.2.13. *Let us come back to the model presented in Section 3.1. We want to utilize it as an example to illustrate the different kinds of relaxations we have discussed in this thesis:*

- *Elimination: Deleting all the constraints but* (3.2) *and* (3.14) *would be possible. In practical terms, this would mean that the rolling time is minimized with the only restriction that each aircraft has to be assigned to exactly one position. The solution achieved is easily guessable: for each aircraft, the position with the minimum distance to the active runway would be chosen - supposing all aircrafts arrive and depart from the same runway, this would mean the same parking position is assigned to every aircraft. Thus, the solution is clearly not feasible for the original problem and the lower bound would be strictly smaller than the original (unknown) objective value.*
- *Linear relaxation: Now we need to relax all the constraints that impose variables to be integer. In this case, the equations* (3.14) *and* (3.15) *shall be reformulated as*

$$x_{p,a} \in [0,1] \qquad \forall\ p \in P, a \in A$$

and

$$y_{a,b}, z_{a,b}, w_{a,b} \in [0,1] \qquad \forall\ a,b \in A, a \neq b$$

We will see in Section 4.4 how it shall be dealt with fractional solutions (that are feasible only in the linear relaxation) for former integer variables.

- *Lagrangian relaxation: [29] (page 180) suggests that* (3.2) *is a good candidate to be eliminated from the constraints and added to the objective function:*

$$z(\lambda) = \min_{x_{\cdot,\cdot}, t_{\cdot,2}} \sum_{r \in R} \sum_{y \in Y} \sum_{p \in P} (d\_o_{y,r,p} * (\sum_{m_1 \in N \cup O} u\_o_{m_1,r} * typ_{y,m_1} * x_{p,m_1}))$$

$$+ \sum_{m_2 \in N \cup I} (t_{m_2,2} - t_{m_2,1}) - \lambda * (1 - \sum_{p \in P} x_{p,a})$$

The feasible region is defined by the constraints (3.3) - (3.15).

As a final remark in this section, we want to spend some words on a possibility to derive upper bounds to a MILP as in (4.1). Based on the linear relaxation presented above, a heuristics to find a feasible solution can be constructed.

Algorithm 1 Dive-and-Fix

```
1. Initialize: Let $(x^\star, y^\star)$ be the solution of the linear relaxation of a MILP where the integer variables are $\in \{0,1\}$
2. $G := \{1 \leq i \leq m : y_i^\star \notin \{0,1\}\}$.
3. As long as $G \neq \emptyset$ go to 4; else: go to 8
4. Find the variable $\in G$ whose value is closest to integer, denote it by $y_{int}^\star$
5. If $y_{int}^\star \leq 0.5$, then $y_{int}^\star \leftarrow 0$; else: $y_{int}^\star \leftarrow 1$ (choose the integer value that is closer to the fractional value)
6. Solve the new $LP_{y_{int}^\star}$ updated with the new value of $y_{int}^\star$ fixed
7. If $LP_{y_{int}^\star}$ has a solution, go back to 2; else: abort for infeasibility
8. Terminate: $G = \emptyset$, and the latest $LP_{y_{int}^\star}$ solution is feasible for the MILP
```

4.3 Cutting planes

We have discussed in Section 4.1 that a MILP over a feasible region X is equivalent to an LP over $\operatorname{conv}(X)$. We have furthermore stated that in the majority of applications it is extremely challenging or even impossible to find $\operatorname{conv}(X)$ in a reasonable amount of time, and even if we can, the enormous number of constraints would make it very hard to solve the LP. Nevertheless, we got to know the concept of better formulations, and following this idea leads naturally to the question if we can at least approximate the convex hull by finding better formulations than the original one.

Definition 4.3.1. *Let $X \subseteq \mathbb{R}^n$. If there exists a vector $\pi \in \mathbb{R}^n$ and a scalar $\pi_0 \in \mathbb{R}$ such that $\pi * x \leq \pi_0 \ \forall x \in X$, the inequality is said to be valid (for X).*

Example 4.3.2. *Let a MILP as in* (4.1) *be given, so: $X = \{(x, y) \in \mathbb{R}^n \times \mathbb{Z}^m : A * x + B * y \leq h\}$, conv(X) $= \{(x, y) \in \mathbb{R}^{n+m} : \tilde{A} * x + \tilde{B} * y \leq \tilde{h}\}$. The inequalities $a_i * x + b_i * y \leq h$ and $\tilde{a}_i * x + \tilde{b_i} * y \leq \tilde{h}$ are obviously valid for X.*

So far, we have not gained a lot, because if only the constraints were valid inequalities, we would have not more than a new fancy definition. Luckily, we will see that we have indeed way more:

Example 4.3.3. *Equation* (3.2) *can be split into two inequalities. Let us take a look at one of them, $\sum_{p \in P} x_{p,a} \leq 1 \ \forall a \in A$. If we examine the whole problem, we understand that we can derive new valid inequalities that are not implied by* (3.2).

$$x_{p,a} \leq 1 \qquad \forall p \in P, \forall a \in A$$

$$x_{p_1,a} + x_{p_2,a} \leq 1 \qquad \forall a \in A, p_1 \neq p_2 \in P$$

These two families are only some examples of the vast number of valid inequalities which arise from the original formulation.

The fact that the number of valid inequalities to add can be enormous will be the main motivation for the cutting plane method, but before we shall discuss how valid inequalities can actually be generated in a more systematic way:

Proposition 4.3.4. *The inequality $\pi * x \leq \pi_0$ is valid for the polyhedron $P := \{x \in \mathbb{R}^n : A * x \leq b\}$ if and only if there exists a vector $u \in \mathbb{R}^n$ such that $A^\top * u \geq \pi$ and $u * b \leq \pi_0$.*

Proof. From the strong duality property of linear programming we know that

$$\max\{\pi * x : A * x \leq b\} = \min\{u * b : A^\top * u \geq \pi\} = z,$$

if the primal or dual problem has a solution. We now can conclude that the right-hand side is $\leq \pi_0$ iff the left-hand side is, and so that $\pi * x \leq \pi_0 \ \forall x \in P \Leftrightarrow \exists u \in \mathbb{R}^n : u * b \leq \pi_0$. $\square$

With this theorem, theoretically we could determine all valid inequalities for a polyhedron P, derive at a new P' and solve the MILP with this new formulation. Since this is done before using a standard solution procedure, we speak of *preprocessing*, which means nothing else than adding valid inequalities a priori. Unfortunately, the countless number of such generated inequalities makes it very hard to solve linearly relaxed problems and to use a standard branch-and-bound approach.

Indeed, it is actually not needed to add all valid inequalities we can find. In the beginning of this section we had stated that we aim for an approximation of conv(X), but we shall be more precise: What we need is a good approximation of conv(X) around the solution of the MILP, i.e., $(x^\star,y^\star)$ of the MILP should be an extreme point of the polyhedron from the linear relaxation, whereas we do not care about other edges being integer when we already know - the way of "knowing this already" will be explained in Section 4.4 - that the optimal solution will not be there.

Definition 4.3.5. *Denote as usual by X the feasible region of a MILP and by P the polyhedron of the linear relaxation. For $x' \in P$, $x' \notin X$ we define a cutting plane as an inequality $\pi * x \leq \pi_0$ that satisfies:*

- $\pi * x \leq \pi_0 \ \forall x \in X$, *thus it is a valid inequality*
- $\pi * x' > \pi_0$

In Figure 4.3 this definition is illustrated and it can be seen graphically what is the crucial importance of a cutting plane: It not only is an arbitrary valid inequality, but it is one that satisfies the condition that it is *not* valid for the linear solution.

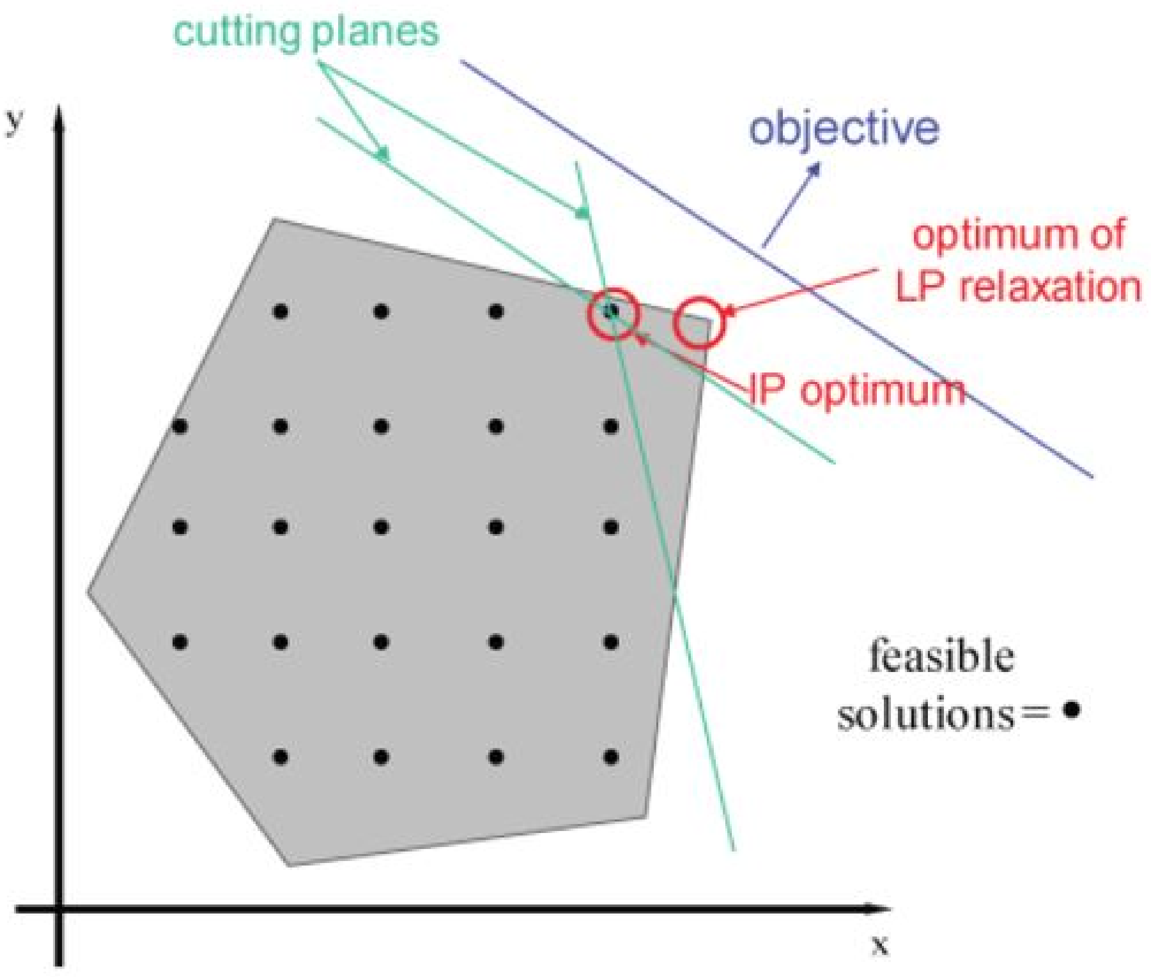

Figure 4.3: Two possible cutting planes that cut off the current LP solution

Easily, we can now present the cutting-plane-algorithm:

Algorithm 2 Cutting plane

```
1. Initialize: For the feasible region of a MILP as in (4.1) be $P_0 = P := \{(x,y) \in \mathbb{R}^{n+m} : A * x + B * y \leq h\}$
2. The linear relaxation problem is given by $\min\{c*x+g*y : (x,y) \in P\}$; solve it and let $(\tilde{x}^\star, \tilde{y}^\star)$ denote the solution.
3. If $(\tilde{x}^\star, \tilde{y}^\star) \in (\mathbb{R}^n \times \mathbb{Z}^m)$: $(\tilde{x}^\star, \tilde{y}^\star) = (x^\star, y^\star)$ is the optimal solution of the MILP, so terminate; else: Find a cutting plane that cuts off the current linear solution from $X := P \cap (\mathbb{R}^n \times \mathbb{Z}^m)$
    a) If such a cutting plane $\pi * (x,y) \leq \pi_0 \; \forall (x,y) \in X, \pi * (\tilde{x}^\star, \tilde{y}^\star) > \pi_0$ is found, update $P = P \cap \{(x,y) \in \mathbb{R}^{n+m} : \pi * (x,y) \leq \pi_0\}$ and go back to 2.
    b) Else: $(\tilde{x}^\star, \tilde{y}^\star)$ is not feasible for the MILP, but there could not be found a cutting plane. The algorithm terminates. We gained a better formulation since $P \subset P_0$.
```

Taking a closer glance at the presented algorithm, we see that step 2 has been discussed previously. What remained open so far is the actual implementation of step 3, thus the core element of the cutting plane method. While Theorem 4.3.4 gives us a way to determine valid inequalities for a polyhedron, we are now in the situation to determine a valid inequality for the mixed integer region X and not for a polyhedron anymore (even stronger: we want to cut off a part of the polyhedron). The rest of this section will thus be dedicated to a more technical part, that first treats the issue of deriving valid inequalities for MILPs and then of providing a procedure to generate cutting planes - so what we need in step 3.

Let us start with the so-called *mixed integer rounding* as a basic principle to generate valid inequalities for MILPs of a special form.

Remark 4.3.6. *We assume for the following results that the notation* $\lfloor h \rfloor := \max\{m \in \mathbb{Z} : m \leq h\}$ *and* $\lceil h \rceil := \min\{m \in \mathbb{Z} : m \geq h\}$ *is known to the reader.*

Theorem 4.3.7. *Let* $X := \{(x,y) : x \in \mathbb{R}_{\geq 0}, y \in \mathbb{Z}, y - x \leq h\}$. *The inequality*

$$y - \frac{1}{1 - f(h)} * x \leq \lfloor h \rfloor \tag{4.4}$$

is valid for conv(X), where $f(h) := h - \lfloor h \rfloor$.

Before we give the proof of this result, it is useful to spend a sentence on the meaning of it: The polyhedron of the linear relaxation (obviously given by $P = \{(x, y) \in \mathbb{R}^2 : x \geq 0, y - x \leq h\}$) can be transformed to the convex hull of X by adding just the valid inequality suggested above. Graphically (Figure 4.4) this means that this inequality cuts off from P the region that is "superfluous" - in the sense of preventing P from being the smallest convex set containing X.

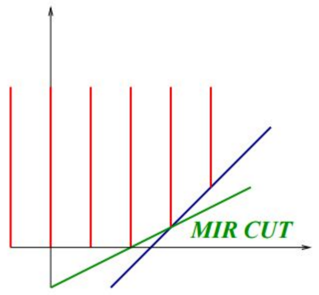

Figure 4.4: The mixed integer rounding principle for $h = \frac{5}{2}$

Proof of Theorem 4.3.7. If $h \in \mathbb{Z}$, the result is trivial. So suppose $h \notin \mathbb{Z}$: We divide the feasible region in two disjoint parts: $X_1 := X \cap \{(x, y) : y \leq \lfloor h \rfloor\}, X_2 := X \cap \{(x, y) : y \geq \lceil h \rceil\}$.

- For X_1: Take the valid inequalities $x \geq 0$ and $y - \lfloor h \rfloor \leq 0$. So, $y - \lfloor h \rfloor$ is non-positive, thus multiplying with the strictly positive term $(1 - f(h))$ does not change the sign, and it holds that $(y - \lfloor h \rfloor) * (1 - f(h)) \leq 0 \leq x$.

- For X_2: Take the valid inequalities $y - x \leq h$ and $y \geq \lceil h \rceil = \lfloor h \rfloor + 1$. Since $y \in \mathbb{Z}$, we can rewrite the latter as $-(y - \lfloor h \rfloor) \leq -1$. Multiplying it by $f(h)$ and adding the 2 inequalities gives the result $y * (1 - f(h)) - x \leq \lfloor h \rfloor * (1 - f(h))$.

(4.4) thus being a valid inequality for both conv(X_1) and conv(X_2) and $X = X_1 \dot{\cup} X_2$, it is also valid for conv(X). $\square$

Let us shortly recapitulate what we have: We know how to find a valid inequality for a MILP region with one continuous and one integer variable. In order to be able to carry out the cutting plane algorithm we still need one more step: We need a description of valid inequalities for a generalized MILP ((4.1)) that also fulfills the cutting plane property for the current holding linear solution. All these properties are provided by the *Gomory mixed integer cutting planes.*

In first place, let us rewrite (4.1) in the standard form:

$$\begin{aligned} \min_{x,y} \ & c * x + g * y \\ A * x + B * y + s &= h \\ x \in \mathbb{R}^n_{\geq 0}, y \in \mathbb{Z}^m_{\geq 0}, s &\in \mathbb{R}^q_{\geq 0} \end{aligned} \tag{4.5}$$

The optimal solution of the linear relaxation is given by the basic variables $(\tilde{x}^\star, \tilde{y}^\star)$. If the vector $\tilde{y}^\star \in \mathbb{Z}^m$, we would be done. Otherwise, there exists an entry $\tilde{y}_i^\star \notin \mathbb{Z}, i \in B$, where B denotes the basis. According to the simplex tableau, in row t one can thus write:

$$\tilde{y}_i + \sum_{j \in N_1} d_{t,j} * \tilde{x}_j + \sum_{j \in N_2} d_{t,j} * \tilde{y}_j = e_t,$$

where N_1 and N_2 denote the sets of non-basic continuous and integer variables, respectively, with the appropriate coefficients $d_{t,i}$ and $e_t \notin \mathbb{Z}$.

Definition 4.3.8. *The inequality*

$$\tilde{y}_i + \sum_{j \in N_2} \left(\lfloor d_{t,j} \rfloor + \frac{max\{0, f(d_{t,j}) - f(e_t)\}}{1 - f(e_t)} \right) \tilde{y}_j \leq \lfloor e_t \rfloor + \sum_{j \in N_1} \left(\frac{max\{0, -(d_{t,j})\}}{1 - f(e_t)} \right) \tilde{x}_j$$

is called Gomory mixed integer (GMI) inequality, where $f(d_{t,j})$ and $f(e_t)$ are defined as in Theorem 4.3.7.

Theorem 4.3.9. *The GMI inequality is valid for* (4.5) *and is a cutting plane with respect to the optimal solution of the linear relaxation* $(\tilde{x}^\star, \tilde{y}^\star)$.

Proof. Algebraic transformations and applications of Theorem 4.3.7, for details see [6] (pp.30) □

Although there is no proof that the method with GMI cuts does terminate in finite time, in practice it has proven to be very efficient ([6]).

4.4 Branch-and-Cut

As a motivation for the Branch-and-Cut algorithm we shall shortly summarize the methods described so far and how they are connected:

- For the lower bound we fall back upon relaxation methods and can use the powerful theory of cutting planes whenever linear relaxations are used.
- For the upper bound we have heuristics as for example in Algorithm 1.
- We can check if the difference between lower and upper bound is $\leq \epsilon$ and if so, we were successful in finding a sufficiently close approximation for the solution.

Theoretically, we thus have a toolbox at hand to solve (M)ILPs. In practice, (M)ILPs are at least NP-hard ([8], page 4) and thus we need to embed this toolbox into an algorithm that breaks down the given (M)ILP into many - easily solvable - subproblems and assembles the gained information to extract from it the solution of the original problem. A basis for the well-known Branch-and-Bound and the here discussed extension Branch-and-Cut is formed by a simple statement:

Proposition 4.4.1. *Suppose we have the usual MILP as in* (4.1). *Denote the feasible region once again by* X. *We decompose as follows:* $X = X_1 \cup \cdots \cup X_p$, *where* $X_i \cap X_j = \emptyset \; \forall i \neq j$ *is recommendable, but not mandatory. Let further be* $z = min\{c * x + g * y : (x, y) \in X\}$ $(z_i = min\{c * x + g * y : (x, y) \in X_i\})$ *the* *minimization problem over X (X_i). Then we get:* $\overline{z} = min_{1\leq k\leq p}\{\overline{z_k}\} \geq z \geq min_{1\leq k\leq p}\{\underline{z_k}\} = \underline{z}$.

Proof. $\overline{z_k}$ is an upper bound for the optimal solution $z_k^\star$ over the region X_k. The union of all regions $X_k, k = 1, \dots, p$ gives X. $min_{1\leq k\leq p}\{\overline{z_k}\} \geq min_{1\leq k\leq p}\{z_k^\star\} = z^\star$ and thus the minimum upper bound of all the subregions is for sure an upper bound to the original problem. The argumentation for the lower bound is analogous. □

The strategy will now be to find lower and upper bounds for the many small subproblems. With Proposition 4.4.1 we know a way to use them as information for the original problem. Now the idea of what will be called Branch-and-Cut algorithm shall be described in a chronological way:

For the problem (4.1) we solve its linear relaxation, where basic preprocessing can be used (redundant constraints are deleted, bounds are improved, see [27]). The solution we will denote as usual by $(\tilde{x}^\star, \tilde{y}^\star)$. Given that $\tilde{y}^\star \notin \mathbb{Z}^m$, we have not found an optimal solution for the MILP. We choose one entry $\tilde{y}_i^\star$ of the vector that is not integer (a popular strategy is for example to take the entry whose fractional part is closest to 0.5) and can divide the MILP as follows:

$$\begin{aligned} z_1 &= min\{c * x + g * y : A * x + B * y \leq h, y_i \leq \lfloor \tilde{y}_i^\star \rfloor, y \in \mathbb{Z}^m\} \\ z_2 &= min\{c * x + g * y : A * x + B * y \leq h, y_i \geq \lceil \tilde{y}_i^\star \rceil, y \in \mathbb{Z}^m\} \end{aligned} \tag{4.6}$$

As we can see, we have created two disjunct regions whose union is the original feasible region. If we knew z_1 and z_2, i.e., the optimal solutions of the two minor MILPs, we would just need to take the smaller one and would have the overall solution, since $z = min\{z_1, z_2\}$. In general, we will not know the MILP solutions, but nothing prevents us from starting a recursive process: We solve the linear relaxations of the subproblems, and branch again as described above. In the case of our model, one shall notice a particularity that does not hold for general MILPs: Since all the integer variables are binary, whenever we branch we create two subproblems where in each one the value of the branching variable is fixed. Once used as a branching variable, it is assured that this same variable will never be fractional again and thus there will not be a branching step based upon this variable again throughout the algorithm.

Now we want to examine how exactly the linear relaxation of a subproblem is solved: An asset of Branch-and-Cut is to provide tight lower bounds. How can this be achieved? We know the answer from Section 4.3: We can try to find a cutting plane for the solution of the linear relaxation, and iterate this procedure in this same node to always tighten the lower bound! The great advantage of this approach is that tight lower bounds can help to avoid the need to explore huge parts of the original feasible region. To see this, the following example can be insightful.

Example 4.4.2. *Suppose we have done the first branching step for a MILP and are thus in the situation of* (4.6)*. With the use of the cutting plane method in each of the two nodes we derive lower bounds* $\underline{z_1} = 0.32$ *and* $\underline{z_2} = 5.2$*. With a primal heuristic (pay attention: if we use for example the Dive-and-Fix-algorithm, we can profit of the cutting plane method there as well) we derive upper bounds* $\overline{z_1} = 4$ *and* $\overline{z_2} = 8$*. At this point, we can prune the node of region* X_2*, since any feasible solution found there will be* ≥ 5.2*, but we have found a feasible solution that lies within the region* X_1 *with value* 4 *and thus the optimal solution of the original problem will yield an objective value* ≤ 4*.*

In a MILP where all integers are binary, in the worst case we have m levels of branches and at each level 2^m many nodes. For all of them, we have to solve a linear relaxation problem. If we invest a lot in a cutting plane method at each node, the bounds get tight and this might lead to a significant lower number of nodes we have to examine. We face a trade-off situation here: while it is computationally cheap to derive weak bounds at a node, generating cutting planes at every node that is examined slows down the algorithm. On the other hand, the drawback on renouncing on the generation of cutting planes is that potentially many more nodes have to be examined.

If the whole tree has been treated (i.e., the nodes where either pruned or their lower and upper bounds were saved), as a last step, and again according to 4.4.1, we take the incumbent lower and upper bound and have found thus two bounds for the optimal solution of the original problem.

What has been outlined in the last paragraphs, can be found now in a compact form:

Algorithm 3 Branch-and-Cut

```
1. For a given MILP as in (4.1) the integrality condition for $y$ is dropped
   and the linear relaxation is solved with solution $\underline{z}$ (global preprocessing
   can be performed). Afterwards a feasible upper bound $\overline{z}$ is determined
   (using heuristics). If the LP is infeasible: terminate, the MILP is
   infeasible; else: go to 2
2. Initialize: $\underline{z_{act}} \leftarrow \underline{z}$, $\overline{z_{act}} \leftarrow \overline{z}$, $N = \emptyset$. If the bounds coincide, terminate
   (optimal solution for MILP found); else: go to 3.
3. At least one variable that is supposed to be integer is for now fractional.
   Branch on one of these variables according to (4.6). These two new
   nodes are added to the set of nodes $N$.
4. While $N \neq \emptyset$:
   a) Pick a node $j \in N$ and update: $N := N \setminus \{j\}$.
   b) Relax the corresponding MILP to an LP to derive $\underline{z_j}$ and find a
      feasible solution $\overline{z_j}$:
      i) If $\underline{z_j} \geq \overline{z_{act}}$ or the LP is infeasible: Prune node and go to 4;
         else: go to 4.b)ii) $\veebar$ 4.b)iii)
      ii) Cut: Add cutting planes that cut off $\underline{z_j}$ and go to 4.b)
      iii) Branch: Check if $\underline{z_j} \geq \underline{z_{act}} \vee \overline{z_j} \leq \overline{z_{act}}$; if so, update: $\underline{z_{act}} \leftarrow$
           $\underline{z_j} \vee \overline{z_{act}} \leftarrow \overline{z_j}$; then, go to 3.
```

[14] mentions that the Branch-and-Cut-algorithm can be further improved by not only relying on linear relaxations, but that in some nodes it might be recommendable (especially if the subproblem has a form that the feasible region is described by many “easy” constraints and a few “difficult” ones) to use the Lagrangian relaxation.

At last we see that an algorithm to solve a MILP does not consist of *the* method, but it is rather an interaction of the methods described in this chapter. Needless to say, this is not an exhaustive description, but there are more approaches - just to mention one: column generation algorithms - that play a very important role in this field. Nevertheless, the author hopes that with the results presented here it can be understood what happens at the “guts” of a MILP solver such as SCIP, and if the reader shall be in the situation to get to choose, e.g, the branching strategy, the reader has now a solid base upon which this can be decided.

5 Solution

This chapter's content is bringing together the previous results from the Chapters 3 (where the model was formulated) and 4 (where the solution theory was developed). All the calculations in the following sections have been performed with the software system GAMS [1] (version 24.7.1) on a Lenovo ideapad 100-15IBY (processor: Intel ® Celeron ® CPU N2840 @ 2.16 GHz, operating system: Windows 10 Home, main memory: 4GB). In the option settings, the default solver for MILPs was set to SCIP [2].

5.1 Baseline model

We return to the model that consists of the equations (3.1) - (3.15). Solving this model in GAMS has caused several difficulties due to its dimension: While there are as many continuous variables $t_{a,2}$ as aircrafts, there exist $p * a$ many binary variables $x_{p,a}$ and $3 * a^2$ many of the auxiliary ones. It is easy to see that one more aircraft that shall be scheduled leads to a multiplication of the variables. When the model was first tried for a test set with only 5 aircrafts, the solution could be found in less than 1 second, but in the actual observation period we had almost 20 times more flight movements to handle. It is still useful to describe, which approaches have been tried until a solution could be found since it emphasizes the features to be paid attention to when using Branch-and-Cut in practice.

In the first attempt, the default settings of SCIP were left unchanged. SCIP has a huge variety of parameters that determine how much effort shall be spent on the different parts of the algorithms as pre-solving or heuristics to find tight

[1] General Algebraic Modeling System

[2] Solving Constraint Integer Programs, developed at Zuse Institute Berlin

upper bounds. Their precise descriptions and default values can be found in the solver documentary. In the standard setting, SCIP spent approximately 15 minutes on pre-solving and could reduce the model size from originally 37.621 variables and over 2 million constraints to over 13.000 variables and 205.000 constraints (of which the vast majority are of logical nature, i.e., similar to the equation block (3.6) - (3.12)). Subsequently, the branch-and-cut-algorithm was initiated, but apparently the dimension was still too big, since there have been performed only two branching steps in 12 hours and a lot of effort spent on solving the linear relaxations, so that the unsatisfying solution returned a gap between lower and feasible upper bound of 50% - even worse, the feasible solution had a higher rolling time than the one recorded for the conducted policy on the 16th of September. So when the solver was stopped by a user interrupt after 12 hours, not only did we have a big gap between the bounds, but furthermore a policy suggestion that would yield a longer rolling time than the known one.

The next idea (that at the end turned out to be the winning one) developed from the first failing try goes as follows: There shall be put a lot of emphasis on presolving in order to bring down as much as possible the number of variables and constraints, and once this is achieved, on each node, the effort on finding upper bounds shall be minimal (i.e., only very fast heuristics shall be used), while the hope is that the linear relaxations now will be solved easier for the reduced size. In this run, the time dedicated to presolving on the original model (the parent node) was more than 90 minutes, and it could be achieved to obtain a reduced model with only 5.550 variables and 100.000 constraints (less than half the size compared to the first run). Only then the first branching step was performed, but after 15 more minutes the solver returned two bounds with a relative gap of 14.12% only. In total, thus, a sufficiently high gap could be obtained after less than 2 hours by a user's change of the priorities - instead of interrupting with a bad result after 12 hours in the previous run. To achieve a relative gap equal to 10%, which is a usual termination criterion, the run continued for another 90 minutes with branch-and-cut-steps.

It shall be said that selecting certain settings or putting more effort in some parts of the algorithm than in others requires (in the author's experience) a good understanding of the model, the used solver and a feeling for where it is worth letting the solver perform exhaustive operations. In our case, this was achieved by taking a close look at the log file of the first, not successful run and the Table 5.1 from below. Nevertheless, probably it is just as much of an art as of science.

Feature	**Speedup factor**
Cuts	54
Presolving	11
Variable fixing	3
Heuristics	1.5

Table 5.1: Average speedup for specific feature (enabling vs. disabling it while all the others are active) [7]

Let us now come to the main result of this chapter: Is it possible to assign the aircrafts to positions in such a way that the rolling time is (significantly) lower than the one observed for the period of interest? The answer to this question is yes, it is possible, and we can give a quantitative formulation: At the last available step before interrupting, the gap was at 8%, and the best feasible solution yields a total rolling time that is $80,67\%$ of the actual one, and the lower bound is at 74.7% of the original time. In other words, there has been found a feasible assignment policy that could reduce the rolling time in the 3 hours taken into consideration by 20%, and in the best possible case there will be found a solution (given that the solver terminates only when the two bounds coincide) that reduces the rolling time even by one forth.

We have to still impose one restriction to the result just presented: There have been a few rules that apply when assigning parking positions that could not be taken into consideration in the solution process since the data did not allow to reconstruct, for which aircraft which rules apply. This makes the results from above a *lower* bound for the solution with these additional constraints - a lower bound in any case that looks promising and constitutes an encouraging sign for further examination.

5.2 Sensitivity analysis

5.2.1 Relax feasible region

We want to take up the discussion of previous section's last paragraph and tackle the question of how much it costs in terms of additional rolling time to account for constraints that go beyond the version including (3.2) - (3.15) only. The aim of solving a model with just these constraints consists of getting a feeling for the price to pay for satisfying requirements that are more than the absolute minimum.

The practical aspect of this exercise is nothing else than regarding the process from Section 2.1 in isolation, i.e., neglecting any necessity to align with landside processes and structural conditions. Also theoretically it is an interesting experiment, since we delete many constraints and enlarge the feasible region. The effects are twofold: On the one hand, constraints that are deleted make the structure of the problem easier (compare with relaxation by deleting, Chapter 4). On the other hand, some of the deleted constraints were of such nature that they fixed binary variables, and loosing them does require more branching steps. These opposing effects in mind, it will be insightful to observe how the run time now behaves in comparison with Section 5.1.

The solver settings where chosen again as in the successful run from before. After the exhaustive presolving on the parent node, the first branching step was performed after 194 minutes, and after 244 minutes two bounds were returned still with a pretty high gap of $25,81\%$. It could subsequently be reduced by one half (to $13,09\%$) when the run was manually stopped after 11 hours. This time, a feasible solution could be found that yields an overall rolling time that lies $8,89\%$ below the one calculated for the baseline model. In addition, we can be sure that the unknown optimal solution will in every case be better than the one from the baseline model, since the best feasible solution is lower than the lower bound for the baseline.

It does can be observed that the alignment with further processes has a negative impact on the rolling time that, in this case, is not huge, but not insignificant either, and this fact shall be kept in mind when evaluating whether or not to introduce further constraints that are not born out of technical needs.

5.2.2 Reduction of available positions

Another topic that shall be raised shortly is the consequence that a closure of certain positions can have. The question asked is again how expensive it is (in terms of additional rolling time) to provide a smaller number of positions to the aircrafts. In a certain sense, this issue tackles the capacity discussion in a reverse sense: Can the demand for parking positions still be satisfied and how much does the decrease in supply of positions enhance the total rolling time?

Whether or not the demand can still be satisfied is simply equivalent to the question, whether the feasible region of the model is still nonempty when certain members from the set of positions are canceled out. If the set is reduced too much, it will just not be possible to offer parking positions for all aircrafts in the period of observation. Here, we renounced on 10% of available positions, and chose the ones to be discarded equally over the blocks so that from each of them at most 2 positions are closed. All the constraints from the baseline model that regard the respective positions are obviously ruled out as well.

It turned out that with the choice of removed positions made, the increase in total rolling time amounted to $2,6\%$ (again, comparing the two best feasible solutions found). This relative moderate increase for a reduction in supply of 10% must not entice to the conclusion that positions can be reduced without having to face a big increase of rolling time, since the deleted positions could have just been “well” chosen (and with a different choice, the increase could be much more significant). However, at least it can be stated that in the period of examination, the number of positions was not a critical bottleneck to avoid an obstruction of operations.

One could argue that it would be a more realistic approach to let the number of positions unvaried, but increase the flight movements for a capacity examination. Although this reasoning has a point, it has to be remembered that increasing the dimension of the model further has a critical impact on the run time. Therefore, this can be a starting point that nevertheless has a practical relevance for itself, since it can be that positions are temporarily out of service.

6 Conclusion

In the previous five chapters, we have been following a red thread: we described a concrete problem from reality, translated it into the language of mathematics, outlined the theory of how such an optimization model can be solved, and finally the solution was determined with the help of a computer. In the aim of not deviating too much from the path taken, there was reduced space for some important notes, remarks and not directly related, but yet interesting results. Therefore, this last chapter will illuminate aspects on the right and left of the path that so far remained in the dark, namely practical suitability, critique and possibilities for further studies.

An issue that turned out throughout the work to be crucial concerns the data, which consist of two different categories: The data needed for the parameter values and the description of all conditions that need to be translated into constraints. They have in common that the more reliable they are, the higher will be the informative value of the model. Obviously, if about the rolling time between the runways and the positions we had no more information than the one provided in Table 2.1, the significance of the objective value would be much lower than if we have at hand the exact rolling time between all possible (runway r - position p)-combinations. Less exact parameter values lead to a falsified result, and this term is justified here, since it is really a wrong result from which no conclusions about the correct solution can be drawn. Fortunately, in this thesis, the parameter data was available in a good quality. Differently, with the data that is translated into constraints (for example, if position p_1 is occupied, p_2 must be free), we can obtain an expedient result the same, but we need to understand how to interpret it: Every constraint we do not take into consideration in the model can be regarded as a relaxation of the “full” model (precisely, a relaxation by deleting, see Chapter 4), and an incomplete model will thus yield a lower bound for the full model solution. This is what we obtained in Chapter 5, namely a lower bound, and we shall add that theoretically the model can be strengthened easily with more constraints, but it is also true that only the collection of all the restrictions and rules to obey is very time-consuming and has to be treated in a further study.

A second aspect that deserves particular consideration is how realistically the actual operations can be depicted by the model of Chapter 3. Let us recapitulate, which assumptions we made explicitly or implicitly when the model was formulated: Most importantly, we have been assuming that the arrival and departure times were fixed parameters. Indeed, this assumption made sense inasmuch as the day of interest was lying in the past and thus all times were known. However, in this thesis the aim was to give an estimation on the gap between the conducted and the optimal policy, while in order to be used in daily operations, it has to specify a policy for the future. Clearly, the arrival and departures times are known insofar as there exists a schedule, but obviously one should calculate with the actual times rather than the scheduled ones. To give the reader a feeling for the importance of this remark: On the 16th of September over one forth of inbound flights had an actual time that deviated for 15 minutes or more from the scheduled one. But this is impossible to achieve, so we somehow need to accumulate information to estimate realistic parameter values. One could thus think of taking a look at a long history of arrival times as one indicator (for a regular flight that has the same schedule every day this could be an approach) or even more sophisticated machine learning approaches to detect patterns that indicate periods with a high probability for delay.

Two further assumptions might be challenged: We assigned equal importance to each aircraft in the sense of simply minimizing the sum of the single rolling times. It might be justified or even desirable to discriminate here: For reasons of costumer relations (favorable treatment for airlines that handle many flights in Vienna), allowing passengers to catch a connecting flight or even offering privileged treatment in exchange for an extra fee. Unlike the stochastic problematic from above, this topic can be handled easily by adding new constraints after having categorized the single aircrafts according to the way they shall be treated.

Secondly, we have supposed the examined process is performed on three different kinds of “machines”, which are the three steps we called K in Chapter 3. One could argue why we restricted ourselves to just three steps, and indeed as a natural extension that enhances the reference to reality, we could for example add the take-off position as a natural bottleneck as a further step and thus account for possible waiting queues there. Nevertheless, it must not be forgotten that an additional step increases the number of variables tremendously and will thus have a negative impact on the running time of the program.

An issue that has been left out so far is the practicability of the model. Meant by that is - neglecting the topic of uncertain parameters in a forward directed model - the matter of *how* it can be used in the daily airport activity. A first idea could consist of assigning the positions once every day for the whole following one. We have seen in Chapter 5 that with the SCIP solver it takes around two hours to determine the schedule for the busy morning hours. The characteristic property of an NP-hard problem is that adding one more aircraft leads to an exponential increase in the run time. Therefore, to determine the run time of a program that describes not only a peak period, but a calendar day, a run with the data for the entire 16th of September would be needed (whereby an educated guess suggests that an acceptable time horizon will probably not be met). In case that a solution cannot be generated in a reasonable amount of time, there would be the option of setting a time limit and taking the current solution after the expiration of this term (with a branch-and-cut algorithm, the current gap between lower and upper bound does also provide an estimate of how much the current solution still differs from the unknown optimum). Another way to tackle this problematic issue can be to cut down the problem "schedule a whole calendar day" in various subproblems (periods of a few hours) and calculate their solutions separately, which requires a dynamic approach since solving to optimality the morning hours without taking into account that this is only a subproblem might generate a very bad starting formation for the following period. The reason why the run time is such a critical bottleneck is simply that the less in advance the schedule is determined, the better short-term external factors such as strikes, weather phenomena or a ban of positions can be taken into consideration.

It has been emphasized intensively in this work that the model derives from job scheduling. It can be thus asked if the model as presented could be used only and exclusively for the Airport Wien-Schwechat or for other airports as well or maybe even for different areas. As for the first question, it can be said that there is no reason preventing the user to customize the model to any airport of choice, as long as the data is available in sufficient quality to calibrate the parameter values and to translate it into new constraints that will differ for every airport. In a broader sense, similar models could be proposed for every sequence of processes in production, operations or traffic which involves unproductive waiting time between two steps that shall be minimized. In the author's personal view, for example, a parallel can be drawn to the issue of smart parking ([16]) that is currently of high interest for profit-orientated companies. Up to 250 million USD are about to be invested in the area of research until 2019, and with a MILP approach similar to the one of Chapter 3, a quantitative estimate could be given for how much gain in efficiency an optimal parking policy could generate.

Having discussed the practical aspect of this thesis so far, we shall not end without spending a few lines on mixed integer linear programs in general. For that purpose, taking a glance back in history teaches us a lot: The development of algorithms and the vast improvement of hardware made it possible that a MILP whose duration of the determination of the optimal solution would have been 124 years back in 1988 can by now be solved in less than one second ([18])! In [18] we can read furthermore that despite there is "still plenty of room for MI[L]P actually" (Bob Bixby, page 55), academia cannot compete with commercial solvers (indeed, the best commercial solvers are 6 times faster than the best freely available ones, see page 6, and GUROBI claims in a self-description to be 9 times faster than SCIP) and has thus turned their interest lately more to MINLP, i.e., mixed integer non-linear programs, following Hotelling's famous quote that "[...] we all know the world is nonlinear" (page 46). But does this comment hold true for scheduling problems? In fact, the huge majority of them are modeled as MILP (both for the reason that their nature is linear as for the fact that the world of MILP is much more explored), but there are some approaches using MINLPs for specific subproblems in the world of scheduling as it can be read in [15].

List of Figures

Bibliography

[1] Tobias Achterberg. “Constraint Integer Programming.” PhD thesis. Technische Universität Berlin, 2007 (cit. on p. 24).

[2] Flughafen Wien AG. *Aviation facts*. Available online at `http://www.viennaairport.com/en/business__partner/aviation/aviation_facts`; Access date: 2017-05-29 (cit. on pp. 9, 10, 14).

[3] Flughafen Wien AG. *Geschäftsbericht 2016*. 2016 (cit. on pp. 3, 5, 11).

[4] Flughafen Wien AG. *Zukunft Flughafen Wien 3. Piste*. Available online at `http://www.viennaairport.com/unternehmen/flughafen_wien_ag/3_piste`; Access date: 2017-05-29. 2017 (cit. on p. 9).

[5] Pressestelle Flughafen Wien AG. *Verkehrsentwicklung im September 2016: Flughafen-Wien-Gruppe verzeichnete 2,9 Mio. Passagiere*. Available online at `https://www.viennaairport.com/unternehmen/investor_relations/news/verkehrsergebnisse?news_beitrag_id=14761742084121`; Access date: 2017-05-24. 2016 (cit. on p. 5).

[6] Edoardo Amaldi. *3.7 Cutting plane methods*. Available online at `http://home.deib.polimi.it/amaldi/SlidesOPT-16-17/IP-cutting-planes-I-16-17.pdf`; Access date: 2017-07-05. 2016 (cit. on p. 41).

[7] Edoardo Amaldi. *3.9 Branch and Cut*. Available online at `http://home.deib.polimi.it/amaldi/SlidesOPT-16-17/IP-Branch-and-Cut-16-17.pdf`; Access date: 2017-07-01. 2016 (cit. on p. 48).

[8] Edoardo Amaldi. *Chapter 3: Discrete Optimization - Integer Programming*. Available online at `http://home.deib.polimi.it/amaldi/SlidesOPT-16-17/IP-models-16-17.pdf`; Access date: 2017-06-09. 2016 (cit. on pp. 16, 41).

[9] Cynthia Barnhart, Peter Belobaba, and Amedeo R. Odoni. "Applications of Operations Research in the Air Transport Industry." In: *Transportation Science* 37.4 (2003), pp. 368–391 (cit. on p. 12).

[10] Lucio Bianco, Paolo Dell'Olmo, and Stefano Giordani. "Coordination of traffic flows in the TMA." In: *New Concepts and Methods in Air Traffic Management.* Ed. by Lucio Bianco, Paolo Dell'Olmo, and Amedeo Odoni. Berlin, Germany: Springer-Verlag, 2001, pp. 95–124 (cit. on p. 13).

[11] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. "Polyhedral Approaches to Mixed Integer Linear Programming." In: *50 Years of Integer Programming 1958-2008.* Ed. by Michael Jünger et al. Berlin, Germany: Springer-Verlag, 2009, pp. 343–385 (cit. on p. 27).

[12] Roger Dear. *The dynamic scheduling of aircraft in the near-terminal area.* Cambridge, Massachusetts: Flight Transportation Laboratory, MIT, 1976 (cit. on p. 12).

[13] EUROCONTROL. *Challenges of Growth 2013. Task 4: European Air Traffic in 2035.* 2013 (cit. on pp. 2, 3).

[14] Marshall L. Fisher. "The Lagrangian Relaxation Method for Solving Integer Programming Problems." In: *Management Science* 50.12 (2004), pp. 1861–1871 (cit. on p. 44).

[15] Ignacio Grossmann. *Overview of Mixed-integer Nonlinear Programming.* Available online at `http://egon.cheme.cmu.edu/ewo/docs/EWOMINLPGrossmann.pdf`; Access date: 2017-07-09. 2016 (cit. on p. 55).

[16] Devavrat Kulkarni. *Parking in the smart city.* Available online at `https://www.ibm.com/blogs/internet-of-things/iot-smart-parking/`; Access date: 2017-07-11. 2017 (cit. on p. 55).

[17] Peter van Leeuwen. *CAED D2: Modelling the Turnaround Process.* National Aerospace Laboratory NLR, Technical University Delft, 2007 (cit. on p. 12).

[18] Jeff Linderoth. *Mixed-Integer (Linear) Programming: Recent Advances, and Future Research Directions.* Available online at `http://www.focapo-cpc.org/pdf/Linderoth.pdf`; Access date: 2017-07-09. 2017 (cit. on p. 55).

[19] Paola de Mascio, Lorenzo Domenichini, and Alessandro Ranzo. *Infrastrutture aeroportuali.* 00184 Roma - Via della Polveriera, 15: Edizioni Ingegneria 2000, 2009 (cit. on pp. 4, 5, 8).

[20] Peter Maurer. *Luftverkehrsmanagement: Basiswissen.* 4th ed. 81671 München - Rosenheimer Straße, 145: Oldenbourg, 2007 (cit. on pp. 3, 8).

[21] George Nemhauser and Laurence Wolsey. *Integer and Combinatorial Optimization.* New York: Wiley-Interscience, 1988 (cit. on pp. 32, 33).

[22] Richard de Neufville and Amedeo R. Odoni. *Airport systems: Planning, Design and Management.* New York: McGraw Hill Education, 2013 (cit. on p. 14).

[23] Civil Aerodrome Operator. *AIRPORT CHARGES REGULATIONS.* Vienna Airport, 2016 (cit. on p. 11).

[24] Bundesverwaltungsgericht Republik Österreich. *Dritte Piste des Flughafens Wien-Schwechat darf nicht gebaut werden.* Available online at `https://www.bvwg.gv.at/presse/dritte_piste_des_flughafens_wien.html`; Access date: 2017-05-24. 2017 (cit. on p. 2).

[25] Manfred Padberg. *Linear Optimization and Extensions.* Berlin: Springer, 1995 (cit. on p. 27).

[26] Nicolas Pujet, Bertrand Delcaire, and Eric Feron. “Input-output modeling and control of the departure process of the congested airports.” In: AIAA Guidance, Navigation Control Conference. Portland, Oregon, USA, 1999 (cit. on pp. 13, 14).

[27] Martin Savelsbergh. “Preprocessing and probing techniques for mixed integer programming problems.” In: Memorandum COSOR; Vol. 9226. Eindhoven, Technische Universiteit Eindhoven, 1992 (cit. on p. 42).

[28] Paulo Wang and David Pitfield. “The derivation and analysis of the passenger peak hour: An empirical application for Brazil.” In: *Journal of Air Transport Management* 5.3 (1999), pp. 135–141 (cit. on p. 4).

[29] Laurence Wolsey. *Integer Programming.* 605 Third Avenue, New York: Wiley-Interscience, 1998 (cit. on pp. 24, 27, 34).