STOCHASTIC AIRPORT GATE ASSIGNMENT PROBLEM

by

MERVE ŞEKER

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

Sabancı University
August 2010

STOCHASTIC AIRPORT GATE ASSIGNMENT PROBLEM

APPROVED BY

Assist. Prof. Nilay Noyan (Thesis Supervisor)	.....
Prof. Gündüz Ulusoy	.....
Assist. Prof. Kerem Bülbül	.....
Assist. Prof. Güvenç Şahin	.....
Assoc. Prof. Orhan Feyzioğlu	.....

DATE OF APPROVAL: .....

to my family

Acknowledgments

I would like to express my deepest gratitude to my thesis advisor Assist. Prof. Nilay Noyan for her invaluable supervision and guidance throughout my thesis project. I would also want to thank my thesis committee members, Prof. Gündüz Ulusoy, Assist. Prof. Kerem Bülbül, Assist. Prof. Güvenç Şahin, and Assoc. Prof. Orhan Feyzioğlu.

I am grateful to all my friends for their caring and support. Very special thanks to my dear friends Elif Özdemir, Gizem Kılıçaslan, Nimet Aksoy, Özlem Çoban, and Semih Yalçındağ.

Lastly, I offer my special regards and blessings to my family for their concern, love and support they provided throughout my life.

STOCHASTIC AIRPORT GATE ASSIGNMENT PROBLEM

Merve Şeker

Industrial Engineering, Master of Science Thesis, 2010

Thesis Supervisor: Assist. Prof. Nilay Noyan

Keywords: airline transportation, gate assignment, random disruptions, stochastic arrival times, robustness, stochastic programming, tabu search

Abstract

The uncertainties inherent in the airport flight arrival and departure traffic may lead to the unavailability of gates when needed to accommodate scheduled flights. Mechanical failures, severe weather conditions, heavy traffic volume at the airport are some typical causes of the uncertainties in the input data. Incorporating such random disruptions is crucial in constructing effective flight-gate assignment plans. We consider the flight-gate assignment problem in the presence of uncertainty in arrival and departure times of the flights and represent the randomness associated with these uncertain parameters by a finite set of scenarios. Using the scenario-based approach, we develop new stochastic programming models incorporating alternate robustness measures to obtain assignments that would perform well under potential random disruptions. In particular, we focus on the number of conflicting flights, the buffer and idle times as robustness measures. Minimizing the expected variance of idle times or the expected semi-deviation of idle times from a buffer time value are some examples of the objectives that we incorporate in our models to appropriately distribute the idle times among gates, and by this way, to decrease the number of potential flight conflicts. The proposed stochastic optimization models are formulated as computationally expensive large-scale mixed-integer programming problems, which are hard to solve. In order to find good feasible solutions in reasonably short CPU times, we employ tabu search algorithms. We conduct an extensive computational study to analyze the proposed alternate formulations and show the computational effectiveness of the proposed solution methods.

RASSAL HAVAALANI KAPI ATAMA PROBLEMİ

Merve Şeker

Endüstri Mühendisliği, Yüksek Lisans Tezi, 2010

Tez Danışman: Yrd. Doç. Dr. Nilay Noyan

Anahtar Kelimeler: hava taşımacılığı, kapı atama, rassal aksaklıklar, rassal varış süreleri, dayanıklılık, rassal programlama, tabu arama

Özet

Uçuşların kalkış ve varış trafiğine özgü belirsizlikler uçuşların planlanan kapılara atanması gerektiğinde kapıların atamaya müsait olmamasına yol açabilmektedir. Teknik arızalar, uygunsuz hava koşulları, havaalanındaki trafik yoğunluğu girdi verisindeki belirsizliklerin tipik sebeplerinden bazılarıdır. Bu rassal aksaklıklar verimli uçuş-kapı atama planları oluşturulmasında büyük öneme sahiptir. Havaalanı kapı atama problemi uçuşların kalkış ve varış zamanlarındaki belirsizlikler gözönüne alınarak incelenmiştir ve bu belirsiz parametrelere dair rassallık bir senaryo kümesi ile ifade edilmiştir. Olası rassal aksaklıklara karşı dayanıklı bir atama elde etmek amacıyla senaryo tabanlı bir yaklaşım kullanılarak alternatif dayanıklılık ölçütlerini içeren yeni rassal programlama modelleri geliştirilmiştir. Özellikle odaklanılan dayanıklılık ölçütleri çakışan uçuş sayısı, tampon ve boş zamanlardır. Boş zamanların kapılar arası düzgün dağılımını sağlayıp olası çakışmaların önüne geçebilmek amacıyla atıl zamanların varyansının ya da atıl zamanların belirli bir tampon değerden toplam sapmalarının beklenen değerinin enküçüklenmesi önerilen modellerde hedeflenen amaçlara örnek olarak verilebilir. Önerilen rassal programlama modelleri çözümü zor olan büyük ölçekli karışık tamsayılı programlama olarak yazılmıştır. Daha kısa hesaplama süresi içerisinde olurlu ve iyi sonuçlar elde edebilmek amacıyla tabu arama sezgisel yöntemleri geliştirilmiştir. Önerilen alternatif formülasyonları analiz etmek ve önerilen çözüm yöntemlerinin hesaplama etkinliğini göstermek amacıyla kapsamlı bir sayısal çalışma yapılmıştır.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION AND MOTIVATION

Airport Gate Assignment Problem (AGAP) mainly focuses on assigning a given set of arriving flights to a given set of gates available at the airport under some constraints. Finding a reasonable flight-gate assignment plan is one of the major tasks in airline operations management and the increase in the volume of the air transport traffic has stimulated the importance and the complexity of the problem.

Here we elaborate on the common constraints and the objective functions considered in the gate assignment problems. The constraints are mainly classified as "strict" and "soft" constraints in the literature. Strict constraints are inherent to the problem and can be described as follows:

- each flight must be assigned to only one gate,
- no two conflicting flights are assigned to the same gate concurrently (referred to as "conflict constraints"). We say that a flight conflict occurs when two flights with overlapping ground times (gate occupation times) are allocated to the same gate.

Besides the strict constraints, additional restrictions related to airport facilities also need to be considered such as the assignment of specific airlines to the predetermined gates, the space restrictions related to the size of available gates and aircrafts, etc..

The problem tries to find an optimal assignment with respect to a specific objective function while satisfying the strict constraints and some soft constraints. Typical objectives specific to the problem can be classified under two main groups:

- Passenger-oriented objectives:
  - minimization of the total passenger walking distance,
  - minimization of the total connection times of the passenger (between gates and from apron to the terminal building), etc.

- Airport-oriented objectives:
  - minimization of the number of un-gated flights (flights assigned to the apron),
  - minimization of the aircraft towing procedures,
  - minimization of the baggage transport distances, etc.

Gate assignment problem in general is formulated for a given set of arrival and departure times of the flights. It is common to refer to the given set of arrival and departure times as the "planned schedule". However, in real life applications the arrival and departure times are not certain and it is crucial to take the uncertainties in these input parameters into the consideration. The assignment obtained based on a given deterministic (estimated) arrival and departure times may perform poorly when the realizations of the data deviate from the estimated input data. Mechanical failures, severe weather conditions, heavy traffic volume at the airport are some typical causes of the disruptions (early or late arrivals and departures) in input data. A delayed/early arrival or departure may cause "flight conflicts". Therefore, it is important to model the stochastic nature of the input data.

We represent the uncertain input data by random variables and we model the randomness by a finite set of scenarios. Note that a scenario represents a joint realization of the arrival and departure times of all the flights. Using the scenario-based approach we propose alternate stochastic programming models. The underlying idea of our proposed models is to allow infeasibilities in the stochastic version of the "conflict constraints", since obtaining a feasible assignment for all the scenarios would be quite conservative and unrealistic. However, since in real life environment the decision makers prefer to have a feasible assignment for the planned schedule, we ensure that the "conflict constraints" are satisfied for that given schedule.

We focus on alternate objective functions that involve some robustness measures such as the number of flight conflicts, buffer times and idle times. Our first stochastic optimization model tries to minimize the expected number of flight conflicts like the model proposed by Lim and Wang [21]. Lim and Wang use an unsupervised estimation function to estimate the probabilities of the flight conflicts based on a single planned schedule with deterministic arrival and departure times. Alternatively, we model the randomness in data using a scenario approach. This approach allows us to model the random deviations from the estimated input data directly. Additionally, in order to

take the effect of the variability in the number of conflicts into consideration, we extend the first model by considering a risk measure and using a mean-risk approach. Since the smaller values of the random number of conflicts are preferred, we consider the absolute semi-deviation as the risk measure and formulate a second model minimizing a combination of the associated expectation and the risk measure. We refer to those two models as "conflict based stochastic models", since the number of flight conflicts is incorporated as a robustness measure. In all of the proposed models this robustness measure based on the number of flight conflicts is considered as the primary one.

The vast majority of the gate assignment literature considers minimizing the passenger walking distance to improve the airport service satisfaction. Due to the nature of this objective, some gates receive high utilization. The basic problem, initially mentioned in [23], is that, even minor deviations in input data will disrupt those heavily utilized gates and so make the obtained assignment more prone to the disruptions. Bolat [5] suggests that distributing "idle times" uniformly among gates provides a robust assignment, where an "idle time" is a non-utilized time period between two successively assigned flights. The motivation is that distributing idle times uniformly helps us to decrease the probability that the delayed departure will be still earlier than the arrival of the next flight. Following this line of thought, Bolat [5] introduces two objectives: the minimization of the variance of idle times of all flights and the minimization of the range of idle times. For further discussion we refer the reader to [4–6]. The modeling approach in [5] is based on the fact that the flights can be sorted in ascending order of their arrival times. This approach is not valid while formulating a stochastic optimization model, since we cannot obtain such an ordering for the random arrival and departure times. In our setup, the idle times are random and each scenario may lead to a different ordering of arrival and departure times. Hence, it is not trivial to incorporate the idle times into a stochastic optimization model. Developing stochastic optimization models involving random idle times is one of our main contributions. We refer to such models as "idle time based models" and in the first one we minimize the expectation of the variance of idle times.

In order to avoid the problem of highly utilized gates and obtain robust assignments plans, Mangoubi and Mathaisel [23] propose a fixed "buffer time" between two continuous flights. Such a buffer time between two continuous flights allocated to the same gate may absorb the stochastic flight delays or earliness. In particular, buffer time is considered as a lower bound on each idle time value. However, in the stochastic

setup this approach requires to introduce lower bounds on the random idle times and it is not trivial. Our novel approach to model the random idle times allows us to also take the buffer time into consideration and leads to other novel stochastic optimization models. In particular, we propose two models involving the idle and buffer times as robustness measures. As a secondary objective, the first model minimizes the expected total semi-deviation of the idle times from the buffer time value, whereas the second one minimizes the expected number of the idle times that are below the buffer time.

We note that we do not directly model the traditional objectives used in the literature. However, they can be incorporated into our models as additional criteria. For example, an upper bound on the total passenger walking distance can be introduced to the models. Moreover, our models are aligned with some of those common objectives. For example, minimizing the number of conflicts also serves the objective of minimizing the number of un-gated flights.

1.1 Contributions

The main purpose of this study is to develop stochastic programming models to obtain assignments that would perform well under potential random disruptions. The contributions of this study can be summarized as follows:

- We develop new stochastic programming models for the airport gate assignment problem.
- We incorporate a risk measure on the random number of flight conflicts into a stochastic gate assignment model.
- Idle time and buffer time concepts are incorporated into stochastic gate assignment models as alternate robustness measures.
- We implement tabu search algorithms to solve the proposed models.
- We conduct an extensive computational study to analyze the proposed models involving alternate robustness measures.

1.2 Outline

Literature review is presented in Chapter 2. Chapter 3 presents the proposed stochastic programming models with alternate robustness measures. We first introduce the

conflict based stochastic programming models and then develop the formulations that incorporate the idle time and the buffer time concepts. We develop tabu search algorithms for the stochastic programming models proposed in Chapter 4. We present numerical results in Chapter 5 to demonstrate the computational efficiency of the implemented tabu search heuristics and the effectiveness of the proposed models, and to comparatively analyze the alternate models. Finally, in Chapter 6 we conclude and discuss future research directions.

CHAPTER 2

LITERATURE REVIEW

In this chapter, we present the existing modeling approaches and solution techniques that are used to formulate and solve the airport gate assignment problem. This problem has been widely studied and we refer the reader to Dorndorf et al. [11] for an extensive review. Research directions in the field of flight-gate assignment can be grouped under two main headings: mathematical programming techniques and rule based expert systems.

2.1 Mathematical Programming Techniques

Babic et al. [1] and Bihr [2] consider the gate assignment problem with the objective of minimizing the total passenger walking distance inside the terminal. They formulate the problem as a linear 0-1 integer program and use a branch-and-bound algorithm to solve the problem. Accordingly, Mangoubi and Mathaisel [23] present a linear relaxation of an integer program formulation and a greedy heuristic to solve the gate assignment problem. Their objective is also to minimize the total passenger walking distance within the terminal as in [1] and [2] but with an addition of transfer passengers. However, even if they take into account the transfer passengers, they do not provide a precise calculation of either the number of the transfer passengers or their walking distances.

Due to the complex nature of the problem, optimal algorithms (e.g. a branch-and-bound algorithm) have difficulty in solving the large-scale gate assignment problems. Thus, exact algorithms fail to provide an optimal solution within a reasonable computational time for large problem instances. Therefore, recent studies mainly focus on developing heuristic algorithms, which do not guarantee optimal solutions but provide near-optimal solutions in reasonable computational times.

Xu and Bailey [28] model the gate assignment problem as a quadratic assignment problem and reformulate it as a mixed 0-1 integer linear program. They consider

the objective of minimizing the total passenger connection time and propose a tabu search heuristic that incorporates different types of neighborhood moves to solve the problem. Similarly, Ding et al. [9] formulate the gate assignment problem as a quadratic assignment problem. As a different approach, they consider the over-constrained gate assignment problem, where some flights need to be assigned to the apron due to the limited number of gates at the airport, and they aim to minimize both the number of un-gated flights and the total passenger walking distance. For that model Ding et al. propose a two-stage solution method that consists of a greedy algorithm to minimize the number of un-gated flights and a tabu search heuristic to minimize the total passenger walking distance. In another study, Ding et al. [10] also use different types of heuristics like the simulated annealing and a hybrid of the simulated annealing and tabu search to solve the same assignment problem proposed in [9]. In the latter study, the authors provide a detailed computational analysis comparing the alternate heuristic methods. Drexl and Nikulin [13] and Pintea et al. [26] also consider the over-constrained gate assignment problem proposed by Ding et al [9]. However, they use a pareto simulated annealing heuristic and a hybrid ant-local search system, respectively.

As an alternate modeling approach Haghani and Chen [18] introduce a time-indexed (multiple-time slot) formulation by dividing the whole study period into the fixed time intervals. They propose a heuristic solution procedure to solve their model that minimizes the total passenger walking distance.

2.1.1 Modeling Uncertainty and Robustness

Flight timetable with arrival and departure times is the main input data for the airport gate assignment problem. In real-life applications, this input data is subject to uncertainty and may change over time due to the weather conditions, air traffic control delays, gate breakdowns, etc.. In particular, these uncertainties inherent in the system have a major impact on the performance of the gate assignment plans. Therefore, several authors focus on improving the performance of gate assignments by considering possible uncertainties in the problem parameters. Indeed, they try to obtain "robust models" providing "robust assignment plans" that are less sensitive to the disruptions in the system. In the literature, there are several studies which incorporate some robustness concepts to deal with the uncertainty. Here we briefly discuss the robustness approaches in the related literature.

The main objective of the proposed robust models is improving the performance of

static assignments by considering the unexpected changes in the flight schedules. Such models incorporate some robustness concepts to deal with the random disruptions. Lim and Wang [21] introduce a model minimizing the expected number of flight conflicts. They estimate the probability of conflict for each flight pair by using an un-supervised estimation function based on the deterministic (estimated) arrival and departure times. Thereby, they formulate the proposed model as a linear 0-1 integer program and they solve it by using a hybrid heuristic combining a tabu search and a local search algorithm.

The airport gate assignment problem under uncertainty is also considered in Mangoubi and Mathaisel [23]. The authors state that highly utilized gates may cause equipment and passenger congestion in the airport. Besides, those highly utilized gates are quite sensitive to the possible random disruptions such as early or late flight arrivals and departures. As a robust approach, Mangoubi and Mathaisel [23] propose to use a fixed buffer time amount between two continuous flights to absorb those stochastic changes in the schedules. Similarly, Hassounah and Steuart [20] argue that buffer time between flights is useful in improving the schedule punctuality. Yan and Chang [29], and Yan and Huo [30] also use a fixed buffer time value to obtain a robust schedule that would perform well under potential random flight delays.

Bolat [4] proposes an alternate approach to obtain a robust gate assignment. He claims that such an assignment can be obtained by distributing the idle times uniformly among gates. The motivation behind his claim is that distributing idle times more uniformly among gates increases the probability that the delayed departure will be still earlier than the arrival of the next flight. He mainly considers the objectives of minimizing the variance of the idle times (see [4–6]) or minimizing the range of the idle times (see [4, 5]). He proposes different heuristic algorithms to solve the proposed robust gate assignment models.

Lim et al. [22] consider minor variabilities in the arrival and departure times of flights to attain a robust assignment. The authors specify a time window in which a ground time of flight can slide, whereas in the previous models a flight is assigned to a gate at its exact arrival time. When flights are assigned to a gate after their time window starts, the delay penalties (proportional to the delay time) are applied in addition to the existing objective of minimizing the total passenger walking distance. As a solution approach they implement a tabu search and a memetic algorithm. A similar approach that incorporates the cost of assigning flights after a specified time (e.g. arrival time) is also used by Yan et al. [32]. They propose a framework with

two main stages, the planning and the real-time stages, and iteratively update the planning stage according to the results obtained in the real-time stage which utilizes a reassignment rule considering the potential real-time disruptions. In the planning stage of this iterative approach, the authors consider a scenario-based stochastic gate assignment model, which is formulated as a multiple commodity network flow problem, and by solving this formulation they obtain an assignment plan. Then for that assignment in the real-time stage the waiting times of the passengers are calculated for each scenario representing the real-time disruptions. These calculated waiting times are incorporated into the objective function of the model used in the planning stage as penalty adjustments and this iterative process ends after a certain number of iterations without any improvement in the best solution found so far.

Another approach recently studied in robust gate assignment problem is recovery strategies. Dorndorf et al. [12] present the gate assignment problem as a resource-constrained project scheduling problem and specify several robustness-related concepts based on resource-switching. In their first model, the objective function involves a robustness measure related with the available number of switchings. While in the second model, the fuzzy membership functions are used to penalize the schedule which is prone to disruptions. The authors do not propose any solution algorithm to solve the presented models.

2.2 Rule-based Expert Systems

Another main research direction in the airport gate assignment literature is simulation and rule-based expert systems. Yan et al. [31] propose a simulation framework to analyze the interrelationship between the planned (static) and real-time gate assignments under stochastic flight delays. The evaluations are done according to the different buffer time amounts and the real-time gate assignment rules. They consider the percentage of flights required to be reassigned and the deviation of the real-time objective function value from the planned one as the measures to reflect the affects of stochastic flight delays.

An expert system provides an assignment by using some special rules based on the knowledge of airport authorities and simulation studies. For expert systems it is important to define the rules and incorporate these rules into the decision process by considering an ordering based on their importance levels. Hamzwawi and Cheng [19] propose a rule-based expert system for simulating gate assignment operations. He

evaluates the effects of different assignment rules according to the improvement in the gate utilization. Recently, Cheng [7, 8] proposes a rule-based expert system that integrates mathematical programming techniques into the proposed expert system.

CHAPTER 3

STOCHASTIC PROGRAMMING MODELS

In this chapter, we discuss how to incorporate stochastic input data into the optimization models, elaborate on the robustness measures we consider and propose alternate stochastic programming formulations. Developing such alternate formulations allows us to model a wider range of preferences.

In traditional airport gate assignment problems the arrival and departure times of flights are assumed to be deterministic. It is common to refer to the given set of arrival and departure times as the "planned schedule". The deterministic models provide assignments based on a single planned schedule. In order to present the traditional gate assignment formulation, we first introduce the following parameters:

N: set of all flights arriving at and/or departing from the airport during the planning horizon;

M: set of gates available at the airport;

n: total number of flights, i.e., $n = |N|$, where $|N|$ denotes the cardinality of N;

m: total number of gates available, i.e., $m = |M|$;

a_i: arrival time of flight i, $i \in N$;

d_i: departure time of flight i, $i \in N$;

g_i: gate occupation time (ground time or apron time) of flight i ($g_i = d_i - a_i$), $i \in N$;

L_i: conflict set associated with flight i, $i \in N$.

Basically, the conflict set of flight i is the set of all flights which land before flight i and still on the ground at the time flight i arrives and it is defined as follows:

$$L_i = \{v \in N \mid a_v \leq a_i \text{ and } d_v > a_i\}.$$

Additionally, x_{ik} is a binary variable which equals to 1 if flight i is assigned to gate k, and 0 otherwise. Then the traditional gate assignment formulation reads:

$$\min_{\mathbf{x}} \quad f(\mathbf{x}) \tag{3.1}$$

$$\text{subject to} \quad \mathbf{x} \in \{0,1\}^{n*m} \tag{3.2}$$

$$\text{strict constraints} \quad \sum_{k \in M} x_{ik} = 1, \quad i \in N, \tag{3.3}$$

$$\sum_{v \in L_i} x_{vk} + x_{ik} \leq 1, \quad i \in N,\ k \in M, \tag{3.4}$$

$$\text{soft constraints} \quad A\mathbf{x} = b, \tag{3.5}$$

where $f(\mathbf{x})$ is the objective function. Note that, different types of objective functions are discussed in Chapter 1. Equations (3.3) guarantee the assignment of each flight to exactly one gate. Constraints (3.4) ensure that no two aircrafts are assigned to the same gate concurrently. We refer to those constraints as "conflict constraints" in the rest of the study. These constraints are defined as inequalities, since some gates may not be utilized in some time intervals. Furthermore, additional soft constraints can also be introduced to the model, which are here represented by (3.5). A detailed description can be found in [23].

As seen from the model the gate assignment problem in general is formulated for a given set of arrival and departure times of the flights. Thus, the stochastic nature of arrival and departure traffic is not considered and therefore, the optimal assignment found by solving such a deterministic model may perform poorly under certain realizations of the stochastic input data. In order to obtain more robust assignments, which would perform better in the presence of variability of the input data, we consider the uncertainty in arrival and departure times of the flights already at the modeling stage.

Decision problems in the presence of uncertainty are at the center of interest of operations research. Stochastic programming is one of the fundamental approaches that can be used to model such problems. It develops models to formulate optimization problems in which uncertain quantities are represented by random variables. We refer the interested reader to the books by Birge and Louveaux [3] and Prékopa [27], which are essential reference books in stochastic programming. In particular, we represent the uncertain arrival and departure times by random variables and characterize those random variables by using a finite set of scenarios. We assume that we are given a discrete set of scenarios representing the potential random disruptions, and their associated

probabilities. Let S denote the finite set of scenarios and p_s denote the probability associated with scenario s, $s \in S$. We can say that a scenario is a set of realizations of joint arrival and departure times of all flights. In our computational study, we generate the realizations of arrival and departures times from the planned schedules by adding or subtracting random deviation amounts. This is just one reasonable way of generating the scenarios, alternative approaches can also be utilized.

Considering the real life applications, the decision makers prefer to obtain a feasible assignment for the planned schedule. We take this preference into consideration, by enforcing the "conflict constraints" (3.4) for the planned schedule. It is important to note that trying to find an assignment which satisfies the "conflict constraints" for all the scenarios representing the random deviations from the planned schedule would be too conservative and unrealistic. In this spirit, we relax the "conflict constraints" and allow the occurrence of flight conflicts for the given set of scenarios representing the random disruptions. As previously described a flight conflict occurs when two flights with overlapping ground times are allocated to the same gate (see Figure 3.1).

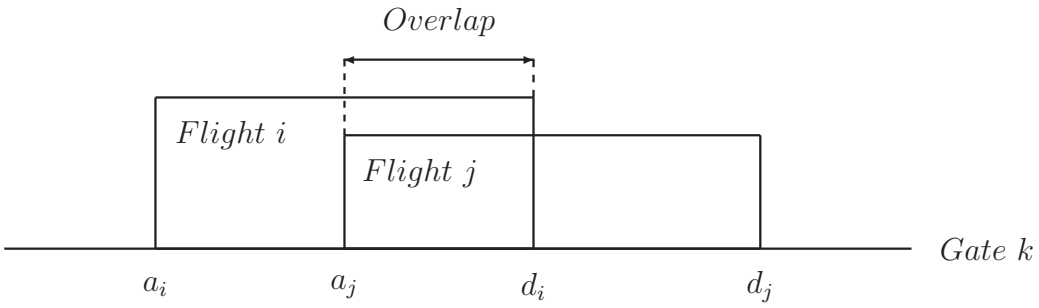

Figure 3.1: An illustrative example of a flight conflict

Basically, the underlying idea of the models we consider is to allow infeasibilities in the stochastic version of the "conflict constraints", while specifying restricting constraints on the amount of their violations. Due to the stochastic input data, the number of flight conflicts is a random variable. Comparing random variables is one of the main interests of decision making under uncertainty. The main objective of our proposed stochastic programming models is to minimize the expected number of flight conflicts. Except from minimizing the expectation, we also consider a risk measure on the random number of flight conflicts and introduce a gate assignment model based on the mean-risk approach. Such a mean-risk approach can also be utilized for the other proposed models which are risk-neutral, i.e., for the models considering the expected

values.

We can obtain an assignment that is less prone to potential random disruptions by uniformly distributing the idle times, the non-utilized times between two successively assigned flights. This observation is our motivation to propose alternate models incorporating the idle times. In the first idle time based model, we minimize the expected value of the variance of the idle times associated with an assignment plan. In the second idle time based model, we also consider the buffer time concept. Buffer time can be considered as a lower bound (threshold) value on an idle time. In this setup, we penalize the deviations of idle times from such threshold values. Note that different threshold values can be specified for idle times associated with different flights. For simplicity we assume that all the threshold values are the same, and therefore, we mention a single buffer time for the rest of the study. However, the proposed models are also valid for different threshold values. In the first model incorporating the buffer time as a robustness measure, we minimize the expected value of the total semi-deviations of idle times from the specified buffer time, whereas in the second model the objective is to minimize the number of idle periods deviating from the buffer time.

In the following sections we present our stochastic programming models under two main headings: the conflict based stochastic models and the idle time based stochastic models. We describe the models incorporating flight conflicts as the robustness measure in Section 3.1. In addition, Section 3.2 presents the models incorporating idle times as the robustness measure.

3.1 Conflict based Stochastic Models

In this section, we present two stochastic gate assignment models. The first model has a fairly similar objective with the model proposed by Lim and Wang [21], which aims to minimized the expected number of flight conflicts. By incorporating only the expectation measure, we cannot take the effect of the variability in the number of conflicts into consideration. Hence, we develop a second model involving a risk measure on the number of conflicts using the mean-risk approach. The mean-risk approach considers the objective of minimizing a combination of the expected value of a random variable and a risk measure on that random variable. Let us denote the random number of flight conflicts by $\mathcal{C}$. Then the mean-risk objective function is given by

$$E[\mathcal{C}] + \lambda\rho(\mathcal{C}),$$

where $\rho(.)$ is a specified risk measure and λ is a nonnegative trade-off coefficient representing the exchange rate of mean for risk. The value of the risk parameter is specified by decision makers according to their risk preferences.

The classical Markowitz [24] model uses the variance as the risk measure. One of the problems associated with the mean-variance formulation is that it treats underperformance equally as over-performance. However, we prefer the smaller values of the random number of conflicts and we should not penalize the values below the expected value. In order to remedy this drawback, models with asymmetric risk measures such as downside risk measures have been proposed (see e.g., Ogryczak and Ruszczyński [25]). Among the popular downside risk measures we focus on the absolute semi-deviation as the risk measure, which is defined as follows:

$$\rho(\mathcal{C}) = E[[\mathcal{C} - E[\mathcal{C}]]_+],$$

where $[z]_+ = \max(0, z),\ z \in \mathbb{R}$.

Here we introduce additional parameters:

Input Data

N^a: modified set of flights that includes two dummy flights representing the opening and closure times of gates;

B_k: opening time of gate k at the beginning of the planning period, $k \in M$;

E_k: closure time of gate k at the end of the planning period, $k \in M$;

$a_{i,s}$: realization of arrival time of flight i under scenario s, $i \in N,\ s \in S$;

$d_{i,s}$: realization of departure time of flight i under scenario s, $i \in N,\ s \in S$;

$L_{i,s} = \{j \in N^a,\ s \in S \mid a_{j,s} \leq a_{i,s} \quad \text{and} \quad d_{j,s} > a_{i,s}\},\ i \in N,\ s \in S.$

It is important to note that $L_{i,s}$ is the conflict set associated with flight i under scenario s and these random conflicting sets lead to stochastic conflicting constraints.

All of the proposed mathematical programming formulations involve the following primary decision variables:

$$x_{i,k} = \begin{cases} 1 & \text{if flight } i \text{ is assigned to gate } k, i \in N^a, k \in M \\ 0 & \text{otherwise.} \end{cases}$$

$$c_{i,j,s} = \begin{cases} 1 & \text{if flight } i \text{ and flight } j \text{ are conflicting under scenario } s, i, j \in N^a, s \in S \\ 0 & \text{otherwise.} \end{cases}$$

3.1.1 Risk-neutral model: Model minimizing the expected number of conflicts

Here we propose the model considering the expectation as the preference criterion while comparing the random variables to find the best assignment; hence, it is a risk-neutral approach. In particular, we try to find an assignment with the minimum expected number of conflicts while satisfying the conflict constraints for the planned schedule. We refer to this model as "MENC" and formulate as a mixed-integer linear programming problem:

$$\min \quad \sum_{i \in N^a} \sum_{j \in N^a} \sum_{s \in S} c_{i,j,s} p_s \tag{3.6}$$

$$\text{subject to} \quad \sum_{k \in M} x_{i,k} = 1, \quad i = 1, \ldots, n, \tag{3.7}$$

$$x_{i,k} = 1, \quad i = 0, (n+1), \quad k \in M, \tag{3.8}$$

$$\sum_{j \in L_{i,s}} x_{j,k} + x_{i,k} \leq 1, \quad i \in N^a,\ k \in M,\ s = 0, \tag{3.9}$$

$$c_{i,j,s} \geq x_{i,k} + x_{j,k} - 1, \quad i \in N^a,\ j \in L_{i,s},\ k \in M,\ s \in S, \tag{3.10}$$

$$x_{i,k} \in \{0, 1\}, \quad i \in N^a,\ k \in M, \tag{3.11}$$

$$c_{i,j,s} \geq 0, \quad i,\ j \in N^a,\ s \in S. \tag{3.12}$$

Constraints (3.7) guarantee the assignment of each flight to exactly one gate. Constraints (3.8) are used to allocate flight 0 and flight $(n+1)$ to all gates, where flight 0 and flight $(n+1)$ represent the opening and closure times of gates, respectively. Constraints (3.9) ensure that no two aircrafts are assigned to the same gate concurrently in the planned schedule. Notice that the subscript s equal to 0 represents the planned schedule. Thus, $L_{i,0}$ defines the conflict set associated with flight i under the planned schedule. Constraints (3.10) are used to determine the conflicting flights under each scenario. Due to the nature of the objective function, the variable $c_{i,j,s}$ takes the value 1 if and only if two flights (flight i and flight j) with overlapping ground times are allocated to the same gate under scenario s. The rest of the constraints are for the non-negativity and binary restrictions.

In the deterministic case it is assumed that all flight arrivals and departures occur in a predefined planning period, where a planning period is the time interval between the opening and the closure times of gates. However, in a stochastic setup we cannot guarantee that a delayed arrival or departure still occur in the planning period due to

the random disruptions in flight arrival and departure times. Hence, a flight conflict may occur because of an arrival before the opening time of a gate or a departure after the closure time of a gate. Therefore, we introduce two dummy flights (flight 0 and flight $(n+1)$) representing the opening and closure times of gates and define the modified flight set N^a. Here, we consider only two dummy flights, since we assume that the gates are homogenous; the opening and closure times of all gates are same, $B_k = B$ and $E_k = E$ for all $k \in M$ and all the gates are utilized in the same time interval (i.e. $[B - E]$).

In order to model the flights conflicting with the opening and closure times of gates, we need to define the arrival and departure times of these two dummy flights as follows:

$$a_{0,s} < \min_{i \in N} \ a_{i,s}, \qquad d_{0,s} = B,$$

$$a_{(n+1),s} = E, \qquad d_{(n+1),s} > \max_{i \in N} \ d_{i,s}.$$

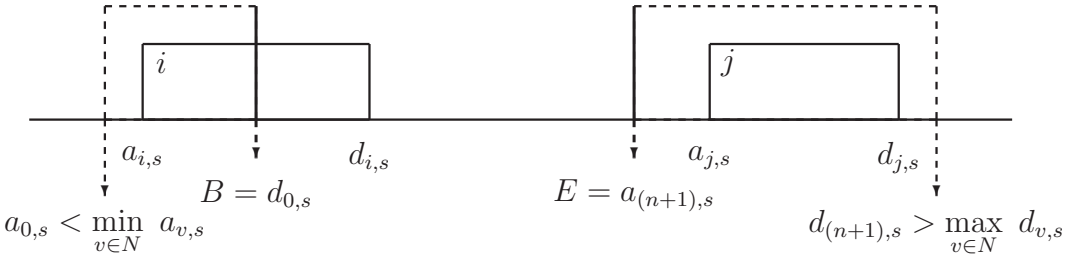

Figure 3.2: Representation of gates as flights

Figure 3.2 shows how a flight may conflict with the opening or closure time of a gate. This figure illustrates how to capture those conflicts by specifying the arrival and departure times of the dummy flights as described above. Note that if we consider heterogeneous gates, we need to define $2m$ dummy flights since all gates may have different opening and closure times.

In this formulation, if there exists two flights having the same arrival times, the flight conflicts are counted twice since they both occur in each others' conflict sets. In order to avoid this, we assume, without loss of generality, that the arrival times of flights are different from each other.

3.1.2 Risk-averse model: Model minimizing the mean-risk function of the number of conflicts

Since the risk-neutral model only considers the expectation as the preference criterion while comparing the random variables (e.g. number of flight conflicts), it does not deal with the variability of random variables. Hence, the obtained solution may show distinctive fluctuations and it may not be reliable under certain realizations of random input data. To overcome this problem, we need to consider the concept of risk. Therefore, we propose an alternative formulation based on the mean-risk approach and we specify the absolute semi-deviation as the risk measure. In this proposed risk-averse model, which we refer to as "MMRNC", we introduce decision variables $\theta_s,\ s \in S$ and use the following constraints in (3.13) to calculate the realizations of the random variable $[\mathcal{C} - E[\mathcal{C}]]_+$. Recall that $\mathcal{C}$ represents the random number of flight conflicts.

$$\theta_s \geq [\sum_{i\in N^a}\sum_{j\in N^a} c_{i,j,s} - \sum_{i\in N^a}\sum_{j\in N^a}\sum_{s\in S} c_{i,j,s}p_s]_+, \quad s \in S. \tag{3.13}$$

Then the proposed stochastic programming model minimizing the combination of the expectation and the absolute semi-deviation risk measure for the random number of flight conflicts becomes

$$\min \quad \sum_{s\in S}\sum_{i\in N^a}\sum_{j\in N^a} c_{i,j,s}p_s + \lambda\sum_{s\in S}\theta_s p_s \tag{3.14}$$

$$\text{subject to} \quad (3.7) - (3.12), \tag{3.15}$$

$$\theta_s \geq \sum_{i\in N^a}\sum_{j\in N^a} c_{i,j,s} - \sum_{i\in N^a}\sum_{j\in N^a}\sum_{s\in S} c_{i,j,s}p_s, \quad s \in S, \tag{3.16}$$

$$\theta_s \geq 0, \quad s \in S. \tag{3.17}$$

Due to the nature of the objective function and the nonnegativity constraints on variables $\theta_s, s \in S$, at the optimal solution constraints in (3.16) associated with scenarios for which the number of flight conflicts is larger than or equal to the expected number of flight conflicts are tight. In other words, if at the optimal solution $\sum_{i\in N^a}\sum_{j\in N^a} c_{i,j,s} > \sum_{i\in N^a}\sum_{j\in N^a}\sum_{s\in S} c_{i,j,s}p_s$, then the constraint associated with scenario s is tight. Otherwise, θ_s takes the value 0. Thus, using the above formulation we properly calculate the realizations of the random variable $[\mathcal{C} - E[\mathcal{C}]]_+$.

3.2 Idle Time based Stochastic Models

In this section, we propose models that incorporate the concept of the idle time. Recall that the idle time or idle period is defined as the non-utilized time period between two successively assigned flights (see Figure 3.3).

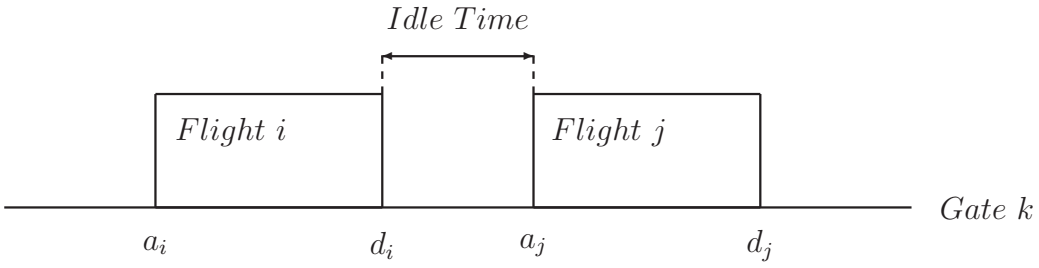

Figure 3.3: An illustrative example of an idle time

The objective of the first idle time based model, which is the minimization of the variance of the idle times, has been introduced by Bolat [5]. The main motivation of this objective is to obtain an assignment that can absorb minor disruptions in the arrival and departure times of flights. Bolat [5] argues that distributing idle times uniformly is expected to increase the probability that the delayed departure of a flight will be still earlier than the arrival of the next flight. He mainly justifies his claim based on the assumptions that the deviations of flight arrivals and departures are independent and equally like to occur. In this line of research, we propose a stochastic programming model minimizing the expected variance of the idle times. Note that we calculate the variance among the idle times associated with all the flights. It is important to note that due to the stochastic setup, the idle times are random and as a function of random variable the variance of the idle times is random. Here we calculate the expected value of this random variance associated with all the idle times.

In the other idle time based models, we also incorporate the buffer time as a robustness measure. In literature, the buffer time (b) is used as a lower bound value on each single idle time. Arrival and departure times of flights are modified according to the buffer time value; hence, for any feasible solution it is guaranteed to have idle times at least equal to the predetermined buffer time value (see Figure 3.4).

We can refer to the idle time illustrated in Figure 3.4 by the "idle time immediately after the departure of flight i" or "idle time immediately before the arrival of flight j". Unfortunately, it is not trivial to incorporate the above buffer time concept into the stochastic models. Basically, the main idea is considering the buffer time as a lower

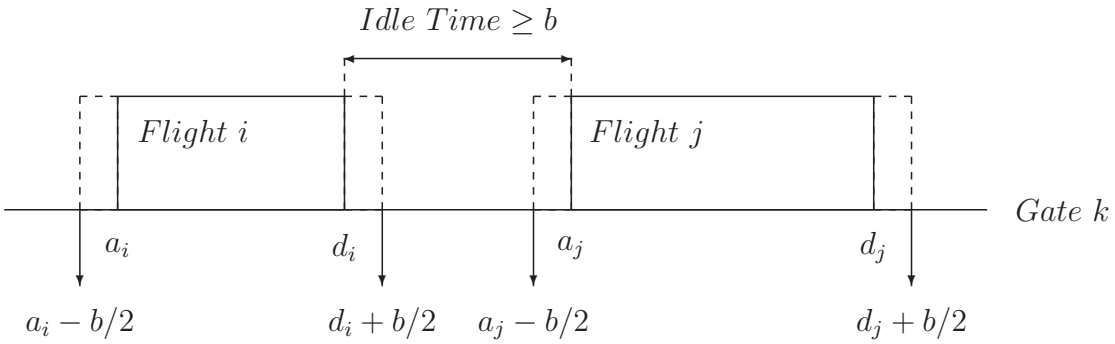

Figure 3.4: An illustrative example of the buffer time

bound on all the idle times, which can be represented by one of the following sets of constraints:

$$(\text{idle time immediately after the departure of flight } i) \geq b \quad \forall i \in N. \tag{3.18}$$

$$(\text{idle time immediately before the arrival of flight } j) \geq b \quad \forall j \in N. \tag{3.19}$$

However, as discussed above, the idle times are random variables in the stochastic setup and so constraints (3.18) and (3.19) are stochastic. As we have done for the conflicting constraints, we allow the infeasibilities for the stochastic constraints, i.e., we allow the idle times to be below the buffer time. In order to control the violation (semi-deviation) amounts we minimize the expectation of total violation amounts or the expected number of violated constraints.

Our primary objective in the idle time based models is minimizing the expected number of flight conflicts. As a secondary objective we try to find the assignment that is best in terms of the robustness measures discussed above. The general form of the objective function for idle time based stochastic programming problems is:

$$\min_{\mathbf{x}} \ \{h(\mathbf{x}) + \Lambda E[\mathcal{C}]\},$$

where $E[\mathcal{C}]$ denotes the expected number of conflicts and $h[\mathbf{x}]$ is the secondary objective function for a decision vector $\mathbf{x}$. Note that Λ is a sufficiently large number.

3.2.1 Model minimizing the expected variance of the idle times and the number of conflicts

We present a novel stochastic optimization model minimizing the expected variance of the idle times, which we refer as "MEVINC". Additional to the number of flight conflicts the uniformity of the idle times is considered as a robustness measure, since uniformly distributed idle times is expected to decrease the probability of flight conflicts under possible random disruptions.

Let us present the related notations:

Input Data:

N^b: modified set of flights that includes $(m+1)$ dummy flights representing the opening and closure times of gates;

$L'_{i,s} = \{j \in N^b,\ s \in S \mid a_{j,s} \geq d_{i,s}\}$: set of all flights which land after the departure of flight i under scenario $s,\ s \in S$ (referred as the non-conflict set associated with flight i);

Decision variables:

$A_{j,s}$: arrival time of the flight which immediately succeeds flight j under scenario s, $j \in N^b \setminus \{0\},\ s \in S$;

$I_{j,s}$: idle time occurs immediately after the departure of flight j under scenario s, $j \in N^b \setminus \{0\},\ s \in S$;

μ_s: mean of idle times under scenario $s,\ s \in S$;

V_s: variance of idle times under scenario $s,\ s \in S$.

Notice that if we have n flights and m gates, we can define exactly $(n+m)$ idle periods. n idle periods occur after the departure of flights, while m idle periods occur just after the opening time of the gates. We calculate the mean and variance of idle times under each scenario as follows:

$$\mu_s = \frac{\sum_{j \in N^b} I_{j,s}}{(n+m)}$$

$$V_s = \frac{\sum_{j \in N^b} (I_{j,s} - \mu_s)^2}{(n+m-1)}$$

Before presenting the stochastic model, we want to describe the underlying deterministic model minimizing the variance of idle times. Recall that such a model has been

initially introduced by Bolat [5]. The modeling approach in [5] is based on the fact that the flights can be sorted in ascending order of their arrival times. This approach is not valid while formulating a stochastic optimization model, since we cannot obtain such an ordering for the random arrival and departure times. In our setup, each scenario may lead to a different ordering of arrival and departure times. Hence, it is not trivial to incorporate the idle times into a stochastic optimization model. Here we propose an alternative deterministic model, which is equivalent to the one proposed by Bolat [5]. This model will allow us to develop the stochastic version. Let us present our deterministic formulation by dropping the scenario indices for the previously defined parameters and variables.

$$\min \quad V \tag{3.20}$$

$$\text{subject to} \quad \sum_{k \in M} x_{i,k} = 1, \quad i = 1, \ldots, n, \tag{3.21}$$

$$x_{0,k} = 1, \quad k \in M, \tag{3.22}$$

$$x_{n+k,k} = 1, \quad k \in M, \tag{3.23}$$

$$\sum_{j \in L_i} x_{j,k} + x_{i,k} \leq 1, \quad i \in N^b,\ k \in M, \tag{3.24}$$

$$A_j \leq (2 - x_{j,k} + x_{i,k})Z + a_i, \quad j \in N^b \setminus \{0\},\ i \in L_j^{'},\ k \in M, \tag{3.25}$$

$$\sum_{j \in N_b \setminus \{0\}} A_j = \sum_{j \in N} a_j + a_0 m, \tag{3.26}$$

$$I_j = A_j - d_j, \quad j \in N^b \setminus \{0\}, \tag{3.27}$$

$$x_{i,k} \in \{0, 1\}, \quad i \in N^b,\ k \in M, \tag{3.28}$$

$$I_j, A_j \geq 0, \quad j \in N^b \setminus \{0\}. \tag{3.29}$$

Here Z is a sufficiently large number; for example, it can be set to the maximum departure time among all flights.

We formulate the problem as a mixed integer programming problem with a nonlinear objective function. The objective minimizes the variance of idle times. Constraints (3.22) are used to assign dummy flight 0 to all gates since it represents the common closure times. Additionally, the remaining m flights representing the gate openings are assigned to the corresponding gates by (3.23) to calculate the idle times at the very beginning of the gate openings. Recall that in the deterministic setup, constraints (3.24) are used to avoid flight conflicts. Constraints (3.25) provide the arrival times of the succeeding flights as upper bounds on A_j values. Remark that we are not minimizing

or maximizing individual idle time values. Therefore, in order to calculate the idle time values exactly, we need to assign the appropriate upper bound values to A_j variables. Since the last assigned m flights denote the gate closure times and for the remaining flights A_j value should keep the arrival of time of the succeeding flight, we add equality (3.26) to guarantee that constraints (3.25) are tight for the upper bound values. Equations (3.27) are used to calculate the idle time values as illustrated in Figure 3.5. The rest of the constraints are for the non-negativity and binary restrictions.

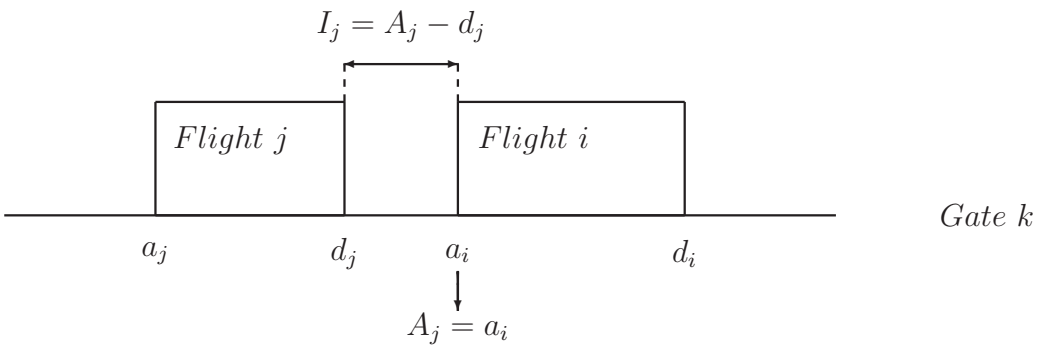

Figure 3.5: An idle time calculation

Recall that in the stochastic setup we allow flight conflicts. In this case we face a problem, since the idle periods are not defined for conflicting flights. In order to overcome this challenge and calculate the idle times properly, we need to identify different types of conflicts and for this purpose we first need to define the following conflict sets:

$L^p_{i,s} = \{j \in N^b,\ s \in S \mid a_{j,s} \geq a_{i,s},\ a_{j,s} < d_{i,s} \quad \text{and} \quad d_{j,s} \geq d_{i,s}\}$: set of all flights which land when flight i is on the ground and depart after the departure of flight i under scenario $s,\ s \in S$ (referred as the partial-conflict set associated with flight i under scenario s);

$L^f_{i,s} = \{j \in L_{i,s},\ s \in S \mid d_{j,s} > d_{i,s}\}$: set of all flights which land before flight i and still on the ground at the time flight i departs under scenario $s,\ s \in S$ (referred as the full-conflict set associated with flight i under scenario s);

Figure 3.6 shows the non-conflict case and two conflict cases corresponding to the described conflict sets. Recall that we define the idle period as the non-utilized time period after the departure of a flight. According to this definition, in the partial and full conflict cases the idle time values related to flight j cannot be defined. In our formulation, the idle time values of those conflicting flights such as flight j in Figure

3.6 are considered as 0. Such an approach is reasonable, since there is no non-utilized time period between conflicting flights.

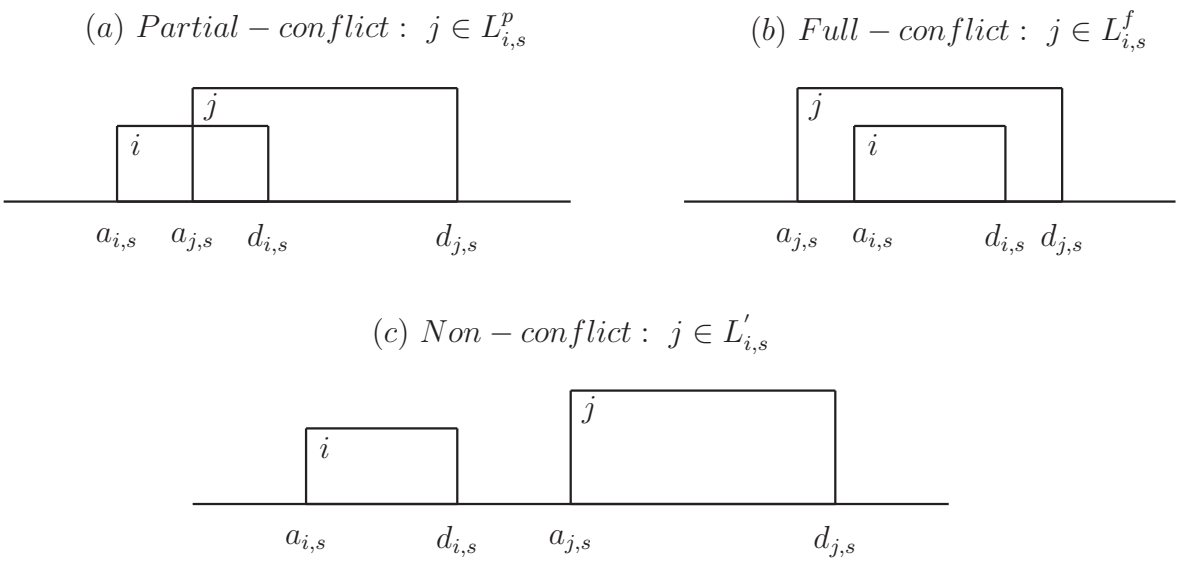

Figure 3.6: Different types of flight conflicts

Considering partial and full conflicts separately requires us to define the conflict variables accordingly as follows:

$c^p_{j,k,s}$: number of partially conflicting flights with flight j at gate k under scenario s, $j \in N^b \setminus \{0\},\ k \in M,\ s \in S$;

$c^f_{j,k,s}$: number of fully conflicting flights with flight j at gate k under scenario s, $j \in N^b \setminus \{0\},\ k \in M,\ s \in S$;

Recall that idle time based models, the primary objective is the minimization of the expected number of flight conflicts and the secondary objective differs according to the additional robustness measures. In this idle time based model, as a secondary objective we try to find the assignment that is best in terms of the uniformity of the idle times. By extending the deterministic formulation (3.20)-(3.29) we obtain the formulation of the proposed stochastic model in the following form:

$$\min \quad \sum_{s\in S} V_s p_s + \Lambda \sum_{i\in N^b} \sum_{k\in M} \sum_{s\in S} (c^p_{i,k,s} + c^f_{i,k,s}) p_s, \tag{3.30}$$

$$\text{subject to} \quad \sum_{k\in M} x_{i,k} = 1, \quad i \in N, \tag{3.31}$$

$$x_{0,k} = 1, \quad k \in M, \tag{3.32}$$

$$x_{n+k,k} = 1, \quad k \in M, \tag{3.33}$$

$$\sum_{j \in L_{i,s}} x_{j,k} + x_{i,k} \leq 1, \quad i \in N^b,\ k \in M,\ s = 0, \tag{3.34}$$

$$A_{j,s} \leq (2 - x_{i,k} - x_{j,k})Z + a_{i,s}, \quad i \in L'_{j,s},\ j \in N^b \setminus \{0\},\ k \in M,\ s \in S, \tag{3.35}$$

$$A_{j,s} \leq (2 - x_{i,k} - x_{j,k})Z + a_{i,s}, \quad i \in L^p_{j,s},\ j \in N^b \setminus \{0\},\ k \in M,\ s \in S, \tag{3.36}$$

$$A_{i,s} \leq (2 - x_{i,k} - x_{j,k})Z + a_{i,s}, \quad i \in N^b \setminus \{0\},\ j \in L^f_{i,s},\ k \in M,\ s \in S, \tag{3.37}$$

$$\sum_{j \in N^b \setminus \{0\}} A_{j,s} = \sum_{j \in N} a_{j,s} + a_{0,s}m, \quad s \in S, \tag{3.38}$$

$$I_{j,s} = A_{j,s} - d_{j,s} + s^p_{j,s} + s^f_{j,s}, \quad j \in N^b \setminus \{0\},\ s \in S, \tag{3.39}$$

$$I_{j,s} \leq (2 - x_{i,k} - x_{j,k})Z, \quad i \in L^p_{j,s},\ j \in N^b \setminus \{0\},\ k \in M,\ s \in S, \tag{3.40}$$

$$I_{i,s} \leq (2 - x_{i,k} - x_{j,k})Z, \quad i \in N^b \setminus \{0\},\ j \in L^f_{i,s},\ k \in M,\ s \in S, \tag{3.41}$$

$$c^p_{j,k,s} \geq \sum_{i \in Lpj,s} x_{i,k} + (x_{j,k} - 1)(n + m), \quad j \in N^b,\ k \in M,\ s \in S, \tag{3.42}$$

$$c^f_{i,k,s} \geq \sum_{j \in Lfi,s} x_{j,k} + (x_{i,k} - 1)(n + m), \quad i \in N^b,\ k \in M,\ s \in S, \tag{3.43}$$

$$s^p_{j,s} \leq (\sum_{k \in M} c^p_{j,k,s})Z, \quad j \in N^b,\ s \in S, \tag{3.44}$$

$$s^f_{i,s} \leq (\sum_{k \in M} c^f_{i,k,s})Z, \quad i \in N^b,\ s \in S, \tag{3.45}$$

$$x_{i,k} \in \{0, 1\}, \quad i \in N^b,\ k \in M, \tag{3.46}$$

$$\text{All remaining variables} \geq 0. \tag{3.47}$$

Here we only elaborate on the new types of constraints introduced to calculate the exact idle time values under each scenario. Let us explain those sets of constraints in detail using an illustrative example.

In order to calculate the exact idle time values we need to obtain the correct values of $A_{j,s}$ variables. In deterministic case it is quite easy to attain the correct values by only using the constraints (3.25), since we do not allow flight conflicts. However, in the stochastic setup flight conflicts may occur and those conflicts prevent us to consider each arrival time exactly once in the idle time calculations. Therefore, we define two types of flight conflicts and add the related constraints (3.36-3.37) that bound the $A_{j,s}$ variables from above. If for all the flights $A_{j,s}$ variables take the upper bound values, we

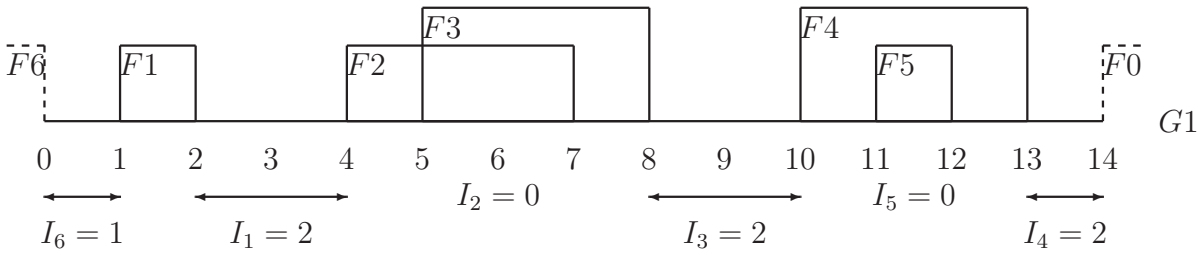

Figure 3.7: A representative example for model MEVINC

guarantee to calculate the exact idle time values. We use the following equation similar to equation (3.26) of the deterministic formulation to guarantee that the constraints (3.35-3.37) are tight for the upper bound values:

$$A_{j,s} = \sum_{j \in N} a_{j,s} + a_{0,s} m.$$

Let us consider the example shown in Figure 3.7:

- There are 5 flights and 1 gate that is available between the time interval $[0, 14]$. The couples of the arrival and departure times are as follows: (1,2); (4,7); (5,8); (10,13); (11,12). Note that we drop the scenario indices for simplicity.

- Flight 1 ($F1$): $F1$ does not conflict with any other flight at gate 1 ($G1$). In other words, $F1$ is not allocated to a gate where some other flights from its conflict set are also allocated. $\implies$ $c^p_{1,1} = 0$; $c^f_{1,1} = 0$, (Constraints (3.42 and 3.43)). Then the corresponding slack variables (s^p_1 and s^f_1) are forced to be 0 (Constraints (3.44 and 3.45)). Note that the slack variables ($s^p_{j,s}$ and $s^f_{j,s}$) are used to make the idle time values of the conflicting (partial or full) flights equal to 0. Constraints (3.38) guarantee that constraints (3.35) are tight for the upper bound values. Thus, we can calculate idle time of $F1$ exactly:

$$A_1 = a_2 = 4 \text{ by (3.35) and (3.38)},$$

$$I_1 = A_1 - d_1 + s^p_1 + s^f_1 = 4 - 2 + 0 + 0 = 2 \text{ by (3.39)}.$$

- Flight 2 ($F2$): $F2$ is partially conflicting with Flight 3 ($F3$), i.e., $F3 \in L^p_2$.

Therefore, the conflict variable associated with $F2$ takes the value 1 due to the nature of the objective, i.e., $c^p_{2,1} = 1$. Then, the slack variable related to $F2$ is allowed to take any nonnegative value smaller than a relatively large number, Z (Constraint (3.44)). Finally, we calculate the idle time of $F2$ exactly:

$$I_2 = 0 \text{ by (3.40)},$$

$$A_2 = a_3 = 5 \text{ by (3.36) and (3.38)},$$

$$I_2 = A_2 - d_2 + s^p_2 + s^f_2 = 5 - 7 + 2 + 0 \text{ by (3.39)}.$$

Notice that s^p_2 takes the value 2 to equalize I_2 to 0, since we cannot determine the idle time of this flight and constraint (3.40) forces the idle time of $F2$ to be 0.

- Flight ($F3$), Flight ($F4$) and Flight ($F6$): Similar to $F1$ we can calculate the corresponding idle times exactly.

- Flight 5 ($F5$): $F5$ is fully conflicting with $F4$ ($F4 \in L^f_5$). Therefore, the conflict variable related to $F5$ takes the value 1 due to the nature of the objective, i.e., $c^f_{5,1} = 1$. Then the slack variable related to $F5$ is allowed to take any nonnegative value smaller than Z (Constraint (3.45)). Similar to the calculations for $F2$ we have:

$$I_5 = 0 \text{ by (3.41)},$$

$$A_5 = a_5 = 11 \text{ by (3.37) and (3.38)},$$

$$I_5 = A_5 - d_5 + s^p_5 + s^f_5 = 11 - 12 + 0 + 1 \text{ by (3.39)}.$$

We keep a_5 as the variable A_5 to make the equation (3.38) works. Moreover, notice that s^f_5 takes the value 1 to equalize I_5 to 0.

3.2.2 Model minimizing the expectation of total semi-deviation and the number of conflicts

A buffer time between two continuous flights allocated to the same gate may absorb the stochastic flight delays. As discussed at the beginning of this chapter, it is not trivial to incorporate the buffer time, as a lower bound on each idle time, into the stochastic optimization models. Our novel approach to model the random idle times, discussed

in the previous section, allows us to also take the buffer time into consideration and leads to other novel stochastic optimization models.

Recall that the idle times are random variables in the stochastic setup, and therefore, constraints (3.18) and (3.19), which involve the idle and buffer times, are stochastic. Basically, under some realizations of input data these constraints can be violated. In our models, we allow such violations; the idle times may be smaller than the predetermined buffer time b and we refer to such deviations as "semi-deviations". It is clear that we should not penalize the idle time values above the buffer time. In order to control the random semi-deviation amounts, we minimize the expectation of the total semi-deviation. We refer to this proposed model as "METDNC". Additional parameters required for the model METDNC can be described as follows:

N^c: modified set of flights that includes $(m+1)$ dummy flights representing the opening and closure times of gates;

$Q_{j,s} = \{i \in N^c,\ s \in S \mid d_{i,s} \leq a_{j,s}\}$: set of all flights which depart before the arrival of flight j under scenario $j \in N,\ s \in S$;

Recall that we also introduce a set, denoted by N^b, similar to N^c for the model MEVINC. Here we elaborate on the differences between these two sets. N^c includes $(m+1)$ dummy flights where the last m dummy flights represent the gate openings and flight 0 represents the gate closures, whereas in the set N^b the last m dummy flights represent the gate closures and flight 0 represents the gate openings. This difference originates from the proposed alternative procedures to calculate the idle times. Notice also that if the gates are assumed to be heterogeneous then we shall define a single set of flights by representing each gate opening and closure as a dummy flight. In other words, when we consider the heterogeneous gates, we shall use the same flight set in all of the proposed formulations. However, when the gates are assumed to be homogenous considering the same set of flights is not necessary and requires us to introduce unnecessary variables and constraints. Therefore, we distinguish the sets of flights such as N^b and N^c for the computational efficiency.

The formulated mathematical programming models involve the following decision variables:

$D_{j,s}$: departure time of the flight which immediately precedes flight j under scenario s, $j \in N^c \setminus \{0\},\ s \in S$;

$I_{j,s}$: idle time occurs immediately before the arrival of flight j under scenario s, $j \in N^c \setminus \{0\},\ s \in S$.

Let us also define the auxiliary decision variables $R_{j,s},\ j \in N^c \setminus \{0\},\ s \in S$, as the semi-deviation amounts associated with each flight and each scenario.

$$R_{j,s} = [b - I_{j,s}]_+ = \max(0, b - I_{j,s}), \quad j \in N^c \setminus \{0\},\ s \in S.$$

Figure (3.8) illustrates how the idle time values are calculated for the models involving buffer time. Observe that this way of calculating the idle times is an alternative to the one illustrated in Figure (3.5).

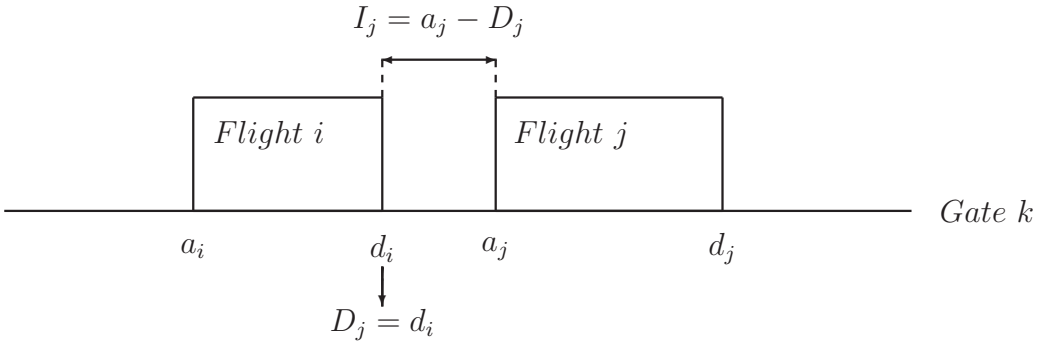

Figure 3.8: Calculation of the idle time values

Then the proposed model is formulated as a mixed integer program as follows:

$$\min \quad \sum_{j \in N^c \setminus \{0\}} \sum_{s \in S} R_{j,s} p_s + \Lambda \sum_{i \in N^c} \sum_{j \in N^c} \sum_{s \in S} c_{i,j,s} p_s \tag{3.48}$$

$$\text{subject to} \quad \sum_{k \in M} x_{i,k} = 1, \qquad i = 1, \ldots, n, \tag{3.49}$$

$$x_{0,k} = 1, \quad k \in M, \tag{3.50}$$

$$x_{n+k,k} = 1, \quad k \in M, \tag{3.51}$$

$$\sum_{j \in L_{i,s}} x_{j,k} + x_{i,k} \leq 1, \quad i \in N^c,\ k \in M,\ s = 0, \tag{3.52}$$

$$c_{i,j,s} \geq x_{i,k} + x_{j,k} - 1, \quad i \in N^c,\ j \in L_{i,s},\ k \in M,\ s \in S, \tag{3.53}$$

$$R_{j,s} \geq b - I_{j,s}, \quad j \in N^c \setminus \{0\},\ s \in S, \tag{3.54}$$

$$D_{j,s} \geq (x_{i,k} + x_{j,k} - 2)Z + d_{i,s}, \quad j \in N^c \setminus \{0\},\ i \in Q_{j,s},\ k \in M,\ s \in S, \tag{3.55}$$

$$I_{j,s} \leq a_{j,s} - D_{j,s} + b \sum_{i \in N^c} c_{j,i,s}, \quad j \in N^c \setminus \{0\},\ s \in S, \tag{3.56}$$

$$x_{i,k} \in \{0,1\}, \quad i \in N^c,\ k \in M, \tag{3.57}$$

$$\text{All remaining variables} \geq 0. \tag{3.58}$$

As for the model MEVINC, we only discuss the new types of constraints involved in the formulation above. Constraints (3.54) are used to calculate the semi-deviation amounts and these calculations are exact due to the structure of the objective function and the nonnegativity restrictions. Constraints (3.55) state that for each flight j and each scenario s the smallest value that the variable $D_{j,s}$ takes is the departure time of the flight that precedes flight j. Constraints (3.56) provide upper bounds on the idle time values according to the values of $D_{j,s}$ and $c_{i,j,s}$ variables. Next, we elaborate on the idle time calculations using constraints (3.55) and (3.56):

- We calculate the idle time values in the non-conflict and conflict cases as shown in Figure (3.9). If flight j is not conflicting with any other flight, then we can precisely define the associated idle time value. However, in the conflict case we cannot specifically define the idle time period. In order to exclude the effect of semi-deviations associated with conflicting flights, we set the values of such semi-deviations to 0. This implies the assumption that the the idle time value related to a conflicting flight is greater than or equal to the buffer time. Such an approach does not favor the conflicting flights while minimizing the total semi-deviation in our models, since the primary objective is to minimize the expected number of flight conflicts.

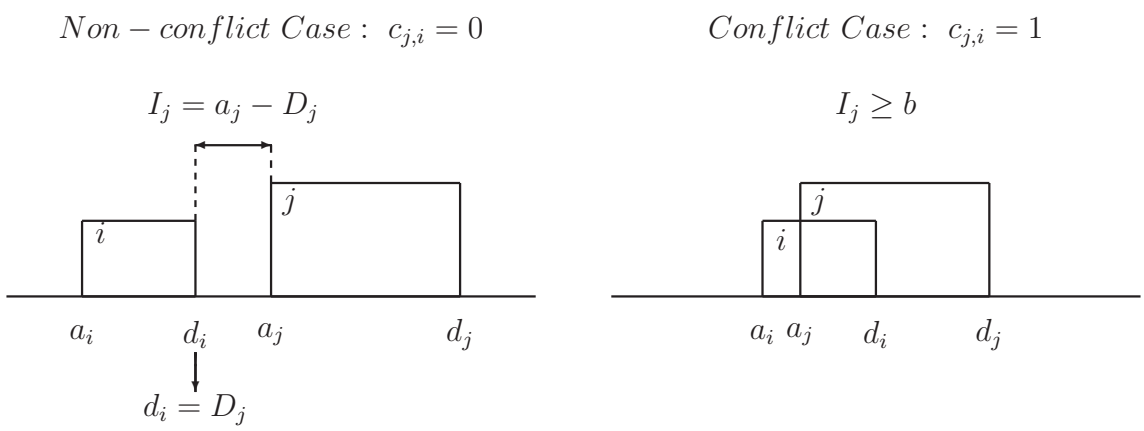

Figure 3.9: Calculation of the idle time values

- In the non-conflict case, the variables $c_{j,i,s}$ corresponding to flight j are equal

to 0. There are two possible cases regarding the idle time value; the idle time is smaller than the buffer time or the idle time is greater than or equal to the buffer time. In order to calculate the objective function value exactly at the candidate solution, we need to calculate the idle times exactly for the first case ($I_{j,s} = a_{j,s} - D_{j,s}$). In the second case, the idle time is greater than or equal to the buffer time and without calculating the exact idle time we obtain the semi-deviation amount as 0 due to the structure of the objective function. Thus, in the second case $I_{j,s}$ is guaranteed to be at least equal to the buffer time value and we do not need to add a penalty associated with this case to the objective function. For the first case we show that the variable $I_{j,s}$ takes the upper bound value in (3.56), which guarantees the exact calculation of the idle time. When the idle time is smaller than the buffer time value, the associated semi-deviation is strictly positive. Since the larger values of idle times are preferred to minimize the associated semi-deviation amount and so the total semi-deviation, $D_{j,s}$ variables take the smallest possible value defined in (3.55) and the idle time $I_{j,s}$ takes the largest possible value according to constraints (3.56).

- In case of flight conflicts, the idle time of the conflicting flight with the latest arrival time can not be defined (e.g. flight j, see Figure 3.9). As we discussed above, for such flights we try to obtain an idle time value greater than or equal to b so that the associated semi-deviation takes the value of 0. Constraint (3.56) guarantees this assumption by incorporating the additional part to the right hand side: $b \sum_{i \in N^c} c_{j,i,s}$. Note that $b \sum_{i \in N^c} c_{j,i,s}$ is greater than or equal to b and the difference $(a_{j,s} - D_{j,s})$ is always nonnegative.

3.2.3 Model minimizing the expected number of semi-deviations and number of conflicts

In the model METDNC presented in the previous section, as a secondary robustness measure we focus on the violation (semi-deviation) amounts for the specified buffer time. Alternatively, in this section we develop a stochastic optimization model which considers the number of semi-deviations instead of semi-deviation amounts. Hence, we refer to this model as "MENDNC".

Let us introduce the following auxiliary decision variables for $j \in N^c,\ s \in S$:

$$H_{j,s} = \begin{cases} 1 & \text{if idle time of flight j is less than buffer time, } i \in N^c, k \in M \\ 0 & \text{otherwise.} \end{cases}$$

The mixed-integer linear programming formulation of the proposed model is given by

$$\min \quad \sum_{j \in N^c \setminus \{0\}} \sum_{s \in S} H_{j,s} p_s + \Lambda \sum_{i \in N^c} \sum_{j \in N^c} \sum_{s \in S} c_{i,j,s} p_s \tag{3.59}$$

$$\text{subject to} \quad (3.49) - (3.58), \tag{3.60}$$

$$H_{j,s} b \geq b - I_{j,s}, \quad j \in N^c \setminus \{0\},\ s \in S, \tag{3.61}$$

$$H_{j,s} \in \{0,1\}, \quad j \in N^c \setminus \{0\},\ s \in S. \tag{3.62}$$

Constraints (3.61) and (3.62) enforce that the variable $H_{j,s}$ takes the value 1 if the idle time is less than the predetermined buffer time value. Otherwise, they guarantee that $H_{j,s}$ takes the value 0 due to the structure of objective function.

Note that specifying the appropriate buffer times for the models incorporating them as lower bounds (METDNC-MENDNC) is significant. Decision makers should determine the appropriate buffer time values based on the available input data (i.e. total available gate time, total ground time, etc.) and their robustness preferences. Here, we suggest some potential alternate methods to determine the buffer time value (b):

- Calculate the total idle time value for the planned schedule, which is basically equal to the difference between the total available gate time and the total ground time. In the best case, this total value can be equally distributed among flights (i.e. $= (\sum_{j \in N^c} I_j)/(n+m)$). A proportion of this equally distributed idle time can be defined as the buffer time value.

- Solve the deterministic model minimizing the variance of idle times for the planned schedule and use one of the following methods to define the buffer time value.

  - Specify the largest idle time value ($\max_{j \in N^c} I_j$) and take a proportion of that value.

  - Take a proportion of the median of idle times.

  - Some proportion of the trimmed mean of idle times can be used. Trimmed mean is computed similar to the arithmetic mean after discarding some percent of the smallest and largest values. Note that, in contrast to arithmetic mean, the trimmed mean is a robust measure of the central tendency.

In our computational study we employ the first suggested method and take the half of the equally distributed idle time value as the buffer time.

Here we briefly elaborate on some potential alternative approaches. For example, we can introduce the buffer time into the model as a decision variable. In such an approach we can also enforce a lower bound on the buffer time (e.g. the buffer time value is supposed to be greater than or equal to a proportion of expected idle times). As an alternative modeling approach, we can assign a probability also for the planned schedule and include the planned schedule in the set S which represents the possible scenarios. In such an approach, we do not need to guarantee that no flight conflicts occur for the planned schedule and can drop the conflict constraints associated with the planned schedule. We prefer our approach for the practical reasons as explained at the beginning of this chapter.

We would also like to emphasize that just for simplicity we assume that the same buffer time value is specified for all the flights. We can easily relax this assumption in the proposed models and specify different buffer time values depending on the decision makers' preferences for the individual flights. Let us consider the buffer time which is introduced as a lower bound on the idle time that occurs just before the arrival of flight j, as illustrated in Figure 3.8. We denote that buffer time corresponding to flight j by b_j. Basically, in the simplified versions of our models, we assume that the flights are equally important and therefore, $b_j = b$ for all flight j. In the modeling approach with flight dependent buffer time values, we basically replace constraints (3.54) and (3.61) by the following ones, respectively:

$$R_{j,s} \geq b_j - I_{j,s}, \quad j \in N^c \setminus \{0\},\ s \in S,$$

$$H_{j,s} b_j \geq b_j - I_{j,s}, \quad j \in N^c \setminus \{0\},\ s \in S.$$

Notice that these new versions of the constraints do not change the size of the problem formulations when the buffer times are given input parameters. However, introducing the buffer times as decision variables would increase the number of buffer time variables and depending on the additionally introduced constraints on the buffer time values, the total number of constraints may increase.

All the proposed stochastic gate assignment models are formulated as mixed-integer programming problems, which are hard to solve. In the next chapter, we propose tabu search heuristics in order to obtain reasonably "good" feasible solutions efficiently.

CHAPTER 4

TABU SEARCH HEURISTIC

The formulations proposed for the stochastic gate assignment models in Chapter 3 are hard to solve by using a standard mixed integer programming (MIP) solver such as CPLEX. In order to illustrate the computational challenges of solving the proposed formulations directly by using CPLEX, we present numerical results in Chapter 5. In practice, even if we cannot solve the problems to optimality, it is important to construct feasible solutions which would perform reasonably well. In this chapter, we propose tabu search (TS) heuristic algorithms which find reasonably "good" feasible solutions and describe the implementation details. We also provide the TS related parameter values used in our computational study.

4.1 Tabu Search Heuristic

Tabu search is a meta-heuristic method conceived by Glover [15,16] and has since been widely used to solve combinatorial optimization problems in the field of scheduling, routing, facility design, and so on. We refer the interested reader to the book by Glover and Laguna [17] and the references therein.

The main motivation of TS heuristic is to enable the search process to escape the trap of the local optimality. In order to achieve this, it allows climbing moves when no neighboring solution improves the previous best solution. Besides, unlike other search techniques, TS avoids to examine previously explored regions recurrently by keeping a tabu list. Tabu list includes the solutions that are considered in the short run. This list forbids some moves to avoid returning to the previous solution unless they satisfy some aspiration criterion.

The general flow of a TS heuristic can be described as follows: start with an initial solution. At each iteration, evaluate neighbor solutions and select the best solution in the neighborhood of the current solution until a termination criterion is met. Note that if this best solution is obtained as a result of a tabu move, check whether or

not the aspiration condition is satisfied. Aspiration condition describes a favorable circumstance under which even a tabu move is allowed to be made. After selecting the new solution, set the selected solution as the current solution and update the tabu list. If the selected solution improves the best solution so far also update the best solution.

4.1.1 Initial Solution

Recall that a feasible solution of any of our stochastic programming models has to satisfy the following constraints:

- each flight must be assigned to only one gate,
- dummy flights representing the gates should be assigned to the respective gates,
- no two conflicting flights are assigned to the same gate concurrently according to the planned schedule (e.g. planned arrival and departure times of flights)

In our TS implementation, we start the search procedure with a feasible solution. In particular, we find a feasible assignment $\mathbf{x}$ which satisfies the following constraints by using CPLEX:

$$\sum_{k \in M} x_{i,k} = 1, \quad i = 1, \ldots, n,$$

$$x_{i,k} = 1, \quad i = 0, (n+1), \quad k \in M,$$

$$\sum_{j \in L_{i,s}} x_{j,k} + x_{i,k} \leq 1, \quad i \in N^a,\ k \in M,\ s = 0.$$

Since the size of the feasibility formulation is quite small, we can easily find an initial feasible solution for all the proposed formulations in few seconds. Notice that in the feasibility formulation we consider the set N^a which is used in the conflict based models. The feasible solution obtained by considering N^a is augmented properly for the sets N^b and N^c to obtain feasible solutions for the idle time based models. To be precise, we extend the decision vector $\mathbf{x}$ by making the assignment of additional m dummy flights to the respective gates. Such an augmentation is possible due to the assumption that the gates are homogeneous.

4.1.2 Neighborhood Strategy

We use two types of neighborhood moves that are widely used in the literature. These moves can be explained as follows:

- Swap(i,j): Interchanges the gates that flight i and j are assigned (see Figure 4.1).
- Insert(i,k): Removes flight i from its current gate and assigns it to gate k (see Figure 4.2).

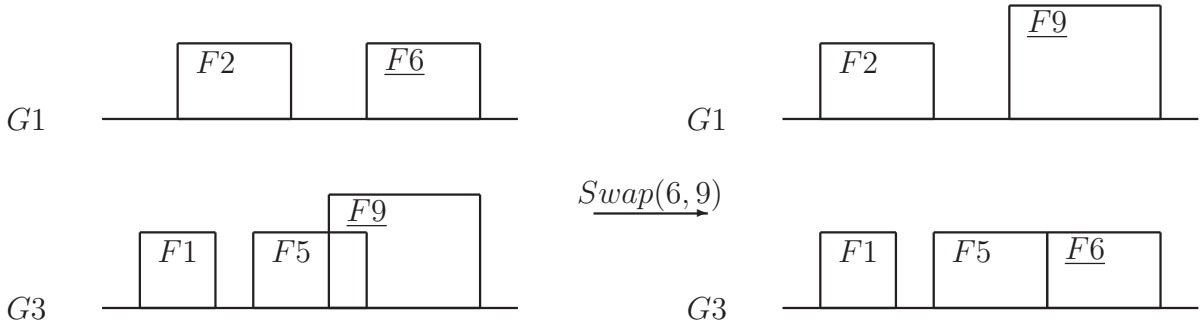

Figure 4.1: An illustrative example of swap move

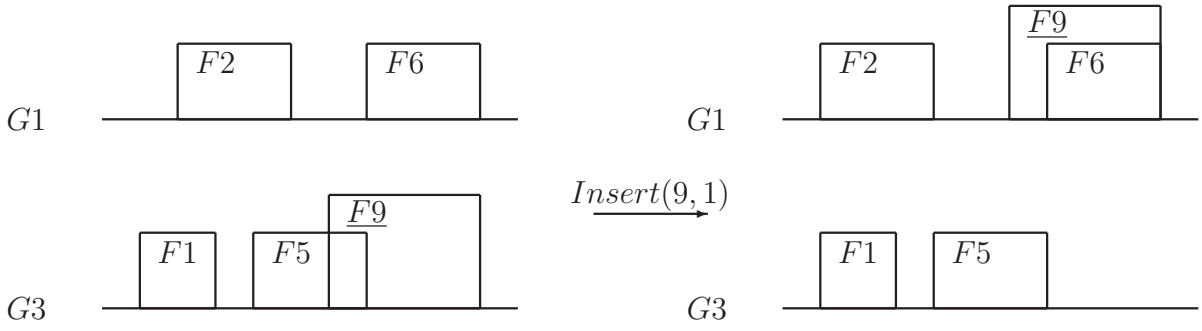

Figure 4.2: An illustrative example of insert move

We define three move functions for applying these two neighborhood moves; swap function, mutation function and insert function. The swap function is used to evaluate all neighbor solutions obtained by the swap moves and select the best neighbor solution, whereas the mutation function randomly leads to a swap move in the neighborhood of the current solution. Similar to the swap function, insert function evaluates all neighbor solutions and select the best one.

In our TS implementation, at each iteration we call the swap function and at every 3 iterations we call both swap and insert functions. Additionally, in order to force the algorithm to search the unexplored areas we use the mutation function. At every 50 iterations, we check whether the number of consecutive iterations without any improvement in the best solution exceeds 50 iterations. If it exceeds 50 iterations,

we call the mutation function and apply only a single randomly selected swap move regardless of the resulting objective function value.

It is quite computationally expensive to calculate the objective function value of the generated solution for idle time based models at each iteration. Therefore, we specify a model specific neighborhood structure for the swap function. Without loss of generality, it is assumed that the flights are sorted in ascending order of their arrival times according to the planned schedule (i.e. $i < j$ implies $a_{i,0} < a_{j,0}$). We consider a swap move between flights i and j if the difference between the indices i and j does not exceed 5 (i.e. $|i - j| \leq 5$). Exchanging the flights with arrival times that are closer to each other is expected not to effect the number of flight conflicts significantly and therefore, the restricted set of swap moves is expected to search among most of the promising solutions. In Chapter 5 the numerical results illustrate that considering the proposed restricted set of swap moves leads to a significant improvement in the computational times and provides us similar and even better solutions, as compared with the ones that are obtained by considering all the possible swap moves.

4.1.3 Solution Evaluation

To force the search into unexplored areas, a move is allowed even if it results in an infeasible solution in terms of the "conflict constraints". If the conflict constraints are violated for the planned schedule, we modify the objective function of the problem by adding the penalty function, $\beta K(\mathbf{x})$. Here $K(\mathbf{x})$ is the total number of flight conflicts calculated according to the planned arrival and departure times for a decision vector $\mathbf{x}$ and β is the penalty coefficient with an initial value of 1. If the assignment is feasible for the planned schedule then $K(\mathbf{x})$ is equal to zero. Every 5 iterations the penalty coefficient β is divided by 2 if all 5 previous solutions were feasible or multiplied by 2 if all were infeasible. This mechanism has perviously been used by Gendreau et al. [14] for the vehicle routing problem to diversify the search procedure.

4.1.4 Tabu List and Aspiration Condition

As a short term memory mechanism TS utilizes the tabu list. The tabu list includes some forbidden (tabu) moves that we cannot consider in the short run unless they satisfy some aspiration condition. We update the tabu list as the search progresses. It is quite important to define the tabu list size properly. As the list size increases we may not identify the local optimal solutions, in contrary to this smaller tabu list size

may cause cycling back to the previously discovered local optimal solutions. The vast majority of the literature suggests defining the tabu list size as an increasing function of the problem size. In our implementation, we define the tabu list size based on the number of flights. For the smallest problem instances that include 25 flights, we set the tabu list size as 5. The rate of increment for the number of flights is 25 for the remaining problem instances. We increase the list size by 3 for every additional 25 flights.

If the selected move is in the tabu list, then in order to except this move we should check whether the aspiration condition is satisfied. If this tabu move leads to a solution that has an objective function value strictly better than the best solution so far, then the aspiration condition is satisfied and this tabu move is excepted.

4.1.5 Termination Criteria

We terminate the search algorithm if one of the following termination criteria is met:

- maximum number of iterations ($iterlim$),
- maximum number of consecutive iterations without any improvement in the best solution so far ($consiterlim$),
- maximum computation time ($timelim$).

In our implementation, the maximum computation time ($timelim$) is defined as 7200 seconds. Similar to the tabu list size we define the maximum number of iterations ($iterlim$) and the maximum number of consecutive iterations without any improvement ($consiterlim$) based on the problem size as follows:

$$iterlim = 600 + 6n \text{ and } consiterlim = 200 + 2n,$$

where n denotes the number of flights.

We conduct a computational study to test the efficiency and effectiveness of the proposed tabu search algorithms and present the corresponding numerical results in the next chapter.

CHAPTER 5

COMPUTATIONAL STUDY

We conduct an extensive computational study to test the computational efficiency of the implemented tabu search heuristics, demonstrate the effectiveness of the proposed models, and to comparatively analyze the alternate models. In this chapter, we first explain the problem instance generation procedure in detail. Then in Section 5.2, we elaborate on the fact that solving the proposed formulations directly by using CPLEX is hard and present numerical results to illustrate the effectiveness of the proposed tabu search heuristics. Section 5.3 presents results to demonstrate how the proposed scenario-based stochastic programming formulations perform against the existing models proposed by Lim and Wang [21] and Bolat [5]. In Section 5.4 we provide numerical results to comparatively analyze the proposed models in terms of the alternate robustness measures.

5.1 Generation of problem instances

In order to test the computational performance of our solution methods, we considered several problem instances of different sizes. We generated two groups of data sets in order to show the performance of the proposed models appropriately. We say that the density level of the input data is high when the gates are generally fully-packed for the corresponding problem instance. The two groups of data sets are different mainly in terms of the density levels.

Data Set I

The first data set has 60 problem instances. In order to see the effectiveness of the proposed conflict-based stochastic models we prefer dense (fully-packed) inputs. Therefore, we limit the number of available gates to 8. We consider the same number of gates for all the problem instances, but we change the number of flights which also leads to the variations in the planning horizon. We summarize the details related to

this set of test instances as follows:

- **Generation of the planned schedule.**
  - Similar to Ding et al. [9] we assume that the interarrival and the ground times of flights are uniformly distributed. We randomly generate the arrival time and the ground time of flight i, $i \in N$, in the intervals $[10i, 10i + 15]$ and $[40, 60]$, respectively. Thus, a flight lands in every 10-25 minutes and remains at the airport between 40 and 60 minutes.
  - Recall that the gates are assumed to be homogeneous. Hence, the opening and closure times of the gates are same and by considering the gates as flights the associated opening and closure times are generated similar to the flight arrivals. In order to guarantee a feasible assignment for the planned schedule, we add the maximum ground time to the randomly generated closure time of a gate.
- **Generation of the scenarios.**

  Recall that scenarios represent the joint realizations of the arrival and departure times of the flights. We construct each scenario by generating random deviation amounts from the planned arrival times. In particular, we obtain the scenarios by deviating from the planned schedule in the following manner:
  - Flight delays (including early arrivals) are randomly generated from a triangular distribution with a negative skewness. According to the airport authorities the flight tardiness is more common than the flight earliness. Therefore, we prefer a negative-skewed distribution to obtain relatively less frequent smaller deviation values. The deviation amounts are assumed to range from -10 to 90 minutes, with the mode being 50 minutes.
  - Scenario probabilities are set to be equal.

  As we mentioned in Chapter 3, having more than one flight with a common arrival time causes miscalculations of the number of conflicts. Therefore, we perturb the recurring flight arrival times by a small constant such as 10^{-3}.

Data Set II

In order to observe the effectiveness of the idle time based stochastic models, we shall not consider dense problem instances like the ones in Data Set I. Therefore, we

increase the number of available gates to 12 to avoid fully-packed gates and generate additional 50 problem instances of different sizes using a similar approach described for the first data set.

We would like to point out that generating the scenarios is not our main concern here. Existing methods can be applied to generate alternate scenarios, or if available, the real historical data (e.g. historical delay patterns) may be employed.

All the proposed problems were modeled with the OPL mathematical programming language running on CPLEX 12.1 solver. The tabu search heuristics were coded in C programming language. The numerical experiments were performed on a 32-bit, 2 quad-core CPU HP Compaq desktop with 2.33GHz processor and 3.46GB of memory. All reported CPU times are in seconds. In our computational study, we terminate CPLEX when the prescribed CPU time limit (t = 7200 seconds) or the prescribed tree size limit (ts = 200 megabytes) is reached. We present computational results for the randomly generated problem instance families, each of which includes 5 problem instances with a specified set of parameters denoted by (n x m x $|S|$). We report the average results for each instance family. Note that instance families 1-12 belong to Data Set I and the remaining ones (13-22) belong to Data Set II.

Table 5.1 and Table 5.2 present the dimensional properties of the generated test problem instances. The size of the problem instances depend on both the model formulations and the structure of the generated input data. For example, the number of conflict related constraints increases as the density level of the input data gets higher.

			MENC		MMRNC	
Instance Family	Size (n x m x $\|S\|$)	Binary Variables	Total Variables	Constraints	Total Variables	Constraints
1	25 x 8 x 50	216	36,666	40,051	36,716	40,151
2	25 x 8 x 100	216	73,116	79,196	73,216	79,296
3	50 x 8 x 50	416	135,616	84,552	135,666	84,652
4	50 x 8 x 100	416	270,816	168,752	270,916	168,952
5	75 x 8 x 50	616	297,066	129,790	297,116	129,890
6	75 x 8 x 100	616	593,516	262,197	593,616	262,397
7	100 x 8 x 50	816	521,016	175,796	521,066	175,846
8	100 x 8 x 100	816	1,041,216	351,219	1,041,316	351,319
9	125 x 8 x 50	1,016	807,466	221,314	807,516	221,364
10	125 x 8 x 100	1,016	1,613,916	438,053	1,614,016	438,153
11	150 x 8 x 50	1,216	1,156,416	267,880	1,156,466	267,930
12	150 x 8 x 100	1,216	2,311,616	528,977	2,311,716	529,077

Table 5.1: Dimensions of the problem instance families for models MENC and MMRNC

Instance Family	Size (n x m x $\|S\|$)	METDNC			MENDNC			MEVINC		
		Binary Variables	Total Variables	Constraints	Binary Variables	Total Variables	Constraints	Binary Variables	Total Variables	Constraints
13	25 x 12 x 50	456	76,356	384,555	2,306	76,356	384,555	456	53,556	494,598
14	25 x 12 x 100	456	152,256	768,605	4,156	152,256	768,605	456	106,656	984,143
15	50 x 12 x 50	756	205,406	1,136,130	3,856	205,406	1,136,130	756	88,856	1,344,190
16	50 x 12 x 100	756	410,056	2,271,430	6,956	410,056	2,271,430	756	176,956	2,691,412
17	75 x 12 x 50	1,056	396,956	2,262,705	5,406	396,956	2,262,705	1,056	124,156	2,571,041
18	75 x 12 x 100	1,056	792,856	4,524,255	9,756	792,856	4,524,255	1,056	247,256	5,133,284
19	100 x 12 x 50	1,356	651,006	3,764,280	6,956	651,006	3,764,280	1,356	159,456	4,165,411
20	100 x 12 x 100	1,356	1,300,656	7,527,080	12,556	1,300,656	7,527,080	1,356	317,556	8,331,733
21	125 x 12 x 50	1,656	967,556	5,640,855	8,506	967,556	5,640,855	1,656	194,756	6,140,586
22	125 x 12 x 100	1,656	1,933,456	11,279,905	15,356	1,933,456	11,279,905	1,656	387,856	12,279,829

Table 5.2: Dimensions of the problem instance families for models METDNC, MENDNC, and MEVINC

5.2 Tabu Search Heuristics

The proposed problems are formulated as large mixed integer models (see Table 5.1 and 5.2), and it is hard to obtain good quality solutions within reasonable times by using a standard mixed integer programming solver such as CPLEX. We first present numerical results to demonstrate the computational challenge of solving the proposed formulations directly by CPLEX. As discussed in Chapter 4, we employ tabu search algorithms in order to find good feasible solutions in short CPU times. In this section, we present results to illustrate the comparative performance of the tabu search heuristic algorithms with respect to the "direct approach". In our study the "direct approach" refers to the approach of solving a proposed formulation directly by using CPLEX.

The proposed formulations for the vast majority of the generated problem instances cannot be solved to optimality within the prescribed time or tree size limits. Therefore, we obtain an upper bound on the relative optimality gap using the best known lower bound on the objective function value found by the branch-and-bound algorithm of CPLEX. Let Obf_t and Obf_{ts} denote the best lower bound on the objective function value that is provided by the CPLEX solver, when the prescribed time limit (t) or tree size limit (ts) is reached, respectively. Obf^*_t and Obf^*_{ts} denote the best available objective function value within the given limits, which define an upper bound on the objective value. Then, we define an upper bound on the relative optimality gap as follows:

$$\text{UBROG}_{t(ts)} = \frac{\text{Obf}^*_{t(ts)} - \text{Obf}_{t(ts)}}{\text{Obf}_{t(ts)}}.$$

Instance Family	Direct MENC		Direct MMRNC	
	CPU	UBROG	CPU	UBROG
1	88	0.00%	5558	10.34%
2	338	0.00%	7207	36.12%
3	2493	92.50%	7208	155.34%
4	2928	101.44%	N/A	N/A
5	4268	242.47%	$7217^{(4)}$	285.56%
6	$5207^{(1)}$	210.37%	N/A	N/A
7	7211	374.33%	N/A	N/A
8	$7204^{(4)}$	286.04%	N/A	N/A
9	7214	536.93%	N/A	N/A
10	N/A	N/A	N/A	N/A
11	$7213^{(3)}$	704.92%	N/A	N/A
12	N/A	N/A	N/A	N/A

Table 5.3: CPU times and UBROG for models MENC and MMRNC
N/A: No solution is available since CPLEX terminated due to solver error (ran out of memory).
$^{(k)}$: No solution is available for k problem instances out of 5.

According to Table 5.3, CPLEX could not even solve the moderate size problem

instances to optimality for the proposed model MENC. CPLEX provides a solution of the model MMRNC only for the small problem instances and cannot even extract the model for larger problem instances due to constraints (3.16). Note also that the calculated upper bounds on the optimality gaps are quite large for moderate and large problem instance families. However, we need to remark that the lower bounds provided by CPLEX are quite loose. Here we try to provide some informal arguments to justify our claim. Let us consider a problem instance from the instance family 3. CPLEX provides the following results for the model MENC: the best available objective function value found within the given limits is 9.12 (obtained within the predefined tree size limit), while the best lower bound on the objective function value is 4.10. For this lower bound value, UBROG_{ts} is equal to 126%, which we claim to be a very loose bound on the true relative optimality gap. For several instances we can argue that the actual relative optimality gap associated with the best integer solution found within the specified limits is significantly small than $\text{UBROG}_{t(ts)}$. In particular, for that specific test instance we informally show that the actual relative optimality gap is at most ten percent by solving the same model with an additional constraint bounding the objective function value. We solve the same problem with the following additional constraint bounding the objective function value of the model MENC:

$$\sum_{i \in N^a} \sum_{j \in N^a} \sum_{s \in S} c_{i,j,s} p_s \leq \frac{9.12}{(1 + 10\%)}.$$

After even approximately 60 million iterations and 24 hours of execution time, CPLEX could not provide a feasible solution for the model MENC with the above additional constraint. This would support the argument that the best available objective function value found so far is probably at most ten percent away from the optimal solution. Figure 5.1 displays the change of lower bound value (objective function value of the best known solution) for the test problem instance from instance family 3. From this figure it can be seen that after a certain number of iterations the number of updates on the lower bound value slows down dramatically and CPLEX provides very loose lower bound values for the proposed hard formulations.

For the idle time based models, the dimensions of the problem instances are larger (see Table 5.2). We did not report any CPU times and UBROG for the model MEVINC, since CPLEX terminated due to memory limit without constructing the branch-and-bound tree for all of the problem instance families. Moreover, we also state that

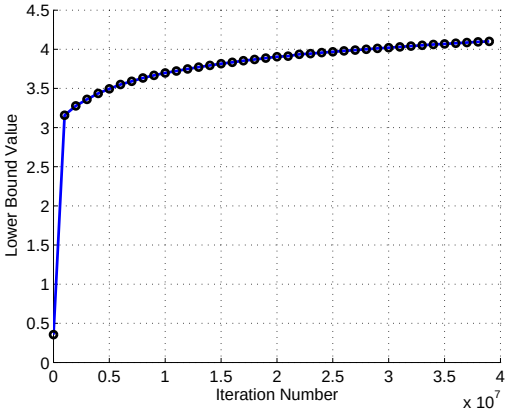

Figure 5.1: Change of the lower bound value

CPLEX requires very long CPU times even for the small problem instances of the models METDNC and MENDNC. In Table 5.4 we present results just for the smallest problem instance family, since it is the only family for which CPLEX provides solutions for the models METDNC and MENDNC.

Instance Family	Direct METDNC		Direct MENDNC	
	CPU	UBROG	CPU	UBROG
1	7203	1.56%	4434	1.55%

Table 5.4: CPU times and UBROG for models METDNC and MENDNC

After illustrating the computational challenge of solving the proposed formulations directly by CPLEX, we next compare the heuristic approach with the direct approach in terms of the solution quality (relative optimality gap) and the computation time. As discussed above, the best available lower bounds provided by CPLEX are in general quite loose. Therefore, for the solution obtained by the heuristic approach we prefer not to calculate an upper bound on the relative optimality gap using such a loose lower bound. Instead, for the solution obtained by the heuristic approach we calculate the relative improvement in the "objective function value (ofv)" with respect to the best solution found by the direct approach.

It is obvious from Tables 5.5 and 5.6 that the proposed heuristic approach produces better solutions than the branch-and-bound algorithm of CPLEX in quite less computation times. In these tables, we only report the relative reduction amounts in CPU, please see Tables 5.9 and 5.13 for the exact CPU times. Observe that for the model MENC, the heuristic algorithm also finds the optimal solutions for small problem instance families. In Table 5.5, we cannot provide relative results for some instances since CPLEX ran out of memory without constructing the branch-and-bound tree. Due to

Instance Family	Heuristic MENC		Heuristic MMRNC	
	Relative Improvement in		Relative Improvement in	
	Ofv (%)	CPU (%)	Ofv (%)	CPU (%)
1	0.00%(*)	97.81%	0.07%	99.97%
2	0.00%(*)	98.82%	0.17%	99.96%
3	2.70%	99.17%	3.24%	99.74%
4	0.07%	99.10%	8.55%	98.88%
5	3.53%	97.99%	N/A	N/A
6	2.84%	97.16%	N/A	N/A
7	2.62%	96.87%	N/A	N/A
8	1.93%	96.34%	N/A	N/A
9	5.23%	92.65%	N/A	N/A
10	N/A	N/A	N/A	N/A
11	2.37%	83.75%	N/A	N/A
12	N/A	N/A	N/A	N/A

Table 5.5: Effectiveness of the heuristics for models MENC and MMRNC
(*): Heuristic also yields optimal solutions.
N/A: No improvement percentage is reported since we do not have any result obtained by CPLEX (CPLEX ran out of memory).

Instance Family	Heuristic METDNC		Heuristic MENDNC	
	Relative Improvement in		Relative Improvement in	
	Ofv (%)	CPU (%)	Ofv (%)	CPU (%)
1	0.00%	99.61%	0.00%	96.60%

Table 5.6: Effectiveness of the heuristics for models METDNC and MENDNC

such CPLEX errors, we could not provide any relative improvement results for the model MEVINC and we present results just for the smallest instance family for the models METDNC and MENDNC.

5.3 Analyzing Alternate Models

In this section, we compare the proposed stochastic programming model (MENC) with the existing related model introduced by Lim and Wang [21] in terms of the number of conflicts associated with the solutions they provided. To make a fair comparison, we use the problem instances for which we can obtain the optimal solutions by solving the direct formulation of the model by Lim and Wang [21]. As we mentioned in Chapter 1, they use an unsupervised estimation function to estimate the probabilities of the flight conflicts based on a single planned schedule with deterministic arrival and departure times. Here in order to implement their formulation, we define alternate approaches to estimate the conflict probability associated with each flight pair for a given set of scenarios.

- Use the sample mean of the realized arrival and departure times to define a single planned schedule and estimate the conflict probabilities based on the unsuper-

vised estimation function presented by Lim and Wang [21].

- Estimate the conflict probabilities using the unsupervised estimation function for each scenario. Then calculate the weighted sum of the conflict probabilities, where the weights are taken as the associated scenario probabilities.

- For each flight pair we identify the scenarios for which the specific two flights are conflicting according to the realized arrival and departure times. Then the summation of the probabilities of those identified scenarios is used as the estimated conflict probability corresponding to that flight pair.

Note that in our computational study we employ the first suggested approach to estimate the conflict probabilities. According to Table 5.7, the model MENC utilizing the set of scenarios significantly reduces the expected number of flight conflicts for the considered problem instances.

Instance Family	% Improvement in E[C] by MENC
1	4.44%
2	3.56%

Table 5.7: Comparative results for MENC with respect to an existing model

We also want to compare our stochastic programming model MEVINC with the deterministic model proposed by Bolat [5]. Bolat [5] assumes that the arrival and departure times of the flights are deterministic. As a common approach we can estimate these parameters using the set of scenarios representing the given realizations of the random arrival and departure times; the sample mean of the realized arrival and departure times are used as point estimators. To make a fair comparison we also incorporate an additional constraint to satisfy the feasibility of the planned schedule as in our model MEVINC. Thus, we solve the deterministic model of Bolat [5] to obtain an assignment which is feasible for both the planned schedule and the schedule with the estimated arrival and departure times. In order to guarantee the existence of such a feasible solution, we modify several problem instance families from Data Set II by extending the gate closure times. We select four problem instance families 13-16 and refer to the modified versions of them by $13'$-$16'$. For these problem instance families involving 5 instances, we provide representative results in Table 5.8. It is well-known that the deterministic model does not take the variability of the random arrival and departure times into account. Therefore, as expected it provides solutions that result

in significantly larger numbers of flight conflicts under certain realizations of the input data compared to the solutions obtained by the model MEVINC. In other words, as seen in Table 5.8 the proposed deterministic model cannot deal with the random flight conflicts and fails to satisfy the primary objective of minimizing the expected number of flight conflicts. Therefore, there is no point to compare them in terms of the corresponding idle times.

Instance Family	% Improvement in E[C] by MEVINC
$13^{'}$	53.90%
$14^{'}$	36.08%
$15^{'}$	67.87%
$16^{'}$	40.48%

Table 5.8: Comparative results for MEVINC with respect to an existing model

5.4 Relative Results based on Alternate Formulations

We present some results indicating how alternate models perform according to different robustness measures. In Table 5.9 we report the CPU times of the tabu search heuristics proposed to solve the conflict-based models and provide comparative results on the absolute semi-deviation risk measure ($\rho[C]$) associated with the solutions obtained by such models. In order to demonstrate the effect of incorporating risk measure on the number of flight conflicts, we present the percentage improvement amounts in Table 5.9. As seen in the table, the model MMRNC provides more reliable solutions by considering the fluctuations on the number of conflicts. Note that the value of the risk coefficient (λ) can be defined by decision makers according to their risk preferences. Here we consider the values of λ as 0.5 and 1. As seen in Table 5.9 the higher λ parameters would represent more risk-averse preferences and provide more robust assignments.

In our study we propose three main robustness measures based on the idle and buffer times: the expected variance of the idle times (EVI), the expectation of the total deviation (ETD), and the expected number of deviations (END). The relative improvement amounts based on the EVI, ETD and END, are presented in Tables 5.10, 5.11 and 5.12, respectively. As seen from the presented results, the idle time based models approximately result in similar improvement amounts. This observation can be interpreted that we also achieve to distribute the idle times uniformly by using the alternate objectives of minimizing the expectation of the total deviation or minimizing

Instance Family	MENC	MMRNC (λ=0.5)		MMRNC (λ=1)	
	CPU	CPU	% Improvement in $\rho[C]$	CPU	% Improvement in $\rho[C]$
1	1.20	1.80	4.05%	1.40	16.84%
2	2.40	3.20	4.14%	2.60	10.05%
3	18.00	20.60	8.39%	18.80	13.04%
4	25.20	53.00	9.70%$^{(1)}$	26.00	13.06%
5	86.80	122.00	19.18%	69.00	32.20%
6	144.40	215.50	7.33%$^{(1)}$	168.00	13.24%
7	225.80	374.20	15.67%	267.80	31.75%
8	398.20	579.80	13.98%	420.80	19.41%
9	530.00	672.60	20.79%$^{(1)}$	420.40	35.53%
10	924.20	1417.40	16.64%	954.80	25.37%
11	950.80	1289.40	23.53%	830.00	35.98%
12	1724.00	1947.00	13.69%$^{(2)}$	1857.80	23.76%

Table 5.9: Comparative results of models MENC and MMRNC based on the CPU and risk value

$^{(k)}$: No improvement is obtained for k problem instances out of 5.

the expected number of deviations. Moreover, as expected, Tables 5.10-5.12 demonstrate that idle time based models result in significant improvements in the idle time related measures with respect to the base model MENC considering only the expected number of flight conflicts. Note also that increasing the number of flights leads to an increase in the number of alternative feasible solutions and in such cases we obtain solutions which perform even better in terms of the specified robustness measures.

According to Table 5.13 the heuristic algorithms for the idle time based models provide solutions which perform similar or even better in terms of the expected number of flight conflicts than the solutions obtained by the heuristic algorithm for the conflict based model MENC. Basically, the idle time based models aim to find the assignments which lead to the optimal expected number of conflicts and are the best in terms of the proposed secondary criteria: EVI, ETD and END. According to Tables 5.10-5.12, the improvement amounts based on these secondary performance measures are quite reasonable, since the number of alternative solutions with the same number of conflicts are limited. Recall that the proposed tabu search heuristics do not guarantee the optimal solutions. Therefore, all these observations are based on the best solutions provided by the heuristics and not on the optimal solutions.

The computational results show that we are able to solve the proposed problems even for large problem instances in reasonable computational times and we obtain assignments that perform well in terms of the defined robustness measures. Moreover, the results demonstrate that the proposed alternate robustness measures are consistent with the existing robustness measures such as the uniformity of idle times.

Instance Family	% Improvement in EVI by		
	MEVINC	METDNC	MENDNC
13	4.89%	4.53%	4.58%
14	1.31%	1.14%	1.13%
15	7.43%	7.69%$^{(1)}$	6.75%
16	2.89%	2.51%	2.68%
17	9.90%	6.94%	8.13%
18	6.55%	6.34%	5.97%
19	7.86%	9.09%	8.28%
20	5.18%	3.37%	4.59%
21	8.71%	8.87%	7.10%
22	5.26%	5.95%	5.02%

Table 5.10: Comparative results based on the EVI
$^{(k)}$: No improvement is obtained for k problem instances out of 5.

Instance Family	% Improvement in ETD by		
	MEVINC	METDNC	MENDNC
13	5.09%	6.66%	5.52%
14	3.50%	4.01%	3.79%
15	7.28%	7.17%	6.48%
16	5.24%	5.81%	5.25%
17	13.95%	12.15%	12.11%
18	5.57%	8.04%	5.74%
19	9.37%	13.26%	10.10%
20	5.98%	5.40%	5.80%
21	10.52%	12.74%	9.83%
22	5.85%	6.32%	5.21%

Table 5.11: Comparative results based on the ETD

Instance Family	% Improvement in END by		
	MEVINC	METDNC	MENDNC
13	5.85%$^{(1)}$	4.45%	6.03%
14	3.09%	3.57%	3.94%
15	6.79%	5.91%	8.49%
16	3.54%	4.16%	4.46%
17	10.11%	7.80%	10.31%
18	5.32%	6.21%	5.85%
19	6.77%	9.94%	9.69%
20	5.01%	3.68%	5.12%
21	7.27%	8.55%	8.59%
22	5.05%	4.97%	5.20%

Table 5.12: Comparative results based on the END
$^{(k)}$: No improvement is obtained for k problem instances out of 5.

Instance Family	MENC	MEVINC		METDNC		MENDNC	
	CPU	CPU	% Improvement in ENC	CPU	% Improvement in ENC	CPU	% Improvement in ENC
13	1	11	0.00%	28	0.00%	23	0.00%
14	2	24	0.00%	60	0.00%	54	0.00%
15	13	108	0.36%	245	0.24%$^{(1)}$	235	0.36%
16	30	239	0.67%	387	0.66%$^{(1)}$	350	0.66%$^{(1)}$
17	61	430	2.80%	1100	1.86%	1001	2.65%
18	126	960	3.06%	2369	3.14%	2442	3.19%
19	154	1185	3.91%	3111	4.21%	3033	3.91%
20	348	2509	3.26%	5812	3.53%	5043	3.16%
21	537	2242	5.90%	6715	5.60%	6950	6.09%
22	755	4213	3.48%	7211	4.14%	7205	3.34%$^{(1)}$

Table 5.13: Comparative results based on the ENC
$^{(k)}$: No improvement is obtained for k problem instances out of 5.

CHAPTER 6

CONCLUSION AND FUTURE RESEARCH

The uncertainties inherent in the flight arrival and departure times may lead the unavailability of gates when needed to accommodate scheduled flights. Therefore, real-time delays are quite crucial in constructing effective flight-gate assignment plans. The traditional deterministic gate assignment models neglect such random disruptions and so may perform poorly under certain realizations of the random arrival and departure times.

In this study, we develop new stochastic programming models to obtain robust gate assignments that are less sensitive to disruptions in the system. The main contribution of our study is developing alternate stochastic programming models that support a wider range of decision making preferences. We propose a risk-averse stochastic gate assignment model, where the trade-off between the expected number of flight conflicts and a risk measure on the random number of flight conflicts is considered. We also develop stochastic optimization models involving alternate robustness measures based on the idle and buffer time concepts. All proposed models are formulated as computationally expensive large-scale mixed-integer programs. It is hard to solve the developed formulations using a standard solver such as CPLEX. Therefore, our second main contribution is developing tabu search algorithms to obtain reasonably "good" feasible solutions. We conduct an extensive computational study to analyze the proposed models and illustrate the computational effectiveness of the tabu-search heuristics. The computational study shows that our models provide reasonably robust assignments and the proposed tabu search algorithms allow us to find good quality solutions in reasonable short CPU times, where CPLEX in general fails even to construct the branch-and-bound tree.

As a future work, the proposed tabu search heuristics can further be improved by using more refined neighborhood structures and including the long-term memory functions. The future research can also focus on incorporated the widely-used objectives

into our stochastic programming models. In our future research, we would also like to incorporate risk measures into the proposed risk-neutral models in order to model the effect of the variability of random input data.

Bibliography

[1] Babic O., Teodorovic C., Tosic V., Aircraft stand assignment to minimize walking, Journal of Transportation Engineering, 110, 55-66, 1984.

[2] Bihr R., A conceptual solution to the aircraft gate assignment problem using 0-1 linear programming, Computers and Industrial Engineering, 10, 280-284, 1990.

[3] Birge J., Louveaux F., Introduction to stochastic programming, Springer, New York, 1997.

[4] Bolat A., Assigning arriving aircraft flights at an airport to available gates, Journal of the Operational Research Society, 50, 23-34, 1999.

[5] Bolat A., Models and a genetic algorithm for static aircraft-gate assignment problem, Journal of the Operational Research Society, 52, 1107-1120, 2000.

[6] Bolat A., Procedures for providing robust gate assignments for arriving aircraft, European Journal of Operations Research, 120, 63-80, 2000.

[7] Cheng Y., A knowledge-based airport gate assignment system integrated with mathematical programming, Computers and Industrial Engineering, 32, 837-852, 1997.

[8] Cheng Y., A rule-based reactive model for the simulation of aircraft on airport gates, Knowledge-Based Systems, 10, 225-236, 1998.

[9] Ding H., Lim A., Rodrigues B., Zhu Y., New heuristics for over-constrained flight to gate assignments, Journal of the Operational Research Society, 55, 760-768, 2004.

[10] Ding H., Lim A., Rodrigues B., Zhu Y., The over-constrained airport gate assignment problem, Computers and Operations Research, 32, 1867-1880, 2005.

[11] Dorndorf U., Drexl A., Nikulin Y., Pesh E., Flight gate scheduling: State-of-the-art and recent developments, The International Journal of Management Science, 35, 326-334, 2007.

[12] Dorndorf U., Jaehn F., Lin C., Ma H., Pesch E., Disruption management in flight gate scheduling, Statistica Neerlandica, 61, 92-114, 2007.

[13] Drexl A., Nikulin Y., Multicriteria airport gate assignment and pareto simulated annealing, IIE Transactions, 40, 385-397, 2008.

[14] Gendreau M., Hertz A., Laporte G., A tabu search heuristic for the vehicle routing, Management Science, 40(10), 1276-1290, 1994.

[15] Glover F., Tabu Search part I, ORSA Journal on Computing 1, 3, 190-206, 1989.

[16] Glover F., Tabu Search part II, ORSA Journal on Computing 2, 1, 4-32, 1990.

[17] Glover F., Laguna M., Tabu Search, Kluwer Academic Publishers, Dordrecht, 1997.

[18] Haghani A., Chen M., Optimizing gate assignments at airport terminals, Transportation Research, 32A, 437-454, 1998.

[19] Hamzwawi S., Management and planning of airport gate capacity: a microcomputer-based gate assignment simulation model, Transportation Planning and Technology, 11, 189-202, 1986.

[20] Hassounah M., Steuart G., Demand for aircraft gates, Transportation Research Record, 1423, 26-33, 1993.

[21] Lim A., Wang F., Robust airport gate assignment, Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence, 2005.

[22] Lim A., Rodrigues B., Zhu Y., Airport gate scheduling with time windows, Artificial Intelligence Review, 24, 5-31, 2005.

[23] Mangoubi R., Mathaisel D., Optimizing gate assignments at airport terminals, Transportation Science, 19, 173-188, 1985.

[24] Markowitz H. M., Portfolio selection, Journal of Finance, 7(1), 77-91, 1952.

[25] Ogryczak W., Ruszczyński A., Dual stochastic dominance and related mean-risks models, SIAM journal of optimization, 13(2), 60-78, 2002.

[26] Pintea C., Pop P., Chira C., Dumitrescu D., A hybrid ant-based system for gate assignment problem, Lecture Notes in Computer Science: Springer Berlin, 5271, 273-280, 2008.

[27] Prékopa, A., Stochastic Programming, Kluwer Academic, Dordrecht, Boston, 1995.

[28] Xu J., Bailey G., The airport gate assignment problem: mathematical model and a tabu search algorithm, Proceedings of the 34th Hawaii International Conference on System Sciences, 2001.

[29] Yan S., Chang C., A network model for gate assignment, Journal of Advanced Transportation, 32, 176-189, 1998.

[30] Yan S., Huo C., Optimization of multiple objective gate assignments, Transportation Research, 35A, 413-432, 2001.

[31] Yan S., Shieh C.Y., Chen M., A simulation framework for evaluating airport gate assignments, Transportation Research, 36A, 885-898, 2002.

[32] Yan S., Tang C.H., A heuristic approach for gate assignments for stochastic flight delays, European Journal of Operational Research, 180, 547-567, 2007.