공학석사학위논문

Airport Gate Assignment Problem under Uncertainty

불확실성하의 공항 게이트 할당 문제

2021 년 2 월

서울대학교 대학원

산업공학과

구 병 주

Airport Gate Assignment Problem under Uncertainty

불확실성하의 공항 게이트 할당 문제

지도교수 이 경 식

이 논문을 공학석사 학위논문으로 제출함

2020 년 11 월

서울대학교 대학원

산업공학과

구 병 주

구병주의 공학석사 학위논문을 인준함

2020 년 12 월

위 원 장 _____ 홍 성 필 _____ (인)

부위원장 _____ 이 경 식 _____ (인)

위 원 _____ 문 일 경 _____ (인)

Abstract

Airport Gate Assignment Problem under Uncertainty

Byungju Goo
Department of Industrial Engineering
The Graduate School
Seoul National University

In this thesis, we study the airport gate assignment problem under uncertainty in arrival and departure times of flights which may cause overlaps between flights assigned to the same gate. In this respect, we establish a gate assignment schedule where overlaps between flights are prevented with probabilistic guarantees, while maximizing the sum of preference values of flights for gates. We propose a network-based integer programming model with chance constraints that limit the probability of an overlap occurrence in each gate less than a given threshold value, for which the probability that a pair of flights overlaps is predicted based on historical data. We also propose a strengthened integer programming model for the problem based on the concept of flight assignment patterns, and devise a branch-and-price algorithm to solve the model. Computational experiments are conducted with real and artificial instances which are based on the real data of Incheon International Airport in 2019. The results

show the efficiency of the proposed algorithm as well as the robustness of the derived schedules compared to the existing approaches.

Keywords: Airport gate assignment, Flight delay, Integer programming, Network flow model, Chance constraint, Branch-and-price algorithm

Student Number: 2019-28381

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

The airport gate assignment problem (AGAP) is one of the major decision problems in the airport industry that has been studied by researchers in the field of optimization [1]. In each gate in airports, each arrival or departure flight needs a ground service time for cleaning, maintenance, embarking or disembarking, etc. Therefore, in AGAP, flights are assigned to airport gates under certain constraints or objectives. It establishes a gate assignment schedule for given start and end times of ground service schedule of each flight, as illustrated in Figure 1.1. While the demand for air transport is large and growing, gates are valuable resources that are quite time-consuming and costly to build more. Therefore, a clever and efficient use of existing gates is required.

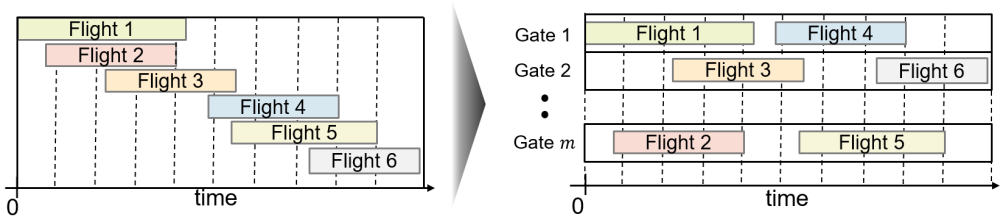

Figure 1.1: Illustration of AGAP

There have been various objectives and constraints considered in AGAP. For example, maximizing the sum of preference values of flights for gates [2, 3, 4], minimizing total passenger walking distance [5, 6, 7, 8] or minimizing the number of towing operations [2, 3, 7] have been considered as objectives. For constraints, flight-gate compatibility [3, 7, 9] which is caused by aircraft size or flight requirements keeps flights from being assigned to inappropriate gates. Adjacency constraints [3, 10, 11] prevent two large aircraft from being assigned to adjacent gates at the same time.

Above all, one of the most important issues in AGAP is the uncertainty of the start and end times of flight schedules. The uncertainty is caused by various factors such as violent weather conditions, staffing issues, delays in previous airports, incidents during ground services, etc.

Figure 1.2 illustrates an example that shows how uncertainty happens in flight schedules and causes overlaps of flight schedules in a gate assignment schedule. The original gate assignment schedule was made according to the nominal start and end times of flight schedules. However, on the day of the

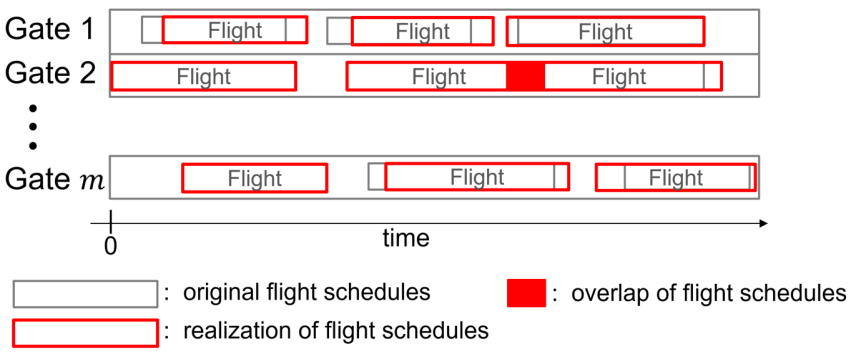

Figure 1.2: Illustration of an overlap in a gate assignment schedule

operation, the realizations of start and end times of flight schedules are changed from the nominal times due to the uncertainty, and an overlap of flight schedules happens in gate 2.

A schedule with overlaps is practically impossible to implement as it is, since one gate can handle only one flight at a time, and the schedule requires some real-time remedies for overlaps. For example, the following flight of the overlapping pair may wait for the assigned gate to be clear, which may cause a chain of delays for other flights assigned to the same gate. Also, a partial or complete rescheduling may be required. All these remedies require additional complex decision making and may have negative impacts on punctuality and passenger satisfaction as well as airport resources. Therefore, the uncertainty should be considered in the airport gate assignment in order to prevent overlaps. In this thesis, we focus on the uncertainty of the start and end times of flight schedules and the issue of preventing overlaps between flights in AGAP.

1.2 Literature Review

To prevent overlaps in AGAP, several studies modelled the uncertainty of flight schedules and considered the expectation of some measures associated with overlaps. Dorndorf et al. [3], Yan and Tang [12], and Seker and Noyan [13] all assumed that the ground service duration of each flight is fixed whereas its real start time is random. Dorndorf et al. [3] used independent gamma distributions to model the uncertainty of real start times of arrival flights. To deal with the uncertainty, they minimized the expected total number of overlaps. They discretized the gamma distributions with 1 minute intervals. The expectation is set to be two minutes early, the median to be six minutes early, and the earliest to be 30 minutes early. Yan and Tang [12] obtained the empirical distribution of real start times from real data of CKS Airport in Taiwan. They minimized the total expected penalty caused by wait times of passengers. About 88 percent of departing flight delays were 0 to 20 minutes, and for arriving flights, about 76 percent of the delays were -9 (early) to 20 minutes. They used a scenario-based model to maximize the weighted sum of preference values and the expected penalty caused by wait times of passengers. Seker and Noyan [13] used independent triangular distributions to model the uncertainty of real start times. They used the distribution with parameters -10, 50, and 90 which is left-skewed to reflect the fact that a delay is more likely than an earliness, and used a scenario-based model. They set the expected number of total overlaps as a primary objective to minimize, and compared different indirect measures as the second objectives to minimize: the expected variance of idle times, the

expected total semi-deviation of idle times, and the expected number of positive semi-deviation of idle times, where the term idle time indicates the nominal time interval between the schedules of consecutively assigned flights.

Yu et al. [6] assumed that the uncertainty lies in both start and end times of the flights. They modelled the uncertainty with independent log-normal distributions, and fitted the overlap probability function to an exponential function of the idle time. They used the expected total overlap duration as a robustness measure to minimize.

Lim and Wang [5], Aoun and El Afia [9], Castaing et al. [14], and Kim et al. [15] assumed that the uncertainty lies in each flight pair, and directly captured the uncertainty of each flight pair. Lim and Wang [5] and Aoun and El Afia [9] used estimation functions to estimate the expectation of the probability that a flight pair overlaps. The estimation functions reflected the fact that the larger idle time results in the smaller probability of overlap. They minimized the expected total number of overlaps. Kim et al. [15] modelled the expected overlap duration as a function of idle time. They fitted the historical data of overlap duration to an exponential function and minimized the expected total overlap duration. Castaing et al. [14] used historical data of overlaps of each flight pair to estimate the probability of overlap and the overlap duration. They used both the expected number of overlaps and the expected overlap duration as objectives, and additionally proposed minimizing worst case expected overlap duration.

There have been approaches that did not explicitly involve uncertainty in their models, but indirectly prevented overlaps in schedules by considering idle

times between flights. Benlic et al. [2] and Kumar and Beirlaire [16] captured short idle times between flights and penalized them. The idea is that short idle times are susceptible to small disruptions leading to overlaps, since an idle time works as a buffer to absorb unexpected changes. In the same sense, Daş [8] used the variance of idle times as a robustness measure and minimized the variance to distribute the idle times evenly.

Xu et al. [17] assumed that the start time of each flight is fixed whereas its real end time is random, and the distribution of each random end time is measured by a regression model, where the delay factors are classified into three classes: common factors, individual predictable factors and unpredictable factors. Unlike others stated above, where the ways of optimizing expectation regarding uncertain elements have been proposed, Xu et al. [17] proposed minimizing $(1-\alpha)-$quantile of total overlap duration to give a quantitative guarantee on the robustness of the schedules.

1.3 Motivations and Contributions

As we introduced in the previous section, most of the studies proposed optimization models considering the expectation of measures associated with overlaps. However, the derived schedules from these approaches may be difficult to implement in some cases depending on how uncertainty is realized. In other words, these approaches cannot give a quantitative guarantee on the implementation of schedules without overlaps. As far as we know, there have been no research that took into account the quantitative guarantee, except for Xu et al. [17]. However, they solve the model approximately which leads to a conservative solution. Also, the gate assignment schedules obtained from the model cannot guarantee the effectiveness of each gate, as the total overlap duration of all gates is considered. This can lead to an imbalance of overlaps among gates.

In this thesis, we propose a mathematical model that establishes a gate assignment schedule that gives a quantitative guarantee to the possibility of implementation, which has not been researched enough in previous studies. Specifically, we assume that the start and end times of ground service schedule for each flight are uncertain. Based on this assumption, we estimate the overlap probability of each pair of flights when they are assigned to the same gate. Based on this estimation, we propose an AGAP model, named overlap chance-constrained airport gate assignment problem (OCAGAP), with chance constraints that limit the probability of an overlap occurrence in each gate less than a given threshold value and maximize the sum of preference values, and also propose a solution approach based on column generation.

1.4 Organization of the Thesis

The remainder of the thesis is organized as follows. In Chapter 2, we present a formal description of the problem and propose mathematical models with chance constraints. In Chapter 3, a non-parametric prediction method of overlap probabilities based on empirical survival function is presented. In Chapter 4, we give a branch-and-price algorithm to solve the model we proposed. Chapter 5 is about computational experiments. We proposed detailed examination of the effectiveness of the model and the efficiency of the algorithm based on comparison with existing ones. Finally, a summary and concluding remarks appear in Chapter 6.

Chapter 2

Models for OCAGAP

In this chapter, we formally define OCAGAP that limit the probability of an overlap occurrence in each gate. Then we give an integer programming (IP) model for OCAGAP using network flow, and further strengthen the IP model using flight assignment patterns.

2.1 Problem Definition

The sets for OCAGAP are listed as follows:

- M : the set of normal (contact) gates. A gate $m \in M$ is a normal gate that can handle one flight at a time.
- A remote gate 0 : Unlike contact gates in the set M, the remote gate is assumed to have unlimited capacity, and more than one flight can be assigned to the gate simultaneously. Since passengers have to walk or ride a bus to move between the remote gate and the airport terminal, the preference value of every flight for the remote gate is set to be 0.
- N : the set of flights. A flight $i \in N$ is defined by its nominal start time $\bar{s}_i$ and nominal end time $\bar{e}_i$ where $\bar{s}_i < \bar{e}_i$. Without loss of generality, we

assume that the flights are sorted in ascending order of $\bar{s}_i$.

- $N(m) \subseteq N,\ \forall m \in M$: the set of flights that can be assigned to gate m, according to the compatibility.

Next, the parameters for OCAGAP are listed as follows:

- $p_{ij},\ \forall i, j \in N$: overlap probability between flights i and j. It indicates the probability that the schedules of flight i and j overlap given that the flights are assigned to the same gate.

- $\pi_{im}, \forall m \in M, \forall i \in N(m)$: preference value of flight i for gate m.

- $\epsilon_m,\ \forall m \in M$: the maximum limit of the probability that an overlap occurs in gate m.

In OCAGAP, a normal gate can handle only one flight at a time, so the inequality $\bar{e}_i \leq \bar{s}_j$ holds for flights i and j that are assigned sequentially to the same gate m. Also, the flight-gate compatibility constraints are considered, i.e., only the flights that belong to $N(m)$ can be assigned to gate m. The objective is to maximize the sum of preference values of flights for gates. Let F_m be the set of consecutively assigned flight pairs (i, j) in gate m. For each $(i, j) \in F_m$, let Y_{ij} be the binary random variable which is equal to 1 if flights i and j overlap. Then, F_m satisfies the following inequality for all $m \in M$:

$$P\left\{\sum_{(i,j)\in F_m} Y_{ij} \geq 1\right\} \leq \epsilon_m,\ \forall m \in M. \tag{2.1}$$

Constraints (2.1) are chance constraints that limit the probability of an overlap in gate m less than the given threshold value ϵ_m. We consider these chance constraints for all $m \in M$ in OCAGAP.

We now show the computational complexity of OCAGAP. It is obvious that the special case of OCAGAP with $\epsilon_m = 1$ for all $m \in M$ is deterministic AGAP whose objective is to maximize the sum of preference values. This problem is proven to be NP-hard by Jaehn [4]. Therefore, OCAGAP is NP-hard.

Remark. *OCAGAP is NP-hard.*

2.2 Integer Programming Models for OCAGAP

In this section, we give integer programming models of OCAGAP using the notation given in Section 2.1. The constraints and objective can be expressed on a network flow model by Maharjan and Matis [18].

The additional inputs for the network representation of gate m are listed as follows:

- s : start node.
- t : end node.
- $N'(m) \subseteq N$, $\forall m \in M$: the set of duplicate nodes $i' \in N$ of $N(m)$.
- V : the set of all nodes that consists of s, t, all nodes in $N(m)$, and all nodes in $N'(m)$.
- A : the set of all arcs that consists of (s, i) for all $i \in N(m)$, (i, i') for all $i \in N(m)$ and $i' = i$, (i', j) for all $i' \in N'(m), j \in N(m)$ such that $\bar{e}_{i'} \leq \bar{s}_j$, and (i', t) for all $i' \in N'(m)$.

Figure 2.1 illustrates an example of the network. The set of compatible flights $N(m)$ is assumed to be $\{1, 2, \ldots, n\}$ here. The network consists of nodes i corresponding to flights and their duplicated nodes i'. Dummy nodes s and t are the start and end nodes of a path. Node i is connected to its duplicated node i', and node i' is connected to nodes j of flights that can come right after the flight i in the gate. Node s is connected to all nodes i and node t is connected from all nodes i'. Then, a path from node s to t represents an

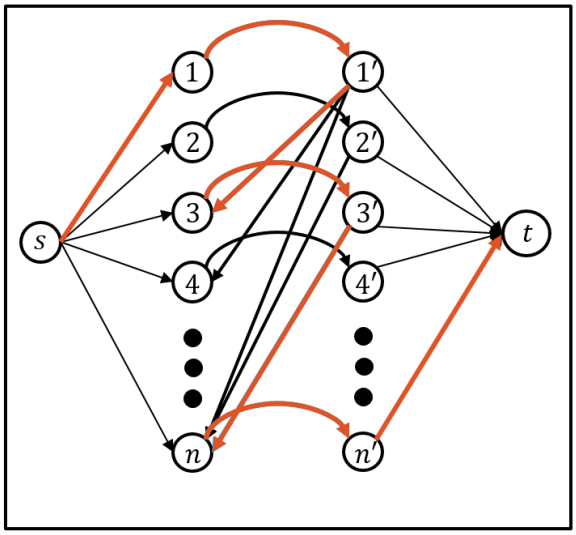

Figure 2.1: Network representation of an assignment schedule of a gate

assignment schedule of a gate that comprises of the flights that the path passes by. For instance, the red path in Figure 2.1 represents an assignment schedule of a gate with flights $1, 3$, and n.

A gate assignment schedule is thus represented by the set of $|M|$ paths in which each flight node $i \in N$ is passed by at most one path. Flights that are not assigned to any path are assigned to the remote gate.

2.2.1 Compact Model for OCAGAP

Based on the network, OCAGAP without considering uncertainty using chance constraints is formulated with an integer programming model defined as follows:

$$\text{maximize} \quad \sum_{m \in M} \sum_{i \in N(m)} \pi_{im} x^m_{ii'} \tag{2.2}$$

$$\text{subject to} \quad \sum_{m \in M \cup \{0\}} x^m_{ii'} = 1, \quad \forall i \in N, \tag{2.3}$$

$$\sum_{i \in N(m)} x^m_{si} = 1, \quad \forall m \in M, \tag{2.4}$$

$$\sum_{j \in \delta^+(i)} x^m_{ij} = \sum_{j \in \delta^-(i)} x^m_{ji}, \quad \forall m \in M, \forall i \in N(m) \cup N'(m), \tag{2.5}$$

$$\sum_{i \in N(m)} x^m_{it} = 1, \quad \forall m \in M, \tag{2.6}$$

$$x^m_{ij} \in \{0, 1\}, \quad \forall m \in M, \forall i, j \in N(m) \cup N'(m) \cup \{s, t\}. \tag{2.7}$$

The decision variables x^m_{ij} correspond to the arcs from node i to j in the network. In terms of the gate assignment schedule, $x^m_{ii'} = 1$ implies that flight i is assigned to gate m and $x^m_{i'j} = 1$ implies that flight i is followed by flight j in gate m. The objective function (2.2) is to maximize the sum of preference values of the schedule. Constraints (2.3) state that each flight i must be assigned to a gate. Constraints (2.4)-(2.6) are the flow balance constraints for the networks of the gates.

With the decision variables x^m_{ij}, the chance constraint (2.1) can be restated

as follows:

$$P\left\{\sum_{i,j\in N(m)} Y_{ij}x^m_{i'j} \leq 0\right\} \geq 1-\epsilon_m,\ \forall m \in M.$$

Also, it is obvious that the following holds:

$$P\left\{\sum_{i,j\in N(m)} Y_{ij}x^m_{i'j} \leq 0\right\} = P\left\{Y_{ij}x^m_{i'j} = 0,\ \forall i,j \in N(m)\right\}$$

Note that for consecutive flight pairs (i,j) and (j,k) in F_m, the following relationship holds:

$$P\{Y_{ij}=0, Y_{jk}=0\} = P\{Y_{jk}=0|Y_{ij}=0\}P\{Y_{ij}=0\}$$

We assume that an overlap between flights i and j does not affect the overlap probability between flights j and k, i.e., Y_{ij} does not affect Y_{jk}, as buffer times in flight schedules can absorb the affect of overlaps. Under the assumption, the following holds:

$$P\{Y_{jk}=0|Y_{ij}=0\} = P\{Y_{jk}=0\}$$

Generalizing these observations, the chance constraints (2.1) can be expressed as follows:

$$\prod_{i,j\in N(m)} P\{Y_{ij}x^m_{i'j}=0\} \geq 1-\epsilon_m, \forall m \in M.$$

Since $P\{Y_{ij} = 0\} = 1 - p_{ij}$, the following holds:

$$\prod_{i,j \in N(m)} \left(1 - p_{ij} x_{i'j}^m\right) \geq 1 - \epsilon_m, \forall m \in M.$$

Then, the chance constraints (2.1) can be linearized by taking logarithms on both sides of the inequality above as follows:

$$\sum_{i,j \in N(m)} \log\left(1 - p_{ij}\right) x_{i'j}^m \geq \log(1 - \epsilon_m), \quad \forall m \in M. \tag{2.8}$$

The constraints (2.8) are what we call the overlap probability chance constraints. Then, the compact model for OCAGAP is defined as follows:

$$\mathbf{C:} \quad \text{maximize} \quad (2.2) \tag{2.9}$$

$$\text{subject to} \quad (2.3) - (2.7), (2.8). \tag{2.10}$$

2.2.2 Pattern-based Model for OCAGAP

We additionally propose a pattern-based model for OCAGAP. Let a flight assignment pattern for gate m be a set of flights that can be assigned together to gate m by satisfying the overlap probability chance constraints and the compatibility constraints. Then for each gate $m \in M$, the set of flight assignment patterns can be defined as Q_m. Note that each $q \in Q_m$ can be expressed as an $|N|$ dimensional binary vector, $\bar{x}^q = (\bar{x}^q_1, ..., \bar{x}^q_{|N|})$ where $\bar{x}^q_i = 1$ when q includes flight i. Note that for any pattern q, $\bar{e}_i \leq \bar{s}_j$ holds for all $i < j \in N$ such that $\bar{x}^q_i = \bar{x}^q_j = 1$. Using this notation, the pattern-based model for OCAGAP is defined as follows:

$$\mathbf{P:} \quad \text{maximize} \quad \sum_{m \in M} \sum_{q \in Q_m} \left(\sum_{i \in N(m)} \pi_{im} \bar{x}^q_i \right) z^q_m \tag{2.11}$$

$$\text{subject to} \quad \sum_{m \in M \cup \{0\}} \sum_{q \in Q_m} \bar{x}^q_i z^q_m = 1, \quad \forall i \in N, \tag{2.12}$$

$$\sum_{q \in Q_m} z^q_m \leq 1, \quad \forall m \in M, \tag{2.13}$$

$$z^q_m \in \{0, 1\}, \quad \forall m \in M, \quad \forall q \in Q_m. \tag{2.14}$$

The binary decision variable z^q_m equals to 1 if gate m chooses flight assignment pattern q. The objective function (2.11) is to maximize the sum of preference values of the schedule. Constraints (2.12) imply that each flight i is assigned to a gate. Constraints (2.13) state that each gate m chooses at most one pattern.

2.2.3 Comparison of the OCAGAP Models

We close this section by comparing the proposed models for OCAGAP. Since there exist exponentially many flight assignment patterns, the number of variables in the pattern-based model is exponential in $|N|$, whereas the number of variables in the compact model is polynomial in $|M|$ and $|N|$. However, the pattern-based model has an upper bound provided by the linear programming (LP) relaxation at least as tight as that of the compact model. Let the upper bound provided by the LP relaxation of the pattern-based model and the compact model be Z^p_{LP} and Z^c_{LP}, respectively. Then, the following theorem holds.

Theorem 2.1. $Z^p_{LP} \leq Z^c_{LP}$.

Proof. We show that a feasible solution of the pattern-based model can be transformed into a feasible solution of the compact model. Since each flight assignment pattern corresponds to a simple path from s to t in the network representation, the feasible solution of the pattern-based model can be interpreted as a linear combination of simple paths. It is clear that this solution is feasible for the compact model. Therefore, a feasible solution of the pattern-based model can be transformed into a feasible solution of the compact model, and $Z^p_{LP} \leq Z^c_{LP}$. □

Chapter 3

Prediction of Overlap Probabilities

In this chapter, we study how to determine the overlap probabilities in OCA-GAP. We first define the overlap probability function, and propose a non-parametric prediction method with which we predict the overlap probability of flights using historical data.

3.1 Definition of Overlap Probability Function

We denote the nominal start and end times of flight i by $\bar{s}_i$ and $\bar{e}_i$, and assume that the start and end times of flights are uncertain due to the unexpected events such as violent weather conditions, staffing issues, delays in previous airports, incidents during ground services, etc. Due to the uncertainty, real start and end times of the flights deviate from the nominal start and end times, and the deviations are denoted by $\hat{s}_i$ and $\hat{e}_i$. Without loss of generality, we assume that $\hat{s}_i$ and $\hat{e}_j$ are independent random variables for different flights $i, j \in N$. An overlap occurs between consecutively assigned flights i and j in one gate if the ground service of flight i ends later than the start time of flight j. The overlap

probability p_{ij} is then set as follows:

$$p_{ij} = P\{\bar{s}_j - \bar{e}_i \leq \hat{e}_i - \hat{s}_j\}.$$

Note that $\bar{s}_j - \bar{e}_i$ is the nominal idle time between flights i and j. Let $\bar{s}_j - \bar{e}_i = t$. Then the overlap probability when the nominal idle time is t is defined as a function $f(t) = P\{t \leq \hat{e}_i - \hat{s}_j\}$. We name $f(t)$ an overlap probability function.

Figure 3.1 illustrates an outline of the graph of $f(t)$ between departure and arrival flights, which is drawn from the historical data of flights in Incheon International Airport in 2019. Specifically, for all days in 2019, we collected every pair of flights i and j on the same day where i is a departure flight and j is an arrival flight. For each $t \in \mathbb{Z}$, $f(t)$ is is assumed to be the proportion of flight pairs that would have overlapped if they had been assigned to the same gate, i.e., $\bar{s}_j - \bar{e}_i \leq \hat{e}_i - \hat{s}_j$, where $\mathbb{Z}$ is the set of integers.

Note that $f(t)$ is similar to a survival function which gives the probability that some object will survive beyond any specified time [19]. If we regard a

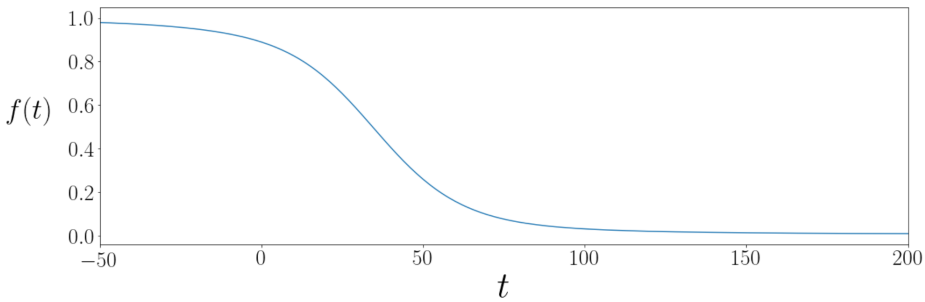

Figure 3.1: An outline of $f(t)$ between departure and arrival flights estimated with flight data in 2019

random variable $\hat{e}_i - \hat{s}_j$ as a random lifespan of an object, the survival function of the object indicates the probability that the object will survive beyond t. The survival function is then defined as $P\{t \leq \hat{e}_i - \hat{s}_j\}$, which is same with the definition of $f(t)$ stated above. Based on this observation, we propose a non-parametric prediction method based on empirical survival function to predict $f(t)$ of the target day.

3.2 Prediction Method based on Empirical Survival Function

Suppose that we consider the dataset of flights in d days just before the target day, and we use the dataset to predict $f(t)$ of the target day. Our speculation is that the uncertainty of flights is affected as time passes, and the uncertainty in the latest data reflects current uncertainty better than the older data. Let the dataset be T_d. Each flight i in T_d has $(\bar{s}_i, \bar{e}_i)$ with realized $(\hat{s}_i, \hat{e}_i)$. Then we can generate samples of $\hat{e}_i - \hat{s}_j$ by calculating $\Delta t_{ij} = \hat{e}_i - \hat{s}_j$ for each $i, j \in T_d$. Let the number of samples be $\mathcal{N}$. Then the empirical survival function can be defined as follows:

$$\hat{f}_{\mathcal{N}}(t) = \frac{1}{\mathcal{N}} \sum_{\{i,j\} \subset T_d} I(\Delta t_{ij} \geq t)$$

where $I(\Delta t_{ij} \geq t)$ is 1 if $\Delta t_{ij} \geq t$, and 0 otherwise. Note that this estimator converges to $f(t)$ derived from the population in T_d asymptotically [20], i.e., $\sup_{t \in \mathbb{R}} |\hat{f}_{\mathcal{N}}(t) - f(t)| \to 0$ for sufficiently large $\mathcal{N}$, where $\mathbb{R}$ is the set of real numbers. However, our purpose is to predict $f(t)$ derived from the population in target day. To justify the use of this estimator as a predictor, in the next section, we conduct a case study for real data of Incheon International Airport in 2019. In the case study, we measure the accuracy of the predictor and also give a recommendation of parameter d.

3.3 Case Study

In this section, we conduct a case study using the real flight data of Incheon International Airport in 2019. We first categorized the flights into 8 different types for finer prediction of $f(t)$: 4 different classes of airlines (Korean Air, Asiana Air, the rest of the Korean airlines, and foreign airlines) for arrival or departure flights. Flight pairs are classified into a total of 64 types according to the combination of the types of two flights in a pair, and $f(t)$ is predicted separately for each type of the flight pairs. For each flight in the data, we are provided only with nominal and real start times for arrival flights, and nominal and real end times for departure flights. Therefore, we assumed that $\hat{s}_i = \hat{e}_i$ for all $i \in N$, and that the ground service duration of each flight is 30 minutes which is the mean value considered in Yu et al. [6]. Combining these information, we derived nominal and real start and end times of ground service schedule for each flight. For example, if flight i is a departure flight and its given nominal and real end times are 13:00 and 13:20, then the nominal and real start times are assumed to be 12:30 and 12:50. We disregard flights with $\hat{s}_i < -60$ or $\hat{s}_i > 120$ as abnormal situations which account for less than 1% of total data.

To give a recommendation on parameter d and measure the accuracy of prediction, we conduct predictions on June and July data. With different d, we computed the predictor $\hat{f}_{\mathcal{N}}(t)$ for each type of the flight pair on each day. For example, if $f(t)$ in June 1st is predicted with $d = 7$, the prediction is based on the flight data from May 25th to 31st. Since the true form of $f(t)$ is not given, we regard the empirical survival function of target day as the $f(t)$ from the

asymptotic convergence property. Since the unit time of $\hat{s}_i$ and $\hat{e}_i$ is 1 minute in the data and we disregarded flights with $\hat{s}_i < -60$ or $\hat{s}_i > 120$, we measure the root mean squared error (RMSE) of two functions as follows:

$$\text{RMSE} = \sqrt{\frac{1}{361} \sum_{t=-180}^{180} (\hat{f}_{\mathcal{N}}(t) - f(t))^2}.$$

Figure 3.2 shows the boxplots of average RMSE of 64 types of predictions for each day in June and July for each d. At first, the RMSE tends to decrease as d increases, but at certain point, the benefit of using larger sample with larger d is offset by the increase of bias by including more outdated data. As a result, the minimum average RMSE of 0.02244 and the minimum median RMSE of 0.02108 is obtained when $d = 35$. Also, the standard deviation is 0.02108, which is the 4th best value. What we recommend is thus using d around 35 or one month, and we used $d = 35$ for the computational experiments.

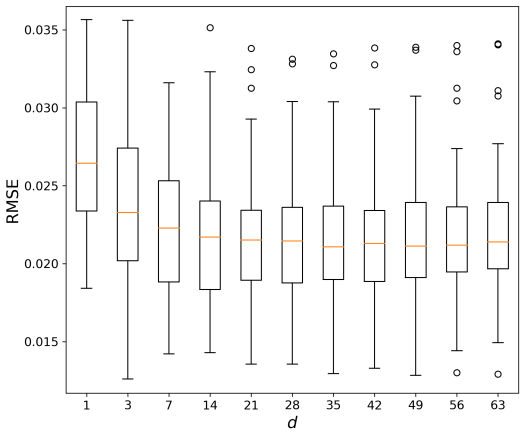

Figure 3.2: Boxplots of RMSE for various d

Chapter 4

Solution Approach for Pattern-based Model

As we mentioned in Section 2.2.3, the pattern-based model has exponentially many variables while it has an advantage that it has LP relaxation bound at least as tight as that of the compact model. To take advantage of the pattern-based model, we devise a branch-and-price algorithm (B&P) based on the column generation method [21, 22] which is practically useful when solving LP problems with a large number of variables.

4.1 Column Generation Method for LP Relaxation

The pattern-based model, named master problem, is solved in a branch-and-bound framework. At each node of the branch-and-bound tree, we solve the LP relaxation of the master problem using the column generation method. Since the pattern sets Q_m have exponentially many patterns, we consider a master problem where the pattern sets Q_m are substituted by the restricted pattern sets $\hat{Q}_m \subset Q_m$ for all $m \in M$. The problem is called restricted master problem (RMP) and defined as follows:

$$\textbf{RMP :} \quad \text{maximize} \quad \sum_{m \in M} \sum_{q \in \hat{Q}_m} \left(\sum_{i \in N(m)} \pi_{im} \bar{x}_i^q \right) z_m^q \tag{4.1}$$

$$\text{subject to} \quad \sum_{m \in M \cup \{0\}} \sum_{q \in \hat{Q}_m} \bar{x}_i^q z_m^q = 1, \quad \forall i \in N, \tag{4.2}$$

$$\sum_{q \in \hat{Q}_m} z_m^q \leq 1, \quad \forall m \in M, \tag{4.3}$$

$$z_m^q \in \{0, 1\}, \quad \forall m \in M, \quad \forall q \in \hat{Q}_m. \tag{4.4}$$

Upon solving the LP relaxation of RMP, we use the dual optimal solutions u_i and v_m of the constraints (4.2) and (4.3) to solve the sub-problems. A flight assignment pattern that improves current objective value of RMP can be generated by solving a sub-problem for each gate. The sub-problem SP(m) for gate m is defined as follows:

$$\textbf{SP}(m) \textbf{ :} \quad \text{minimize} \quad \sum_{i \in N} (u_i - \pi_{im}) x_{ii} - v_m \tag{4.5}$$

$$\text{subject to} \quad \sum_{i \in N(m)} x_{si} = 1, \tag{4.6}$$

$$\sum_{j \in \delta^+(i)} x_{ij} = \sum_{j \in \delta^-(i)} x_{ji}, \quad \forall i \in N(m) \cup N'(m), \tag{4.7}$$

$$\sum_{i \in N'(m)} x_{it} = 1, \tag{4.8}$$

$$\sum_{i \in N(m)} \log(1 - p_{ij}) x_{i'j} \geq \log(1 - \epsilon_m), \tag{4.9}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j \in N(m) \cup N'(m) \cup \{s, t\}. \tag{4.10}$$

If the optimal objective value of SP(m) is greater than 0, the corresponding flight assignment pattern is profitable, and it is added to $\hat{Q}_m$ to improve the objective value of RMP. The column generation method for master problem is iterated to solve the LP relaxation of RMP and SP(m) for each $m \in M$ until no flight assignment pattern is added to $\hat{Q}_m$ for all $m \in M$. Then, the optimal LP relaxation solution of the master problem is obtained.

4.2 Branching Variable

If the optimal solution of the LP relaxation of the master problem is not integral, it is not a feasible solution of the master problem, and we need to branch the node to search for integral solution. Note that a feasible solution $\hat{z}$ of the master problem can be translated into a feasible solution $\hat{x}$ of the LP relaxation of the compact model as follows:

$$\hat{x}_{ii'}^{m} = \sum_{q \in \hat{Q}_m} \bar{x}_i^q \hat{z}_m^q, \quad \forall i \in N(m), \quad \forall m \in M.$$

Branching on the variables in the pattern-based model itself may lead to slow convergence of the algorithm because of the large number of variables. Therefore, in this branch-and-price algorithm, we branch on the most fractional variable $\hat{x}_{ii'}^{m}$ translated from $\hat{z}$. Flight i is assigned to gate m in its left child node, i.e., $x_{ii'}^{m} = 1$, while the assignment is not allowed in its right child node, i.e., $x_{ii'}^{m} = 0$. Note that this branching rule just fixes some variables $x_{ii'}^{m}$ to 0 or 1 in the sub-problems. Therefore, additional constraints are not required in the sub-problems for each branched node and its successors, and the structure of the sub-problems is not destroyed.

Further, we have the following proposition that ensures that the branching rule we choose gives an integer solution with respect to the variables in the pattern-based model.

Proposition 4.1. *If $\hat{z}_m^q$ is fractional, then there exists $i \in N(m)$ with fractional*

$\hat{x}^m_{ii'}$. *[21]*

Proof. Let F be the set of indices of patterns with fractional $\hat{z}^q_m$ and assume that there is no fractional $\hat{x}^m_{ii'}$. We assume that there are at least two patterns in F since if $F = \{q\}$, then $\hat{x}^m_{ii'}$ is fractional for every i such that $\bar{x}^q_i = 1$. Since $\sum_{q \in Q_m} \hat{z}^q_m \leq 1$ in (2.13), we have $\sum_{q \in F} \hat{z}^q_m \leq 1$. Then, $\sum_{q \in F} \bar{x}^q_i \hat{z}^q_m \leq \sum_{q \in F} \hat{z}^q_m \leq 1$ for $i \in N$. If $\sum_{q \in F} \bar{x}^q_i \hat{z}^q_m = 0$, then $\bar{x}^q_i = 0$ for all $q \in F$. If $\sum_{q \in F} \bar{x}^q_i \hat{z}^q_m = 1$, then $\bar{x}^q_i = 1$ for all $q \in F$. It means that the patterns in F are same, which is a contradiction. $\square$

4.3 Initialization

An initial feasible solution gives a lower bound that helps reduce the size of the branch-and-bound tree. Since RMP is always feasible as gate 0 is a remote gate that can handle unlimited number of flights simultaneously, a feasible solution can be derived by solving RMP with any kind of patterns in the restricted pattern sets. Therefore, we obtain a feasible solution by solving RMP after generating a number of columns and using them as initial columns. Specifically, at node 0 of the branch-and-bound tree, an initial LP is solved with the column generation method, and a number of patterns are generated. Then, before branching, we apply a typical branch-and-bound algorithm to obtain the optimal feasible solution of RMP with the restricted pattern set. It is a feasible solution of the master problem, and the initial feasible solution gives an initial lower bound of the optimal solution that helps reduce the size of the search tree. Finally, the flowchart for the overall branch-and-price algorithm is described in Figure 4.1, where S denotes the set of active nodes in the branch-and-bound tree, and lb and ub denote the lower and upper bound respectively.

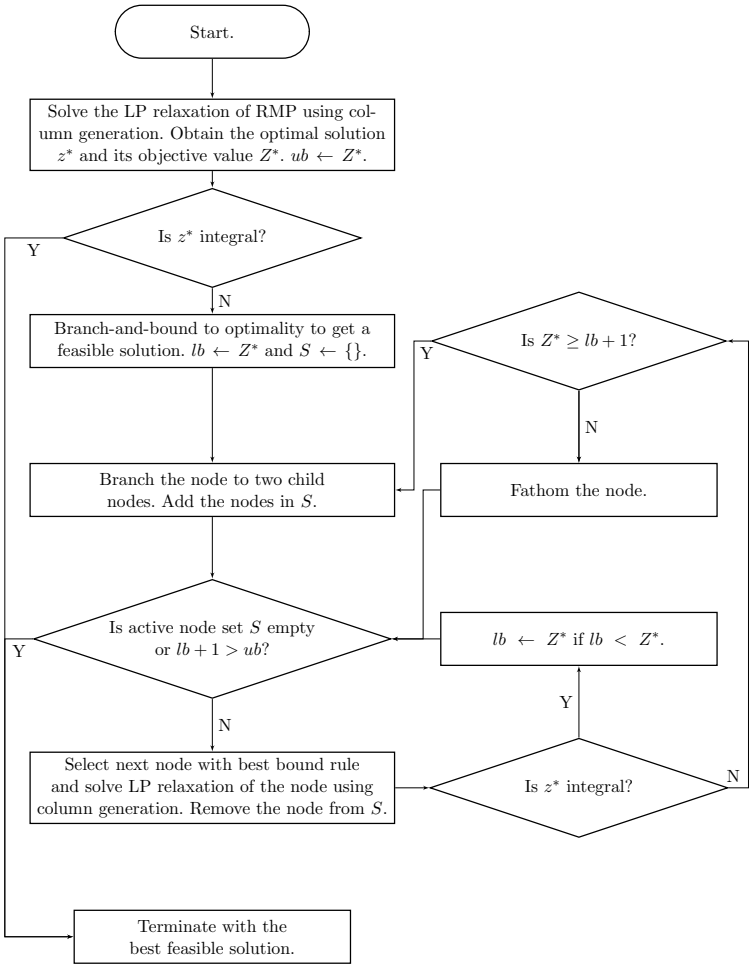

Figure 4.1: Overall procedure of the branch-and-price algorithm

Chapter 5

Computational Results

Computational tests are performed to evaluate the efficiency of the proposed algorithm and the effectiveness of the gate assignment schedules obtained from OCAGAP. We compare the branch-and-price algorithm to a commercial MIP solver to test the efficiency, and then we compare the gate assignment schedules obtained from OCAGAP to the schedules obtained an existing AGAP model to test the effectiveness. Proposed models and algorithm were implemented with version 8.9 of Xpress [23]. We choose the best-bound rule for node selection in the branch-and-price algorithm, and also, the sub-problems for all gates were solved simultaneously using parallel processing. All tests were performed on a PC with Intel Core 3.60 GHz processors and 32 GB RAM. The computational time is limited to 600 seconds. Notation of the result tables is given as follows:

- $\%Gap$: the relative gap between the LP relaxation bound Z_{LP} and the optimal objective value Z^*. If no optimal solution is found within the time limit, the value of the best feasible solution found is used instead.

$$\%Gap = \frac{Z_{LP} - Z^*}{Z^*} \times 100$$

- $\#Node$: the number of nodes in the branch-and-bound tree
- $\#Col$: the number of columns generated in the branch-and-price algorithm
- $Time$: the computational time until the optimal solution is found. If the optimality is not achieved within the time limit, then $Time = 600$.
- $\#Opt$: the number of optimal solutions among 10 instances in the configuration
- $Date$: the date of the instance in August 2019.
- Obj : the best objective value found
- $\#RG$: the number of flights assigned to the remote gate in the gate assignment schedule
- $\#OG$: the number of gates with at least one overlap in the gate assignment schedule obtained
- $\#OF$: the number of overlapping flight pairs in the gate assignment schedule obtained
- OD : the total overlap duration in minutes in the gate assignment schedule obtained

5.1 Implementation Issue of Overlap Probability

Since the logarithms $\log(1 - p_{ij})$ in the linearized chance constraints (2.8) can be irrational and numerical errors arise when computing with irrational numbers, we need to determine how to deal with the logarithms in the models for OCAGAP before implementation. We deal with them by slightly overestimating the overlap probabilities, which is same as underestimating the logarithms, in order not to compromise the feasibility of the constraints. Specifically, we use natural logs for the logarithms, and round the logarithms up to three decimal places. Therefore, the chance constraints (2.8) are modified as follows:

$$\sum_{i,j \in N(m)} \frac{\lfloor 1000 \log(1 - p_{ij}) \rfloor}{1000} x^m_{i'j} \geq \log(1 - \epsilon_m), \quad \forall m \in M. \tag{5.1}$$

To analyze the errors induced by the changed constraints, let $\bar{p}_{ij}$ be the overestimated overlap probability for p_{ij}. That is,

$$\log(1 - \bar{p}_{ij}) := \frac{\lfloor 1000 \log(1 - p_{ij}) \rfloor}{1000}. \tag{5.2}$$

Then, the following holds:

$$\log(1 - \bar{p}_{ij}) > \log(1 - p_{ij}) - 0.001. \tag{5.3}$$

Putting both sides of the inequality to the power of the natural constant e yields the following:

$$1 - \bar{p}_{ij} > \frac{1 - p_{ij}}{e^{0.001}}. \tag{5.4}$$

The inequality is organized in terms of $\bar{p}_{ij}$ as follows:

$$\bar{p}_{ij} < 1 - \frac{1}{e^{0.001}} + \frac{p_{ij}}{e^{0.001}}. \tag{5.5}$$

Note that $1 - \frac{1}{e^{0.001}}$ is slightly smaller than 0.001 and $\frac{p_{ij}}{e^{0.001}} \leq p_{ij}$. Moreover, $p_{ij} \leq \bar{p}_{ij}$ according to the definition of $\bar{p}_{ij}$. Thus, the following relationship holds:

$$p_{ij} \leq \bar{p}_{ij} < p_{ij} + 0.001. \tag{5.6}$$

The inequality above indicates that the error of the overlap probability caused by rounding is smaller than 0.001. In other words, each overlap probability is overestimated by at most 0.001, which is quite small and negligible.

5.2 Computational Comparison of Proposed Models

To compare the computational performances of proposed models, we generate artificial instances from real data of Incheon International Airport in August 2019. For each instance, we first choose $|M|$ gates where $|M| \in \{5, 10\}$, and randomly choose $|N|$ flights where $|N| \in \{60, 80, 100, 120\}$ that are compatible with the chosen gates, with their airline types, compatible gates, and the nominal start and end times in minutes in the real data, and we generate 10 instances for each combination of $|M|$ and $|N|$. For each gate, we distinguished whether or not a given flight is compatible with the gate depending on the airline of the flight; we regarded the flights of the airlines that have never assigned a flight to the gate in 2019 as incompatible with the gate.

For each instance, we conducted experiments for each of the three different types of preference values π_{im}, since computational performances are dependent on the preference values. For the first type, each preference value is set to be the average number of passengers carried by the flights of the airline with the same destination or origin according to the real data in 2019. Therefore, the objective is to maximize the number of passengers assigned to the contact gates. The second type is to set every preference value to 1. The objective is thus to maximize the number of flights assigned to the contact gates. The last type is to randomly generate the preference values from the set $\{1, 2, 3\}$. Flights can have various degrees of preferences for the gates depending on the aircraft size, number of passengers, airline, origin or destination, etc. We assume that the preference values can be classified into three classes: 1, 2, and 3. We randomly

give one of the three values for each combination of a flight and a gate. The objective is thus to maximize the sum of preference values of flights for gates.

For simplicity, we assume that every deviation of start and end times of the flights is generated from an identical probability distribution. For the probability distribution, we chose gamma distribution, which is used in Dorndorf et al. [3] to model the uncertainty of the flight schedules. We specifically used the deviations of start and end times of all the flights handled in Incheon International Airport fitted to the gamma distribution. Before fitting, we disregard flights with $\hat{s}_i < -60$ or $\hat{s}_i > 120$ as abnormal situations which account for less than 1% of total data, and since gamma distributions are defined for positive numbers, we added 60 to each deviation. The resulting distribution is $\Gamma(12.7, 0.18)$; the shape parameter of the distribution is 12.7, and the rate parameter is 0.18. We assume that we are provided with the parameters of underlying gamma distribution of the flight delays, so we construct the overlap probabilities p_{ij} from the convolution of the two independent and identically distributed gamma distributions.

The pattern-based model is solved by the branch-and-price algorithm we proposed. The compact model is solved by a branch-and-bound algorithm with the same variable branching and node selection strategy. The average results of the 10 instances for each $|M|$ and $|N|$ are given except for the column $\#Opt$.

The results when the objective is to maximize the number of passengers assigned to the contact gates are reported in Table 5.1 and Table 5.2. $\%Gap$ results show that the LP relaxation value of the pattern-based model is not only tighter than that of the compact model, but also close to the optimal

Table 5.1: Computational comparison when maximizing the number of passengers ($|M| = 5$)

$\|M\|$	$\|N\|$	ϵ	Pattern-based					Compact			
			$\%Gap$	$\#Node$	$\#Col$	$Time$	$\#Opt$	$\%Gap$	$\#Node$	$Time$	$\#Opt$
5	60	0.05	0.087	157.0	382.3	11.9	10	1.463	2.1	0.6	10
		0.1	0.070	95.9	338.7	9.0	10	2.353	24.7	1.1	10
		0.15	0.108	171.9	438.5	15.3	10	1.632	12.5	1.2	10
		0.2	0.096	107.9	427.6	13.3	10	1.424	10.9	1.3	10
	80	0.05	0.061	160.3	328.6	23.1	10	1.844	41.4	1.8	10
		0.1	0.032	14.5	239.5	5.6	10	1.865	24.2	1.8	10
		0.15	0.056	247.5	565.3	64.2	10	1.899	453.2	3.4	10
		0.2	0.069	401.8	719.2	90.7	9	1.738	303.6	3.5	10
	100	0.05	0.038	10.2	197.3	4.5	10	1.413	26.4	2.3	10
		0.1	0.033	9.2	242.0	7.4	10	1.614	327.8	3.6	10
		0.15	0.074	324.7	685.0	73.7	10	1.523	896.2	5.7	10
		0.2	0.035	46.3	403.3	25.2	10	1.433	1,670.0	6.4	10
	120	0.05	0.064	81.3	233.7	19.3	10	1.621	127.1	3.6	10
		0.1	0.051	187.7	397.4	69.0	9	1.674	1,014.0	5.8	10
		0.15	0.040	192.8	427.6	69.0	9	1.384	1,947.5	8.5	10
		0.2	0.021	6.7	232.8	12.8	10	1.254	2,178.4	9.6	10

Table 5.2: Computational comparison when maximizing the number of passengers ($|M| = 10$)

$\|M\|$	$\|N\|$	ϵ	Pattern-based					Compact			
			$\%Gap$	$\#Node$	$\#Col$	$Time$	$\#Opt$	$\%Gap$	$\#Node$	$Time$	$\#Opt$
10	60	0.05	0.000	119.2	774.8	9.0	10	1.343	0.9	0.6	10
		0.1	0.000	141.9	894.8	10.6	10	1.361	0.9	0.5	10
		0.15	0.000	180.4	910.4	15.3	10	1.161	1.0	0.8	10
		0.2	0.000	284.1	1,140.7	22.1	10	1.493	0.9	0.9	10
	80	0.05	0.051	336.2	1,470.4	74.1	10	1.814	11.9	2.5	10
		0.1	0.021	186.7	1,287.5	69.3	9	1.796	162.2	2.9	10
		0.15	0.033	340.7	1,770.4	89.0	9	1.447	34.5	3.5	10
		0.2	0.065	379.7	2,214.4	120.5	9	1.519	225.1	3.5	10
	100	0.05	0.110	752.9	2,584.5	364.2	4	1.574	2,385.4	35.4	10
		0.1	0.062	444.5	1,951.0	205.1	7	1.587	1,461.0	20.3	10
		0.15	0.141	505.8	3,058.6	370.6	4	1.598	7,225.3	117.3	9
		0.2	0.054	624.2	3,397.7	382.2	4	1.320	2,292.1	33.5	10
	120	0.05	0.103	913.2	2,429.3	408.7	4	1.498	9,090.4	125.3	9
		0.1	0.119	796.5	2,997.8	484.9	2	1.477	20,040.4	139.7	9
		0.15	0.082	559.6	2,861.5	430.9	3	1.292	10,321.3	182.5	9
		0.2	0.095	349.9	2,959.8	382.6	4	1.232	10,643.0	163.6	9

objective value. However, in spite of the difference in the LP relaxation bounds, the pattern-based model spent more computational time in general because of the column generation procedure. Therefore, the compact model generally gave optimal solutions in short times compared to the pattern-based model. Moreover, when $|M| = 10$, the pattern-based model could not give optimal solutions for most of the instances compared to the compact model.

The results when the objective is to maximize the number of flights assigned to the contact gates are reported in Table 5.3 and Table 5.4. The overall results are similar to the results of the first experiment where the objective is to maximize the number of passengers assigned to the contact gates. $\%Gap$ of the pattern-based model is smaller than that of the compact model, but the column generation procedure needed a significant amount of computational time. Therefore, the compact model generally gave optimal solutions in short times compared to the pattern-based model, but the difference between the two models were smaller than in the results in Table 5.1 and Table 5.2. Also, the numbers of instances that the optimal solutions are found are similar in the two models.

The results when the preference values are 1, 2, or 3 are presented in Table 5.5 and Table 5.6. $\%Gap$ results show that the LP relaxation value of the pattern-based model is far tighter than that of the compact model, and they are close to the optimal objective value. As a result,the pattern-based model searched significantly less nodes in the branch-and-bound tree compared to the compact model, when every instance reached the optimal solution, especially in large instances when $|N| = 100$ and $|N| = 120$ at both $|M|$. Overall, the

Table 5.3: Computational comparison when maximizing the number of flights ($|M| = 5$)

$\|M\|$	$\|N\|$	ϵ	Pattern-based					Compact			
			$\%Gap$	$\#Node$	$\#Col$	$Time$	$\#Opt$	$\%Gap$	$\#Node$	$Time$	$\#Opt$
5	60	0.05	0.287	26.4	272.4	2.9	10	2.829	39.2	0.8	10
		0.1	0.465	90.1	473.1	6.6	10	3.070	7.6	0.7	10
		0.15	0.524	87.6	639.3	11.6	10	2.802	1,011.2	3.1	10
		0.2	0.444	181.2	788.4	30.6	10	1.949	328.7	1.5	10
	80	0.05	0.167	9.4	157.3	2.3	10	2.337	44.7	1.5	10
		0.1	0.541	255.1	540.4	31.4	10	2.541	411.3	2.3	10
		0.15	0.519	42.3	366.2	10.1	10	2.399	1,158.3	5.4	10
		0.2	0.650	50.6	457.8	11.5	10	2.369	115.7	2.1	10
	100	0.05	0.414	51.7	310.3	11.4	10	2.749	737.0	4.8	10
		0.1	0.600	51.2	385.1	13.8	10	2.352	2,079.0	4.8	10
		0.15	0.262	170.0	646.0	45.3	10	2.635	26,825.1	43.4	10
		0.2	0.227	457.1	887.5	104.3	9	2.134	21,063.5	29.8	10
	120	0.05	0.333	21.8	170.5	6.3	10	2.653	664.6	4.6	10
		0.1	0.278	37.8	209.2	9.3	10	2.262	1,607.0	5.8	10
		0.15	0.241	13.9	181.9	8.1	10	2.182	974.1	4.5	10
		0.2	0.641	80.4	374.1	21.8	10	2.559	73,687.6	86.0	9

Table 5.4: Computational comparison when maximizing the number of flights ($|M| = 10$)

$\|M\|$	$\|N\|$	ϵ	Pattern-based					Compact			
			$\%Gap$	$\#Node$	$\#Col$	$Time$	$\#Opt$	$\%Gap$	$\#Node$	$Time$	$\#Opt$
10	60	0.05	0.000	40.1	638.6	2.6	10	1.358	1.0	0.7	10
		0.1	0.000	46.7	685.5	3.0	10	0.993	0.9	0.8	10
		0.15	0.000	48.5	726.5	3.3	10	0.835	1.0	1.0	10
		0.2	0.000	49.3	727.6	3.4	10	0.985	1.0	0.8	10
	80	0.05	0.000	68.3	1,039.1	11.1	10	1.798	73.5	3.0	10
		0.1	0.000	89.5	1,109.2	9.7	10	2.310	42.8	3.5	10
		0.15	0.000	118.0	1,345.2	15.5	10	1.047	840.2	10.8	10
		0.2	0.000	134.6	1,390.8	15.3	10	0.879	1.0	3.4	10
	100	0.05	0.193	190.2	1,871.9	83.7	10	1.607	11,739.2	160.5	9
		0.1	0.225	290.5	2,688.9	125.3	9	2.016	313.3	16.5	10
		0.15	0.310	341.6	2,570.6	175.5	8	1.692	10,901.6	232.9	7
		0.2	0.228	393.9	3,784.6	245.5	7	1.842	19,263.4	393.0	5
	120	0.05	0.332	296.4	2,045.0	171.0	9	1.560	23,020.7	388.0	6
		0.1	0.691	557.0	4,142.8	461.4	3	1.559	21,364.3	292.8	6
		0.15	0.604	386.9	3,785.9	364.7	6	1.436	8,872.1	335.3	7
		0.2	0.598	381.8	4,235.9	349.8	7	1.291	5,736.5	295.5	6

Table 5.5: Computational comparison when $\pi \in \{1, 2, 3\}$ ($|M| = 5$)

$\|M\|$	$\|N\|$	ϵ	Pattern-based					Compact			
			$\%Gap$	$\#Node$	$\#Col$	$Time$	$\#Opt$	$\%Gap$	$\#Node$	$Time$	$\#Opt$
5	60	0.05	0.031	0.8	122.4	0.7	10	2.296	0.8	0.4	10
		0.1	0.084	1.7	164.9	0.8	10	2.442	0.6	0.4	10
		0.15	0.030	0.3	154.8	1.0	10	2.118	0.4	0.5	10
		0.2	0.129	1.6	204.9	1.3	10	2.027	0.2	0.5	10
	80	0.05	0.230	3.2	111.7	1.1	10	2.483	3.3	0.8	10
		0.1	0.081	1.4	109.5	0.9	10	2.276	0.5	0.5	10
		0.15	0.000	1.1	121.4	1.3	10	2.500	8.5	1.2	10
		0.2	0.016	1.9	140.9	1.7	10	1.873	0.8	0.8	10
	100	0.05	0.045	0.9	94.3	1.1	10	2.247	2.0	1.3	10
		0.1	0.035	1.4	111.3	1.5	10	1.947	13.9	1.4	10
		0.15	0.102	1.1	126.4	2.1	10	2.201	74.0	2.1	10
		0.2	0.154	5.6	173.6	3.5	10	2.129	366.6	2.8	10
	120	0.05	0.074	3.4	89.7	1.8	10	2.360	13.2	2.1	10
		0.1	0.049	1.2	87.5	1.6	10	1.992	22.6	1.9	10
		0.15	0.089	0.5	101.9	2.0	10	1.689	45.1	2.4	10
		0.2	0.111	1.5	118.6	2.6	10	1.742	86.5	2.7	10

Table 5.6: Computational comparison when $\pi \in \{1, 2, 3\}$ ($|M| = 10$)

$\|M\|$	$\|N\|$	ϵ	Pattern-based					Compact			
			$\%Gap$	$\#Node$	$\#Col$	$Time$	$\#Opt$	$\%Gap$	$\#Node$	$Time$	$\#Opt$
10	60	0.05	0.041	12.7	499.7	2.1	10	1.589	1.0	0.5	10
		0.1	0.000	5.0	442.6	1.8	10	1.365	1.0	0.5	10
		0.15	0.000	9.5	557.6	2.5	10	1.134	1.0	0.5	10
		0.2	0.040	15.0	560.8	2.8	10	1.004	1.0	0.5	10
	80	0.05	0.076	4.9	502.5	3.4	10	1.631	0.9	1.3	10
		0.1	0.115	10.5	687.1	5.5	10	2.072	0.8	1.7	10
		0.15	0.122	17.7	830.8	8.7	10	1.558	4.2	2.8	10
		0.2	0.128	15.1	853.5	8.0	10	1.338	1.3	1.5	10
	100	0.05	0.254	26.6	673.7	11.3	10	1.945	497.1	12.0	10
		0.1	0.313	58.0	1,104.4	25.8	10	2.150	55.1	8.6	10
		0.15	0.265	64.5	1,161.7	34.9	10	1.827	1,121.2	15.6	10
		0.2	0.301	38.4	1,126.1	24.0	10	1.831	362.0	14.4	10
	120	0.05	0.396	146.7	1,071.4	58.8	10	2.193	5,059.5	80.5	9
		0.1	0.316	97.5	1,069.8	44.7	10	1.991	2,397.0	66.2	10
		0.15	0.368	308.6	2,054.0	153.0	10	1.931	5,369.2	134.2	9
		0.2	0.313	109.0	1,674.1	81.7	10	1.792	5,552.8	144.8	9

number of nodes in the branch-and-bound tree, the number of columns generated in the branch-and-price algorithm, and the computation time spent were significantly less than in the previous experiments. Both models could give optimal solutions for almost every instance. In the previous two experiments that maximized the number of passengers or flights assigned to the contact gates, any flight has an identical preference value for every contact gate, whereas in this experiment, a flight is allowed to have different preference values for the different gates. It means that the preferable flight-gate pairs are more easily distinguished. Therefore the computations were easier with the preference values in this experiment.

Since almost every instance reached its optimal solution within the time limit when the preference values are 1, 2, or 3, an additional experiment is performed for larger instances with $|M| = 10$ and $|N| \in \{140, 160, 180, 200\}$, and the results are reported in Table 5.7. $\%Gap$ results show that the LP relaxation value of the pattern-based model is tighter than that of the compact model and close to the optimal objective value. Due to the gap, the number of nodes in the branch-and-bound tree to obtain the optimal solution is significantly less in the pattern-based model than in the compact model. Compact model could not converge to the optimal solutions within the time limit in most of the instances where the pattern-based model gave the optimal solutions. These results of $Time$ and $\#Opt$ are opposite to the results where the objective is to maximize the number of passengers assigned to the contact gates, which shows that computational performances are highly dependent on the choice of the objective function.

Table 5.7: Computational comparison when $\pi \in \{1, 2, 3\}$ (larger instances)

$\|M\|$	$\|N\|$	ϵ	Pattern-based					Compact			
			$\%Gap$	$\#Node$	$\#Col$	$Time$	$\#Opt$	$\%Gap$	$\#Node$	$Time$	$\#Opt$
10	140	0.05	0.351	85.0	847.2	51.5	10	2.084	13,565.0	114.3	10
		0.1	0.411	204.6	1,311.4	123.6	9	1.952	24,056.2	214.2	9
		0.15	0.302	102.0	1,234.0	87.5	10	1.737	18,050.0	153.3	10
		0.2	0.380	138.2	1,757.3	131.7	10	1.735	29,264.5	256.1	8
	160	0.05	0.239	52.0	587.5	43.5	10	2.012	18,980.8	148.3	10
		0.1	0.324	142.6	1,268.8	130.3	10	1.914	21,874.8	243.9	9
		0.15	0.254	153.8	1,266.2	143.8	9	1.721	19,672.0	238.1	8
		0.2	0.228	128.0	1,394.4	137.0	10	1.579	43,917.7	387.1	7
	180	0.05	0.193	28.7	463.5	31.0	10	1.874	20,028.0	189.3	10
		0.1	0.224	180.3	894.5	136.4	9	1.868	33,455.5	313.5	7
		0.15	0.184	120.1	832.7	112.1	9	1.774	38,113.5	446.5	3
		0.2	0.243	172.0	1,328.7	180.3	9	1.670	45,961.9	532.7	2
	200	0.05	0.156	21.5	340.6	26.6	10	2.009	29,683.3	412.2	5
		0.1	0.165	74.6	636.8	91.5	10	1.743	21,981.2	398.2	5
		0.15	0.127	76.7	679.4	98.9	9	1.617	34,887.3	440.3	5
		0.2	0.151	61.1	804.0	104.7	10	1.781	30,412.6	531.4	2

5.3 Effectiveness of Gate Assignment Schedules Obtained from OCAGAP

To evaluate the effectiveness of the gate assignment schedules obtained from OCAGAP, we compare them to the schedules obtained from deterministic AGAP model with buffer time (BAGAP). Mangoubi and Mathaisel [24] proposed using a fixed buffer time between two consecutive flights to absorb the deviations of flight schedules. We put buffer time b in start and end times for each flight, i.e., each flight's nominal start and end times $\bar{s}_i$ and $\bar{e}_i$ are changed to $\bar{s}_i - b$ and $\bar{e}_i + b$ respectively. Therefore, the nominal idle time between consecutively assigned flights is at least $2b$ in the schedule obtained from BAGAP.

5.3.1 Tests on Real Instances

In practice, the gates can be classified into several groups according to their locations. Furthermore, specific airlines can be assigned to each group of gates. To simulate more realistic situation in the experiment, we select 5 gates in the western side of the terminal 1 in Incheon International Airport and the corresponding airlines as an instance for each day from August 1 to August 20. The set of flights N of the instance is the flights that are handled in the gate that day. The minimum, maximum, and average size of the flight sets $|N|$ of the 20 instances is 46, 60, and 53.2. We predict $f(t)$ with the data of the 35 latest days before each day according to the result in Figure 3.2. For example, for the instance in August 1, we predict $f(t)$ with the empirical survival function based on the dataset of flights from June 27 to July 31. The preference values are set

to be the numbers of passengers as in Section 5.2.

Table 5.8 and Table 5.9 shows Obj, $\#RG$, $\#OG$, $\#OF$ and OD obtained from solving the pattern-based model of OCAGAP. With smaller ϵ, OCAGAP gives a schedule with less $\#OG$, $\#OF$ and OD while the number of passengers assigned to the contact gates is decreased and the number of flights assigned to the remote gate is increased. For each schedule, the expected number of gates with overlaps is $|M|\epsilon$, which is 5ϵ in the instances, and the average $\#OG$ in the result are similar to the expectations. It indicates that the number of gates with overlaps is predictable with historical data.

Also, Obj, $\#RG$, $\#OG$, $\#OF$ and OD of the schedules obtained from BAGAP are reported in Table 5.10 and Table 5.11. The schedules with buffer time $b = 0$ shows the largest objective values, but overlaps take place in a lot of gates for every instance, and the schedules are impossible to be implemented as they are because of the overlaps. If the buffer time b is increased, resulting $\#OG$, $\#OF$ and OD is decreased, while Obj becomes larger and $\#RG$ becomes smaller.

Figure 5.1 illustrates the average $\#OG$ and Obj of the schedules obtained from OCAGAP and BAGAP. When the Obj are similar, the schedules obtained from OCAGAP give smaller $\#OG$ than the schedules obtained from BAGAP. For example, compared to the result for BAGAP with $b = 60$, the result for OCAGAP with $\epsilon = 0.25$ shows smaller $\#OG$ while Obj is slightly larger.

The x-axis of Figure 5.1 is substituted with average $\#OF$ and average OD in Figure 5.2 and Figure 5.3 each. The results show the same tendency; the schedules obtained from OCAGAP give smaller $\#OF$ or OD than the

Table 5.8: Test results of OCAGAP when $\epsilon \in \{0.05, 0.1, 0.15\}$

Date	$\epsilon = 0.05$					$\epsilon = 0.1$					$\epsilon = 0.15$				
	Obj	*#RG*	*#OG*	*#OF*	*OD*	*Obj*	*#RG*	*#OG*	*#OF*	*OD*	*Obj*	*#RG*	*#OG*	*#OF*	*OD*
1	6,498	23	0	0	0	7,537	17	1	1	29	8,226	14	1	1	29
2	7,023	21	1	1	7	8,161	15	3	3	338	8,841	11	3	5	374
3	6,245	23	0	0	0	7,247	18	0	0	0	7,857	14	3	3	70
4	6,497	20	0	0	0	7,478	15	0	0	0	8,110	11	1	1	7
5	6,297	27	0	0	0	7,233	22	0	0	0	7,964	18	1	1	22
6	5,792	22	0	0	0	6,851	16	0	0	0	7,493	13	0	0	0
7	6,529	21	0	0	0	7,484	16	0	0	0	8,124	12	0	0	0
8	6,161	30	0	0	0	7,117	25	0	0	0	7,832	20	1	1	17
9	6,326	27	1	1	5	7,251	21	0	0	0	7,830	17	1	2	103
10	6,217	22	0	0	0	7,210	16	0	0	0	7,830	12	0	0	0
11	6,109	19	0	0	0	7,019	14	0	0	0	7,512	11	1	1	7
12	6,269	25	0	0	0	7,130	20	0	0	0	7,766	15	0	0	0
13	5,783	20	0	0	0	6,674	15	0	0	0	7,229	11	0	0	0
14	6,548	26	0	0	0	7,682	20	0	0	0	8,323	16	1	1	12
15	6,396	26	0	0	0	7,468	22	0	0	0	8,136	18	0	0	0
16	6,529	30	0	0	0	7,680	24	0	0	0	8,385	20	1	1	3
17	5,995	17	0	0	0	7,065	12	0	0	0	7,565	9	0	0	0
18	6,240	19	0	0	0	7,224	14	1	1	22	7,830	10	0	0	0
19	6,500	27	0	0	0	7,475	22	0	0	0	8,124	18	1	1	10
20	6,612	28	0	0	0	7,678	23	0	0	0	8,368	18	1	1	39
Avg	6,328	23.65	0.1	0.1	0.6	7,333	92	0.25	0.25	20	7,967	14.4	0.8	0.95	35

Table 5.9: Test results of OCAGAP when $\epsilon \in \{0.2, 0.25, 0.3\}$

Date	$\epsilon = 0.2$					$\epsilon = 0.25$					$\epsilon = 0.3$				
	Obj	*#RG*	*#OG*	*#OF*	*OD*	*Obj*	*#RG*	*#OG*	*#OF*	*OD*	*Obj*	*#RG*	*#OG*	*#OF*	*OD*
1	8,844	10	2	2	31	9,288	8	3	3	99	9,662	5	3	4	267
2	9,518	8	3	4	277	9,985	5	3	4	308	10,397	3	4	7	552
3	8,421	11	1	1	2	8,873	9	3	4	155	9,330	6	3	3	85
4	8,596	9	1	1	7	8,987	7	1	1	2	9,348	5	2	2	36
5	8,579	15	1	1	22	9,050	12	2	2	44	9,433	9	1	1	7
6	8,094	10	0	0	0	8,573	7	0	0	0	8,890	5	0	0	0
7	8,689	9	0	0	0	9,049	7	0	0	0	9,340	5	0	0	0
8	8,342	17	0	0	0	8,886	15	1	1	3	9,288	12	1	1	5
9	8,402	15	3	3	146	8,833	12	3	4	66	9,177	10	1	2	42
10	8,320	10	0	0	0	8,753	7	0	0	0	9,111	5	0	0	0
11	7,872	9	0	0	0	8,201	7	0	0	0	8,516	5	0	0	0
12	8,341	13	0	0	0	8,780	10	2	2	46	9,189	8	2	2	46
13	7,676	9	1	1	30	8,077	6	0	0	0	8,407	4	2	2	23
14	8,888	14	1	1	8	9,362	11	1	1	12	9,700	9	1	1	4
15	8,713	15	1	1	3	9,248	11	0	0	0	9,678	9	0	0	0
16	9,002	17	1	1	3	9,526	14	2	2	10	9,973	11	3	4	77
17	8,117	6	0	0	0	8,424	4	1	1	8	8,686	2	1	1	43
18	8,410	7	1	1	22	8,723	5	0	0	0	8,996	3	2	2	12
19	8,720	15	2	3	75	9,161	12	2	2	60	9,550	10	3	4	97
20	9,025	15	2	2	44	9,549	12	2	3	36	9,886	10	2	3	36
Avg	8,528	11.7	1	1.1	34	8,966	9.05	1.3	1.5	42	9,328	6.8	1.55	1.95	67

Table 5.10: Test results of BAGAP when $2b \in \{150, 90, 60\}$

Date	$2b = 150$					$2b = 120$					$2b = 90$				
	Obj	*#RG*	*#OG*	*#OF*	*OD*	*Obj*	*#RG*	*#OG*	*#OF*	*OD*	*Obj*	*#RG*	*#OG*	*#OF*	*OD*
1	5,931	25	0	0	0	6,921	20	1	1	40	7,581	17	0	0	0
2	6,503	23	1	1	100	7,588	17	4	4	240	8,525	13	5	6	525
3	5,598	26	0	0	0	6,786	21	3	3	77	7,448	17	2	2	18
4	6,045	23	0	0	0	7,164	17	0	0	0	7,933	12	1	1	2
5	6,065	28	0	0	0	6,907	23	0	0	0	7,693	19	0	0	0
6	5,325	25	0	0	0	6,574	18	0	0	0	7,179	15	0	0	0
7	6,167	23	0	0	0	6,905	19	0	0	0	7,592	15	0	0	0
8	5,617	33	0	0	0	6,726	27	0	0	0	7,548	22	0	0	0
9	5,948	29	1	1	34	6,775	24	2	2	39	7,542	19	2	2	42
10	5,866	24	0	0	0	6,842	18	0	0	0	7,590	14	0	0	0
11	5,332	24	0	0	0	6,312	18	0	0	0	7,045	14	0	0	0
12	5,956	26	0	0	0	6,721	22	0	0	0	7,510	17	0	0	0
13	5,193	23	0	0	0	5,948	18	0	0	0	6,701	14	1	1	2
14	6,074	29	0	0	0	7,059	24	0	0	0	7,739	20	0	0	0
15	6,296	28	0	0	0	7,370	22	0	0	0	8,141	17	0	0	0
16	6,120	32	0	0	0	7,390	26	1	1	3	8,186	21	2	2	36
17	5,690	19	0	0	0	6,551	15	0	0	0	7,221	11	0	0	0
18	5,780	22	0	0	0	6,821	16	0	0	0	7,497	12	0	0	0
19	5,949	29	0	0	0	7,063	24	0	0	0	7,577	21	1	1	63
20	6,170	31	0	0	0	7,314	24	0	0	0	8,132	19	1	1	27
Avg	5,881	26.1	0.1	0.1	7	6,887	20.65	0.55	0.55	20	7,619	16.45	0.75	0.8	36

Table 5.11: Test results of BAGAP when $2b \in \{60, 30, 0\}$

Date	$2b = 60$					$2b = 30$					$2b = 0$				
	Obj	*#RG*	*#OG*	*#OF*	*OD*	*Obj*	*#RG*	*#OG*	*#OF*	*OD*	*Obj*	*#RG*	*#OG*	*#OF*	*OD*
1	9,183	8	1	1	29	10,099	3	4	6	342	10,675	0	4	11	407
2	9,460	8	4	6	507	10,564	2	5	8	589	10,839	0	4	12	915
3	8,861	9	2	3	165	9,922	2	3	3	56	10,072	1	4	7	268
4	8,791	8	1	1	12	9,526	4	2	2	94	1,0107	1	3	8	125
5	9,025	12	1	1	92	10,271	4	4	6	140	10,877	0	4	8	338
6	8,335	8	0	0	0	9,156	3	2	3	67	9,647	0	4	8	180
7	8,873	8	2	3	173	9,773	2	4	5	140	10,071	0	4	11	387
8	8,683	16	1	1	17	10,013	7	3	6	110	11,018	0	3	9	236
9	8,370	15	1	2	100	9,092	10	2	2	88	10,272	3	4	7	523
10	8,490	9	1	1	32	9,449	3	2	3	78	9,914	0	2	3	79
11	7,355	12	0	0	0	8,646	4	2	2	146	9,229	0	3	4	171
12	9,099	7	1	1	9	9,970	2	3	5	112	10,381	0	3	7	285
13	7,536	9	2	2	24	8,491	3	4	6	74	8,958	0	4	8	333
14	9,310	11	1	1	8	10,334	5	3	5	81	11,125	0	5	8	283
15	9,112	12	1	1	36	10,154	5	3	3	154	10,934	0	3	11	376
16	9,649	12	1	1	38	11,003	4	4	8	309	11,568	0	5	14	391
17	8,314	4	0	0	0	8,919	0	2	2	8	8,919	0	2	2	15
18	8,533	6	1	1	22	9,147	2	2	3	34	9,462	0	4	8	164
19	8,879	13	2	2	74	10,192	5	3	6	99	10,813	0	4	12	462
20	9,397	13	2	3	61	10,812	3	4	5	215	11,215	0	3	9	329
Avg	8,763	10	1.25	1.55	70	9,777	3.65	3.1	4.45	147	10,305	0.25	3.6	8.35	313

schedules obtained from BAGAP when Obj are similar. Overall, compared to the schedules obtained from BAGAP, OCAGAP gives schedules with smaller number of overlapping flights, gates with overlaps, and the total duration of overlaps when the number of passengers assigned to the normal gates is similar.

Figure 5.4 shows the average $\#OG$ and $\#RG$ of the schedules obtained from solving OCAGAP and BAGAP. In the schedules obtained from OCAGAP, $\#OG$ is generally smaller even when $\#RG$ is smaller. It means that in OCAGAP, fewer flights are handled in the remote gate and normal gates handle more flights than AGAP while keeping the number of gates with overlaps small.

To see why these results appeared, we finally made an additional observation of the obtained schedules. For the instance of August 7, OCAGAP with $\epsilon = 0.25$ and $\epsilon = 0.3$ obtained schedules with more number of passengers assigned to the contact gates and less number of flights assigned to the remote gate, while the numbers of gates and flight pairs with overlaps and the total duration of overlaps are smaller, compared to the schedule obtained from BAGAP with $2b = 60$. We first observed the nominal idle times between flights in in the schedule obtained from BAGAP with $2b = 60$. The minimum and the maximum idle times are 60 and 255 minutes respectively. The variance of the idle times are 1,932. Next, the idle times of the schedule obtained from OCAGAP with $\epsilon = 0.25$ has the minimum of 15 minutes, which is smaller than that of BAGAP, and the maximum of 310 minutes, which is larger than that of BAGAP. Also, variance is 3,369, which is far larger than that of BAGAP. In the same sense, the idle times of the schedule obtained from OCAGAP with $\epsilon = 0.3$ have the minimum of 30 minutes, the maximum of 385 minutes, and the variance 4,344. We can

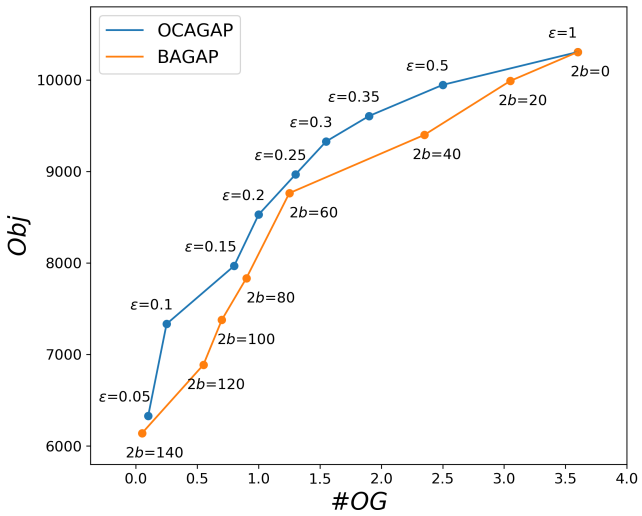

Figure 5.1: Comparison of average Obj and $\#OG$

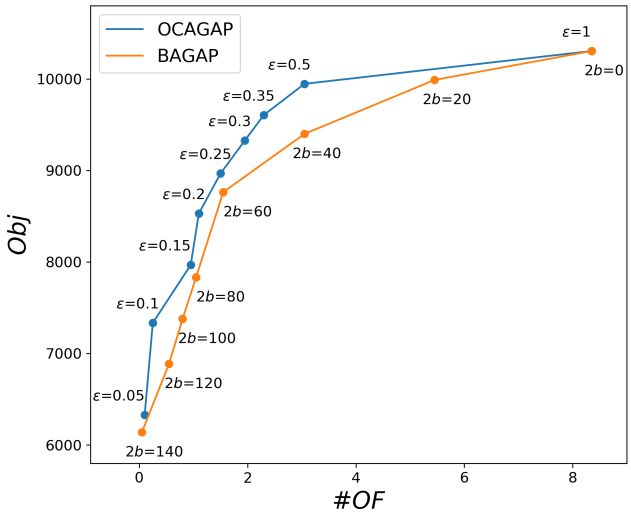

Figure 5.2: Comparison of average Obj and $\#OF$

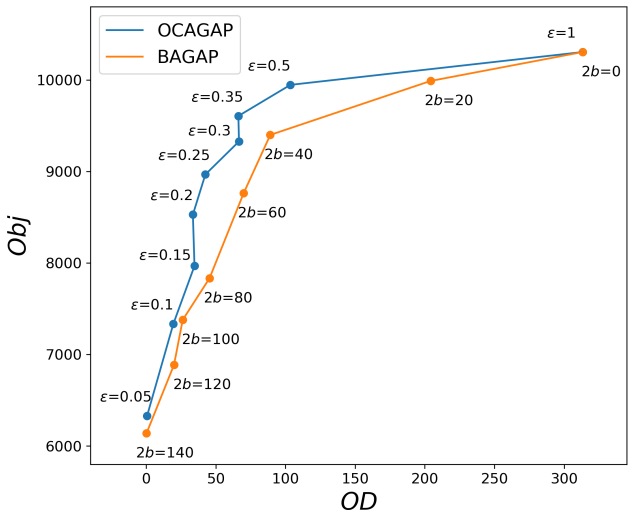

Figure 5.3: Comparison of average Obj and OD

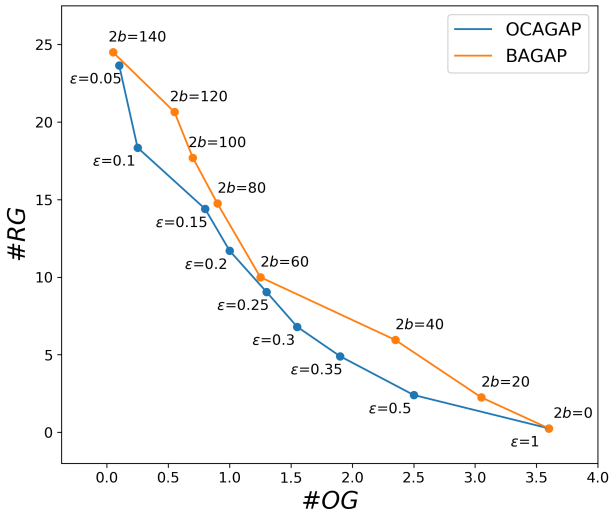

Figure 5.4: Comparison of average $\#RG$ and $\#OG$

see that the idle times obtained in the schedule obtained from OCAGAP are allocated more flexibly in order to meet the limit of the overlap probability and assign flights more effectively. This explains why the results in Figure 5.1, Figure 5.2, Figure 5.3, and Figure 5.4 appeared.

5.3.2 Tests on Artificial Instances with Various Delay Cases

In Section 5.3.1, we used real data of Incheon International Airport in 2019 including the historical delays of flight schedules to test the effectiveness of OCAGAP. Since there are various factors that affect the uncertainty of flight schedules, the data we used cannot reflect some harsh or moderate circumstances. For example, if the weather condition around the airport is violent, the uncertainty of flight schedules may increase. If airlines establish policies that put more emphasis on punctuality of flight schedules, the uncertainty may decrease. The point is that the degree to which flight schedules are uncertain also varies depending on the situation. Therefore, we evaluated the effectiveness of OCAGAP on several different delay cases.

In order to model different delay cases, we use the gamma distribution used in Section 5.2. We fitted the deviations of start and end times of all the flights handled in Incheon International Airport to the gamma distribution, and the resulting distribution is $\Gamma(12.7, 0.18)$. Based on the distribution, we consider 5 distributions $\Gamma(12.7k, 0.18)$ for $k \in \{0.25, 0.5, 1, 1.5, 2\}$. Figure 5.5 illustrates the probability distributions depending on k. The larger k is, the worse the uncertainty gets; the variance of the probability distribution goes higher. When $k \in \{0.25, 0.5, 1\}$, we translated each distribution by 60 in the negative direction and then limited the domain of the function from -60 to 120. When $k = 1.5$, the distribution is translated by 90 in the negative direction and the domain is limited from -60 to 120. When $k = 2$, the distribution is translated by 120 in the negative direction and the domain is limited from -60 to 120.

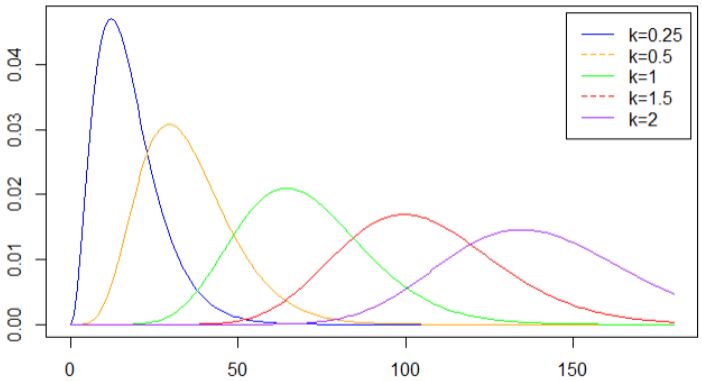

Figure 5.5: Gamma distributions of 5 different delay cases

We generate 10 instances when $|M| = 5$ and $|N| = 50$ in the same way in Section 5.2, except that we generate 5 cases for each instance with the flight delays generated from each of the 5 gamma distributions mentioned above. For each delay case, we assume that we are provided with the parameters of underlying gamma distribution of the flight delays, and construct the overlap probabilities p_{ij} from the convolution of the two independent and identically distributed gamma distributions.

The average Obj, $\#RG$, $\#OG$, $\#OF$ and OD of 10 instances for each k and ϵ obtained from solving the pattern-based model of OCAGAP are given in Table 5.12. When k is same, the larger ϵ gives schedules with larger Obj and smaller $\#RG$, while it gives larger $\#OG$, $\#OF$ and OD. When k becomes larger, the variance of the underlying probability distribution of flight uncertainty is increased. Thus, $\#OG$, $\#OF$ and OD becomes larger for similar Obj or $\#RG$.

Table 5.13 shows the average Obj, $\#RG$, $\#OG$, $\#OF$ and OD of 10 instances for each k and b obtained from solving BAGAP with various buffer times. For same b, Obj and $\#RG$ are same for all k, because the predicted

Table 5.12: Test results of OCAGAP for various delays

k	ϵ	Obj	$\#RG$	$\#OG$	$\#OF$	OD
0.25	0.05	7,080.6	7.1	0.3	0.3	1.6
	0.1	7,235.1	6.0	0.3	0.3	1.1
	0.15	7,273.5	5.3	0.6	0.6	3.1
	0.2	7,279.0	5.2	0.9	0.9	4.4
	0.25	7,335.1	5.0	0.8	0.8	8.1
	0.3	7,346.7	4.8	0.7	0.7	7.1
	0.4	7,426.5	4.3	1.6	1.7	18.2
	0.5	7,445.2	4.2	1.4	1.6	16.3
	0.6	7,546.8	3.8	2.5	2.5	30.9
0.5	0.05	6,807.1	9.6	0.8	0.9	12.8
	0.1	7,004.0	8.0	1.4	1.5	18.5
	0.15	7,093.8	7.0	1.1	1.1	20.1
	0.2	7,221.0	6.2	1.8	1.9	30.4
	0.25	7,261.0	5.7	1.8	1.9	32.3
	0.3	7,273.5	5.3	1.7	2.1	31.2
	0.4	7,336.2	4.8	2.4	2.7	41.1
	0.5	7,419.1	4.4	2.6	3.1	48.0
	0.6	7,521.2	3.9	3.0	3.6	62.0
1	0.05	6,476.3	12.2	0.6	0.6	8.9
	0.1	6,646.1	10.7	1.5	1.5	34.9
	0.15	6,824.3	9.5	2.0	2.0	42.1
	0.2	6,940.9	8.6	1.7	1.7	33.0
	0.25	7,053.8	7.6	2.4	2.8	58.0
	0.3	7,143.7	6.6	2.3	2.7	71.9
	0.4	7,267.8	5.8	2.8	3.6	86.6
	0.5	7,350.5	4.9	3.1	4.1	98.2
	0.6	7,452.1	4.4	3.9	5.0	155.4
1.5	0.05	6,161.9	14.3	1.0	1.0	28.7
	0.1	6,468.4	12.5	0.9	1.0	23.0
	0.15	6,586.6	11.5	1.1	1.1	26.5
	0.2	6,754.9	10.2	2.2	2.4	62.6
	0.25	6,867.3	9.2	2.3	2.4	57.7
	0.3	6,963.3	8.5	2.7	3.0	80.6
	0.4	7,143.3	6.8	3.5	4.5	103.7
	0.5	7,279.3	5.6	3.4	4.7	111.3
	0.6	7,388.6	4.9	4.3	5.2	138.7
2	0.05	5,933.9	16.0	0.7	0.7	17.3
	0.1	6,252.7	13.8	1.0	1.0	26.4
	0.15	6,449.7	12.6	1.2	1.3	30.6
	0.2	6,565.3	11.7	1.4	1.6	54.5
	0.25	6,698.8	10.9	2.5	2.6	70.9
	0.3	6,814.2	10.1	2.0	2.2	61.6
	0.4	7,017.7	8.3	3.4	3.8	94.6
	0.5	7,205.2	6.5	3.6	4.3	107.8
	0.6	7,342.9	5.2	3.5	4.9	147.7

Table 5.13: Test results of BAGAP for various delays

k	$2b$	Obj	$\#RG$	$\#OG$	$\#OF$	OD
0.25	120	5,033.0	21.8	0.0	0.0	0.0
	105	5,279.2	20.3	0.0	0.0	0.0
	90	5,595.3	17.8	0.0	0.0	0.0
	75	5,898.8	15.9	0.0	0.0	0.0
	60	6,392.1	12.8	0.0	0.0	0.0
	45	6,646.2	10.6	0.0	0.0	0.0
	30	7,066.9	7.6	0.2	0.2	0.7
	15	7,279.0	5.2	0.5	0.5	3.3
	0	7,587.3	3.4	3.0	4.4	66.0
0.5	120	5,033.0	21.8	0.0	0.0	0.0
	105	5,279.2	20.3	0.0	0.0	0.0
	90	5,595.3	17.8	0.0	0.0	0.0
	75	5,898.8	15.9	0.0	0.0	0.0
	60	6,392.1	12.8	0.2	0.2	3.1
	45	6,646.2	10.6	0.3	0.3	5.8
	30	7,066.9	7.6	1.1	1.1	16.4
	15	7,279.0	5.2	2.1	2.7	45.2
	0	7,587.3	3.4	4.1	6.1	130.3
1	120	5,033.0	21.8	0.0	0.0	0.0
	105	5,279.2	20.3	0.0	0.0	0.0
	90	5,595.3	17.8	0.0	0.0	0.0
	75	5,898.8	15.9	0.0	0.0	0.0
	60	6,392.1	12.8	0.5	0.5	7.6
	45	6,646.2	10.6	1.5	1.5	22.2
	30	7,066.9	7.6	2.6	2.8	70.0
	15	7,279.0	5.2	2.7	3.7	82.5
	0	7,587.3	3.4	4.3	7.8	202.5
1.5	120	5,033.0	21.8	0.0	0.0	0.0
	105	5,279.2	20.3	0.0	0.0	0.0
	90	5,595.3	17.8	0.2	0.2	8.4
	75	5,898.8	15.9	0.5	0.5	6.3
	60	6,392.1	12.8	1.8	1.8	47.3
	45	6,646.2	10.6	1.4	1.5	35.0
	30	7,066.9	7.6	3.0	4.0	98.0
	15	7,279.0	5.2	3.7	5.7	163.2
	0	7,587.3	3.4	4.0	7.7	244.3
2	120	5,033.0	21.8	0.1	0.1	1.5
	105	5,279.2	20.3	0.3	0.3	14.3
	90	5,595.3	17.8	0.4	0.4	13.3
	75	5,898.8	15.9	0.7	0.8	17.2
	60	6,392.1	12.8	1.4	1.7	50.5
	45	6,646.2	10.6	1.9	2.1	57.7
	30	7,066.9	7.6	3.4	4.3	133.7
	15	7,279.0	5.2	4.2	6.2	183.9
	0	7,587.3	3.4	4.1	8.6	329.1

overlap probabilities are not reflected in the model. With smaller buffer time, AGAP gives schedules with larger Obj and smaller $\#RG$, while $\#OG$, $\#OF$ and OD becomes larger.

To test the effectiveness of OCAGAP in a harsh condition, we give a further comparison of the gate assignment schedule obtained from OCAGAP to the schedule obtained from BAGAP when $k = 2$. Figure 5.6 shows the average $\#OG$ and Obj of the schedules obtained from OCAGAP and BAGAP. For OCAGAP, $\epsilon \in \{0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.6, 0.75, 1\}$, and for AGAP, $2b \in \{75, 70, 65, 60, 55, 50, 45, 40, 35, 30, 15, 0\}$. Like in Figure 5.1, the schedules obtained from OCAGAP give smaller $\#OG$ than the schedules obtained from BAGAP when Obj are similar in general. Likewise, in Figure 5.7 and Figure 5.8, the results show that the schedules obtained from OCAGAP generally give smaller $\#OF$ and OD than the schedules obtained from BAGAP when Obj are similar. The average $\#OG$ and $\#RG$ of the schedules obtained from OCAGAP and BAGAP are shown in Figure 5.9. It shows that $\#OG$ is generally smaller in the schedules obtained from OCAGAP when $\#RG$ is smaller. However, there are several points out of the tendency. Unlike the experiment in Section 5.3.1, in this experiment, we did not use the flight or airline information to predict uncertainty separately, so the difference of the effectiveness between OCAGAP and BAGAP is not more significant than in Section 5.3.1. However, the overall results show that OCAGAP is generally effective even when the uncertainty of flight schedules is larger.

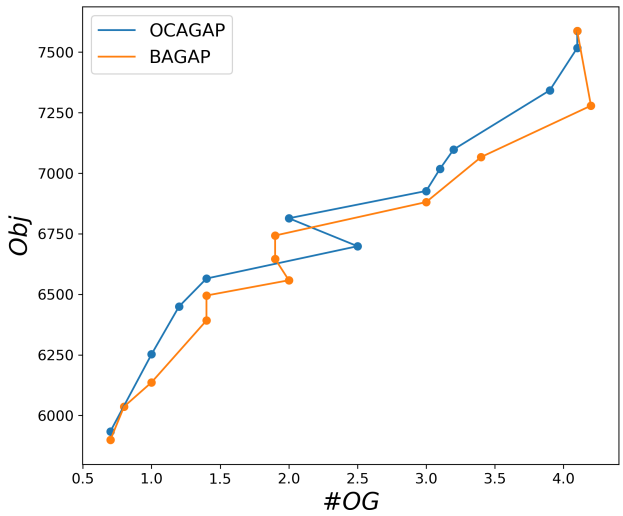

Figure 5.6: Comparison of average Obj and $\#OG$ when k=2

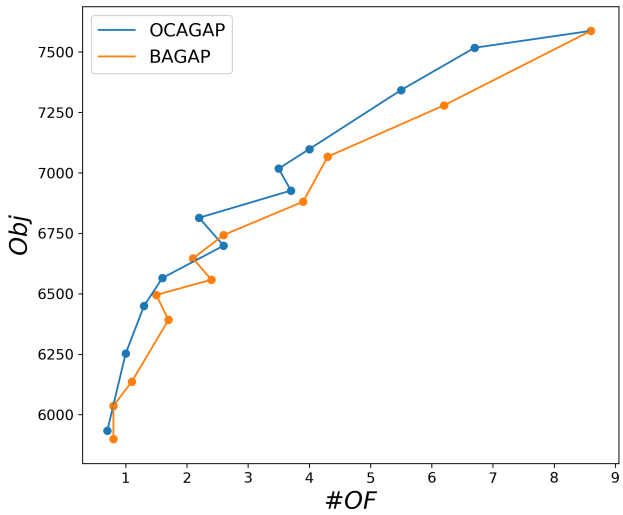

Figure 5.7: Comparison of average Obj and $\#OF$ when k=2

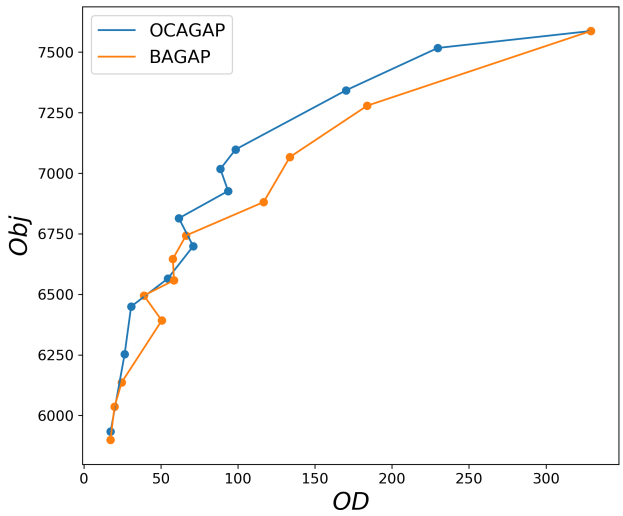

Figure 5.8: Comparison of average Obj and OD when k=2

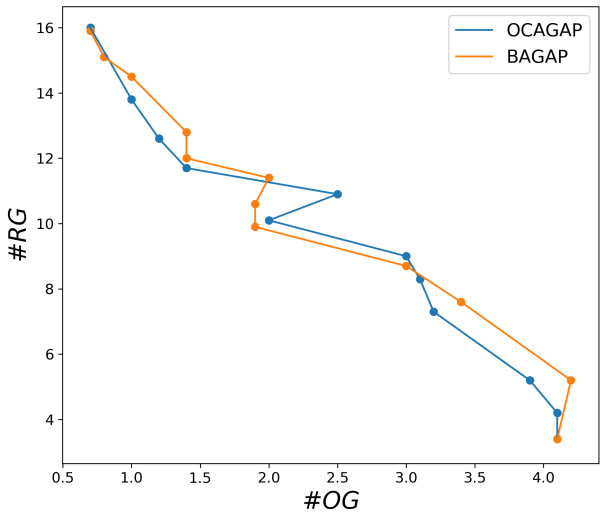

Figure 5.9: Comparison of average $\#RG$ and $\#OG$ when k=2

Chapter 6

Conclusion

In the thesis, we studied the airport gate assignment problem considering overlaps between flights caused by the uncertainty in arrival and departure times of flights. We propose an integer programming model with chance constraints which restrict the probability of an overlap occurrence in each gate, and strengthen the model using flight assignment patterns. To apply these models in practice, a non-parametric prediction method based on empirical survival function is proposed to predict overlap probabilities between flights using historical data. Also, we devised a branch-and-price algorithm to solve the strengthened model. Computational experiments are conducted on real instances of Incheon International Airport in 2019 and artificial instances based on the real data. The computational performances of the branch-and-price algorithm and the commercial solver were dependent on the choice of the objective. Moreover, the results show that the schedules obtained from these models give gate assignment schedules with smaller number of gates with overlaps, overlapping flights, and total overlap duration when the numbers of passengers assigned to contact gates are similar compared to the schedules obtained from the deterministic AGAP model considering buffer time between flights.

For further studies, the prediction method for the overlap probability should be improved to support these models. Also, in the branch-and-price algorithm, we solved the sub-problems to generate profitable flight assignment pattern with a commercial MIP solver. However, these sub-problems are special cases of resource constrained shortest path problem [25] and they may be solved more efficiently with other algorithms based on a dynamic programming. The computational performance of the branch-and-price algorithm should also be analyzed more delicately considering the factors such as number of gates, number of flights, choice of objective, etc. Furthermore, an AGAP model that includes both overlap probability chance constraints and the buffer times should be analyzed with regard to the effectiveness of the gate assignment schedules obtained from the model.

Bibliography

[1] G. S. Daş, F. Gzara, T. Stützle, A review on airport gate assignment problems: Single versus multi objective approaches, Omega 92 (2020) 102–146.

[2] U. Benlic, E. K. Burke, J. R. Woodward, Breakout local search for the multi-objective gate allocation problem, Computers & Operations Research 78 (2017) 80–93.

[3] U. Dorndorf, F. Jaehn, E. Pesch, Flight gate assignment and recovery strategies with stochastic arrival and departure times, OR spectrum 39 (1) (2017) 65–93.

[4] F. Jaehn, Solving the flight gate assignment problem using dynamic programming, Zeitschrift für Betriebswirtschaft 80 (10) (2010) 1027–1039.

[5] A. Lim, F. Wang, Robust airport gate assignment, in: Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence, 2005, pp. 74–81.

[6] C. Yu, D. Zhang, H. Y. Lau, Mip-based heuristics for solving robust gate assignment problems, Computers & Industrial Engineering 93 (2016) 171–191.

[7] Y. Nikulin, A. Drexl, Theoretical aspects of multicriteria flight gate scheduling: deterministic and fuzzy models, Journal of Scheduling 13 (3) (2010) 261–280.

[8] G. S. Daş, New multi objective models for the gate assignment problem, Computers & Industrial Engineering 109 (2017) 347–356.

[9] O. Aoun, A. El Afia, Using markov decision processes to solve stochastic gate assignment problem, in: 2014 International Conference on Logistics Operations Management, IEEE, 2014, pp. 42–47.

[10] J. Guépet, R. Acuna-Agost, O. Briant, J.-P. Gayon, Exact and heuristic approaches to the airport stand allocation problem, European Journal of Operational Research 246 (2) (2015) 597–608.

[11] M. Pternea, A. Haghani, An aircraft-to-gate reassignment framework for dealing with schedule disruptions, Journal of Air Transport Management 78 (2019) 116–132.

[12] S. Yan, C. H. Tang, A heuristic approach for airport gate assignments for stochastic flight delays, European Journal of Operational Research 180 (2) (2007) 547–567.

[13] M. Şeker, N. Noyan, Stochastic optimization models for the airport gate assignment problem, Transportation Research Part E: Logistics and Transportation Review 48 (2) (2012) 438–459.

[14] J. Castaing, I. Mukherjee, A. Cohn, L. Hurwitz, A. Nguyen, J. J. Müller, Reducing airport gate blockage in passenger aviation: Models and analysis, Computers & Operations Research 65 (2016) 189–199.

[15] S. H. Kim, E. Feron, J. P. Clarke, A. Marzuoli, D. Delahaye, Airport gate scheduling for passengers, aircraft, and operations, Journal of Air Transportation 25 (4) (2017) 109–114.

[16] V. Prem Kumar, M. Bierlaire, Multi-objective airport gate assignment problem in planning and operations, Journal of advanced Transportation 48 (7) (2014) 902–926.

[17] L. Xu, C. Zhang, F. Xiao, F. Wang, A robust approach to airport gate assignment with a solution-dependent uncertainty budget, Transportation Research Part B: Methodological 105 (2017) 458–478.

[18] B. Maharjan, T. I. Matis, Multi-commodity flow network model of the flight gate assignment problem, Computers & Industrial Engineering 63 (4) (2012) 1135–1144.

[19] D. G. Kleinbaum, M. Klein, Survival analysis, Springer, 2010.

[20] H. G. Tucker, A generalization of the glivenko-cantelli theorem, The Annals of Mathematical Statistics 30 (3) (1959) 828–830.

[21] M. W. Savelsbergh, A branch-and-price algorithm for the generalized assignment problem, Operations research 45 (6) (1997) 831–841.

[22] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. Savelsbergh, P. H. Vance, Branch-and-price: Column generation for solving huge integer programs, Operations research 46 (3) (1998) 316–329.

[23] FICO® Xpress Optimization, 2020. [Online], `https://www.fico.com/`.

[24] R. Mangoubi, D. F. Mathaisel, Optimizing gate assignments at airport terminals, Transportation Science 19 (2) (1985) 173–188.

[25] J. E. Beasley, N. Christofides, An algorithm for the resource constrained shortest path problem, Networks 19 (4) (1989) 379–394.

국문초록

본 논문은 불확실성하의 공항 게이트 할당 문제를 다룬다. 항공기 출/도착 시각의 불확실성은 수립된 공항 게이트 할당 계획 실행 시에 같은 게이트 내의 항공기 스케줄간의 중첩을 야기하며, 이는 항공기의 지연, 게이트 할당 계획 재수립과 같은 차질을 유발한다. 따라서 본 논문은 실제 항공 데이터를 활용하여 두 항공기 간 중첩이 일어날 확률을 예측하는 방법을 제안하고, 이를 이용하여 각 게이트에서 항공기 스케줄 간 중첩이 발생할 확률을 정량적으로 제한하는 수리모형을 제시한다. 또한 항공기 할당 패턴을 이용하여 강화된 정수최적화 모형을 제시하고 분지평가 알고리즘을 해법으로 제안한다. 인천국제공항의 2019년 실제 데이터를 기반으로 한 실험을 통하여 본 논문에서 제시한 알고리즘의 효율성과 도출된 게이트 할당 계획의 효과성을 확인하였다.

주요어: 공항 게이트 할당 문제, 항공기 지연, 정수최적화, 확률 제약, 분지평가 알고리즘

학번: 2019-28381