# Column Generation Algorithms for Airline Network Revenue Management Problems

Aybike Ulusan

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, 2014

Column Generation Algorithms for Airline Network Revenue Management Problems

Approved by:

| | |
|---|---|
| Prof. İlker Birbil<br>(Thesis Supervisor) | ..... |
| Assoc. Prof. Tonguç Ünlüyurt | ..... |
| Assoc Prof. Özgur Gürbüz | ..... |

Date of Approval:

## Acknowledgements

First of all I would like thank to my thesis supervisor İlker Birbil for his ever ending encouragement and support. Without his wisdom and his guidance I would have never been able to earn a master's degree or write a thesis.

I am thankful to many of our faculty members for their helpful advices and support. In addition, I would like to express my sincere gratitudes to Assist. Prof. İbrahim Muter, faculty member of Bahçeşehir University. Also I am grateful to my dear colleagues Nurşen, Semih, Halil, Belma, Cansu, Ceren, Esra and Rabia, for sharing their experiences and for their endless emotional support. Especially I am thankful to my dearest colleague, Deniz Beşik for her contributions and her vital suggestions during the entire process.

I would like to thank to TÜBİTAK for providing me the financial support.

Finally, I would like to thank to my family for believing in me from the very beginning and standing by me through thick and thin.

Column Generation Algorithms for Airline Network Revenue Management Problems

Aybike Ulusan

Industrial Engineering, Master's Thesis, 2014

Thesis Supervisor: Prof. İlker Birbil

**Keywords:** airline revenue management, column generation, linearization, origin-destination based decomposition.

## Abstract

At the heart of the airline revenue management problem (ARM) lies the seat allocation problem, which has the ultimate aim of finding the right combination of passengers that will result in maximum profit. Due to the dynamic nature of the problem, optimal seat allocations can change continuously over the reservation period. In addition, widely used bid-price booking control policy which necessitates the dual information is obliged to be updated as the demand and capacity values adjust over the reservation period. Thus, in order to make changes in an interactive basis, it is crucial to solve the seat allocation problem in a small amount of time.

This study embodies column generation algorithms applied to ARM problems. Network-based ARM problems are computationally hard to solve even if the airline network is small. However, in this study we challenged ourselves with large-scale airline networks. For computational efficiency, the network is divided into subnetworks by means of date and time information. The overall network is decomposed to origin destination pairs, so that each pair is treated as a single-leg problem. The resulting seat allocation models (static, dynamic and deterministic linear programming) having a non-linear objective

function are linearized by means of the transformation technique proposed by Dantzig which embodies a transformation only by means of additional decision variables. Since column generation can not cope with problems extending row-wise, Dantzigs formulation is the perfect fit. After applying column generation, the numerical results for the models is demonstrated.

Havayolu Gelir Yönetimi Problemleri için Kolon Türetme Algoritmaları

Aybike Ulusan

Endüstri Mühendisliği, Yüksek Lisans Tezi, 2014

Tez Danışmanı: Prof. Dr. İlker Birbil

**Anahtar Kelimeler:**havayolu gelir yönetimi, kolon türetme, doğrusallaştırma,başlangıç ve bitiş çiftlerine ayrıştırma.

## Özet

Havayolu gelir yönetimi problemlerinin özünde, doğru yolcu kombinasyonlarını bularak kârı enbüyükleme amacı olan kapasite dağıtım problemi bulunmaktadır. Problemin dinamik doğası yüzünden en iyi kapasite dağılımları rezervasyon süresi boyunca değişiklik gösterir. Rezervasyon süresi boyunca talep ve kapasite değerlerinin değişmesi yüzünden ikincil bilgiye ihtiyacı olan alım fiyatı rezervasyon konrol stratejisi sürekli olarak güncellenmelidir. Bu sebeplerden ötürü, kapasite dağıtım problemi olabilidiğince hızlı çözülmelidir ki, verilerde olan en ufak bir değiliklik ile kapasite dağıtımları güncellenebilsin.

Bu çalışma, havayolu gelir yönetimi problemlerine uygulanan kolon türetme algoritmalarını bünyesinde bulundurmaktadr. Ağ esaslı gelir yönetimi problemlerinin çözülmesi, havayolu ağı ne kadar küçük olursa olsun hesaplama yükü bakımından oldukça zordur. Yine de, biz bu çalışmada kendimizi büyük ölçekli havayolu ağlarından oluşan problemleri çözmekle sınadık. Daha verimli hesaplama yapabilmek adna, büyük ölçekli havayolu ağını gün ve zaman bilgisinden yararlanarak daha küçük ağlara böldük. Ayrıca, havayolu ağını başlangıç ve bitiş çiftlerine ayrıştırarak, ağ esaslı problemi her bir çift için tek bacak problemi haline getirdik. Bu işlemler sonunda oluşturduğumuz doğrusal olmayan amaç fonksiyonuna sahip olan modeller (statik, dinamik ve deterministik doğrusal

programlama) öncelikle Dantzig'in ileri sürdüğü teknik ile doğrusal hale getirildi. Bu teknik fazladan kısıtlama getirmeden doğrusal olmayan amaç fonksiyonunu sadece ilave değişkenler yardımıyla doğrusallaştırır. Daha sonrasında uygulayacağımız kolon türetme algoritması üretilen her kolonla birlikte eklenmesi gereken kısıtlamalarla başa çıkamadağından Dantzig'in yöntemi bizim için oldukça uygundur. Kolon türetme algoritması uygulandıktan sonra, her model için sayısal sonuçlar verildi.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Revenue management (RM) can be defined as the art of selling each product to the right customer at the right time for the best price with the objective of maximizing the revenue generated from a limited capacity of a product over a finite horizon. RM originated from the airline industry, particularly after the Airline Deregulation Act in 1979. As the deregulation loosened the control of airline prices, airline executives encountered the inevitable decision making process to maximize the total revenue gained. The tactical planning problem of managing limited seat inventories among the various fare classes is referred as seat inventory control (seat allocation) problem in the RM literature.

Airline seat inventory control constitutes allocating seats to different fare classes so that with the right combination of the passengers on the flights, the revenue gained would be maximized. Ever since the potential profitability of RM in airline industry was recognized many mathematical models have been proposed. These models can be divided into two; the models which incorporate leg-based and network-based seat allocation problems. However, prior to elaboration of these problems, the basic concepts that arise in airline revenue management (ARM) problems will be explicated. As widely known, airline companies offer tickets for many origin destination itineraries in various fare classes. Looking from the customers' end, there are only a couple of classes; business, economy and other widely known fare class selections. However, this is not the case from the companies' end. The fare class number can increase up to 20 or more. This increase relates to airline executives struggling with the famous seat allocation problem have to differentiate classes according to the discount levels with various sale conditions and restrictions. An origin-destination itinerary may either be composed of a single flight or multiple interconnected flights and we will refer to these as single-leg and multiple-leg itineraries respectively.

That is to say, the term leg stands for a flight from one city to another.

The central problem tackled in the leg based models is how to optimally allocate the capacity of a single-leg to various classes. The first mathematical approaches to ARM were focused on single-leg problems. Generally single-leg problems can be divided into two classes: static and dynamic models. Dynamic models can cope with the changes in the arrival of the demand, whereas the static models are not responsive to the volatility of the demand. Remaining capacity of the aircraft and the remaining amount of time in the reservation period are the two significant factors that affect the arrival of the demand. In travel industry as well as in airline industry, reservation systems provide various contrivances to control availability. In other words, the optimal allocation of seat inventories are translated into booking control policies. However, in this study we only focus on finding the optimal seat allocations and the process upon finding these allocations, determining a convenient booking control policy, is out of our scope. Bid-price strategy is another significant booking control policy that we refer to several times throughout the thesis. Bid-price is a threshold value which is set for each leg in the network. Simply, it reflects the opportunity cost of reducing the capacity of a specific leg by one seat. Upon finding optimal seat capacities, bid-price policy asserts that a booking request for an origin-destination itinerary is accepted only if its fare exceeds the sum of the bid-prices of the legs the requested itinerary traverses. In single-leg ARM problems, commonly used controlling policies make use of booking limits, protection levels and bid-prices.

Network-based seat inventory control is aimed at optimizing the seat allocation problem on a complete network of flight legs. Many practical seat allocation problems observed in the airline industry are network based, however single-leg based models also play an important role. The fundamental reason is the difficulty of solving network based seat allocation problems. In order to overcome this difficulty, several approximation methods based on decomposition and mathematical programming are proposed. One of the most prominent decomposition methods is the leg-based decomposition which is effective when locally optimizing the airline network. Clearly, this decomposition method undermines the network effects of shared aircraft capacities among multi-leg itineraries between origin-destination (OD) pairs. In this thesis, we adopted the approach proposed by Birbil et al. (2013), which decomposes the network into OD pairs so that each pair can be treated as a single-leg seat allocation problem.

The following sections cover the definition of the capacity allocation problem as well as our contributions embodying the employed solution approaches and the incentives that motivated us to use these approaches.

## 1.1 Problem Definition

This section embodies the definition of the general seat inventory control problem. The objective of this problem is to maximize the revenue generated from the supply of origin-destination itineraries. These itineraries may be either composed of a single flight or multiple interconnected flights. All in all, each itinerary may have multiple fare classes subjected to various ticket prices. All the flight legs in an airline network have a certain capacity and the problem is to allocate the limited flight leg capacity to itineraries in the most profitable way. Consequently, for each origin-destination itinerary there are unique decision variables which indicates the seat allocations. Hence, each seat in each flight leg is reserved only for that origin-destination itinerary. In this study, network-based seat inventory control problem is solved via decomposing the network into origin-destination pairs. That means every multi-leg route is represented as an OD-pair. Thus, after allocating capacities to each OD-pair, the network-based seat allocation problem boils down to solving a single-leg seat allocation problems. Birbil et al. (2013) proposed a generic mathematical model incorporating the OD based decomposition. We also use the same decomposition approach in this thesis and created our models accordingly.

## 1.2 Motivation

The objective in revenue management is to maximize profits. An airline revenue management problem is about finding the optimal booking control policy that will yield the maximum profit. In other words, the fundamental airline revenue management decision is whether to accept a booking request or not.

At the heart of the airline revenue management problem lies the seat inventory control problem. One of the significant characteristics of this problem is that the booking requests have a probabilistic nature. Seasonal changes, holidays, hour of day, day of week are just some factors that create these fluctuations in the demand. Upon solving the seat inventory control problem and finding the optimal seat allocations to various class fares; these allocations are translated into a booking control policy so that a justifiable decision shall be made. However, due to the dynamic nature of the demand a decision made in one second may not be the optimal decision for the next. Therefore, one should continuously monitor the seat inventory control system and should make adjustments to seat allocations when needed (Williamson, 1992). In other words, it is crucial that the solution time of a network seat inventory control problem is small enough to make rapid adjustments.

Consider a widely used booking control policy, the bid-price. It is a very simple method for managing seat inventories, and it is very easy to implement, hence it is widely used in ARM. Basically, bid-price is a shadow price for the capacity constraint. According to this policy, a booking request for an itinerary is accepted only if its fare exceeds the sum of bid-prices of the flight legs traversed by that specific itinerary. Once a booking request for an itinerary is accepted, that itinerary is designated as open to booking and fill up its capacity based on the booking control policy proposed with the initial bid-prices. However, for the sake of making decisions based on the current status of the reservation period, bid-prices should be updated after every accepted booking request. This necessitates to solve the seat allocation problem as quickly as possible.

In addition to that, Durham (1995) emphasizes the importance of making rapid decisions by reporting that at peak times, large computer reservations systems must cope with five thousand transactions per second. Hence, the decision of whether to accept a booking request or not must be reached within milliseconds of the request's arrival. Thus, the seat allocation problem should attain a rapid solution.

While solving large-scale network-based seat inventory control problems, simplex algorithm solving the overall problem data may not reach to optimality within the desired time frame. In this study, our ultimate goal is to solve large-scale network-based seat inventory control problems quickly. Thus, we propose to use column generation approach which may significantly reduce the solution time in large-scale problems. In addition, we divide the airline network into subnetworks in order to make use of the parallel capacity power.

## 1.3 Contributions

The fundamental contribution we made in this study is proving that the solution time of a large-scale network-based seat allocation problem can be diminished by effective management of the columns in the problem. We adopted column generation (CG) approach while managing the columns in the problem. This approach prefers to reach to optimality by solving a small set of the problem data several times, hence computationally it is a more efficient approach than solving the entire network at once.

First of all we introduce the mathematical models that are covered throughout the study. Static and dynamic models are two different seat allocation problems which have discrete concave functions. In order to apply column generation, we transform these two non-linear programming models to linear programming models. There are many differ-

ent transformation strategies; they either transform the function by means of additional constraints or additional variables or both. Since our focus is applying merely column generation, we need to make transformations only by increasing the number of decision variables. Dantzig (1956) proposed a transformation technique which only increases the problem size column-wise. By using Dantzig's formulation (DF), we linearize our static and dynamic models and then apply column generation. In addition, we emphasize the relation between Dantzig's formulation and the well-known deterministic linear programming model. Then we discuss that if the non-linear seat allocation model with deterministic demand is linearized by means of DF, then the resulting model is the famous deterministic linear programming model (DLP).

In this study; static, dynamic seat allocation models and DLP formulation is solved on a large-scale airline network data retrieved from a Turkish airline company. The data is divided into subnetworks by means of the real date and time information. The subnetwork structure is exploited when solving the pricing subproblem (PSP) and generating itineraries for the airline network. We report extensive numerical results on a realistic size airline network.

## 1.4 Outline

The outline of the study is as follows. A general overview of the airline revenue management problems is given in Chapter 3. The mathematical models of the three problems that are covered throughout the study is given in Chapter 3. Chapter 4 incorporates a method for linearizing the resulting non-linear programming models. In addition, column generation approach is explicated throughout the section. The deviation from the conventional column generation approach and its adaptation to the variations of the generic model are also discussed. The airline network data as well as the generation of the booking requests that we used in our computational study are given in Chapter 5. In addition, solving the ARM problems by column management is compared to solving the whole network at once by an off-the-shelf optimization solver. Finally, in Chapter 6 we emphasize the driven conclusions and give some final remarks on the study as well as on the possible future work.

# Chapter 2

# Literature Review

The fundamental goal of this thesis is to solve seat allocation problems in a small time so that changes in the booking control policy can be done on an interactive basis. As mentioned before, the framework proposed by Birbil et al. (2013) is used while constructing the problems, so that the single-leg methods are extended to a network setting. After the multi-leg airline network is decomposed to its OD-pairs, the non-linear cost function of the models are linearised by means of additional decision variables, constraints or both. The resulting models are solved via column generation. An important remark is that, in these models we do not consider overbooking, cancellations, customer choice behaviour or robustness. We begin with a brief account of the seat allocation problems and then review the ARM literature. Afterwards, we discuss solution approaches for the resulting models. Once more, it should be noted that our emphasis is on applying column generation technique when the objective function is non-linear but piecewise linear concave.

At the heart of airline revenue management lies the seat inventory control problem. This problem depends on finding the best trade-off between the revenue gained through greater demand for the discounted seats against the opportunity cost when full-fare reservation requests which are denied due the accepted discounted sales. Therefore, solution methods for the seat allocation problem are concerned with the estimation of these opportunity costs and finding the best control policy maximizing the gained revenue.

The seat inventory problem can be approached from a variety of perspectives; one stream of papers focus on controlling the seat inventories over individual flight legs whereas the other focuses on controlling over the entire airline network. Developments in the field of leg-based models start with Littlewood (1972) where he propose a seat inventory control rule with only two fare classes. He asserts that as long as the revenue value

of the discount fare bookings exceeds the expected revenue of future full fare bookings, the discounted sales should be accepted. Littlewood's two fare class model is generalized by Belobaba (1987). He propose the Expected Marginal Seat Revenue (EMSR) method to solve static single-leg problems when there are more than 2 fare classes. However, EMSR method is merely a heuristic, it can not yield optimal booking limits when all classes are considered. Later on Brumelle and McGill (1993) investigate this heuristic and propose a solution methodology to find the optimal seat allocations. A significant assumption while finding the optimal allocations is made on the arrival of the booking requests. All the mentioned researchers assume that booking requests come in sequentially in the order of increasing fare level. That is to say, low fare passengers arrive before high fare passengers, and this assumption is vital when finding optimal seat allocations. On the other hand, dynamic solution methods do not assume such arrival patterns. Lee and Hersh (1993) are the first researchers who relax this assumption and formulate a dynamic model as a Markov decision process.

Clearly, maximizing individual leg revenues is not the same as maximizing the total network revenue. Williamson (1992) illustrate this statement with a small example and we refer the reader to her study for a detailed discussion. The first mathematical model for the network seat allocation problem with deterministic demand is proposed by Glover et al. (1982). They model the problem using a minimum cost network flow formulation and their primary focus is on the network effects rather than stochastic elements. A drawback in their formulation is the lack of discrimination of the routes within a specific OD-pair. In other words, the model is applicable only when the passengers are path indifferent. The formulation is advantageous because it is easy to solve, large network problems can be optimized in a short amount of time. Wang (1983) proposes an algorithm for sequential allocation of seats under uncertain demand when there are multiple fares and multiple flight legs. Demand uncertainty is also considered by Wollmer (1986) in a large-scale mathematical model. He incorporates the expected marginal seat revenues to the objective function as cost coefficients and for a greater efficiency in the solution, this formulation can be converted into a minimum cost network formulation. Curry (1990) extends Glover's approach into a two stage method and aims at incorporating the nesting of the fare classes within each OD-pair. The optimization problem uses piecewise linear approximations to the revenue function. Similar to Curry (1990), Birbil et al. (2013) propose an OD-based decomposition approach and convert the problem into a single-leg problem for each OD-pair. In this thesis, we adopt their approach when constructing our models with a non-linear objective function.

As stated before, the objective function of the seat allocation problem is piecewise linear thus non-linear. Besides the seat allocation problems, there are many problems in the management and engineering field having a piecewise linear objective function. Thus, there exist many studies which have conducted research on piecewise linear functions. Most notably, we can mention the transformation techniques. These techniques either incorporate additional variables, additional constraints or both. These extra variables and constraints determine the solution efficiency of the problem, as we also illustrate by means of numerical results. A superior transformation technique of a piecewise linear function can reduce the problem size and enhance the computational efficiency in a significant way (Lin et al., 2013). Consider a piecewise function with a single variable $x$ and $m + 1$ breakpoints. A widely used transformation technique necessitates adding extra $m$ binary variables and extra $4m$ constraints. However, when $m$ is large this representation of piecewise functions can increase the computational burden in a significant way. Another transformation framework using fewer binary variables with respect to the aforesaid traditional method is proposed by Li et al. (2009). However, Vielma et al. (2010) studied the framework proposed by Li et al. (2009) and stated that the transformation technique proposed is computationally inferior to the standard transformation method even though the number of binary variables are fewer. Hence, the extra number of constraints required in the linearization process obviously impact the computational performance. Another technique is proposed by Dantzig (1956) which incorporates a transformation only by means of additional variables.

Pioneered by Dantzig and Wolfe (1960), column generation approach allows to solve large-scale linear programming problems. The main idea is to generate columns as needed to improve the objective function value. For the first time, Gilmore and Gomory (1961) put this technique to actual use and apply it to a cutting-stock problem. Nowadays, column generation is a prominent method to cope with a huge number of variables (Lübbecke and Desrosiers, 2005). Conventional column generation can be applied when the number of constraints in the problem is fixed. Recently, Muter et al. (2013), propose a simultaneous row and column generation approach which can cope with problems that grow both column-wise and row-wise. These problems have a special structure, where the constraints depend on the variables. In this thesis, the three models we solve have a piecewise objective function and one of the adopted solution strategies is column generation which requires a fixed number of constraints and a linear objective function. Hence, we employ Dantzig's formulation to avoid constraints depending on variables. Extensive discussion on this topic is made under Chapter 4.

# Chapter 3

# Mathematical Models

In this chapter, first of all we give a generic model of the OD-based seat allocation problem as set forth by Birbil et al. (2013). Then we explicate three models that fit to this generic model. These are static, dynamic and deterministic linear programming models.

In practice, an important issue is where to use these models since all models have some drawbacks. Therefore, when choosing the convenient model for a certain application, one should consider the assumptions of all models, and decide on which set of approximations are more acceptable and what data is available in that application (McGill and Van Ryzin, 1999).

## 3.1 Generic Model

A generic solution approach to network-based seat allocation problem is given by decomposing the network into OD-pairs (Birbil et al., 2013). Let $\mathcal{S} = \{1, \ldots, S\}$ denote the set of OD-pairs on the airline network and $\mathcal{J} = \{1, \ldots, J\}$ be the set of flight legs. The available seat capacity for each flight leg $j \in \mathcal{J}$ is denoted as $C_j \in \mathbb{Z}_+$. Moreover the parameter $a_{js}$ is defined as

$$a_{js} = \begin{cases} 1, & \text{if flight leg } j \text{ is on OD-pair } s; \\ 0, & \text{otherwise.} \end{cases}$$

Thus, one can give the generic model as follows

$$\text{maximize} \quad \sum_{s=1}^{S} \phi_s(x_s), \tag{3.1}$$

$$\text{subject to} \quad \sum_{s=1}^{S} a_{js} x_s \leq C_j, \qquad j \in \mathcal{J}, \tag{3.2}$$

$$x_s \in \mathbb{Z}_+, \qquad s \in \mathcal{S}, \tag{3.3}$$

where $x_s$ denotes the capacity that will be allocated to OD-pair $s$. Solving this problem for every OD-pair $s \in \mathcal{S}$ is the same as solving a single-leg problem with $x_s$ capacities.

Clearly, the main concern is to allocate capacities in such a way that the total revenue is maximized. It is assumed that the function $x_s \mapsto \phi_s(x_s)$ is a discrete concave function. The first set of constraints (3.2) ensures for each flight leg $j \in \mathcal{J}$ the capacity of the aircraft is not exceeded. In other words the number of seats allocated to an OD-pair should not exceed the bottleneck capacity. The bottleneck capacity can be defined as,

$$B_s = \min_{j \in \mathcal{J}} \{C_j \mid a_{js} = 1\}.$$

That is to say for a specific OD-pair, bottleneck capacity equals to the minimum of the flight leg capacities involved in that OD-pair. Constraint (3.3) ensures that the allocated capacities are integer values.

There are multiple fare classes in each OD-pair. For a specific OD-pair $s \in \mathcal{S}$, the maximum number of fare classes is $I_s$ and the set of fare classes is denoted by $\mathcal{I}_s = \{1 \ldots, I_s\}$. Let us assume that the first class is the cheapest fare class; the revenue generated from the first class is the lowest, and class $I_s$ is the highest fare class. Hence for a specific OD-pair $s$, we have

$$0 < r_{1s} < r_{2s} < \ldots < r_{I_s - 1 s} < r_{I_s s}.$$

The no-sales class is simply represented by $0$ with $r_0 = 0$.

The solution of the generic model is the capacity allocation to each OD-pair, $x_s$. However, static and DLP models compute the capacities allocated to various fare classes within an OD-pair as well. In order to observe the partitioned seat allocations, we can denote the relation between the number of reserved fare class $i$ seats in OD-pair $s$ by $x_{is}$. Then we obtain

$$x_s = \sum_{i=1}^{I_s} x_{is}. \tag{3.4}$$

As a result of relation (3.4), the seat capacity constraints (3.2) in the generic model can

be written as

$$\sum_{s=1}^{S} a_{js} \sum_{i=1}^{I_s} x_{is} \leq C_j, j \in \mathcal{J}.$$

Due to the discrete concave nature of the objective function (3.1), the model we are facing is a non-linear programming model. In addition to that, the decision variables have to be integers which complicates the problem even more. In fact Birbil et al. (2013) showed that problem (3.1)-(3.3) is $\mathcal{NP}$-hard (2013, p.11). Hence, we attack the problem by relaxing the integrality constraint. Upon solving the relaxation of the linear programming model, the optimal values of the decision variables may have fractional values. Thus, we round down the values of the decision variables.

Besides providing computational convenience, another significant advantage of using the linear relaxation of (3.1)-(3.3) is the procurement of the shadow prices. That is to say, the resulting dual variables demonstrate the impact of a change in the capacities of the flight legs to the revenue function. Actually, this is the same as using the bid-price control. Notice that, dual variables attain the same value whether the relaxation is solved or the corresponding integer programming model is solved.

## 3.2 Static Model

The static model we employ in our study makes several assumptions that are worth indicating. The first assumption is that requests for different classes are independent random variables. The primary advantage of this assumption is the analytical convenience it provides. The second assumption is that the demand for a given class does not depend on the capacity control. Third, the static model neglects the group bookings. Finally, we do not consider overbooking, cancellations and customer choice behaviour.

As stated in the first assumption the demand for each fare class can be expressed as a random variable. Let random variable $\mathbf{D}_{is}$ be the demand for fare class $i$ in OD-pair $s$. Consequently, for each realization $\mathbf{D}_{is}(\omega)$, the number of occupied seats by fare class $i$ in OD-pair $s$ is denoted by $\min\{x_{is}, \mathbf{D}_{is}(\omega)\}$. Hence, the expected number of occupied seats of fare class $i$ in OD-pair $s$ is $\mathbb{E}[\min\{x_{is}, \mathbf{D}_{is}\}]$. The static model of the proposed decomposition approach then becomes

$$\text{maximize} \quad \sum_{s=1}^{S} f_s(x_s), \tag{3.5}$$

$$\text{subject to} \quad \sum_{s=1}^{S} a_{js} x_s \leq C_j, \qquad j \in \mathcal{J}, \tag{3.6}$$

$$x_s \in \mathbb{Z}_+, \qquad s \in \mathcal{S}, \tag{3.7}$$

where

$$f_s(x_s) = \max \left\{ \sum_{i=1}^{I_s} r_{is} \mathbb{E}[\min\{x_{is}, \mathbf{D}_{is}\}] \mid \sum_{i=1}^{I_s} x_{is} \leq x_s, x_{is} \in \mathbb{Z}_+, i \in \mathcal{I}_s \right\}. \tag{3.8}$$

Notice that for every $s \in \mathcal{S}$ the objective function (3.8) is the same as solving a single-leg seat allocation problem with a flight leg capacity $x_s$. For a given $x_s$, (3.8) can calculate the optimal number of reserved seats for each of the fare classes in OD-pair $s$, which is the partitioned booking limits. Birbil et al. (2009) propose a fast static algorithm to solve $f_s(x_s)$, and in our study we use the same algorithm to solve the cost function and obtain the expected marginal revenue values computed from a unit increase in the allocated capacities. In addition, the solution specifies the optimal partitioned allocations for every possible capacity value in OD-pair $s \in \mathcal{S}$; for every intermediate value in $\{1, \ldots, B_s\}$ partitioned allocations are computed. In Chapter 4, we will scrutinize on how the relaxation of the static model is solved.

## 3.3 Dynamic Model

Before elaborating on the mathematical formulation of the dynamic model, let us state the assumptions that we adopted. First, it is assumed that at each time period at most one request for a fare class of an OD-pair arrives. Second, we do not consider overbooking, cancellations and customer choice behaviour as in the static model. Also, throughout the thesis, we assume that in the dynamic model capacity allocations to OD-pairs are set at the beginning of the reservation period. In other words, as the capacity of an OD-pair is being depleted throughout the reservation period we do not adjust the capacity allocations with respect to the remaining capacities. Adversely, a complete dynamic model designates capacity allocations along the whole period, as stated in the common literature (McGill and Van Ryzin, 1999). However the complete dynamic model is intractable even in small-scale problems because keeping track of the remaining capacities leads to an explosion of the state space (McGill and Van Ryzin, 1999).

Let $g_s^t(x_s)$ be the expected optimal revenue for OD-pair $s$ with available capacity $x_s$

from period $t$ to the departure of the last period. The dynamic model of the proposed decomposition approach can be given as

$$\text{maximize} \quad \sum_{s=1}^{S} g_s^1(x_s), \tag{3.9}$$

$$\text{subject to} \quad \sum_{s=1}^{S} a_{js} x_s \leq C_j, \qquad j \in \mathcal{J}, \tag{3.10}$$

$$x_s \in \mathbb{Z}_+, \qquad s \in \mathcal{S}. \tag{3.11}$$

Let $T$ be the length of the reservation period and $\hat{T}$ be the number of discretization periods (epochs). In addition, $\xi_t$ denotes the random revenue generated at time $t$ and its discrete density is given by

$$\mathbb{P}(\xi_t = r_i) = p_{it}, \qquad i = \{0, 1, \ldots, I_s\}, t = \{1, 2, \ldots, T\}.$$

Thus, dynamic programming recursion in order to calculate the cost function (3.9) can be given as

$$g_s^t(x_s) = \mathbb{E}[\max\{\xi_t + g_s^{t+1}(x_s - 1), g_s^{t+1}(x_s)\}] \tag{3.12}$$

which has the following boundary condition

$$g_s^{\hat{T}}(x_s) = \begin{cases} \mathbb{E}[\xi_{\hat{T}}], & \text{if} \quad x_s > 0; \\ 0, & \text{if} \quad x_s = 0. \end{cases}$$

For the above optimal value function, Lautenbacher and Stidham Jr (1999) have shown that for any given t, the function $g_s^{t+1}(x_s) - g_s^{t+1}(x_s - 1)$ is nonnegative and nonincreasing in $x_s$. Note that the dynamic model (3.9)-(3.11), has a discrete concave objective function $x_s \mapsto g_s^1(x_s)$, thus it is a special case of the generic model (3.1)-(3.3).

## 3.4 Deterministic Linear Programming Model

Suppose that, the expected demand $d_{is} \in \mathbb{Z}_+$ for fare class $i$ in OD-pair $s$ is given. Then we obtain the following mathematical programming model

$$\text{maximize} \quad \sum_{s=1}^{S} h_s(x_s), \tag{3.13}$$

$$\text{subject to} \quad \sum_{s=1}^{S} a_{js} x_s \leq C_j, \qquad j \in \mathcal{J}, \tag{3.14}$$

$$x_s \in \mathbb{Z}_+, \qquad s \in \mathcal{S}, \tag{3.15}$$

where $\min\{x_{is}, d_{is}\}$ is the number of occupied seats and

$$h_s(x_s) = \max\left\{ \sum_{i=1}^{I_s} r_{is}[\min\{x_{is}, d_{is}\}] \mid \sum_{i=1}^{I_s} x_{is} \leq x_s, x_{is} \in \mathbb{Z}_+, i \in \mathcal{I}_s \right\}. \tag{3.16}$$

Note that, again the objective function $x_s \mapsto h_s(x_s)$ is discrete concave, so (3.13)-(3.15) fits to the generic model. Wollmer (1986), McGill and Van Ryzin (1999), de Boer et al. (1999) discuss that the previous model can be rewritten as

$$\text{maximize} \quad \sum_{s=1}^{S} \sum_{i=1}^{I_s} r_{is} \min\{x_{is}, d_{is}\}, \tag{3.17}$$

$$\text{subject to} \quad \sum_{s=1}^{S} a_{js} \sum_{i=1}^{I_s} x_{is} \leq C_j, \qquad j \in \mathcal{J}, \tag{3.18}$$

$$x_{is} \in \mathbb{Z}_+, \qquad s \in \mathcal{S}, i \in I_s. \tag{3.19}$$

An equivalent integer programming formulation is then becomes,

$$\text{maximize} \quad \sum_{s=1}^{S} \sum_{i=1}^{I_s} r_{is} x_{is}, \tag{3.20}$$

$$\text{subject to} \quad \sum_{s=1}^{S} a_{js} \sum_{i=1}^{I_s} x_{is} \leq C_j, \qquad j \in \mathcal{J}, \tag{3.21}$$

$$x_{is} \leq d_{is}, \qquad s \in \mathcal{S}, i \in I_s \tag{3.22}$$

$$x_{is} \in \mathbb{Z}_+, \qquad s \in \mathcal{S}, i \in I_s. \tag{3.23}$$

At this point, we can assert the equivalence of problems (3.13)-(3.15) and (3.20)-(3.23). This assertion may be demonstrated in various ways. For instance, a proof may

be find in Birbil et al. (2013). We will discuss and show the equivalence of (3.13)-(3.15) and (3.20)-(3.23) from a different point of view in Chapter 4.

Relaxing the integrality constraints (3.23), problem (3.20)-(3.23) becomes the well-known deterministic linear programming (DLP) in the literature (Williamson, 1992), (Wollmer, 1986). It is common practice to solve and use DLP for benchmarking rather than the integer programming problem (3.17)-(3.19). The reason is that, when the number of decision variables and constraints gets larger, (3.17)-(3.19) becomes really hard to solve. On the other hand, DLP is computationally very efficient to solve. A significant disadvantage of DLP is that, it neglects the uncertainty of the demand forecast and approximates demand by its mean.

# Chapter 4

# Solution Approach

The fundamental goal of this study is to solve problems on large-scale networks. In this chapter, we discuss how column generation can be used to solve such problems. Before we elaborate on column generation, let us remark that column generation can not cope with non-linear problems. Therefore, our first step is converting any non-linear problem into a linear programming problem. We begin with a brief explanation of the linearization scheme proposed by Dantzig, and then continue with the column generation approach. Finally, we will elaborate on how to apply of Dantzig's linearization scheme (Dantzig's formulation) and column generation to the relaxation of the static (3.5)-(3.7), dynamic (3.9)-(3.11) and deterministic linear programming (3.20)-(3.23) problems.

## 4.1 Dantzig's Formulation

Recall the static model, (3.5)-(3.7) with the objective function (3.8) and the dynamic model (3.9)-(3.11) with the objective function (3.12). As stated before both of these objective functions, $x_s \mapsto f_s(x_s)$ and $x_s \mapsto g_s^1(x_s)$ are discrete concave for every OD-pair $s \in \mathcal{S}$. We define discrete concave function as polyhedral concave so that each piece in these functions are linear in terms of $x_s$. Any piecewise-linear program can be converted to an equivalent linear program (Charnes and Lemke, 1954), (Dantzig, 1956), (Dantzig et al., 1958), (Ho, 1985). All of these transformations either increase the number of variables and constraints by defining at least one variable for each linear piece in each objective term, or one simple constraint for each linear piece, or both. After obtaining a linear programming model, they are solved by the simplex algorithm. Another algorithm to solve a piecewise-linear program without converting it into a linear program is proposed

by Fourer (1985). By extending the simplex algorithm Fourer (1985) proposes piecewise-linear simplex algorithm.

However, merely applying simplex algorithm may fall short when solving large-scale networks. As mentioned, in this study the airline network we are solving is huge. Therefore, in our study we adopted the column generation approach to attack these large networks. In the following section, this approach is discoursed in detail, but in order to fully comprehend the reason behind selecting DF, we first briefly explain CG approach in the light of the linearizing schemes.

The main motivation of column generation is to avoid unnecessary columns from the basis by initiating the solution process with a small set of columns and introducing new ones that will improve the solution. Hence, if a non-basic variable becomes a basic variable the problem will enlarge column-wise and not row-wise. In other words, in order for column generation to be a convenient solution approach, the new basic variable should not introduce new constraints to the problem. If it does, the dual information can not be computed correctly hence the reduced cost which will determine the new basic variables will be unknown. Now suppose that, with each variable entering to the basis, a constraint associated with that variable is introduced into the problem. These problems said to have column dependent rows. In order to cope with additional variables and additional constraints at the same time, simultaneous row and column generation should be applied. We refer the reader to Muter et al. (2013), in which they propose a framework to solve problems with column dependent rows. Thus, merely CG will not be sufficient if the non-linear model is transformed to a linear model with additional constraints. Hence, we focus on converting the non-linear models into linear ones by means of additional variables but no new constraints. Dantzig (1956) proposes a linearizing scheme by means of additional variables. In this thesis, we also apply his method. We refer to this method as Dantzig's formulation.

Before discoursing on how to apply DF to the problems tackled in this study, let us first discuss how problems with a non-linear objective function are converted into linear models by using DF. Consider a situation in which the objective function has the form

$$\sum_{s=1}^{n} \phi_s(x_s),$$

where $x_s \geq 0$ and $\phi_s(x_s)$ is a piecewise linear convex function. Let us assume, the number and the length of the pieces are fixed and there are $k_s$ pieces for each $\phi_s(x_s)$, and the length of each piece is denoted by $u_{is}$, $i \subseteq 1, \ldots, k_s$ . Figure 4.1 illustrates a piecewise convex

function; slopes of the pieces are denoted by $\alpha_{is}$ and widths of the pieces are denoted by $u_{is}$, $i \subseteq 1, \ldots, k_s$.

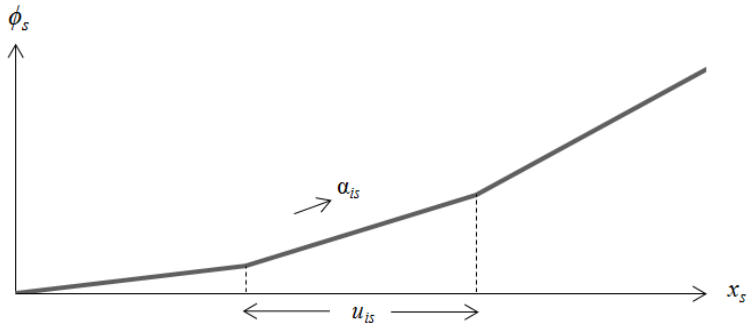


Figure 4.1: Illustration of a piecewise convex function.

Due to convexity, the slope of the pieces follow $\alpha_{is} \leq \alpha_{i+1s}$, $i \subseteq 1, \ldots, k_s$ for each $s$. The main trick in DF is to introduce nonnegative decision variables $\Delta_{is}$ to each piece in the function. In other words, for each function $s$ the number of decision variables $x_s$is replicated by the number of intervals. Hence, the decision variable $x_s$ can be rewritten as

$$x_s = \Delta_{1s} + \Delta_{2s} + \ldots + \Delta_{ks}. \tag{4.1}$$

By employing this approach and substituting the value of $x_s$ into objective function $\phi_s(x_s)$, we get the following linear objective function

$$\phi(x_s) = \sum_{s=1}^{n} \sum_{i=1}^{k} \alpha_{is} \Delta_{is} \tag{4.2}$$

where $0 \leq \Delta_{is} \leq u_{is}$. Simply, $\Delta_{(i+1)s}$ can not attain a value larger than $0$ without $\Delta_{is}$ hitting its upper bound $u_{is}$. In other words, $\Delta_{(i+1)s}$ can not be a basic variable unless the variables corresponding to the pieces with smaller slopes are all non-basic variables. Notice that, among all the decision variables corresponding to each piece of the piecewise function, only one of them can be designated as a basic variable. This manner of treating convex piecewise functions increases the number of variables without increasing the number of constraints. However, it is the number of constraints that, as a rule, determines the work in the simplex method (Dantzig, 1956). Therefore, we may conclude that this transformation technique is also advantageous in terms of computational burden.

The following section explains the column generation approach in general, and then investigates its application to OD-based ARM problems. The application of DF to the

mathematical models is discussed in section Section 4.3.

## 4.2 Column Generation

Column generation technique is pioneered by Dantzig and Wolfe (1960) and Gilmore and Gomory (1961). It is frequently employed when solving large-scale linear programming problems; CG does not intend to solve a large-scale problem at once, it reaches to optimality gradually by solving subproblems at each iteration. LP is initialized with a feasible initial basis, a small set of columns which is referred as restricted master problem (RMP). At each iteration, RMP is enlarged by means of additional columns that will lead the solution to optimality. This iterative solution approach is employed until the problem is proved optimal.

Conventionally the columns that will be added to RMP are found through solving a pricing subproblem (PSP). Suppose we are trying to solve a standard LP problem. Simply, PSP is a subproblem which has the objective of minimizing the reduced costs of columns that exist in RMP. Hence, when PSP is solved to optimality, the solution will specify the column which improves the objective function value at most. Upon solving PSP, if the objective function value is negative, there exists a column with a negative reduced cost. Thus, when this column is added to the RMP, the objective function value of the problem improves.

While implementing CG, DLP is solved to optimality by introducing new columns by means of a PSP. Despite other models, DLP has an advantage of making use of the given revenues for each flight leg. Therefore, with the assumption of setting the total revenue gained from an OD-pair as the summation of the revenues gained by the flight legs traversed by that OD-pair, a shortest path problem as PSP can generate a new OD-pair from scratch. Note that, an OD-pair corresponds to columns in RMP.

However, it is not always possible to find a suitable PSP for the problem. Actually, this is the problem we encounter while applying CG to static and dynamic models. In these models, we lack the information of the expected marginal revenue of the flight legs in a specific OD-pair. In addition, the expected marginal revenue of an OD-pair as a whole is missing as well. In order to calculate these, one has to know the flight legs traversed by that OD-pair; thus the OD-pair should be known. However, without the associated revenue information PSP can not generate an OD-pair. In order to overcome this problem, we created all possible OD-pairs, meaning all possible columns, and calculated their expected marginal revenues. Among the set of all columns, a column can be selected

manually based on the magnitude of the improvement it makes to the objective function and added to the RMP. Algorithm 1 is the pseudocode of the algorithm we proposed to generate all the OD-pairs with a maximum of three flight legs. We restricted the number of stops by three since itineraries with more than three legs is quite rare. The algorithm basically connects the nodes until the number of nodes exceeds the maximum number of nodes; three. All the nodes have unique labels which indicates the previous possible paths to reach each node and the number of arcs traversed with these paths. In the end, the number of labels accumulated in the sink node is the same as the number of all paths with a maximum of 3 nodes. To sum it up, we can easily say that, in this study the adopted column generation approach is divided into two types with respect to the procurement strategy of the column that is added to the RMP.

After pointing out two variations of the column generation approaches, let us retrace our steps and investigate what it is meant by the greatest improvement and discuss ways to calculate this greatest improvement. Simply, for a standard LP, the greatest improvement in the objective function may be obtained through the column which has the most negative reduced cost. Of course, that specific column is just the illustration of the non-basic variable which has the greatest affirmative impact on the objective function value when designated as a basic variable. However, in our case, the obtained linear programming models via DF have decision variables which are bounded above. Then, for a minimization problem, the column with a negative reduced cost will not improve the solution if the decision variable associated with that column is a non-basic variable on its upper bound. The bounded simplex method asserts that, for a minimization problem, a change in a decision variable's value on its lower bound will improve the objective function if the associated reduced cost is negative. In addition, a change in a decision variable's value on its upper bound will improve the objective function if the associated reduced cost is positive. Therefore, while we are checking for columns with a negative reduced cost, we also have to check whether the associated decision variable is a non-basic variable on its lower bound or not. Also it is certain that, a column associated with a non-basic variable on its upper bound is already among the columns in the RMP. Hence, it wouldn't be true if that column is procured as the next column that will be added to the RMP.

Upon observing that the improvement in the objective function is measured by means of the reduced cost, let us go one more step back and discuss how to calculate the reduced cost. Let us remind that, for the given standard LP,

**Algorithm 1:** Finding all the itineraries with a maximum of 3 nodes.

index=1 // Initialization
LabelNumber[index]=1 // First Label
maxnode=3
PreviousLabel[index]=0
NumberofArc[index]= 0
NodeLabel[source]=1 // Label numbers on each node
$\mathcal{S} \leftarrow \emptyset$
**forall the** *node* $v$ *in* $V$ **do**
    NodeLabel[$v$]=undefined
**end**
**while** $\mathcal{S} \neq V$ **do**
    **forall the** *nodes* $v$ *in* $V$ **do**
        $\mathcal{L} \leftarrow$ All Labels of node $v$
        $\mathcal{A} \leftarrow$ All adjacent nodes of $v$
        // for every adjacent node
        **forall the** *nodes* $u$ *in* $\mathcal{A}$ **do**
            **if** $u$== *sink* **then**
                maxnode ++ // Imposing connection with u & sink
            **end**
            // Traverse all labels
            **forall the** *labels* $l$ *in* $\mathcal{L}$ **do**
                **if** *NumberofArc[*$l$*]* $<$ *maxnode* **then**
                    index ++
                    NodeLabel[$u$]= NodeLabel[$u$]$\bigcup$ index
                    LabelNumber[index]=index
                    PreviousLabel[index]= LabelNumber[$l$]
                    NumberofArc[index]= NumberofArc[$l$]
                **end**
            **end**
        **end**
    **end**
    $\mathcal{S} = \mathcal{S} \bigcup \{v\}$
    maxnode $--$
**end**
Backtrack all the labels accumulated in the sink node and find all the itineraries.

$$
\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{n} c_j x_j & & \\
\text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j = b_i, & & i = \{1, \ldots, m\}, \\
& x_j \geq 0, & & j = \{1, \ldots, n\},
\end{aligned}
$$

the reduced cost is equal to

$$\bar{c} = c - y^T A,$$

where $y$ denotes the optimal dual variables associated with each constraint.

Going back to the ARM problem, let us first emphasize that, for convenience we convert the concave objective functions of the tackled ARM problems to convex functions. It is simply done by multiplying the functions with $-1$. From now on, we will presume a minimization objective for the tackled ARM problems. Consider a specific OD-pair $s$ with $I_s$ fare classes. That means that OD-pair has $I_s$ itineraries. These itineraries are composed of same flight legs, since they belong to the same OD-pair, however their revenues change so do their reduced costs. Consequently, the problem of according to which itinerary's revenue the reduced costs should be calculated emerges; because we are seeking for an OD-pair which yields the greatest improvement, but each OD-pair has $I_s$ different reduced costs. We prefer to calculate the reduced costs with the cost parameters associated with the highest fare class. Hence, each OD-pair's objective function improvement is evaluated according to a single reduced cost. Upon finding an OD-pair, along with the highest fare class itinerary, all other itineraries corresponding to the rest of the fare classes of that OD-pair are added to the RMP whether they have a negative reduced cost or not. The other fare classes may not improve the solution; and if that is the case, simplex algorithm takes care of it, and specify them as non-basic variables on their lower bounds. Hence, the solution is not affected if we add all itineraries of an OD-pair to RMP. Notice that, the highest fare class corresponds to the first piece having the lowest slope of the piecewise convex objective function. Therefore, we can apply the same idea to the linearized static and dynamic models and state that the reduced costs are calculated by means of the expected marginal revenues of increasing the OD-pair capacity from 0 to 1, which again indicates the first piece. These points are emphasized within Section 4.3 once more. Also, this method is not the only way to handle multiple itineraries. Other possible ways are demonstrated as a future work in Chapter 6.

As a final remark, we want to underscore that the discussed application of CG embodies a single network. In this study, we are dividing a large-scale airline network to subnetworks, and we diminish our computational burden in a significant way by solving these subnetworks separately but at once instead of solving a large-scale network. We are exploiting the subnetwork structure in two ways; while generating a new column by solving a PSP, and while generating all the OD-pairs of the whole airline network. Generation of all the OD-pairs is in the picture when discussing CG applied to static and dynamic models, whereas solving PSP directly interests DLP. Divided airline network is embarrassingly parallel, therefore one can solve independent small networks instead of solving a huge network.

Until now we discussed to solve an ARM problem with a single airline network. In order to consider all the subnetworks, all of the discussed solution steps should be done for each and every subnetwork. For instance, while trying to find a column to add to the RMP, we investigated the column providing the maximum improvement for each subnetwork. That is to say, at each iteration every subnetwork offers OD-pairs which are eligible to be added to RMP. Consequently, at each iteration PSP is solved for each small subnetwork. At each iteration, we preferred to enlarge the RMP with the itineraries of OD-pair(s) from each subnetwork. However, another one can prefer to choose a single OD-pair among all subnetworks and decide to add to the RMP. As explicated before, when there are several options, we selected the path which will bring us closer to our goal, this does not mean there is a single truth.

## 4.3 Applications to Mathematical Models

In this section, the application of DF and CG to the mathematical models are discussed; the solution steps are demonstrated in detail. We caution the reader once more that from now on, all the models having a concave objective function are treated as convex functions. The conversion is done by taking the negative of the maximization objective function. Initially, DLP is discussed. This is because, DF and CG can be applied more conveniently to DLP. As mentioned DLP does not suffer from the lack of a PSP, its cost parameters are predefined. Hence, the column generation approach can be applied in the conventional way. Afterwards, applications of DF and CG to static and dynamic models are discussed.

### 4.3.1 Deterministic Linear Programming Model

Now, consider the case where the expected demand $d_{is}$ is given, so that we can drop off the expectation from the objective function and obtain the problem (3.13)-(3.15) having a piecewise linear objective function Recall that, an equivalent integer programming problem with a linear objective function is given in (3.20)-(3.22). As the term "equivalent" indicates, the optimal objective values of the problems (3.13)-(3.15) and (3.20)-(3.22) are the same. This statement was explicated in Section 3.4 followed by mentioning the proof of this statement can be done in various ways. Now, we show that this proposition can be demonstrated by means of DF. We indicate how DF applied on (3.13)-(3.15) can result to the problem (3.20)-(3.22).

The objective function of (3.13)-(3.15) $x_s \mapsto h_s(x_s)$ is discrete convex. Let us demonstrate the objective function (3.16) with a piecewise linear graph. For simplicity, suppose that OD-pair $s$ has 3 fare classes and the revenue gained through fare class $i$ is denoted by $r_{is}$. Due to convexity, the following inequality is trivial;

$$0 < r_{1s} < r_{2s} < r_{3s}.$$

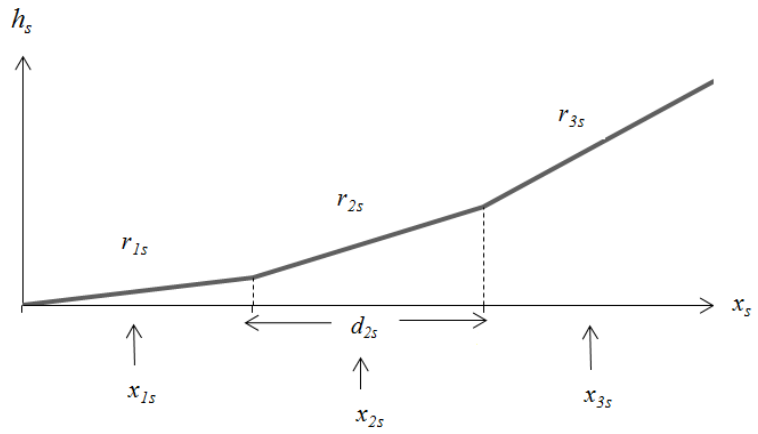


Figure 4.2: Illustration of the piecewise objective function of a DLP.

The above graph is a small illustration of the function (3.16) which takes values with respect to the allocated capacities. One could plainly see that, the number of breakpoints is equal to the number of fare classes; once the maximum number of reserved seats for a fare class is reached, reservation of seats of the other fare class begins. For a fare class $i$, the number of reserved seats can have the maximum value of $d_{is}$. Therefore, as observed in the graph, x-axis values of every breakpoint is equal to the cumulative value of the

associated demands. Note that, gained revenue from a unit capacity increase for a fare class is different with respect to other fare classes. On the other hand, each piece is linear in terms of the allocated capacities.

Recall DF, which transforms a piecewise objective function into a linear objective function by introducing additional variables to every piece in the objective function. Let us apply DF to the small example that is discussed in the preceding paragraph. Since there are 3 pieces, there are 3 additional variables which are introduced to every piece. Let $x_{is}$ denotes the variable associated with the $i^{\text{th}}$ piece. These 3 additional variables add up to the total number of reserved seats for OD-pair $s$ which is denoted by $x_s$ and can be denoted as

$$x_s = x_{1s} + x_{2s} + x_{3s}. \tag{4.3}$$

Also the following inequality is trivial

$$x_{is} \leq d_{is}, \qquad s \in \mathcal{S},\ i \in \{1, 2, 3\}.$$

To sum it all up, once we write $\sum_{i=1}^{I_s} x_{is}$ instead of $x_s$ and introduce the generalized adaptation of (4.3) as a constraint to the problem (3.13)-(3.15), the resulting problem becomes (3.20)-(3.23).

Once we rewrite problem (3.13)-(3.15); either by benefiting from the DF approach or simply by reasoning the qualifications of the problem, the resulting problem becomes a linear programming problem. The next step is to employ CG in order to solve the relaxation of (3.20)-(3.23).

We benefit from the columns of the identity matrix while constructing the initial basis for DLP. Note that, each column of the identity matrix corresponds to different OD-pairs with single flight legs. Each unique column of the identity matrix is replicated by the number of fare classes associated with that specific OD-pair. Consequently, for every OD-pair the number of itineraries is the same as the number of fare classes. After the initial basis is constructed, the problem is solved to optimality and the dual information is obtained. The dual variables associated with (3.21) indicate the shadow prices of a unit capacity increase for each flight leg $j \in \mathcal{J}$. As mentioned before, these dual variables are used to calculate the reduced cost associated with each OD-pair. The next step in our solution strategy is to find a column with the lowest reduced cost, and one can easily calculate the reduced cost by means of the dual variables and the revenues for each flight leg. As mentioned, while solving static and dynamic models, the cost parameters in the objective function are uncertain unless an itinerary is given. However, the revenues for

each flight leg $j \in \mathcal{J}$ are given in DLP, so instead of generating all possible itineraries and investigating their reduced costs, we benefit from a PSP in order to generate a column with the most negative reduced cost. We specify our PSP as the shortest path problem for a directed network $G = (V, A)$. We restrict the number of nodes with 3 since we presumed that the maximum number of legs in an OD-pair con not exceed 3. This presumption is explained in detail in Chapter 5. Algorithm 2 demonstrates the pseudocode that we propose to find the shortest path with a maximum of 3 nodes.

**Algorithm 2:** Finding the shortest path with a maximum of 3 nodes.

```
1  forall the nodes v in V do
2      dist[v]=infinity
3      previous[v]=undefined
4  end
5  dist[source]=0
6  numbernode[source]=0
7  maxnode=3
8  Q ← V
9  while Q ≠ 0 do
10     u = node in Q with minimum dist[]
11     while numbernode[u] > 3 do
12         Q = Q \ {u}
13         u = node in Q with minimum dist[]
14     end
15     forall the neighbours v of u do
16         if dist[u]+ c(u, v) < dist[v] then
17             dist[v]= dist[u]+ c(u, v)
18             previous[v]=u
19         end
20     end
21 end
```

The algorithm we propose aims to find the shortest path from a source node $s$ to the sink node $t \in V$. Notice that, this algorithm is nothing but the Dijkstra's shortest path algorithm with an additional constraint on the number of arcs traversed. Retracing to our problem DLP, the next step in the solution strategy is to designate a non-basic variable as a candidate to be a basic variable. In other words, a column should be generated in order to be added to the RMP and this column should have the minimum reduced cost. The fact that each flight leg corresponding to each node in our network has an associated

dual variable, we can specify each arc cost as the dual variable for a specific flight leg subtracted from that revenue gained by the highest fare class of that specific flight. Once we connect source node to each node, and each node to the sink node by directed arcs, our problem of finding the shortest path becomes finding the route which has the smallest reduced cost. However, for each OD-pair the number of reduced costs is the same as the number of fare classes, since each fare class have different revenues. Remember the discussion in Section 4.2 about calculating the reduced cost for the decision variable which corresponds to the first piece in the objective function. Again, for each OD-pair $s \in \mathcal{S}$ we calculate the reduced cost by benefiting from the first class' revenue. Next step is to add all the itineraries of the selected OD-pair to the RMP. The PSP and RMP are solved iteratively until a column with a negative reduced cost is not found. And if not found, the problem is proved optimal. Note that, we only discussed the case of adding a single OD-pair to RMP from each subnetwork at each iteration. However, the number of OD-pairs may vary depending on the preference of the researcher conducting the study.

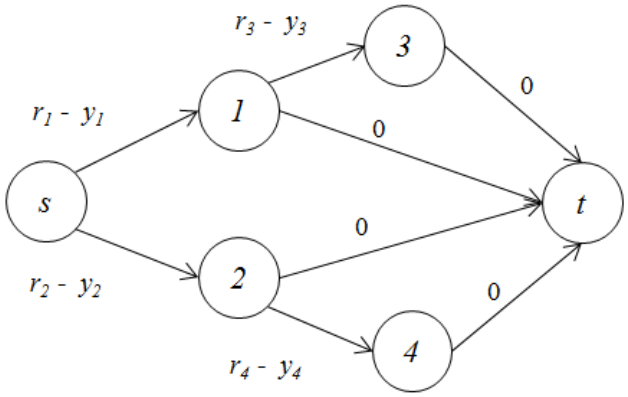


Figure 4.3: Illustration of the airline network adjusted for PSP.

Figure 4.3 is a small illustration of a network where nodes indicate the flight legs, $r_j$ denotes the revenues gained by flight leg $j \in \mathcal{J}$ and $y_j$ is the associated dual variable with flight leg $j \in \mathcal{J}$. In the shortest path problem, the arc costs are additive. That is to say, ultimate path cost is the sum of the arc costs traversed by this path. Looking back to an airline network problem, this may not always be the case. Consider a multi-leg itinerary from İstanbul to Van with a single stop in Ankara. Suppose, the cost of both single flight legs is 100 TL. Airline executives adopt two different approaches while calculating the cost of the itinerary from İstanbul to Van. While certain major airline companies calculate

it by simply adding up the single flight leg's prices, so the total price will be 200 TL in our example, the others make a certain discount on the sum of the single flight leg's prices. In our example, we assume that the price of a multi-leg itinerary is additive, so that shortest path is a convenient pricing subproblem.

In common practice, DLP is one of the mostly preferred models because of the dual information it provides. Williamson (1992) investigates the DLP model for establishing the marginal revenue values for incremental seats on different flight legs, and these construct the basis for the bid-price booking control policy. Recall the primary disadvantage of the bid-price policy, which is the lack of control of the number of bookings when an itinerary is open for booking. In order to overcome this drawback, the bid-prices should be updated with the updated capacity information and demand information. Hence, the dual variables should be calculated in as small time possible to make adjustments in an interactive basis. CG approach pays off in terms of computational time. Numerical results are demonstrated in Chapter 6.

### 4.3.2 Static and Dynamic Models

Recall the static model, (3.5)-(3.7) and dynamic model, (3.9)-(3.11). Both of the models are discussed under the same roof since both of the models have very similar mathematical models; the only difference is the objective functions. Hence, besides the cost parameter calculations, all the solution steps for both of the problems are the same. Primarily, we discuss how DF fits into these models. For each OD-pair $s \in \mathcal{S}$ these problems have a piecewise linear convex function, therefore the derivative is nondecreasing. Slopes and widths of the pieces in these functions are the primary determinant of the functions' structures. The width of each piece is equal to 1, and there are $B_s$ pieces for an OD-pair $s$. Also, the nondecreasing slopes are equal to the expected revenue gained from a unit increase in the capacity. The justification of these models' structure is done in the following manner. The integer decision variable, $x_s$ denotes the allocated number of seats to OD-pair $s$, and clearly it can not exceed the bottleneck capacity $B_s$. Thus, (3.5) and (3.9) will take different values for all intermediate values of $x_s \in \{1, \ldots, B_s\}$. In addition as the value of $x_s$ gets larger, the expected marginal revenue that is gained by a unit increase in seat capacities increases due to convexity. Hence, we conclude that the width of each piece is 1 and the slopes of the pieces are the expected marginal revenues gained by a unit increase in capacity associated with each OD-pair $s \in \mathcal{S}$. Consequently, there are $B_s$ pieces, since $x_s$ increases one by one and the maximum value it can get is $B_s$ .

Now, consider the DF approach which transforms a non-linear function to a linear function without changing the number of constraints but by increasing the number of decision variables. As asserted in DF, the linearization is done by introducing a new decision variable to each piece in a piecewise function, and replacing the original decision variable $x_s$ by the summation of the new decision variables. Hence for each OD-pair $s \in \mathcal{S}$, the number of decision variables is equal to the bottleneck capacity $B_s$, and $x_s$ is out of the picture. The following graph summarizes how DF is applied to a small static or dynamic problem; $\alpha_{ks}$ denotes the slopes of the pieces and $z_{ks}$, $k \in \{1, \ldots, B_s\}$ denotes the new decision variables introduced to each piece.

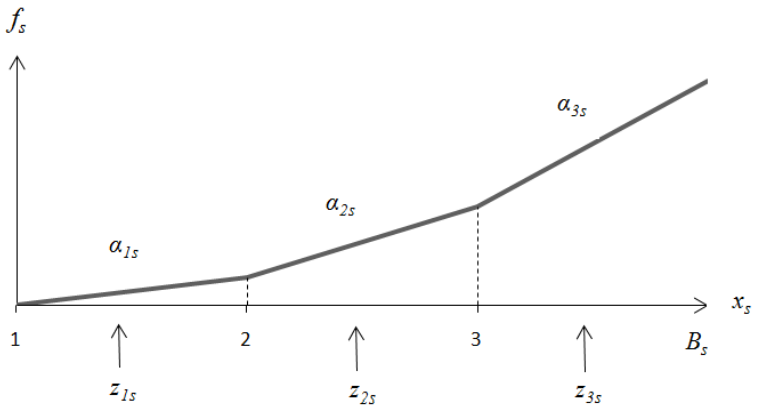


Figure 4.4: Illustration of the DF on piecewise objective function of the static model.

Upon converting the static model (3.5)-(3.7) and the dynamic model (3.9)-(3.11) into a linear programming problem with additional variables we obtain the following model

$$\text{minimize} \quad -\sum_{s=1}^{S}\sum_{k=1}^{B_s} \alpha_{ks} z_{ks}, \tag{4.4}$$

$$\text{subject to} \quad \sum_{s=1}^{S} a_{js} \sum_{k=1}^{B_s} z_{ks} \leq C_j, \qquad j \in J, \tag{4.5}$$

$$z_{ks} \in \{0, 1\}, \qquad s \in S, k \in \{1, \ldots, B_s\}, \tag{4.6}$$

where $\alpha_{ks}$ denotes the expected marginal revenue for each interval $k \in \{1, \ldots, B_s\}$, $s \in \mathcal{S}$, and $z_{ks}$ is the decision variable associated with each piece $k$. Notice that, as the bottleneck capacity increases the problem grows drastically with the additional decision variables. Thus, solving the relaxation of the static and dynamic seat allocation model becomes equivalent to solving (4.4)-(4.6) with relaxing the integrality constraint. The

fractional values of the decision variables are managed by rounding them down to the nearest integer. Note that, even though the proposed model looks exactly the same for the static and dynamic problems, the computation of the cost parameters vary.

The only missing information that prevents us to solve this problem is the value of the expected marginal revenues. While calculating these we use two different algorithms proposed by Birbil et al. (2009). Since the objective functions of the static and dynamic model varies, the computation of the expected marginal revenues differ as well as their value. These two different algorithms are explained in Birbil et al. (2009) in a detailed way.

Once we constructed our model, we solve it by means of column generation. Since the input data is quite large even if a single network is used, the simplex method is not the most efficient way to attack this problem. That is why, we benefited from CG approach. Upon relaxing the integrality constraint (4.6), we initiate CG with a feasible initial basis. As in DLP, we set our RMP from the columns of an identity matrix. Hence, each unique column in the basis corresponds to an OD-pair with a single flight leg. Recall that, each OD-pair $s$ has $B_s$ decision variables, itineraries, with different expected marginal revenues. Therefore, we replicate each unique column of the identity matrix by the value of the bottleneck capacity corresponding to that specific OD-pair and conclude that the replicated identity matrix is the initial RMP. After the construction of the initial RMP, we solve this subproblem to optimality and obtain the dual information. Recall that, the dual variables are used to compute the reduced cost associated with each OD-pair not included in the basis. For the relaxation of (4.4)-(4.6), the reduced cost of the $k^{th}$ itinerary belonging to OD-pair $s$ is calculated as follows

$$\bar{c}_{ks} = \alpha_{ks} - y^T A_{.s},$$

where $\alpha_{ks}$ denotes the expected marginal revenue of the $k^{th}$ piece belonging to OD-pair $s$ and $A_{.s}$ demonstrates the itinerary of the OD-pair $s$ where the flight legs used are indicated by $1$ and otherwise by $0$. So basically, the reduced cost for a specific itinerary of an OD-pair $s$ is equal to dual variables associated with the flight legs used in that OD-pair subtracted from the expected marginal revenue of that itinerary.

As discussed before, the algorithms proposed by Birbil et al. (2009) has the capability of calculating the expected marginal revenue of a specific OD-pair only when the used flight legs in that OD-pair are given. This assertion establishes the basis of our assumption while employing column generation. We assume that we know all the possible OD-pairs

of the problem so that we can calculate their expected marginal revenues. Once all the OD-pairs are generated their reduced costs are calculated. In order to detect the OD-pair yielding the greatest improvement, all OD-pairs should be compared with respect to a single measure. However, for each OD-pair $s$, there are $k$ itineraries, hence $k$ reduced costs, $k$ measures. The same discussion in DLP applies here as well. In order to calculate a single reduced cost value for an OD-pair, we specify the cost component of the reduced cost as the expected marginal revenue gained from the first piece of the function of that OD-pair.

Afterwards, by means of the dual variables and the expected marginal revenues of the first pieces, the reduced cost for every OD-pair is calculated. Suppose there are several OD-pairs having a negative reduced cost value. Theoretically this means among the set of non-basic variables on their lower bounds, several of them yields a negative reduced cost, hence upon entering the basis they will definitely improve the objective function value. After calculating the reduced cost for every OD-pair, we investigate which OD-pair can yield the greatest improvement; has the most negative cost. We conclude the first iteration by enlarging the initial basis with all the itineraries associated with the selected OD-pair. Then, again the problem is solved to optimality via simplex method and with the new dual information new OD-pairs enter the basis. Until an OD-pair with a negative reduced cost is not found, this process is carried on. Note that, we only explained the situation of adding a single OD-pair from each subnetwork to RMP in every iteration. However, there is no restriction on the number of columns that will be added to RMP. It may vary depending on the researchers desire.

Actually, this type of column generation may be referred as a incomplete column generation by some researchers. The reason is, new columns are not generated by means of a PSP. Thus, the computational efficiency gained by the PSP is lost; instead of procuring a column with the most negative reduced cost by solving a simple PSP, the solution approach undergoes the process of generating all the possible OD-pairs, calculating their reduced costs and comparing them. However, despite all these computational burden overall computation time of this column generation approach is better than solving an entire network all at once. The numerical results pertaining the computation times are given in Chapter 5.

# Chapter 5

# Computational Study

The necessity to solve seat allocation problems as fast as possible constitutes our primary incentive to conduct this study. In this chapter, we prove that if the tackled seat allocation problem is a large-scale problem, solving the whole problem data at once is not an efficient approach. On the other hand, solving the problem in partitions by managing the columns is more advantageous in terms of computational time. We first begin by explaining the airline network data, continue with setting up the arrival of booking requests, explicate the benchmarking strategies and finally demonstrate the numerical results.

In this study, we use a real network data structure obtained from a major Turkish airline. This data includes the actual flight legs scheduled for $105$ days. All of these flights are single-leg flights, and their departure and arrival times as well as their capacities are included in the data. However, the number of fare classes for each flight leg and their prices are not included. Therefore, first of all we generate the number of fare classes for each flight leg. Afterwards, we generate a ticket price value for each flight leg by taking the real life ticket prices into account. These prices generate for each flight leg serve as a basis price; this price was scaled for every fare class in that flight leg. In other words, this basis price is set as the revenue gained from the lowest fare class. The revenues for the remaining fare classes are generated through using the following equation,

$$r_{ij} = \text{Base Price} + \text{Base Price} \times \frac{i-1}{I_j}$$

where $I_j$ is the number of fare class in flight leg $j$ and $i$ denotes the fare class.

Another significant absence in the data is the information of OD-pairs. By OD-pairs, we mean the routes from an origin to a destination which can be composed of a single

or multiple flight legs. On the other hand, itinerary term is used for every route having a different revenue. Thus, an OD-pair will have multiple itineraries traversing the same flight legs but having different revenues. Since the information of the OD-pairs is missing, we can not specify the flight legs which can be connected. Thus, we lack the adjacency information of the flight legs. We create the adjacency information of the flight legs by taking the time, date and terminal information of the flight legs. Clearly, an OD-pair can not be composed of multiple legs that have inconsistent arrival and departure terminals. For instance, consider two connecting flights. If the first flight arrives to city A, the second flight must depart from city A. As mentioned before, the network consists of flight legs scheduled for the upcoming $105$ days given along with the departure and arrival times. While generating the adjacency of the flight legs, we make use of the given time and date information and restrict the connection of flights. For convenience, we allow the connection of the flights which have a maximum of $10$ and a minimum of $1$ hour interval between the arrival and departure times. After the generation of the adjacency of the flight legs, we put restrictions on the OD-pairs that are generated by means of the adjacency information. First of all, we introduce a restriction on the duration of the entire journey. We restrict this duration with $2$ days. That is to say, a passenger must arrive at the desired destination within $48$ hours. Without this constraint, there is a possibility that a passenger who is trying to go from city A to city B will begin her journey on day $1$ and will arrive to her destination in $105$ days. This restriction constitutes the fundamental reason of dividing the networks into subnetworks composed of the flight legs of $2$ consecutive days. Secondly, the number of stops is restricted. Without this constraint, an OD-pair may be composed of excessive amount of flights. In order to be realistic, we allow a maximum number of 3 stops. This restriction is used when generating the OD-pairs for every network by means of the proposed algorithm in Section 4.2. Since an itinerary can not be composed more than 3 legs, the shortest path algorithm proposed in Section 4.3.1 is endorsed to find paths with a maximum number of 3 legs.

The overall network is composed of $1,850,328$ OD-pairs and $59,781$ legs. To best of our knowledge, this is by far the largest number of OD-pairs among those reported in the airline revenue management literature. The largest network that we are aware of is studied by Birbil et al. (2013), and this network includes $678$ flight legs and $5,000$ OD-pairs.

However, in this thesis we do not attack the whole network at once. To facilitate the solution process, we divide the network to subnetworks by taking the restriction imposed on the duration of the journey into account. This restriction presumes that an OD-pair can not be composed of flights which have more than $48$ hours in between. Therefore, we

divided the overall network to 105 subnetworks and each of this networks include flights in two consecutive days. For instance, first subnetwork includes the flights in days 1 and 2, the second network includes flights in days 2 and 3 etc.

To give an idea of our network structure, the following three figures report the frequency of the subnetworks in the specified intervals for the number of flight legs, OD-pairs and itineraries respectively.

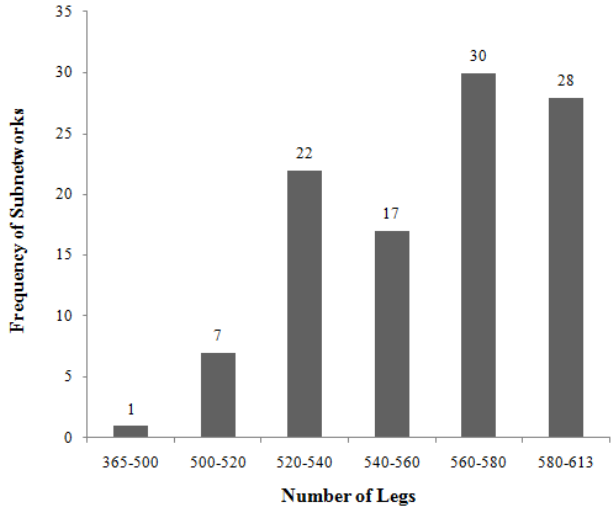


Figure 5.1: Distribution of subnetworks with respect to the number of legs.

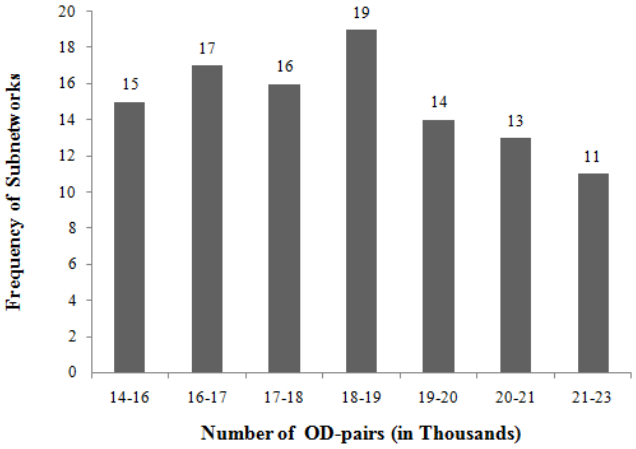


Figure 5.2: Distribution of subnetworks with respect to the number of OD-pairs in thousands.

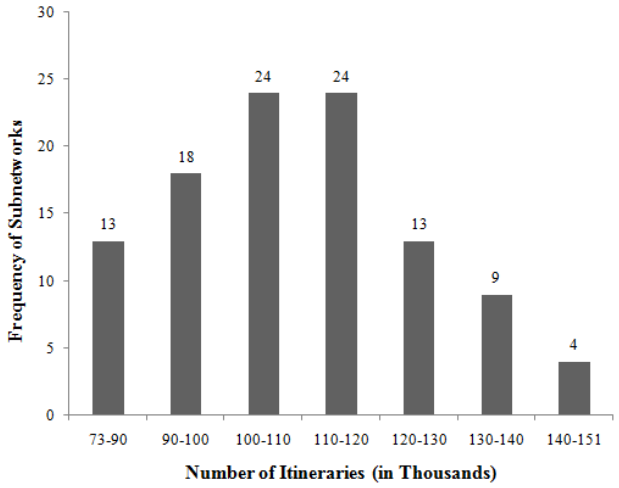


Figure 5.3: Distribution of subnetworks with respect to the number of itineraries in thousands.

| | Descriptive Statistics | | | |
|---|---|---|---|---|
| | **Minimum** | **Maximum** | **Mean** | **Median** |
| Number of OD-pairs | 14,893 | 22,759 | 17,847 | 18,095 |
| Number of Flight Legs | 365 | 613 | 566 | 572 |
| Number of Itineraries | 73,636 | 150,694 | 108,091 | 107,614 |

Table 5.1: Descriptive Statistics of the 105 subnetworks.

Notice that, the features of the subnetworks do not vary that much besides some outliers. As far as it is observed from the histograms, most of the subnetworks have similar characteristics. The ultimate aim is to solve the entire network with 105 subnetworks which corresponds to 3.5 months. However, in order to indicate the impact of the solution approach, we begin by solving a segment of the entire network and then we enlarge the problem by merging the existing network with the flight legs of the consecutive days. Since the consecutive subnetworks consist flight legs of the consecutive days, for convenience, we merge the subnetworks consecutively. Recall that, we have 105 subnetworks and as the number of networks merged gets larger, the overall network size grows drastically.

## 5.1 Simulation Setup

Once we generate our network structure, the only missing information to solve the problems is the arrival of reservation requests since the actual demand distributions are not

specified. So, we simulate the arrival of reservation requests in a planning horizon of length $T$ (Birbil et al., 2013).

We assume that the requests for OD-pair $s \in \mathcal{S}$ arrive according to a homogeneous Poisson process with rate $\lambda_s$. Let $p_{is}(t)$ denotes the probability of a request for OD-pair $s$arrives at time $t$. Clearly, $p_{is}(t) \geq 0$ and $\sum_{i=1}^{I_s} p_{is}(t) = 1$ for all $s \in \mathcal{S}$. Hence, by using multinomial probabilities which are changing over time, $p_{is}(t)$, $i = 1, \ldots, I_s$, $0 \leq t \leq T$, each arriving request is labelled as a certain fare class request. While setting the multinomial probabilities, we consider the fact that in reality lower fare class requests arrive more frequently in the early periods than the higher fare classes. The multinomial probabilities are given as

$$p_{is}(t) = \frac{(v_{is}^T - v_{is}^0)t + Tv_{is}^0}{\sum_{j=1}^{I_s}(v_{js}^T - v_{is}^0)t + Tv_{js}^0}, \qquad i = 1, \ldots, I_s,$$

where $0 \leq v_{I_s s}^0 < v_{I_s - 1 s}^0 < \ldots < v_{1s}^0$, and $0 \leq v_{1s}^T < v_{2s}^T < \ldots < v_{I_s s}^T$. These parameters are predefined. Now we discuss how we simulate the arrival of reservation requests. First of all, during the planning horizon $T$, we generate the arrival time of a booking request for each OD-pair. Then, we set the minimum time among the booking request times as the next event time. Afterwards, by using the multinomial probabilities the fare class of the request is found and booking policy is applied. When the number of reservations for all periods in the specific OD-pair are updated, the simulation will carry on with determining the next event time. The following demonstrates the estimated arrival rate $\mu_j$, for each flight leg $j \in \mathcal{J}$

$$\mu_j = \frac{C_j}{TS_j},$$

where $S_j$ denotes the number of OD-pairs in which flight leg $j$ is used. By using the estimated arrival rates, we calculate the arrival rate for an OD-pair $s$ as follows

$$\lambda_s = \frac{\sum_{j=1}^{J_s} \mu_j}{J_s}$$

where $J_s$ denotes the number of flight legs that are used by OD-pair $s$.

## 5.2 Benchmarking Strategies

In this section, we scrutinize on the benchmarking strategies. As mentioned several times throughout the thesis, there are three models that are solved; DLP (3.20)-(3.23), static (3.5)-(3.7) and dynamic (3.9)-(3.11). Different solution methods to these problems constitute our benchmarking strategies. Illustrative methods of these strategies are explained below.

1. Deterministic Linear Programming (DLPA): Recall the DLP (3.20)-(3.23) which is a widely used model due to its simplicity and the dual information it provides. This strategy embodies the solution of (3.20)-(3.23) via simplex algorithm. All the itineraries of the seat allocation problem, the whole problem data, is solved at once.

2. Deterministic Linear Programming with Column Generation (DLPB): This strategy embodies the column generation approach to DLP (3.20)-(3.23). As a subpricing problem to obtain an OD-pair having the minimum reduced cost, the proposed shortest path algorithm with fixed nodes, Algorithm 2 is used. The cost coefficients for the shortest path problem are set as the revenues gained from the highest fare class of a flight leg. The shortest path problem is solved for every subnetwork and at each iteration OD-pair(s) obtained from every subnetwork is added to the problem with all of its itineraries.

3. Static Model with Additional Variables & Constraints (STA): Recall the static model, (3.5)-(3.7) with the piecewise objective function (3.8). This strategy linearizes the piecewise objective function by adding each piece of the function as a constraint to the problem. Additional constraints are bounded with auxiliary decision variables, and the new objective function in constituted as the summation of all the auxiliary variables. The overall problem is solved via simplex algorithm.

4. Static Model with Additional Variables (STB): This strategy linearizes the piecewise objective function of the static model (3.8) via introducing an additional variables to each piece. Note that, this is Dantzig's formulation explicated in Chapter 4. The overall problem is solved via bounded simplex algorithm.

5. Static Model with Additional Variables and Column Generation (STC): This strategy merges the linearization technique proposed by Dantzig and column generation approach. After linearizing the static problem with DF, CG approach is applied.

At each iteration of the CG, OD-pair(s) that are added to the problem are obtained through scanning the reduced costs of all the OD-pairs. Thus, before initializing CG, all the OD-pairs are generated through the proposed algorithm, Algorithm 1.

6. Dynamic Model with Additional Variables & Constraints (DYA): Recall the dynamic model (3.9)-(3.11). The objective function of this model is computed through the dynamic recursion given it equation 3.12. The same linearization technique and solution strategy with STA is applied to this dynamic model.

7. Dynamic Model with Additional Variables (DYB): This strategy is the same strategy with STB except the objective function of the model tackled. The same linearization technique and solution methodology of STB is applied to dynamic model.

8. Dynamic Model with Additional Variables and Column Generation (DYC): This strategy embodying both DF and CG is the same strategy as STC except it is applied on dynamic model.

## 5.3 Numerical Results

In this thesis, a PC with $3.40$ GHz Core-$i7-4770$ CPU, $16$ GB of RAM is used. CPLEX $12.6$ is used as a solver and MATLAB $R2012b$ is used for coding the algorithms. We test the benchmarking strategies with the problems having different sizes. In order to emphasize the impact of the solution approach, each benchmarking strategy is solved for the seat allocation problem of $k = 5, 10, 15, 30, 45, 60, 75, 90, 105$ days. This is nothing but constructing the problems by merging the first $5, 10, 15, 30, 45, 60, 75, 90, 105$ subnetworks. For instance, in order to create a seat allocation problem of $30$ days, first $30$ subnetworks are merged. Before explicating the numerical results, let us indicate the values of the several parameters used. Firstly, multinomial probabilities, $v_{is}^T$ and $v_{is}^0$ for each $s \in \mathcal{S}$ are uniformly distributed from the interval $(0, 3I_s)$ and then sorted. The reservation period length is set as $T = 100$ and the discretization mesh size is set to $1.0e-1$. The average column numbers and computational times for different benchmarking strategies are reported over $5$ simulation runs.

We begin by illustrating the results for the first two benchmarking strategies; DLPA and DLPB. As mentioned earlier, while conducting column generation the number of columns that are added to the problem in each iteration can vary and can affect the computational time. In order to observe this impact, we solve DLPB in twofolds: first by

setting the number of entering OD-pairs from each subnetwork to $n = 1$ and then $n = 3$. Figure 5.4 specifies the average computational times for the $8$ instances when $n = 1$ and then $n = 3$.

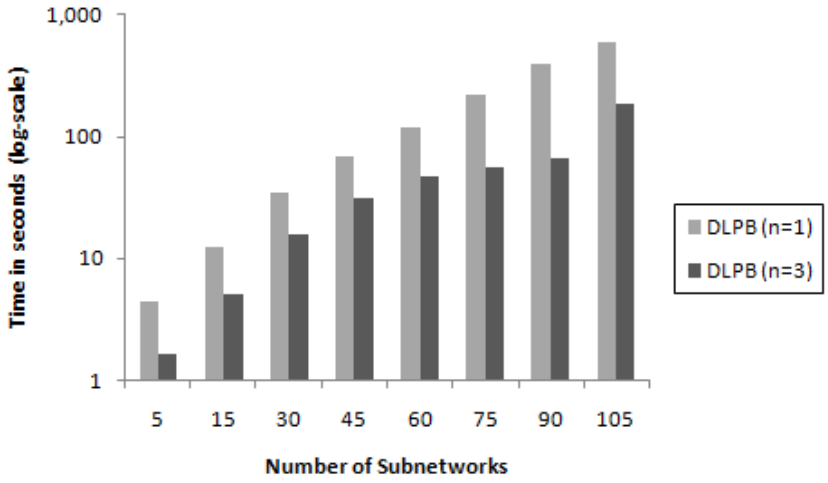


Figure 5.4: The average computational times of DLPB with $n = 1$ and $n = 3$.

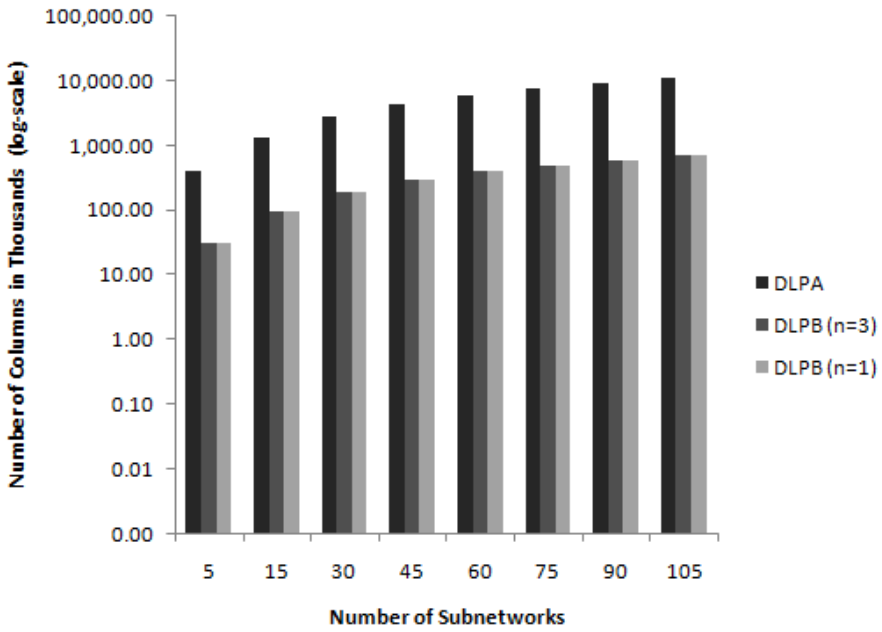


Figure 5.5: The average number of columns in the problems when they are proved optimal.

Figure 5.5 illustrates the total number of columns on a logarithmic plot for the three benchmarking strategies (the number of columns in the vertical axis for the problems with

different sizes are given in logarithmic scale). Notice that, the number of columns indicate the total number of itineraries in the problems when the problems are declared optimal. Even though the column numbers are scaled on a logarithmic basis, the difference between the number of columns of DLPA and DLPB strategies can be observed clearly. It is easy to see that, the problem size DLPA is solving is huge with respect to DLPB. On the other hand, it is really hard to observe a significant difference between the two DLPB strategies; DLPB with $n = 1$ reaches to optimal with slightly less number of columns.

One can observe from Figure 5.4 as the size of the network gets larger, DLPB with $n = 3$ outperforms DLPB with $n = 1$ even though it reaches optimal with more columns. Thus, at each iteration DLPB with $n = 1$ solves a slightly bigger problem. This is due to the trade-off between reaching optimality with a fewer number of iterations and with a smaller set of columns. In our case, diminishing the number of iterations by adding more columns at every iteration is more advantageous. This is because, the overall network solved at every iteration is very large and the number of extra columns is small with respect to the size of the overall network. This also explains why DLPB with $n = 3$ draw away from DLPB with $n = 1$ as the network size gets larger. So, instead of enlarging the network with a single OD-pair from each subnetwork, adding multiple OD-pairs can also increase the computational efficiency. The only question is what will be the limit on the number of columns added in each iteration; if the computational burden of the unnecessary columns surpass solving the entire network, the limit can be levelled down and if not levelled up. For convenience we had set $n$ to $3$ and illustrate the comparison between DLPA and DLPB accordingly. We report in Figure 5.7 the average computational times of DLPA and DLPB. As Figure 5.6 illustrates, as the network size gets larger the average computational times of the two strategies get close to each other. When the problem is solved for approximately $2$ months, that is to say when the number of subnetworks merged is around $60$, the computational times of two strategies coincide, and after this point DLPB begins to outperform DLPA. At this point with $60$ subnetworks, the number of legs equal to $33,317$ and OD-pairs equal to $965,115$. Consequently we can state that, after a certain size, simplex algorithm falls short when solving a problem to optimality. On the other hand, CG is computationally more efficient when solving large-scale networks.

Table 5.2 reports the ratio of the average computational times of the six benchmarking strategies; DLPA, DLPB, STB, STC, DYB and DYC, so that the column generation approach and solving the overall network can be compared numerically.

Figure 5.9 reports the number of columns, decision variables, in the problems after

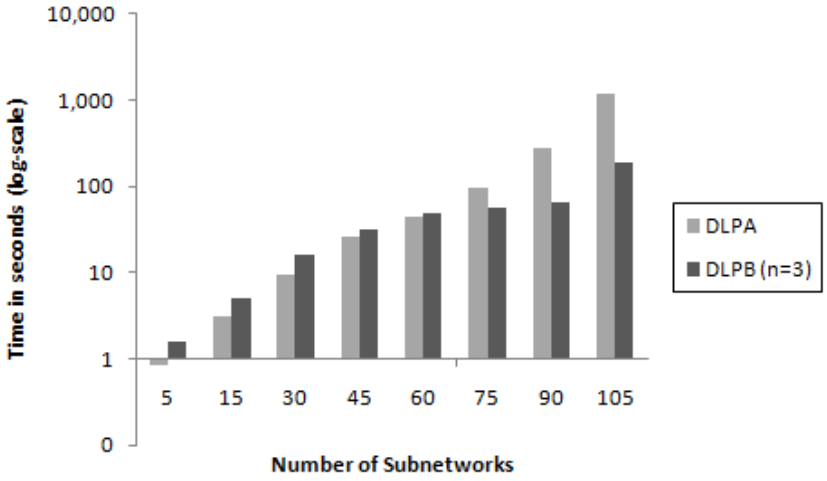


Figure 5.6: The average computational times of DLPA and DLPB.

the linearization schemes proposed by the STA (DYA) and STB (DYB) strategies. On the other hand, Figure 5.10 reports the number of rows, constraints in the problems after the linearization schemes proposed by STA (DYA) and STB (DYB) strategies. As observed, the number of additional variables introduced to STB (DYB) is a lot more than STA (DYB), however the number of constraints introduced to STA (DYA) are more than STB (DYB). Thus, we can state that in our case it is the number of constraints that determines the computational burden rather than the decision variables. Another comparison is made between STB (STC) and DYB (DYC). STC (DYC) embodies column generation technique when solving the static (dynamic) model. The number of OD-pairs that are added to the RMP at each iteration from each subnetwork is set to $n = 3$. Figure 5.7 and 5.8 clearly demonstrates that STC (DYC) outperforms STB (DYB) as the problem size increases. This is due to the advantage CG approach provides when solving large-scale networks. After a certain problem size, the computational burden of the simplex algorithm increases drastically as the problem size gets larger. On the other hand, CG approach shows a rather steady increase because the problem CG solves is a subset of the overall problem. Thus, the increase in the problem size CG is tackling is smaller than the increase in the overall problem size. Recall that, the same discussion is valid for the DLPA and DLPB strategies as well.

| Number of Subnetworks | $\frac{DLPA}{DLPB(n=3)}$ | $\frac{STA}{STB}$ | $\frac{DYA}{DYB}$ |
|---|---|---|---|
| 5 | 0.535 | 0.659 | 0.680 |
| 15 | 0.602 | 0.741 | 0.793 |
| 30 | 0.609 | 0.845 | 0.923 |
| 45 | 0.828 | 0.909 | 0.948 |
| 60 | 0.931 | 1.142 | 1.189 |
| 75 | 1.757 | 1.649 | 1.688 |
| 90 | 3.835 | 2.797 | 2.885 |
| 105 | 6.368 | 4.961 | 5.081 |

Table 5.2: Ratio of the average computational times of the strategies embodying solving the overall network with bounded simplex algorithm and by means of column generation approach.

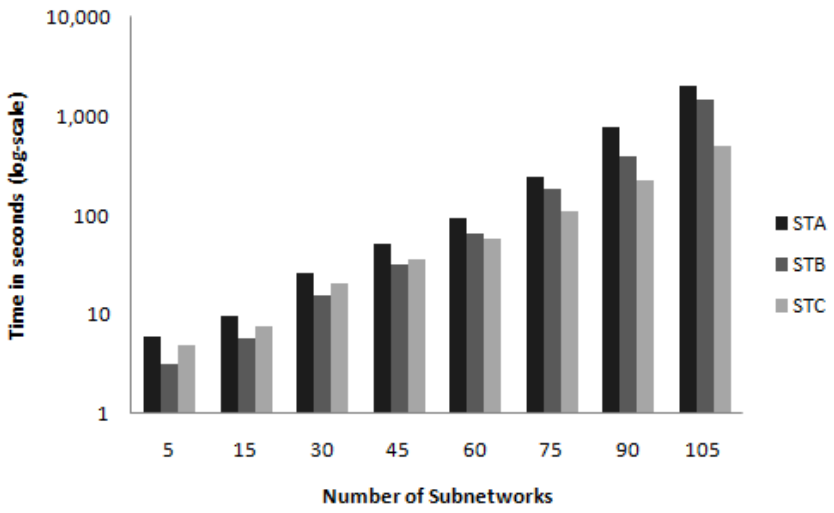


Figure 5.7: The average computational times of STA, STB and STC.

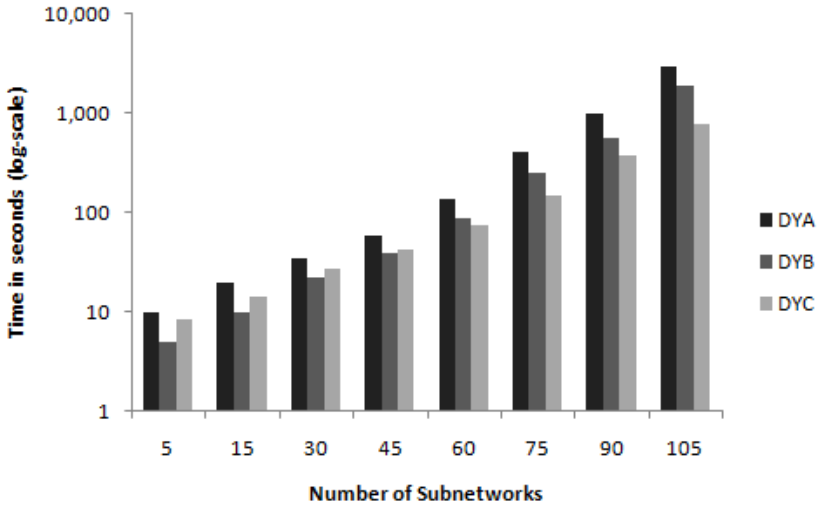


Figure 5.8: The average computational times of DYA, DYB and DYC.

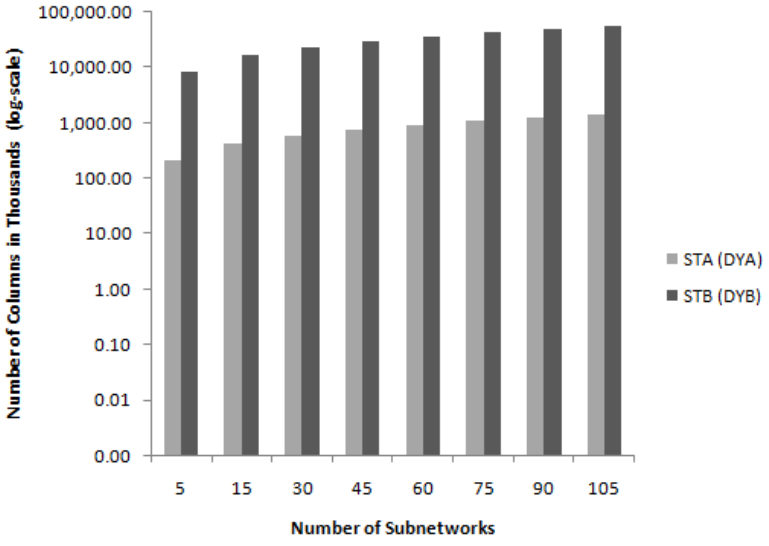


Figure 5.9: The number of columns after the linearization schemes used in STA & DYA and STB & DYB.

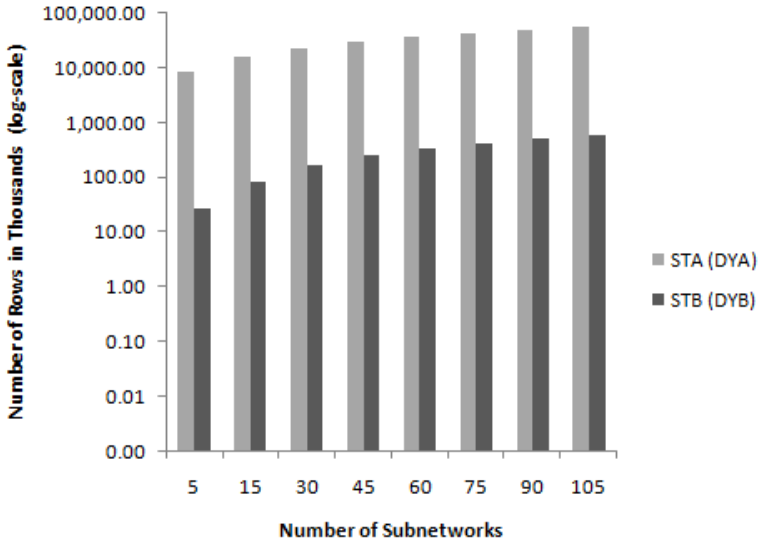


Figure 5.10: The number of rows after the linearization schemes used in STA & DYA and STB & DYB.

# Chapter 6

# Conclusion and Future Research

In this study, we consider network-based seat inventory control problems, and propose a solution approach to solve these problems. We use a large-scale airline network data retrieved from a large Turkish airline company. The magnitude of the airline network data leads us to the idea of attacking a subset of the ARM problems instead of solving the overall problem data at once. Therefore, at the heart of our solution approach lies column generation.

Initially, the data structure is prepared. The large-scale airline network data is divided into subnetworks by means of the given date and time information. The subnetworks are independent from each other and this feature of the data structure is exploited when solving the PSP and generating all the possible OD-pairs. After organizing the data, the network-based problems are decomposed based on their origin and destinations so that solving a seat allocation problem for a fixed OD-pair corresponds to solving a single-leg seat allocation problem. Static, dynamic and DLP formulations are introduced in the light of the OD based decomposition. Since CG approach can not cope with non-linear objective functions, the objective functions of the static and dynamic models are linearized by means of introducing additional variables, which is referred as Dantzig's formulation. We use two variations of the column generation approach; these vary in terms of the procurement strategy of the OD-pair that will be added to RMP. Shortest path problem as PSP is adopted to generate an OD-pair with the smallest reduced cost while solving DLP model with CG approach. Both while generating the OD-pairs and solving the PSP, the subnetwork structure is exploited. Instead of finding a shortest path on the whole network, shortest problem is solved on subnetworks in parallel. By this way, we do not only decrease the computational burden in a significant way, but also find

several short paths. The set of the obtained paths from the subnetworks is not specifically the first $k$ shortest paths, since a single subnetwork can constitute the first $k$ shortest paths, but only one of them is generated. However, due to the similar characteristics of the subnetworks, without loss of generality we assumed that the set does not necessarily consist the best $k$ paths, but it certainly consists paths having a total length close to the shortest path. In short, by solving several shortest path problems, several powerful paths to improve the objective function are found. On the other hand, solving several shortest path problems on a large-scale network can be very incommodious. Static and dynamic models suffer from a vicious cycle due to the used algorithms proposed by Birbil et al. (2009) while computing the objective function values; if an OD-pair is not known the revenue gained from that OD-pair can not be computed, however if the revenues are not known an OD-pair can not be generated. Thus, unlike DLP shortest path problem can not be set as the pricing subproblem. Therefore, with the given adjacency information all OD-pairs are computed, with all the itineraries all the cost coefficients so as the reduced costs are computed. Based on the reduced costs, OD-pairs eligible to enter the basis are detected. New columns that improves the objective function of the seat allocation problems are computed either by means of a PSP or generating all the reduced costs and optimal seat allocations are found when column generation dictates the non-existence of the columns with a negative reduced cost.

The computational study conducted on a set of real life instances illustrates for a large-scale network column generation approach performs better. In addition, linearization techniques increasing the number of constraints adversely affects the computational time more than the linearization techniques increasing the number of decision variables.

Now, let us mention the possible future work. Recall DF, which linearizes the piecewise separable functions without adding any constraints. The fundamental reason behind selecting DF is the inadequacy of CG when column-dependent-rows exist. If the linearization method enlarges the problem both column-wise and row-wise, row generation should be applied in addition to column generation. This is referred as simultaneous row and column generation (RCG) and proposed by Muter et al. (2013). As a future study, a strategy adopting linearization technique with merely additional variables and applying CG can be compared with a strategy adopting a linearization technique both incorporating additional variables and constraints and applying RCG.

As explicitly discussed in Chapter 4, while computing the reduced cost of an OD-pair, the cost coefficient of the first piece in the piecewise objective function is used. Afterwards, when an OD-pair is selected to be added to the RMP based on its reduced cost,

all itineraries of that specific OD-pair is added to the RMP. However, not all of these itineraries are supposed to improve the objective function. Some of the decision variables corresponding to those itineraries are going to be designated as non-basic variables on their lower bounds, and won't have any effect on the objective function. Hence, adding them to the RMP would only increase the problem size. A way to avoid the unnecessary growth of the problem size is to start calculating the reduced costs with the first piece's revenues and then keep adding a single itinerary from the OD-pairs until an OD-pair with a negative reduced cost does not exist. Then, the reduced costs are calculated with the revenues of the second piece. The same process is continued until the OD-pairs with a negative cost is investigated for all the pieces. This method will only add the itineraries that will improve the solution, there won't be any unnecessary columns in the basis. Therefore, the problem will reach optimality with a smaller set of columns with respect to our method of adding all the itineraries. However, the time spent on calculating the reduced costs for every piece may be longer than the gained time on solving a smaller set of columns, which creates a trade-off between the two approaches.

As mentioned, with the information of the flight legs in OD-pairs, an algorithm proposed by Birbil et al. (2009) can compute the expected marginal revenues of OD-pairs. Since the total expected revenue is from an origin to destination, we can not apply shortest path as a PSP. However, if this expected revenue can be fractioned in such a way that it demonstrates the expected revenues of a unit capacity increase for the legs traversed by that OD-pair, the network would have a convenient structure for the application of the shortest path problem. Hence, instead of computing the reduced costs manually and finding columns, shortest path problem can generate new OD-pairs, columns. However, fractioning of the expected revenues is a challenging task and requires a series of assumptions hence loses its credibility. That is why it is excluded in this study, but can be proposed as a future work.

# Bibliography


Belobaba, P. (1987). Air travel demand and airline seat inventory management. Technical report, Cambridge, MA: Flight Transportation Laboratory, Massachusetts Institute of Technology.

Birbil, Ş. İ., Frenk, J., Gromicho, J. A., and Zhang, S. (2009). The role of robust optimization in single-leg airline revenue management. *Management Science*, 55(1):148–163.

Birbil, Ş. İ., Frenk, J., Gromicho, J. A., and Zhang, S. (2013). A network airline revenue management framework based on decomposition by origins and destinations. *Transportation Science*.

Brumelle, S. and McGill, J. I. (1993). Airline seat allocation with multiple nested fare classes. *Operations Research*, 41(1):127–137.

Charnes, A. and Lemke, C. E. (1954). Minimization of non-linear separable convex functionals. *Naval Research Logistics Quarterly*, 1(4):301–312.

Curry, R. E. (1990). Optimal airline seat allocation with fare classes nested by origins and destinations. *Transportation Science*, 24(3):193–204.

Dantzig, G., Johnson, S., and White, W. (1958). A linear programming approach to the chemical equilibrium problem. *Management Science*, 5(1):38–43.

Dantzig, G. B. (1956). Recent advances in linear programming. *Management Science*, 2(2):131–144.

Dantzig, G. B. and Wolfe, P. (1960). Decomposition principle for linear programs. *Operations research*, 8(1):101–111.

de Boer, S. V., Freling, R., and Piersma, N. (1999). Stochastic programming for multiple-leg network revenue management. Technical report, Econometric Institute Research Papers.

Durham, M. J. (1995). The future of sabre. *Handbook of Airline Economics, NY*, pages 485–491.

Fourer, R. (1985). A simplex algorithm for piecewise-linear programming i: Derivation and proof. *Mathematical Programming*, 33(2):204–233.

Gilmore, P. C. and Gomory, R. E. (1961). A linear programming approach to the cutting-stock problem. *Operations research*, 9(6):849–859.

Glover, F., Glover, R., Lorenzo, J., and McMillan, C. (1982). The passenger-mix problem in the scheduled airlines. *Interfaces*, 12(3):73–80.

Ho, J. K. (1985). Relationships among linear formulations of separable convex piecewise linear programs. In *Mathematical Programming Essays in Honor of George B. Dantzig Part I*, pages 126–140. Springer.

Lautenbacher, C. J. and Stidham Jr, S. (1999). The underlying markov decision process in the single-leg airline yield-management problem. *Transportation Science*, 33(2):136–146.

Lee, T. C. and Hersh, M. (1993). A model for dynamic airline seat inventory control with multiple seat bookings. *Transportation Science*, 27(3):252–265.

Li, H.-L., Lu, H.-C., Huang, C.-H., and Hu, N.-Z. (2009). A superior representation method for piecewise linear functions. *INFORMS Journal on Computing*, 21(2):314–321.

Lin, M.-H., Carlsson, J. G., Ge, D., Shi, J., and Tsai, J.-F. (2013). A review of piecewise linearization methods. *Mathematical Problems in Engineering*, 2013.

Littlewood, K. (1972). Forecasting and control of passengers. *AGIFORS Symposium Proceedings*, 12:95–117.

Lübbecke, M. E. and Desrosiers, J. (2005). Selected topics in column generation. *Operations Research*, 53(6):1007–1023.

McGill, J. I. and Van Ryzin, G. J. (1999). Revenue management: Research overview and prospects. *Transportation Science*, 33(2):233–256.

Muter, İ., Birbil, Ş. İ., and Bülbül, K. (2013). Simultaneous column-and-row generation for large-scale linear programs with column-dependent-rows. *Mathematical Programming*, 142(1-2):47–82.

Vielma, J. P., Ahmed, S., and Nemhauser, G. (2010). A note on a superior representation method for piecewise linear functions. *INFORMS Journal on Computing*, 22(3):493–497.

Wang, K. (1983). Optimum seat allocation for multi-leg flights with multiple fare types. In *AGIFORS PROCEEDINGS–*.

Williamson, E. L. (1992). Airline network seat inventory control: Methodologies and revenue impacts. Technical report, [Cambridge, Mass.: Massachusetts Institute of Technology, Dept. of Aeronautics & Astronautics], Flight Transportation Laboratory.

Wollmer, R. (1986). A hub-spoke seat management model. *Unpublished Internal Report, Mc Donnell Douglas Corporation, Long Beach, CA*.