DYNAMIC PROGRAMMING DECOMPOSITION METHODS FOR CAPACITY ALLOCATION AND NETWORK REVENUE MANAGEMENT PROBLEMS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Alexander Erdélyi

February 2010

DYNAMIC PROGRAMMING DECOMPOSITION METHODS FOR CAPACITY ALLOCATION AND NETWORK REVENUE MANAGEMENT PROBLEMS

Alexander Erdélyi, Ph.D.

Cornell University 2010

In this thesis, we develop decomposition-based approximate dynamic programming methods for problems in capacity allocation and network revenue management. Noting that the dynamic programming formulation of these problems suffers from the "curse of dimensionality", we demonstrate that a set of single-dimensional dynamic problems can be employed to provide approximate solutions to the original dynamic program. We show that the proposed approximations have two important characteristics: First, they provide relatively tight performance bounds on the optimal value of the stochastic optimization problem under consideration. Second, they give rise to policies that on average perform significantly better than a variety of benchmark methods found in the literature.

We begin by focusing on network revenue management problems. We assume a profit maximizing airline operating a network of flight legs and processing itinerary requests arriving randomly over time. We consider several variants of the basic model and for each show that the dynamic programming formulation can be decomposed by flight legs into a set of single-leg revenue management problems. Furthermore, we demonstrate that the appropriate decomposition method gives rise to an upper bound on the optimal total expected revenue and that this upper bound is tighter than the upper bound provided by a deterministic linear program known from the literature. Finally, computational experiments show that the pol-

icy based on the suggested value function approximation performs significantly better than a set of standard benchmark methods.

In addition to network revenue management applications, we also consider a capacity allocation problem with a fixed amount of daily processing capacity. Here, the decision maker tries to minimize the cost of scheduling a set of jobs arriving randomly over time to be processed within a given planning horizon. The scheduling (holding) cost of a given job depends on its priority level and the length of its scheduled waiting period. In this setting, the decomposition approach that we suggest decomposes the problem by booking days. In particular, we replace the original dynamic program with a sequence of single-dimensional dynamic programs, each of which is concerned with capacity limitations on one particular booking day only. We show that our approach provides tight lower bounds on the minimum total expected holding cost. Furthermore, it gives rise to a scheduling policy that on average performs better than a variety of benchmark methods known from the literature.

BIOGRAPHICAL SKETCH

Alex received a Masters degree in Financial Management from Comenius University in Bratislava in 2001 and a Masters degree in Economic and Financial Mathematics from Comenius University in Bratislava in 2003. He joined the graduate program in Operations Research at Cornell in 2004.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor, Huseyin Topaloglu. Without his constant support, guidance, and infinite patience, this work would not have been possible. I have greatly benefited from his curious nature, mathematical insight as well as from his ability to be always available, friendly and attentive. For this, I am greatly indebted.

Next, I would like to thank Michael Todd and Levon Barseghyan for their valuable input while serving as my special committee members. I am also grateful to David Williamson and Viktor Tsyrennikov for agreeing to serve as proxies during my final examination.

More generally, I would like to extend my gratitude to all faculty members in the School of Operations Research and Information Engineering. I have greatly benefited from the quality of teaching and friendly environment that this department offers. Additionally, I would like to thank Kathy King, Cindy Jay, Selene Cammer and Eric Johnson for their kind and efficient administrative and technical support.

I am also grateful to my fellow graduate students for making my Cornell experience so much more enjoyable. I could not ask for better classmates than Bikram, Damla, Dennis, Nico and Spyros and I greatly enjoyed the time I spent together with them. Furthermore, I would like to thank Martina, Peter, Marianka, Miloš, Tomáš, Broňa and Lea, the members of wonderful Slovak community in Ithaca, for making my life in the U.S. much more easier.

Finally, nothing would have been possible without the love and encouragement of my family. I am grateful to my parents, Soňa and Ladislav for their constant support, and my brother, Vlado, for taking care of all important things in Slovakia during my absence.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

There are countless examples of decision making under uncertainty in real-world applications. Airlines need to decide whether to sell a given seat on the flight at a discount or, at the risk of not selling this seat at all, to keep this capacity available until the departure at a regular price. Hospitals need to decide whether to schedule a routine MRI test for a low-priority patient at the risk of not being able to serve a randomly arriving high-priority patient later. A baseball manager has to decide whether to keep an effective reliever in a close game for another inning at the risk of not having this pitcher available for the game next day. A common feature of these problems is that the decision is made while the consequences of each potential decision are not completely known. Given that each decision is made at a given time, subject to certain constraints while optimizing a known performance measure, we can naturally model these problems as dynamic programs. Assuming x_t is the state of the system at the beginning of time period t and $V_t(x_t)$ denotes the maximum revenue that the system can generate starting from state x_t at time period t to the end of the planning horizon, we can establish the optimal action at time period t by solving the Bellman equation

$$V_t(x_t) = \max_{a_t \in \mathcal{A}_t(x_t)} \mathbb{E}\left\{r_t(x_t, a_t, \omega_t) + V_{t+1}(X_{t+1}(x_t, a_t, \omega_t))\right\}. \tag{1.1}$$

Here, $\mathcal{A}_t(x_t)$ denotes the set of feasible actions at time period t assuming x_t the state of the system, $r_t(\cdot,\cdot,\cdot)$ the revenue function at time period t, ω_t the random element in the system and $X_{t+1}(\cdot,\cdot,\cdot)$ the state in time period $t+1$.

Aside from the difficulty of taking expectations over potentially unknown distributions, the main challenge with solving equation (1.1) is the (high) dimensionality

of the state space. In fact, for most practical applications, the size of the state vector renders solving the Bellman equation above computationally intractable. This phenomenon is referred to in the literature as the "curse of dimensionality".

In dynamic programming literature, this difficulty is addressed by proposing various types of approximation methods. Roughly speaking, we can identify three main approximation strategies. We note that these are not necessarily mutually exclusive. The first group of methods tries to simplify the optimization problem under consideration using techniques such as aggregating the state or decision variables or eliminating randomness by replacing random variables by their expected values. An example of this approach in network revenue management setting is the static deterministic linear program. The obvious problem with this approach is that it eliminates randomness and hence ignores any local dynamics driven by the random element in each time period. The second group of methods approximates optimal policies by optimizing over a chosen parametric family of control policies. For instance, in network revenue management, one can concentrate on nested protection levels policies only. The problem of this approach is that the chosen parametric family in general does not include the optimal policy. Finally, the third strategy is to decompose the original (multi-dimensional) dynamic program into a collection of one-dimensional dynamic programs. This thesis builds on this last stream of literature.

A unifying feature of our decomposition methods is the idea to approximate the original (multi-dimensional) dynamic programming formulation by decomposing it into a set of one-dimensional dynamic programs. We apply our approach to problems in capacity allocation and network revenue management. In the network revenue management setting, the decomposition approach can be visualized

as an effort to allocate revenue generated by selling a given itinerary among all flight legs associated with this itinerary. Once all itinerary revenues are allocated among its respective flight legs, we can solve a sequence of revenue management problems, each taking place over a single flight leg. As a result, we approximate the original value functions by solving a number of one-dimensional dynamic programs. Alternatively, the decomposition method approximations can be visualized as an application of Lagrangian relaxation to the original dynamic programming formulation. Once the linking constraints in the original dynamic programming formulation are identified, these constraints are relaxed by associating them with corresponding Lagrange multipliers.

The organization of this thesis is as follows. In Chapter 2, we develop a revenue management model to jointly make the capacity allocation and overbooking decisions over an airline network. Our approach begins with the dynamic programming formulation of the capacity allocation and overbooking problem and uses an approximation strategy to decompose the dynamic programming formulation by the flight legs. This decomposition idea opens up the possibility of obtaining approximate solutions by concentrating on one flight leg at a time, but the capacity allocation and overbooking problem that takes place over a single flight leg still turns out to be intractable. We use a state aggregation approach to obtain high quality solutions to the single leg problem. Overall, our model constructs separable approximations to the value functions, which can be used to make the capacity allocation and overbooking decisions for the whole airline network. Computational experiments indicate that our model performs significantly better than a variety of benchmark strategies from the literature.

In Chapter 3, we develop two methods for making pricing decisions in network

revenue management problems. We consider a setting where the probability of observing a request for an itinerary depends on the prices and the objective is to dynamically adjust the prices so as to maximize the total expected revenue. The idea behind both of our methods is to decompose the dynamic programming formulation of the pricing problem by the flight legs and to obtain value function approximations by focusing on one flight leg at a time. We show that our methods provide upper bounds on the optimal total expected revenue and these upper bounds are tighter than the one provided by a deterministic linear program commonly used in practice. Our computational experiments yield two important results. First, our methods provide substantial improvements over the deterministic linear program. The average gap between the total expected revenues obtained by our methods and the deterministic linear program is 7.11%. On average, our methods tighten the upper bounds obtained by the deterministic linear program by 3.66%. Second, the two methods that we develop have different strengths. In particular, while one method is able to obtain tighter upper bounds, the other one is able to obtain pricing policies that yield higher total expected revenues.

In Chapter 4, we consider a dynamic capacity allocation problem with a fixed amount of daily processing capacity. Jobs of different priorities arrive randomly over time and we need to decide which jobs should be scheduled on which days. The jobs that are waiting to be processed incur a holding cost depending on their priority levels. The goal is to minimize the total expected cost over a finite planning horizon. It is possible to formulate this problem as a dynamic program, but this formulation quickly gets intractable for practical problem instances. In this chapter, we propose a dynamic programming decomposition method that addresses this difficulty. The idea behind our model is to decompose the problem by booking days and solve a sequence of single-day capacity allocation problems. We

show that this approach both can be used to make capacity allocation decisions and it provides a lower bound on the optimal total expected costs. Computational experiments indicate that our method performs significantly better than a variety of benchmark strategies.

Chapter 2

A Dynamic Programming Decomposition Method for Making Overbooking Decisions over an Airline Network

2.1 Introduction

Capacity allocation and overbooking are two main ingredients of network revenue management. In particular, capacity allocation deals with the question of which itineraries to keep open for purchase and which itineraries to close as the remaining capacities on the flight legs are depleted over time with the customer purchases. Overbooking deals with the question of to what extent the sales should exceed the physically available capacity on the flight legs, given that not everyone with a reservation ends up showing up at the departure time. The capacity allocation and overbooking decisions are inherently connected. What fare classes to make available for purchase depends on how many seats in excess of the physically available capacity the airline is willing to sell. On the other hand, how much to overbook depends on what itineraries the airline keeps open and the probability that a customer who purchases a reservation for one of the open itineraries shows up at the departure time.

In this chapter, we propose a revenue management model that makes the joint capacity allocation and overbooking decisions over an airline network. Our approach formulates the problem as a dynamic program and uses an approximation

strategy to decompose the dynamic programming formulation by the flight legs. This decomposition idea opens up the possibility of obtaining approximate solutions by concentrating on one flight leg at a time, though the capacity allocation and overbooking problem that takes place over a single flight leg still happens to be intractable. In particular, the state variable in the dynamic programming formulation of the single leg problem involves a large number of dimensions in practical applications. We overcome this difficulty by using state aggregation to obtain high quality solutions to the single leg capacity allocation and overbooking problem. Ultimately, our model provides separable approximations to the value functions, which can be used to construct a capacity allocation and overbooking policy for the whole airline network.

Our work in this chapter draws on two streams of literature. The first stream of literature is the work on dynamic programming decomposition methods in network revenue management. Dynamic programming decomposition methods date back to Belobaba (1987) and they are approximate methods aimed at decomposing the network revenue management problem by the flight legs. The basic idea is to associate a displacement adjusted fare with each itinerary over each flight leg, which is different from the actual fare that the airline charges. The displacement adjusted fares immediately allow us to solve a sequence of single leg revenue management problems. In the single leg revenue management problem that takes place over a particular flight leg, we only concentrate on the remaining capacity on this flight leg and assume that the fares associated with the itineraries are equal to the displacement adjusted fares over this flight leg. Once we have solved the single leg problem over each flight leg, we add up the value functions obtained for different flight legs to obtain separable approximations to the value functions. In this chapter, we use a similar idea to decompose the capacity allocation and overbooking

problem. The important distinction of our approach is that we explicitly deal with overbooking, whereas the earlier decomposition methods work exclusively under the assumption that overbooking is not possible and all reservations show up at the departure time. Our extension to overbooking is nontrivial and has important practical implications as overbooking plays a major role in airline operations.

The second stream of literature that we draw on is the work on solving the capacity allocation and overbooking problem over a single flight leg. This stream of literature becomes especially useful when we try to solve the single leg capacity allocation and overbooking problems after decomposing the original problem. As mentioned above, the single leg capacity allocation and overbooking problem is intractable, as its dynamic programming formulation involves a high dimensional state variable. To be able to solve this problem, we build on the approach proposed by Subramanian, Stidham and Lautenbacher (1999). In particular, we assume that the proportions of the reservations that we have for different itineraries are fixed and known. This allows us to keep track of only the total number of reservations, rather than the number of reservations for each itinerary, in our dynamic programming formulation. In this case, the state variable in the dynamic programming formulation of the single leg problem collapses to a scalar and the dynamic programming formulation becomes tractable.

In this chapter, we make the following research contributions. 1) We develop a model to make the capacity allocation and overbooking decisions over an airline network. The idea behind our model is to decompose the problem by the flight legs and to solve a sequence of single leg problems. 2) We show that our approach provides an upper bound on the optimal total expected profit as long as we can solve the single leg problems accurately. 3) However, noting that the capacity

allocation and overbooking problem over a single flight leg is still intractable, we show how to obtain approximate solutions to the single leg problems in a tractable manner. 4) Computational experiments indicate that our model performs significantly better than many of the existing models in the literature.

The rest of the chapter is organized as follows. In Section 2.2, we review the other literature that is related to our work. Then, in Section 2.3, we give a dynamic programming formulation for the capacity allocation and overbooking problem over an airline network. Section 2.4 gives a simple deterministic linear program that can be used to develop control policies. In Section 2.5, we describe our model by using the deterministic linear program as a starting point. Next, Section 2.6 presents our computational experiments. Finally, in Section 2.7 we provide concluding remarks.

2.2 Review of Related Literature

Despite the fact that there is substantial literature on capacity allocation over an airline network, the interaction between capacity allocation and overbooking is not thoroughly studied. Early models focus on the single leg version of the problem. Beckmann (1958), Thompson (1961) and Coughlan (1999) develop single leg capacity allocation and overbooking models under the assumption that the demands from different fare classes are static random variables. These models ignore the temporal dynamics of the demand process and their goal is to decide how many seats to allocate to different fare classes. Later models by Chatwin (1992), Chatwin (1999) and Subramanian et al. (1999) also consider the single leg problem, but they try to capture the dynamics of the demand process more accurately.

Subramanian et al. (1999) note the intractability of the dynamic programming formulation of the single leg problem and propose the approximation strategy that we build on in this chapter. It is important to contrast their observation with the single leg capacity allocation problem without overbooking. If overbooking is not allowed, then the dynamic programming formulation of the capacity allocation problem involves a scalar state variable and can easily be solved. Therefore, the possibility of overbooking, by itself, brings nontrivial challenges even for the single leg case. Karaesmen and van Ryzin (2004*b*) describe an overbooking model for multiple flight legs that operate between the same origin destination pair and can serve as substitutes of each other.

There are a few papers that consider the overbooking decisions over an airline network. A popular method to make the capacity allocation decisions in network revenue management problems is to solve a deterministic linear program. This linear program is built under the assumption that the itinerary requests are known in advance and they take on their expected values. There is a constraint associated with each flight leg in the linear program and the right sides of these constraints are the leg capacities. Therefore, the optimal values of the dual variables associated with these constraints are used to estimate the opportunity cost of a seat on different flights legs. In this case, one can construct a capacity control policy, where the fare from an itinerary request is compared with the total opportunity cost of the capacities that would be consumed by this itinerary request. If the fare exceeds the total opportunity cost, then the itinerary request is accepted. There are many variants of the linear programming idea and Section 3.3 in Talluri and van Ryzin (2004) describes these variants. We do not go into the details of these variants, as most of them deal only with capacity allocation decisions. However, Bertsimas and Popescu (2003) show how to build a deterministic linear program

to deal with overbooking and no shows. We use a variant of their deterministic linear program as a benchmark strategy in our computational experiments. Karaesmen and van Ryzin (2004*a*) develop a capacity allocation and overbooking model, where they compute booking limits by using the optimal objective value of the deterministic linear program as an estimate of the total expected revenue from the itinerary requests. Gallego and van Ryzin (1997) provide theoretical support for the deterministic linear program by showing that the control policy obtained from a variant of the deterministic linear program is asymptotically optimal as the leg capacities and the expected number of itinerary requests increase linearly at the same rate. Kleywegt (2001*a*) constructs a pricing and overbooking model in continuous time. The demand process that he uses is deterministic and he utilizes Lagrangian duality to solve the model.

The literature on decomposition of network revenue management problems is also related to our work. Williamson (1992) is one of the first to decompose the network revenue management problem by the flight legs. Her goal is to apply the expected marginal seat revenue heuristic of Belobaba (1987) on each flight leg individually to construct value function approximations. Section 3.4.4 in Talluri and van Ryzin (2004) describes a more refined variant of her approach in the sense that this variant does not assume that the demand from different fare classes arrive over nonoverlapping time intervals. Liu and van Ryzin (2008) and Bront, Mendez-Diaz and Vulcano (2008) show how to extend decomposition methods to address customer choice behavior among the different itineraries that are available for purchase. Topaloglu (2009) shows that decomposition methods can be visualized as an application of Lagrangian relaxation to the dynamic programming formulation of the network revenue management problem. Kunnumkal and Topaloglu (2009*a*) show that it is possible to extend the observations of Topaloglu (2009) to address

the customer choice behavior. There is some recent work on decomposing the capacity allocation and overbooking problem over an airline network. Erdelyi and Topaloglu (2009*c*) use separable functions to approximate the value functions in the dynamic programming formulation of the capacity allocation and overbooking problem. In this respect, their paper is connected to our work. However, their approximations are separable by the itineraries and the number of scalar functions they keep is equal to the number of possible itineraries. In contrast, our approximations are separable by the flight legs and the number of scalar functions we keep is equal to the number of flight legs. The number of flight legs is generally smaller than the number of itineraries. Furthermore, they use simulation to construct their scalar functions, whereas we solve small dynamic programs to construct our value function approximations. We use their model as a benchmark strategy.

There are recent approaches for the capacity allocation problem over an airline network. Adelman (2007) uses linear value function approximations for the capacity allocation problem and he chooses the slopes and intercepts of the value function approximations by solving a linear program that represents the dynamic programming formulation of the problem. Zhang and Adelman (2009) extend this approach to deal with customer choice behavior. They also show that decomposition methods can provide upper bounds on the optimal total expected revenue. Meissner and Strauss (2008) refine the approach proposed by Adelman (2007) by using piece-wise linear value function approximations. An important advantage of the recent approaches is that they provide upper bounds on the optimal total expected revenue. However, the recent approaches do not address the interaction between capacity allocation and overbooking and the goal of our paper is to fill this gap.

2.3 Problem Formulation

We consider a set of flight legs over an airline network that can be used to serve the itinerary requests arriving randomly over time. At each time period, an itinerary request arrives and we need to decide whether to accept or reject this itinerary request. An accepted itinerary request becomes a reservation, whereas a rejected itinerary request simply leaves the system. At the departure time of the flight legs, a certain portion of reservations shows up and we need to decide which of these reservations should be allowed boarding. The objective is to maximize total expected profit defined as the difference between the expected revenue obtained by accepting itinerary requests and the expected penalty cost incurred by denying boarding to reservations.

The set of flight legs is $\mathcal{L}$ and the set of itineraries is $\mathcal{J}$. We note that a flight leg is referred to as a resource and an itinerary is referred to as a product in some settings. The problem takes place over the finite planning horizon $\{\tau, \ldots, 0\}$. The itinerary requests arrive over time periods $\mathcal{T} = \{\tau, \ldots, 1\}$ and the flights depart at time period 0. We assume that a time period corresponds to a small enough time interval that there is at most one itinerary request at each time period. The probability that there is a request for itinerary j at time period t is p_{jt}. Accepting a request for itinerary j generates a revenue of f_j and this reservation shows up at the departure time with probability q_j. If a reservation for itinerary j shows up at the departure time and it is denied boarding, then we incur deny penalty cost of θ_j. If we allow boarding to a reservation for itinerary j, then we consume a_{ij} units of capacity on flight leg i. The capacity on flight leg i is c_i. We assume that the arrivals of the itinerary requests at different time periods and the show up decisions of different reservations at the departure time are independent. We also

assume that the reservations are not canceled over time periods $\{\tau, \ldots, 1\}$ and we do not give refunds to the no shows, but these assumptions are for brevity and one can make extensions to address cancellations and refunds.

We let x_{jt} denote the total number of reservations for itinerary j at the beginning of time period t so that $x_t = \{x_{jt} : \ j \in \mathcal{J}\}$ captures the state of the reservations. Assuming that the number of reservations for itinerary j at the beginning of time period 0 is x_{j0}, we use $S_j(x_{j0})$ to denote the number of reservations for itinerary j that show up at the departure time. Given the assumption that the show up decisions of different reservations are independent, $S_j(x_{j0})$ has a binomial distribution with parameters (x_{j0}, q_j). If we use $S(x_0) = \{S_j(x_{j0}) : \ j \in \mathcal{J}\}$ to denote the state of the reservations that show up at the departure time, then we can compute the penalty cost associated with the denied reservations by solving the problem

$$\Gamma(S(x_0)) \ = \ \min \quad \sum_{j \in \mathcal{J}} \theta_j \, w_j \tag{2.1}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{J}} a_{ij} \left[S_j(x_{j0}) - w_j \right] \leq c_i \qquad i \in \mathcal{L} \tag{2.2}$$

$$w_j \leq S_j(x_{j0}) \qquad j \in \mathcal{J} \tag{2.3}$$

$$w_j \in \mathbb{Z}_+ \qquad j \in \mathcal{J}, \tag{2.4}$$

where w_j is the number of reservations for itinerary j that we deny boarding. The objective function of the problem above corresponds to the penalty costs associated with the denied reservations. Constraints (2.2) ensure that the reservations that we allow boarding do not exceed the leg capacities, whereas constraints (2.3) ensure that the numbers of denied reservations do not exceed the numbers of reservations that show up at the departure time. It is important to observe that problem (2.1)-(2.4) assumes that we can jointly decide which reservations should be denied

boarding throughout the network and this can be an optimistic assumption. Letting e_j be the $|\mathcal{J}|$ dimensional unit vector with a one in the element corresponding to j, we can find the optimal policy by computing the value functions through the optimality equation

$$V_t(x_t) = \sum_{j \in \mathcal{J}} p_{jt} \max\{f_j + V_{t-1}(x_t + e_j),\, V_{t-1}(x_t)\} + \Big[1 - \sum_{j \in \mathcal{J}} p_{jt}\Big] V_{t-1}(x_t) \quad (2.5)$$

with the boundary condition that $V_0(x_0) = -\mathbb{E}\{\Gamma(S(x_0))\}$. In this case, if the state of the reservations at the beginning of time period t is given by x_t, then it is optimal to accept a request for itinerary j at time period t whenever

$$f_j \geq V_{t-1}(x_t) - V_{t-1}(x_t + e_j). \quad (2.6)$$

Unfortunately, even for modest sized applications, the state vector x_t involves hundreds of dimensions rendering exact solution to the optimality equation in (2.5) computationally intractable. In the next section, we begin by describing an approximate solution method that involves solving a deterministic linear program. Following this, we build on the deterministic linear program to develop a more sophisticated approximate solution method.

2.4 Deterministic Linear Program

A standard solution method for the network revenue management problem described in the previous section involves solving a deterministic linear program. This linear program is formulated under the assumption that the arrivals of the itinerary requests and the show up decisions of the reservations take on their expected values. In particular, if we let z_j be the number of requests for itinerary j

that we plan to accept over the planning horizon and w_j be the number of reservations that we plan to deny boarding, then this linear program can be formulated as

$$\max \quad \sum_{j \in \mathcal{J}} f_j \, z_j - \sum_{j \in \mathcal{J}} \theta_j \, w_j \tag{2.7}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{J}} a_{ij} \, [q_j \, z_j - w_j] \leq c_i \qquad i \in \mathcal{L} \tag{2.8}$$

$$z_j \leq \sum_{t \in \mathcal{T}} p_{jt} \qquad j \in \mathcal{J} \tag{2.9}$$

$$w_j - q_j \, z_j \leq 0 \qquad j \in \mathcal{J} \tag{2.10}$$

$$z_j, \, w_j \geq 0 \qquad j \in \mathcal{J}. \tag{2.11}$$

In the problem above, we assume that if we accept z_j requests for itinerary j, then $q_j \, z_j$ reservations for itinerary j show up at the departure time. Constraints (2.8) ensure that the numbers of reservations that we allow boarding do not exceed the leg capacities. Constraints (2.9) ensure that the numbers of itinerary requests that we accept do not exceed the expected numbers of itinerary requests. Constraints (2.10) ensure that the numbers of denied reservations do not exceed the expected numbers of reservations that show up at the departure time. The deterministic linear programming formulation for the network revenue management problem is widely known under the assumption that overbooking is not possible and all reservations show up at the departure time; see Talluri and van Ryzin (1998). Problem (2.7)-(2.11) extends this formulation to handle overbooking and no shows. Although this extension is quite intuitive, to our knowledge, Bertsimas and Popescu (2003) is the only reference to this extension.

One use of problem (2.7)-(2.11) is that its dual solution can be used to construct a policy to accept or reject the itinerary requests. Letting $\{\lambda_i^* : i \in \mathcal{L}\}$ be optimal values of the dual variables associated with constraints (2.8) in problem (2.7)-

(2.11), we use λ_i^* to estimate the opportunity cost of a unit of capacity on flight leg i. In this case, if the revenue from an itinerary request exceeds the total expected opportunity cost of the capacities consumed by this itinerary request or if the revenue from an itinerary request exceeds the expected penalty cost, then we accept the itinerary request. In other words, if we have

$$f_j \geq \min\Big\{q_j \sum_{i \in \mathcal{L}} a_{ij}\, \lambda_i^*,\, q_j\, \theta_j\Big\}, \tag{2.12}$$

then we accept a request for itinerary j. The two arguments of the $\min\{\cdot,\cdot\}$ operator above capture two effects. If the total expected opportunity cost of the capacities consumed by a request for itinerary j is small enough that we have $f_j \geq q_j \sum_{i \in \mathcal{L}} a_{ij}\, \lambda_i^*$, then we accept a request for itinerary j. Furthermore, if we have $f_j \geq q_j\, \theta_j$, then we can generate revenue, in expectation, simply by accepting a request for itinerary j and denying boarding to this reservation at the departure time. We refer to the decision rule in (2.12) as the DLP policy, standing for deterministic linear program. This decision rule is also used by Bertsimas and Popescu (2003).

One other use of problem (2.7)-(2.11) is that its optimal objective value provides an upper bound on the optimal total expected profit. In other words, letting z_{LP} be the optimal objective value of problem (2.7)-(2.11) and $\bar{0}$ be the $|\mathcal{J}|$ dimensional vector of zeros, it is possible to show that $V_\tau(\bar{0}) \leq z_{LP}$. For future reference, we state this result as a proposition below. The proof of this proposition can be found in Erdelyi and Topaloglu (2009*c*).

Proposition 2.4.1 *We have* $V_\tau(\bar{0}) \leq z_{LP}$.

The upper bound in Proposition 2.4.1 can be useful when assessing the optimality gap of a suboptimal decision rule such as the DLP policy in (2.12).

2.5 Dynamic Programming Decomposition

There are several shortcomings of the deterministic linear program. It only uses the total expected numbers of the itinerary requests, ignoring the probability distributions and the temporal dynamics of the arrivals of the itinerary requests. Furthermore, it assumes that the numbers of reservations that show up at the departure time take on their expected values. In this section, we build on the deterministic linear program to develop a solution method that captures the temporal dynamics of the itinerary requests somewhat more accurately.

2.5.1 Decomposing into Single Leg Revenue Management Problems

The starting point for our approach is a duality argument on the deterministic linear program to decompose the network revenue management problem into a sequence of single leg revenue management problems. We begin letting $\{\lambda_i^* : i \in \mathcal{L}\}$ be the optimal values of the dual variables associated with constraints (2.8) in problem (2.7)-(2.11). We choose an arbitrary flight leg i and relax constraints (2.8) in problem (2.7)-(2.11) for all other flight legs by associating the dual multipliers $\{\lambda_l^* : l \in \mathcal{L} \setminus \{i\}\}$. In this case, linear programming duality implies that problem (2.7)-(2.11) has the same optimal objective value as the problem

$$\begin{aligned} \max \quad & \sum_{j \in \mathcal{J}} \Big[f_j - q_j \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj}\, \lambda_l^* \Big] z_j - \sum_{j \in \mathcal{J}} \Big[\theta_j - \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj}\, \lambda_l^* \Big] w_j + \sum_{l \in \mathcal{L} \setminus \{i\}} \lambda_l^*\, c_l \\ \text{subject to} \quad & \sum_{j \in \mathcal{J}} a_{ij}\, [q_j\, z_j - w_j] \leq c_i \\ & (2.9), (2.10), (2.11). \end{aligned}$$

We note that the problem above includes the capacity constraint only for flight leg i. For notational brevity, we let

$$\Lambda_j^i = \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj}\, \lambda_l^* \qquad F_j^i = f_j - q_j\, \Lambda_j^i \qquad \Theta_j^i = \theta_j - \Lambda_j^i. \tag{2.13}$$

Omitting the constant term $\sum_{l \in \mathcal{L} \setminus \{i\}} \lambda_l^*\, c_l$, we write the problem above as

$$\max \quad \sum_{j \in \mathcal{J}} F_j^i\, z_j - \sum_{j \in \mathcal{J}} \Theta_j^i\, w_j \tag{2.14}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{J}} a_{ij}\, [q_j\, z_j - w_j] \leq c_i \tag{2.15}$$

$$(2.9), (2.10), (2.11), \tag{2.16}$$

in which case, the optimal objective value of problem (2.14)-(2.16) differs from z_{LP} by $\sum_{l \in \mathcal{L} \setminus \{i\}} \lambda_l^*\, c_l$.

The decision variables z_j and w_j do not appear in constraint (2.15) whenever itinerary j does not use the capacity on flight leg i. This observation allows us to decompose problem (2.14)-(2.16) into two problems, one of which involves the itineraries that use the capacity on flight leg i and the other one involves the remaining itineraries. To this end, we let $\mathcal{J}^i = \{j \in \mathcal{J} : a_{ij} > 0\}$ so that $\mathcal{J}^i$ is the set of itineraries that use the capacity on flight leg i. In this case, it is easy to see that the optimal objective value of problem (2.14)-(2.16) is equal to the sum of the optimal objective values of the problem

$$\max \quad \sum_{j \in \mathcal{J}^i} F_j^i\, z_j - \sum_{j \in \mathcal{J}^i} \Theta_j^i\, w_j \tag{2.17}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{J}^i} a_{ij}\, [q_j\, z_j - w_j] \leq c_i \tag{2.18}$$

$$z_j \leq \sum_{t \in \mathcal{T}} p_{jt} \qquad j \in \mathcal{J}^i \tag{2.19}$$

$$w_j - q_j\, z_j \leq 0 \qquad j \in \mathcal{J}^i \tag{2.20}$$

$$z_j,\, w_j \geq 0 \qquad j \in \mathcal{J}^i, \tag{2.21}$$

which involves only the decision variables $\{z_j : j \in \mathcal{J}^i\}$ and $\{w_j : j \in \mathcal{J}^i\}$, and the problem

$$\max \quad \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} F_j^i \, z_j - \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \Theta_j^i \, w_j \tag{2.22}$$

$$\text{subject to} \quad z_j \leq \sum_{t \in \mathcal{T}} p_{jt} \qquad j \in \mathcal{J} \setminus \mathcal{J}^i \tag{2.23}$$

$$w_j - q_j \, z_j \leq 0 \qquad j \in \mathcal{J} \setminus \mathcal{J}^i \tag{2.24}$$

$$z_j, \, w_j \geq 0 \qquad j \in \mathcal{J} \setminus \mathcal{J}^i, \tag{2.25}$$

which involves only the decision variables $\{z_j : j \in \mathcal{J} \setminus \mathcal{J}^i\}$ and $\{w_j : j \in \mathcal{J} \setminus \mathcal{J}^i\}$. It turns out that the optimal objective value of problem (2.22)-(2.25) can easily be obtained by mere inspection. In particular, we show in Appendix A.1 that the optimal objective value of problem (2.22)-(2.25) is equal to $\sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} p_{jt} \, \max\{F_j^i, F_j^i - q_j \, \Theta_j^i, 0\}$. Therefore, summing up the discussion so far in this section, if we let z_{LP}^i be the optimal objective value of problem (2.17)-(2.21), then we have

$$z_{LP} = z_{LP}^i + \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} p_{jt} \, \max\{F_j^i, F_j^i - q_j \, \Theta_j^i, 0\} + \sum_{l \in \mathcal{L} \setminus \{i\}} \lambda_l^* \, c_l. \tag{2.26}$$

Comparing problem (2.17)-(2.21) with problem (2.7)-(2.11), we can observe that problem (2.17)-(2.21) is the deterministic linear program corresponding to a single leg revenue management problem that takes place over flight leg i. In this single leg revenue management problem, only the requests for the itineraries in the set $\mathcal{J}^i$ are considered. If we accept a request for itinerary j, then we generate a revenue of F_j^i. If we deny boarding to a reservation for itinerary j, then we incur a penalty cost of Θ_j^i. Since z_{LP}^i denotes the optimal objective value of problem (2.17)-(2.21), Proposition 2.4.1 implies that z_{LP}^i provides an upper bound on the optimal total expected profit for the single leg revenue management problem that takes place over flight leg i.

On the other hand, we can compute the optimal total expected profit for the above described single leg revenue management problem taking place over flight leg i by solving the corresponding dynamic program. To this end, we introduce some new notation. We let $\mathcal{R}^i(\cdot)$ be the operator that restricts the components of a $|\mathcal{J}|$ dimensional vector to those that correspond to the elements of $\mathcal{J}^i$. For example, we have $\mathcal{R}^i(x_t) = \{x_{jt} : j \in \mathcal{J}^i\}$ and $\mathcal{R}^i(S(x_0)) = \{S_j(x_{j0}) : j \in \mathcal{J}^i\}$. In this case, the optimality equation for the single leg revenue management problem that takes place over flight leg i reads

$$V_t^i(\mathcal{R}^i(x_t)) = \sum_{j \in \mathcal{J}^i} p_{jt} \max\{F_j^i + V_{t-1}^i(\mathcal{R}^i(x_t + e_j)), V_{t-1}^i(\mathcal{R}^i(x_t))\} + \Big[1 - \sum_{j \in \mathcal{J}^i} p_{jt}\Big] V_{t-1}^i(\mathcal{R}^i(x_t)) \tag{2.27}$$

with the boundary condition that $V_0^i(\mathcal{R}^i(x_0)) = -\mathbb{E}\{\Gamma^i(\mathcal{R}^i(S(x_0)))\}$. Here, $\Gamma^i(\cdot)$ accounts for the penalty cost of denied boarding at the departure time in the single leg revenue management problem that takes place over flight leg i and it is given by

$$\Gamma^i(\mathcal{R}^i(S(x_0))) = \min \quad \sum_{j \in \mathcal{J}^i} \Theta_j^i \, w_j \tag{2.28}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{J}^i} a_{ij} \left[S_j(x_{j0}) - w_j\right] \leq c_i \tag{2.29}$$

$$w_j \leq S_j(x_{j0}) \qquad j \in \mathcal{J}^i \tag{2.30}$$

$$w_j \in \mathbb{Z}_+ \qquad j \in \mathcal{J}^i. \tag{2.31}$$

We recall that z_{LP}^i provides an upper bound on the optimal total expected profit for the single leg revenue management problem that takes place over flight leg i. This optimal total expected profit is given by $V_\tau^i(\mathcal{R}^i(\bar{0}))$ so that we obtain $V_\tau^i(\mathcal{R}^i(\bar{0})) \leq z_{LP}^i$. The next proposition shows the relationship between the so-

lutions to the optimality equations in (2.5) and (2.27). Its proof is in Appendix A.2.

Proposition 2.5.1 *For all $t \in \mathcal{T}$, we have*

$$V_t(x_t) \leq V_t^i(\mathcal{R}^i(x_t)) - \sum_{j \in \mathcal{J}^i} q_j \, \Lambda_j^i \, x_{jt} - \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \min \left\{ q_j \sum_{l \in \mathcal{L}} a_{lj} \, \lambda_l^*, \, q_j \, \theta_j \right\} x_{jt}$$
$$+ \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \sum_{s=1}^{t} p_{js} \, \max\{F_j^i, F_j^i - q_j \, \Theta_j^i, 0\} + \sum_{l \in \mathcal{L} \setminus \{i\}} \lambda_l^* \, c_l. \quad (2.32)$$

Using Proposition 2.5.1 with $t = \tau$ and $x_t = \bar{0}$, the discussion just before this proposition implies that

$$V_\tau(\bar{0}) \leq V_\tau^i(\mathcal{R}^i(\bar{0})) + \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \sum_{s=1}^{\tau} p_{js} \, \max\{F_j^i, F_j^i - q_j \, \Theta_j^i, 0\} + \sum_{l \in \mathcal{L} \setminus \{i\}} \lambda_l^* \, c_l$$
$$\leq z_{LP}^i + \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \sum_{s=1}^{\tau} p_{js} \, \max\{F_j^i, F_j^i - q_j \, \Theta_j^i, 0\} + \sum_{l \in \mathcal{L} \setminus \{i\}} \lambda_l^* \, c_l = z_{LP},$$

where the last equality follows from (2.26). Therefore, we can obtain an upper bound on the optimal total expected profit by solving the optimality equation in (2.27) and this upper bound is tighter than the one provided by the optimal objective value of problem (2.7)-(2.11). Nevertheless, the state variable in the optimality equation in (2.27) has still $|\mathcal{J}^i|$ dimensions, which can be quite large for many practical applications. Before we describe one method to approximate the solution to this optimality equation, we take a quick detour in the next section and describe how we can use the upper bound in Proposition 2.5.1 to construct a policy to accept or reject the itinerary requests.

2.5.2 Approximating the Optimal Decision Rule

Proposition 2.5.1 suggests approximating $V_t(x_t)$ with the upper bound given by the expression on the right side of (2.32). In particular, using $\tilde{V}_t^i(x_t)$ to denote the expression on the right side of (2.32), we can replace $V_{t-1}(x_t) - V_{t-1}(x_t + e_j)$ in the decision rule in (2.6) with $\tilde{V}_{t-1}^i(x_t) - \tilde{V}_{t-1}^i(x_t + e_j)$ and follow this decision rule to accept or reject the itinerary requests. One ambiguous aspect of this approach is that the choice of flight leg i is arbitrary and the performance of the proposed decision rule can depend on the choice of this flight leg. We work around this ambiguity by computing $\{\tilde{V}_t^i(\cdot) : t \in \mathcal{T}\}$ for all $i \in \mathcal{L}$ so that we can use the average $\sum_{i \in \mathcal{L}} \tilde{V}_t^i(x_t)/|\mathcal{L}|$ as an approximation to $V_t(x_t)$. Noting that $\tilde{V}_t^i(x_t) \geq V_t(x_t)$ for all $i \in \mathcal{L}$, we still have the upper bound that $\sum_{i \in \mathcal{L}} \tilde{V}_t^i(x_t)/|\mathcal{L}| \geq V_t(x_t)$. Thus, we propose approximating $V_{t-1}(x_t) - V_{t-1}(x_t + e_j)$ on the right side of (2.6) by $\sum_{i \in \mathcal{L}} \tilde{V}_{t-1}^i(x_t)/|\mathcal{L}| - \sum_{i \in \mathcal{L}} \tilde{V}_{t-1}^i(x_t + e_j)/|\mathcal{L}|$. The definition of $\tilde{V}_t^i(x_t)$ in (2.32) implies that

$$\tilde{V}_{t-1}^i(x_t) - \tilde{V}_{t-1}^i(x_t + e_j) = \begin{cases} V_{t-1}^i(\mathcal{R}^i(x_t)) - V_{t-1}^i(\mathcal{R}^i(x_t + e_j)) & \\ \qquad\qquad + q_j \, \Lambda_j^i & \text{if } j \in \mathcal{J}^i \\ \min\Big\{q_j \sum_{l \in \mathcal{L}} a_{lj} \, \lambda_l^*, \, q_j \, \theta_j\Big\} & \text{if } j \in \mathcal{J} \setminus \mathcal{J}^i. \end{cases}$$

Therefore, letting $\mathbf{1}(\cdot)$ be the indicator function, if the state of the reservations at time period t is given by x_t, then we accept a request for itinerary j whenever we have

$$f_j \geq \frac{1}{|\mathcal{L}|} \sum_{i \in \mathcal{L}} \mathbf{1}(j \in \mathcal{J}^i) \Big\{V_{t-1}^i(\mathcal{R}^i(x_t)) - V_{t-1}^i(\mathcal{R}^i(x_t + e_j)) + q_j \, \Lambda_j^i\Big\} + \frac{1}{|\mathcal{L}|} \sum_{i \in \mathcal{L}} \mathbf{1}(j \in \mathcal{J} \setminus \mathcal{J}^i) \, \min\Big\{q_j \sum_{l \in \mathcal{L}} a_{lj} \, \lambda_l^*, \, q_j \, \theta_j\Big\}. \quad (2.33)$$

One possible way to look at the decision rule in (2.33) is that each flight leg

contributes one term to the expression on the right side. If flight leg i is used by itinerary j, then this flight leg contributes the term $V^i_{t-1}(\mathcal{R}^i(x_t)) - V^i_{t-1}(\mathcal{R}^i(x_t + e_j)) + q_j \Lambda^i_j$. If, on the other hand, flight leg i is not used by itinerary j, then this flight leg contributes the term $\min\{q_j \sum_{l \in \mathcal{L}} a_{lj} \lambda^*_l, q_j \theta_j\}$. The important observation is that the term $\min\{q_j \sum_{l \in \mathcal{L}} a_{lj} \lambda^*_l, q_j \theta_j\}$ is identical to the right side of the DLP policy in (2.12). Therefore, the flight legs that are not used by itinerary j do not provide any additional information over what is already provided by the deterministic linear program. Furthermore, the number of flight legs that are not used by itinerary j is likely to be substantially larger than the number of flight legs that are used by itinerary j, which implies that the right side of the expression above is likely to be dominated by the term $\min\{q_j \sum_{l \in \mathcal{L}} a_{lj} \lambda^*_l, q_j \theta_j\}$. Thus, one conjectures that the decision rule in (2.33) performs very much like the DLP policy. A small set of computational experiments confirmed this conjecture.

To overcome this shortcoming, instead of averaging over all flight legs and using $\sum_{i \in \mathcal{L}} \tilde{V}^i_t(x_t)/|\mathcal{L}|$ as an approximation to $V_t(x_t)$, we average only over the flight legs that are used by a particular itinerary. In particular, we let $\mathcal{L}^j = \{i \in \mathcal{L} : a_{ij} > 0\}$ so that $\mathcal{L}^j$ is the set of flight legs that are used by itinerary j. In this case, whenever we need to make a decision for itinerary j, we use $\sum_{i \in \mathcal{L}^j} \tilde{V}^i_t(x_t)/|\mathcal{L}^j|$ as an approximation to $V_t(x_t)$. We note that we still have the upper bound that $\sum_{i \in \mathcal{L}^j} \tilde{V}^i_t(x_t)/|\mathcal{L}^j| \geq V_t(x_t)$. Thus, if the state of the reservations at time period t is given by x_t, then we accept a request for itinerary j whenever we have

$$f_j \geq \frac{1}{|\mathcal{L}^j|} \sum_{i \in \mathcal{L}^j} \Big\{ V^i_{t-1}(\mathcal{R}^i(x_t)) - V^i_{t-1}(\mathcal{R}^i(x_t + e_j)) + q_j \Lambda^i_j \Big\}. \tag{2.34}$$

The state variable in the optimality equation in (2.27) has $|\mathcal{J}^i|$ dimensions. Theoretically, this is an improvement in comparison to the optimality equation in (2.5), which involves a state variable with $|\mathcal{J}|$ dimensions. Practically, however,

this improvement is irrelevant as $|\mathcal{J}^i|$ is on the order of hundreds or thousands even for modest applications. Therefore, it is still quite difficult to compute the value functions $\{V_t^i(\cdot) : t \in \mathcal{T}\}$ and to use the decision rule in (2.34). In the next section, we give one method to approximate the value functions $\{V_t^i(\cdot) : t \in \mathcal{T}\}$, which seems to work particularly well for our application context.

2.5.3 Reducing the State Space

In this section, we consider the single leg revenue management problem that takes place over flight leg i whose dynamic programming formulation is given in (2.27). Our goal is to approximate the value functions $\{V_t^i(\cdot) : t \in \mathcal{T}\}$ by using simple scalar functions. We observe that the optimality equation in (2.27) has to keep track of the "identities" of the reservations so that the penalty cost given by the optimal objective value of problem (2.28)-(2.31) can be computed properly. On the other hand, if we assume that knowing the total number of reservations is adequate to compute the penalty cost, then the state variable in the optimality equation in (2.27) collapses to a scalar. Our approximation builds on this observation and it is based on approximating the expected penalty cost at the departure time by using only the total number of reservations.

We begin by introducing some new notation. We use $\mathcal{A}^i(\cdot)$ to denote the operator that adds up the components of a $|\mathcal{J}|$ dimensional vector corresponding to the elements of the set $\mathcal{J}^i$. For example, we have $\mathcal{A}^i(x_t) = \sum_{j \in \mathcal{J}^i} x_{jt}$ and $\mathcal{A}^i(x_t)$ is the total number of reservations at the beginning of time period t for the itineraries that use flight leg i. Our approximation is based on the assumption that if we have a total of $\mathcal{A}^i(x_0)$ reservations at the beginning of time period 0 for the itineraries that use flight leg i, then a fixed portion, say α_j^i, of these reservations

are for itinerary j. In this case, recalling that the random variable $S_j(\cdot)$ captures the number of reservations for itinerary j that show up at the departure time and defining the vectors $\alpha^i = \{\alpha^i_j : j \in \mathcal{J}^i\}$, $\alpha^i \mathcal{A}^i(x_t) = \{\alpha^i_j \mathcal{A}^i(x_t) : j \in \mathcal{J}^i\}$ and $S^i(\alpha^i \mathcal{A}^i(x_0)) = \{S_j(\alpha^i_j \mathcal{A}^i(x_0)) : j \in \mathcal{J}^i\}$, we can approximate the penalty cost at the departure time by $\Gamma^i(S^i(\alpha^i \mathcal{A}^i(x_0)))$. In this expression, the vector $\alpha^i \mathcal{A}^i(x_0)$ approximates the numbers of reservations that we have at the beginning of time period 0, whereas the vector $S^i(\alpha^i \mathcal{A}^i(x_0))$ gives the numbers of reservations that show up at the departure time. The function $\Gamma^i(\cdot)$ is given by the optimal objective value of problem (2.28)-(2.31) and it computes the penalty cost for the single leg revenue management problem that takes place over flight leg i. This approximation to the penalty cost at the departure time, in turn, allows us to approximate the solution to the optimality equation in (2.27) by using the solution to the optimality equation

$$v^i_t(\mathcal{A}^i(x_t)) = \sum_{j \in \mathcal{J}^i} p_{jt} \max\{F^i_j + v^i_{t-1}(\mathcal{A}^i(x_t + e_j)), v^i_{t-1}(\mathcal{A}^i(x_t))\} + \Big[1 - \sum_{j \in \mathcal{J}^i} p_{jt}\Big] v^i_{t-1}(\mathcal{A}^i(x_t)) \qquad (2.35)$$

with the boundary condition that $v^i_0(\mathcal{A}^i(x_0)) = -\mathbb{E}\{\Gamma^i(S^i(\alpha^i \mathcal{A}^i(x_0)))\}$. We note that the optimality equation above involves a scalar state variable and it can be solved quite efficiently.

There are three issues that need to be resolved to be able to find a numerical solution to the optimality equation in (2.35). The first issue is related to the choice of α^i_j. We use the DLP policy in (2.12) for this purpose. In particular, we simulate the trajectory of the DLP policy under M itinerary request realizations. Letting $\{x^m_{jt} : j \in \mathcal{J},\ t \in \mathcal{T}\}$ be the state trajectory in the mth itinerary request

realization, we let

$$\alpha_j^i = \frac{\sum_{m=1}^{M} x_{j0}^m}{\sum_{m=1}^{M} \sum_{\tilde{\jmath} \in \mathcal{J}^i} x_{\tilde{\jmath}0}^m}.$$

In practice, it is common to use the DLP policy to come up with an average probability that a reservation shows up at the departure time. Our choice of α_j^i closely follows this approach.

The second issue arises due to the fact that the argument of $S_j(\cdot)$ in the vector $S^i(\alpha^i \mathcal{A}^i(x_0)) = \{S_j(\alpha_j^i \mathcal{A}^i(x_0)) : j \in \mathcal{J}^i\}$ is not necessarily integer. We recall that $S_j(x_{j0})$ is a binomially distributed random variable with parameters (x_{j0}, q_j), but a binomially distributed random variable with a fractional trial parameter is ill-defined. We overcome this issue by always visualizing $S_j(x_{j0})$ as a mixture of two binomially distributed random variables. In particular, letting $\lfloor \cdot \rfloor$ be the round down function, with probability $\lfloor x_{j0} \rfloor + 1 - x_{j0}$, $S_j(x_{j0})$ is equal to a binomially distributed random variable with parameters $(\lfloor x_{j0} \rfloor, q_j)$, and with probability $x_{j0} - \lfloor x_{j0} \rfloor$, $S_j(x_{j0})$ is equal to a binomially distributed random variable with parameters $(\lfloor x_{j0} \rfloor + 1, q_j)$. With this convention, if x_{j0} is integer, then $S_j(x_{j0})$ continues to be binomially distributed with parameters (x_{j0}, q_j). If, however, x_{j0} is fractional, then $S_j(x_{j0})$ is not necessarily binomially distributed, but its expected value continues to be $(\lfloor x_{j0} \rfloor + 1 - x_{j0})\, q_j \lfloor x_{j0} \rfloor + (x_{j0} - \lfloor x_{j0} \rfloor)\, q_j\, (\lfloor x_{j0} \rfloor + 1) = q_j\, x_{j0}$.

Finally, the third issue becomes apparent when we note that the boundary condition of the optimality equation in (2.35) requires computing the expectation $\mathbb{E}\{\Gamma^i(S^i(\alpha^i \mathcal{A}^i(x_0)))\}$ over the multi-dimensional random variable $S^i(\alpha^i \mathcal{A}^i(x_0))$. There is no closed form expression for this expectation and we simply approximate it through Monte Carlo samples.

Once we agree on the resolution of the three issues described above, we can obtain $\{v_t^i(\cdot) : i \in \mathcal{L},\ t \in \mathcal{T}\}$ through the optimality equation in (2.35) and use

$\{v_t^i(\cdot) : i \in \mathcal{L},\ t \in \mathcal{T}\}$ as approximations to $\{V_t^i(\cdot) : i \in \mathcal{L},\ t \in \mathcal{T}\}$ in the decision rule in (2.34). In particular, if the state of the reservations at the beginning of time period t is given by x_t, then we accept a request for itinerary j whenever we have

$$f_j \geq \frac{1}{|\mathcal{L}^j|} \sum_{i \in \mathcal{L}^j} \Big\{ v_{t-1}^i(\mathcal{A}^i(x_t)) - v_{t-1}^i(\mathcal{A}^i(x_t + e_j)) + q_j \, \Lambda_j^i \Big\}. \tag{2.36}$$

We refer to this decision rule as the DPD policy, standing for dynamic programming decomposition.

2.6 Computational Experiments

In this section, we compare the performances of the decision rules in (2.12) and (2.36), along with other benchmark strategies. We begin by describing the experimental setup and the benchmark strategies. Following this, we present our computational results.

2.6.1 Experimental Setup

We consider an airline network that consists of a hub and N spokes. This is a key network structure that frequently arises in practice. There are two flight legs associated with each spoke. One of these is from the hub to the spoke, whereas the other one is from the spoke to the hub. The airline offers a high fare and a low fare itinerary associated with each origin destination pair. Therefore, the number of flight legs is $2N$ and the number of itineraries is $2N\,(N+1)$. The fare associated with a high fare itinerary is κ times the fare associated with the corresponding low fare itinerary. The penalty cost of denying boarding to a reservation for itinerary

j is given by $\theta_j = \delta\, f_j + \sigma\, \max\{f_{\tilde{j}} : \tilde{j} \in \mathcal{J}\}$, where δ and σ are two parameters that we change. For request probabilities $\{p_{jt} : j \in \mathcal{J}, t \in \mathcal{T}\}$, we assume the following mechanism. First, we assume that for each origin destination pair (o, d), there is a constant $R_{od} \in \{0, 1\}$ that denotes the total probability that either one of the two associated itineraries is requested at any time period. During the duration of the first third of the booking horizon, i.e. over time periods $\{\tau, \ldots, \lfloor \frac{2}{3}\tau \rfloor\}$, we assume that the probability of requesting a high fare itinerary is zero while the probability of requesting a low fare itinerary is equal to the corresponding R_{od}. Then, in the remaining time periods, we assume that the probability R_{od} is linearly transferred from the low fare itinerary to the high fare itinerary so that in the last time period the probability of requesting the high fare itinerary is equal to the full value of R_{od}. We note that R_{od} values are generated randomly at the beginning so that their sum over all origin destination pairs is equal to 1. The probability that a reservation shows up at the departure time is q and it does not depend on the itinerary. Noting that the total expected demand for the capacity on flight leg i is given by $q \sum_{t\in\mathcal{T}} \sum_{j\in\mathcal{J}} a_{ij}\, p_{jt}$, we measure the tightness of the leg capacities by

$$\rho = \frac{q \sum_{i\in\mathcal{L}} \sum_{t\in\mathcal{T}} \sum_{j\in\mathcal{J}} a_{ij}\, p_{jt}}{\sum_{i\in\mathcal{L}} c_i}.$$

We label our test problems by $(N, \kappa, \delta, \sigma, q, \rho)$ and use $N \in \{4, 8\}$, $\kappa \in \{4, 8\}$, $(\delta, \sigma) \in \{(4, 0), (8, 0), (1, 1)\}$, $q \in \{0.90, 0.95\}$ and $\rho \in \{1.2, 1.6\}$. This provides 48 test problems for our experimental setup. In all of our test problems, we have $\tau = 240$. The online supplement provides the data files for all of our test problems. We describe the format of the data files in Appendix A.3.

It is worthwhile to note that the interaction between κ and (δ, σ) creates interesting situations. For example, when we have $\kappa = 8$ and $(\delta, \sigma) = (4, 0)$, if the revenue associated with a low fare itinerary is f, then the penalty cost associated with this itinerary is $4f$ and the revenue associated with the corresponding high

fare itinerary is $8f$. In this case, if we have a request for the high fare itinerary and a flight leg in this itinerary is already overbooked with a reservation for the low fare itinerary, then we can still accept the high fare itinerary request and deny boarding to the low fare reservation to make a net profit of $8f - 4f$. This corresponds to the case where a high fare itinerary trivially preempts the corresponding low fare itinerary. On the other hand, when we have $\kappa = 4$ and $(\delta, \sigma) = (1, 1)$, such preemptions do not occur. We also note that the test problems with $(\delta, \sigma) = (1, 1)$ tend to have higher penalty costs than the test problems with $(\delta, \sigma) = (8, 0)$, which, in turn, tend to have higher penalty costs than the test problems with $(\delta, \sigma) = (4, 0)$.

2.6.2 Benchmark Strategies

We compare the performances of the following seven benchmark strategies.

Dynamic programming decomposition (DPD) This benchmark strategy corresponds to the DPD policy given by (2.36). We use $M = 100$ when computing $\{\alpha_j^i : i \in \mathcal{L},\ j \in \mathcal{J}\}$. We estimate all expectations through 1,000 Monte Carlo samples. With these settings, the 95% confidence interval for the expectation of α_j^i has precision $\mp 4.1\%$ on the average, whereas the 95% confidence interval for the expected penalty cost incurred at the departure time has precision $\mp 1.8\%$ on the average.

Deterministic linear program (DLP) This benchmark strategy corresponds to the DLP policy in (2.12). The basic variant of this strategy simply solves problem (2.7)-(2.11) to obtain the optimal values of the dual variables associated with constraints (2.8) and uses these dual variable to implement the DLP policy. We use a reoptimized variant of this strategy, where we divide the planning horizon into

K equal segments and resolve an updated version of problem (2.7)-(2.11) for each segment. In particular, given that the state of the reservations at the beginning of the kth segment is $x_{\tau(K-k+1)/K}$, we replace the right hand side of constraints (2.8) with $\{c_i - \sum_{j \in \mathcal{J}} a_{ij}\, q_j\, x_{j,\tau(K-k+1)/K} : i \in \mathcal{L}\}$, the right hand side of constraints (2.9) with $\{\sum_{t=1}^{\tau(K-k+1)/K} p_{jt} : j \in \mathcal{J}\}$ and the right hand side of constraints (2.10) with $\{q_j\, x_{j,\tau(K-k+1)/K} : j \in \mathcal{J}\}$, and solve this modified version of problem (2.7)-(2.11). Letting $\{\lambda_i^* : i \in \mathcal{L}\}$ be the optimal values of the dual variables associated with constraints (2.8), we use these updated values in the decision rule in (2.12) until we resolve problem (2.7)-(2.11) at the beginning of the next segment. We use $K = 20$ in our computational experiments.

Finite differences in the deterministic linear program (FDD) Given that the state of the reservations at the beginning of time period t is x_t, FDD approximates the optimal total expected profit over the time periods $\{t, \ldots, 0\}$ by using the optimal objective value of the problem

$$\begin{aligned}
\max \quad & \sum_{j \in \mathcal{J}} f_j\, z_j - \sum_{j \in \mathcal{J}} \theta_j\, w_j & \\
\text{subject to} \quad & \sum_{j \in \mathcal{J}} a_{ij}\, [q_j\, z_j - w_j] \leq c_i - \sum_{j \in \mathcal{J}} a_{ij}\, q_j\, x_{jt} & i \in \mathcal{L} \\
& z_j \leq \sum_{\tilde{t}=1}^{t} p_{j\tilde{t}} & j \in \mathcal{J} \\
& w_j - q_j\, z_j \leq q_j\, x_{jt} & j \in \mathcal{J} \\
& z_j,\, w_j \geq 0 & j \in \mathcal{J}.
\end{aligned}$$

We denote $L_t(x_t)$ the optimal objective value of the problem above and let FDD use $L_t(x_t)$ as an approximation to $V_t(x_t)$. In this case, we can make the decisions by replacing $\{V_t(\cdot) : t \in \mathcal{T}\}$ in the decision rule in (2.6) with $\{L_t(\cdot) : t \in \mathcal{T}\}$. This approach is proposed in Bertsimas and Popescu (2003).

Similar to DLP, we use a reoptimized version of FDD, where we divide the planning horizon into K equal segments and retune the decision rule at the beginning of each segment. Given that the state of the reservations at the beginning of the kth segment is $x_{\tau(K-k+1)/K}$, we compute $L_{\tau(K-k+1)/K}(x_{\tau(K-k+1)/K})$ $- L_{\tau(K-k+1)/K}(x_{\tau(K-k+1)/K} + e_j)$ for all $j \in \mathcal{J}$. Following the decision rule in (2.6), if we have

$$f_j \geq L_{\tau(K-k+1)/K}(x_{\tau(K-k+1)/K}) - L_{\tau(K-k+1)/K}(x_{\tau(K-k+1)/K} + e_j)$$

then we always accept a request for itinerary j until we reach the beginning of the next segment and retune the decision rule. We use $K = 20$.

Virtual capacities based on a naive computation (VCN) In this benchmark strategy, the airline sets virtual capacities on the flight legs by assuming that the no shows take on their expected values. Following this, the airline makes the capacity allocation decisions under the assumption that all of the reservations show up, but the capacities on the flight legs are equal to the virtual capacities. In other words, noting that a reservation shows up at the departure time with probability q, the airline sets the virtual capacity on flight leg i as $u_i = \lfloor c_i/q \rfloor$ and solves a version of the deterministic linear program in (2.7)-(2.11), which can be stated as

$$\begin{aligned}
\max \quad & \sum_{j \in \mathcal{J}} f_j \, z_j & & (2.37)\\
\text{subject to} \quad & \sum_{j \in \mathcal{J}} a_{ij} \, z_j \leq u_i \qquad & i \in \mathcal{L} \qquad & (2.38)\\
& z_j \leq \sum_{t \in \mathcal{T}} p_{jt} \qquad & j \in \mathcal{J} \qquad & (2.39)\\
& z_j \geq 0 \qquad & j \in \mathcal{J}. \qquad & (2.40)
\end{aligned}$$

We denote $\{\lambda_i^* : i \in \mathcal{L}\}$ the optimal values of the dual variables associated with the first set of constraints above and let VCN use the DLP policy in (2.12). Similar to DLP and FDD, we use a reoptimized version of VCN with 20 reoptimizations.

Virtual capacities based on an economic model (VCE) One criticism for VCN is that it chooses the virtual capacities under the assumption that the no shows take on their expected values. However, depending on the tradeoffs between the fares, penalty costs and show up probabilities, we may want to be more or less aggressive than what the expected values of the no shows suggest. The goal of VCE is to make up for this shortcoming. VCE is proposed in Karaesmen and van Ryzin (2004*a*) and it is based on the following three assumptions. First, the revenue that we make from one unit of capacity on a flight leg is known. We let r_i be the revenue that we make from one unit of capacity on flight leg i. Second, if a reservation uses the capacities on multiple flight legs, then we can allow boarding to this reservation on one flight leg, while denying boarding to the same reservation on another flight leg. Furthermore, the penalty cost of denying boarding to a reservation on a flight leg is known. We let g_i be the penalty cost of denying boarding to a reservation on flight leg i. Third, if the airline sets the virtual capacity on flight leg i as u_i, then it sells exactly u_i reservations on flight leg i.

By the third assumption, if we set the virtual capacity on flight leg i as u_i, then we sell u_i reservations on flight leg i, in which case, the first assumption implies that we generate a revenue of $r_i\, u_i$. On the other hand, if we let $B_i(u_i)$ be a binomially distributed random variable with parameters (u_i, q), then the number of reservations that show up at the departure time is given by $B_i(u_i)$ and the second assumption implies that the expected penalty cost that we incur on flight leg i is $g_i\, \mathbb{E}\{\max\{B_i(u_i) - c_i, 0\}\}$. Therefore, VCE solves the problem $\max_{u_i} r_i\, u_i - g_i\, \mathbb{E}\{\max\{B_i(u_i) - c_i, 0\}\}$ to set the virtual capacity on flight leg i. Once the virtual capacities have been set, VCE proceeds in the same way as VCN.

Karaesmen and van Ryzin (2004*a*) suggest several choices for r_i and g_i. Fol-

lowing their work, we let $R_j = f_j / \sum_{l \in \mathcal{L}} a_{lj}$ and $G_j = \theta_j / \sum_{l \in \mathcal{L}} a_{lj}$ for all $j \in \mathcal{J}$ to evenly distribute the revenue and penalty cost associated with an itinerary over the flight legs that it uses. In this case, we try choosing r_i as $r_i = \sum_{j \in \mathcal{J}^i} R_j / |\mathcal{J}^i|$ or $r_i = \max\{R_j : j \in \mathcal{J}^i\}$ or $r_i = \min\{R_j : j \in \mathcal{J}^i\}$, and g_i as $g_i = \sum_{j \in \mathcal{J}^i} G_j / |\mathcal{J}^i|$ or $g_i = \max\{G_j : j \in \mathcal{J}^i\}$ or $g_i = \min\{G_j : j \in \mathcal{J}^i\}$. Using all combinations of these choices, we have nine different choices for r_i and g_i. We test the performances of all of these nine choices for all of our test problems, but for brevity, only report the results corresponding to the best choice. For different test problems, the best choice for r_i and g_i can be different. Similar to VCN, we use a reoptimized version of VCE with 20 reoptimizations.

Virtual capacities joint with capacity allocation decisions (VCJ) Both VCN and VCE use the assumption that we can set the virtual capacities first, and then, come up with a policy to accept or reject the itinerary requests. In contrast, VCJ uses the penalty cost $g_i \, \mathbb{E}\{\max\{B_i(u_i) - c_i, 0\}\}$ in problem (2.37)-(2.40) to jointly set the virtual capacities and come up with a policy to accept or reject the itinerary requests. In particular, VCJ solves the problem

$$\begin{aligned}
\max \quad & \sum_{j \in \mathcal{J}} f_j \, z_j - \sum_{i \in \mathcal{L}} g_i \, \mathbb{E}\{\max\{B_i(u_i) - c_i, 0\}\} \\
\text{subject to} \quad & \sum_{j \in \mathcal{J}} a_{ij} \, z_j - u_i \leq 0 \qquad i \in \mathcal{L} \\
& z_j \leq \sum_{t \in \mathcal{T}} p_{jt} \qquad j \in \mathcal{J} \\
& z_j, u_i \geq 0 \qquad i \in \mathcal{L}, \; j \in \mathcal{J},
\end{aligned}$$

where we use interpolations of the function $\mathbb{E}\{\max\{B_i(u_i) - c_i, 0\}\}$ to be able to compute it at a fractional u_i. We denote $\{\lambda_i^* : i \in \mathcal{L}\}$ the optimal values of the dual variables associated with the first set of constraints above, and let VCJ use the DLP policy in (2.12). This approach is proposed in Karaesmen and van Ryzin

(2004a). Similar to VCE, we try three different choices for g_i and report the results corresponding to the best choice. We use a reoptimized version of VCJ with 20 reoptimizations.

Separable penalty costs (SPC) This benchmark strategy is developed by Erdelyi and Topaloglu (2009c). The fundamental observation behind SPC is that if the penalty cost of denying boarding to the reservations were given by a separable function of the form $\Gamma(S(x_0)) = \sum_{j \in \mathcal{J}} \gamma_j(S_j(x_{j0}))$, then the optimality equation in (2.5) would decompose by the itineraries. To exploit this observation, SPC approximates $\Gamma(S(x_0))$ in problem (2.1)-(2.4) with a separable function of the form $\sum_{j \in \mathcal{J}} \gamma_j(S_j(x_{j0}))$ and solves the optimality equation in (2.5) with the approximate boundary condition that $V_0(x_0) = -\mathbb{E}\{\sum_{j \in \mathcal{J}} \gamma_j(S_j(x_{j0}))\}$. The value functions $\{V_t(\cdot) : t \in \mathcal{T}\}$ obtained in this fashion are used to construct a policy to accept or reject the itinerary requests. SPC uses a simulation based method to construct the separable approximation $\sum_{j \in \mathcal{J}} \gamma_j(S_j(x_{j0}))$ to the penalty cost. Roughly speaking, we simulate the DLP policy in (2.12) to have a general idea about the numbers of reservations that show up at the departure time. Following this, we compute the slopes of $\Gamma(\cdot)$ at these numbers of reservations along different directions and use these slopes to construct the scalar functions $\{\gamma_j(\cdot) : j \in \mathcal{J}\}$. An exact description of this benchmark strategy is beyond the scope of this chapter and we refer the reader to Erdelyi and Topaloglu (2009c) for the details. Similar to the other benchmark strategies, we retune the separable approximation five times over the planning horizon. It turns out that retuning SPC more than five times does not provide any additional benefit.

2.6.3 Computational Results

Our main computational results are summarized in Tables 2.1 and 2.2. In particular, these two tables respectively show the results for the test problems with four and eight spokes. The first column in Tables 2.1 and 2.2 gives the parameters of the test problem. The second column gives the upper bound on the optimal total expected profit provided by the optimal objective value of problem (2.7)-(2.11). The next seven columns give the total expected profits obtained by DPD, DLP, FDD, VCN, VCE, VCJ and SPC. These total expected profits are estimated by simulating the performances of the different policies under 50 itinerary request trajectories. We use common itinerary request trajectories when simulating the performances of the different policies. The tenth column gives the percent gap between the total expected profits obtained by DPD and DLP. This column also includes a "✓" whenever DPD performs significantly better than DLP and a "⊙" whenever there is no statistically significant performance gap between the two methods at 95% level. The next five columns do the same thing as the tenth column, but they compare the performance of DPD with FDD, VCN, VCE, VCJ and SPC. Finally, the last column shows the percentage gap between performance obtained by DPD and the profit bound reported in the second column. The size of this gap gives an upper bound on the percentage gap between the performance of DPD and the optimal policy.

The results indicate that DPD performs substantially better than all of the benchmark strategies that use a linear programming formulation. Among the linear programming based benchmark strategies, FDD performs the best and it is followed by VCJ, DLP, VCE and VCN. The superiority of FDD over DLP is also observed by Bertsimas and Popescu (2003).

Table 2.1: Computational results for the test problems with four spokes.

Problem	Profit	Total expected profit obtained by							DPD vs.						
$(N, \kappa, \delta, \sigma, q, \rho)$	bound	DPD	DLP	FDD	VCN	VCE	VCJ	SPC	DLP	FDD	VCN	VCE	VCJ	SPC	bnd
$(4, 4, 4, 0, 0.90, 1.2)$	15,223	14,575	14,283	14,386	13,671	14,158	14,359	14,442	2.00 ✓	1.30 ✓	6.21 ✓	2.87 ✓	1.48 ✓	0.91 ✓	4.26
$(4, 4, 4, 0, 0.90, 1.6)$	20,997	19,618	19,085	19,395	18,164	18,716	19,237	19,506	2.72 ✓	1.14 ✓	7.41 ✓	4.60 ✓	1.94 ✓	0.57 ✓	6.57
$(4, 4, 4, 0, 0.95, 1.2)$	23,450	22,108	21,962	21,998	21,510	21,734	21,904	22,062	0.66 ✓	0.50 ✓	2.71 ✓	1.69 ✓	0.92 ✓	0.21 ⊙	5.72
$(4, 4, 4, 0, 0.95, 1.6)$	21,753	20,647	20,208	20,373	19,645	19,873	20,088	20,532	2.13 ✓	1.33 ✓	4.85 ✓	3.75 ✓	2.71 ✓	0.56 ✓	5.08
$(4, 4, 8, 0, 0.90, 1.2)$	23,136	21,672	21,014	21,144	20,695	21,103	21,359	21,487	3.03 ✓	2.43 ✓	4.50 ✓	2.62 ✓	1.44 ✓	0.85 ✓	6.33
$(4, 4, 8, 0, 0.90, 1.6)$	12,177	11,157	10,326	10,620	10,350	10,649	10,741	11,047	7.45 ✓	4.81 ✓	7.24 ✓	4.55 ✓	3.73 ✓	0.99 ✓	8.38
$(4, 4, 8, 0, 0.95, 1.2)$	19,206	17,883	17,285	17,491	17,291	17,431	17,582	17,838	3.35 ✓	2.19 ✓	3.31 ✓	2.53 ✓	1.68 ✓	0.25 ⊙	6.89
$(4, 4, 8, 0, 0.95, 1.6)$	15,995	14,808	14,059	14,316	13,999	14,119	14,198	14,526	5.06 ✓	3.32 ✓	5.46 ✓	4.66 ✓	4.12 ✓	1.90 ✓	7.42
$(4, 4, 1, 1, 0.90, 1.2)$	18,418	16,970	16,362	16,529	16,241	16,543	16,667	16,894	3.58 ✓	2.60 ✓	4.30 ✓	2.52 ✓	1.79 ✓	0.45 ⊙	7.86
$(4, 4, 1, 1, 0.90, 1.6)$	10,626	9,813	9,159	9,372	9,209	9,432	9,506	9,727	6.66 ✓	4.50 ✓	6.16 ✓	3.89 ✓	3.13 ✓	0.88 ✓	7.65
$(4, 4, 1, 1, 0.95, 1.2)$	19,782	18,538	17,797	17,968	17,731	17,897	18,087	18,332	4.00 ✓	3.08 ✓	4.35 ✓	3.46 ✓	2.43 ✓	1.11 ✓	6.29
$(4, 4, 1, 1, 0.95, 1.6)$	17,345	16,072	15,264	15,522	15,210	15,408	15,627	16,019	5.03 ✓	3.42 ✓	5.36 ✓	4.13 ✓	2.77 ✓	0.33 ⊙	7.34
$(4, 8, 4, 0, 0.90, 1.2)$	30,754	29,514	29,286	29,329	26,911	28,065	29,217	29,445	0.77 ✓	0.63 ✓	8.82 ✓	4.91 ✓	1.01 ✓	0.23 ✓	4.03
$(4, 8, 4, 0, 0.90, 1.6)$	31,744	30,841	30,324	30,483	27,204	28,548	30,187	30,679	1.68 ✓	1.16 ✓	11.79 ✓	7.43 ✓	2.12 ✓	0.52 ✓	2.84
$(4, 8, 4, 0, 0.95, 1.2)$	28,983	27,676	27,386	27,445	25,854	26,201	27,160	27,533	1.05 ✓	0.83 ✓	6.58 ✓	5.33 ✓	1.86 ✓	0.51 ✓	4.51
$(4, 8, 4, 0, 0.95, 1.6)$	23,995	22,983	22,720	22,825	21,200	21,616	22,322	22,901	1.14 ✓	0.69 ✓	7.76 ✓	5.95 ✓	2.88 ✓	0.36 ✓	4.22
$(4, 8, 8, 0, 0.90, 1.2)$	26,932	25,888	25,115	25,182	23,503	24,376	25,198	25,825	2.98 ✓	2.72 ✓	9.21 ✓	5.84 ✓	2.66 ✓	0.24 ⊙	3.88
$(4, 8, 8, 0, 0.90, 1.6)$	30,670	28,617	27,314	27,731	25,859	26,718	27,910	28,527	4.55 ✓	3.10 ✓	9.64 ✓	6.64 ✓	2.47 ✓	0.31 ⊙	6.69
$(4, 8, 8, 0, 0.95, 1.2)$	33,136	31,787	31,134	31,267	30,054	30,551	30,989	31,816	2.05 ✓	1.64 ✓	5.45 ✓	3.89 ✓	2.51 ✓	-0.09 ⊙	4.07
$(4, 8, 8, 0, 0.95, 1.6)$	27,926	26,747	25,456	25,781	24,482	24,712	24,931	26,533	4.83 ✓	3.61 ✓	8.47 ✓	7.61 ✓	6.79 ✓	0.80 ✓	4.22
$(4, 8, 1, 1, 0.90, 1.2)$	26,673	25,187	23,050	23,446	23,238	24,046	24,446	25,114	8.48 ✓	6.92 ✓	7.74 ✓	4.53 ✓	2.94 ✓	0.29 ⊙	5.57
$(4, 8, 1, 1, 0.90, 1.6)$	31,470	29,730	27,215	28,178	26,748	27,776	28,623	29,498	8.46 ✓	5.22 ✓	10.03 ✓	6.57 ✓	3.72 ✓	0.78 ✓	5.53
$(4, 8, 1, 1, 0.95, 1.2)$	21,959	20,858	19,411	19,832	19,506	19,799	20,062	20,668	6.94 ✓	4.92 ✓	6.48 ✓	5.08 ✓	3.82 ✓	0.91 ✓	5.01
$(4, 8, 1, 1, 0.95, 1.6)$	26,138	24,341	22,010	22,825	21,959	22,105	22,423	24,040	9.58 ✓	6.23 ✓	9.79 ✓	9.19 ✓	7.88 ✓	1.24 ✓	6.88
Average									4.09	2.85	6.82	4.76	2.87	0.63	5.72

Table 2.2: Computational results for the test problems with eight spokes.

Problem	Profit	Total expected profit obtained by							DPD vs.						
$(N, \kappa, \delta, \sigma, q, \rho)$	bound	DPD	DLP	FDD	VCN	VCE	VCJ	SPC	DLP	FDD	VCN	VCE	VCJ	SPC	bnd
$(8, 4, 4, 0, 0.90, 1.2)$	22,706	20,671	20,338	20,463	19,114	19,773	20,428	20,426	1.61 ✓	1.00 ✓	7.53 ✓	4.34 ✓	1.17 ✓	1.18 ✓	8.96
$(8, 4, 4, 0, 0.90, 1.6)$	17,715	15,951	15,555	15,783	14,491	15,124	15,647	15,823	2.48 ✓	1.05 ✓	9.15 ✓	5.19 ✓	1.90 ✓	0.80 ✓	9.96
$(8, 4, 4, 0, 0.95, 1.2)$	21,809	19,960	19,640	19,795	18,903	19,094	19,596	19,829	1.60 ✓	0.83 ✓	5.29 ✓	4.34 ✓	1.82 ✓	0.65 ✓	8.48
$(8, 4, 4, 0, 0.95, 1.6)$	15,625	14,112	13,771	14,025	13,058	13,080	13,606	13,982	2.41 ✓	0.61 ✓	7.47 ✓	7.31 ✓	3.59 ✓	0.92 ✓	9.68
$(8, 4, 8, 0, 0.90, 1.2)$	19,963	17,747	16,783	17,223	16,529	17,006	17,510	17,603	5.43 ✓	2.95 ✓	6.87 ✓	4.18 ✓	1.34 ✓	0.82 ✓	11.10
$(8, 4, 8, 0, 0.90, 1.6)$	13,868	12,189	11,408	11,740	11,211	11,644	11,854	11,961	6.40 ✓	3.68 ✓	8.02 ✓	4.47 ✓	2.75 ✓	1.87 ✓	12.11
$(8, 4, 8, 0, 0.95, 1.2)$	21,134	18,957	18,203	18,589	18,119	18,239	18,382	18,740	3.98 ✓	1.95 ✓	4.42 ✓	3.79 ✓	3.04 ✓	1.15 ✓	10.30
$(8, 4, 8, 0, 0.95, 1.6)$	17,056	14,658	13,768	14,251	13,904	13,920	14,047	14,484	6.08 ✓	2.78 ✓	5.14 ✓	5.04 ✓	4.17 ✓	1.19 ✓	14.06
$(8, 4, 1, 1, 0.90, 1.2)$	20,019	17,932	16,785	17,274	16,878	17,325	17,546	17,867	6.39 ✓	3.67 ✓	5.88 ✓	3.38 ✓	2.15 ✓	0.36 ⊙	10.43
$(8, 4, 1, 1, 0.90, 1.6)$	15,712	13,638	12,488	13,057	12,440	12,902	13,113	13,498	8.43 ✓	4.26 ✓	8.78 ✓	5.39 ✓	3.85 ✓	1.03 ✓	13.20
$(8, 4, 1, 1, 0.95, 1.2)$	16,179	14,145	13,427	13,816	13,637	13,807	13,891	14,096	5.08 ✓	2.32 ✓	3.59 ✓	2.39 ✓	1.79 ✓	0.35 ⊙	12.57
$(8, 4, 1, 1, 0.95, 1.6)$	19,803	17,255	16,119	16,792	16,431	16,492	16,655	17,171	6.58 ✓	2.68 ✓	4.78 ✓	4.42 ✓	3.48 ✓	0.49 ⊙	12.87
$(8, 8, 4, 0, 0.90, 1.2)$	35,075	33,024	32,742	32,905	29,371	30,806	32,739	32,879	0.85 ✓	0.36 ✓	11.06 ✓	6.71 ✓	0.86 ✓	0.44 ✓	5.85
$(8, 8, 4, 0, 0.90, 1.6)$	24,105	22,528	22,120	22,285	19,609	20,448	22,054	22,336	1.81 ✓	1.08 ✓	12.96 ✓	9.23 ✓	2.10 ✓	0.85 ✓	6.54
$(8, 8, 4, 0, 0.95, 1.2)$	33,872	32,052	31,832	31,954	29,486	29,958	31,401	31,808	0.69 ✓	0.30 ✓	8.00 ✓	6.53 ✓	2.03 ✓	0.76 ✓	5.37
$(8, 8, 4, 0, 0.95, 1.6)$	25,920	24,254	23,844	24,056	22,142	22,256	23,403	24,069	1.69 ✓	0.82 ✓	8.71 ✓	8.24 ✓	3.51 ✓	0.76 ✓	6.43
$(8, 8, 8, 0, 0.90, 1.2)$	31,831	29,377	28,228	28,680	26,432	27,564	28,667	29,046	3.91 ✓	2.37 ✓	10.03 ✓	6.17 ✓	2.42 ✓	1.13 ✓	7.71
$(8, 8, 8, 0, 0.90, 1.6)$	37,769	34,116	32,225	33,149	29,732	31,176	32,782	33,955	5.54 ✓	2.84 ✓	12.85 ✓	8.62 ✓	3.91 ✓	0.47 ⊙	9.67
$(8, 8, 8, 0, 0.95, 1.2)$	28,695	26,540	25,549	25,984	24,661	24,860	25,327	26,245	3.74 ✓	2.09 ✓	7.08 ✓	6.33 ✓	4.57 ✓	1.11 ✓	7.51
$(8, 8, 8, 0, 0.95, 1.6)$	32,840	30,092	28,585	29,292	27,706	27,703	28,119	29,804	5.01 ✓	2.66 ✓	7.93 ✓	7.94 ✓	6.56 ✓	0.96 ✓	8.37
$(8, 8, 1, 1, 0.90, 1.2)$	29,394	26,832	23,601	24,840	24,442	25,266	25,681	26,660	12.04 ✓	7.42 ✓	8.90 ✓	5.84 ✓	4.29 ✓	0.64 ✓	8.72
$(8, 8, 1, 1, 0.90, 1.6)$	28,433	25,863	22,104	23,688	22,464	23,741	24,129	25,534	14.53 ✓	8.41 ✓	13.14 ✓	8.20 ✓	6.71 ✓	1.27 ✓	9.04
$(8, 8, 1, 1, 0.95, 1.2)$	26,884	24,647	22,162	23,190	22,743	22,886	23,096	24,396	10.09 ✓	5.91 ✓	7.73 ✓	7.15 ✓	6.29 ✓	1.02 ✓	8.32
$(8, 8, 1, 1, 0.95, 1.6)$	28,228	25,919	22,559	24,123	23,051	23,117	23,244	25,701	12.96 ✓	6.93 ✓	11.06 ✓	10.81 ✓	10.32 ✓	0.84 ✓	8.18
Average									5.39	2.87	8.18	6.08	3.53	0.88	9.39

The performance gaps between DPD and DLP, FDD, VCN, VCE and VCJ are statistically significant for all of the test problems. The average performance gaps between DPD and DLP, FDD, VCN, VCE and VCJ are respectively 4.09, 2.85, 6.82, 4.76 and 2.82 for the test problems with four spokes. The same gaps increase to 5.39, 2.87, 8.18, 6.08 and 3.53 for the test problems with eight spokes. The average percentage gap between the DPD performance and the upper bound from the deterministic linear program is 5.72 for the test problems with four spokes and 9.39 for the test problems with eight spokes. Among the three benchmark strategies that use virtual capacities, VCJ performs better than VCN and VCE. It is interesting to note that VCJ performs better than DLP as well. There are test problems where VCN performs better than DLP, despite the fact that VCN is essentially an ad hoc modification of DLP that does not carefully address the possibility of no shows. However, the performance of VCN is not robust as indicated by the test problems with $(\delta, \sigma) = (4, 0)$. There is not a clear distinction between DLP and VCE, but there are test problems where VCE can perform substantially worse than DLP.

The performance gap between DPD and SPC is on the order of half a percent to a percent. We emphasize that a percent revenue difference is still considered significant in the revenue management setting. On a majority of the test problems, DPD performs better than SPC and in the remaining test problems, there does not exist a statistically significant gap between the two benchmark strategies. Similar to DPD, SPC performs noticeably better than all of the benchmark strategies that use a linear programming formulation. Therefore, DPD and SPC, by working with the dynamic programming formulation of the capacity allocation and overbooking problem, provide significant improvements over using a deterministic linear programming formulation, which ignores the temporal dynamics of the arrivals of the

itinerary requests.

It is possible to observe a few trends in the performance gaps. In particular, the performance gaps between DPD and the linear programming based benchmark strategies tend to increase as the fare difference between the high fare and low fare itineraries, penalty costs and overall tightness of the leg capacities increase. For test problems with large fare differences, large penalty costs and tight leg capacities, the "regret" associated with making an "incorrect" decision is relatively large. For example, when the fare difference between the high fare and low fare itineraries is large, accepting a request for a low fare itinerary "incorrectly" may preclude accepting a request for a high fare itinerary later in the planning horizon and the revenue forgone in this case can be quite large. Similarly, when the penalty costs are large, it is costly to deny boarding to a reservation that was accepted "by mistake". When the leg capacities are tight, it is important to make the itinerary acceptance decisions more "carefully", as it is not possible to accommodate all itinerary requests. Thus, it is encouraging that a careful stochastic model pays off and DPD performs significantly better than the linear programming based benchmark strategies as the fare differences, penalty costs and tightness of the leg capacities increase. To display some of these trends, Table 2.3 shows the performance gaps between DPD and the other benchmark strategies averaged over a number of test problems with a particular characteristic. For example, the second column shows the performance gaps averaged over the test problems with four spokes. The trends that we mention can be observed from this table.

Table 2.4 shows the CPU seconds required to compute one set of value function approximations for DPD and SPC. All of the computational experiments are run on a Pentium IV desktop PC with 2.4 GHz CPU and 1 GB RAM. Since the

Table 2.3: Comparison of the performances of DPD and the other benchmark strategies for different sets of test problems.

Benchmark strategies	N		κ		(δ, σ)			ρ	
	4	8	4	8	(4,0)	(8,0)	(1,1)	1.2	1.6
DPD vs. DLP	4.09	5.39	4.26	5.22	1.58	4.59	8.05	3.93	5.55
DPD vs. FDD	2.85	2.87	2.43	3.29	0.85	2.82	4.91	2.54	3.18
DPD vs. VCN	6.82	8.18	5.78	9.22	7.89	7.23	7.38	6.50	8.50
DPD vs. VCE	4.76	6.08	3.98	6.86	5.53	5.30	5.43	4.43	6.41
DPD vs. VCJ	2.87	3.53	2.47	3.93	1.99	3.38	4.21	2.53	4.05
DPD vs. SPC	0.63	0.88	0.83	0.68	0.64	0.87	0.75	0.65	0.86

Table 2.4: CPU seconds for DPD and SPC.

Benchmark strategy	N	
	4	8
DPD	48	76
SPC	118	220

number of spokes appears to be the primary factor affecting the computation times, we give the average CPU seconds over different test problems. The two rows in Table 2.4 show the CPU seconds for DPD and SPC. The second and third columns respectively correspond to the test problems with four and eight spokes. The CPU seconds for DPD includes the operations required to estimate $\{\alpha_j^i : i \in \mathcal{L},\ j \in \mathcal{J}\}$ and to compute $\{v_t^i(\cdot) : i \in \mathcal{L},\ t \in \mathcal{T}\}$. The results indicate that DPD takes significantly less time than SPC and scales more favorably. Considering its performance, DPD appears to be preferable to SPC. DLP, FDD, VCN, VCE and VCJ take at most a few seconds to reoptimize their decision rules. Despite this extra computational burden, the computational requirement for DPD is still reasonable. Given the substantial improvements that it provides over the other benchmark strategies, DPD appears to be a viable choice.

2.7 Conclusions

In this chapter, we developed a network revenue management model to jointly make the capacity allocation and overbooking decisions over an airline network. Our approach is based on decomposing the network revenue management problem into a sequence of single leg revenue management problems and exploiting the observation that if the proportions of the reservations at the departure time were known, then the dynamic programming formulation of the single leg revenue management problems would involve only a scalar state variable. Using these observations, we constructed tractable approximations to the value functions. Computational experiments demonstrated that the resulting policies perform significantly better than the benchmark strategies.

APPENDIX

A Appendix to Chapter 2

A.1 Optimal Objective Value of Problem (2.22)-(2.25)

Letting $\mathbf{1}(\cdot)$ be the indicator function and noting constraints (2.24), we note that the optimal values of the decision variables $\{w_j : j \in \mathcal{J} \setminus \mathcal{J}^i\}$ are $\{\mathbf{1}(\Theta^i_j \leq 0)\, q_j\, z_j : j \in \mathcal{J} \setminus \mathcal{J}^i\}$. Therefore, letting $[\cdot]^+ = \max\{\cdot, 0\}$, we can write problem (2.22)-(2.25) as

$$\begin{aligned} \max \quad & \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \left[F^i_j + [-\Theta^i_j]^+ q_j\right] z_j & \\ \text{subject to} \quad & z_j \leq \sum_{t \in \mathcal{T}} p_{jt} & j \in \mathcal{J} \setminus \mathcal{J}^i \\ & z_j \geq 0 & j \in \mathcal{J} \setminus \mathcal{J}^i. \end{aligned}$$

In the problem above, the optimal values of the decision variables $\{z_j : j \in \mathcal{J} \setminus \mathcal{J}^i\}$ are $\{\mathbf{1}([F^i_j + [-\Theta^i_j]^+ q_j] \geq 0) \sum_{t \in \mathcal{T}} p_{jt} : j \in \mathcal{J} \setminus \mathcal{J}^i\}$. Therefore, the optimal objective value of the problem above is

$$\sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \left[F^i_j + [-\Theta^i_j]^+ q_j\right]^+ p_{jt} = \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \max\{F^i_j, F^i_j - q_j\, \Theta^i_j, 0\}\, p_{jt}.$$

A.2 Proof of Proposition 2.5.1

To simplify the proof, we introduce auxiliary value functions $\{\psi^i_t(\cdot) : i \in \mathcal{L},\ t \in \mathcal{T}\}$ by letting

$$\psi^i_t(x_t) = \sum_{j \in \mathcal{J}} p_{jt} \max\{F^i_j + \psi^i_{t-1}(x_t + e_j), \psi^i_{t-1}(x_t)\} + \Big[1 - \sum_{j \in \mathcal{J}} p_{jt}\Big] \psi^i_{t-1}(x_t) \quad \text{(A.1)}$$

with the boundary condition that $\psi_0^i(x_0) = -\mathbb{E}\{\phi^i(S(x_0))\}$, where

$$\phi^i(S(x_0)) = \min \quad \sum_{j \in \mathcal{J}} \Theta_j^i \, w_j \tag{A.2}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{J}} a_{ij} \, [S_j(x_{j0}) - w_j] \leq c_i \tag{A.3}$$

$$w_j \leq S_j(x_{j0}) \qquad j \in \mathcal{J} \tag{A.4}$$

$$w_j \in \mathbb{Z}_+ \qquad j \in \mathcal{J}. \tag{A.5}$$

The following two results provide the intermediate steps to prove Proposition 2.5.1.

Lemma A.2.1 *For all $t \in \mathcal{T}$, we have*

$$\psi_t^i(x_t) = V_t^i(\mathcal{R}^i(x_t)) + \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} q_j \, [-\Theta_j^i]^+ \, x_{jt} + \sum_{s=1}^{t} \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} p_{js} \, \max\{F_j^i, F_j^i - q_j \, \Theta_j^i, 0\}.$$

Proof of Lemma A.2.1 We show the result by induction over the time periods. Noting the upper bounds on the decision variables $\{w_j : j \in \mathcal{J} \setminus \mathcal{J}^i\}$ in problem (A.2)-(A.5), we observe that the optimal values of these decision variables are $\{\mathbf{1}(\Theta_j^i \leq 0) \, S_j(x_{j0}) : j \in \mathcal{J} \setminus \mathcal{J}^i\}$. Thus, since we have $a_{ij} = 0$ for all $j \in \mathcal{J} \setminus \mathcal{J}^i$, we have

$$\phi^i(S(x_0)) = \Gamma^i(\mathcal{R}^i(S(x_0))) - \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} [-\Theta_j^i]^+ \, S_j(x_{j0}),$$

where $\Gamma^i(\mathcal{R}^i(S(x_0)))$ is the optimal objective value of problem (2.28)-(2.31). Taking expectations in the expression above and noting that $S_j(x_{j0})$ has a binomial distribution with parameters (x_{j0}, q_j), we obtain $\psi_0^i(x_0) = -\mathbb{E}\{\phi^i(S(x_0))\} = -\mathbb{E}\{\Gamma^i(\mathcal{R}^i(S(x_0)))\} + \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} q_j \, [-\Theta_j^i]^+ x_{j0} = V_0^i(\mathcal{R}^i(x_0)) + \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} q_j \, [-\Theta_j^i]^+ x_{j0}$ and the result holds for the last time period. Assuming that the result holds for

time period $t-1$, we have

$$\psi^i_{t-1}(x_t + e_j) - \psi^i_{t-1}(x_t) = \begin{cases} V^i_{t-1}(\mathcal{R}^i(x_t + e_j)) - V^i_{t-1}(\mathcal{R}^i(x_t)) & \text{if } j \in \mathcal{J}^i \\ q_j \, [-\Theta^i_j]^+ & \text{if } j \in \mathcal{J} \setminus \mathcal{J}^i. \end{cases} \tag{A.6}$$

Therefore, we have

$$\begin{aligned} \psi^i_t(x_t) &= \sum_{j \in \mathcal{J}} p_{jt} \max\{F^i_j + \psi^i_{t-1}(x_t + e_j) - \psi^i_{t-1}(x_t), 0\} + \psi^i_{t-1}(x_t) \\ &= \sum_{j \in \mathcal{J}^i} p_{jt} \max\{F^i_j + V^i_{t-1}(\mathcal{R}^i(x_t + e_j)) - V^i_{t-1}(\mathcal{R}^i(x_t)), 0\} \\ &\quad + \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} p_{jt} \max\{F^i_j + q_j \, [-\Theta^i_j]^+, 0\} + V^i_{t-1}(\mathcal{R}^i(x_t)) \\ &\quad + \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} q_j \, [-\Theta^i_j]^+ \, x_{jt} + \sum_{s=1}^{t-1} \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} p_{js} \max\{F^i_j, F^i_j - q_j \, \Theta^i_j, 0\}, \end{aligned}$$

where the first equality follows from (A.1) and the second equality follows from (A.6) and the induction assumption. Since $\max\{F^i_j + q_j \, [-\Theta^i_j]^+, 0\} = \max\{F^i_j, F^i_j - q_j \, \Theta^i_j, 0\}$, the result follows by collecting the terms on the right side of the expression above and noting the definition of $V^i_t(\mathcal{R}^i(x_t))$ in (2.27). □

Lemma A.2.2 *For all $t \in \mathcal{T}$, we have*

$$V_t(x_t) \leq \psi^i_t(x_t) - \sum_{j \in \mathcal{J}} q_j \, \Lambda^i_j \, x_{jt} + \sum_{l \in \mathcal{L} \setminus \{i\}} \lambda^*_l \, c_l.$$

Proof of Lemma A.2.2 We show the result by induction over the time periods. We let $\{w_j^* : j \in \mathcal{J}\}$ be the optimal solution to problem (2.1)-(2.4). We have

$$\begin{aligned}\Gamma(S(x_0)) &= \sum_{j\in\mathcal{J}} \theta_j\, w_j^* \\ &\geq \sum_{j\in\mathcal{J}} \theta_j\, w_j^* + \sum_{l\in\mathcal{L}\setminus\{i\}} \lambda_l^* \Big\{ \sum_{j\in\mathcal{J}} a_{lj}\, [S_j(x_{j0}) - w_j^*] - c_l \Big\} \\ &= \sum_{j\in\mathcal{J}} \Theta_j^i\, w_j^* + \sum_{j\in\mathcal{J}} \Lambda_j^i\, S_j(x_{j0}) - \sum_{l\in\mathcal{L}\setminus\{i\}} \lambda_l^*\, c_l \\ &\geq \phi^i(S(x_0)) + \sum_{j\in\mathcal{J}} \Lambda_j^i\, S_j(x_{j0}) - \sum_{l\in\mathcal{L}\setminus\{i\}} \lambda_l^*\, c_l, \qquad \text{(A.7)}\end{aligned}$$

where the first inequality follows from the fact that the solution $\{w_j^* : j \in \mathcal{J}\}$ satisfies constraints (2.2) and $\lambda_l^* \geq 0$ for all $l \in \mathcal{L} \setminus \{i\}$, the second equality follows from (2.13) and the second inequality follows from the fact that $\{w_j^* : j \in \mathcal{J}\}$ is a feasible but not necessarily an optimal solution to problem (A.2)-(A.5). Taking expectations in the expression above, we obtain $V_0(x_0) = -\mathbb{E}\{\Gamma(S(x_0))\} \leq -\mathbb{E}\{\phi^i(S(x_0))\} - \sum_{j\in\mathcal{J}} q_j\, \Lambda_j^i\, x_{j0} + \sum_{l\in\mathcal{L}\setminus\{i\}} \lambda_l^*\, c_l = \psi_0^i(x_0) - \sum_{j\in\mathcal{J}} q_j\, \Lambda_j^i\, x_{j0} + \sum_{l\in\mathcal{L}\setminus\{i\}} \lambda_l^*\, c_l$ and the result holds for the last time period. Assuming that the result holds for time period $t-1$, the induction assumption immediately implies that

$$\begin{aligned}\max\{f_j + V_{t-1}(x_t + e_j), V_{t-1}(x_t)\} \leq \max\{f_j + \psi_{t-1}^i(x_t + e_j) - q_j\, \Lambda_j^i, \psi_{t-1}^i(x_t)\} \\ - \sum_{j\in\mathcal{J}} q_j\, \Lambda_j^i\, x_{jt} + \sum_{l\in\mathcal{L}\setminus\{i\}} \lambda_l^*\, c_l.\end{aligned}$$

Recalling that $F_j^i = f_j - q_j\, \Lambda_j^i$, one can combine the inequality above with (2.5) and (A.1) to obtain the result for time period t. □

We are now ready to finalize the proof of Proposition 2.5.1. Lemmas A.2.1 and Lemma A.2.2 imply that

$$V_t(x_t) \leq V_t^i(\mathcal{R}^i(x_t)) + \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} q_j \, [-\Theta_j^i]^+ \, x_{jt} - \sum_{j \in \mathcal{J}} q_j \, \Lambda_j^i \, x_{jt} + \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \sum_{s=1}^{t} p_{js} \max\{F_j^i, F_j^i - q_j \, \Theta_j^i, 0\} + \sum_{l \in \mathcal{L} \setminus \{i\}} \lambda_l^* \, c_l.$$

The result follows by noting that the sum of the second and third terms on the right side of the expression above can be written as

$$\begin{aligned}
&\sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} q_j \, [-\Theta_j^i]^+ \, x_{jt} - \sum_{j \in \mathcal{J}} q_j \, \Lambda_j^i \, x_{jt} \\
&\quad = \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} q_j \, \max\{-\theta_j + \Lambda_j^i, 0\} \, x_{jt} - \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} q_j \, \Lambda_j^i \, x_{jt} - \sum_{j \in \mathcal{J}^i} q_j \, \Lambda_j^i \, x_{jt} \\
&\quad = - \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} q_j \, \min\{\theta_j, \Lambda_j^i\} \, x_{jt} - \sum_{j \in \mathcal{J}^i} q_j \, \Lambda_j^i \, x_{jt} \\
&\quad = - \sum_{j \in \mathcal{J} \setminus \mathcal{J}^i} \min\Big\{q_j \, \theta_j, q_j \sum_{l \in \mathcal{L}} a_{lj} \, \lambda_l^*\Big\} \, x_{jt} - \sum_{j \in \mathcal{J}^i} q_j \, \Lambda_j^i \, x_{jt},
\end{aligned}$$

where the last equality follows from the fact that $\Lambda_j^i = \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj} \, \lambda_l^*$ and $a_{ij} = 0$ whenever $j \in \mathcal{J} \setminus \mathcal{J}^i$.

A.3 Description of the Data Files

The data files that we use in our computational experiments are provided as an online supplement. The goal of this section is to describe the format of the data files. All of our data files are labeled as `rm_A_B_C.C_D.D_E.E_F.F.txt`, where `A` corresponds to the number of spokes in the airline network, `B` corresponds to the fare difference between a high fare and its corresponding low fare itinerary, `C.C` and `D.D` correspond to the parameters that we use to compute the penalty cost,

E.E corresponds to the probability that a reservation shows up at the departure time and F.F corresponds to the ratio of the total expected demand to the total expected capacity. In other words, following the notation in Section 5.1, A, B, (C.C, D.D), EE and F.F respectively correspond to N, κ, (δ, σ), q and ρ.

In all of our data sets, we assume that we serve N spokes out of a single hub. Location 0 corresponds to the hub and locations $\{1, \ldots, N\}$ correspond to the spokes. The itineraries that connect the hub to a spoke or a spoke to the hub include one flight leg. The itineraries that connect two spokes include two flight legs, one from the origin spoke to the hub and one from the hub to the destination spoke.

Table A.5 shows the organization of the data file for a test problem with $\tau = 3$ and $N = 2$. The character "#" indicates a comment line and such lines are skipped. The entries in the five portions of the data file have the following interpretations. The first portion of the data file shows the number of time periods in the planning horizon. The second portion of the data file shows the flight legs in the airline network. The first line in this portion shows the number of flight legs. After this first line, each line corresponds to one flight leg and shows the origin location, destination location and capacity of the flight leg. The third portion of the data file shows the itineraries. The first line in this portion shows the number of itineraries. After this first line, each line corresponds to one itinerary and shows the origin location, destination location, fare level, revenue and penalty cost for the itinerary. Fare level 0 indicates a low fare itinerary and fare level 1 indicates a high fare itinerary. We emphasize that the itineraries that connect two spokes include two flight legs, one from the origin spoke to the hub and one from the hub to the destination spoke. The fourth portion of the data file shows the arrival probabilities

for the requests for different itineraries.

Table A.5: Organization of the data file for a test problem with $\tau = 3$ and $N = 2$.

```
# beginning of data file
# portion 1
# number of time periods indecision horizon 3

# portion 2
# list of flights [in format origin location, destination location, capacity]
# first line is number of flights
4
1 0 16
2 0 21
0 1 12
0 2 20

# portion 3
# list of itineraries [in format origin location, destination location,
# fare level, revenue, penalty cost]
# first line is number of itineraries
7
0 1 0 24.0 48.0
0 1 1 192.0 384.0
0 2 0 34.0 68.0
1 0 0 192.0 384.0
1 2 0 53.0 106.0
2 1 0 53.0 106.0
2 1 1 212.0 442.0

# portion 4
# list of request arrival probabilities [in format itinerary, probability]
# first entry in each line indicates time period
0  [0 1 0] 0.1    [0 1 1] 0.1    [0 2 0] 0.1    [1 0 0] 0.1    [1 2 0] 0.1
1  [0 1 0] 0.1    [0 1 1] 0.1    [0 2 0] 0.1    [1 0 0] 0.1    [1 2 0] 0.1
2  [0 1 0] 0.1    [0 1 1] 0.1    [0 2 0] 0.1    [1 0 0] 0.1    [1 2 0] 0.1

# portion 5
# list of show up probabilities [in format itinerary, probability]
[0 1 0] 0.9
[0 1 1] 0.9
[0 2 0] 0.9
[1 0 0] 0.9
[1 2 0] 0.9
[2 1 0] 0.9
[2 1 1] 0.9

# end of data file
```

Each line in this portion corresponds to a time period in the planning horizon. Each line first shows an itinerary indicated by the triplet [origin location, destination location, fare level], followed by the probability that we observe a request for this itinerary. For example, the probability that we observe a request for the low fare itinerary from location 2 to 1 at the first time period is 0.2. Since we may not observe any itinerary arrivals at a particular time period, the probabilities in a particular line do not necessarily add up to one. The fifth portion of the data file shows the show up probabilities. Each line in this portion corresponds to one itinerary. Each line first shows an itinerary indicated by the triplet [origin location, destination location, fare level], followed by the probability that a reservation for this itinerary shows up at the departure time.

Chapter 3

Using Decomposition Methods to Solve Pricing Problems in Network Revenue Management

3.1 Introduction

Capacity allocation has traditionally been regarded as the prevalent control policy for the network revenue management systems operated by the airlines. In particular, a capacity allocation policy fixes the prices for the itineraries at prespecified levels and decides which itineraries to close and which itineraries to keep open for sale so as to maximize the total expected revenue. It has been argued that the airlines are suitable for capacity allocation since their promotion and administrative needs require them to fix the prices for the itineraries in advance of the sales and a capacity allocation policy indeed allows them to work with fixed prices. However, this argument has started to lose its validity with the advent of online sales channels allowing the airlines to dynamically adjust the prices for the itineraries as the sales take place. As a result, pricing has started to emerge as a feasible control policy for the network revenue management systems operated by the airlines.

One of the traditional approaches for making pricing decisions in network revenue management problems is based on a deterministic linear program. This deterministic linear program assumes that the arrivals of the itinerary requests are given by deterministic functions of the prices. The decision variables correspond to the numbers of time periods in the planning horizon for which we charge the different price levels for the itineraries. The deterministic linear program dates back to the

work of Gallego and van Ryzin (1997) and it has received a lot of attention from academics and practitioners over the years, but due to its deterministic nature, it is not able to capture the temporal dynamics of the itinerary requests accurately.

In this paper, we propose two new methods suitable for making pricing decisions in network revenue management problems. In the setting we consider, the probability of observing a request for an itinerary depends on the prices that we charge for the itineraries and the objective is to dynamically adjust the prices so as to maximize the total expected revenue. Both of the methods that we propose use the deterministic linear program mentioned above as a starting point. In particular, by using the dual solution to the deterministic linear program, we first allocate the immediate revenue associated with a certain price level among the different flight legs. Once we have allocated the immediate revenue associated with a certain price level among the different flight legs, we can solve a sequence of revenue management problems, each taking place over a single flight leg. In the single leg revenue management problem that takes place over a particular flight leg, if we charge a certain price level for a certain itinerary, then the revenue that we obtain is given by the portion of the price level that is allocated to the flight leg. By solving the dynamic programming formulation of the single leg revenue management problem for each flight leg, we obtain a value function from each one of the flight legs, in which case, we sum up these value functions to obtain a value function approximation for the original network revenue management problem. Ultimately, both of the methods that we propose construct separable approximations to the value functions.

The methods that we propose in this paper provide advantages when compared with the deterministic linear program. To begin with, since our methods use dy-

namic programming formulations for the single leg revenue management problems, they are likely to capture the temporal dynamics of the itinerary requests more accurately than the deterministic linear program. In addition, it is possible to show that the deterministic linear program provides an upper bound on the total expected revenue obtained by the optimal control policy. Such upper bounds become useful when we assess the optimality gap of a suboptimal control policy. We show that our methods also obtain upper bounds on the optimal total expected revenue and the upper bounds obtained by our methods are provably tighter than those from the deterministic linear program. Finally, our computational experiments demonstrate that the two methods that we propose can provide substantial improvements over the deterministic linear program. Averaging over all of the test problems in our experimental setup, we observe that the gap between the total expected revenues obtained by our methods and the deterministic linear program is 7.11%, whereas the gap between the upper bounds obtained by our methods and the deterministic linear program is 3.66%.

The basic idea behind the two methods that we propose is to decompose the pricing problem over an airline network by the flight legs and to obtain value function approximations by solving a sequence of single leg revenue management problems. Therefore, our methods can be visualized as dynamic programming decomposition approaches. There are some other dynamic programming decomposition approaches in the literature that try to construct good control policies by focusing on one flight leg at a time. However, these approaches exclusively use capacity control policies, whereas our focus is on pricing. To our knowledge, there are few practical algorithms for pricing and the transition from capacity control to pricing is nontrivial and practically important. Zhang and Adelman (2009) were the first to show that a dynamic programming decomposition approach can pro-

vide upper bounds on the optimal total expected revenue in the capacity allocation setting. Liu and van Ryzin (2008) use dynamic programming decomposition approaches to model customer choice behavior, where each customer observes the set of itineraries that are available for sale and makes a choice among them. Topaloglu (2009) demonstrates that it is possible to develop dynamic programming decomposition approaches by using a suitable Lagrangian relaxation argument on the dynamic programming formulation of a capacity allocation problem. Erdelyi and Topaloglu (2009*b*) follow a dynamic programming decomposition idea to develop a joint capacity allocation and overbooking model.

Although pricing is a fundamental control mechanism in network revenue management, most of the pricing papers in the literature focus on pricing a single product in isolation, whereas the network revenue management setting requires pricing multiple itineraries that interact with each other. Gallego and van Ryzin (1994) analyze the problem of dynamically adjusting the price of a single product and characterize the form of the optimal policy. They also show that a single price policy is asymptotically optimal as the initial inventory of the product and the length of the selling horizon increases linearly at the same rate. Feng and Gallego (2000), Feng and Xiao (2000) and Zhao and Zheng (2000) extend the analysis in Gallego and van Ryzin (1994) to incorporate more complicated demand dynamics and pricing constraints. Feng and Gallego (1995) consider the case where the price of a product can be adjusted only once, either from high to low or from low to high. They characterize the optimal timing of the price change. Maglaras and Meissner (2006) establish that certain pricing problems can be converted into equivalent capacity allocation problems and this result immediately allows them to extend the structural properties for capacity allocation problems to pricing problems.

The literature is thinner when we focus on pricing over an airline network. Gallego and van Ryzin (1997) propose a deterministic optimization problem for pricing multiple itineraries that interact with each other. They show that the pricing decisions made by this deterministic optimization problem are asymptotically optimal in the same sense as in Gallego and van Ryzin (1994). We use a variant of their approach as a benchmark strategy in our computational experiments. Kleywegt (2001*b*) develops a joint pricing and overbooking model, where the itinerary requests are deterministic functions of the prices and he solves the model by using Lagrangian duality arguments. Zhang and Cooper (2006) consider the problem of pricing substitutable flights that operate between the same origin destination pair. They build upper and lower bounds on the value functions and use these bounds to construct pricing policies, but their approach does not appear to extend to a general airline network. Kunnumkal and Topaloglu (2009*b*) propose a stochastic approximation algorithm for making pricing decisions over an airline network. The review papers by McGill and van Ryzin (1999), Bitran and Caldentey (2003) and Elmaghraby and Keskinocak (2003) and the book by Talluri and van Ryzin (2004) provide extensive coverage of pricing models in network revenue management.

In this paper, we make the following research contributions. 1) We develop two methods for making pricing decisions in network revenue management problems. Our methods are based on decomposing the pricing problem over an airline network by the flight legs and obtaining value function approximations by focusing on one flight leg at a time. Since our methods use dynamic programming formulations, they capture the temporal dynamics of the itinerary requests more accurately than the deterministic linear program mentioned above. 2) We show that both of our methods provide upper bounds on the total expected revenue obtained by the optimal control policy. It is possible to show that the deterministic linear program also

provides such an upper bound, but the upper bounds provided by our methods are provably tighter than those provided by the deterministic linear program. 3) Our computational experiments demonstrate that the methods that we propose can provide substantial improvements over the deterministic linear program. On average, the total expected revenues obtained by our methods improve those obtained by the deterministic linear program by 7.11% and there are test problems where the performance gap can be as high as 17.02%. Similarly, the average gap between the upper bounds obtained by our methods and the deterministic linear program is 3.66% and there are test problems where the gap between the upper bounds is as high as 7.98%. Furthermore, our computational experiments indicate that the two methods that we propose complement each other as they provide improvements over the deterministic linear program. In particular, one of the methods is successful in obtaining tight upper bounds on the optimal total expected revenue, whereas the other method is successful in identifying pricing policies that yield high total expected revenues.

The rest of the paper is organized as follows. In Section 3.2, we formulate the pricing problem over an airline network as a dynamic program. Then, in Section 3.3, we describe the deterministic linear program, show that it provides an upper bound on the optimal total expected revenue and demonstrate how it can be used to construct a pricing policy. Sections 3.4 and 3.5 propose two new methods for making pricing decisions. In each one of these sections, we focus on one of the two methods and show that the method in question provides an upper bound on the optimal total expected revenue and this upper bound is tighter than the one provided by the deterministic linear program. Furthermore, we demonstrate how our methods can be used to construct pricing policies. Next, Section 3.6 provides computational experiments. Finally, in Section 3.7, we conclude.

3.2 Problem Formulation

We have a set of flight legs that can be used to serve the requests for itineraries that arrive randomly over time. At each time period, we adjust the prices for the itineraries, which, in turn, determine the probability of observing a request for an itinerary. Pricing serves as the only control mechanism and we do not have the option of rejecting an itinerary request. Whenever there is an itinerary request, we accept the itinerary request, generate a revenue that reflects the price for the itinerary and consume the capacities on the relevant flight legs. The objective is to adjust the prices for the itineraries over time so as to maximize the total expected revenue.

The problem takes place over the finite planning horizon $\mathcal{T} = \{1, \ldots, \tau\}$ and time period $\tau+1$ is the departure time of the flight legs. A time period corresponds to a small enough interval of time that there is at most one itinerary request at each time period. This is a standard modeling approach in the network revenue management literature. The set of flight legs in the airline network is $\mathcal{L}$ and the set of itineraries is $\mathcal{J}$. The total available capacity on flight leg i is c_i. If we serve a request for itinerary j, then we consume a_{ij} units of capacity on flight leg i. For notational brevity, we denote $A^j = \{a_{ij} : i \in \mathcal{L}\}$ the resources consumed by itinerary j. The set of possible prices for itinerary j is given by the finite set $\{p_j^k : k \in \mathcal{K}\}$ and the price that we charge for itinerary j has to take a value in this set. If we charge the price p_j^k for itinerary j, then we observe a request for itinerary j at a time period with probability λ_j^k. For notational brevity, we let $r_j^k = \lambda_j^k \, p_j^k$ so that r_j^k is the expected revenue that we generate at a time period from itinerary j when we charge the price p_j^k for this itinerary. Throughout the paper, we employ a few assumptions for the price and probability pairs $\{(p_j^k, \lambda_j^k) : k \in \mathcal{K}\}$. First,

we assume that $\sum_{j \in \mathcal{J}} \max_{k \in \mathcal{K}} \{\lambda_j^k\} \leq 1$ so that irrespective of the prices that we charge, there is at most one itinerary request at each time period. Second, we assume that there exists some $\phi \in \mathcal{K}$ such that $\lambda_j^\phi = 0$. In this case, if we do not have enough capacity to serve a request for itinerary j, then we can simply charge the price p_j^ϕ to ensure that we do not observe a request for itinerary j. Third, as implicitly evident from our notation, we assume that the probability of observing a request for itinerary j depends only on the price for itinerary j, but not on the prices for the other itineraries. This assumption is reasonable when the itineraries do not serve as substitutes of each other. Furthermore, it is relatively simple to relax this assumption and we point out possible relaxations throughout the paper.

We use x_{it} to denote the remaining capacity on flight leg i at the beginning of time period t so that $x_t = \{x_{it} : i \in \mathcal{L}\}$ gives the state of the remaining leg capacities. We use $u_t = \{u_{jt}^k : j \in \mathcal{J},\ k \in \mathcal{K}\}$ to capture the decisions at time period t, where $u_{jt}^k = 1$ if we charge the price p_j^k for itinerary j at time period t and $u_{jt}^k = 0$ otherwise. In this case, the set of feasible decisions at time period t is given by

$$\mathcal{U}(x_t) = \Big\{ u_t \in \{0,1\}^{|\mathcal{J}||\mathcal{K}|} : \sum_{k \in \mathcal{K}} a_{ij}\, \lambda_j^k\, u_{jt}^k \leq x_{it} \qquad \forall\, i \in \mathcal{L},\ j \in \mathcal{J} \tag{3.1}$$

$$\sum_{k \in \mathcal{K}} u_{jt}^k = 1 \qquad \forall j \in \mathcal{J} \Big\}. \tag{3.2}$$

Since $\lambda_j^\phi = 0$, constraints (3.1) ensure that if we do not have enough capacity to serve a request for itinerary j, then we charge the price p_j^ϕ for this itinerary. Constraints (3.2) ensure that each itinerary is offered at a single price at each time period. We use $V_t(x_t)$ to denote the maximum total expected revenue that can be obtained over the time periods $\{t, \ldots, \tau\}$ given that the state of the remaining leg capacities at the beginning of time period t is x_t. We can evaluate the value

functions $\{V_t(\cdot) : t \in \mathcal{T}\}$ by solving the optimality equation

$$V_t(x_t) = \max_{u_t \in \mathcal{U}(x_t)} \left\{ \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} u_{jt}^k \left\{ r_j^k + \lambda_j^k \, V_{t+1}(x_t - A^j) \right\} + \left[1 - \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} u_{jt}^k \, \lambda_j^k \right] V_{t+1}(x_t) \right\} \tag{3.3}$$

with the boundary condition that $V_{\tau+1}(\cdot) = 0$. If the state of the remaining leg capacities at the beginning of time period t is given by x_t, then we can find the optimal pricing decisions by solving the problem on the right side of the optimality equation in (3.3).

Unfortunately, the optimality equation in (3.3) involves a high dimensional state variable and the computation of $\{V_t(\cdot) : t \in \mathcal{T}\}$ easily gets intractable for practical problems. In the next section, we begin by formulating a linear programming approximation to the optimality equation in (3.3). This linear program later serves as a starting point for our solution methods.

3.3 Deterministic Linear Program

Under the assumption that the itinerary requests take on their expected values, it is possible to formulate a deterministic linear program to approximate the total expected revenue over the planning horizon. In particular, letting w_j^k be the number of time periods at which we charge the price p_j^k for itinerary j, we can solve

the problem

$$\max \quad \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} r_j^k\, w_j^k \tag{3.4}$$

$$\text{subject to} \quad \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} a_{ij}\,\lambda_j^k\, w_j^k \le c_i \qquad \forall\, i\in\mathcal{L} \tag{3.5}$$

$$\sum_{k\in\mathcal{K}} w_j^k = \tau \qquad \forall\, j\in\mathcal{J} \tag{3.6}$$

$$w_j^k \ge 0 \qquad \forall\, j\in\mathcal{J},\ k\in\mathcal{K} \tag{3.7}$$

to approximate the total expected revenue. In the problem above, the objective function accounts for the total expected revenue over the planning horizon. Constraints (3.5) ensure that our decisions do not violate the leg capacities. Constraints (3.6) ensure that the total number of time periods at which we charge the different prices is equal to the number of time periods in the planning horizon.

There are two uses of problem (3.4)-(3.7). First, the optimal objective value of problem (3.4)-(3.7) provides an upper bound on the total expected revenue obtained by the optimal control policy. Such an upper bound becomes useful when assessing the optimality gap of a suboptimal control policy. In particular, if we denote $c = \{c_i : i \in \mathcal{L}\}$ the vector of available capacities on the flight legs, $V_1(c)$ is the optimal total expected revenue over the planning horizon. In this case, if we let $\hat{z}_{LP}$ be the optimal objective value of problem (3.4)-(3.7), the next proposition shows that $\hat{z}_{LP}$ provides an upper bound on $V_1(c)$. The proofs of all of our results can be found in the Appendix B.

Proposition 3.3.1 *We have* $V_1(c) \le \hat{z}_{LP}$.

Gallego and van Ryzin (1997) show an analogue of Proposition 3.3.1. The appealing aspect of their result is that they assume that the prices can take values

over a continuum and the itinerary requests arrive in continuous time according a Poisson process. However, their result requires that the expected revenue from an itinerary is a concave function of the arrival probability, whereas Proposition 3.3.1 does not make an assumption for the relationship between r_j^k and λ_j^k.

A second use of problem (3.4)-(3.7) occurs when we try to make the pricing decisions. In particular, if we let $\{\hat{w}_j^k : j \in \mathcal{J},\ k \in \mathcal{K}\}$ be the optimal solution to problem (3.4)-(3.7), then one alternative for making the pricing decisions is to charge the price p_j^k for itinerary j with probability $\hat{w}_j^k/\tau$ at each time period. If we do not have enough capacity to serve a request for itinerary j, then we naturally charge the price p_j^ϕ for itinerary j. We refer to this decision rule as DLP-P, where DLP stands for deterministic linear program and P stands for primal. Another alternative for making the pricing decisions is to use the optimal dual solution to problem (3.4)-(3.7). In particular, if we let $\{\hat{\pi}_i : i \in \mathcal{L}\}$ be the optimal values of the dual variables associated with constraints (3.5) in problem (3.4)-(3.7), then we can use $\hat{\pi}_i$ to capture the opportunity cost of a seat on flight leg i. This allows us to approximate the value functions $\{V_t(\cdot) : t \in \mathcal{T}\}$ with linear functions $\{\tilde{V}_t(\cdot) : t \in \mathcal{T}\}$ of the form $\tilde{V}_t(x_t) = \sum_{i \in \mathcal{L}} \hat{\pi}_i\, x_{it}$. In this case, we can replace the value functions $\{V_t(\cdot) : t \in \mathcal{T}\}$ on the right side of problem (3.3) with the linear value function approximations $\{\tilde{V}_t(\cdot) : t \in \mathcal{T}\}$ and solve this problem to make the pricing decisions at time period t. We refer to this decision rule as DLP-D, where D stands for dual.

Closing this section, we briefly elaborate on how to extend problem (3.4)-(3.7) to cover the case where the probability of observing a request for itinerary j does not depend only on the price for itinerary j, but also on the prices for the other itineraries. To cover this case, we let $\{p^k : k \in \mathcal{K}\}$ be the set of possible joint

prices for the itineraries, where the vector $p^k = \{p^k_j : j \in \mathcal{J}\}$ includes the prices of all itineraries. If we charge the prices p^k for the itineraries, then we observe a request for itinerary j at a time period with probability λ^k_j and λ^k_j can depend on the whole vector of prices p^k. In this case, if we let w^k be the number of time periods at which we charge the prices p^k for the itineraries, then all we need to do is to replace the decision variables $\{w^k_j : j \in \mathcal{J}\}$ in problem (3.4)-(3.7) with a single decision variable w^k. In this case, the objective function of problem (3.4)-(3.7) becomes $\sum_{k \in \mathcal{K}} \sum_{j \in \mathcal{J}} r^k_j \, w^k$, where the term $\sum_{j \in \mathcal{J}} r^k_j$ can be interpreted as the expected revenue that we generate at a time period from all itineraries when we charge the prices p^k. The first set of constraints become $\sum_{k \in \mathcal{K}} \sum_{j \in \mathcal{J}} a_{ij} \, \lambda^k_j \, w^k \leq c_i$ for all $i \in \mathcal{L}$, where the term $\sum_{j \in \mathcal{J}} a_{ij} \, \lambda^k_j$ can be interpreted as the expected capacity consumption at a time period on flight leg i when we charge the prices p^k. The second set of constraints become $\sum_{k \in \mathcal{K}} w^k = \tau$. We note that the number of possible joint prices $\{p^k : k \in \mathcal{K}\}$ can be very large in a practical application, which implies that the number of decision variables in problem (3.4)-(3.7) can also be very large. However, the number of constraints in problem (3.4)-(3.7) is always manageable. Therefore, we can solve problem (3.4)-(3.7) in a tractable fashion by using column generation.

3.4 Decomposition by Revenue Allocation

A shortcoming of the DLP-P and DLP-D decision rules is that they are based on the assumption that the itinerary requests take on their expected values. In this section, we build on problem (3.4)-(3.7) to develop a decision rule that addresses the stochastic nature of the itinerary requests more accurately. We begin by augmenting the set of flight legs with a fictitious flight leg ψ so that the set

of flight legs becomes $\mathcal{L} \cup \{\psi\}$. We assume that none of the itineraries use the fictitious flight leg so that its capacity is irrelevant. In this case, using the decision variables $\{w_{ij}^k : i \in \mathcal{L} \cup \{\psi\},\ j \in \mathcal{J},\ k \in \mathcal{K}\}$ instead of the decision variables $\{w_j^k : j \in \mathcal{J},\ k \in \mathcal{K}\}$, we observe that problem (3.4)-(3.7) is equivalent to the problem

$$\max \quad \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} r_j^k \, w_{\psi j}^k \tag{3.8}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} a_{ij} \, \lambda_j^k \, w_{ij}^k \leq c_i \qquad \forall \, i \in \mathcal{L} \tag{3.9}$$

$$\sum_{k \in \mathcal{K}} w_{ij}^k = \tau \qquad \forall \, i \in \mathcal{L},\ j \in \mathcal{J} \tag{3.10}$$

$$w_{\psi j}^k - w_{ij}^k = 0 \qquad \forall \, i \in \mathcal{L},\ j \in \mathcal{J},\ k \in \mathcal{K} \tag{3.11}$$

$$w_{ij}^k \geq 0 \qquad \forall \, i \in \mathcal{L},\ j \in \mathcal{J},\ k \in \mathcal{K}. \tag{3.12}$$

To see the equivalence between problems (3.4)-(3.7) and (3.8)-(3.12), we note that we can use constraints (3.11) to replace all of the decision variables $\{w_{ij}^k : i \in \mathcal{L}\}$ in problem (3.8)-(3.12) with a single decision variable $w_{\psi j}^k$. In this case, we can drop constraints (3.11) from problem (3.8)-(3.12) and problems (3.4)-(3.7) and (3.8)-(3.12) become equivalent to each other. Therefore, recalling the notation in Section 3.3, we note that the optimal objective value of problem (3.8)-(3.12) is still $\hat{z}_{LP}$.

We let $\{\hat{\mu}_{ij}^k : i \in \mathcal{L},\ j \in \mathcal{J},\ k \in \mathcal{K}\}$ be the optimal values of the dual variables associated with constraints (3.11) in problem (3.8)-(3.12). If we dualize these constraints by associating the multipliers $\{\hat{\mu}_{ij}^k : i \in \mathcal{L},\ j \in \mathcal{J},\ k \in \mathcal{K}\}$ with them, then the objective function of problem (3.8)-(3.12) reads $\sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} [r_j^k - \sum_{i \in \mathcal{L}} \hat{\mu}_{ij}^k] \, w_{\psi j}^k + \sum_{i \in \mathcal{L}} \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} \hat{\mu}_{ij}^k \, w_{ij}^k$. By the constraints in the dual of problem (3.8)-(3.12) associated with the decision variables $\{w_{\psi j}^k : j \in \mathcal{J},\ k \in \mathcal{K}\}$, we have $\sum_{i \in \mathcal{L}} \hat{\mu}_{ij}^k = r_j^k$ for all $j \in \mathcal{J}$, $k \in \mathcal{K}$. Therefore, the term $[r_j^k - \sum_{i \in \mathcal{L}} \hat{\mu}_{ij}^k]$ in the last

expression is equal to zero and the optimal objective value of problem (3.8)-(3.12) is equal to the optimal objective value of the problem

$$\max \quad \sum_{i\in\mathcal{L}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \hat{\mu}_{ij}^k \, w_{ij}^k \tag{3.13}$$

$$\text{subject to} \quad (3.9), (3.10), (3.12). \tag{3.14}$$

The crucial observation here is that the objective function and all of constraints (3.9), (3.10) and (3.12) in problem (3.13)-(3.14) decompose by the flight legs. This implies that problem (3.13)-(3.14) decomposes into $|\mathcal{L}|$ subproblems and the subproblem corresponding to flight leg i has the form

$$\max \quad \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \hat{\mu}_{ij}^k \, w_{ij}^k \tag{3.15}$$

$$\text{subject to} \quad \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} a_{ij} \, \lambda_j^k \, w_{ij}^k \leq c_i \tag{3.16}$$

$$\sum_{k\in\mathcal{K}} w_{ij}^k = \tau \qquad \forall \, j \in \mathcal{J} \tag{3.17}$$

$$w_{ij}^k \geq 0 \qquad \forall \, j \in \mathcal{J}, \; k \in \mathcal{K}. \tag{3.18}$$

Summing up the discussion in the last two paragraphs, if we let $\hat{z}_{LP}^i$ be the optimal objective value of problem (3.15)-(3.18), then we have $\hat{z}_{LP} = \sum_{i\in\mathcal{L}} \hat{z}_{LP}^i$.

Comparing problem (3.15)-(3.18) with problem (3.4)-(3.7), we observe that problem (3.15)-(3.18) corresponds to the deterministic linear program for a revenue management problem that takes place over the single flight leg i. In this single leg revenue management problem, if we charge the price p_j^k for itinerary j, then the expected revenue that we generate at a time period from itinerary j is given by $\hat{\mu}_{ij}^k$. Therefore, we can visualize $\hat{\mu}_{ij}^k$ as the portion of the expected revenue r_j^k that is allocated to flight leg i. Since by Proposition 3.3.1, the optimal objective value of the deterministic linear program provides an upper bound on the optimal total

expected revenue, $\hat{z}^i_{LP}$ provides an upper bound on the optimal total expected revenue in the single leg revenue management problem that takes place over flight leg i.

On the other hand, we can compute the optimal total expected revenue in the single leg revenue management problem that takes place over flight leg i by solving the optimality equation

$$v^i_t(x_{it}) = \max_{u_t \in \mathcal{U}^i(x_{it})} \left\{ \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} u^k_{jt} \left\{ \hat{\mu}^k_{ij} + \lambda^k_j \, v^i_{t+1}(x_{it} - a_{ij}) \right\} + \left[1 - \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} u^k_{jt} \, \lambda^k_j \right] v^i_{t+1}(x_{it}) \right\} \tag{3.19}$$

with the boundary condition that $v^i_{\tau+1}(\cdot) = 0$. The optimality equation above is similar to the one in (3.3), but the state variable only keeps track of the remaining capacity on flight leg i. The superscript i in the value functions emphasizes that the optimality equation above computes the optimal total expected revenue for the single leg revenue management problem that takes place over flight leg i. The set of feasible decisions $\mathcal{U}^i(x_{it})$ is given by

$$\mathcal{U}^i(x_{it}) = \left\{ u_t \in \{0,1\}^{|\mathcal{J}||\mathcal{K}|} : \sum_{k \in \mathcal{K}} a_{ij} \, \lambda^k_j \, u^k_{jt} \leq x_{it} \text{ and } \sum_{k \in \mathcal{K}} u^k_{jt} = 1 \quad \forall j \in \mathcal{J} \right\}.$$

We note that the definition of $\mathcal{U}^i(x_{it})$ is similar to that of $\mathcal{U}(x_t)$ in (3.1)-(3.2), but $\mathcal{U}^i(x_{it})$ only imposes the capacity availability on flight leg i.

The optimal total expected revenue in the single leg revenue management problem that takes place over flight leg i is given by $v^i_1(c_i)$. Furthermore, by the discussion above, $\hat{z}^i_{LP}$ provides an upper bound on the optimal total expected revenue in this single leg revenue management problem. This implies that $v^i_1(c_i) \leq \hat{z}^i_{LP}$. If we add over all $i \in \mathcal{L}$ and recall that we have $\sum_{i \in \mathcal{L}} \hat{z}^i_{LP} = \hat{z}_{LP}$, then we obtain $\sum_{i \in \mathcal{L}} v^i_1(c_i) \leq \hat{z}_{LP}$. On the other hand, the next proposition shows that

$V_1(c) \leq \sum_{i \in \mathcal{L}} v_1^i(c_i)$ and we obtain $V_1(c) \leq \sum_{i \in \mathcal{L}} v_1^i(c_i) \leq \hat{z}_{LP}$. Therefore, we can solve the optimality equation in (3.19) to obtain an upper bound on the optimal total expected revenue and this upper bound is tighter than the one provided by the optimal objective value of problem (3.4)-(3.7). Solving the optimality equation in (3.19) is tractable since this optimality equation involves a one-dimensional state variable.

Proposition 3.4.1 *We have* $V_t(x_t) \leq \sum_{i \in \mathcal{L}} v_t^i(x_{it})$ *for all* $t \in \mathcal{T}$.

In addition to bounding the optimal total expected revenue, we can use the optimality equation in (3.19) to make the pricing decisions. In particular, we can approximate the value functions $\{V_t(\cdot) : t \in \mathcal{T}\}$ with separable upper bounds $\{\tilde{V}_t(\cdot) : t \in \mathcal{T}\}$ of the form $\tilde{V}_t(x_t) = \sum_{i \in \mathcal{L}} v_t^i(x_{it})$. In this case, we can replace the value functions $\{V_t(\cdot) : t \in \mathcal{T}\}$ on the right side of problem (3.3) with the separable value function approximations $\{\tilde{V}_t(\cdot) : t \in \mathcal{T}\}$ and solve this problem to make the pricing decisions at time period t. We refer to this decision rule as DRA, standing for decomposition by revenue allocation. Our choice of terminology is motivated by the fact that $\{\hat{\mu}_{ij}^k : i \in \mathcal{L}\}$ serve as the portions of the expected revenue r_j^k that are allocated to the different flight legs.

3.5 Decomposition by Leg Relaxation

In this section, we describe a second decision rule that also addresses the stochastic nature of the itinerary requests. Similar to the DRA decision rule in the previous section, the starting point for this decision rule is a duality argument on problem (3.4)-(3.7), but the specifics of the duality argument are different. We let $\{\hat{\pi}_i : i \in$

$\mathcal{L}\}$ be the optimal values of the dual variables associated with constraints (3.5) in problem (3.4)-(3.7). We pick an arbitrary flight leg i and relax constraints (3.5) for all other flight legs by associating the dual multipliers $\{\hat{\pi}_l : l \in \mathcal{L} \setminus \{i\}\}$. In this case, by linear programming duality, problem (3.4)-(3.7) has the same optimal objective value as the problem

$$\max \quad \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} \left[r_j^k - \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj} \, \lambda_j^k \, \hat{\pi}_l \right] w_j^k + \sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l \, c_l \tag{3.20}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} a_{ij} \, \lambda_j^k \, w_j^k \leq c_i \tag{3.21}$$

$$(3.6), (3.7). \tag{3.22}$$

We recall that we use $\hat{z}_{LP}$ to denote this common optimal objective value. Ignoring the constant term $\sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l \, c_l$ in the objective function above and comparing problem (3.20)-(3.22) with problem (3.4)-(3.7), we observe that problem (3.20)-(3.22) corresponds to the deterministic linear program for a revenue management problem that takes place over the single flight leg i. In this single leg revenue management problem, if we charge the price level k for itinerary j, then the expected revenue that we generate at a time period from itinerary j is given by $r_j^k - \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj} \, \lambda_j^k \, \hat{\pi}_l$. Since by Proposition 3.3.1, the optimal objective value of the deterministic linear program provides an upper bound on the optimal total expected revenue, $\hat{z}_{LP} - \sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l \, c_l$ provides an upper bound on the optimal total expected revenue in the single leg revenue management problem that takes place over flight leg i.

We can use an optimality equation similar to the one in (3.19) to compute the optimal total expected revenue in the single leg revenue management problem that

takes place over flight leg i. In particular, this optimality equation is given by

$$\vartheta_t^i(x_{it}) = \max_{u_t \in \mathcal{U}^i(x_{it})} \left\{ \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} u_{jt}^k \left\{ r_j^k - \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj}\, \lambda_j^k\, \hat{\pi}_l + \lambda_j^k\, \vartheta_{t+1}^i(x_{it} - a_{ij}) \right\} + \left[1 - \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} u_{jt}^k\, \lambda_j^k \right] \vartheta_{t+1}^i(x_{it}) \right\} \quad (3.23)$$

with the boundary condition that $\vartheta_{\tau+1}^i(\cdot) = 0$. The optimal total expected revenue in the single leg revenue management problem that takes place over flight leg i is $\vartheta_1^i(c_i)$ and by the discussion in the previous paragraph, we have $\vartheta_1^i(c_i) \leq \hat{z}_{LP} - \sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l\, c_l$. On the other hand, the next proposition shows that $V_1(c) \leq \vartheta_1^i(c_i) + \sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l\, c_l$ and we obtain $V_1(c) \leq \vartheta_1^i(c_i) + \sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l\, c_l \leq \hat{z}_{LP}$. Therefore, we can solve the optimality equation in (3.23) to obtain an upper bound on the optimal total expected revenue and this upper bound is tighter than the one provided by the optimal objective value of problem (3.4)-(3.7). Furthermore, since the choice of flight leg i is completely arbitrary, the last chain of inequalities hold for all $i \in \mathcal{L}$, in which case, we can take the minimum over all $i \in \mathcal{L}$ and use

$$\min_{i \in \mathcal{L}} \left\{ \vartheta_1^i(c_i) + \sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l\, c_l \right\}$$

as the tightest possible upper bound on the optimal total expected revenue and this upper bound is also tighter than the one provided by $\hat{z}_{LP}$.

Proposition 3.5.1 *We have* $V_t(x_t) \leq \vartheta_t^i(x_{it}) + \sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l\, x_{lt}$ *for all* $i \in \mathcal{L}$, $t \in \mathcal{T}$.

Each of $\{\vartheta_t^i(x_{it}) + \sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l\, x_{lt} : i \in \mathcal{L}\}$ provides an upper bound on $V_t(x_t)$, but it is not clear which one of these upper bounds to use as a value function approximation when making the pricing decisions. A natural approach is to average over all $i \in \mathcal{L}$ and make the pricing decisions by using

$$\tilde{V}_t(x_t) = \frac{1}{|\mathcal{L}|} \sum_{i \in \mathcal{L}} \left\{ \vartheta_t^i(x_{it}) + \sum_{l \in \mathcal{L} \setminus \{i\}} \hat{\pi}_l\, x_{lt} \right\} \quad (3.24)$$

as an approximation to $V_t(x_t)$. Unfortunately, this idea does not perform too much better than the DLP-D decision rule. To see the reason, we first observe that if we make the pricing decisions by replacing the value functions $\{V_t(\cdot) : t \in \mathcal{T}\}$ in problem (3.3) by any value function approximations $\{\tilde{V}_t(\cdot) : t \in \mathcal{T}\}$, then we need to solve a problem of the form

$$\max_{u_t \in \mathcal{U}(x_t)} \left\{ \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} u_{jt}^k \left\{ r_j^k + \lambda_j^k \, \tilde{V}_{t+1}(x_t - A^j) \right\} + \left[1 - \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} u_{jt}^k \, \lambda_j^k \right] \tilde{V}_{t+1}(x_t) \right\}$$
$$= \max_{u_t \in \mathcal{U}(x_t)} \left\{ \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} u_{jt}^k \left\{ r_j^k + \lambda_j^k \, \tilde{V}_{t+1}(x_t - A^j) - \lambda_j^k \, \tilde{V}_{t+1}(x_t) \right\} \right\} + \tilde{V}_{t+1}(x_t). \tag{3.25}$$

Focusing on the second problem above, since the last term $\tilde{V}_{t+1}(x_t)$ is independent of the pricing decisions at time period t, we observe that the term that really affects the quality of the pricing decisions is the difference $\tilde{V}_{t+1}(x_t) - \tilde{V}_{t+1}(x_t - A^j)$.

We proceed to compare the form of the difference $\tilde{V}_{t+1}(x_t) - \tilde{V}_{t+1}(x_t - A^j)$ for the value function approximations used by the DLP-D decision rule and for the value function approximations given in (3.24). As described in Section 3.3, the value function approximations used by the DLP-D decision rule is of the form $\tilde{V}_t(x_t) = \sum_{i \in \mathcal{L}} \hat{\pi}_i \, x_{it}$ for all $t \in \mathcal{T}$. Therefore, for the value function approximations used by the DLP-D decision rule, we have $\tilde{V}_{t+1}(x_t) - \tilde{V}_{t+1}(x_t - A^j) = \sum_{i \in \mathcal{L}} a_{ij} \, \hat{\pi}_i$. On the other hand, for the value function approximations given in (3.24), we have

$$\tilde{V}_{t+1}(x_t) - \tilde{V}_{t+1}(x_t - A^j) = \frac{1}{|\mathcal{L}|} \sum_{i \in \mathcal{L}} \left\{ \vartheta_{t+1}^i(x_{it}) - \vartheta_{t+1}^i(x_{it} - a_{ij}) + \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj} \, \hat{\pi}_l \right\}. \tag{3.26}$$

We let $\mathcal{L}^j$ be the set of flight legs that are used by itinerary j. If $i \notin \mathcal{L}^j$, then we have $a_{ij} = 0$ by definition so that the sum in the curly brackets above can succinctly be written as $\sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj} \, \hat{\pi}_l = \sum_{l \in \mathcal{L}} a_{lj} \, \hat{\pi}_l$ whenever $i \notin \mathcal{L}^j$. Furthermore, we have

$\vartheta^i_{t+1}(x_{it}) - \vartheta^i_{t+1}(x_{it} - a_{ij}) = 0$ for all $i \notin \mathcal{L}^j$. In this case, we can write (3.26) as

$$\tilde{V}_{t+1}(x_t) - \tilde{V}_{t+1}(x_t - A^j) = \frac{1}{|\mathcal{L}|}\left\{ \sum_{i \in \mathcal{L}^j} \left\{ \vartheta^i_{t+1}(x_{it}) - \vartheta^i_{t+1}(x_{it} - a_{ij}) + \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj}\, \hat{\pi}_l \right\} + \sum_{i \in \mathcal{L} \setminus \mathcal{L}^j} \Big[\sum_{l \in \mathcal{L}} a_{lj}\, \hat{\pi}_l \Big] \right\}. \tag{3.27}$$

One way to visualize the expression on the right side above is that each flight leg contributes one term to the average. A flight leg $i \in \mathcal{L}^j$ contributes the term $\vartheta^i_{t+1}(x_{it}) - \vartheta^i_{t+1}(x_{it} - a_{ij}) + \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj}\, \hat{\pi}_l$, whereas a flight leg $i \notin \mathcal{L}^j$ contributes the term $\sum_{l \in \mathcal{L}} a_{lj}\, \hat{\pi}_l$. In general, the number of flight legs in the airline network that are not used by itinerary j is much larger than the number of flight legs that are used by itinerary j. This implies that we would expect the average in (3.27) to be dominated by the terms $\sum_{l \in \mathcal{L}} a_{lj}\, \hat{\pi}_l$ contributed by the flight legs that are not used by itinerary j, in which case, the average in (3.27) would be very close to $\sum_{l \in \mathcal{L}} a_{lj}\, \hat{\pi}_l$. Therefore, for the value function approximations used by the DLP-D decision rule and for the value function approximations given in (3.24), the differences $\tilde{V}_{t+1}(x_t) - \tilde{V}_{t+1}(x_t - A^j)$ are very similar to each other and using the value function approximations in (3.24) does not provide too much improvement over using the DLP-D decision rule.

To deal with this difficulty, instead of taking an average over all flight legs as in (3.27), we only take an average over the flight legs $i \in \mathcal{L}^j$. In particular, we replace the difference $\tilde{V}_{t+1}(x_t) - \tilde{V}_{t+1}(x_t - A^j)$ in problem (3.25) with

$$\frac{1}{|\mathcal{L}^j|} \sum_{i \in \mathcal{L}^j} \left\{ \vartheta^i_{t+1}(x_{it}) - \vartheta^i_{t+1}(x_{it} - a_{ij}) + \sum_{l \in \mathcal{L} \setminus \{i\}} a_{lj}\, \hat{\pi}_l \right\}$$

and solve this problem to make the pricing decisions at time period t. We refer to this decision rule as DLR, standing for decomposition by leg relaxation. Our choice of terminology is motivated by the fact that the DLR decision rule is obtained by

relaxing the capacity constraints in problem (3.4)-(3.7).

Both the DRA and DLR decision rules are obtained by building on problem (3.4)-(3.7). Therefore, it is possible to follow the discussion at the end of Section 3.3 so as to extend the DRA and DLR decision rules to handle the case where the probability of observing a request for itinerary j does not depend only on the price for itinerary j, but also on the prices for the other itineraries.

3.6 Computational Experiments

In this section, we numerically compare the upper bounds and total expected revenues obtained by the decision rules that we describe in Sections 3.3, 3.4 and 3.5.

3.6.1 Experimental Setup and Benchmark Strategies

In our computational experiments, we consider two types of functions that capture the relationship between the price and the probability of observing an itinerary request. In the first type of function, we assume that the probability of observing an itinerary request is a linear function of the price. In particular, we let $\Lambda_j(p) = \rho_j [1 - p/\kappa_j]$ so that if we charge the price p for itinerary j, then the probability of observing a request for itinerary j at a time period is given by $\Lambda_j(p)$. The parameter ρ_j can be interpreted as the probability of observing a request for itinerary j when we do not charge anything for this itinerary and the parameter κ_j can be interpreted as the price sensitivity. The price for itinerary j ranges over the interval $[0, \kappa_j]$ so that we have $\Lambda_j(p) \in [0, \rho_j]$ for all $p \in [0, \kappa_j]$. To work with a finite set of

price and probability pairs $\{(p_j^k, \lambda_j^k) : k \in \mathcal{K}\}$, we discretize the interval $[0, \rho_j]$ into 40 equal pieces and focus on the probabilities $\lambda_j^k = (k-1)\rho_j/40$ and the prices $p_j^k = \Lambda_j^{-1}(\lambda_j^k)$ for all $k = 1, \ldots, 41$, where $\Lambda_j^{-1}(\cdot)$ denotes the functional inverse of $\Lambda_j(\cdot)$. In the second type of function, we assume that the probability of observing a request for an itinerary is an exponential function of the price. In particular, we let $\Lambda_j(p) = \rho_j \, e^{-p/\kappa_j}$, where the interpretations for ρ_j and κ_j are the same as in the linear case. We assume that the price for itinerary j ranges over the interval $[0, \ln(10)\,\kappa_j]$ so that we have $\Lambda_j(p) \in [\rho_j/10, \rho_j]$ for all $p \in [0, \ln(10)\,\kappa_j]$. Similar to the linear case, we discretize the interval $[\rho_j/10, \rho_j]$ into 40 equal pieces and focus on the probabilities $\lambda_j^k = [\rho_j/10] + (k-1)\,9\,\rho_j/400$ and the prices $p_j^k = \Lambda_j^{-1}(\lambda_j^k)$ for all $k = 1, \ldots, 41$. In addition to these 41 price and probability pairs, to obtain some $\phi \in \mathcal{K}$ such that $\lambda_j^\phi = 0$, we assume that ∞ is an admissible price and if we charge this price, then the probability of observing an itinerary request is zero.

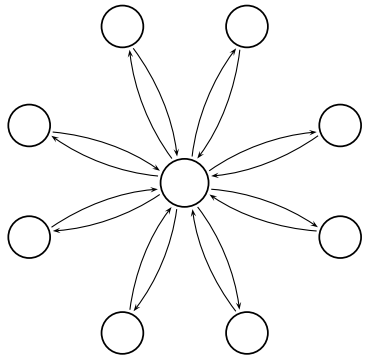

Figure 3.1: Airline network with eight spokes.

Our test problems are based on those in Kunnumkal and Topaloglu (2009*b*). We consider an airline network that serves N spokes from a single hub. There is one flight leg from each spoke to the hub and another flight leg from the hub to each spoke. Figure 3.1 shows the airline network with $N = 8$. There are two itineraries associated with every possible origin destination pair. One of these

itineraries is highly price sensitive and the other one is moderately price sensitive. Therefore, there are $2N$ flight legs and $2N(N+1)$ itineraries, $4N$ of which include one flight leg and $2N(N-1)$ of which include two flight legs. The price sensitivity associated with a highly price sensitive itinerary is κ times larger than the price sensitivity associated with the corresponding moderately price sensitive itinerary. To measure the tightness of the leg capacities, we let $\hat{k}^j = \text{argmax}_{k \in \mathcal{K}}\{r_j^k\}$ so that the price and probability pair $(p_j^{\hat{k}^j}, \lambda_j^{\hat{k}^j})$ maximizes the one period expected revenue from itinerary j. If we charge the prices $\{p_j^{\hat{k}^j} : j \in \mathcal{J}\}$ for the itineraries, then the total expected demand for the capacity on flight leg i is $\tau \sum_{j \in \mathcal{J}} a_{ij} \lambda_j^{\hat{k}^j}$ where τ is the number of time periods in the planning horizon. Then, we measure the tightness of the leg capacities by

$$\gamma = \frac{\tau \sum_{i \in \mathcal{L}} \sum_{j \in \mathcal{J}} a_{ij} \lambda_j^{\hat{k}^j}}{\sum_{i \in \mathcal{L}} c_i}.$$

We use (T, N, γ, κ) to label our test problems, where $T \in \{\text{L}, \text{E}\}$ denotes whether $\{\Lambda_j(\cdot) : j \in \mathcal{J}\}$ are linear or exponential functions and the remaining three components are as described above. We vary (T, N, γ, κ) over $\{\text{L}, \text{E}\} \times \{4, 8\} \times \{1.2, 1.6, 2.0\} \times \{2, 4, 8\}$ and this provides 36 test problems.

We use three benchmark strategies. Our first benchmark strategy corresponds to the DLP-P decision rule that we describe at the end of Section 3.3. Our practical implementation of this decision rule divides the planning horizon into S equal segments and resolves problem (3.4)-(3.7) at time periods $\{1 + (s-1)\tau/S : s = 1, \ldots, S\}$. In particular, at the beginning of segment s, we replace the right side of constraints (3.5) with the current remaining leg capacities $\{x_{i,1+(s-1)\tau/S} : i \in \mathcal{L}\}$ and the right side of constraints (3.6) with the current remaining number of time periods $\tau - (s-1)\tau/S$. We solve problem (3.4)-(3.7) and letting $\{\hat{w}_j^k : j \in \mathcal{J},\ k \in \mathcal{K}\}$ be an optimal solution to this problem, we charge the price p_j^k for itinerary j with probability $\hat{w}_j^k/[\tau - (s-1)\tau/S]$ until we reach the beginning of the next

segment. A few setup runs indicated that increasing S beyond 12 does not improve the performance of the DLP-P decision rule noticeably so that we use $S = 12$. The performances of the DLP-P and DLP-D decision rules turn out to be virtually identical in all of our test problems and we do not provide detailed results for the DLP-D decision rule. Our second benchmark strategy corresponds to the DRA decision rule that we describe at the end of Section 3.4, whereas our third benchmark strategy corresponds to the DLR decision rule that we describe at the end of Section 3.5. For the DRA and DLR decision rules, it is also possible to divide the planning horizon into equal segments and resolve problems (3.8)-(3.12) and (3.4)-(3.7) at the beginning of each segment to obtain new values for $\{\hat{\mu}_{ij}^k : i \in \mathcal{L},\ j \in \mathcal{J},\ k \in \mathcal{K}\}$ and $\{\hat{\pi}_i : i \in \mathcal{L}\}$, but it turns out that this extension does not provide any noticeable improvement for these decision rules.

3.6.2 Computational Results

Our main computational results are summarized in Tables 3.1 and 3.2. In particular, Table 3.1 and 3.2 respectively show the results for the test problems where $\{\Lambda_j(\cdot) : j \in \mathcal{J}\}$ are linear and exponential functions. The first column in these tables shows the problem characteristics. The second, third and fourth columns respectively show the upper bounds on the optimal total expected revenue obtained by DLP-P, DRA and DLR. The fifth and sixth columns show the percent gaps between the upper bounds obtained by DRA and the remaining two benchmark strategies. The upper bounds obtained by DRA are consistently the tightest and we use DRA as a reference when comparing the upper bounds. The seventh, eighth and ninth columns respectively show the total expected revenues obtained by DLP-P, DRA and DLR.

Table 3.1: Performances of DLP-P, DRA and DLR for the test problems where $\{\Lambda_j(\cdot) : j \in \mathcal{J}\}$ are linear functions.

Problem	Upper Bound by			% Gap with DRA		Total Exp. Rev. by			% Gap with DLR perf.		
(T, N, γ, κ)	DLP-P	DRA	DLR	DLP-P	DLR	DLP-P	DRA	DLR	DLP-P	DRA	DRA bnd.
(L, 4, 1.2, 2)	6,606	6,495	6,531	1.72	0.57	6,049	6,060	6,189	2.27	2.10	4.94
(L, 4, 1.2, 4)	9,538	9,388	9,435	1.59	0.50	8,654	8,729	9,017	4.03	3.20	4.11
(L, 4, 1.2, 8)	15,416	15,236	15,286	1.18	0.33	13,938	14,233	14,776	5.68	3.68	3.11
(L, 4, 1.6, 2)	5,922	5,760	5,835	2.80	1.30	5,304	5,324	5,396	1.70	1.35	6.75
(L, 4, 1.6, 4)	8,792	8,548	8,669	2.86	1.42	7,789	7,905	8,087	3.68	2.25	5.70
(L, 4, 1.6, 8)	14,640	14,306	14,474	2.33	1.18	12,963	13,354	13,766	5.83	2.99	3.92
(L, 4, 2.0, 2)	5,271	5,098	5,186	3.39	1.72	4,674	4,673	4,738	1.35	1.37	7.60
(L, 4, 2.0, 4)	8,084	7,803	7,956	3.61	1.97	7,067	7,171	7,337	3.69	2.27	6.35
(L, 4, 2.0, 8)	13,897	13,511	13,724	2.86	1.58	12,144	12,618	12,949	6.22	2.56	4.34
(L, 8, 1.2, 2)	6,273	6,108	6,209	2.70	1.65	5,454	5,501	5,655	3.57	2.72	8.01
(L, 8, 1.2, 4)	8,924	8,697	8,839	2.61	1.63	7,587	7,768	8,087	6.19	3.95	7.54
(L, 8, 1.2, 8)	14,239	13,961	14,132	1.99	1.23	11,951	12,429	13,118	8.90	5.25	6.43
(L, 8, 1.6, 2)	5,680	5,431	5,607	4.57	3.24	4,774	4,822	4,930	3.17	2.19	10.16
(L, 8, 1.6, 4)	8,281	7,894	8,179	4.90	3.60	6,800	6,975	7,226	5.90	3.47	9.24
(L, 8, 1.6, 8)	13,569	13,056	13,434	3.93	2.90	10,959	11,521	12,119	9.57	4.93	7.73
(L, 8, 2.0, 2)	5,152	4,897	5,079	5.22	3.73	4,261	4,309	4,396	3.08	1.98	11.40
(L, 8, 2.0, 4)	7,710	7,282	7,602	5.87	4.40	6,176	6,412	6,587	6.24	2.66	10.55
(L, 8, 2.0, 8)	12,972	12,368	12,825	4.88	3.70	10,241	10,917	11,367	9.91	3.96	8.81
Average				3.28	2.04				5.05	2.94	7.04

Table 3.2: Performances of DLP-P, DRA and DLR for the test problems where $\{\Lambda_j(\cdot) : j \in \mathcal{J}\}$ are exponential functions.

Problem	Upper Bound by			% Gap with DRA		Total Exp. Rev. by			% Gap with DLR perf.		
(T, N, γ, κ)	DLP-P	DRA	DLR	DLP-P	DLR	DLP-P	DRA	DLR	DLP-P	DRA	DRA bnd.
$(E, 4, 1.2, 2)$	4,271	4,205	4,225	1.58	0.48	3,899	3,994	4,073	4.26	1.95	3.24
$(E, 4, 1.2, 4)$	6,149	6,057	6,086	1.52	0.48	5,506	5,715	5,886	6.45	2.91	2.91
$(E, 4, 1.2, 8)$	9,909	9,795	9,826	1.15	0.31	8,742	9,251	9,567	8.62	3.30	2.38
$(E, 4, 1.6, 2)$	4,018	3,892	3,958	3.23	1.69	3,551	3,643	3,697	3.97	1.46	5.27
$(E, 4, 1.6, 4)$	5,866	5,691	5,782	3.08	1.61	5,067	5,322	5,488	7.67	3.01	3.70
$(E, 4, 1.6, 8)$	9,608	9,375	9,491	2.48	1.24	8,138	8,760	9,101	10.58	3.75	3.01
$(E, 4, 2.0, 2)$	3,723	3,556	3,654	4.69	2.75	3,209	3,297	3,354	4.34	1.72	6.02
$(E, 4, 2.0, 4)$	5,531	5,280	5,425	4.75	2.74	4,614	4,845	4,981	7.37	2.72	6.00
$(E, 4, 2.0, 8)$	9,242	8,885	9,085	4.01	2.25	7,476	8,016	8,331	10.26	3.78	6.65
$(E, 8, 1.2, 2)$	4,050	3,949	4,013	2.56	1.62	3,456	3,571	3,707	6.76	3.67	6.53
$(E, 8, 1.2, 4)$	5,748	5,601	5,699	2.62	1.75	4,751	5,038	5,282	10.05	4.62	6.04
$(E, 8, 1.2, 8)$	9,147	8,941	9,084	2.30	1.60	7,305	8,058	8,523	14.29	5.46	4.90
$(E, 8, 1.6, 2)$	3,827	3,643	3,777	5.06	3.68	3,070	3,213	3,299	6.93	2.60	10.43
$(E, 8, 1.6, 4)$	5,501	5,224	5,431	5.30	3.96	4,192	4,542	4,740	11.56	4.17	10.21
$(E, 8, 1.6, 8)$	8,884	8,498	8,790	4.54	3.43	6,466	7,361	7,792	17.02	5.53	9.06
$(E, 8, 2.0, 2)$	3,558	3,320	3,498	7.17	5.34	2,719	2,860	2,912	6.64	1.78	14.01
$(E, 8, 2.0, 4)$	5,194	4,810	5,104	7.98	6.11	3,686	4,026	4,181	11.85	3.71	15.04
$(E, 8, 2.0, 8)$	8,547	8,002	8,417	6.81	5.19	5,747	6,522	6,878	16.45	5.17	16.34
Average				3.94	2.57				9.17	3.41	7.32

We estimate these total expected revenues by simulating the performances of the different benchmark strategies under multiple sample paths. We use common random numbers when simulating the performances of the different benchmark strategies and all of the performance gaps are statistically significant at 95% level. The tenth and eleventh columns show the percent gaps between the total expected revenues obtained by DLR and the remaining two benchmark strategies. The total expected revenues obtained DLR are consistently the highest and we use DLR as a reference when comparing the total expected revenues. The twelfth column reports the percent gap between the most successful strategy, DLR, and the tightest bound obtained by DRA.

Comparing the upper bounds obtained by the three benchmark strategies, we observe that the upper bounds obtained by DRA are significantly tighter than those obtained by DLP-P and DLR. For the test problems in Table 3.1, the average gap between the upper bounds obtained by DRA and DLP-P is 3.28%, whereas the average gap between the upper bounds obtained by DRA and DLR is 2.04%. When we move to the test problems in Table 3.2, the upper bounds obtained by DRA improve those obtained by DLP-P and DLR by respectively 3.94% and 2.57% on average. There are test problems where the gap between the upper bounds obtained by DRA and DLP-P is as high as 7.98% and the gap between the upper bounds obtained by DRA and DLR is as high as 6.11%. In all of our test problems, the upper bounds obtained by DRA are uniformly tighter than those obtained by DLP-P and DLR.

Comparing the total expected revenues obtained by the three benchmark strategies, we observe that DLR obtains significantly higher total expected revenues than DLP-P and DRA. For the test problems in Table 3.1, the average gap between the

total expected revenues obtained by DLR and DLP-P is 5.05%, whereas the average gap between the total expected revenues obtained by DLP and DRA is 2.94%. When we move to the test problems in Table 3.2, the total expected revenues obtained by DLR improve those obtained by DLP-P and DRA by respectively 9.17% and 3.41% on average. There are test problems where the gap between the total expected revenues obtained by DLR and DLP-P is as high as 17.02% and the gap between the total expected revenues obtained by DLR and DRA is as high as 5.53%. In all of our test problems, the total expected revenues obtained by DLR are uniformly higher than those obtained by DLP-P and DRA. Although the total expected revenues obtained by DRA are not as high as those obtained by DLR, DRA also provides significant improvements over DLP-P.

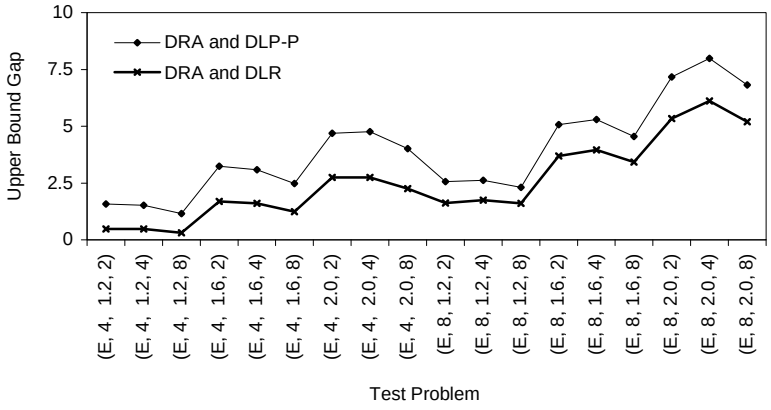

Figure 3.2: Percent gaps between the upper bounds obtained by DRA and the remaining two benchmark strategies for the test problems where $\{\Lambda_j(\cdot) : j \in \mathcal{J}\}$ are exponential functions.

DLP-P represents one of the traditional approaches for solving pricing problems and our results indicate that DRA and DLR complement each other as they provide improvements over DLP-P. In particular, DRA tightens the upper bounds and allows us to assess the optimality gaps more accurately, whereas DLR obtains higher total expected revenues. To give a feel for the problem parameters that

affect the relative performances of the three benchmark strategies, Figure 3.2 plots the gaps between the upper bounds obtained by DRA and the remaining two benchmark strategies and Figure 3.3 plots the gaps between the total expected revenues obtained by DLR and the remaining two benchmark strategies. The test problems in the horizontal axis in these figures are arranged in such a fashion that the first and last nine test problems respectively involve four and eight spokes. The leg capacities get tighter as we move from left to right within a block of

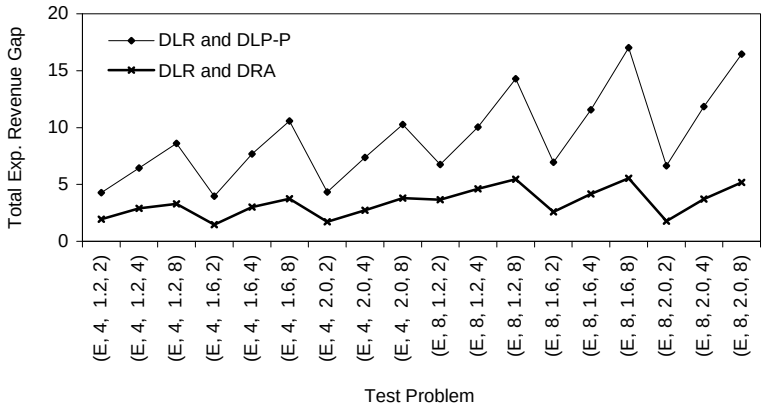

Figure 3.3: Percent gaps between the total expected revenues obtained by DLR and the remaining two benchmark strategies for the test problems where $\{\Lambda_j(\cdot) : j \in \mathcal{J}\}$ are exponential functions.

nine test problems. The difference between the price sensitivities of the highly and moderately price sensitive itineraries gets larger as we move from left to right within a block of three test problems. For economy of space, Figures 3.2 and 3.3 consider only the case where $\{\Lambda_j(\cdot) : j \in \mathcal{J}\}$ are exponential functions. Figure 3.2 indicates that the gaps between the upper bounds obtained by DRA and the remaining two benchmark strategies get larger as the leg capacities get tighter, whereas Figure 3.3 indicates that the gaps between the total expected revenues obtained by DLR and the remaining two benchmark strategies get larger as the difference in the price sensitivities gets larger. If we had infinite capacity on the flight legs, then

the different time periods in the planning horizon would not interact. In this case, letting $\hat{k}^j = \text{argmax}_{k \in \mathcal{K}}\{r_j^k\}$, it would be trivially optimal to charge the prices $\{p_j^{\hat{k}^j} : j \in \mathcal{J}\}$ for the itineraries. Therefore, we intuitively expect the test problems with tight leg capacities to be more difficult. On the other hand, if the difference in the price sensitivities of the highly and moderately price sensitive itineraries gets larger, then we use a richer set of prices to obtain good performance. Therefore, we also intuitively expect the test problems with larger differences in the price sensitivities to be more difficult. These observations indicate that DRA obtains especially tighter upper bounds and DLR obtains especially higher total expected revenues for the test problems that we intuitively expect to be more difficult.

Table 3.3: CPU seconds for DRA and DLR.

No. Time Pers. (τ)	CPU Seconds DRA	DLR
180	3.43	0.44
360	4.17	1.34
720	7.67	5.02
1,440	22.16	19.52

No. Spokes (N)	CPU Seconds DRA	DLR
4	0.68	0.58
6	1.81	1.06
8	4.17	1.34
10	9.05	1.74

Table 3.3 shows the CPU seconds for DRA and DLR with different numbers of time periods in the planning horizon and with different numbers of spokes in the airline network. All of our computational experiments are carried out on a Pentium IV PC running Windows XP with 2.4 Ghz CPU and 1 GB RAM. The CPU seconds for DRA correspond to the time required to solve problem (3.8)-(3.12) and the optimality equation in (3.19), whereas the CPU seconds for DLR correspond to the time required to solve problem (3.4)-(3.7) and the optimality equation in (3.23). The results indicate that the CPU seconds for DRA are noticeably longer than those for DLR. This discrepancy is due to the fact that DRA is based on problem (3.8)-(3.12) and this problem is significantly larger than problem (3.4)-

(3.7), which forms the starting point for DLR. Even for the largest test problems, the CPU seconds for both DRA and DLR are quite reasonable. The CPU seconds for DLP-P are on the order of a fraction of a second and we do not provide detailed CPU seconds for DLP-P. Overall, considering their improvements over DLP-P in terms of both upper bounds and total expected revenues, we conclude that DRA and DLR are strong candidates for solving practical pricing problems.

3.7 Conclusions

In this paper, we developed two methods for making pricing decisions in network revenue management problems. Both methods decompose the dynamic programming formulation of the problem by the flight legs and solve dynamic programs with one-dimensional state variables. We established that our methods obtain upper bounds on the optimal total expected revenue and these upper bounds are tighter than the one obtained by the deterministic linear program. Our computational experiments demonstrated significant improvements over the deterministic linear program and indicated that the two methods complement each other. In particular, the first method is useful in obtaining tight bounds on the optimal total expected revenue, whereas the pricing policy from the second method obtains higher total expected revenues.

APPENDIX

B Omitted proofs from Chapter 3

B.1 Proof of Proposition 3.3.1

We let $\{\hat{U}_{jt}^k : j \in \mathcal{J},\ k \in \mathcal{K},\ t \in \mathcal{T}\}$ be the pricing decisions under the optimal control policy, where $\hat{U}_{jt}^k = 1$ if we charge the price p_j^k for itinerary j at time period t and $\hat{U}_{jt}^k = 0$ otherwise. Similarly, we let $\hat{S}_{jt} = 1$ if we serve a request for itinerary j at time period t under the optimal control policy and $\hat{S}_{jt} = 0$ otherwise. We note that $\{\hat{U}_{jt}^k : j \in \mathcal{J},\ k \in \mathcal{K},\ t \in \mathcal{T}\}$ and $\{\hat{S}_{jt} : j \in \mathcal{J},\ t \in \mathcal{T}\}$ are random variables and we have $\sum_{k \in \mathcal{K}} \hat{U}_{jt}^k = 1$ for all $j \in \mathcal{J}$, $t \in \mathcal{T}$ by the feasibility of the pricing decisions. Furthermore, using $\hat{U}_{jt}$ to denote the vector $\{\hat{U}_{jt}^k : k \in \mathcal{K}\}$, we have $\mathbb{E}\{\hat{S}_{jt}\} = \mathbb{E}\{\mathbb{E}\{\hat{S}_{jt} \,|\, \hat{U}_{jt}\}\} = \sum_{k \in \mathcal{K}} \mathbb{E}\{\hat{S}_{jt} \,|\, \hat{U}_{jt}^k = 1\}\, \mathbb{P}\{\hat{U}_{jt}^k = 1\} = \sum_{k \in \mathcal{K}} \lambda_j^k\, \mathbb{E}\{\hat{U}_{jt}^k\}$, where the last equality follows from the fact that $\mathbb{E}\{\hat{S}_{jt} \,|\, \hat{U}_{jt}^k = 1\} = \lambda_j^k$ and $\mathbb{P}\{\hat{U}_{jt}^k = 1\} = \mathbb{E}\{\hat{U}_{jt}^k\}$ since $\hat{U}_{jt}^k$ is a Bernoulli random variable. Under the optimal control policy, the price that we charge for itinerary j at time period t is $\sum_{k \in \mathcal{K}} p_j^k\, \hat{U}_{jt}^k$. Thus, letting $\hat{\Pi}$ be the optimal total expected revenue, we have

$$\hat{\Pi} = \mathbb{E}\left\{ \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} \hat{S}_{jt} \left[\sum_{k \in \mathcal{K}} p_j^k\, \hat{U}_{jt}^k \right] \right\} = \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} p_j^k\, \mathbb{E}\{\hat{S}_{jt}\, \hat{U}_{jt}^k\}$$

$$= \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} p_j^k\, \mathbb{E}\{\hat{S}_{jt} \,|\, \hat{U}_{jt}^k = 1\}\, \mathbb{P}\{\hat{U}_{jt}^k = 1\} = \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} p_j^k\, \lambda_j^k \left[\sum_{t \in \mathcal{T}} \mathbb{E}\{\hat{U}_{jt}^k\} \right]. \quad \text{(B.1)}$$

On the other hand, since the itinerary requests that we serve satisfy the capacity constraints, we have $\sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} a_{ij}\, \hat{S}_{jt} \leq c_i$ for all $i \in \mathcal{L}$. Taking expectations in the last inequality and noting that $\mathbb{E}\{\hat{S}_{jt}\} = \sum_{k \in \mathcal{K}} \lambda_j^k\, \mathbb{E}\{\hat{U}_{jt}^k\}$, we obtain

$$\sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} a_{ij}\, \mathbb{E}\{\hat{S}_{jt}\} = \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} a_{ij}\, \lambda_j^k \left[\sum_{t \in \mathcal{T}} \mathbb{E}\{\hat{U}_{jt}^k\} \right] \leq c_i. \quad \text{(B.2)}$$

By the feasibility of the pricing decisions, we have $\sum_{k\in\mathcal{K}} \hat{U}_{jt}^k = 1$ for all $j \in \mathcal{J}$, $t \in \mathcal{T}$. If we take expectations and add over all time periods, then we obtain

$$\sum_{k\in\mathcal{K}} \Big[\sum_{t\in\mathcal{T}} \mathbb{E}\{\hat{U}_{jt}^k\} \Big] = \tau. \tag{B.3}$$

By (B.2) and (B.3), $\{\sum_{t\in\mathcal{T}} \mathbb{E}\{\hat{U}_{jt}^k\} : j \in \mathcal{J},\ k \in \mathcal{K}\}$ is a feasible solution to problem (3.4)-(3.7). Furthermore, the objective value provided by this feasible solution is $\sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}} r_j^k \left[\sum_{t\in\mathcal{T}} \mathbb{E}\{\hat{U}_{jt}^k\}\right] = \hat{\Pi}$, where the equality follows by (B.1). Thus, the optimal objective value of problem (3.4)-(3.7) is at least $\hat{\Pi}$. □

B.2 Proof of Proposition 3.4.1

We show the result by induction over the time periods. For time period $\tau+1$, we have $V_{\tau+1}(\cdot) = 0$ and $v_{\tau+1}^i(\cdot) = 0$ for all $i \in \mathcal{L}$ so that the result holds trivially for time period $\tau+1$. Assuming that the result holds for time period $t+1$ and letting $\hat{u}_t$ be the optimal solution to problem (3.3), we have

$$\begin{aligned}
V_t(x_t) &= \sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}} \hat{u}_{jt}^k \Big\{ r_j^k + \lambda_j^k\, V_{t+1}(x_t - A^j) \Big\} + \Big[1 - \sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}} \hat{u}_{jt}^k\, \lambda_j^k \Big]\, V_{t+1}(x_t) \\
&\leq \sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}} \hat{u}_{jt}^k \Big\{ r_j^k + \lambda_j^k \sum_{i\in\mathcal{L}} v_{t+1}^i(x_{it} - a_{ij}) \Big\} \\
&\qquad\qquad + \Big[1 - \sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}} \hat{u}_{jt}^k\, \lambda_j^k \Big] \sum_{i\in\mathcal{L}} v_{t+1}^i(x_{it}) \\
&= \sum_{i\in\mathcal{L}} \Bigg\{ \sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}} \hat{u}_{jt}^k \Big\{ \hat{\mu}_{ij}^k + \lambda_j^k\, v_{t+1}^i(x_{it} - a_{ij}) \Big\} \\
&\qquad\qquad + \Big[1 - \sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}} \hat{u}_{jt}^k\, \lambda_j^k \Big]\, v_{t+1}^i(x_{it}) \Bigg\} \\
&\leq \sum_{i\in\mathcal{L}} v_t^i(x_{it}),
\end{aligned}$$

where the first inequality follows from the induction assumption, the second equality follows from the fact that $r_j^k = \sum_{i\in\mathcal{L}} \hat{\mu}_{ij}^k$ for all $j \in \mathcal{J}$, $k \in \mathcal{K}$ and the last

inequality follows from the fact that $\hat{u}_t$ is a feasible but not necessarily an optimal solution to problem (3.19). □

B.3 Proof of Proposition 3.5.1

We use an induction argument that is similar to the proof of Proposition 3.4.1. For time period $\tau+1$, we have $V_{\tau+1}(\cdot)=0$, $\vartheta^i_{\tau+1}(\cdot)=0$ for all $i\in\mathcal{L}$ and $\hat{\pi}_l\geq 0$ for all $l\in\mathcal{L}$ by dual feasibility to problem (3.4)-(3.7). Therefore, the result holds for time period $\tau+1$. Assuming that the result holds for time period $t+1$ and letting $\hat{u}_t$ be the optimal solution to problem (3.3), we have

$$\begin{aligned}
V_t(x_t) &= \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \hat{u}^k_{jt}\left\{r^k_j+\lambda^k_j\, V_{t+1}(x_t-A^j)\right\}+\left[1-\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}\hat{u}^k_{jt}\,\lambda^k_j\right]V_{t+1}(x_t)\\
&\leq \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \hat{u}^k_{jt}\left\{r^k_j+\lambda^k_j\,\vartheta^i_{t+1}(x_{it}-a_{ij})+\lambda^k_j\sum_{l\in\mathcal{L}\setminus\{i\}}\hat{\pi}_l\,[x_{lt}-a_{lj}]\right\}\\
&\qquad\qquad +\left[1-\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}\hat{u}^k_{jt}\,\lambda^k_j\right]\left[\vartheta^i_{t+1}(x_{it})+\sum_{l\in\mathcal{L}\setminus\{i\}}\hat{\pi}_l\,x_{lt}\right]\\
&= \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \hat{u}^k_{jt}\left\{r^k_j-\sum_{l\in\mathcal{L}\setminus\{i\}}a_{lj}\,\lambda^k_j\,\hat{\pi}_l+\lambda^k_j\,\vartheta^i_{t+1}(x_{it}-a_{ij})\right\}\\
&\qquad\qquad +\left[1-\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}\hat{u}^k_{jt}\,\lambda^k_j\right]\vartheta^i_{t+1}(x_{it})+\sum_{l\in\mathcal{L}\setminus\{i\}}\hat{\pi}_l\,x_{lt}\\
&\leq \sum_{i\in\mathcal{L}}\vartheta^i_t(x_{it})+\sum_{l\in\mathcal{L}\setminus\{i\}}\hat{\pi}_l\,x_{lt},
\end{aligned}$$

where the first inequality follows from the induction assumption and the last inequality follows from the fact that $\hat{u}_t$ is a feasible but not necessarily an optimal solution to problem (3.23). □

Chapter 4

A Dynamic Programming Decomposition Method for Capacity Allocation Problems

4.1 Introduction

The problem of allocating limited capacity among competing jobs in a dynamic fashion occurs in many settings. Clinics allocate appointment slots to patients with different needs, production plants allocate production capacity to orders with different priorities, hotels allocate room inventory to customers with different willingness to pay amounts. Controlling such systems requires careful planning for several reasons. To begin with, the decisions for the current day are made with limited information about the future job arrivals. The problem is dynamic in the sense that the future capacity that is not committed to the jobs that arrive today can be committed to the jobs that arrive tomorrow. Finally, it is crucial to keep a balance between committing the capacity to a lower priority job that is available today and reserving the capacity for a potential higher priority job that may arrive tomorrow.

In this paper, we consider a capacity allocation problem that captures the tradeoffs described above. We have a fixed amount of daily processing capacity. Jobs of different priorities arrive randomly over time and we need to decide which jobs should be scheduled on which days. The jobs that are waiting to be processed incur holding costs depending on their priority levels. The goal is to minimize the total expected cost over a planning horizon. It is possible to formulate this problem

as a dynamic program, but this formulation quickly gets intractable for practical problem instances. To overcome this difficulty, we decompose the dynamic programming formulation of the capacity allocation problem by booking days. As a result, we obtain tractable approximations to the value functions which can be used to make capacity allocation decisions over time.

Our approach starts with a dynamic programming formulation of the capacity allocation problem. To obtain a lower bound on the total expected costs of the optimal policy, we formulate a deterministic linear problem based on the assumption that job arrivals take on their expected values and it is possible to schedule fractional portions of jobs for different days. Using a duality argument, we reformulate this linear problem so that it can be interpreted as a linear program corresponding to a capacity allocation problem where the daily capacity is restricted only for one day in the planning horizon. We refer to this problem as a single-day capacity allocation problem. Concentrating on the dynamic programming formulation of this single-day problem, we show that it gives rise to tractable approximations to the value functions corresponding to the original capacity allocation problem. We use these value functions approximations to propose a capacity allocation policy. Furthermore, we show that these value functions approximations provide lower bounds on the total expected costs of the optimal policy and that these bounds are tighter than the lower bound provided by the deterministic linear program discussed above.

Several papers study variations of our capacity allocation problem. The most related one is Patrick, Puterman and Queyranne (2008), where the authors consider the allocation of appointment slots in a clinic. The authors begin by formulating their capacity allocation problem as a dynamic program. Since this formulation

quickly gets intractable for practical problem instances, they develop an approximate dynamic programming method that uses linear approximations to the value functions. To choose the slope parameters of the linear approximations, they substitute the linear approximations into a linear program that characterizes the dynamic programming formulation of the capacity allocation problem. A second paper that is related to our work is Erdelyi and Topaloglu (2009*a*). Considering the same capacity allocation problem as here, the authors focus on a class of policies that are characterized by a set of protection levels and develop a stochastic approximation method to find a good set of protection levels. Protection level policies are also considered in Gerchak, Gupta and Henig (1996). The authors concentrate on the capacity allocation problem with two priority levels. The main contribution of their paper is to characterize the structure of the optimal policy and to show that protection level policies are not necessarily optimal.

We borrow the dynamic problem decomposition idea from the revenue management literature. Williamson (1992) is one of the first to decompose the network revenue management problem by flight legs. A discussion of early decomposition heuristics methods is given in Talluri and van Ryzin (2004). Liu and van Ryzin (2008) and Kunnumkal and Topaloglu (2009*a*) extend decomposition heuristics ideas to the network revenue management problem that incorporates customer choice behavior. Zhang and Adelman (2009) are the first to show that a dynamic programming decomposition is capable of providing upper bounds on the optimal total expected revenue in the network revenue management problem. Topaloglu (2009) demonstrates that decomposition methods can be visualized as an application of Lagrangian relaxation to the dynamic programming formulation of the network revenue management problem. Erdelyi and Topaloglu (2009*b*) and Erdelyi, Kunnumkal and Topaloglu (2009) extend the decomposition ideas to network

revenue management problems involving overbooking and pricing decisions, respectively. It is interesting to note that the capacity allocation problem that we consider in this paper can be viewed as a revenue management problem as it involves making the most use out of limited and perishable capacity. However, traditional revenue management problems involve only the decision of whether to accept or reject a job, whereas our capacity allocation problem involves the decision of when to schedule a job along with whether the job should be accepted or rejected.

We make the following research contributions in this paper. 1) We develop a method to make dynamic scheduling decisions in a capacity allocation model involving jobs of different priorities. The idea behind our model is to decompose the problem by booking days and solve a sequence of single-day capacity allocation problems. 2) We show that our approach provides a lower bound on the total expected cost obtained by the optimal policy and this bound is tighter than the one provided by the deterministic linear program formulation. 3) Computational experiments demonstrate that our method performs significantly better than standard benchmark stategies.

The rest of the paper is organized as follows. In Section 4.2, we describe the capacity allocation problem and give a precise formulation of the corresponding dynamic program. Then, Section 4.3 formulates a deterministic linear program that both gives rise to potential control policies as well as provides a lower bound on the optimal total expected cost. Section 4.4 decomposes the dynamic program into a sequence of single-day capacity allocation problems while Section 4.5 addresses practical issues with solving these single-day allocation problems. In Section 4.6, we further specify the implementation of the policy suggested by the decomposition

approach. Next, Section 4.7 presents computational experiments. Finally, Section 4.8 gives concluding remarks.

4.2 Problem Formulation

We have a fixed amount of daily processing capacity. At the beginning of each day, we observe the arrivals of jobs of different priorities and we need to decide which jobs should be scheduled for which days. The jobs that are waiting to be processed incur a holding cost depending on their priority levels and we also have the option of rejecting a job by incurring a penalty cost. We are interested in minimizing the total expected cost over a finite planning horizon.

The problem takes place over the set of days $\mathcal{T} = \{1, \ldots, \tau\}$. The set of possible priority levels for the jobs is $\mathcal{P} = \{1, \ldots, P\}$ where priority 1 is the highest priority and priority P is the lowest priority. The number of priority p jobs that arrive on day t is given by the random variable ω_t^p so that $\omega_t = \{\omega_t^p : p \in \mathcal{P}\}$ captures the job arrivals on day t. We assume that the job arrivals on different days are independent. If we schedule a priority p job that arrives on day t for day j, then we incur a holding cost of h_{jt}^p. The penalty cost of rejecting a priority p job that arrives on day t is d_t^p. We let $h_{jt}^p = \infty$ whenever it is infeasible to schedule a priority p job arriving on day t for day j. For example, since it is infeasible to schedule a job for a day in the past, we have $h_{jt}^p = \infty$ whenever $j < t$. Furthermore, we assume that all jobs can be scheduled only within a booking horizon of length S days. In other words, a job arriving on day t can only be scheduled for a day $j \in \mathcal{S}_t$ where $\mathcal{S}_t = \{t, \ldots, t + S - 1\}$. Consequently, we let $h_{jt}^p = \infty$ whenever $j \geq t + S$. Once we schedule a job for a particular day, this decision is fixed and cannot be

changed. We assume that all jobs consume one unit of processing capacity, but this assumption is only for notational brevity and it is straightforward to extend our approach to multiple units of capacity consumption.

Given that we are on day t, we let x_{jt} be the remaining capacity on day j so that $x_t = \{x_{jt} : j \in \mathcal{T}\}$ captures the state of the remaining capacities observed at the beginning of day t. We let u_{jt}^p be the number of priority p jobs that we schedule for day j on day t so that $u_t = \{u_{jt}^p : p \in \mathcal{P},\ j \in \mathcal{T}\}$ captures the decisions that we make on day t. In this case, the set of feasible decisions on day t is given by

$$\mathcal{U}(x_t, \omega_t) = \Big\{ u_t \in \mathbb{Z}_+^{|\mathcal{P}||\mathcal{T}|} : \sum_{p \in \mathcal{P}} u_{jt}^p \leq x_{jt} \quad \forall\, j \in \mathcal{T}, \quad \sum_{j \in \mathcal{T}} u_{jt}^p \leq \omega_t^p \quad \forall\, p \in \mathcal{P} \Big\}, \tag{4.1}$$

where the first set of constraints ensure that the decisions that we make comply with the remaining capacities and the second set of constraints ensure that the decisions that we make comply with the job arrivals. On the other hand, the total cost that we incur on day t is given by

$$\sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} h_{jt}^p\, u_{jt}^p + \sum_{p \in \mathcal{P}} d_t^p \Big[\omega_t^p - \sum_{j \in \mathcal{T}} u_{jt}^p \Big],$$

where the first term corresponds to the holding cost for the jobs that are scheduled for different days in the planning horizon and the second term corresponds to the penalty cost for the jobs that are rejected. We note that without loss of generality, it is possible to assume that $d_t^p = 0$ for all $p \in \mathcal{P},\ t \in \mathcal{T}$. To see this, we write the cost function above as $\sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} \big[h_{jt}^p - d_t^p\big] u_{jt}^p + \sum_{p \in \mathcal{P}} d_t^p\, \omega_t^p$. Since $\sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} d_t^p\, \omega_t^p$ is a constant that is independent of the decisions, assuming that $d_t^p = 0$ for all $p \in \mathcal{P}$, $t \in \mathcal{T}$ is equivalent to letting $c_{jt}^p = h_{jt}^p - d_t^p$ and working with the holding cost c_{jt}^p instead of h_{jt}^p. Throughout the rest of the paper, we indeed assume that $d_t^p = 0$ for all $p \in \mathcal{P},\ t \in \mathcal{T}$.

We use $V_t(x_t)$ to denote the minimum possible total expected cost over days $\{t, \ldots, \tau\}$ given that the state of the remaining capacities on day t is x_t. Letting e_j be the $|\mathcal{T}|$ dimensional unit vector with a one in the element corresponding to $j \in \mathcal{T}$, we can compute the value functions $\{V_t(\cdot) : t \in \mathcal{T}\}$ by solving the optimality equation

$$V_t(x_t) = \mathbb{E}\left\{ \min_{u_t \in \mathcal{U}(x_t, \omega_t)} \left\{ \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} c_{jt}^p \, u_{jt}^p + V_{t+1}(x_t - \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} u_{jt}^p \, e_j) \right\} \right\} \tag{4.2}$$

with the boundary condition that $V_{\tau+1}(\cdot) = 0$. The optimal total expected cost over the whole planning horizon is $V_1(x_1)$, where x_1 gives the initial state of the remaining capacities and it is a part of the problem data. The knowledge of $\{V_t(\cdot) : t \in \mathcal{T}\}$ can be used to find the optimal decisions on each day. In particular, if the state of the remaining capacities and the job arrivals on day t are respectively given by x_t and ω_t, then we can solve the optimization problem inside the expectation in (4.2) to find the optimal decisions on this day.

Unfortunately, the number of dimensions of the state vector x_t is equal to the number of days, which can easily be on the order of hundreds for practical applications. Furthermore, solving the optimality equation in (4.2) requires taking an expectation over the job arrivals on each day and this expectation can be difficult to compute. These considerations render finding the exact solution to the optimality equation in (4.2) intractable. In the next two sections, we give two possible approaches for finding approximate solutions to the optimality equation in (4.2). The first approach involves solving a linear program, whereas the second approach builds on this linear program to construct separable approximations to the value functions.

4.3 Linear Programming Formulation

One approach for finding approximate solutions to the optimality equation in (4.2) involves solving a linear program that is formulated under the assumption that the job arrivals take their expected values and it is possible to schedule fractional numbers of jobs for different days. In particular, using the decision variables $\{u_{jt}^p : p \in \mathcal{P},\ j \in \mathcal{T},\ t \in \mathcal{T}\}$ with the same interpretation as in the previous section, we can solve the problem

$$\begin{aligned}
\min \quad & \sum_{t\in\mathcal{T}} \sum_{j\in\mathcal{T}} \sum_{p\in\mathcal{P}} c_{jt}^p\, u_{jt}^p && && (4.3)\\
\text{subject to} \quad & \sum_{t\in\mathcal{T}} \sum_{p\in\mathcal{P}} u_{jt}^p \leq x_{j1} && j \in \mathcal{T} && (4.4)\\
& \sum_{j\in\mathcal{T}} u_{jt}^p \leq \mathbb{E}\{\omega_t^p\} && p \in \mathcal{P},\ t \in \mathcal{T} && (4.5)\\
& u_{jt}^p \geq 0 && p \in \mathcal{P},\ j \in \mathcal{T},\ t \in \mathcal{T} && (4.6)
\end{aligned}$$

to approximate the optimal total expected cost over the planning horizon. Constraints (4.4) in the problem above ensure that the decisions that we make over the planning horizon do not violate the remaining capacity on each day, whereas constraints (4.5) ensure that the decisions that we make on each day do not violate the expected numbers of job arrivals.

One important use of problem (4.3)-(4.6) occurs when we want to obtain lower bounds on the optimal total expected cost. In particular, if we denote z_{LP}^* the optimal objective value of problem (4.3)-(4.6), the next proposition shows that we have $z_{LP}^* \leq V_1(x_1)$ so that the optimal objective value of problem (4.3)-(4.6) provides a lower bound on the optimal total expected cost. Such lower bounds are useful to get a feel for the optimality gap of any suboptimal control policy.

Proposition 4.3.1 *We have* $z^*_{LP} \leq V_1(x_1)$.

Proof Our proof is standard, but it allows us to demonstrate a useful inequality. Given that the job arrivals over the planning horizon are $\omega = \{\omega_t^p : p \in \mathcal{P},\ t \in \mathcal{T}\}$, we let $\pi^*(\omega)$ be the total cost incurred by the optimal policy and $z^*_{LP}(\omega)$ be the optimal objective value of problem (4.3)-(4.6) that is obtained after replacing the right side of constraints (4.5) with $\{\omega_t^p : p \in \mathcal{P},\ t \in \mathcal{T}\}$. Replacing the right side of constraints (4.5) with $\{\omega_t^p : p \in \mathcal{P},\ t \in \mathcal{T}\}$ and solving problem (4.3)-(4.6) corresponds to making the decisions after observing first all of the job arrivals over the whole planning horizon. Furthermore, problem (4.3)-(4.6) allows scheduling fractional numbers of jobs. On the other hand, the optimal policy makes the decisions for a particular day after observing the job arrivals only on that day. Furthermore, the optimal policy schedules integer numbers of jobs. Therefore, it holds that $z^*_{LP}(\omega) \leq \pi^*(\omega)$. Taking expectations and noting that $z^*_{LP}(\omega)$ is a convex function of ω, Jensen's inequality implies that

$$z^*_{LP}(\mathbb{E}\{\omega\}) \leq \mathbb{E}\{z^*_{LP}(\omega)\} \leq \mathbb{E}\{\pi^*(\omega)\}. \tag{4.7}$$

The first expression in the chain of inequalities above is equal to z^*_{LP}, whereas the last expression is equal to the total expected cost incurred by the optimal policy, which is given by $V_1(x_1)$. □

Therefore, (4.7) implies that both z^*_{LP} and $\mathbb{E}\{z^*_{LP}(\omega)\}$ provide lower bounds on the optimal total expected cost. The lower bound provided by z^*_{LP} is looser, but this lower bound can be computed simply by solving problem (4.3)-(4.6). On the other hand, the lower bound provided by $\mathbb{E}\{z^*_{LP}(\omega)\}$ is tighter, but there is no closed form expression for the expectation in $\mathbb{E}\{z^*_{LP}(\omega)\}$ and computing this expectation requires estimation through Monte Carlo samples. Nevertheless, our computation experiments indicate that this extra computation burden usually pays

off and the lower bound provided by $\mathbb{E}\{z^*_{LP}(\omega)\}$ can be significantly tighter than the lower bound provided by z^*_{LP}.

Another important use of problem (4.3)-(4.6) occurs when we want to construct an approximate policy to make the decisions on each day. In particular, letting $\{\mu^*_j : j \in \mathcal{T}\}$ be the optimal values of the dual variables associated with constraints (4.4), we can use μ^*_j to estimate the value of a unit of remaining capacity on day j. This idea suggests using $\sum_{j\in\mathcal{T}} \mu^*_j\, x_{jt}$ as an approximation to $V_t(x_t)$. In this case, if the state of the remaining capacities and the job arrivals on day t are respectively given by x_t and ω_t, then we can replace $V_{t+1}(x_t - \sum_{j\in\mathcal{T}} \sum_{p\in\mathcal{P}} u^p_{jt}\, e_j)$ on the right side of (4.2) with $\sum_{j\in\mathcal{T}} \mu^*_j \big[x_{jt} - \sum_{p\in\mathcal{P}} u^p_{jt}\big]$ and solve the problem

$$\min_{u_t\in\mathcal{U}(x_t,\omega_t)} \left\{ \sum_{j\in\mathcal{T}} \sum_{p\in\mathcal{P}} c^p_{jt}\, u^p_{jt} + \sum_{j\in\mathcal{T}} \mu^*_j \big[x_{jt} - \sum_{p\in\mathcal{P}} u^p_{jt}\big] \right\} \tag{4.8}$$

to make the decisions on day t. We use this approach as a benchmark strategy later in the paper.

In the next section, we show how we can use problem (4.3)-(4.6) to construct separable approximations to the value functions.

4.4 Dynamic Programming Decomposition

The method that we develop in this section constructs separable approximations to the value functions by decomposing the optimality equation in (4.2) into a sequence of optimality equations, each involving a scalar state variable. We begin with a duality argument on problem (4.3)-(4.6). We let $\{\mu^*_j : j \in \mathcal{T}\}$ be the optimal values of the dual variables associated with constraints (4.4) in problem (4.3)-(4.6). We choose an arbitrary day i in the planning horizon and relax constraints (4.4) for

all of the other days by associating the dual multipliers $\{\mu_j^* : j \in \mathcal{T} \setminus \{i\}\}$ with them. In this case, if we let $\mathbf{1}(\cdot)$ be the indicator function, the objective function of problem (4.3)-(4.6) can be written as

$$\sum_{t\in\mathcal{T}}\sum_{j\in\mathcal{T}}\sum_{p\in\mathcal{P}} c_{jt}^p\, u_{jt}^p + \sum_{j\in\mathcal{T}\setminus\{i\}} \mu_j^* \Big[x_{j1} - \sum_{t\in\mathcal{T}}\sum_{p\in\mathcal{P}} u_{jt}^p\Big]$$
$$= \sum_{t\in\mathcal{T}}\sum_{j\in\mathcal{T}}\sum_{p\in\mathcal{P}} \big[c_{jt}^p - \mathbf{1}(j \neq i)\,\mu_j^*\big] u_{jt}^p + \sum_{j\in\mathcal{T}\setminus\{i\}} \mu_j^*\, x_{j1}.$$

Therefore, by linear programming duality, the optimal objective value of problem (4.3)-(4.6) is the same as the optimal objective value of the problem

$$\min \quad \sum_{t\in\mathcal{T}}\sum_{j\in\mathcal{T}}\sum_{p\in\mathcal{P}} \big[c_{jt}^p - \mathbf{1}(j \neq i)\,\mu_j^*\big] u_{jt}^p + \sum_{j\in\mathcal{T}\setminus\{i\}} \mu_j^*\, x_{j1} \tag{4.9}$$

$$\text{subject to} \quad \sum_{t\in\mathcal{T}}\sum_{p\in\mathcal{P}} u_{it}^p \leq x_{i1} \tag{4.10}$$

$$\sum_{j\in\mathcal{T}} u_{jt}^p \leq \mathbb{E}\{\omega_t^p\} \qquad p \in \mathcal{P},\ t \in \mathcal{T} \tag{4.11}$$

$$u_{jt}^p \geq 0 \qquad p \in \mathcal{P},\ j \in \mathcal{T},\ t \in \mathcal{T}. \tag{4.12}$$

Comparing problem (4.9)-(4.12) with problem (4.3)-(4.6) and ignoring the constant term $\sum_{j\in\mathcal{T}\setminus\{i\}} \mu_j^*\, x_{j1}$ in the objective function, we observe that problem (4.9)-(4.12) is the linear programming formulation for a capacity allocation problem where there is limited capacity only on day i and the capacities on all of the other days are infinite. In this capacity allocation problem, if we schedule a priority p job that arrives on day t for day j, then we incur a holding cost of $c_{jt}^p - \mathbf{1}(j \neq i)\,\mu_j^*$. We refer to this problem as the capacity allocation problem focused on day i. Then, the discussion in this paragraph and Proposition 4.3.1 imply that $z_{LP}^* - \sum_{j\in\mathcal{T}\setminus\{i\}} \mu_j^*\, x_{j1}$ provides a lower bound on the optimal total expected cost for the capacity allocation problem focused on day i.

On the other hand, if there is limited capacity only on day i and the capacities on all of the other days are infinite, then the set of feasible decisions can be written

as

$$\mathcal{U}_i(x_{it}, \omega_t) = \left\{ u_t \in \mathbb{Z}_+^{|\mathcal{P}||\mathcal{T}|} : \sum_{p \in \mathcal{P}} u_{it}^p \leq x_{it}, \quad \sum_{j \in \mathcal{T}} u_{jt}^p \leq \omega_t^p \quad \forall\, p \in \mathcal{P} \right\}. \tag{4.13}$$

The definition of the feasible set above is similar to the one in (4.1), but we only pay attention to the remaining capacity on day i. In this case, if there is limited capacity only on day i and the holding cost of scheduling a priority p job that arrives on day t for day j is $c_{jt}^p - \mathbf{1}(j \neq i)\, \mu_j^*$, then we can obtain the optimal total expected cost by computing the value functions $\{v_{it}(\cdot) : t \in \mathcal{T}\}$ through the optimality equation

$$v_{it}(x_{it}) = \mathbb{E}\left\{ \min_{u_t \in \mathcal{U}_i(x_{it}, \omega_t)} \left\{ \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} \left[c_{jt}^p - \mathbf{1}(j \neq i)\, \mu_j^* \right] u_{jt}^p + v_{i,t+1}(x_{it} - \sum_{p \in \mathcal{P}} u_{it}^p) \right\} \right\} \tag{4.14}$$

with the boundary condition that $v_{i,\tau+1}(\cdot) = 0$. The optimality equation in (4.14) is similar to the one in (4.2), but it keeps track of the remaining capacity only on day i. The subscript i in the value functions emphasizes this. By the discussion in the previous paragraph, we have $z_{LP}^* - \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^*\, x_{j1} \leq v_{i1}(x_{i1})$. Furthermore, the next proposition shows that $v_{i1}(x_{i1}) + \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^*\, x_{j1} \leq V_1(x_1)$ and we obtain

$$z_{LP}^* \leq v_{i1}(x_{i1}) + \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^*\, x_{j1} \leq V_1(x_1).$$

Noting that the result above holds for any day i, we also have

$$z_{LP}^* \leq \max_{i \in \mathcal{T}} \left\{ v_{i1}(x_{i1}) + \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^*\, x_{j1} \right\} \leq V_1(x_1). \tag{4.15}$$

Therefore, we can obtain a lower bound on the optimal total expected cost in the capacity allocation problem by solving the optimality equation in (4.14) and this lower bound is tighter than the one provided by the optimal objective value of problem (4.3)-(4.6). We refer to the bound from (4.15) as the dynamic programming decomposition (DPD) bound.

Proposition 4.4.1 *We have* $v_{it}(x_{it}) + \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^* \, x_{jt} \leq V_t(x_t)$ *for all* $t \in \mathcal{T}$ *and all* $i \in \mathcal{T}$.

Proof We fix $i \in \mathcal{T}$ and use a standard induction argument over the days of the planning horizon. The result is easy to show for the last day. We assume that the result holds for day $t+1$ and let $\{\hat{u}_{jt}^p : p \in \mathcal{P}, \ j \in \mathcal{T}\}$ be an optimal solution to the optimization problem inside the expectation (4.2), $\vartheta_t(x_t, \omega_t)$ be the optimal objective value of this problem and $\vartheta_{it}(x_{it}, \omega_t)$ be the optimal objective value of the optimization problem inside the expectation in (4.14). We have

$$\begin{aligned}
\vartheta_t(x_t, \omega_t) &= \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} c_{jt}^p \, \hat{u}_{jt}^p + V_{t+1}(x_t - \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} \hat{u}_{jt}^p \, e_j) \\
&\geq \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} c_{jt}^p \, \hat{u}_{jt}^p + v_{i,t+1}(x_{it} - \sum_{p \in \mathcal{P}} \hat{u}_{it}^p) + \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^* \Big[x_{jt} - \sum_{p \in \mathcal{P}} \hat{u}_{jt}^p \Big] \\
&= \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} \big[c_{jt}^p - \mathbf{1}(j \neq i) \, \mu_j^* \big] \hat{u}_{jt}^p + v_{i,t+1}(x_{it} - \sum_{p \in \mathcal{P}} \hat{u}_{it}^p) + \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^* \, x_{jt} \\
&\geq \vartheta_{it}(x_{it}, \omega_t) + \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^* \, x_{jt},
\end{aligned}$$

where the first inequality follows from the induction assumption and the second inequality follows from the fact that $\{\hat{u}_{jt}^p : p \in \mathcal{P}, \ j \in \mathcal{T}\}$ is a feasible but not necessarily an optimal solution to the optimization problem inside the expectation in (4.14). The result follows by taking expectations in the chain of inequalities above and noting that $V_t(x_t) = \mathbb{E}\{\vartheta_t(x_t, \omega_t)\}$ and $v_{it}(x_{it}) = \mathbb{E}\{\vartheta_{it}(x_{it}, \omega_t)\}$. $\square$

Proposition 4.4.1 suggests possible approximations to $V_t(x_t)$. On the other hand, since each of $\{v_{it}(x_{it}) + \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^* \, x_{jt} : i \in \mathcal{T}\}$ provides a lower bound on $V_t(x_t)$, it is not clear which of these approximations to use. We propose two different methods to resolve this ambiguity. First, we consider approximating

$V_t(x_t)$ by

$$\frac{1}{|\mathcal{T}|} \sum_{i \in \mathcal{T}} \left\{ v_{it}(x_{it}) + \sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^* \, x_{jt} \right\}, \tag{4.16}$$

i.e. by averaging all available approximations. To motivate our second approach, we note that in the approximation of $V_t(x_t)$ corresponding to day i, the component $v_{it}(x_{it})$ captures the value of the remaining capacity on day i, whereas the component $\sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^* \, x_{jt}$ indicates the value of the remaining capacities on days other than day i. We note that the component $\sum_{j \in \mathcal{T} \setminus \{i\}} \mu_j^* \, x_{jt}$ is somewhat trivial in the sense that it exactly corresponds to how the linear programming formulation evaluates the capacities on days other than day i. To obtain more sophisticated estimates for the values of the remaining capacities on days other than day i, we propose computing $\{v_{it}(\cdot) : t \in \mathcal{T}\}$ for all $i \in \mathcal{T}$. In this case, given that we are on day t, the component $v_{it}(x_{it})$ captures the value of the remaining capacity on day i and we can use $\sum_{i \in \mathcal{T}} v_{it}(x_{it})$ as an approximation to $V_t(x_t)$. Preliminary simulation runs indicate that this (second) approach to value functions approximation performs better than the approach suggested by (4.16). Hence, in the computational results section of this paper, we only report the results obtained using the second approach and refer to is as the dynamic programming decomposition (DPD) policy.

In the next section, we re-solve the computational issues that are related to solving the optimality equation in (4.14).

4.5 Solving the Optimality Equation

The value functions $\{v_{it}(\cdot) : t \in \mathcal{T}\}$ computed through the optimality equation in (4.14) involve a scalar state variable. Therefore, it is straightforward to store these

value functions. However, solving the optimality equation in (4.14) still requires dealing with an optimization problem with $|\mathcal{P}||\mathcal{T}|$ decision variables. The next proposition shows that the optimization problem inside the expectation in (4.14) can be solved efficiently.

Proposition 4.5.1 *We can solve the optimization problem inside the expectation in* (4.14) *by a sort operation.*

Proof We let $\bar{K} = \max_{j \in \mathcal{T}}\{x_{j1}\}$ so that the remaining capacities are always less than or equal to $\bar{K}$. By using induction over the days of the planning horizon, it is possible to show that $\{v_{it}(\cdot) : t \in \mathcal{T}\}$ are convex functions in the sense that $2\, v_{it}(k) \leq v_{it}(k+1) + v_{it}(k-1)$ for all $k = 1, \ldots, \bar{K} - 1$, $t \in \mathcal{T}$. In this case, letting $\mathcal{K} = \{0, \ldots, \bar{K} - 1\}$ and $\Delta_{it}^k = v_{i,t+1}(k+1) - v_{i,t+1}(k)$ for all $k \in \mathcal{K}$, we associate the decision variables $\{w_{it}^k : k \in \mathcal{K}\}$ with the first differences $\{\Delta_{it}^k : k \in \mathcal{K}\}$ and write the optimization problem inside the expectation in (4.14) as

$$\begin{aligned}
\min \quad & \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} \left[c_{jt}^p - \mathbf{1}(j \neq i)\, \mu_j^*\right] u_{jt}^p + \sum_{k \in \mathcal{K}} \Delta_{it}^k\, w_{it}^k + \; v_{i,t+1}(0) \\
\text{subject to} \quad & \sum_{p \in \mathcal{P}} u_{it}^p + \sum_{k \in \mathcal{K}} w_{it}^k = x_{it} \\
& \sum_{j \in \mathcal{T}} u_{jt}^p + y_t^p = \omega_t^p && p \in \mathcal{P} \\
& w_{it}^k \leq 1 && k \in \mathcal{K} \\
& u_{jt}^p, y_t^p, w_{it}^k \in \mathbb{Z}_+ && p \in \mathcal{P},\ j \in \mathcal{T},\ k \in \mathcal{K},
\end{aligned}$$

where we use the slack variables $\{y_t^p : p \in \mathcal{P}\}$ in the second set of constraints above.

For a particular priority level p, we consider the decision variables $\{u_{jt}^p : j \in \mathcal{T} \setminus \{i\}\}$ and y_t^p in the problem above. As far as the constraints are concerned,

these decision variables appear only in the second set of constraints. Therefore, if any one of the decision variables $\{u^p_{jt} : j \in \mathcal{T} \setminus \{i\}\}$ and y^p_t takes a nonzero value in the optimal solution, then this decision variable has to be the one with the smallest objective function coefficient. This implies that we can replace all of these decision variables with a single decision variable, say z^p_t, and the objective function coefficient of z^p_t would be equal to the smallest of the objective function coefficients of the decision variables $\{u^p_{jt} : j \in \mathcal{T} \setminus \{i\}\}$ and y^p_t. Since the objective function coefficients of the decision variables $\{u^p_{jt} : j \in \mathcal{T} \setminus \{i\}\}$ are $\{c^p_{jt} - \mu^*_j : j \in \mathcal{T} \setminus \{i\}\}$ and the objective function coefficient of the decision variable y^p_t is zero, letting $\hat{c}^p_t = \min\{\min\{c^p_{jt} - \mu^*_j : j \in \mathcal{T} \setminus \{i\}\}, 0\}$, the last problem above becomes

$$\begin{aligned}
\min \quad & \sum_{p\in\mathcal{P}} c^p_{it}\, u^p_{it} + \sum_{p\in\mathcal{P}} \hat{c}^p_t\, z^p_t + \sum_{k\in\mathcal{K}} \Delta^k_{it}\, w^k_{it} + \ v_{i,t+1}(0) && && (4.17)\\
\text{subject to} \quad & \sum_{p\in\mathcal{P}} u^p_{it} + \sum_{k\in\mathcal{K}} w^k_{it} = x_{it} && && (4.18)\\
& u^p_{it} + z^p_t = \omega^p_t && p \in \mathcal{P} && (4.19)\\
& w^k_{it} \le 1 && k \in \mathcal{K} && (4.20)\\
& u^p_{it}, z^p_t, w^k_{it} \in \mathbb{Z}_+ && p \in \mathcal{P},\ k \in \mathcal{K}. && (4.21)
\end{aligned}$$

Using constraints (4.19), we write $z^p_t = \omega^p_t - u^p_{it}$ for all $p \in \mathcal{P}$ and substitute these decision variables into the objective function to obtain

$$\begin{aligned}
&\sum_{p\in\mathcal{P}} c^p_{it}\, u^p_{it} + \sum_{p\in\mathcal{P}} \hat{c}^p_t \big[\omega^p_t - u^p_{it}\big] + \sum_{k\in\mathcal{K}} \Delta^k_{it}\, w^k_{it} + \ v_{i,t+1}(0)\\
&\qquad = \sum_{p\in\mathcal{P}} \big[c^p_{it} - \hat{c}^p_t\big] u^p_{it} + \sum_{p\in\mathcal{P}} \hat{c}^p_t\, \omega^p_t + \sum_{k\in\mathcal{K}} \Delta^k_{it}\, w^k_{it} + \ v_{i,t+1}(0).
\end{aligned}$$

Therefore, dropping the constant term $\sum_{p\in\mathcal{P}} \hat{c}_t^p\, \omega_t^p + v_{i,t+1}(0)$ from the objective function, we write problem (4.17)-(4.21) as

$$\begin{aligned}
\min \quad & \sum_{p\in\mathcal{P}} \left[c_{it}^p - \hat{c}_t^p\right] u_{it}^p + \sum_{k\in\mathcal{K}} \Delta_{it}^k\, w_{it}^k && && (4.22)\\
\text{subject to} \quad & \sum_{p\in\mathcal{P}} u_{it}^p + \sum_{k\in\mathcal{K}} w_{it}^k = x_{it} && && (4.23)\\
& u_{it}^p \le \omega_t^p && p\in\mathcal{P} && (4.24)\\
& w_{it}^k \le 1 && k\in\mathcal{K} && (4.25)\\
& u_{it}^p, w_{it}^k \in \mathbb{Z}_+ && p\in\mathcal{P},\ k\in\mathcal{K}. && (4.26)
\end{aligned}$$

Problem (4.22)-(4.26) is a knapsack problem with each item consuming one unit of capacity. The items are indexed by $p \in \mathcal{P}$ and $k \in \mathcal{K}$. The (dis)utilities of items $p \in \mathcal{P}$ and $k \in \mathcal{K}$ are respectively $c_{it}^p - \hat{c}_t^p$ and Δ_{it}^k. The capacity of the knapsack is x_{it}. We can put at most ω_t^p units of item $p \in \mathcal{P}$ and one unit of item $k \in \mathcal{K}$ into the knapsack. The result follows by the fact that a knapsack problem with each item consuming one unit of capacity can be solved by sorting the objective function coefficients and filling the knapsack starting from the item with the smallest objective function coefficient. □

Another difficulty with the optimality equation in (4.14) is that we need to compute the expectation of the optimal objective value of the optimization problem inside the first set of curly brackets. The proof of Proposition 4.5.1 shows that this optimization problem is equivalent to problem (4.22)-(4.26), which is, in turn, a knapsack problem, where each item consumes one unit of capacity and there is a random upper bound on how many units of a particular item we can put into the knapsack. Powell and Cheung (1994) derive a closed form expression for the expectation of the optimal objective value of such a knapsack problem. We briefly describe how their result relates to our setting.

To reduce the notational clutter, we note that for appropriate choices of the set of items $\mathcal{N} = \{1, \ldots, N\}$, (dis)utilities $\{\beta_n : n \in \mathcal{N}\}$, integer valued knapsack capacity Q and integer valued random upper bounds $\alpha = \{\alpha_n : n \in \mathcal{N}\}$, problem (4.22)-(4.26) is a knapsack problem of the form

$$\begin{aligned} \min \quad & \sum_{n \in \mathcal{N}} \beta_n \, z_n & & (4.27) \\ \text{subject to} \quad & \sum_{n \in \mathcal{N}} z_n = Q & & (4.28) \\ & z_n \leq \alpha_n & n \in \mathcal{N} & (4.29) \\ & z_n \in \mathbb{Z}_+ & n \in \mathcal{N}. & (4.30) \end{aligned}$$

Using $\zeta(\alpha)$ to denote the optimal objective value of the problem above as a function of α, we are interested in computing $\mathbb{E}\{\zeta(\alpha)\}$. For a given realization of α, we can solve problem (4.27)-(4.30) by sorting the objective function coefficients of the decision variables and filling the knapsack starting from the item with the smallest objective function coefficient. We assume that $\beta_1 \leq \beta_2 \leq \ldots \leq \beta_N$, in which case it is optimal to start from the item with the smallest index.

We let $\phi(n, q)$ be the probability that the nth item uses the qth unit of the available knapsack capacity in the optimal solution. If we know $\phi(n, q)$ for all $n \in \mathcal{N},\ q = 1, \ldots, Q$, then we can compute the expectation of the optimal objective value of problem (4.27)-(4.30) as

$$\mathbb{E}\{\zeta(\alpha)\} = \sum_{q=1}^{Q} \sum_{n \in \mathcal{N}} \beta_n \, \phi(n, q).$$

Computing $\phi(n, q)$ turns out to be not too difficult. Since the optimal solution fills the knapsack starting from the item with the smallest index, for the nth item to use the qth unit of capacity in the knapsack, the total capacity consumed by the first $n - 1$ items should be strictly less than q and the total capacity consumed

by the first n items should be greater than or equal to q. Therefore, we have $\phi(n,q) = \mathbb{P}\{\alpha_1 + \ldots + \alpha_{n-1} < q \leq \alpha_1 + \ldots + \alpha_n\}$ and we can compute $\phi(n,q)$ as long as we can compute the convolutions of the distributions of $\{\alpha_n : n \in \mathcal{N}\}$.

The discussion in the previous two paragraphs provides a method to compute the expectation of the optimal objective value of the optimization problem inside the first set of curly brackets in (4.14). In the next section, we decribe how we can use the value functions computed through the optimality equation in (4.14) to make the decisions on each day.

4.6 Applying the Greedy Policies

The discussion at the end of Section 4.4 suggests using $\sum_{j \in \mathcal{T}} v_{jt}(x_{jt})$ as an approximation to $V_t(x_t)$. In this case, if the state of the remaining capacities and the job arrivals on day t are respectively given by x_t and ω_t, then we can replace $V_{t+1}(x_t - \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} u_{jt}^p \, e_j)$ on the right side of (4.2) with $\sum_{j \in \mathcal{T}} v_{j,t+1}(x_{jt} - \sum_{p \in \mathcal{P}} u_{jt}^p)$ and solve the problem

$$\min_{u_t \in \mathcal{U}(x_t, \omega_t)} \left\{ \sum_{j \in \mathcal{T}} \sum_{p \in \mathcal{P}} c_{jt}^p \, u_{jt}^p + \sum_{j \in \mathcal{T}} v_{j,t+1}(x_{jt} - \sum_{p \in \mathcal{P}} u_{jt}^p) \right\} \tag{4.31}$$

to make the decisions on day t. The problem above involves $|\mathcal{P}||\mathcal{T}|$ decision variables. The next proposition shows that we can efficiently solve this problem as a min-cost network flow problem.

Proposition 4.6.1 *We can solve problem* (4.31) *as a min-cost network flow problem.*

Proof As mentioned in the proof of Proposition 4.5.1, it is possible to show that $\{v_{jt}(\cdot) : j \in \mathcal{T},\ t \in \mathcal{T}\}$ are convex functions. In this case, letting $\mathcal{K}$ and Δ^k_{jt} be defined as in the proof of Proposition 4.5.1, we associate the decision variables $\{w^k_{jt} : k \in \mathcal{K},\ j \in \mathcal{T}\}$ with the first differences $\{\Delta^k_{jt} : k \in \mathcal{K},\ j \in \mathcal{T}\}$ and write problem (4.31) as

$$\begin{aligned}
\min \quad & \sum_{j\in\mathcal{T}}\sum_{p\in\mathcal{P}} c^p_{jt}\, u^p_{jt} + \sum_{j\in\mathcal{T}}\sum_{k\in\mathcal{K}} \Delta^k_{jt}\, w^k_{jt} \\
\text{subject to} \quad & \sum_{p\in\mathcal{P}} u^p_{jt} + \sum_{k\in\mathcal{K}} w^k_{jt} = x_{jt} && j \in \mathcal{T} \\
& \sum_{j\in\mathcal{T}} u^p_{jt} \leq \omega^p_t && p \in \mathcal{P} \\
& w^k_{jt} \leq 1 && k \in \mathcal{K},\ j \in \mathcal{T} \\
& u^p_{jt}, w^k_{jt} \in \mathbb{Z}_+ && p \in \mathcal{P},\ k \in \mathcal{K},\ j \in \mathcal{T}
\end{aligned}$$

We define the new decision variables $\{y^p_t : p \in \mathcal{P}\}$ as $y^p_t = \sum_{j\in\mathcal{T}} u^p_{jt}$ for all $p \in \mathcal{P}$, in which case the problem above becomes

$$\begin{aligned}
\min \quad & \sum_{j\in\mathcal{T}}\sum_{p\in\mathcal{P}} c^p_{jt}\, u^p_{jt} + \sum_{j\in\mathcal{T}}\sum_{k\in\mathcal{K}} \Delta^k_{jt}\, w^k_{jt} && && (4.32) \\
\text{subject to} \quad & \sum_{p\in\mathcal{P}} u^p_{jt} + \sum_{k\in\mathcal{K}} w^k_{jt} = x_{jt} && j \in \mathcal{T} && (4.33) \\
& \sum_{j\in\mathcal{T}} u^p_{jt} - y^p_t = 0 && p \in \mathcal{P} && (4.34) \\
& y^p_t \leq \omega^p_t && p \in \mathcal{P} && (4.35) \\
& w^k_{jt} \leq 1 && k \in \mathcal{K},\ j \in \mathcal{T} && (4.36) \\
& u^p_{jt}, y^p_t, w^k_{jt} \in \mathbb{Z}_+ && p \in \mathcal{P},\ k \in \mathcal{K},\ j \in \mathcal{T}. && (4.37)
\end{aligned}$$

The problem above is a min-cost network flow problem. In particular, we consider a network composed of two sets of nodes $\mathcal{N}_1 = \{j \in \mathcal{T}\}$ and $\mathcal{N}_2 = \{p \in \mathcal{P}\}$, along with a sink node. The decision variable u^p_{jt} corresponds to an arc connecting node

$j \in \mathcal{N}_1$ to node $p \in \mathcal{N}_2$. The decision variable y_t^p corresponds to an arc connecting node $p \in \mathcal{N}_2$ to the sink node. Finally, the decision variable w_{jt}^k corresponds to an arc connecting node $j \in \mathcal{N}_1$ to the sink node. The supply of node $j \in \mathcal{N}_1$ is x_{jt} and constraints (4.33) are the flow balance constraints for the nodes in $\mathcal{N}_1$. The supply of node $p \in \mathcal{P}$ is zero and constraints (4.34) are the flow balance constraints for the nodes in $\mathcal{N}_2$. The flow balance constraint for the sink node is redundant and it is omitted. Constraints (4.35) and (4.36) act as simple upper bounds on the arcs. Figure 4.1 shows the structure of the network for the case where $\mathcal{T} = \{1, 2\}$, $\mathcal{P} = \{a, b\}$ and $\mathcal{K} = \{k, l\}$.

Since problem (4.31) can be solved as a min-cost network flow problem, its continuous relaxation naturally provides integer solutions. □

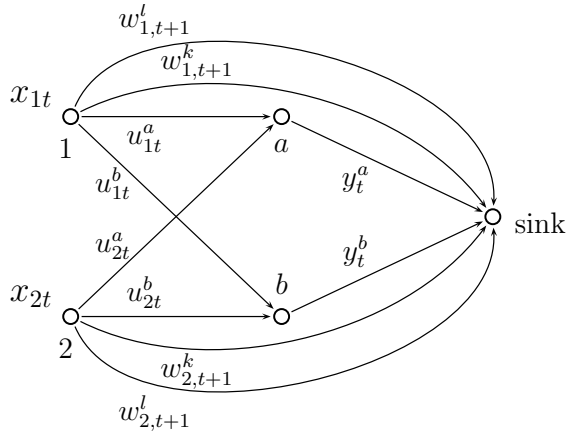

Figure 4.1: Network for $\mathcal{T} = \{1, 2\}$, $\mathcal{P} = \{a, b\}$ and $\mathcal{K} = \{k, l\}$.

4.7 Computational Experiments

In this section, we compare the performance of the separable value function approximations computed through the method in Section 4.4 with a number of benchmark

strategies.

4.7.1 Experimental Setup and Benchmark Strategies

Our experimental setup is based on Erdelyi and Topaloglu (2009*a*). In particular, we first generate a base problem and then modify its parameters to obtain test problems with different characteristics. In the base problem, we have $|\mathcal{T}| = 100$ for the length of the planning horizon, $|\mathcal{P}| = 3$ for the number of priority levels and $S = 7$ for the length of the booking horizon. The holding costs are $h_{jt}^p = 2^{|\mathcal{P}|-p}\, 1.25^{j-t}$ for all $t \in \mathcal{T}$, $p \in \mathcal{P}$ and $j \in \mathcal{S}_t$ where $\mathcal{S}_t = \{t, \ldots, t+S-1\}$. The penalty costs are $d_t^p = \beta\, 2^{|\mathcal{P}|-p}\, 1.25^{S-1}$ for all $t \in \mathcal{T}$ and $p \in \mathcal{P}$. In the base problem, we assume that $\beta = 5$. The daily job arrivals for the first, second and third priority levels are uniformly distributed over $[5, 15]$, $[10, 30]$ and $[20, 60]$, respectively. We note that this scenario corresponds to a coefficient of variation equal to $\frac{1}{\sqrt{12}} \cong 0.3$. We assume that the maximum daily processing capacity is $c = 70$. We define $\rho = \frac{1}{c\,S} \sum_{s \in \mathcal{S}_1} (c - x_{s1})$ so that ρ corresponds to the portion of the total capacities in the first S days already reserved before job arrivals on day 1 are processed. In the base problem, we assume $\rho = 32\%$.

We compare the performance of the dynamic programming decomposition (DPD) algorithm introduced in Section 4.4 with three benchmark strategies. Our first benchmark strategy corresponds to the decision rule described at the end of Section 4.3. Since this rule employs linear approximations to value functions, we refer to this strategy as LAX. In addition to the basic version of the algorithm as explained in Section 4.3, we test a dynamic version of the algorithm where values of dual variables $\{\mu_j^* : j \in \mathcal{T}\}$ are updated during simulation runs. In particular, we divide the planning horizon into M equal segments and re-solve (4.3)-(4.6) at time

periods $\{t_m = \lfloor 1 + (m-1)\tau/M \rfloor : m = 1, \ldots, M\}$. We note that that at the beginning of segment m, we replace $\mathcal{T}$ in problem (4.3)-(4.6) with $\mathcal{T}_{t_m} = \{t_m, \ldots, \tau\}$ and the right side of constraints (4.4) with the current remaining daily capacities $\{x_{j,t_m} : j \in \mathcal{T}_{t_m}\}$. Letting $\{\mu_j^* : j \in \mathcal{T}_{t_m}\}$ be the optimal values of dual variables corresponding to the updated capacity constraints (4.4), we use these when computing linear approximations to value functions until we reach the beginning of next segment. For the test problems reported in next section, we tested both the static version of LAX as well as its dynamic version with $M = 5$. Since the dynamic version did not always prove to be an improvement over the static one, we always report the better of the two performances for each test problem.

The second benchmark strategy we use is a simple first come first serve policy. Starting with the highest priority jobs and continuing with jobs of lower priority levels in a descending order, this strategy schedules all jobs to the earliest available day in the booking horizon. If the policy runs out of space in the booking horizon, the remaining (not yet scheduled) jobs are denied. We refer to this policy as FC. Although this strategy is not expected to provide quality results, it can be used as an indicator of how challenging a particular test problem is.

To motivate our third benchmark strategy, we note that LAX policy uses the dual solution to the linear program (4.3)-(4.6). Naturally, however, one can consider using the primal solution to a similar linear program to make capacity allocation decisions. In particular, at time period t given remaining capacities $\{x_{jt} : j \in \mathcal{T}\}$ and observed job arrivals $\{\omega_t^p : p \in \mathcal{P}\}$, the idea is to consider the

following linear program

$$\min \quad \sum_{s\in\mathcal{T}_t}\sum_{j\in\mathcal{T}_t}\sum_{p\in\mathcal{P}} c_{js}^{p}\, u_{js}^{p} \tag{4.38}$$

$$\text{subject to} \quad \sum_{s\in\mathcal{T}_t}\sum_{p\in\mathcal{P}} u_{js}^{p} \le x_{jt} \qquad j\in\mathcal{T}_t \tag{4.39}$$

$$\sum_{j\in\mathcal{T}_t} u_{jt}^{p} \le \omega_t^{p} \qquad p\in\mathcal{P} \tag{4.40}$$

$$\sum_{j\in\mathcal{T}_t} u_{js}^{p} \le \mathbb{E}\{\omega_s^{p}\} \qquad p\in\mathcal{P},\ s\in\mathcal{T}_{t+1} \tag{4.41}$$

$$u_{js}^{p} \ge 0 \qquad p\in\mathcal{P},\ j\in\mathcal{T}_t,\ s\in\mathcal{T}_t. \tag{4.42}$$

We note that the difference between problems (4.3)-(4.6) and (4.38)-(4.42) is that the latter is only concerned with the portion of the planning horizon starting at time period t. Furthermore, rather than expected values of job arrivals at time period t, the program above uses actually observed job arrivals for this time period. Assuming job arrivals take on their expected values in subsequent time periods, the optimal solution to the above problem, denoted $\{u_{jt}^{*} : j\in\mathcal{T}_t\}$, gives the capacity allocation at time period t. Although this policy is not necessarily optimal with stochastic job arrivals, we tested it as a benchmark. In several setup runs we encountered both scenarios when the policy performed relatively well and scenarios when it performed poorly. Later, we discovered that when the performance of the strategy is poor, it helped significantly to replace the expected values on the right side of constraints (4.41) with (random) realizations of job arrivals $\{\omega_s^{p} : p\in\mathcal{P}, s\in\mathcal{T}_{t+1}\}$. We refer to the original variant of the method as LPP_{E} and its randomized version as LPP_{R}. For the test problems reported in next section, we tested both variants of the policy and always reported the better of the two results. We refer to this compound policy as LPP.

4.7.2 Computational Results

Table 4.1 shows the computational results for the base problem. The second column gives the lower bound on the optimal total expected cost provided by problem (4.3)-(4.6). The third column gives the estimate of the expected value of the lower bound provided by the perfect hindsight linear program. We omit reporting the DPD bounds because it turns out that these are not significantly better than the bounds obtained from the deterministic linear program. The fourth, fifth, sixth and seventh columns in this table show the estimates of total expected costs incurred by FC, LAX, LPP and DPD, respectively. We estimate these costs by simulating the performances of different methods under 100 job arrival trajectories using common random numbers for all methods. The numbers in the brackets in the second row of these columns indicate the percentage of all jobs rejected by FC, LAX, LPP and DPD, respectively. We note that, almost exclusively, the lowest priority jobs are rejected. The eighth column shows the percent gap between the performance of DPD and FC. Similarly, the ninth and tenth columns show the performance gaps between DPD and the remaining two methods. The eleventh column shows the percent gap between the total expected cost incurred by DPD and the lower bound reported in the third column. Consequently, the eleventh columns gives an upper bound on the gap between the performances of DPD and the optimal policy.

Table 4.1: Computational results for the base problem.

Test	Lower Bound		Total Expected Cost				% Gap with DPD			DPD
Prob.	$z(\mathbb{E}\{d\})$	$\mathbb{E}\{z(d)\}$	FC	LAX	LPP	DPD	FC	LAX	LPP	gap
Base	13,979	14,771	18,747	16,528	17,238	15,885	18.02	4.05	8.52	7.54
			(0.02)	(2.46)	(4.23)	(2.05)				

The results suggest that DPD performs significantly better than all benchmark

methods for the base problem. On the other hand, FC performs noticeably worse than all other methods highlighting the importance of protecting capacity for future high priority job arrivals. Comparing the performance of LAX and LPP suggests that LAX performs better, presumably because it does not make a mistake of rejecting too many jobs. Naturally, the least percentage of jobs is rejected by FC. This, however, comes at the expense of having too many high priority jobs waiting too long in the system. Overall, DPD appears to find the best balance between the percentage of rejected jobs and protecting the capacities for future high priority jobs. The simulations also confirm that the lower bound $\mathbb{E}\{z^*_{LP}(\omega)\}$ is tighter than the lower bound $z^*_{LP}(\mathbb{E}\{\omega\})$ with a noticeable gap of 5.36% between the two bounds for the base problem.

In Tables 4.2-4.8, we vary different parameters of the base problem to generate test problems with modified characteristics. The first columns in these tables indicate the value of the modified parameter. The interpretations of other columns in Tables 4.2-4.8 remain the same as in Table 4.1.

Table 4.2: Computational results with varying number of priority levels.

	Lower Bound		Total Expected Cost				% Gap with DPD			DPD
N	$z(\mathbb{E}\{d\})$	$\mathbb{E}\{z(d)\}$	FC	LAX	LPP	DPD	FC	LAX	LPP	gap
2	11,654	12,574	15,545	14,233	15,184	13,804	12.61	3.11	10.00	9.78
			(0.13)	(2.53)	(4.41)	(2.10)				
3	13,979	14,771	18,747	16,528	17,238	15,885	18.02	4.05	8.52	7.54
			(0.02)	(2.46)	(4.23)	(2.05)				
4	17,234	17,893	23,577	19,966	20,108	18,964	24.33	5.28	6.03	5.99
			(0.02)	(2.58)	(4.04)	(2.05)				
5	19,838	20,520	27,594	22,832	22,656	21,581	27.86	5.80	4.98	5.17
			(0.00)	(2.63)	(4.07)	(2.19)				

In Table 4.2, we vary the number of priority levels. We note that DPD performs consistently better than all benchmark strategies for all test problems although the performance gap between DPD and LPP decreases with increasing number of

priority levels.

In Table 4.3, we vary the length of the booking horizon. Again, DPD is noticeably better than FC and LAX for all test problems. However, for longer booking horizons, DPD is outperformed by LPP. We note that the strong performance of LPP for large booking horizons is obtained due to LPP_E which not typical as, for most test problems, LPP_R performs better that LPP_E. We deduce that LPP_E benefits from longer booking horizons as it allows the policy to avoid the mistake of rejecting too many jobs if the capacities appear tight.

Table 4.3: Computational results with varying booking horizon length.

	Lower Bound		Total Expected Cost				% Gap with DPD			DPD
S	$z(\mathbb{E}\{d\})$	$\mathbb{E}\{z(d)\}$	FC	LAX	LPP	DPD	FC	LAX	LPP	gap
3	11,994	13,018	14,390	14,217	15,063	14,012	2.70	1.46	7.50	7.64
			(0.72)	(1.31)	(6.89)	(1.04)				
5	12,566	13,590	16,417	15,197	16,426	14,722	11.51	3.23	11.57	8.33
			(0.21)	(1.65)	(6.15)	(1.32)				
7	13,979	14,771	18,747	16,528	17,238	15,885	18.02	4.05	8.52	7.54
			(0.02)	(2.46)	(4.23)	(2.05)				
9	16,527	16,936	21,390	20,075	17,810	17,885	19.60	12.24	-0.42	5.60
			(0.00)	(1.68)	(2.63)	(1.96)				
11	19,977	20,374	24,448	24,424	20,972	21,578	13.30	13.19	-2.81	5.91
			(0.00)	(0.84)	(1.87)	(1.19)				

In Table 4.4, we vary the coefficient of variation for the job arrivals. DPD performs significantly better than all benchmark methods and the performace gap between DPD and both LAX and LPP increases with increasing variation. This is encouraging because addressing temporal dynamics of job arrivals was one of the goals behind developing the DPD rule.

Table 4.4: Computational results with varying coefficient of variation values.

	Lower Bound		Total Expected Cost				% Gap with DPD			DPD
CV	$z(\mathbb{E}\{d\})$	$\mathbb{E}\{z(d)\}$	FC	LAX	LPP	DPD	FC	LAX	LPP	gap
0.0	13,979	13,979	17,867	13,979	13,979	13,979	27.81	0.00	0.00	0.00
			(0.00)	(2.40)	(2.40)	(2.40)				
0.1	13,979	14,154	17,891	14,637	14,992	14,411	24.15	1.57	4.03	1.82
			(0.00)	(2.41)	(2.89)	(2.23)				
0.2	13,979	14,380	18,099	15,434	15,985	14,991	20.73	2.96	6.63	4.25
			(0.00)	(2.42)	(3.42)	(2.16)				
0.3	13,979	14,771	18,747	16,528	17,238	15,885	18.02	4.05	8.52	7.54
			(0.02)	(2.46)	(4.23)	(2.05)				
0.4	13,979	15,222	19,515	17,718	18,549	16,909	15.41	4.78	9.70	11.08
			(0.16)	(2.52)	(5.16)	(1.96)				
0.5	13,979	15,810	20,455	19,251	20,135	18,325	11.62	5.05	9.88	15.91
			(0.41)	(2.70)	(6.33)	(1.93)				

In Table 4.5, we vary the daily available capacity. We note that with increasing capacity, the problem becomes less challenging and the performance gaps between different policies become negligible as all policies perform close to the optimal policy. By contrast, with tighter capacities, the problem becomes more challenging and it is encouraging that DPD then outperforms LAX and LPP by a wider margin. On the other hand, it appears that for very tight capacities, the test problem again becomes somewhat less challenging and the performances of all three non-trivial policies level up and approach the performance of optimal policy.

Table 4.5: Computational results with varying daily available capacities.

	Lower Bound		Total Expected Cost				% Gap with DPD			DPD
c	$z(\mathbb{E}\{d\})$	$\mathbb{E}\{z(d)\}$	FC	LAX	LPP	DPD	FC	LAX	LPP	gap
55	34,841	34,930	57,896	37,428	36,524	35,868	61.41	4.35	1.83	2.69
			(18.22)	(20.11)	(20.13)	(18.75)				
65	20,023	20,365	36,237	24,688	23,828	21,588	67.86	14.36	10.38	6.01
			(3.58)	(9.33)	(9.59)	(6.92)				
70	13,979	14,771	18,747	16,528	17,238	15,885	18.02	4.05	8.52	7.54
			(0.02)	(2.46)	(4.23)	(2.05)				
75	11,858	12,218	12,515	12,746	12,265	12,305	1.71	3.58	-0.33	0.71
			(0.00)	(0.00)	(0.00)	(0.00)				
85	11,342	11,389	11,448	11,459	11,396	11,397	0.45	0.54	-0.01	0.07
			(0.00)	(0.00)	(0.00)	(0.00)				

Table 4.6: Computational results with varying holding costs.

	Lower Bound		Total Expected Cost				% Gap with DPD			DPD
ϕ	$z(\mathbb{E}\{d\})$	$\mathbb{E}\{z(d)\}$	FC	LAX	LPP	DPD	FC	LAX	LPP	gap
1.2	10,111	10,841	12,578	11,997	13,241	11,645	8.01	3.02	13.71	7.42
			(0.02)	(2.46)	(4.04)	(2.03)				
2.0	13,979	14,771	18,747	16,528	17,238	15,885	18.02	4.05	8.52	7.54
			(0.02)	(2.46)	(4.23)	(2.05)				
3.0	20,489	21,260	29,064	24,119	23,772	22,638	28.39	6.54	5.01	6.48
			(0.02)	(2.46)	(4.24)	(2.04)				
4.0	28,859	29,595	42,278	33,850	32,150	31,131	35.81	8.73	3.27	5.19
			(0.02)	(2.46)	(4.24)	(2.03)				

In Tables 4.6 and 4.7, we vary the holding and the penalty costs. We work with holding costs of the form $h_{jt}^p = \phi^{3-p}\, 1.25^{j-t}$ for all $t \in \mathcal{T}$, $p \in \mathcal{P}$ and $j \in \mathcal{S}_t$ and penalty costs of the form $d_t^p = \beta\, \phi^{3-p}\, 1.25^6$ for all $t \in \mathcal{T}$ and $p \in \mathcal{P}$. We recall that the base case corresponds to $\phi = 2$ and $\beta = 5$. We observe that DPD in general significantly outperforms all other methods. The only exception is the performance of LPP method for a test problem with very large penalty costs. Again, this can be explained by the earlier observation that the LPP policy in general tends to reject more jobs than it would be optimal. However, it appears that for extremely high penalty costs, the LPP policy avoids making this mistake.

Table 4.7: Computational results with varying penalty costs.

	Lower Bound		Total Expected Cost				% Gap with DPD			DPD
β	$z(\mathbb{E}\{d\})$	$\mathbb{E}\{z(d)\}$	FC	LAX	LPP	DPD	FC	LAX	LPP	gap
2.0	12,194	13,074	18,729	14,697	14,944	14,372	30.32	2.26	3.98	9.93
			(0.02)	(2.46)	(8.24)	(2.13)				
4.0	13,384	14,286	18,741	15,918	17,064	15,383	21.83	3.48	10.93	7.68
			(0.02)	(2.46)	(5.63)	(2.06)				
5.0	13,979	14,771	18,747	16,528	17,238	15,885	18.02	4.05	8.52	7.54
			(0.02)	(2.46)	(4.23)	(2.05)				
6.0	14,575	15,195	18,753	17,138	16,813	16,378	14.50	4.64	2.66	7.79
			(0.02)	(2.46)	(2.95)	(2.03)				
8.0	15,413	15,904	18,765	19,107	16,771	16,829	11.50	13.54	-0.34	5.82
			(0.02)	(0.81)	(1.80)	(1.15)				

Finally, in Table 4.8, we vary the parameter corresponding to the initial ca-

pacity occupancy. We conclude that DPD consistently outperforms all benchmark strategies independently of the value of ρ.

Table 4.8: Computational results with varying initial occupancy.

	Lower Bound		Total Expected Cost				% Gap with DPD			DPD
ρ	$z(\mathbb{E}\{d\})$	$\mathbb{E}\{z(d)\}$	FC	LAX	LPP	DPD	FC	LAX	LPP	gap
0.00	11,160	12,925	13,783	13,783	14,997	13,469	2.33	2.33	11.34	4.21
			(0.00)	(0.00)	(2.33)	(0.00)				
0.16	13,801	14,579	18,106	16,347	17,008	15,686	15.43	4.21	8.43	7.59
			(0.02)	(2.31)	(4.02)	(1.86)				
0.32	13,979	14,771	18,747	16,528	17,238	15,885	18.02	4.05	8.52	7.54
			(0.02)	(2.46)	(4.23)	(2.05)				
0.50	15,622	16,382	24,388	18,263	18,799	17,517	39.22	4.26	7.32	6.93
			(0.20)	(3.88)	(5.49)	(3.40)				
0.70	17,694	18,399	31,773	20,292	20,603	19,513	62.83	3.99	5.59	6.05
			(0.70)	(5.35)	(6.77)	(4.83)				

4.8 Conclusion

In this paper, we considered a capacity allocation problem that involves allocating a fixed amount of daily processing capacity to jobs of different priority levels arriving randomly over time. While finding the optimal policy would involve solving a dynamic program with a large number of state variables, we proposed a dynamic programming decomposition idea, which required solving dynamic programs with only one state variable. Computational results indicated that the decomposition policy performs significantly better than alternative policies.

Chapter 5

Summary

In this thesis, we develop dynamic programming decomposition methods for capacity allocation and network revenue management problems. We demonstrate that our approach in general gives rise to two important results. First, the suggested decomposition methods are useful in providing tight bounds on optimal total expected values of the underlying optimization problems. Second, the policies implied by the value functions approximations generated by the decomposition methods significantly outperform a variety of known benchmark strategies.

In Chapter 2, we consider a network revenue management problem in which the airline is assumed to perform capacity allocation and overbooking decisions jointly. We demonstrate that, by concentrating on one flight leg at a time, it is possible to obtain approximate solutions to the dynamic programming formulation of the underlying revenue optimization problem. These approximate solutions are obtained by solving a sequence of single leg revenue optimization problems. Furthermore, we show that a state aggregation technique can be used to obtain tractable high quality solutions to the single leg problems. Overall, our model constructs separable approximations to the value functions, which can be used to make the capacity allocation and overbooking decisions for the whole airline network. Computational experiments indicate that our model performs significantly better than a variety of benchmark strategies from the literature.

In Chapter 3, we consider a network revenue management problem in which the airline is assumed to control prices of its offered itineraries. Specifically, we consider a setting where the probability of observing a request for an itinerary depends on

the prices and the objective is to dynamically adjust the prices so that the total expected revenue is maximized. We construct two different approximations to value functions of the dynamic programming formulation of this pricing problem. Both methods share the idea of decomposing the dynamic programming formulation by the flight legs and obtaining value function approximations by focusing on one flight leg at a time. We show that our methods provide upper bounds on the optimal total expected revenue and these upper bounds are tighter than the one provided by a deterministic linear program commonly used in practice. Our computational experiments yield two important results. First, our methods provide substantial performance improvements over the deterministic linear program. Second, the two methods that we develop have different strengths. In particular, while one method is able to obtain tighter upper bounds, the other one is able to obtain pricing policies that yield higher total expected revenues.

In Chapter 4, we consider a capacity allocation problem with a fixed amount of daily processing capacity. Jobs of different priorities arrive randomly over time and need to be scheduled for processing. The jobs that are waiting to be processed incur a holding cost depending on their priority levels and the length of the scheduled waiting period. The goal is to minimize the total expected cost over a finite planning horizon. We propose an approximation method that decomposes the dynamic programming problem by booking days and solves a sequence of single-day capacity allocation problems. We show that this approach both provides a lower bound on the optimal total expected costs and can be used to make capacity allocation decisions. Computational experiments indicate that our method performs significantly better than a variety of benchmark strategies.

BIBLIOGRAPHY

Adelman, D. (2007), 'Dynamic bid-prices in revenue management', *Operations Research* **55**(4), 647–661.

Beckmann, M. J. (1958), 'Decision and team problems in airline reservations', *Econometrica* **26**(1), 134–145.

Belobaba, P. P. (1987), Air Travel Demand and Airline Seat Inventory Control, PhD thesis, Massachusetts Institute of Technology, Cambridge, MA.

Bertsimas, D. and Popescu, I. (2003), 'Revenue management in a dynamic network environment', *Transportation Science* **37**, 257–277.

Bitran, G. and Caldentey, R. (2003), 'An overview of pricing models for revenue management', *Manufacturing & Service Operations Management* **5**(3), 203–229.

Bront, J. J. M., Mendez-Diaz, I. and Vulcano, G. (2008), 'A column generation algorithm for choice-based network revenue management', *Operations Research* (to appear).

Chatwin, R. E. (1992), 'Multiperiod airline overbooking with a single fare class', *Operations Research* **46**(6), 805–819.

Chatwin, R. E. (1999), 'Continuous-time airline overbooking with time-dependent fares and refunds', *Transportation Science* **33**(2), 182–191.

Coughlan, J. (1999), 'Airline overbooking in the multi-class case', *The Journal of the Operational Research Society* **50**(11), 1098–1103.

Elmaghraby, W. and Keskinocak, P. (2003), 'Dynamic pricing in the presence of inventory considerations: Research overview, current practices, and future directions', *Management Science* **49**(10), 1287–1309.

Erdelyi, A., Kunnumkal, S. and Topaloglu, H. (2009), Using decomposition methods to solve pricing problems in network revenue management, Technical report, Cornell University, School of Operations Research and Information Engineering. Available at `http://legacy.orie.cornell.edu/~huseyin/publications/publications.html`.

Erdelyi, A. and Topaloglu, H. (2009*a*), 'Computing protection level policies for dynamic capacity allocation problems by using stochastic approximation methods', *IIE Transactions* (to appear).

Erdelyi, A. and Topaloglu, H. (2009*b*), 'A dynamic programming decomposition method for making overbooking decisions over an airline network', *INFORMS Journal on Computing* (to appear).

Erdelyi, A. and Topaloglu, H. (2009*c*), 'Separable approximations for joint capacity control and overbooking decisions in network revenue management', *Journal of Revenue and Pricing Management* **8**(1), 3–20.

Feng, Y. and Gallego, G. (1995), 'Optimal starting times for end-of-season sales and optimal stopping times for promotional fares', *Management Science* **41**(8), 1371–1391.

Feng, Y. and Gallego, G. (2000), 'Perishable asset revenue management with markovian time dependent demand intensities', *Management Science* **46**(7), 641–656.

Feng, Y. and Xiao, B. (2000), 'A continuous-time yield management model with multiple prices and reversible price changes', *Management Science* **46**(5), 644–657.

Gallego, G. and van Ryzin, G. (1994), 'Optimal dynamic pricing of inventories with stochastic demand over finite horizons', *Management Science* **40**(8), 999–1020.

Gallego, G. and van Ryzin, G. (1997), 'A multiproduct dynamic pricing problem and its applications to yield management', *Operations Research* **45**(1), 24–41.

Gerchak, Y., Gupta, D. and Henig, M. (1996), 'Reservation planning for elective surgery under uncertain demand for emergency surgery', *Management Science* **42**, 321–334.

Karaesmen, I. and van Ryzin, G. (2004*a*), Coordinating overbooking and capacity control decisions on a network, Technical report, Columbia Business School.

Karaesmen, I. and van Ryzin, G. (2004*b*), 'Overbooking with substitutable inventory classes', *Operations Research* **52**(1), 83–104.

Kleywegt, A. J. (2001*a*), An optimal control problem of dynamic pricing, Technical report, School of Industrial and Systems Engineering, Georgia Institute of Technology.

Kleywegt, A. J. (2001*b*), An optimal control problem of dynamic pricing, Technical report, School of Industrial and Systems Engineering, Georgia Institute of Technology.

Kunnumkal, S. and Topaloglu, H. (2009*a*), 'A new dynamic programming decomposition method for the network revenue management problem with customer choice behavior', *Production and Operations Management* (to appear).

Kunnumkal, S. and Topaloglu, H. (2009*b*), A stochastic approximation algorithm for making pricing decisions in network revenue management problems, Technical report, Cornell University, School of Operations Research and Information

Engineering.

Available at http://legacy.orie.cornell.edu/∼huseyin/publications/publications.html.

Liu, Q. and van Ryzin, G. (2008), 'On the choice-based linear programming model for network revenue management', *Manufacturing & Service Operations Management* **10**(2), 288–310.

Maglaras, C. and Meissner, J. (2006), 'Dynamic pricing strategies for multiproduct revenue management problems', *Manufactuting & Service Operations Management* **8**(2), 136–148.

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

Meissner, J. and Strauss, A. K. (2008), Network revenue management with inventory-sensitive bid prices and customer choice, Technical report, Lancaster University Management School, Department of Management Science,.

Patrick, J., Puterman, M. and Queyranne, M. (2008), 'Dynamic multi-priority patient scheduling for a diagnostic resource', *Operations Research* (to appear).

Powell, W. B. and Cheung, R. K. (1994), 'Stochastic programs over trees with random arc capacities', *Networks* **24**, 161–175.

Subramanian, J., Stidham, S. and Lautenbacher, C. J. (1999), 'Airline yield management with overbooking, cancellations and no-shows', *Transportation Science* **33**(2), 147–167.

Talluri, K. T. and van Ryzin, G. J. (2004), *The Theory and Practice of Revenue Management*, Kluwer Academic Publishers.

Talluri, K. and van Ryzin, G. (1998), 'An analysis of bid-price controls for network revenue management', *Management Science* **44**(11), 1577–1593.

Thompson, H. R. (1961), 'Statistical problems in airline reservation control', *Journal of the Operational Research Society* **12**(3), 167–185.

Topaloglu, H. (2009), 'Using Lagrangian relaxation to compute capacity-dependent bid-prices in network revenue management', *Operations Research* **57**(3), 637–649.

Williamson, E. L. (1992), Airline Network Seat Control, PhD thesis, Massachusetts Institute of Technology, Cambridge, MA.

Zhang, D. and Adelman, D. (2009), 'An approximate dynamic programming approach to network revenue management with customer choice', *Transportation Science* (to appear).

Zhang, D. and Cooper, W. L. (2006), 'Pricing substitutable flights in airline revenue management', *European Journal of Operational Research* (to appear).

Zhao, W. and Zheng, Y.-S. (2000), 'Optimal dynamic pricing for perishable assets with nonhomogenous demand', *Management Science* **46**(3), 375–388.