A Tractable Revenue Management Model for Capacity Allocation and Overbooking over an Airline Network

Sumit Kunnumkal
Indian School of Business, Gachibowli, Hyderabad, 500032, India
sumit_kunnumkal@isb.edu

Huseyin Topaloglu
School of Operations Research and Information Engineering,
Cornell University, Ithaca, New York 14853, USA
topaloglu@orie.cornell.edu

April 20, 2009

Abstract

In this paper, we develop a revenue management model to jointly make the capacity allocation and overbooking decisions over an airline network. The crucial observation behind our model is that if the penalty cost of denying boarding to the reservations were given by a separable function, then the optimality equation for the joint capacity allocation and overbooking problem would decompose by the itineraries. We exploit this observation by building an approximation to the penalty cost that is separable by the numbers of reservations for different itineraries. In this case, we can obtain an approximate solution to the optimality equation by plugging the separable approximation into the boundary condition of the optimality equation. Our computational experiments compare our approach with a standard deterministic linear programming formulation, as well as a recent joint capacity allocation and overbooking model. When compared with the standard deterministic linear programming formulation, our approach can provide significant profit improvements. On the other hand, when compared with the recent joint capacity allocation and overbooking model, our approach can provide similar profit performance with substantially shorter runtimes.

Capacity allocation and overbooking form two important components of network revenue management operations. Capacity allocation deals with the question of what itinerary requests should be accepted given that there is uncertainty about the future itinerary requests, whereas overbooking deals with the question of how many seats in excess of the physically available seat inventory should be sold given that not all reservations show up at the departure time. These two classes of decisions clearly interact. In particular, what itinerary requests should be accepted depends on how much we are willing to overbook, and how much we are willing to overbook depends on what itinerary requests we tend to accept and the likelihood that these accepted itinerary requests show up at the departure time. Nevertheless, despite this clear interaction, the capacity allocation and overbooking decisions are traditionally made in a sequential manner. First, an overbooking model is solved to obtain the overbooking pads that indicate how many seats in excess of the physically available seat inventory should be sold. Following this, a capacity allocation model is solved under the assumption that the capacities on the flight legs are equal to the sum of the physically available seat inventory and the overbooking pads.

In this paper, we present a network revenue management model that jointly makes the capacity allocation and overbooking decisions. We begin by formulating the capacity allocation and overbooking problem as a dynamic program. In this dynamic program, the decisions that we make during the intermediate time periods of the planning horizon are related to whether we should accept or reject the itinerary requests, whereas the decisions that we make at the end of the planning horizon are related to which reservations we should deny boarding. As it is commonly the case for many practical problems, the dynamic programming formulation of the capacity allocation and overbooking problem quickly gets intractable, but this formulation allows us to observe that if the penalty cost of denying boarding to the reservations were given by a separable function, then the problem would decompose by the possible itineraries in the airline network. Therefore, our approach exploits this observation by building a separable approximation to the penalty cost of denying boarding to the reservations at the departure time. In particular, this approximation is separable by the numbers of reservations for different itineraries. In this case, we can obtain an approximate solution to the dynamic program by plugging the separable approximation into the boundary condition. To construct a separable approximation to the penalty cost, we start with a "reasonable" policy to accept and reject the itinerary requests. We simulate the behavior of this policy to get a rough estimate for the numbers of reservations that show up at the departure time. We fix the numbers of reservations that show up at the departure time at these estimates and vary the number of reservations for only one itinerary at a time. By observing how the penalty cost changes as we vary the number of reservations for only one itinerary, we construct a separable approximation to the penalty cost of denying boarding to the reservations.

Although the literature on the capacity allocation problem is quite rich, there are surprisingly few papers that consider the interactions between the capacity allocation and overbooking decisions. Early models consider the capacity allocation and overbooking decisions over a single flight leg rather than an airline network. For example, the papers by Beckmann (1958), Thompson (1961) and Coughlan (1999) develop capacity allocation and overbooking models over a single flight leg that treat the demand from different fare classes as static random variables. These models essentially ignore the temporal dynamics of the arrivals of the itinerary requests and assume that the total demand from different fare classes are observed in a fixed order. The goal is to find the booking limits that prescribe how many seats should be sold to different fare classes. On the other hand, Rothstein (1971, 1974), Chatwin (1992, 1999) and Subramanian, Stidham and Lautenbacher (1999) develop models that capture the temporal dynamics of the itinerary requests more accurately. Of particular interest to us is the paper by Subramanian et al. (1999), which formulates the capacity allocation and overbooking problem over a single flight leg as a dynamic program. This paper shows that if the probability of showing up at the departure time varies by the fare classes, then the dynamic programming formulation of the capacity allocation and overbooking problem over a single flight leg involves a high dimensional state variable and the problem is intractable even over a single flight leg. We contrast this observation with the case where there is no overbooking and all reservations show up at the departure time. In the latter case, the capacity allocation problem over a single flight leg can be formulated as a dynamic program with a scalar state variable. Similar observations can be made by comparing the overbooking models in Rothstein (1971, 1974). Rothstein (1971) considers a single fare class and gives a dynamic programming formulation with a scalar state variable. Rothstein (1974), however, extends this formulation to two fare classes and his extension ends up using a two dimensional state variable.

A traditional method to make the capacity allocation decisions over an airline network is based on solving a deterministic linear program that is formulated under the assumption that the numbers of itinerary requests take on their expected values. This deterministic linear program dates back to Williamson (1992), though her development does not consider overbooking decisions. We refer the reader to Talluri and van Ryzin (2004) for a detailed coverage of the capacity allocation literature that revolves around the deterministic linear program. Bertsimas and Popescu (2003) show how to extend the deterministic linear program to handle overbooking decisions. Karaesmen and van Ryzin (2004*a*) develop a joint capacity allocation and overbooking model by using the deterministic linear program to estimate the revenue from the accepted itinerary requests. Gallego and van Ryzin (1997) show that the capacity allocation policy obtained from a variant of the deterministic linear program is asymptotically optimal as the capacities on the flight legs and the numbers of itinerary requests scale linearly with the same rate. Karaesmen and van Ryzin (2004*b*) describe a capacity allocation and overbooking model that is useful when dealing with multiple flight legs that can serve as substitutes of each other, which is the case for multiple flights in a day that connect the same origin destination pair. Kleywegt (2001) develops a joint pricing and overbooking model over an airline network. This model assumes that the itinerary requests are deterministic and it is solved by using decomposition and duality ideas. Erdelyi and Topaloglu (2008) also use decomposition and duality ideas to decompose the capacity allocation and overbooking problem over an airline network into a sequence of single flight leg problems. As mentioned above, the capacity allocation and overbooking problem over a single flight leg is still a difficult problem and the authors use heuristics to solve their single flight leg problems.

The basic underlying observation that motivates our approach has been used earlier by Erdelyi and Topaloglu (2009) to develop an overbooking model. In particular, Erdelyi and Topaloglu (2009) obtain an approximate solution to the dynamic programming formulation of the problem by using a separable approximation to the penalty cost that is incurred at the departure time. Nevertheless, there are crucial features of our approach that make it significantly more attractive from the viewpoint of practical applications. First, the method that we use to build the separable approximation is different from the one that is used by Erdelyi and Topaloglu (2009). Erdelyi and Topaloglu (2009) use stochastic approximation ideas, whereas we try to estimate the numbers of reservations that show up at the departure time by using a somewhat "reasonable" capacity allocation policy. This "reasonable" capacity allocation policy is based on the aforementioned deterministic linear program and it is clear from our interactions with the industry that revenue managers are more comfortable with using the familiar deterministic linear program rather than a stochastic approximation idea. Second, the approach followed by Erdelyi and Topaloglu (2009) is significantly more complex as it uses tuneable step size parameters whose values need to be set through trial and error. The difficulty of finding good step size parameters for stochastic approximation methods is indeed acknowledged in Chapter 6 of Powell (2007). Third, the runtimes for the model proposed by Erdelyi and Topaloglu (2009) are substantially longer than those for our model. The differences in the runtimes become particularly apparent for larger test problems. Finally, our computational experiments indicate that the profit performances of the two approaches can be comparable. Considering these features, our approach can be visualized as a practically appealing extension of the work by Erdelyi and Topaloglu (2009) to handle industrial size problems.

In this paper, we make the following research contributions. 1) We present a joint capacity allocation and overbooking model over an airline network. Since our model builds on the dynamic programming formulation of the problem, it captures the probabilistic nature of the arrivals of the itinerary requests and the show up decisions of the reservations. 2) We develop a method to construct a separable approximation to the penalty cost that is incurred at the departure time. This approximation, in turn, allows us to solve the dynamic programming formulation of the capacity allocation and overbooking problem approximately. 3) Our computational experiments indicate that the capacity allocation and overbooking decisions made by our model can perform noticeably better than those made by several benchmark methods. This is especially the case when the leg capacities are tight and the penalty cost of denying boarding to a reservation is relatively high.

The rest of the paper is organized as follows. Section 1 formulates the capacity allocation and overbooking problem as a dynamic program. Section 2 presents a deterministic linear programming formulation of the problem and shows that this formulation provides an upper bound on the optimal total expected profit. Section 3 develops our capacity allocation and overbooking model, which uses a separable approximation to the penalty cost that is incurred at the departure time. We construct this approximation by building on the deterministic linear programming formulation. Section 4 provides computational experiments. Section 5 concludes.

1 Problem Formulation

We have a set of flight legs that can be used to satisfy the itinerary requests that arrive randomly over time. At each time period, an itinerary request arrives into the system and we need to decide whether this itinerary request should be accepted or rejected. An accepted itinerary request generates a revenue and becomes a reservation, whereas a rejected itinerary request simply leaves the system. At the departure time of the flight legs, a portion of the reservations show up and we need to decide which of these reservations should be allowed boarding. The objective is to maximize the total expected profit, which is the difference between the expected revenue obtained by accepting the itinerary requests and the expected penalty cost incurred by denying boarding to the reservations.

We count the time periods backwards starting from time period τ to time period 0. The itinerary requests arrive over the time periods $\mathcal{T} = \{\tau, \ldots, 1\}$. Time period 0 corresponds to the departure time of the flight legs, which is when a portion of the reservations show up and we need to decide which of these reservations should be allowed boarding. The set of flight legs is $\mathcal{L}$ and the set of itineraries is $\mathcal{J}$. There is at most one itinerary request at each time period and the probability of having a request for itinerary j at time period t is p_{jt}. Having at most one itinerary request at a time period is not a restrictive assumption as we can choose the time periods to correspond to small enough time intervals so that the probability of having two or more itinerary requests in such a small time interval is negligible. Furthermore, one may argue that having at most one itinerary request at a time period models the problem more accurately as the decision to accept or reject each itinerary request is made one by one, rather than batching a certain number of itinerary requests and jointly deciding which of these itinerary requests should be accepted or rejected. If we accept a request for itinerary j, then we generate a revenue of f_j. There may be more than one itinerary that connects the same origin destination pair with the same connecting flight legs, but these itineraries may be offered at different price levels. This observation allows us to model multiple fare classes. A reservation for itinerary j shows up at the departure time with probability q_j. If we allow boarding to a reservation for itinerary j, then we use a_{ij} units of capacity on flight leg i. We note that a_{ij} may take values greater than one when we model group reservations. In the presence of group reservations, however, an itinerary j corresponds not only to a particular origin destination pair, connecting flight legs and a price level, but also to a particular group size. If we deny boarding to a reservation for itinerary j, then we incur a penalty cost of γ_j. The total capacity available 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 are independent. For notational brevity, we assume that the reservations are not canceled over the time periods $\{\tau, \ldots, 1\}$, but our approach can easily be extended to the case where a reservation for itinerary j is canceled at time period t with probability θ_{jt}, as long as the cancellation decisions at different time periods and the cancellation decisions of different reservations are independent. In this case, the reservations that occur at the earlier time periods naturally have smaller probabilities of showing up at the departure time. Finally, we assume that we do not give refunds to the reservations that do not show up at the departure time, but this assumption is also for notational brevity and it is straightforward to incorporate refunds into our approach.

We let x_{jt} be the number of reservations for itinerary j at the beginning of time period t. Given that the number of reservations for itinerary j at the beginning of time period 0 is x_{j0}, we use $S_{j0}(x_{j0})$ to denote the number of reservations for itinerary j that show up at the departure time. Since the show up decisions of different reservations are independent, $S_{j0}(x_{j0})$ has a binomial distribution with parameters x_{j0} and q_j. We let y_j be the number of reservations for itinerary j that we deny boarding at the departure time. Throughout the paper, we use x_t and $S_0(x_0)$ to respectively denote the vectors $\{x_{jt} : j \in \mathcal{J}\}$ and $\{S_{j0}(x_{j0}) : j \in \mathcal{J}\}$. We also use $s_0 = \{s_{j0} : j \in \mathcal{J}\}$ whenever we need to refer to a realization of $S_0(x_0)$. If the numbers of reservations that show up at the departure time are given by $s_0 = \{s_{j0} : j \in \mathcal{J}\}$, then we can compute the penalty cost of denying boarding to the reservations by solving the problem

$$\begin{aligned} V_0(s_0) = \min \quad & \sum_{j \in \mathcal{J}} \gamma_j \, y_j && && (1) \\ \text{subject to} \quad & \sum_{j \in \mathcal{J}} a_{ij} \, [s_{j0} - y_j] \le c_i && \forall\, i \in \mathcal{L} && (2) \\ & y_j \le s_{j0} && \forall\, j \in \mathcal{J} && (3) \\ & y_j \in \mathbb{Z}_+ && \forall\, j \in \mathcal{J}. && (4) \end{aligned}$$

Constraints (2) in the problem above ensure that the numbers of reservations that we allow boarding do not exceed the leg capacities, whereas constraints (3) ensure that the numbers of reservations that we deny boarding do not exceed the numbers of reservations that show up. In the objective function, we incur a cost of γ_j for each reservation for itinerary j that we deny boarding, but we can replace this objective function with a convex function of the form $\sum_{j \in \mathcal{J}} \Gamma_j(y_j)$, where $\{\Gamma_j(\cdot) : j \in \mathcal{J}\}$ are scalar convex functions capturing the fact that denying an additional reservation gets more costly as the number of denied reservations increases. By virtue of problem (1)-(4), our approach deals not only with the question of how much to overbook, but also with the question of which reservations should be denied boarding. It is also important to note that problem (1)-(4) assumes that we can jointly decide with reservations should be denied boarding throughout the network. This is admittedly an optimistic assumption, but problem (1)-(4) is frequently used to approximate the penalty cost; see Bertsimas and Popescu (2003). Finally, although there is an underlying airline network, problem (1)-(4) does not necessarily have a unimodular constraint matrix and its linear programming relaxation may not yield integer solutions. This is clearly the case when there are group reservations and $\{a_{ij} : i \in \mathcal{L},\ j \in \mathcal{J}\}$ take values greater than one. However, Cooper and Homem de Mello (2007) show that problem (1)-(4) may still not have a unimodular constraint matrix even when $a_{ij} \in \{0, 1\}$ for all $i \in \mathcal{L}$, $j \in \mathcal{J}$.

We use x_t as the state variable at the beginning of time period t. In this case, letting e_j be the $|\mathcal{J}|$ dimensional unit vector with a one in the element corresponding to $j \in \mathcal{J}$, we can find the optimal policy to accept or reject the itinerary request at each time period by computing the value functions $\{v_t(\cdot) : t \in \mathcal{T}\}$ through the optimality equation

$$v_t(x_t) = \sum_{j \in \mathcal{J}} p_{jt} \max\{f_j + v_{t-1}(x_t + e_j), v_{t-1}(x_t)\} + \left\{1 - \sum_{j \in \mathcal{J}} p_{jt}\right\} v_{t-1}(x_t) \tag{5}$$

with the boundary condition that $v_0(x_0) = -\mathbb{E}\{V_0(S_0(x_0))\}$. We note that we negate $V_0(\cdot)$ in the boundary condition since $v_0(\cdot)$ accounts for a profit figure, whereas $V_0(\cdot)$ accounts for a cost figure. If the state of the reservations at the beginning of time period t is given by x_t and we have

$$f_j + v_{t-1}(x_t + e_j) \ge v_{t-1}(x_t), \tag{6}$$

then it is optimal to accept a request for itinerary j at time period t. In the optimality equation in (5), the number of possible values for the state variable x_t increases exponentially with the number of itineraries. Therefore, it can be computationally difficult to solve this optimality equation. In the next two sections, we describe approximate methods to obtain good policies.

2 Deterministic Linear Program

A traditional solution method for the problem described in the previous section involves solving a deterministic linear program that 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. If we let z_j be the number of requests for itinerary j that we plan to accept over the planning horizon and y_j be the number of reservations that we plan to deny boarding, then this linear program has the form

$$\max \quad \sum_{j \in \mathcal{J}} f_j \, z_j - \sum_{j \in \mathcal{J}} \gamma_j \, y_j \tag{7}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{J}} a_{ij} \, [q_j \, z_j - y_j] \le c_i \qquad \forall \, i \in \mathcal{L} \tag{8}$$

$$z_j \le \sum_{t \in \mathcal{T}} p_{jt} \qquad \forall \, j \in \mathcal{J} \tag{9}$$

$$y_j - q_j \, z_j \le 0 \qquad \forall \, j \in \mathcal{J} \tag{10}$$

$$z_j, \, y_j \ge 0 \qquad \forall \, j \in \mathcal{J}. \tag{11}$$

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 (8) and (10) in problem (7)-(11) are respectively analogous to constraints (2) and (3) in problem (1)-(4). Constraints (9) in problem (7)-(11) ensure that the numbers of itinerary requests that we accept do not exceed the expected numbers of itinerary requests. The deterministic linear programming formulation for the network revenue management problem is widely used in practice under the assumption that overbooking is not possible and all reservations show up at the departure time; see Talluri and van Ryzin (1998). Problem (7)-(11) extends this formulation to handle overbooking and no shows. Although this extension is quite intuitive, there are not very many references to problem (7)-(11) in the literature. The earliest reference we are aware of is Bertsimas and Popescu (2003).

There are two important uses of problem (7)-(11). First, problem (7)-(11) can be used to decide whether we should accept or reject the itinerary requests. In particular, letting $\{\hat{\mu}_i : i \in \mathcal{L}\}$ be optimal values of the dual variables associated with constraints (8) in problem (7)-(11), the idea is to use $\hat{\mu}_i$ 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

$$f_j \ge \min \Big\{ q_j \sum_{i \in \mathcal{L}} a_{ij} \, \hat{\mu}_i, \, q_j \, \gamma_j \Big\}, \tag{12}$$

then we accept a request for itinerary j. The decision rule in (12) captures two effects. If the total expected opportunity cost of the capacities consumed by a request for itinerary j is small enough that we have $f_j \ge q_j \sum_{i \in \mathcal{L}} a_{ij} \, \hat{\mu}_i$, then we accept a request for itinerary j. Furthermore, if we have $f_j \ge q_j \, \gamma_j$, then we can, in expectation, generate revenue simply by accepting a request for itinerary j and denying boarding to this reservation at the departure time. We accept a request for itinerary j in this case as well. The decision rule in (12) is also used by Bertsimas and Popescu (2003). In the network revenue management vocabulary, $\hat{\mu}_i$ is referred to as the bid price associated with flight leg i and it is common practice to accept or reject an itinerary request by comparing the revenue from the itinerary request with the opportunity cost of the capacities consumed by the itinerary request. The decision rule in (12) extends this approach to handle overbooking and no shows.

The second use of problem (7)-(11) is that its optimal objective value provides an upper bound on the optimal total expected profit. In other words, letting $\hat{z}_{LP}$ be the optimal objective value of problem (7)-(11) and $\bar{0}$ be the $|\mathcal{J}|$ dimensional vector of zeros, we have $v_\tau(\bar{0}) \leq \hat{z}_{LP}$. This result is widely known when overbooking is not possible and all reservations show up at the departure time. The following proposition gives an extension of this widely known result to cover overbooking and no shows. Erdelyi and Topaloglu (2009) present a proof for Proposition 1.

Proposition 1 *We have $v_\tau(\bar{0}) \leq \hat{z}_{LP}$.*

The upper bound in Proposition 1 can be useful when assessing the optimality gap of a suboptimal decision rule such as the one in (12).

An important advantage of the deterministic linear program is its simplicity. The decision rule in (12) has an intuitive interpretation, which makes it quite appealing to practitioners. On the other hand, an important disadvantage of the deterministic linear program is that it assumes that all random quantities take on their expected values. In particular, if we have $f_j \geq q_j \, \gamma_j$, then it is possible to show that problem (7)-(11) accepts all requests for itinerary j and we have $z_j = \sum_{t \in \mathcal{T}} p_{jt}$ in the optimal solution. This is not a huge drawback noting that if we have $f_j \geq q_j \, \gamma_j$, then it is indeed optimal to accept all requests for itinerary j, since we can generate revenue in expectation simply by accepting a request for itinerary j and denying boarding to this reservation at the departure time. However, if we have $f_j < q_j \, \gamma_j$, then it is also possible to show that problem (7)-(11) does not overbook for itinerary j at all and have $y_j = 0$ in the optimal solution. Therefore, the deterministic linear program does not overbook when it is not trivially optimal to do so. This is an important drawback, which arises solely from the fact that problem (7)-(11) is a deterministic approximation to a problem that actually takes place under uncertainty. However, the deterministic linear program is frequently used in practice due to its simplicity and its empirical performance is generally quite reasonable.

3 Value Function Approximation Strategy

The deterministic linear program uses only the total expected numbers of itinerary requests and it ignores the order in which the itinerary requests are likely to arrive. In this section, we develop a solution method that captures the temporal dynamics of the itinerary requests somewhat more accurately. Our solution method is based on two observations. First, if the penalty cost of denying boarding to the reservations were given by a separable function, then the optimality equation in (5) would decompose by the itineraries and it can be solved by focusing on one itinerary at a time. Second, if we want to build a separable approximation to a function, then we can vary one argument of this function at a time and investigate how the value of the function changes as we vary one argument. Carrying this out for all of the arguments, we can build a separable approximation to the function. We make these observations precise in the following two subsections.

3.1 Decomposing the Optimality Equation

The first observation that our solution method is based on is that if the penalty cost of denying boarding to the reservations were given by a separable function, then the optimality equation in (5) would decompose by the itineraries. More specifically, assume for the moment that the penalty cost of denying boarding to the reservations is given by a separable function of the form

$$V_0(s_0) = \sum_{j \in \mathcal{J}} V_{j0}(s_{j0}),$$

where $\{V_{j0}(\cdot) : j \in \mathcal{J}\}$ are scalar functions. This means that if s_{j0} reservations for itinerary j show up at the departure time, then the penalty cost of denying boarding to the reservations for itinerary j is $V_{j0}(s_{j0})$. Therefore, we can find the optimal policy to accept or reject the requests for itinerary j by computing the value functions through the optimality equation

$$v_{jt}(x_{jt}) = p_{jt} \max\{f_j + v_{j,t-1}(x_{jt}+1), v_{j,t-1}(x_{jt})\} + [1 - p_{jt}]\, v_{j,t-1}(x_{jt}) \tag{13}$$

with the boundary condition that $v_{j0}(x_{j0}) = -\mathbb{E}\{V_{j0}(S_{j0}(x_{j0}))\}$. The next proposition shows that if the penalty cost of denying boarding to the reservations were given by a separable function of the form $V_0(s_0) = \sum_{j \in \mathcal{J}} V_{j0}(s_{j0})$, then we could obtain the solution to the optimality equation in (5) by solving the optimality equation in (13). The practical significance of this result is that the state variable in the optimality equation in (13) is scalar and we can solve this optimality equation efficiently. A similar result is given in Erdelyi and Topaloglu (2009).

Proposition 2 *Assume that $V_0(\cdot)$ in problem* (1)-(4) *is a separable function of the form $V_0(s_0) = \sum_{j \in \mathcal{J}} V_{j0}(s_{j0})$ and let $\{v_t(\cdot) : t \in \mathcal{T}\}$ and $\{v_{jt}(\cdot) : t \in \mathcal{T}\}$ respectively be the solutions to the optimality equations in* (5) *and* (13)*. We have $v_t(x_t) = \sum_{j \in \mathcal{J}} v_{jt}(x_{jt})$ for all $t \in \mathcal{T}$.*

Proof We show the result by a standard induction argument over the time periods. Since we assume that $V_0(\cdot)$ is a separable function of the form $V_0(s_0) = \sum_{j \in \mathcal{J}} V_{j0}(s_{j0})$, the boundary conditions for the optimality equations in (5) and (13) imply that $v_0(x_0) = -\mathbb{E}\{V_0(S_0(x_0))\} = -\sum_{j \in \mathcal{J}} \mathbb{E}\{V_{j0}(S_{j0}(x_{j0}))\} = \sum_{j \in \mathcal{J}} v_{j0}(x_{j0})$ and the result holds for time period 0. Assuming that the result holds for time period $t-1$, we have

$$\begin{aligned} v_t(x_t) &= \sum_{j \in \mathcal{J}} p_{jt} \max\{f_j + v_{t-1}(x_t + e_j) - v_{t-1}(x_t), 0\} + v_{t-1}(x_t) \\ &= \sum_{j \in \mathcal{J}} p_{jt} \max\{f_j + v_{j,t-1}(x_{jt}+1) - v_{j,t-1}(x_{jt}), 0\} + \sum_{j \in \mathcal{J}} v_{j,t-1}(x_{jt}) \\ &= \sum_{j \in \mathcal{J}} p_{jt} \max\{f_j + v_{j,t-1}(x_{jt}+1), v_{j,t-1}(x_{jt})\} + \sum_{j \in \mathcal{J}} [1 - p_{jt}]\, v_{j,t-1}(x_{jt}), \end{aligned}$$

where the first equality follows from (5), the second equality follows from the induction assumption that $v_{t-1}(\cdot)$ is a separable function of the form $v_{t-1}(x_{t-1}) = \sum_{j \in \mathcal{J}} v_{j,t-1}(x_{j,t-1})$ and the third equality follows by adding and subtracting $\sum_{j \in \mathcal{J}} p_{jt}\, v_{j,t-1}(x_{jt})$. The result follows by noting that the last expression in the chain of equalities above is equal to $\sum_{j \in \mathcal{J}} v_{jt}(x_{jt})$ by (13). $\Box$

Therefore, if the penalty cost of denying boarding to the reservations were given by a separable function of the form $V_0(s_0) = \sum_{j \in \mathcal{J}} V_{j0}(s_{j0})$, then we could obtain the solution to the optimality equation in (5) by solving the optimality equation in (13). We note that the penalty cost of denying boarding to the reservations is not separable in general and the assumption in Proposition 2 may hold only for very special problem instances. In any case, we are not interested in directly applying Proposition 2 to obtain the optimal policy for a general network revenue management problem. Instead, the important insight to derive from Proposition 2 is that if we can approximate the penalty cost of denying boarding to the reservations with a separable function of the form $\hat{V}_0(s_0) = \sum_{j \in \mathcal{J}} \hat{V}_{j0}(s_{j0})$, then we can obtain the value function approximations $\{\hat{v}_t(\cdot) : t \in \mathcal{T}\}$ by solving the optimality equation in (5) with the boundary condition that $\hat{v}_0(x_0) = -\mathbb{E}\{\hat{V}_0(S_0(x_0))\}$. Proposition 2 shows that $\{\hat{v}_t(\cdot) : t \in \mathcal{T}\}$ are separable functions of the form $\hat{v}_t(x_t) = \sum_{j \in \mathcal{J}} \hat{v}_{jt}(x_{jt})$ and the scalar functions $\{\hat{v}_{jt}(\cdot) : j \in \mathcal{J},\ t \in \mathcal{T}\}$ can be obtained by solving the optimality equation in (13). Another, perhaps a more subtle, insight to derive from Proposition 2 is that the possibility of overbooking and no shows moves the interactions between the accepted itinerary requests to the departure time of the flight legs. In particular, the capacities committed to the reservations do not play a role until the departure time. If we can break the interactions between the accepted itinerary requests at the departure time by approximating the penalty cost with a separable function of the form $\hat{V}_0(s_0) = \sum_{j \in \mathcal{J}} \hat{V}_{j0}(s_{j0})$, then the dynamic programming formulation of the joint capacity allocation and overbooking problem decomposes by the itineraries at every time period. As a result, the possibility of overbooking and no shows allows us to decompose the dynamic programming formulation in a tractable manner while adding a new layer of realism that has not been thoroughly explored in the existing literature.

The next question is how to build a separable approximation to the penalty cost of denying boarding to the reservations. We dwell on this question in the next subsection.

3.2 Constructing a Separable Approximation to the Penalty Cost

The second observation that our solution method is based on is that if we are interested in building a separable approximation to the penalty cost of denying boarding to the reservations, then we can focus on one itinerary at a time and vary the number of reservations for this itinerary that show up at the departure time. In this way, we can investigate how the penalty cost of denying boarding to the reservations changes as we vary the number of reservations for this itinerary. Carrying this out for all of the itineraries, we can build a separable approximation to the penalty cost. To make this idea precise, assume that the numbers of reservations that show up at the departure time are roughly given by the deterministic numbers $\hat{s}_0 = \{\hat{s}_{j0} : j \in \mathcal{J}\}$. We shortly specify how we choose $\hat{s}_0$. In this case, the scalar function $\hat{V}_{j0}(\cdot\,; \hat{s}_0)$ defined as

$$\hat{V}_{j0}(s_{j0}; \hat{s}_0) = V_0(s_{j0}\, e_j + \hat{s}_0 - \hat{s}_{j0}\, e_j) \tag{14}$$

captures how the penalty cost of denying boarding to the reservations changes as we vary the number of reservations for itinerary j that show up at the departure time. The expression on the right side of (14) can be evaluated by fixing the numbers of reservations that show up at the departure time at $\hat{s}_0 = \{\hat{s}_{j0} : j \in \mathcal{J}\}$ and only varying the number of reservations for itinerary j in problem (1)-(4). This amounts to solving problem (1)-(4) multiple times with different values for s_{j0}. Carrying this out for all of the itineraries, we obtain the separable approximation

$$\hat{V}_0(s_0; \hat{s}_0) = \sum_{j \in \mathcal{J}} \hat{V}_{j0}(s_{j0}; \hat{s}_0). \tag{15}$$

It is important to note that we can still use (14) and (15) even if the penalty cost of denying boarding to the reservations at the departure time is computed in some fashion other than solving problem (1)-(4). As long as there exists a function $V_0(s_0)$ that computes the penalty cost as a function of the numbers of reservations that show up at the departure time, we can use (14) and (15) to construct a separable approximation to the penalty cost.

We now address the question of how we choose $\hat{s}_0$. It is not possible to know the numbers of reservations that show up at the departure time without knowing what policy is used to accept or reject the itinerary requests. We work around this difficulty by simulating the behavior of a "reasonable" policy for multiple replications so that we can get an approximation to the numbers of reservations that show up at the departure time. For this purpose, we utilize the deterministic linear program and simulate the behavior of the decision rule in (12). Therefore, each simulation replication of the decision rule in (12) provides an estimate of the numbers of reservations that show up at the departure time and we denote the estimate from the kth simulation replication by $\hat{s}_0^k = \{\hat{s}_{j0}^k : j \in \mathcal{J}\}$. In this case, we can build the separable approximation $\hat{V}_0(s_0; \hat{s}_0^k) = \sum_{j \in \mathcal{J}} \hat{V}_{j0}(s_{j0}; \hat{s}_0^k)$ from the kth simulation replication as in (14) and (15). Averaging the separable approximations from a total of K simulation replications, we ultimately use $\hat{V}_0(s_0) = \sum_{k=1}^K \hat{V}_0(s_0; \hat{s}_0^k)/K$ as a separable approximation to the penalty cost of denying boarding to the reservations.

3.3 Complete Solution Methodology

We give a description of our complete solution methodology in Figure 1. Step 1 in this figure simulates the behavior of the decision rule in (12) for multiple replications and builds the separable approximation $\hat{V}_0(s_0; \hat{s}_0^k) = \sum_{j \in \mathcal{J}} \hat{V}_{j0}(s_{j0}; \hat{s}_0^k)$ from each one of these replications. In particular, Step 1.a initializes the numbers of reservations to zero. Steps 1.b and 1.c solve the deterministic linear program in (7)-(11) at each time period. An important point is that when solving the deterministic linear program at time period t, we adjust for the current state of the reservations and the expected numbers of the future itinerary requests by replacing the right side of constraints (8) with $\{c_i - \sum_{j \in \mathcal{J}} a_{ij}\, q_j\, x_{jt}^k : i \in \mathcal{L}\}$, the right side of constraints (9) with $\{\sum_{t'=1}^t p_{jt'} : j \in \mathcal{J}\}$ and the right side of constraints (10) with $\{q_j\, x_{jt}^k : j \in \mathcal{J}\}$. We note that $\sum_{j \in \mathcal{J}} a_{ij}\, q_j\, x_{jt}^k$ is the expected capacity consumed on flight leg i by the reservations that have already been accepted by time period t. On the other hand, $\sum_{t'=1}^t p_{jt'}$ is the total expected number of requests for itinerary j starting from time period t until the end of the planning horizon. Steps 1.d, 1.e and 1.f follow the decision rule in (12) to make the decisions at each time period. Steps 1.g and 1.h sample the numbers of reservations that show up at the departure time and construct the separable approximation $\hat{V}_0(s_0; \hat{s}_0^k) = \sum_{j \in \mathcal{J}} \hat{V}_{j0}(s_{j0}; \hat{s}_0^k)$. We note that computing $\hat{V}_{j0}(s_{j0}; \hat{s}_0^k) = V_0(s_{j0}\, e_j + \hat{s}_0^k - \hat{s}_{j0}^k\, e_j)$ as in (14) requires fixing the numbers of reservations that show up at the departure time at $\hat{s}_0^k = \{\hat{s}_{j0}^k : j \in \mathcal{J}\}$ and only varying the number of reservations for itinerary j in problem (1)-(4). Therefore, computing $\hat{V}_{j0}(\,\cdot\,; \hat{s}_0^k)$ requires solving problem (1)-(4) multiple times with different values for s_{j0}. As mentioned above, problem (1)-(4) does not necessarily have a unimodular constraint matrix and its linear programming relaxation may not yield integer solutions. According to our experience, however, commercial optimization codes are quite effective for solving problem (1)-(4). Furthermore, after solving problem (1)-(4) with a particular value for s_{j0}, their warm start abilities allow them to solve problem (1)-(4) with another value for s_{j0} quite efficiently.

After a total of K simulation replications, Step 2 computes the separable approximation $\hat{V}_0(s_0) = \sum_{k=1}^K \hat{V}_0(s_0; \hat{s}_0^k)/K$. In this case, we can obtain the value function approximations $\{\hat{v}_t(\cdot) : t \in \mathcal{T}\}$ by solving the optimality equation in (5) with the boundary condition that $\hat{v}_0(x_0) = -\mathbb{E}\{\hat{V}_0(S_0(x_0))\}$. By Proposition 2, the value function approximations $\{\hat{v}_t(\cdot) : t \in \mathcal{T}\}$ are separable functions of the form $\hat{v}_t(x_t) = \sum_{j \in \mathcal{J}} \hat{v}_{jt}(x_{jt})$ and the scalar functions $\{\hat{v}_{jt}(\cdot) : j \in \mathcal{J},\ t \in \mathcal{T}\}$ can be obtained by solving the optimality equation in (13). We emphasize that our goal in Figure 1 is to obtain a set of value function approximations. Once we have the value function approximations $\{\hat{v}_t(\cdot) : t \in \mathcal{T}\}$, we decide whether we should accept or reject an itinerary request by replacing $\{v_t(\cdot) : t \in \mathcal{T}\}$ in the decision rule in (6) with $\{\hat{v}_t(\cdot) : t \in \mathcal{T}\}$. That is, if the state of the reservations at the beginning of time period t is given by x_t and we have $f_j + \hat{v}_{t-1}(x_t + e_j) \geq \hat{v}_{t-1}(x_t)$, then we accept a request for itinerary j at time period t.

4 Computational Experiments

In this section, we compare the performance of our value function approximation method with the performances of several benchmark strategies. We begin by describing our benchmark strategies and the experimental setup. Following this, we describe our computational results.

4.1 Benchmark Strategies

We test the performances of the following four benchmark strategies.

Value function approximations (VFA) VFA is the solution method that we describe in Section 3, but our practical implementation divides the planning horizon into κ equal segments and refines the value function approximations at the beginning of each segment. In particular, if the state of the reservations at the beginning of segment l is given by $x_{\tau(\kappa-l+1)/\kappa}$, then we carry out the algorithm in Figure 1 by simulating the trajectory of the system over the time periods $\{\tau(\kappa-l+1)/\kappa, \ldots, 1\}$ and starting with the state of the reservations $x_{\tau(\kappa-l+1)/\kappa}$. This provides the value function approximations $\{\hat{v}_t(\cdot) : t = \tau(\kappa-l+1)/\kappa, \ldots, 1\}$. We make the decisions by using these value function approximations on the right side of the decision rule in (6) until we reach the beginning of the next segment and refine the value function approximations again. The goal of our practical implementation is to mimic the traditional airline practice, where the capacity allocation and overbooking policy is periodically refreshed on certain reading days before the departure time.

```
Step 1 Set the iteration counter $k$ to 1.
    Step 1.a Initialize the state of the reservations $x_\tau^k = \{x_{j\tau}^k : j \in \mathcal{J}\}$ by letting $x_{j\tau}^k = 0$ for all $j \in \mathcal{J}$.
             Set the time counter $t$ to $\tau$.
    Step 1.b Solve the deterministic linear program in (7)-(11) after adjusting for the current state of the
             reservations and the expected numbers of the future itinerary requests. In particular, replace
             the right side of constraints (8) with $\{c_i - \sum_{j \in \mathcal{J}} a_{ij}\, q_j\, x_{jt}^k : i \in \mathcal{L}\}$, the right side of constraints
             (9) with $\{\sum_{t'=1}^{t} p_{jt'} : j \in \mathcal{J}\}$ and the right side of constraints (10) with $\{q_j\, x_{jt}^k : j \in \mathcal{J}\}$.
             Solve problem (7)-(11).
    Step 1.c Let $\{\hat{\mu}_{it}^k : i \in \mathcal{L}\}$ be the optimal values of the dual variables associated with constraints (8)
             in the deterministic linear program. In this case, if we follow the decision rule in (12), then
             we accept the requests for the itineraries in the set
                 $\mathcal{J}_t^k = \Big\{ j \in \mathcal{J} : f_j \geq \min \Big\{ q_j \sum_{i \in \mathcal{L}} a_{ij}\, \hat{\mu}_{it}^k,\ q_j\, \gamma_j \Big\} \Big\}.$
    Step 1.d Sample the itinerary request $j_t^k$ at time period $t$ by using the probabilities $\{p_{jt} : j \in \mathcal{J}\}$. We
             note that there may not be any itinerary requests at a particular time period.
    Step 1.e Letting $\mathbf{1}(\cdot)$ be the indicator function, if there is an itinerary request at time period $t$, then
             compute the state of the reservations $x_{t-1}^k = \{x_{j,t-1}^k : j \in \mathcal{J}\}$ at the next time period by
             $x_{t-1}^k = x_t^k + \mathbf{1}(j_t^k \in \mathcal{J}_t^k)\, e_{j_t^k}$. Otherwise, let $x_{t-1}^k = x_t^k$.
    Step 1.f Decrease $t$ by 1. If we have $t \geq 1$, then go to Step 1.b.
    Step 1.g Sample $\hat{s}_{j0}^k$ from the binomial distribution with parameters $x_{j0}^k$ and $q_j$ for all $j \in \mathcal{J}$ so that the
             numbers of reservations that show up at the departure time are given by $\hat{s}_0^k = \{\hat{s}_{j0}^k : j \in \mathcal{J}\}$.
    Step 1.h Compute the scalar functions $\{\hat{V}_{j0}(\cdot\,; \hat{s}_0^k) : j \in \mathcal{J}\}$ as $\hat{V}_{j0}(s_{j0}; \hat{s}_0^k) = V_0(s_{j0}\, e_j + \hat{s}_0^k - \hat{s}_{j0}^k\, e_j)$ and
             let $\hat{V}_0(s_0; \hat{s}_0^k) = \sum_{j \in \mathcal{J}} \hat{V}_{j0}(s_{j0}; \hat{s}_0^k)$.
    Step 1.i Increase $k$ by 1. Letting $K$ be a fixed iteration counter limit, if we have $k \leq K$, then go to
             Step 1.a.
Step 2 Let $\hat{V}_0(s_0) = \sum_{k=1}^{K} \hat{V}_0(s_0; \hat{s}_0^k)/K$. Compute the value function approximations $\{\hat{v}_t(\cdot) : t \in \mathcal{T}\}$ by
       solving the optimality equation in (5) with the boundary condition that $\hat{v}_0(x_0) = -\mathbb{E}\{\hat{V}_0(S_0(x_0))\}$.
```

Figure 1: Building value function approximations.

Deterministic linear program (DLP) DLP is the solution method that we describe in Section 2, and similar to VFA, our practical implementation divides the planning horizon into κ equal segments and resolves problem (7)-(11) at the beginning of each segment to refine the decision rule. If the state of the reservations at the beginning of segment l is given by $x_{\tau(\kappa-l+1)/\kappa}$, then we replace the right side of constraints (8) with $\{c_i - \sum_{j\in\mathcal{J}} a_{ij}\, q_j\, x_{j,\tau(\kappa-l+1)/\kappa} : i \in \mathcal{L}\}$, the right side of constraints (9) with $\{\sum_{t=1}^{\tau(\kappa-l+1)/\kappa} p_{jt} : j \in \mathcal{J}\}$ and the right side of constraints (10) with $\{q_j\, x_{j,\tau(\kappa-l+1)/\kappa} : j \in \mathcal{J}\}$. We solve problem (7)-(11) to obtain the optimal values of the dual variables associated with constraints (8). We use these optimal values in the decision rule in (12) until we resolve problem (7)-(11) at the beginning of the next segment to refine the decision rule again.

Finite differences in the deterministic linear program (FDD) This solution method is due to Bertsimas and Popescu (2003), and similar to DLP, it is based on problem (7)-(11). If the state of the reservations at the beginning of time period t is given by x_t, then FDD uses the optimal objective value of the problem

$$\begin{aligned}
\max \quad & \sum_{j\in\mathcal{J}} f_j\, z_j - \sum_{j\in\mathcal{J}} \gamma_j\, y_j \\
\text{subject to} \quad & \sum_{j\in\mathcal{J}} a_{ij}\,[q_j\, z_j - y_j] \le c_i - \sum_{j\in\mathcal{J}} a_{ij}\, q_j\, x_{jt} && \forall\, i \in \mathcal{L} \\
& z_j \le \sum_{t'=1}^{t} p_{jt'} && \forall\, j \in \mathcal{J} \\
& y_j - q_j\, z_j \le q_j\, x_{jt} && \forall\, j \in \mathcal{J} \\
& z_j,\, y_j \ge 0 && \forall\, j \in \mathcal{J}.
\end{aligned}$$

to approximate the total expected profit in the remaining portion of the planning horizon. In this case, letting $L_t(x_t)$ be the optimal objective value of the problem above, we can use $\{L_t(\cdot) : t \in \mathcal{T}\}$ as approximations to the value functions $\{v_t(\cdot) : t \in \mathcal{T}\}$ in the decision rule in (6). Similar to VFA and DLP, our practical implementation of FDD divides the planning horizon into κ equal segments and refines the decision rule at the beginning of each segment. In particular, if the state of the reservations at the beginning of segment l is given by $x_{\tau(\kappa-l+1)/\kappa}$, then we compute $L_{\tau(\kappa-l+1)/\kappa}(x_{\tau(\kappa-l+1)/\kappa})$ and $L_{\tau(\kappa-l+1)/\kappa}(x_{\tau(\kappa-l+1)/\kappa} + e_j)$ for all $j \in \mathcal{J}$. Following the decision rule in (6), if we have

$$f_j + L_{\tau(\kappa-l+1)/\kappa}(x_{\tau(\kappa-l+1)/\kappa} + e_j) \ge L_{\tau(\kappa-l+1)/\kappa}(x_{\tau(\kappa-l+1)/\kappa}),$$

then we always accept a request for itinerary j until we reach the beginning of the next segment and refine the decision rule again.

Separable approximations using stochastic approximation (SAS) This is the solution method developed by Erdelyi and Topaloglu (2009). SAS is similar to VFA in the sense that it builds a separable approximation to the penalty cost of denying boarding to the reservations and it obtains an approximate solution to the dynamic programming formulation of the problem by using the separable approximation in the boundary condition. SAS utilizes a stochastic approximation method to build the separable approximation, where an initial approximation is iteratively updated by using the sampled trajectories of the system. Being a stochastic approximation method, SAS requires careful tuning for the step size parameters and this method, overall, is more computationally intensive than VFA. We refer the reader to Erdelyi and Topaloglu (2009) for the details of SAS.

4.2 Experimental Setup

In our computational experiments, we consider an airline network that serves N spokes out of a single hub. There are two flight legs associated with each spoke. One of these flight legs is from the hub to the spoke and the other one is from the spoke to the hub. There is a low-fare and a high-fare itinerary that connect each origin destination pair. Therefore, we have $2N$ flight legs and $2N(N+1)$ itineraries, $4N$ of which include one flight leg and $2N(N-1)$ of which include two flight legs. In Figure 2, we show the structure of the airline network for the case where $N = 8$. The fare associated with a high-fare itinerary is ρ times the fare associated with the corresponding low-fare itinerary. The probabilities that a reservation for a low-fare and a high-fare itinerary shows up at the departure are respectively q^l and q^h. These probabilities do not depend on the origin and destination locations of the itinerary. Letting $\bar{f}^l$ and $\bar{f}^h$ respectively be the average fare associated with the low-fare and high-fare itineraries, if j corresponds to a low-fare itinerary, then the penalty cost of denying boarding to a reservation for itinerary j is $\sigma \max\{\bar{f}^l, f_j\}$, whereas if j corresponds to a high-fare itinerary, then the penalty cost of denying boarding to a reservation for itinerary j is $\sigma \max\{\bar{f}^h, f_j\}$. The terms $\bar{f}^l$ and $\bar{f}^h$ ensure that the penalty costs associated with the itineraries within the same fare class are comparable to each other, whereas the term f_j ensures that the penalty cost associated with itinerary j is comparable to the fare associated with itinerary j. It is desirable to have the penalty cost and the fare comparable to each other since the penalty cost may include the refund. Noting that the total expected demand for the capacity on flight leg i is $\sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} a_{ij}\, q_j\, p_{jt}$, we measure the tightness of the leg capacities by

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

We label our test problems by $(\rho, \sigma, q^l, q^h, \alpha)$ and use $\rho \in \{1.5, 3.0\}$, $\sigma \in \{1.5, 3.0, 4.5\}$, $q^l \in \{0.7, 0.9\}$, $q^h \in \{0.7, 0.9\}$ and $\alpha \in \{1.2, 1.6\}$. This provides 48 test problems in our experimental setup. In all of our test problems, we have 8 spokes in the airline network and 360 time periods in the planning horizon. We use $K = 100$ in the algorithm in Figure 1. We use $\kappa = 10$ for VFA, DLP and FDD. A number of preliminary runs showed that refining the value function approximations and the decision rules more frequently does not provide any significant improvement in the performance. Due to the high computational burden for SAS, we refine the value function approximations for SAS five times over the planning horizon. We provide a comparison of runtimes in our computational results. To make sure that we do not put SAS at a disadvantage, we tried refining the value function approximations for SAS more frequently in a few test problems and the performance of SAS did not improve noticeably.

4.3 Computational Results

Our primary computational results are summarized in Tables 1 and 2. In particular, Table 1 shows the computational results for the test problems where the revenue difference between the high-fare and low-fare itineraries is 1.5, whereas Table 2 shows the computational results for the test problems where the same revenue difference is 3.0. The first column in these tables shows the problem characteristics. The second column shows the upper bound on the optimal total expected profit provided by the optimal objective value of problem (7)-(11). The next four columns show the total expected profits obtained by VFA, DLP, FDD and SAS. We estimate these total expected profits by simulating the performances of the different benchmark strategies under multiple demand trajectories. We use common random numbers when simulating the performances of the different benchmark strategies; see Law and Kelton (2000). The seventh column shows the percent gap between the total expected profits obtained by VFA and DLP. This column also includes a "✓" if VFA performs better than DLP, a "$\times$" if DLP performs better than VFA and a "$\odot$" if there does not exist a statistically significant difference between VFA and DLP at 95% significance level. The last two columns do the same thing as the seventh column, but they compare VFA with FDD and SAS.

The results indicate that VFA performs noticeably better than DLP and FDD. The performance gap between VFA and DLP can be as large as 5.2%, whereas the performance gap between VFA and FDD can be as large as 2.7%. In all of the test problems, VFA performs at least as well as DLP and FDD. We also note that the performance gap between VFA and DLP is statistically significant in all of our test problems, whereas the performance gap between VFA and FDD is statistically significant in 41 out of 48 test problems. On the other hand, the performances of VFA and SAS are generally comparable, but there are a few test problems where SAS improves on VFA by about 1.0%. In 41 out of 48 test problems, VFA performs at least as well as SAS.

To give a feel for the problem characteristics that affect the performance gap between the different benchmark strategies, Figure 3 plots the performance gaps between VFA and the other three benchmark strategies for the test problems with $\rho = 3.0$. In particular, the solid, dashed and dotted data series in this figure respectively plot the performance gaps of VFA with DLP, FDD and SAS. We show the problem characteristics in the horizontal axis. The test problems are arranged in such a fashion that two consecutive test problems in the horizontal axis only differ in the tightness of the leg capacities, whereas blocks of eight consecutive test problems share the same penalty costs. As we move from left to right, the penalty costs tend to get larger. Comparing the performance gaps for blocks of eight consecutive test problems indicates that the performance gaps of VFA with DLP and FDD get larger as the penalty costs get larger. For problems with small penalty costs, the regret associated with accepting an "incorrect" itinerary request is small. In this case, all benchmark strategies tend to perform quite well. However, the advantage of using a more sophisticated solution method becomes apparent as the penalty costs get larger and it becomes costly to deny boarding to an "incorrectly" accepted itinerary request. The performance gaps between VFA and SAS are relatively stable, though the three largest performance gaps occur when the penalty costs are large.

Letting $\hat{z}_{LP}$ be the optimal objective value of problem (7)-(11), we note that $\hat{z}_{LP}$ provides an upper bound on the optimal total expected profit. Therefore, we can estimate the optimality gap of a benchmark strategy by comparing the total expected profit obtained by the benchmark strategy with the upper bound on the optimal total expected profit provided by $\hat{z}_{LP}$. However, it is important to emphasize that the optimality gap that we estimate in this fashion is pessimistic since the upper bound on the optimal total expected profit provided by $\hat{z}_{LP}$ can be loose and the actual optimal total expected profit can be significantly smaller than $\hat{z}_{LP}$. In Figure 4, we plot the percent gaps between $\hat{z}_{LP}$ and the total expected profits obtained by the four benchmark strategies for the test problems with $\rho = 3.0$. In particular, the thin solid, thick solid, dashed and dotted data series in this figure respectively plot the percent gaps between $\hat{z}_{LP}$ and the total expected profits obtained by VFA, DLP, FDD and SAS. We show the problem characteristics in the horizontal axis and the arrangement of the test problems in the horizontal axis is the same as the one in Figure 3. Comparing the percent gaps for two consecutive test problems indicates that the estimates of the optimality gaps get larger as the leg capacities get tighter, whereas comparing the percent gaps for blocks of eight consecutive test problems indicates that the estimates of the optimality gaps get larger as the penalty costs get larger. For the test problems with small penalty costs, the estimates of the optimality gaps are on the order of 6.0%. Since our estimates of the optimality gaps are pessimistic, this indicates that it is impossible to improve the performance by more than about 6.0% for these test problems. On the other hand, for the test problems with large penalty costs, the estimates of the optimality gaps can be as large as 18.1%. However, for these test problems, it is impossible to tell which portion of the 18.1% figure is due to the fact that the upper bound on the optimal total expected profit is loose and which portion of the 18.1% figure is due to the fact that the benchmark strategies simply do not perform well.

Figure 5 shows the performance of VFA on test problem $(1.5, 4.5, 0.7, 0.9, 1.2)$ with different values for K in the algorithm in Figure 1. We vary K over $\{1, 2, 5, 10, 25, 50, 100\}$ in this figure. The solid data series plot the total expected profits obtained by VFA with different values for K. The dashed and dotted data series respectively plot the total expected profits obtained by DLP and FDD. It turns out that VFA can provide good performance with a modest number of simulation replications. The performance of VFA gets better as we increase K from 1 to 25 and increasing K beyond 25 does not provide any significant improvement in the performance. It is also interesting to note that the performance of VFA even with $K = 5$ can be better than the performances of DLP and FDD.

Table 3 shows the CPU seconds required by VFA and SAS to construct one set of value function approximations. The portions of the table on the left and right side respectively show the CPU seconds for different numbers of spokes in the airline network and different numbers of time periods in the planning horizon. All of the computational experiments are carried out on a desktop PC running Windows XP with Intel Core 2 Duo 2.8 GHz CPU and 4 GB RAM. Table 3 indicates that the CPU seconds for VFA are substantially shorter than the CPU seconds for SAS. For some test problems, the ratios of the CPU seconds for the two benchmark strategies can be as high as 13 to one and the gaps in the CPU seconds become particularly apparent for the test problems with large numbers of spokes. We note that the sizes of our test problems are quite large when compared with those in the earlier network revenue management literature. Nevertheless, we believe that the difference in the CPU seconds for the two benchmark strategies will get even more significant when one works with industrial size problems involving hundred or so spokes. The CPU seconds required to refine the decision rules used by DLP and FDD are on the order of a fraction of a second. As a result, we do not provide detailed CPU seconds for DLP and FDD.

5 Conclusions

In this paper, we developed a network revenue management model that jointly makes the capacity allocation and overbooking decisions. The crucial observation behind our model is that if the penalty cost of denying boarding to the reservations were given by a separable function, then the optimality equation would decompose by the itineraries. Our approach exploits this observation by building a separable approximation to the penalty cost. In this case, we obtain an approximate solution to the optimality equation by using this separable approximation in the boundary condition. Computational experiments indicate that our model improves on deterministic linear programming methods, and it yields total expected profits that are comparable to those from a recent overbooking model, but with significantly shorter runtimes.

There are several features of our approach that are worth emphasizing. First, unlike the deterministic linear program, our approach is based on a dynamic programming formulation and it addresses the stochastic and dynamic aspects of the capacity allocation and overbooking problem. Second, by virtue of problem (1)-(4), our approach deals not only with the question of how much to overbook, but also with the question of which reservations should be denied boarding. This is in contrast with some of the overbooking literature where the focus is on how much to overbook, but not on which reservations should be denied boarding. Third, although we use problem (1)-(4) to compute the penalty cost of denying boarding to the reservations at the departure time, our approach is not dependent on the form of the penalty cost. As long as there exists a function $V_0(s_0)$ that computes the penalty cost as a function of the numbers of reservations that show up at the departure time, we can use the idea in Section 3.2 to construct a separable approximation to the penalty cost as in (14) and (15).

An important insight to derive from this paper is that if we can build a separable approximation to the penalty cost of denying boarding to the reservations, then the dynamic programming formulation of the capacity allocation and overbooking problem decomposes by the itineraries. The method described in Section 3 represents only one way of building a separable approximation to the penalty cost. An immediate research direction to purse is to devise other methods to build separable approximations.

References

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

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

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.

Cooper, W. L. and Homem de Mello, T. (2007), 'Some decomposition methods for revenue management', *Transportation Science* **41**(3), 332–353.

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

Erdelyi, A. and Topaloglu, H. (2008), A dynamic programming decomposition method for making overbooking decisions over an airline network, 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), 'Separable approximations for joint capacity control and overbooking decisions in network revenue management', *Journal of Revenue and Pricing Management* **8**(1), 3–20.

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

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), An optimal control problem of dynamic pricing, Technical report, School of Industrial and Systems Engineering, Georgia Institute of Technology.

Law, A. L. and Kelton, W. D. (2000), *Simulation Modeling and Analysis*, McGraw-Hill, Boston, MA.

Powell, W. B. (2007), *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, John Wiley & Sons, Hoboken, NJ.

Rothstein, M. (1971), 'An airline overbooking model', *Transportation Science* **5**(2), 180–192.

Rothstein, M. (1974), 'Hotel overbooking as a Markovian sequential decision process', *Decision Sciences* **5**(3), 389–404.

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.

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

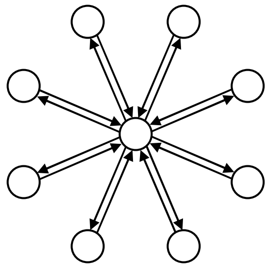

Figure 2: Structure of the airline network for the case where $N = 8$.

Problem	Profit	Total expected profit				Perc. perf. gap with VFA		
$(\rho, \sigma, q^l, q^h, \alpha)$	bound	VFA	DLP	FDD	SAS	DLP	FDD	SAS
$(1.5, 1.5, 0.7, 0.7, 1.2)$	61,767	58,775	58,352	58,397	58,561	0.72 ✓	0.64 ✓	0.36 ✓
$(1.5, 1.5, 0.7, 0.7, 1.6)$	51,129	47,690	47,153	46,994	47,754	1.13 ✓	1.46 ✓	-0.13 ⊙
$(1.5, 1.5, 0.7, 0.9, 1.2)$	59,853	56,107	55,470	55,712	55,998	1.13 ✓	0.70 ✓	0.20 ⊙
$(1.5, 1.5, 0.7, 0.9, 1.6)$	50,989	47,568	46,761	46,783	47,404	1.70 ✓	1.65 ✓	0.35 ✓
$(1.5, 1.5, 0.9, 0.7, 1.2)$	53,197	50,291	49,969	49,981	50,157	0.64 ✓	0.62 ✓	0.27 ✓
$(1.5, 1.5, 0.9, 0.7, 1.6)$	52,009	48,510	48,135	48,139	48,489	0.77 ✓	0.77 ✓	0.04 ⊙
$(1.5, 1.5, 0.9, 0.9, 1.2)$	52,594	49,791	49,498	49,607	49,498	0.59 ✓	0.37 ✓	0.59 ✓
$(1.5, 1.5, 0.9, 0.9, 1.6)$	51,634	47,665	47,349	47,511	47,404	0.66 ✓	0.32 ✓	0.55 ✓
$(1.5, 3.0, 0.7, 0.7, 1.2)$	60,911	55,661	54,087	54,501	55,488	2.83 ✓	2.08 ✓	0.31 ⊙
$(1.5, 3.0, 0.7, 0.7, 1.6)$	49,076	43,258	42,466	42,816	43,030	1.83 ✓	1.02 ✓	0.53 ⊙
$(1.5, 3.0, 0.7, 0.9, 1.2)$	56,330	51,327	49,515	50,076	51,214	3.53 ✓	2.44 ✓	0.22 ⊙
$(1.5, 3.0, 0.7, 0.9, 1.6)$	50,651	44,753	43,392	43,913	44,444	3.04 ✓	1.88 ✓	0.69 ⊙
$(1.5, 3.0, 0.9, 0.7, 1.2)$	53,374	48,719	47,447	48,041	48,674	2.61 ✓	1.39 ✓	0.09 ⊙
$(1.5, 3.0, 0.9, 0.7, 1.6)$	46,531	40,869	40,326	40,681	41,033	1.33 ✓	0.46 ⊙	-0.40 ⊙
$(1.5, 3.0, 0.9, 0.9, 1.2)$	56,484	51,764	50,724	51,091	51,629	2.01 ✓	1.30 ✓	0.26 ⊙
$(1.5, 3.0, 0.9, 0.9, 1.6)$	45,666	40,212	38,781	39,549	40,134	3.56 ✓	1.65 ✓	0.19 ⊙
$(1.5, 4.5, 0.7, 0.7, 1.2)$	58,398	51,211	50,039	50,734	51,740	2.29 ✓	0.93 ⊙	-1.03 ×
$(1.5, 4.5, 0.7, 0.7, 1.6)$	50,020	40,936	39,170	40,326	41,059	4.31 ✓	1.49 ⊙	-0.30 ⊙
$(1.5, 4.5, 0.7, 0.9, 1.2)$	58,432	51,736	49,263	50,355	51,939	4.78 ✓	2.67 ✓	-0.39 ⊙
$(1.5, 4.5, 0.7, 0.9, 1.6)$	47,633	40,244	38,773	40,131	40,744	3.66 ✓	0.28 ⊙	-1.24 ×
$(1.5, 4.5, 0.9, 0.7, 1.2)$	55,252	48,972	47,576	48,384	49,612	2.85 ✓	1.20 ✓	-1.31 ×
$(1.5, 4.5, 0.9, 0.7, 1.6)$	54,161	45,805	43,677	44,974	46,390	4.65 ✓	1.81 ✓	-1.28 ×
$(1.5, 4.5, 0.9, 0.9, 1.2)$	57,396	51,616	49,665	50,768	51,515	3.78 ✓	1.64 ✓	0.20 ⊙
$(1.5, 4.5, 0.9, 0.9, 1.6)$	47,751	40,862	38,731	40,366	40,963	5.21 ✓	1.21 ⊙	-0.25 ⊙

Table 1: Computational results for the test problems with $\rho = 1.5$.

Problem	Profit	Total expected profit				Perc. perf. gap with VFA		
$(\rho, \sigma, q^l, q^h, \alpha)$	bound	VFA	DLP	FDD	SAS	DLP	FDD	SAS
$(3.0, 1.5, 0.7, 0.7, 1.2)$	90,990	86,664	86,140	86,139	86,553	0.60 ✓	0.61 ✓	0.13 ✓
$(3.0, 1.5, 0.7, 0.7, 1.6)$	86,607	82,135	81,199	81,025	82,121	1.14 ✓	1.35 ✓	0.02 ⊙
$(3.0, 1.5, 0.7, 0.9, 1.2)$	85,877	81,094	80,533	80,672	80,867	0.69 ✓	0.52 ✓	0.28 ✓
$(3.0, 1.5, 0.7, 0.9, 1.6)$	78,696	74,075	73,041	73,201	73,729	1.40 ✓	1.18 ✓	0.47 ✓
$(3.0, 1.5, 0.9, 0.7, 1.2)$	93,106	88,762	88,175	88,311	88,553	0.66 ✓	0.51 ✓	0.23 ✓
$(3.0, 1.5, 0.9, 0.7, 1.6)$	91,712	86,466	85,959	86,036	86,408	0.59 ✓	0.50 ✓	0.07 ⊙
$(3.0, 1.5, 0.9, 0.9, 1.2)$	92,189	88,175	87,788	87,960	87,931	0.44 ✓	0.24 ✓	0.28 ✓
$(3.0, 1.5, 0.9, 0.9, 1.6)$	85,868	81,163	80,601	80,701	80,958	0.69 ✓	0.57 ✓	0.25 ✓
$(3.0, 3.0, 0.7, 0.7, 1.2)$	92,330	84,606	83,153	83,728	84,392	1.72 ✓	1.04 ✓	0.25 ⊙
$(3.0, 3.0, 0.7, 0.7, 1.6)$	77,471	69,220	67,701	68,535	69,171	2.19 ✓	0.99 ✓	0.07 ⊙
$(3.0, 3.0, 0.7, 0.9, 1.2)$	90,913	83,906	82,907	83,404	83,666	1.19 ✓	0.60 ✓	0.29 ⊙
$(3.0, 3.0, 0.7, 0.9, 1.6)$	85,067	76,821	74,902	75,777	76,928	2.50 ✓	1.36 ✓	-0.14 ⊙
$(3.0, 3.0, 0.9, 0.7, 1.2)$	94,630	88,214	86,548	87,219	88,133	1.89 ✓	1.13 ✓	0.09 ⊙
$(3.0, 3.0, 0.9, 0.7, 1.6)$	90,467	81,626	80,371	81,261	81,934	1.54 ✓	0.45 ⊙	-0.38 ⊙
$(3.0, 3.0, 0.9, 0.9, 1.2)$	86,274	80,846	79,477	80,181	80,549	1.69 ✓	0.82 ✓	0.37 ✓
$(3.0, 3.0, 0.9, 0.9, 1.6)$	88,528	80,226	78,553	79,916	80,113	2.09 ✓	0.39 ⊙	0.14 ⊙
$(3.0, 4.5, 0.7, 0.7, 1.2)$	93,748	85,254	82,596	83,691	85,683	3.12 ✓	1.83 ✓	-0.50 ⊙
$(3.0, 4.5, 0.7, 0.7, 1.6)$	89,359	77,076	73,841	75,313	77,430	4.20 ✓	2.29 ✓	-0.46 ⊙
$(3.0, 4.5, 0.7, 0.9, 1.2)$	91,364	82,417	79,828	81,010	82,659	3.14 ✓	1.71 ✓	-0.29 ⊙
$(3.0, 4.5, 0.7, 0.9, 1.6)$	77,621	66,842	63,576	65,543	67,206	4.89 ✓	1.94 ✓	-0.54 ⊙
$(3.0, 4.5, 0.9, 0.7, 1.2)$	94,648	87,208	84,737	85,749	87,835	2.83 ✓	1.67 ✓	-0.72 ×
$(3.0, 4.5, 0.9, 0.7, 1.6)$	91,041	80,707	77,998	79,409	81,432	3.36 ✓	1.61 ✓	-0.90 ×
$(3.0, 4.5, 0.9, 0.9, 1.2)$	91,299	84,138	80,982	82,343	84,521	3.75 ✓	2.13 ✓	-0.45 ×
$(3.0, 4.5, 0.9, 0.9, 1.6)$	87,610	77,503	74,812	76,093	77,749	3.47 ✓	1.82 ✓	-0.32 ⊙

Table 2: Computational results for the test problems with $\rho = 3.0$.

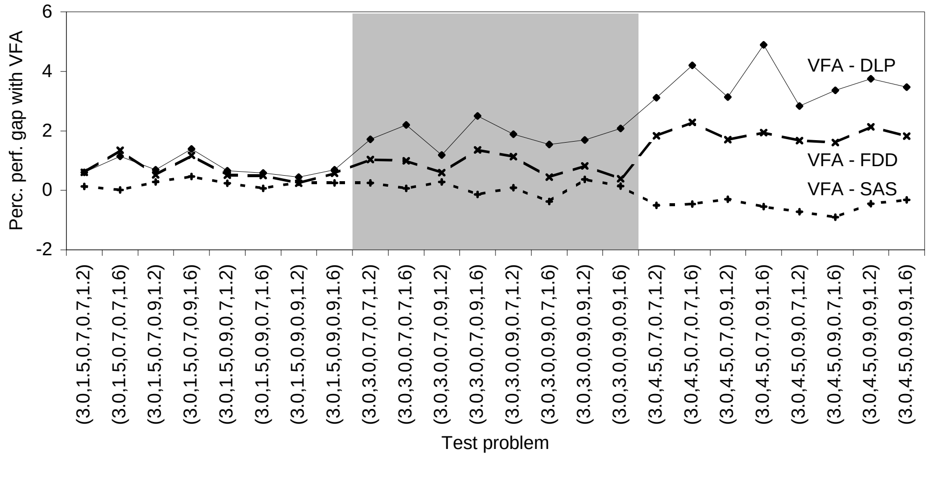

Figure 3: Performance gaps between VFA and the other three benchmark strategies.

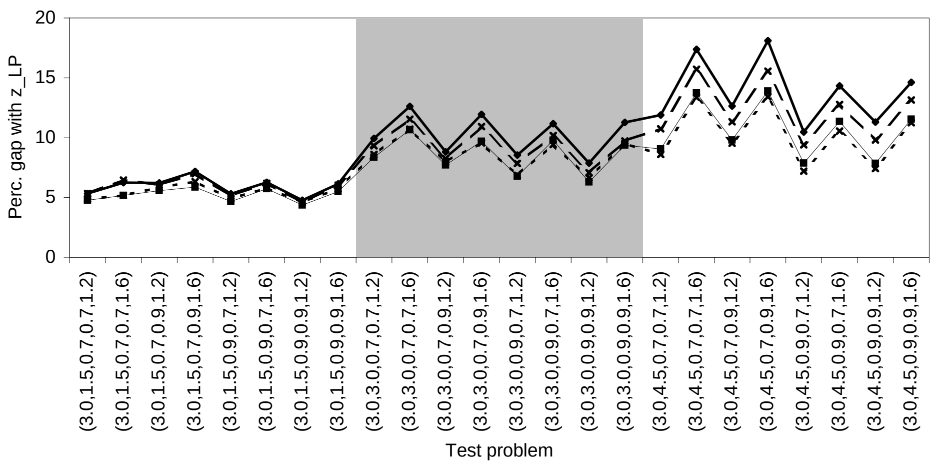

Figure 4: Percent gaps between $\hat{z}_{LP}$ and the total expected profits obtained by the four benchmark strategies.

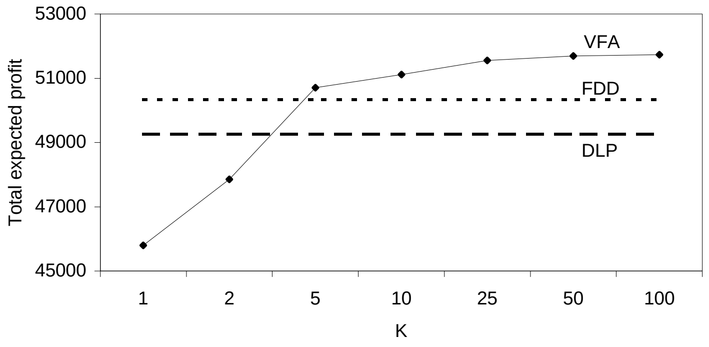

Figure 5: Performance of VFA with different values for K in the algorithm in Figure 1.

No. of spokes	CPU secs. VFA	CPU secs. SAS
2	6	52
4	10	71
8	28	211
16	121	1,185
32	934	12,608

No. of time periods	CPU secs. VFA	CPU secs. SAS
180	20	98
360	28	211
540	39	306
720	47	456
1,440	77	923

Table 3: CPU seconds for VFA and SAS as a function of the number of spokes in the airline network and the number of time periods in the planning horizon.