

# Revenue Management using Sampling-Based Optimization and Markov Decision Processes

William L. Cooper  
Department of Mechanical Engineering  
University of Minnesota, Minneapolis, MN 55455  
billcoop@me.umn.edu

Tito Homem-de-Mello  
Department of Industrial, Welding and Systems Engineering  
Ohio State University, Columbus, OH 43210-1271  
homem-de-mello.1@osu.edu

May 3, 2002

## Abstract

We study revenue management policies that combine aspects of mathematical programming approaches and Markov decision process methods. The policies employ allocations far from the time of departure, and use a more-detailed decision rule close to the time of departure. Working within a Markov decision process framework, we present a formulation that yields such policies, and we describe sampling-based stochastic optimization methods for solving the formulation. Numerical results for two-leg problems suggest that these hybrid policies do perform strongly. By viewing the Markov decision process as a large stochastic program, we derive some structural properties of two-leg problems. We also show that these properties cannot be extended to larger networks.

## 1 Introduction

A wide number of industries face problems in which various products, comprised of combinations of resources, need to be sold to multiple classes of customers over a finite time horizon. Different classes of customers pay different amounts and consume different quantities of resources. Resources that are not sold prior to the end of the time horizon yield no revenue, and consequently “perish.” The collection of methods for dynamically controlling the availability of the different products in order to increase revenues is known as revenue management. These methods are particularly important in the airline industry, where the

resources are seats on flight legs. Revenue management also plays an important role in the hotel, rental car, and broadcasting industries. For the sake of concreteness, we use the terminology of airlines throughout.

For an extensive survey of revenue management literature, see McGill and van Ryzin (1999). Much of the current research in revenue management has focused in two separate areas: Markov decision process (MDP) models for single-leg problems and mathematical programming formulations for multi-leg network problems.

Lee and Hersh (1993) consider an MDP model for managing sales of airline seats to multiple customer classes. They prove a number of structural properties of the problem. Their work has subsequently been extended by numerous authors. Of note are Subramanian et al. (1999), who consider cancellations and no-shows, and Talluri and van Ryzin (2000) who model consumer-choice behavior. Others have considered closely-related continuous-time models; some work of this nature includes Kleywegt and Papastavrou (1998) and Feng and Xiao (2000). With a few exceptions mentioned later, most work within the MDP framework has dealt exclusively with single-leg problems. There appear to be two reasons for this: first, the so-called curse of dimensionality renders MDPs impractical for larger problems; second, it has proved difficult to obtain structural results for larger networks.

A large body of literature that considers mathematical programming approaches to revenue management has also developed. Such work aims to capture the tradeoffs among the many different itineraries that can be built across a network of flights. However, such methods typically do not completely capture the dynamics or the randomness of the booking process. Nevertheless, such techniques do yield computationally-tractable and practically-implementable methods. Williamson (1992) nicely summarizes many of these techniques. For references to both MDP and math programming work, see McGill and van Ryzin (1999).

The primary focus of this paper is a group of “hybrid” policies that combine aspects of both the MDP and the mathematical programming approaches. The goal is to balance the optimal but computationally expensive MDP with inexpensive but sub-optimal mathematical programming-based heuristics. The basic idea is, when far from the time of departure, to use a simple allocation policy, and then to switch to the decision rule specified by an optimal policy when we are close enough to departure. We present a formulation that yields such policies — notably, the formulation explicitly accounts for the fact that such a switch

will take place. If the “switching time” is close enough to departure time, this type of policy will result in substantial savings in terms of computation time and data storage, compared to using the MDP procedure for the entire horizon. Of course, there is a tradeoff, since the longer the optimal policy is in effect, the higher we expect revenues to be. We present numerical results that suggest that these hybrid policies perform strongly. In fact, in a wide range of one- and two-leg problems, our experiments have shown that policies that first use a heuristic and then switch to the MDP-based decision rule close to time of departure perform nearly as well the optimal policy.

These hybrid policies are motivated in part by recent results that suggest that, in a certain sense, relatively crude control methods are good when remaining capacity and expected demand are both large; see Gallego and van Ryzin (1994, 1997) and Cooper (2001). In addition, Subramanian et al. (1999) suggest (p. 163) that near the time of departure, when remaining demand and capacity are typically small, it is important to employ decision rules based upon a detailed, accurate model.

The hybrid formulation we propose falls into the general category of static stochastic optimization problems, where the goal is to minimize (or maximize) the expected value of a given random function. This class of problems has long been studied, and some algorithms have been proposed, notably the stochastic approximation method and its variants. We take a different approach and apply recently-developed techniques that incorporate Monte Carlo sampling into a deterministic optimization method. Our choice is driven by successful applications of this type of method to other problems, and by the fact that sampling is readily accomplished in our model, since underlying distributions are assumed to be known.

A number of authors have employed simulation-based optimization or related methods to solve revenue management problems. Robinson (1995) employs Monte Carlo integration to solve an EMSR formulation. McGill and van Ryzin (1999) describe a Robbins-Monro-like approach (an SA-type method) for a similar formulation. Talluri and van Ryzin (1999) use Monte Carlo estimates to obtain bid-price controls, and Karaesmen and van Ryzin (1998) employ a stochastic-gradient method (another SA variant) to solve an overbooking problem.

Few papers have considered how to blend the MDP and math programming approaches. The work closest to ours is Bertsimas and de Boer (2000). They describe a procedure designed to obtain a nested booking-limit policy. They rely on stochastic-gradient techniques as well

as a carefully devised procedure to estimate derivatives by finite differences. In order to handle larger problems they discuss methods for approximating an MDP value function. Günther (1998) employs related approximation methods, which also allow for a blend of math programming and MDP approaches. One of his key ideas is to use the solutions of math programs as a proxy for the MDP value function when making booking decisions.

There are two notable differences between the present paper and the previous work on blending MDPs and math programming. One is that our solution procedure departs from the usual stochastic approximation setting and, as such, is capable of fully exploiting some optimization techniques devised for deterministic problems. Another difference is that our proposed policies do actually employ the MDP solution for a portion of the booking process, whereas the others use approximations of the MDP value function as ingredients for obtaining other types of policies. By using the MDP solution, we ensure the optimal management of the final portion of the booking process. Because of the computational cost of such policies, our approach seems to be promising for smaller problems that consider two or three flight legs at a time, whereas the former methods are geared towards addressing larger networks. Since airlines do face these smaller problems (which are nevertheless considerably more complex than the single-leg situation), we feel that our work fills an important niche.

We also derive some structural properties for the model under study. By recasting the MDP problem as a multi-stage stochastic linear program and employing results from integer programming, we are able to show that the value function for two-leg problems is *concave*. You (1999) proves componentwise concavity of the value function for two-leg problems; however, our result is stronger, because concavity implies componentwise concavity, but not vice-versa. See, e.g., Lee and Hersh (1993) for results in the single-leg case. We present an example that demonstrates that the value function need not be componentwise concave when there are three legs in the network. Also, we prove that the expected revenue from the hybrid policy is non-increasing as a function of the switch time. Thus, the blend of techniques from MDP theory, stochastic programming, Monte Carlo methods, and integer programming allows us to derive theoretical properties as well as a numerical method.

In summary, the primary contributions of this paper are (1) the description and analysis of hybrid policies within a formal stochastic and dynamic framework, (2) presentation of a formulation for obtaining these types of policies, (3) adaptation of recently-developed

stochastic optimization techniques to solve the formulation, (4) derivation of some structural results on the MDP value function for multiple-leg problems, and (5) discussion of numerical examples that show hybrid policies to be a promising technique.

## 2 Problem Framework

We model the revenue management problem as a finite-horizon discrete-time Markov decision process with time units  $t = 1, \dots, \tau$ . We assume that time moves forward; this means  $t = 1$  is the time at which sales commence and  $t = \tau$  is the last period in which sales can occur. There are  $m$  resources, combinations of which can be used to create  $n$  possible products. Each arriving customer is assumed to belong to one of the  $n$  product classes, where a class- $j$  customer is defined to be someone who requests product  $j$ . We will use the terms demand, request, and arrival interchangeably. (Nevertheless, the general approach that we describe — switching from rough heuristics to optimal decision rules near departure, and using sampling-based optimization to determine the heuristic via a two-stage stochastic programming formulation — could be also be applied to models that do allow for choice behavior such as those in Talluri and van Ryzin 2000. Whether or not such methods would perform well remains to seen.)

Let  $p_{t,j}$  be the probability that a class- $j$  arrival occurs at time  $t$ , and  $D_{t,j}$  be the random demand for class  $j$  at time  $t$ . So,  $P(D_{t,j} = 1) = p_{t,j}$  and  $P(D_{t,j} = 0) = 1 - p_{t,j}$ . Assume there is at most one unit of demand per time period, so that  $P(\exists j_1, j_2 : j_1 \neq j_2, D_{t,j_1} = D_{t,j_2} = 1) = 0$ . The arrivals during different time units are assumed to be independent, and the arrival process is not affected by the acceptance-rejection policy (of course, the policy does affect which customers are accepted and which are rejected). For each  $t$ , we use  $D_t$  to denote the  $n$ -vector with entries  $D_{t,1}, \dots, D_{t,n}$ . We will use a similar convention throughout for vectors; we simply drop the subscript to denote the vector of appropriate dimension. For example  $p_t$  is the  $n$ -vector  $p_t \equiv (p_{t,j})$ . Let  $p_{t,0} = P(D_t = 0)$ .

There is an  $m$ -vector of resources  $c \equiv (c_i)$ , and an  $(m \times n)$ -matrix  $A \equiv (a_{ij})$ . The entry  $c_i > 0$  represents the (integer) amount of resource  $i$  present at the beginning of the time horizon (just prior to  $t = 1$ ), and the entry  $a_{ij} \in \{0, 1\}$  is the amount of resource  $i$  required by a class  $j$  customer. In the airline setting, this model can represent demand for a network of flights that depart within a particular day. In this case, the resources are leg-

cabin combinations (so the vector  $c$  represents the number of seats in leg-cabins throughout the network), and the products are itinerary-fare class combinations. Row  $i$  of  $A$  has 0s in the columns corresponding to fare class-itineraries that do not use leg-cabin  $i$ , and 1s in the columns corresponding to itinerary-fare classes that do use leg-cabin  $i$ .

Whenever a class- $j$  product is sold, we say that we have accepted a class  $j$  customer. In this case he pays his fare  $f_j$ , consumes his resource requirement, and thereafter there are fewer resources available to sell to future customers. On the other hand, we say that a customer's request is rejected if he is not sold his requested product. In this case, no revenue is realized and no resources are consumed.

We will use  $\pi$  to denote a generic policy. If we are willing to accept class  $j$  at time  $t$ , then  $\pi_{t,j} = 1$ . On the other hand, if we are not willing to accept class  $j$  at time  $t$  then  $\pi_{t,j} = 0$ . It is important to note that  $\pi_{t,j}$  is a function mapping the history up to time  $t$  to  $\{0, 1\}$  that may, in general, depend in a complex manner upon what has occurred up to time  $t$ . This means that decisions depend upon more than simply  $t$  and  $j$ . In the interest of readability, we suppress this from our generic notation. The objective is to maximize the expected revenue

$$\max_{\pi \in \Pi} \mathbb{E} \sum_{t=1}^{\tau} f \cdot N_t^{\pi}, \quad (1)$$

where  $N_t^{\pi}$  has entries  $N_{t,j}^{\pi} = D_{t,j} \pi_{t,j}$ . In (1),  $\Pi$  is the set of “admissible” policies — those policies for which the decision at time  $t$  does not depend upon future information unavailable at time  $t$ . Since we are not considering overbooking issues, we also require that  $\pi$  never causes us to accept more customers than we have capacity to handle.

### 3 Markov Decision Process Methods

Under the preceding assumptions, there exists a Markovian optimal policy (see, e.g., Puterman 1994). This means there is a policy that maximizes (1) and that for each  $t$  and  $j$  is a function solely of the remaining capacity vector. Furthermore, one can compute this policy and the corresponding maximum in (1) via the recursive optimality equations

$$v_t(r) = \max_{u \in \{0,1\}^n} \mathbb{E}[f \cdot (uD_t) + v_{t+1}(\phi(r, u, D_t))] \quad \text{subject to} \quad P(\phi(r, u, D_t) \geq 0) = 1, \quad (2)$$

and  $v_{\tau+1}(r) = 0$ , where  $\phi : \mathbb{Z}^m \times \{0, 1\}^n \times \{0, 1\}^n \rightarrow \mathbb{Z}^m$  is defined by  $\phi(r, u, d) = r - A(ud)$ . In the above,  $uD_t$  and  $ud$  should be interpreted as componentwise multiplications; i.e.,  $uD_t$  is

the  $n$ -vector with entries  $u_j \times D_{t,j}$ . For each  $t \in \{1, \dots, \tau + 1\}$ ,  $v_t(r)$  represents the maximum possible expected revenue that can be generated in periods  $t, \dots, \tau$ , provided that  $r$  is the remaining capacity vector just prior to period  $t$ . Moreover,  $v_1(c)$  is the maximum in (1).

In the following,  $\mathbb{I}\{\cdot\}$  is the indicator function,  $e_j$  is the vector with a 1 in the  $j$ -position and 0s elsewhere, and  $A^j$  is the  $j$ -th column of the matrix  $A$ . For each  $t, j$  and  $r$  define

$$\pi_{t,j}^*(r) \equiv \mathbb{I}\{f_j \geq v_{t+1}(r) - v_{t+1}(\phi(r, e_j, e_j))\} \mathbb{I}\{A^j \leq r\}. \quad (3)$$

Then,  $\pi^*$  is a Markovian optimal policy for (1). We omit proof of this fact, since results of this type have been obtained elsewhere; see, e.g., Lautenbacher and Stidham (1999) or Talluri and van Ryzin (1998). In (3), the parenthetical  $r$  indicates that for each  $t$  and  $j$ , the booking decision dictated by  $\pi^*$  depends only upon the remaining capacity  $r$ . Intuitively, (3) states that we should accept a booking request if and only if there is enough capacity and the corresponding fare exceeds the loss in future expected revenue that would be caused by the accepting the request.

Each  $D_{t,j}$  is a Bernoulli random variable with  $P(D_{t,j} = 1) = p_{t,j}$ . Moreover,  $P(D_t = e_j) = p_{t,j}$  and  $\sum_j P(D_t = e_j) + P(D_t = 0) = 1$ , so  $\mathbb{E}D_t = p_t$ . Hence, the constraint set in (2) can be written as

$$\mathcal{U} \equiv \{u \in \{0, 1\}^n : r - A(uD_t) \geq 0 \text{ w.p.1}\} = \{u \in \{0, 1\}^n : r - A^j u_j \geq 0, j = 1, \dots, n\}. \quad (4)$$

So, problem (2) can be rewritten as

$$v_t(r) = \max_{u \in \mathcal{U}} \mathbb{E}[f \cdot (uD_t) + v_{t+1}(\phi(r, u, D_t))]. \quad (5)$$

Formulation (2) — or, more precisely, its rewritten form (5) — can be interpreted as a *multi-stage*  $\{0, 1\}$ -*stochastic linear program* (SLP). Indeed, the objective function in (5) consists of a linear term plus the expected value of the optimal value function of the next stage; also, the constraint set  $\mathcal{U}$  is defined by linear inequalities plus the 0-1 constraints. To make the point more precise, let us reformulate the value function in (5). For each  $t = 1, \dots, \tau$ , let  $D^t \equiv (D_1, \dots, D_t)$  and  $u^t \equiv (u_1, \dots, u_t)$ . Also, define  $f_t \equiv (f\mathbb{E}D_t) = (fp_t)$ . For any fixed  $r$ , define the function  $V_t(u^{t-1}, D^{t-1})$  as

$$\begin{aligned} V_t(u^{t-1}, D^{t-1}) &\equiv \max_{u_t \in \{0, 1\}^n} f_t \cdot u_t + \mathbb{E}[V_{t+1}(u^t, D^t)] \\ &\text{s. to} \\ &A^j u_{t,j} \leq r - A(u_1 D_1) - \dots - A(u_{t-1} D_{t-1}), \quad j = 1, \dots, n, \end{aligned} \quad (6)$$

where  $V_{\tau+1} \equiv 0$ . Let  $\mathbf{0}_{t-1}$  denote the  $(t-1)$ -dimensional vector of zeros. It is clear that  $V_t(\mathbf{0}_{t-1}, \mathbf{0}_{t-1}) = v_t(r)$  as defined in (5). Furthermore, problem (6) is in the standard form of multi-stage stochastic linear programs.

One important property of stochastic linear programs is that, when all underlying random variables have finite support, the model can be written as a (typically large) single linear programming problem. This LP has a block-angular structure which is amenable to decomposition methods such as Benders's. This has led to the development of a class of methods, typically called *L-shaped*. For a discussion of multi-stage methods and models, see Birge and Louveaux (1997), Prékopa (1995) and references therein. Observe that this is exactly the situation in the present case — indeed, each  $D_t$  takes on at most  $n+1$  values. Of course, we must add the 0-1 constraints to the resulting LP.

For each  $t = 1, \dots, \tau$ , let  $\omega^t$  denote the history of randomness up to time  $t$ . Let  $\omega_{k_1, \dots, k_t}^t$  be the specific element in  $\omega^t$  defined by the  $k_1$ th outcome at stage 1, the  $k_2$ th outcome at stage 2, and so on. The “0th” outcome at stage  $t$  means  $D_t = 0$ , whereas the “ $j$ th” outcome at stage  $t$  means  $D_t = e_j$ . Let  $u_t(\omega^{t-1})$  be the vector of decision variables of the optimization problem defined by  $V_t(u^{t-1}, D^{t-1})$  in (6). By convention,  $u_{t,0} \equiv 0$ ,  $A^0 \equiv 0$ , and  $u_1(\omega^0) \equiv u_1$ . Below, all summations are from 0 to  $n$ . We can now rewrite (6) (for  $t = 1$ ) in a single LP:

$$\begin{aligned} \max \quad & (fp_1) \cdot u_1 + \sum_{k_1} p_{1,k_1} [(fp_2) \cdot u_2(\omega_{k_1}^1)] + \sum_{k_1} p_{1,k_1} \sum_{k_2} p_{2,k_2} [(fp_3) \cdot u_3(\omega_{k_1, k_2}^2)] + \dots \\ & + \sum_{k_1} p_{1,k_1} \sum_{k_2} p_{2,k_2} \dots \sum_{k_{\tau-1}} p_{\tau-1, k_{\tau-1}} [(fp_\tau) \cdot u_\tau(\omega_{k_1, k_2, \dots, k_{\tau-1}}^{\tau-1})] \\ \text{s.to} \quad & \hspace{20em} \text{(MSSLP)} \end{aligned}$$

$$\begin{aligned} A^{k_1} u_{1,k_1} &\leq r, \quad k_1 = 0, \dots, n \\ A^{k_1} u_{1,k_1} + A^{k_2} u_{2,k_2}(\omega_{k_1}^1) &\leq r, \quad k_1, k_2 = 0, \dots, n \\ &\vdots \\ A^{k_1} u_{1,k_1} + A^{k_2} u_{2,k_2}(\omega_{k_1}^1) + \dots + A^{k_\tau} u_{\tau, k_\tau}(\omega_{k_1, \dots, k_{\tau-1}}^{\tau-1}) &\leq r, \quad k_1, \dots, k_\tau = 0, \dots, n \quad (7) \\ u_t(\omega^{t-1}) &\in \{0, 1\}^n, \quad t = 1, \dots, \tau. \end{aligned}$$

Notice that each block of inequalities is successively contained in the next; therefore, we can eliminate all but (7). Notice also that the above formulation corresponds to  $v_1(r)$ ; for a

general  $t \leq \tau$ , we simply add the constraints  $u_s(\omega^{s-1}) = 0$ ,  $s = 1, \dots, t-1$ , and set  $p_{s,0} = 1$ ,  $s = 1, \dots, t-1$  (recall that, as discussed above,  $V_t(\mathbf{0}_{t-1}, \mathbf{0}_{t-1}) = v_t(r)$ ).

Although the possibility of reducing the problem to a 0-1 linear program seems appealing, in the present case it does not appear to help in terms of yielding an alternative to the iterative solution of optimality equations. The reason is that the number of possible *scenarios* is huge; since the demands in different periods are assumed to be independent, there are  $(n+1)^{\tau-1}$  overall possibilities. From the above formulation, we can see that we have  $(n+1)^\tau - 1$  decision variables in the LP. Moreover, there are  $mn^\tau$  inequalities, plus the 0-1 constraints. So, we can see that even with moderate values of  $n$  and  $\tau$  it is impractical to solve the LP exactly, even with the aid of decomposition methods. In contrast, with the MDP approach the optimal policy can be found in  $O(mn\tau)$  time, although it requires the storage of the optimal policy at every stage for each value of  $r$  (which takes  $O(n\tau \prod_{i=1}^m c_i)$  space). Nevertheless, the above MSSLP formulation will allow us to derive important properties of the original problem; see section 5.

## 4 Mathematical Programming Methods

The model in section 3 provides *exact* solutions to the revenue management problem under study. As mentioned above, however, both the MDP and the multi-stage SLP approaches are limited to somewhat small problems. Thus, for larger problems, heuristic methods are needed. In this section, we discuss some standard techniques based on mathematical programming. To this end, we need to introduce a bit more notation. For each  $t \in \{0, \dots, \tau\}$ , we define the random variables  $D_{(t),j} = \sum_{s=1}^t D_{s,j}$ , which represent the cumulative demand for class  $j$  up to time  $t$ . We shall use the convention that an empty sum equals zero.

We start with a heuristic based on linear programs, which has been well studied in the literature. In the following,  $\mu$  is an  $n$ -vector where the  $j$ -th entry is the expected class  $j$  demand over  $(0, \tau]$ ;  $\mu_j \equiv \mathbb{E}D_{(\tau),j}$ . The linear program is given by

$$\max_x \{f \cdot x : Ax \leq c, 0 \leq x \leq \mu\}. \quad (8)$$

We will denote by  $x^{\text{LP}}$  an optimal solution of the linear program. The policy  $\pi^{\text{LP}}$  is defined by  $\pi_j^{\text{LP}}(t) \equiv \mathbb{I}\{D_{(t),j} \leq x_j^{\text{LP}}\}$ . The policy  $\pi^{\text{LP}}$  requires solving an LP just once, at the beginning of the booking process, and then using the resulting allocation throughout the entire booking

horizon. In practice, it is common to solve such an LP at various points in time during the booking process, modifying the LP each time to account for the fact that mean future demand and remaining capacity have changed. The derived policy may then be followed only up to the next re-solve point.

The policy  $\pi^{\text{LP}}$  allocates space to the various classes; so, policies of this type are also often called discrete allocation policies. Note that optimal policies are generally not of this form. For certain multi-class, single-leg models, it has been shown that there is an optimal “nested” policy. This means that seats that are available to lower-class demand are always available to higher-class demand. It is also common to obtain other types of policies from (8). One such method is to use leg dual values as “bid prices,” rather than creating allocations. Key references include Williamson (1992) and Talluri and van Ryzin (1998, 1999).

The LP-based method described above, while simple, has a major drawback — its inability to take randomness into account. The allocations are solely based upon the *expected* demand. This suggests that, in order to capture the randomness inherent to the problem, one should replace the LP by something that explicitly models randomness. One such formulation is

$$\max_{x \in \mathbb{Z}_+^n} \mathbb{E} [f \cdot \min\{x, D_{(\tau)}\}] \quad \text{subject to} \quad Ax \leq c. \quad (9)$$

We can use an optimal solution  $x^*$  of (9) to obtain a policy for the MDP — define  $\pi_{t,j}^{\text{SP}} = \mathbb{I}\{D_{(t),j} \leq x_j^*\}$ . Observe that  $\pi^{\text{SP}}$  yields a feasible solution to (1), so the expected revenue from following  $\pi^{\text{SP}}$ , which is given by the optimal objective value of (9), is a lower bound on  $v_1(c)$ . Problem (9) is a special case of a two-stage stochastic program (SP), falling into the category of *simple integer recourse* models. The advantage of such models is that the resulting problem is separable, and therefore can be solved using methods described in, e.g., Birge and Louveaux (1997). A similar formulation is discussed in de Boer et al. (1999). Observe, however, that both the LP and SP procedures are static, since allocations are decided *a priori*. This motivates the study of hybrid methods, as we discuss in the next section.

## 5 A Hybrid Method

We describe now a procedure that combines the optimal MDP and the stochastic programming approaches. The basic idea is to use a simple policy, like the one derived from (9) when we are far from departure, and then switch to the optimal policy when we are close enough

to departure. If the “switching time” is close enough to departure time, this will result in substantial savings in terms of computation time and data storage.

The first step in the construction of our hybrid solution procedure is to specify the time  $\sigma \in \{1, \dots, \tau + 1\}$  at which the switch to the optimal policy occurs. Next, we compute  $v_\sigma(\cdot)$  using the recursions (2). In doing so, we keep track of the optimal policy for the time periods  $\sigma, \dots, \tau$ . Our hybrid formulation is as follows:

$$\max_{x \in \mathbb{Z}_+^n} \mathbb{E} [f \cdot \min\{x, D_{(\sigma-1)}\} + v_\sigma(c - A \min\{x, D_{(\sigma-1)}\})] \quad \text{subject to} \quad Ax \leq c. \quad (10)$$

Observe that the pure MDP approach and the pure stochastic programming approach are both special cases of the above. Take  $\sigma = 1$  to obtain the MDP approach, or take  $\sigma = \tau + 1$  to obtain the stochastic program (9).

From the formulation (10), we obtain the hybrid policy  $\pi^{\text{H}(\sigma)}$  defined by

$$\pi_{t,j}^{\text{H}(\sigma)} = \begin{cases} \mathbb{I} \{D_{(t),j} \leq x_j^{\text{H}(\sigma)}\} & \text{if } t < \sigma \\ \pi_{t,j}^* & \text{if } t \geq \sigma, \end{cases} \quad (11)$$

where  $x^{\text{H}(\sigma)}$  is a maximizer of (10). We shall use  $R^{\text{H}(\sigma)}$  to denote the random revenue that accrues from following the policy (11). Observe that using (11) does indeed yield the optimal objective function value in (10), so  $\mathbb{E} [R^{\text{H}(\sigma)}]$  equals the maximum in (10). As pointed out earlier, optimal policies do not use allocations. It is possible to use the solution from (10) create a policy in which classes with identical resource requirements, but that pay different fares, are nested. However, the formulation does not explicitly account for this. For methods specifically designed for obtaining nested policies, see Bertsimas and de Boer (2000). The basic approach above would also work if we were to assume arrivals to be a continuous-time process such as those in Kleywegt and Papastavrou (1998) or Feng and Xiao (2000).

Formulation (10) constitutes, of course, a considerably more-complicated stochastic optimization problem than the one described in section 4. While (10) can still be viewed as a two-stage procedure, the second stage is no longer a linear program, as it involves the optimal value function resulting from an MDP model. Nevertheless, we can derive some useful properties.

**Proposition 1** *Consider the relaxed value function defined by*

$$v_t^{\text{R}}(r) = \max_{u \in [0,1]^n} \{ \mathbb{E} [f \cdot (uD_t) + v_{t+1}^{\text{R}}(r - A(uD_t))] : P(r - A(uD_t) \geq 0) = 1 \} \quad (12)$$

with  $v_{\tau+1}^R(\cdot) \equiv 0$  (notice that  $u$  is no longer required to be integer). Then,  $v_t^R(\cdot)$  is concave and piecewise linear. Moreover, both  $v_t^R(\cdot)$  and  $v_t(\cdot)$  are nondecreasing and superadditive, i.e.,  $v_t(r_1) \geq v_t(r_2)$  if  $r_1 \geq r_2$  and  $v_t(r_1) + v_t(r_2) \leq v_t(r_1 + r_2)$ .

**Proof.** The result can be proved by (backwards) induction, by noticing that (12) has concave objective function and a convex feasibility set [we can rewrite the constraint set of (12) similar to (4)]. Indeed, using that argument, concavity follows even when  $D_t$  has continuous distribution. Alternatively, we can use the MSSLP formulation discussed in section 3. It is clear that  $v_t^R(r)$  is the optimal value of an LP as a function of the right-hand side. Therefore,  $v_t^R(r)$  is concave and piecewise linear. The monotonicity and superadditivity properties follow from the fact that  $v_t^R(r)$  and  $v_t(r)$  are optimal value functions of (integer) LPs. ■

The above proposition shows that the relaxed function  $v_t^R(r)$  is indeed concave. Our goal now is to show that  $v_t(r)$  and  $v_t^R(r)$  are directly related to each other. In order to do so, we shall first recall the notion of *total unimodularity*. An  $m \times n$  matrix  $Q$  is totally unimodular (TU) if the determinant of each square submatrix of  $Q$  is 0, 1, or  $-1$ . TU matrices are important because if  $Q$  is TU, then the set  $\mathcal{P} = \{y \in \mathbb{R}_+^n : Qy \leq b\}$  is integral for all  $b \in \mathbb{Z}^m$  for which it is nonempty (Schrijver 1986). “Integral” means that all vertices of the polyhedron defined by  $\mathcal{P}$  have integer coordinates. The theorem below shows the connection between these concepts and our model, in the particular case of two-leg problems.

**Theorem 2** Consider the function  $v_t^R(r)$  defined in (12). Suppose that the resource consumption matrix  $A$  is a  $\{0, 1\}$ -matrix with only two rows. Then,  $v_t(r) = v_t^R(r)$  for all  $r \in \mathbb{Z}_+^n$ . In that sense, the value function  $v_t(r)$  is concave.

**Proof.** We use again the MSSLP formulation discussed in section 3. By properly arranging the variables, we see that the matrix of constraints in (7) has the following recursive form:

$$B_\tau = \begin{pmatrix} \tilde{B}_{\tau-1}^0 & & & \\ & \tilde{B}_{\tau-1}^1 & & \\ & & \ddots & \\ & & & \tilde{B}_{\tau-1}^n \end{pmatrix}. \quad (13)$$

In the above,  $\tilde{B}_t^i$  is defined as  $[\tilde{A}_t^i \ B_t]$ , where  $\tilde{A}_t^i$  is a column vector formed by multiple replications of  $A^i$ , the  $i$ th column of matrix  $A$  (recall that, by convention,  $A^0$  is a vector of

zeros). The base matrix  $B_1$  is defined as

$$B_1 \equiv \begin{pmatrix} A^0 & & & \\ & A^1 & & \\ & & \ddots & \\ & & & A^n \end{pmatrix}. \quad (14)$$

The matrix  $B_\tau$  has two fundamental properties, which can be verified directly from (13):

(I) Each column of  $B_\tau$  is of the form

$$\begin{pmatrix} 0 \\ \tilde{A}_t^i \\ 0 \end{pmatrix} \quad (15)$$

for some  $i$  and some  $t$ . Notice that, by assumption,  $A$  has only two rows. That means that each column of  $A$  is one of the following vectors:  $a_0 \equiv [0 \ 0]'$ ,  $a_1 \equiv [1 \ 0]'$ ,  $a_2 \equiv [0 \ 1]'$  or  $a_3 \equiv [1 \ 1]'$ . Therefore, each column of  $B_\tau$  is formed by successive replications of some  $a_j$ , with an even number of zeros both above and below  $\tilde{A}_t^i$ .

(II) Let  $b$  be an arbitrary column of  $B_\tau$ . Let  $\tilde{A}_t^i$  be the vector in that column (cf. (15)), and let  $\ell$  denote the row corresponding to the first element of  $\tilde{A}_t^i$ . Then, the submatrix formed by columns  $1, \dots, b-1$  and rows  $\ell, \dots, s$  (where  $s$  is the total number of rows in  $B_\tau$ ) has *identical* blocks of two rows each.

If the matrix of constraints in (7) — called  $B_\tau$  above — is TU, then the solution of the corresponding linear problem, with the integrality constraints relaxed, coincides with the solution of MSSLP whenever the right-hand side is integral. In other words,  $v_1(r) = v_1^R(r)$  whenever  $r \in \mathbb{Z}_+^n$ . In fact, this property can be extended to  $v_t(r)$  and  $v_t^R(r)$  for a general  $t \leq \tau$  — just notice that, as remarked in the paragraph following the MSSLP formulation, the constraint matrix  $B_\tau^t$  for the problem corresponding to  $v_t$  is formed by  $B_\tau$  and an identity matrix. Thus,  $B_\tau^t$  is TU if  $B_\tau$  is TU.

It remains to prove that  $B_\tau$  is indeed TU. To do so, we shall show by induction that the determinant of each square submatrix of  $B_\tau$  is 0, 1 or  $-1$ . It is easy to see that, because  $B_\tau$  has only  $\{0, 1\}$ -entries, the determinant of any  $2 \times 2$  square submatrix of  $B_\tau$  is 0, 1 or  $-1$ . Suppose now this is true for any square submatrix of order  $k$ . We want to show the property is valid for any square submatrix of order  $k+1$ .

Let  $M$  be a square submatrix of  $B_\tau$  of order  $k + 1$ . Consider the last column of  $M$ , call it  $M^{k+1}$ . By property (I) above,  $M^{k+1}$  has the form (15). Notice that, if  $M^{k+1}$  contains more than one replication of some column  $A^i$ , then by property (II)  $M$  contains two identical rows and so the determinant of  $M$  is zero. Thus, we only need to consider the case where  $M^{k+1}$  is formed by one of the vectors  $a_j$  defined above and zeros in the remaining components.

If  $M^{k+1}$  contains  $a_0$ , then obviously  $\det(M) = 0$ . If  $M^{k+1}$  contains  $a_1$  or  $a_2$ , then we have  $\det(M) = (\pm 1) \det(M')$ , where  $M'$  is a matrix of order  $k$ . By the induction hypothesis,  $\det(M') \in \{-1, 0, 1\}$  and thus  $\det(M) \in \{-1, 0, 1\}$ .

Consider now the case where  $M^{k+1}$  contains the vector  $a_3$ . Then, by properties (I) and (II) above, either  $M$  contains duplicate rows (in which case  $\det(M) = 0$ ), or  $M$  has one of the following forms:

$$(i) \begin{pmatrix} C & 0 & 0 \\ \bar{Y} & \bar{a}_j & 0 \\ Y & a_j & a_3 \\ D & 0 & 0 \end{pmatrix} \quad (ii) \begin{pmatrix} C & 0 & 0 \\ Y & a_3 & a_3 \\ D & 0 & 0 \end{pmatrix} \quad (iii) \begin{pmatrix} C & 0 & 0 \\ \bar{Y} & \bar{a}_j & 0 \\ E & 0 & 0 \\ Y & 0 & a_3 \\ D & 0 & 0 \end{pmatrix} \quad (iv) \begin{pmatrix} C & 0 & 0 \\ Y & a_3 & 0 \\ E & 0 & 0 \\ Y & 0 & a_3 \\ D & 0 & 0 \end{pmatrix}.$$

In the above,  $C$ ,  $D$ , and  $E$  are certain (possibly empty) matrices with  $k - 1$  columns,  $Y$  is a  $2 \times (k - 1)$  matrix,  $\bar{a}_j$  denotes either the vector  $a_j$  or a subset of it (including the empty subset), and  $\bar{Y}$  denotes the part of  $Y$  corresponding to  $\bar{a}_j$  (cf. property (II) above). Also, the 0s denote column vectors of zeros of appropriate size. Moreover, in case (i) the index  $j$  ranges from 0 to 3, whereas in case (iii) the index  $j$  ranges from 0 to 2.

In all four cases, we have  $\det(M) = \pm[\det(M_1) - \det(M_2)]$ , where  $M_1$  and  $M_2$  are  $k \times k$  matrices formed from  $M$  by deleting column  $k + 1$  and deleting the row corresponding, respectively, to the first and second non-zero entry of  $M^{k+1}$ .

In case (i), it is clear that, if  $\bar{a}_j$  is non-empty, then either  $M_1$  or  $M_2$  (or both) has two identical rows; thus, we have either  $\det(M_1)$  or  $\det(M_2)$  is zero and thus, by the induction hypothesis,  $\det(M) = (\pm 1) \det(M_i) \in \{-1, 0, 1\}$  for some  $i = 1, 2$ . If  $\bar{a}_j$  is empty and  $j \neq 3$  (otherwise we are in case (ii)) then column  $k$  has only one nonzero entry and thus either  $\det(M_1)$  or  $\det(M_2)$  is zero. As before, this implies that  $\det(M) \in \{-1, 0, 1\}$ . In case (ii) the matrix  $M$  has two identical columns and thus  $\det(M) = 0$ .

For case (iii), let  $M'$  be the matrix obtained from  $M$  by swapping columns  $k$  and  $k + 1$ .

Then,  $\det(M) = -\det(M')$ . Since  $M'$  has at most one nonzero entry in the last column, it follows from the induction hypothesis that  $\det(M') \in \{-1, 0, 1\}$  and thus  $\det(M) \in \{-1, 0, 1\}$ .

Finally, consider case (iv). For  $i = 1, 2; j = 1, 2$ , let  $M_{ij}$  denote the  $(k-1) \times (k-1)$  matrix obtained from  $M_i$  by deleting the  $k$ -th column and the row corresponding to the  $j$ -th non-zero entry of column  $k$ . Observe that  $M_{11}$  and  $M_{22}$  both contain duplicate rows, so  $\det(M_{11}) = \det(M_{22}) = 0$ . Furthermore,  $\det(M_{12}) = -\det(M_{21})$ , because  $M_{21}$  is obtained from  $M_{12}$  by exchanging two rows. Therefore,  $\det(M_1) = \pm[0 - \det(M_{12})]$  and  $\det(M_2) = \pm[\det(M_{21}) - 0]$ , and the sign  $\pm$  must be the same in both cases. Therefore  $\det(M_1) - \det(M_2) = 0$ , and hence  $\det(M) = 0$ . ■

The above result shows that, in case of *two-leg* problems with single-seat bookings, the functions  $v_t^R(r)$  and  $v_t(r)$  coincide when  $r \in \mathbb{Z}_+^n$ . This is an important property, as it implies (from Proposition 1) that  $v_t(r)$  is, in a sense, concave. A slightly weaker result can be found in Theorem 3.3 of You (1999), which states that the value function for two-leg flights is componentwise concave. In addition, our proof, based upon the formulation of the problem as a MSSLP, is of independent interest because it uses completely different mathematical techniques. Incidentally, our argument can be easily adapted to yield an alternative proof of concavity of one-leg problems — just notice that, in such case, the vectors  $a_0, \dots, a_3$  defined in the proof of Theorem 2 are replaced by  $a_0 = 0, a_1 = 1$ . The proof then becomes much easier, since there is no need to analyze the case where a column has two nonzero entries.

It is natural to ask whether Theorem 2 is valid for problems with more than two legs. Unfortunately, the answer is negative — indeed, the proof of Theorem 2 hinges on the fact that the matrix  $A$  has only two rows. Example 3 shows that for a three-leg problem, the value function need not be componentwise concave, and consequently need not be concave.

**Example 3** The following deterministic example shows that for networks with three or more legs, the MDP value function is, in general, not concave. Suppose that the three legs are labeled 1, 2, and 3. There are three customer classes, also labeled 1, 2, and 3. Class 1 requires a seat on both legs 1 and 2, class 2 requires a seat on legs 1 and 3, and class 3 requires a seat on both legs 2 and 3. Therefore, the resource consumption matrix is composed of three columns:  $[1 \ 1 \ 0]'$ ,  $[1 \ 0 \ 1]'$ , and  $[0 \ 1 \ 1]'$ . This example describes a situation where there are three cities; A, B, and C so that leg 1 goes from A to B, leg 2 goes from B to C, and leg 3 goes from C to A. Class 1 flies from A to C via B, class 2 flies from C to B via A, and

class 3 flies from B to A via C. Hence, there is a cycle, of sorts, in the network. (Note we are assuming there is no leg from, say, B to A.)

Assume that each customer pays a fare of \$100. Suppose that  $\tau = 6$  and that the arrival process is deterministic with a class 1 arrival in periods 1 and 2, a class 2 arrival in periods 3 and 4, and a class 3 arrival in periods 5 and 6. Recall that  $v_1(r_1, r_2, r_3)$  denotes the maximum expected revenue from  $t = 1$  onward, when remaining capacity on legs 1, 2, and 3 is respectively  $r_1$ ,  $r_2$ , and  $r_3$ . Simple calculations now show that  $v_1(2, 2, 0) = 200$ ,  $v_1(2, 2, 1) = 200$ , and  $v_1(2, 2, 2) = 300$ . Therefore, the value function is not concave. Furthermore, the value function also does not satisfy the weaker condition of componentwise concavity.

The failure of componentwise concavity for networks with three or more legs is significant. Without this, one cannot show the existence of threshold-type optimal policies for the MDP. Such threshold-type optimal policies can be employed to save computational effort and storage space for one- and two-leg problems, but evidently, they may not exist for larger networks without imposition of additional assumptions. Notice also that, even when  $v_t$  is concave, such property does not imply the concavity of the objective function in (10), because the term  $v_\sigma(c - A \min\{x, D_{(\sigma-1)}\})$  in (10) consists of a composition of  $v_\sigma$  with a componentwise-convex mapping. Nevertheless, we can derive heuristic solution methods, described in the next section.

A closer look at Example 3 shows that the network discussed there has a certain type of “cycle.” It appears that the existence of such cycles is related to concavity of the value function  $v_t$  — indeed, we have not been able to find examples of acyclic networks where  $v_t$  is not concave. However, it is possible to construct simple examples of acyclic three-leg networks where the matrix of constraints  $B_\tau$  in (7) fails to be TU. Thus, the technique used in the proof of Theorem 2 — using total unimodularity of the constraint matrix to show that there exists an integral optimal solution — cannot be extended to more general cases. Nevertheless, we can apply that approach to the LP formulation discussed in section 4. To do so, we first need to recall the concept of a *network matrix* as described in, e.g., Schrijver (1986). Let  $D = (V, E)$  be a directed graph and let  $T = (V, E_0)$  be a *directed spanning tree* on  $V$ . Let  $M$  be an  $E_0 \times E$  matrix defined as follows. For each  $a = (u, v) \in E$  and each  $e \in E_0$ ,  $M_{e,a}$  is set to: +1 if the unique  $u$ - $v$ -path in  $T$  passes forward through  $e$ ; -1 if the unique  $u$ - $v$ -path in  $T$  passes backward through  $e$ ; and 0 if the unique  $u$ - $v$ -path in  $T$  does not

pass through  $e$ . Then  $M$  is the network matrix defined by  $D$  and  $T$ . A key (and well-known) property of network matrices is that they are totally unimodular.

The proposition below shows that, for acyclic networks, a slightly modified version of the LP problem (8) yields integral solutions.

**Proposition 4** *Consider the resource consumption matrix  $A$  defined in the airline context in section 1. Consider a directed graph  $T$ , where each arc corresponds to a leg of the flight network, and the nodes are the cities. Suppose that  $T$  does not have any cycles (directed or not). Then,  $A$  is a network matrix. In particular, this implies that the LP problem*

$$\max_x \{f \cdot x : Ax \leq c, 0 \leq x \leq \lfloor \mu \rfloor\}$$

*has an integer optimal solution, where  $\mu_j \equiv \mathbb{E}D_{(\tau),j}$ .*

**Proof.** Consider a directed graph  $G$  with the same nodes as  $T$ , where each arc  $(o, d)$  of  $G$  corresponds to an particular customer class whose itinerary begins at  $o$  and ends at  $d$  (notice that we allow multiple arcs between  $o$  and  $d$  in  $G$  as there are possibly multiple fare classes with the same resource requirements). Recall that, by assumption, each class takes one seat on each leg used in its itinerary — so  $A$  has an entry 1 in the  $(\ell, (o, d))$ -position for each leg  $\ell$  used by class  $(o, d)$ . The legs used by this class correspond to the directed path from  $o$  to  $d$  in  $T$ . Because there are no cycles, it follows that  $T$  is a tree, and without loss of generality we may also assume it is a spanning tree, since otherwise the problem can be decomposed. Therefore, the network matrix defined by  $G$  and  $T$  coincides with  $A$ .

The second assertion of the theorem follows from the fact that the polyhedron  $\{Ax \leq c, 0 \leq x \leq \lfloor \mu \rfloor\}$  is integral whenever  $A$  is totally unimodular. ■

We conclude this section with the following result, which provides a quantitative relationship among the hybrid policies by describing how the expected revenue behaves as a function of the switch time. The result is intuitive — the longer the optimal MDP policy is in effect (and the higher the computational cost), the higher we expect revenue to be. The importance of Theorem 5 lies in the fact that it allows for consideration of an *optimal* switching time based on the total computational budget.

**Theorem 5** *The expected revenue  $\mathbb{E}[R^{\text{H}(\sigma)}]$  is non-increasing as a function of switch time  $\sigma$ .*

**Proof.** We must show that  $\mathbb{E}[R^{\text{H}(\sigma_1)}] \geq \mathbb{E}[R^{\text{H}(\sigma_2)}]$  for any  $\sigma_1 \leq \sigma_2$ . To do so, consider the policy  $\tilde{\pi}$  defined by  $\tilde{\pi}_{t,j} = \mathbb{I}\{D_{(t),j} \leq x^{\text{H}(\sigma_2)}\}$  if  $t < \sigma_1$  and  $\tilde{\pi}_{t,j} = \pi_{t,j}^*$  if  $t \geq \sigma_1$ . In words,  $\tilde{\pi}$  tells us to follow the allocation prescribed for switch time  $\sigma_2$ , but to implement the switch to the optimal policy earlier, at time  $\sigma_1$ . We shall denote the revenue from policy  $\tilde{\pi}$  by  $\tilde{R}$ . From the definition of  $x^{\text{H}(\sigma_1)}$ , we have  $\mathbb{E}R^{\text{H}(\sigma_1)} \geq \mathbb{E}\tilde{R}$ . Therefore, to prove the proposition, it suffices to show that  $\mathbb{E}\tilde{R} \geq \mathbb{E}R^{\text{H}(\sigma_2)}$ . Keeping in mind that  $D^{\sigma_1-1} = (D_1, \dots, D_{\sigma_1-1})$ , in order to show  $\mathbb{E}\tilde{R} \geq \mathbb{E}R^{\text{H}(\sigma_2)}$ , it is sufficient to demonstrate that

$$\mathbb{E}[\tilde{R}|D^{\sigma_1-1} = d^{\sigma_1-1}] \geq \mathbb{E}[R^{\text{H}(\sigma_2)}|D^{\sigma_1-1} = d^{\sigma_1-1}] \quad (16)$$

for any  $d^{\sigma_1-1} = (d_1, \dots, d_{\sigma_1-1})$ . To this end, observe that

$$\tilde{R} = f \cdot \min\{x^{\text{H}(\sigma_2)}, D_{(\sigma_1-1)}\} + \sum_{t=\sigma_1}^{\tau} f \cdot N_t^{\tilde{\pi}} \quad (17)$$

$$R^{\text{H}(\sigma_2)} = f \cdot \min\{x^{\text{H}(\sigma_2)}, D_{(\sigma_1-1)}\} + \sum_{t=\sigma_1}^{\tau} f \cdot N_t^{\pi^{\text{H}(\sigma_2)}}. \quad (18)$$

Fix  $d^{\sigma_1-1} = (d_1, \dots, d_{\sigma_1-1})$  and define  $d = \sum_{t=1}^{\sigma_1-1} d_t$ . Taking conditional expectations, (17)–(18) now yield

$$\mathbb{E}[\tilde{R}|D^{\sigma_1-1} = d^{\sigma_1-1}] = f \cdot \min\{x^{\text{H}(\sigma_2)}, d\} + v_{\sigma_1}(c - A \min\{x^{\text{H}(\sigma_2)}, d\}) \quad (19)$$

$$\mathbb{E}[R^{\text{H}(\sigma_2)}|D^{\sigma_1-1} = d^{\sigma_1-1}] = f \cdot \min\{x^{\text{H}(\sigma_2)}, d\} + \mathbb{E}\left[\sum_{t=\sigma_1}^{\tau} f \cdot N_t^{\pi^{\text{H}(\sigma_2)}} \middle| D^{\sigma_1-1} = d^{\sigma_1-1}\right] \quad (20)$$

To see why (19) is true, observe that given  $D^{\sigma_1-1} = d^{\sigma_1-1}$ , the remaining capacity prior to time  $\sigma_1$  when following policy  $\tilde{\pi}$  is precisely  $c - A \min\{x^{\text{H}(\sigma_2)}, d\}$ . Furthermore, policy  $\tilde{\pi}$  employs an optimal decision rule for periods  $\sigma_1, \dots, \tau$ . So, given  $D^{\sigma_1-1} = d^{\sigma_1-1}$ , the conditional expectation of the second term on the right side of (17) is  $v_{\sigma_1}(c - A \min\{x^{\text{H}(\sigma_2)}, d\})$ .

When  $D^{\sigma_1-1} = d^{\sigma_1-1}$ , the remaining capacity at  $\sigma_1$  under  $\pi^{\text{H}(\sigma_2)}$  is  $c - A \min\{x^{\text{H}(\sigma_2)}, d\}$ . Therefore (see, e.g., Theorem 4.5.1 part (a) of Puterman 1994), we have that

$$v_{\sigma_1}(c - A \min\{x^{\text{H}(\sigma_2)}, d\}) \geq \mathbb{E}\left[\sum_{t=\sigma_1}^{\tau} f \cdot N_t^{\pi^{\text{H}(\sigma_2)}} \middle| D^{\sigma_1-1} = d^{\sigma_1-1}\right]. \quad (21)$$

In light of (19)–(20), this proves (16) and thereby completes the proof.  $\blacksquare$

## 6 Solution Procedures

We now discuss some procedures to solve (10). In its stated form, (10) is an integer nonlinear stochastic optimization problem, and as such is notoriously difficult to solve. However,

since the purpose of (10) is to provide a heuristic allocation, it is reasonable to consider approximations to that problem. In particular, we will relax the constraint  $x \in \mathbb{Z}_+^n$  to  $x \in \mathbb{R}_+^n$ .

To put the problem into the standard framework of stochastic optimization, let  $G(x, D)$  denote the function

$$G(x, D) = f \cdot \min\{x, D_{(\sigma-1)}\} + v_\sigma(\lfloor c - A \min\{x, D_{(\sigma-1)}\} \rfloor), \quad (22)$$

and let  $g(x) = \mathbb{E}[G(x, D)]$ . We can now formulate the relaxed version of (10) as

$$\max \{g(x) : Ax \leq c, x \geq 0\}. \quad (23)$$

One of the most-studied techniques for stochastic optimization problems is the *Stochastic Approximation* (SA) method; see, e.g., Kushner and Yin (1997). Although widely applicable, this procedure has some drawbacks, especially the slow convergence and the difficulty in implementing good stopping criteria. A somewhat different approach is based on *Monte Carlo*-type methods. The basic idea is to replace the expected value  $\mathbb{E}[G(x)]$  with the approximation  $g_N(x) \equiv N^{-1} \sum_{i=1}^N G(x, D^i)$ , where  $D^1, \dots, D^N$  are i.i.d. samples of  $D_{(\sigma-1)}$ , and then solve the approximating problem

$$\min_{x \in X} g_N(x). \quad (24)$$

The rationale is that, by the strong law of large numbers,  $g_N(x) \rightarrow g(x)$  as  $N \rightarrow \infty$  with probability one, and thus (24) should provide a good approximation to (23) for  $N$  large. This idea, of course, is not new, and has been object of study in different areas for many years. The convergence properties of such procedures are well-established; see, for instance, Shapiro (2002) for a compilation of results.

An alternative to the basic Monte Carlo idea is proposed by Shapiro and Homem-de-Mello (1998). Instead of fixing a sample from the beginning and then minimizing the resulting deterministic function, they use *different samples* along the algorithm. In our context, this means that at the  $k$ th iteration of the algorithm we use the approximating function

$$\hat{g}_k(x) := \frac{G(x, D^{k,1}) + \dots + G(x, D^{k,N_k})}{N_k}, \quad (25)$$

where  $D^{k,1}, \dots, D^{k,N_k}$  are i.i.d. the samples of  $D_{(\sigma-1)}$ , drawn at iteration  $k$ . In reality, there is no need to re-sample at every iteration — we can keep the same sample for a few iterations, particularly at the early stages of the process.

There are a few advantages with the above approach: one is that the sample sizes  $N_k$  can increase along the algorithm, so that sampling effort is not wasted in initial iterations of the process. Also, because the estimates at different iterations are independent, one can perform *statistical tests* to compare the estimates, which in turn can lead to *stopping criteria* for the algorithm. Moreover, the re-sampling procedure tends to minimize the chance of obtaining a bad solution because of a “bad” sample path, which can occur in (24) if  $N$  is not large enough. The disadvantage is that the function being optimized changes at every iteration — which means that standard deterministic optimization packages cannot be used. Rather, the code must be adapted to incorporate sampling in it. In Shapiro and Homem-de-Mello (1998), one such code, suitable for convex problems with linear constraints, is presented. A similar method is used by Homem-de-Mello et al. (1999), who successfully apply these techniques to the optimization of release times of jobs in a single-line production environment.

Our procedure to solve the hybrid problem (23) is based on such variable-sample methods. The procedure incorporates sampling and statistical techniques into a nonlinear programming method that combines line search with projection of the gradient of the objective function onto the null space of active constraints; see, for example, Bertsekas (1999) for a thorough discussion of the latter topic. Notice that, strictly speaking, the objective function in (23) is not differentiable, due to the rounding operation. In what follows, the operator  $\nabla$  denotes an approximate gradient (cf. section 6.1 below).

Since the optimization method we implement is very similar to the one used in Homem-de-Mello et al. (1999), we will not repeat its description here in detail; refer to that paper for a full discussion. For the sake of completeness, we present below an outline of the algorithm.

1. Choose a (small) sample size  $N = N_1$  and initial value  $x^1$  of allocations; set  $k := 1$ .
2. For the current iterate  $k$ , generate an i.i.d. sample of size  $N_k$  of vectors  $D^1, \dots, D^{N_k}$ ; compute the estimators  $\hat{g}_k(x^k)$  (cf. (25)) and  $\nabla \hat{g}_k(x^k)$  (cf. section 6.1 below).
3. Compute an ascent direction  $\delta_k$  by projecting the estimator  $\nabla \hat{g}_k(x^k)$  onto an appropriate space and compute a stepsize  $\alpha_k$  by a (crude) line search.
4. Set  $x^{k+1} := x^k + \alpha_k \delta_k$  and compute  $\hat{g}_k(x^{k+1})$  and  $\nabla \hat{g}_k(x^{k+1})$  using the same demand-vector sample  $D^1, \dots, D^{N_k}$ .

5. If the increase  $\hat{g}_k(x^{k+1}) - \hat{g}_k(x^k)$  is significantly large, then go back to step 3 with  $k \leftarrow k + 1$ . Otherwise, generate a new sample  $\tilde{D}^1, \dots, \tilde{D}^{N_k}$  and compute new estimators  $\tilde{\hat{g}}_k(x^{k+1})$  and  $\nabla \tilde{\hat{g}}_k(x^{k+1})$ .
6. Compute a new appropriate sample size  $N_{k+1}$  and extend the sample to  $\tilde{D}^1, \dots, \tilde{D}^{N_k}, \tilde{D}^{N_{k+1}}, \dots, \tilde{D}^{N_{k+1}}$ . Compute  $\hat{g}_{k+1}(x^{k+1}), \nabla \hat{g}_{k+1}(x^{k+1})$  based on that sample.
7. Test statistical stopping criteria based on the difference  $\hat{g}_{k+1}(x^{k+1}) - \hat{g}_k(x^k)$  and on the estimator  $\nabla \hat{g}_{k+1}(x^{k+1})$ . If none of them is satisfied, go to step 3 with  $k \leftarrow k + 1$ . Otherwise, STOP.

Notice that, due to the lack of convexity/concavity in the present case, the procedure may end up at a stationary but perhaps sub-optimal point. A closer look at the objective function  $G(x, D)$  in (22), however, shows that the lack of concavity occurs only outside the region  $\{x \leq D_{(\sigma-1)}\}$ ; therefore, for larger values of  $\sigma$  — and consequently for larger  $D_{(\sigma-1)}$  —  $G(x, D)$  will be concave in the region of interest with higher probability. Due to the rounding operation in (22), “concavity” here should be understood in the context of Theorem 2.

It is clear that the procedure discussed above provides only a heuristic approximation to (10). Nevertheless, the numerical results in section 7 (and others not shown) suggest that the solutions obtained with such heuristics typically are satisfactory.

## 6.1 Computing derivatives

Implementation of the algorithm described above requires calculation of an “approximate gradient”  $\nabla g$  of the objective function  $g(x)$ . Consider the function  $G^{\text{R}}(x, D)$  defined as  $G$  in (22), but with the relaxed function  $v_\sigma^{\text{R}}$  defined in (12) in place of  $v_\sigma$ . Let  $\partial G^{\text{R}}(x, D)$  denote the *generalized gradient* of  $G^{\text{R}}(\cdot, D)$  at  $x$ , and let  $\nabla G^{\text{R}}(x, D)$  denote an arbitrary element of  $\partial G^{\text{R}}(x, D)$ . The generalized gradient can be viewed as an extension of the concept of subdifferential set, standard in convex analysis, to cases when the function is not convex; for details, see Clarke (1983). By writing  $\min\{x_j, D_{(\sigma-1),j}\} = x_j \mathbb{I}\{x_j < D_{(\sigma-1),j}\} + D_{(\sigma-1),j} \mathbb{I}\{x_j \geq D_{(\sigma-1),j}\}$  we have that

$$[\nabla G^{\text{R}}(x, D)]_j = f_j \mathbb{I}\{x_j < D_{(\sigma-1),j}\} - A_j^T \mathbb{I}\{x_j < D_{(\sigma-1),j}\} \nabla v_\sigma^{\text{R}}(c - A \min\{x, D_{(\sigma-1)}\}),$$

where  $A_j$  is the  $j$ th column of  $A$ . Now, finiteness of support of  $D_{(\sigma-1)}$  implies that we can exchange the derivative and expectation operators, i.e.,  $\partial g^{\text{R}}(x) = \mathbb{E}[\partial G^{\text{R}}(x, D)]$ .

Using the above result, we can estimate  $\nabla g^R$  via sampling, provided we can estimate  $\nabla v_\sigma^R$ . The latter quantity can be approximated by *two-sided* finite differences, i.e.,

$$[\nabla v_\sigma^R(r)]_j \approx \frac{v_\sigma^R(r + e_j) - v_\sigma^R(r - e_j)}{2}. \quad (26)$$

Here,  $e_j$  is the vector with 1 in the  $j$ th component and zeros otherwise. The above approximation must be adjusted for points at the boundary of the domain, that is, when  $r_j = 0$  or  $r_j = c_j$ . In those cases, a one-sided finite difference is used. Finally, by replacing  $v_\sigma^R$  by  $v_\sigma$  in (26) and rounding down the arguments of  $v_\sigma$  to get integer values, we obtain an estimate for  $\nabla g$  that can be used in the algorithm. We use this technique to obtain the derivative estimates in the algorithm described above. A similar procedure is used by Bertsimas and de Boer (2000), though they work exclusively with one-sided derivatives.

## 7 Numerical Experiments

In this section we present some numerical examples that enable us to compare the hybrid policy (11) and some of its simpler “relatives” with certain other baseline heuristics as well as with optimal policies. In our tests, we consider three different types of hybrid policies. The simplest, which we refer to as LP-MDP, involves solving the linear program (8) at the beginning of the time horizon (at  $t = 0$ ) and following the resulting allocations up to a predetermined switch time, at which point we begin using the optimal policy. This can be viewed as the simplest way to implement the idea of using math programming far from time of departure, and switching to an optimal decision rule near time of departure.

The second type of hybrid policy that we consider is again based upon the linear program. The policy LP(200)-MDP involves solving the LP at time 0 and following the resulting allocation for 200 time periods, at which point the LP is re-solved with capacity replaced by remaining capacity and expected demand replaced by expected remaining demand. The resulting policy is followed until time 400. The procedure of successively re-solving and myopically applying the allocations is repeated every 200 time units, until the switch time  $\sigma$  is reached, at which point we commence using optimal decision rules. Note that although it is reasonable to expect these policies to yield higher expected revenue than the policies LP-MDP, this need not be the case (see Cooper 2001). We also consider the policies LP(25)-MDP and LP(50)-MDP, which instead re-solve every 25 and 50 time periods respectively, prior to switching to the MDP.

The third type of hybrid policy we test is an approximation to  $\pi^{\text{H}(\sigma)}$ , obtained from (23). Observe that for a particular switch time,  $\pi^{\text{H}(\sigma)}$  necessarily yields higher expected revenue than does the policy from LP-MDP, because the policy followed by LP-MDP is a feasible solution to (10). This is not the case for LP(25)-MDP, LP(50)-MDP, and LP(200)-MDP, since (10) does not consider policies that modify their allocations prior to the switch time. In the tables and charts, SP-MDP is used to denote the policy obtained from the variable-sample stochastic optimization routine that we employ to solve (23), which approximates (10). Since this solution is not guaranteed to be optimal for (10), it could possibly be the case that the expected revenue obtained from following it will be worse than that from LP-MDP. However, we have not observed this phenomenon in any examples.

The algorithm for computing the policies used to generate the SP-MDP policies requires samples from  $D_{(\sigma-1)}$ . To simplify the algorithm, rather than sampling from the distribution of  $D_{(\sigma-1)}$ , we instead drew samples from a Poisson distribution with mean  $\mathbb{E}D_{(\sigma-1)}$ . Formal justification for this can be found in Chapter 10 of Ross (1996). As a practical matter, our experience with a variety of problem instances suggests that this method does work well.

## 7.1 Simulation methodology

The following methodology was applied to all the examples described below. For each fixed switch time  $\sigma$ , we simulated the five policies discussed above by generating a realization of the actual Bernoulli arrival process described in section 3. We then computed the average revenue for each policy over 1000 independent replications. For each of the 1000 simulation runs, the *same stream* of random numbers was used to generate the arrival process for each policy; hence for a fixed  $\sigma$ , all policies under consideration see the same stream of demands.

The process above was repeated for  $\sigma = 1, 201, 401, 601, 801, \text{ and } 1001$  (the examples use  $\tau = 1000$ ) to understand the behavior of the policies as a function of the switch time. We used *independent* random numbers for the simulations carried out with different values of  $\sigma$ ; e.g., the demand stream for evaluating the policies with  $\sigma = 201$  was different from (and independent of) that for evaluating the policies with  $\sigma = 401$ .

Results are presented in Tables 1–4, and depicted graphically in Figures 1–2. In Tables 1 and 3, there are two numbers in each cell. The first number is an estimate of the expected revenue from following the policy determined by the column with the switch time determined

by the row. This estimate is the sample average of 1000 simulation runs of the policy in question. The second number is the half-width of a 98% confidence interval for the expected revenue. Notice that, by computing a 98% confidence interval for each policy, we ensure that the *overall* confidence on the intervals obtained for all five policies (for a particular switch time  $\sigma$ ) is at least 90%; this follows immediately from Bonferroni’s inequality.

The purpose of Tables 2 and 4 is to compare each LP-based policy with the SP-MDP policy. Again, there are two numbers in each cell. The first number is an estimate of  $\mathbb{E}[R^{\text{SP-MDP}} - R^{\text{LP(n)-MDP}}]$ , where  $R^{\text{SP-MDP}}$  is the revenue obtained from the SP-MDP policy and  $R^{\text{LP(n)-MDP}}$  is the revenue for the corresponding LP-based policy (both for the value of  $\sigma$  specified by the row). The second number is the half-width of a 95% confidence interval for the difference  $\mathbb{E}[R^{\text{SP-MDP}} - R^{\text{LP(n)-MDP}}]$ . Again, Bonferroni’s inequality implies that the overall confidence on the four intervals obtained for each fixed switch time is at least 80%.

The above confidence intervals for the differences should be interpreted as follows. If zero is contained in such interval, then for the switch time in question, there is no indication that the policy SP-MDP gives higher revenue than the corresponding LP-based policy. Otherwise, if the calculated interval is to the right of zero, then there is indication that the SP-MDP policy performs better than the corresponding LP-based policy for that particular switch time. Such cases are signaled with “+” in the difference tables. Similarly, a “−” is displayed when the confidence interval falls to the left of zero, which indicates superiority of the corresponding LP-based policy over the SP-MDP policy for that switch time.

## 7.2 A two-leg example

Consider a two-leg problem with capacity of 150 seats on each leg. There are three possible itineraries (numbered 0 for the through itinerary that uses both legs, 1 for the local itinerary that uses only leg 1, and 2 for the local itinerary that uses only leg 2), each with three fare classes. (We are now using class and itinerary in the more traditional sense.) The through itinerary fares are \$600, \$100, and \$75. The fares for each local itinerary are \$500, \$75, and \$50. There are  $\tau = 1000$  time periods. The arrival probabilities for the highest (Class 1), second-highest (Class 2), and lowest (Class 3) class on each itinerary are given by  $P(D_{t,(1,\ell)} = 1) = \alpha\beta_{(1,\ell)} \sin\left(\frac{t\pi}{2\tau}\right)$ ,  $P(D_{t,(2,\ell)} = 1) = \alpha\beta_{(2,\ell)} \sin\left(\frac{t\pi}{\tau}\right)$ , and  $P(D_{t,(3,\ell)} = 1) = \alpha\beta_{(3,\ell)} \sin\left(\frac{\pi}{2} + \frac{t\pi}{2\tau}\right)$ , where  $D_{t,(k,\ell)}$  is the demand for class  $k$ , itinerary  $\ell$  at time  $t$ . Observe

that this means that high-class (Class 1) expected demand per time-unit grows as we get closer to time of departure, whereas low-class (Class 3) expected demand per time-unit falls as we approach time of departure. Class 2 expected demand per time-unit peaks in the middle of the booking process. In this example, we use  $\alpha = .46$ ,  $\beta_{(1,0)} = 0.05$ ,  $\beta_{(2,0)} = 0.07$ ,  $\beta_{(3,0)} = 0.08$ , and  $\beta_{(1,1)} = \beta_{(1,2)} = 0.10$ ,  $\beta_{(2,1)} = \beta_{(2,2)} = 0.12$ ,  $\beta_{(3,1)} = \beta_{(3,2)} = 0.18$ . This yields a total expected demand of approximately 175 seats per leg.

Results are presented in Tables 1 and 2 and Figure 1. For this example, exact computation for the MDP shows that the maximum expected revenue is  $v_1(c) = 49737.23$ , which does lie within the confidence interval shown in the  $\sigma = 1$  row. Examination of Tables 1 and 2 shows that SP-MDP appears to be slightly stronger than the other heuristics, except when  $\sigma = 1001$ . Although the difference in performance is statistically significant in many of the cases shown, the estimated differences in revenues are indeed quite small. It is interesting to point out that all of the switching heuristics can obtain “most” of the benefit from the MDP by using optimal decision rules in less than half of the booking horizon. Even when switching with only one-fifth of the horizon remaining, three of the four heuristics performed quite well, and the fourth (LP-MDP) performed reasonably well.

### 7.3 Another two-leg example

We now briefly describe a two-leg example where the SP-MDP is the clear winner among the heuristics. Consider a two-leg problem with three fare classes, 150 seats, and  $\tau = 1000$ . The through fares are \$1000, \$900, and \$100, and the local fares are \$800, \$700, and \$50. All class 2 and class 3 itineraries have identical constant arrival probabilities of 0.0656 from  $t = 1$  through 800, and zero thereafter. Up to time 800 there are no class 1 arrivals on any itinerary. After time 800, the class 1 arrival probabilities are constant at 0.0375. The expected total demand per leg is therefore 225. In Tables 3 and 4 and Figure 2, we see that SP-MDP performs significantly better than all the other heuristics, and nearly as well as the pure MDP, which has a value of 142344.7. In this example, the difference between high-class fares and low-class fares is quite high; such cases are the ones in which the stochastic programming formulation appears to be of more importance, since randomness plays a larger role. We will say a bit more about this in the next section.

## 7.4 Some general comments on computation

Aside from the examples presented in the previous section, we performed tests on a variety of one- and two-leg problems. In this section, we describe some of our observations based upon these experiments. We first discuss the issue of computational time and data storage. The stochastic optimization algorithm does seem to yield significant savings in this respect, particularly for large problems. For example, it took 11 seconds (on a 1.7GHz Pentium) to run the MDP code for a typical two-leg example such as the ones described in sections 7.2 and 7.3, with  $\tau = 1001$  and 150 seats per leg. The hybrid method with switch time, say,  $\sigma = 601$  — which typically “nearly” preserved the optimal value — took between 7 and 10 seconds for the stochastic optimization routine, plus 5 seconds for the MDP algorithm, which in this case was run from time 601 onwards. So, while there was no gain in computational time, there was an advantage in storage requirements, since the MDP routine needs to store the optimal decision rules.

The above comparison becomes starker when the problem is scaled up. For example, it is easy to check that in two-leg problems, the time taken by the MDP routine is proportional to the product of the capacity of the legs; thus, by doubling the capacity of each leg, we expect the MDP algorithm to take four times as long. We verified this empirically, using a two-leg example with  $\tau = 1001$  and 300 seats per leg. The MDP took 46 seconds to find the solution, as opposed to 11 seconds as before. The stochastic optimization routine, however, is largely unaffected by changes in capacity, since such a modification simply translates into a change on the right-hand side of the constraints. Again, we verified this conjecture numerically. Using  $\sigma = 601$  as before, the stochastic optimization routine took the same 7-10 seconds as in the 150-seat case, whereas the MDP from time 600 onwards took 20 seconds. That is, the overall time for the hybrid method was 27-30 seconds, compared to 46 seconds to the full MDP — a savings of 40% in time, with even bigger savings in terms of memory requirements and virtually no loss in the value of the objective function.

The time required by the simulation-based optimization routine to solve (10) depends upon the sample sizes used in Step 2 of the algorithm and the significance level used for the statistical stopping criteria in Step 7 (among other things). Naturally, the routine tends to be slower when we use larger sample sizes and lower significance levels. Moreover, the time spent within the stochastic optimization routine increases as the switch time becomes closer

to departure. This occurs because the sampling variance of  $D_{(\sigma-1)}$  increases as  $\sigma$  increases. However, this is counterbalanced by a decrease in the time spent on the MDP, computing  $v_\sigma$ . Therefore the net effect is that computational times decrease as  $\sigma$  increases.

Policy SP-MDP tends to perform better than LP( $n$ )-MDP when there are different classes with very different fares but the same resource requirements arriving simultaneously, as in the example in section 7.3. In part, this is because LP-based allocations have difficulties with such situations, regardless of how often the policy is updated. When there are no such significant overlaps, LP( $n$ )-MDP and SP-MDP typically perform similarly, with LP( $n$ )-MDP with “frequent enough” updating sometimes performing a bit better than SP-MDP given a fixed switch time  $\sigma$ , but with SP-MDP usually beating LP( $n$ )-MDP with infrequent updating and large  $\sigma$ . Nevertheless, we cannot make a blanket statement that one is better than the other, since performance is greatly influenced by the specifics of a particular problem. Our numerical experience also suggests that, in most cases, re-solving the LP more frequently yields higher or equivalent average revenues compared to less frequent re-solving. This is not so surprising, since re-solving does incorporate new information into the optimization problem. The advantages are more noticeable for larger values of  $\sigma$ .

Another comment concerns the switch time  $\sigma$ . The example in section 7.2 suggests that, if we switch to the MDP by time 401 or 601 (for the 1000 time-unit problem), the switching heuristics all tend to yield nearly the same (from a statistical point of view) mean revenue as the pure MDP. This is again consistent with our original motivation for using switching methods — namely, that it suffices to use relatively crude decision rules far from departure so long as the key decisions near departure are made using the MDP methods. However, in the example in section 7.3, the switching time appears to be critical for all but SP-MDP.

## 8 Conclusions

Many authors have considered variants of Markov decision process models for revenue management. Likewise, there is a large body of work that describes how to solve revenue management problems using mathematical programming techniques. In this paper, we have studied a class of hybrid revenue management policies that combine aspects of MDP and math-programming approaches. We have presented a formulation that casts the problem as a two-stage stochastic optimization problem, and have adapted recently-developed variable-

sample stochastic optimization techniques to approximately solve it. Our numerical results suggest that policies derived from our formulation, as well as some simpler variants, do appear to perform well. In addition, by viewing the two-leg problem as a multi-stage stochastic linear program, we have proven new structural properties of the MDP value function. Moreover, we have given an example that shows that such properties do not, in general, hold for networks with three or more legs.

We have not considered overbooking, cancellations, or consumer choice behavior. Since MDP models that incorporate these effects do exist, see, e.g., Subramanian et al. (1999) and Talluri and van Ryzin (2000), it is not difficult from a conceptual standpoint to incorporate such effects into our method. In addition, stochastic optimization techniques appear to be equally well-suited for handling some such problems; see Karaesmen and van Ryzin (1998). Other directions for future work include incorporation of approximation techniques for solving the MDP portion of the problem in the case of larger networks, as well as methods for addressing uncertainty in the forecasts of model parameters.

## Acknowledgment

We would like to thank Dan Zhang for conducting numerical tests. This material is based upon work supported by the National Science Foundation under Grant No. DMI-0115385.

## References

- D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 2nd ed., 1999.
- D. Bertsimas and S. de Boer. A stochastic booking-limit control policy for airline network revenue management. Working paper, Operations Research Center, Massachusetts Institute of Technology, 2000.
- J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer Series in Operations Research. Springer-Verlag, New York, NY, 1997.
- F. H. Clarke. *Optimization and Nonsmooth Analysis*. John Wiley & Sons, Inc., New York, 1983. Reprint by SIAM, Philadelphia, 1990.

- W. L. Cooper. Asymptotic behavior of an allocation policy for revenue management. Forthcoming in *Operations Research*, 2001.
- S. V. de Boer, R. Freling, and N. Piersma. Stochastic programming for multi-leg network revenue management. Working paper, Erasmus University, 1999.
- Y. Feng and B. Xiao. Optimal policies of yield management with multiple predetermined prices. *Operations Research*, 48:332–343, 2000.
- G. Gallego and G. van Ryzin. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management Science*, 40:999–1020, 1994.
- G. Gallego and G. van Ryzin. A multiproduct dynamic pricing problem and its applications to network yield management. *Operations Research*, 45:24–41, 1997.
- D. Günther. *Airline Yield Management: Optimal Bid Prices, Markov Decision Processes, and Routing Considerations*. PhD thesis, Georgia Institute of Technology, 1998.
- T. Homem-de-Mello, A. Shapiro, and M. L. Spearman. Finding optimal material release times using simulation based optimization. *Management Science*, 45:86–102, 1999.
- I. Karaesmen and G. J. van Ryzin. Overbooking with substitutable inventory classes. Working paper, Columbia University, Graduate School of Business, 1998.
- A. J. Kleywegt and J. D. Papastavrou. The dynamic and stochastic knapsack problem. *Operations Research*, 46:17–35, 1998.
- H. J. Kushner and G. G. Yin. *Stochastic Approximation Algorithms and Applications*. Springer-Verlag, Berlin, Germany, 1997.
- C. J. Lautenbacher and S. Stidham. The underlying Markov decision process in the single-leg airline yield management problem. *Transportation Science*, 33:136–146, 1999.
- T. C. Lee and M. Hersh. A model for dynamic airline seat inventory control with multiple seat bookings. *Transportation Science*, 27:252–265, 1993.
- J. I. McGill and G. J. van Ryzin. Revenue management: Research overview and prospects. *Transportation Science*, 33:233–256, 1999.

- A. Prékopa. *Stochastic Programming*. Kluwer Academic Publishers, Netherlands, 1995.
- M. L. Puterman. *Markov Decision Processes*. John Wiley & Sons, New York, 1994.
- L. W. Robinson. Optimal and approximate control policies for airline booking with sequential nonmonotonic fare classes. *Operations Research*, 43:252–263, 1995.
- S. M. Ross. *Stochastic Processes*. John Wiley & Sons, New York, 1996.
- A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, New York, NY, 1986.
- A. Shapiro. Monte Carlo sampling methods. In A. Ruszczyński and A. Shapiro., editors, *Handbook of Stochastic Optimization*. Elsevier Science Publishers B.V., Amsterdam, Netherlands, 2002. (Forthcoming).
- A. Shapiro and T. Homem-de-Mello. A simulation-based approach to two-stage stochastic programming with recourse. *Mathematical Programming*, 81:301–325, 1998.
- J. Subramanian, S. Stidham, and C. J. Lautenbacher. Airline yield management with overbooking, cancellations, and no-shows. *Transportation Science*, 33:147–168, 1999.
- K. Talluri and G. van Ryzin. An analysis of bid-price controls for network revenue management. *Management Science*, 44:1577–1593, 1998.
- K. Talluri and G. van Ryzin. A randomized linear programming method for computing network bid prices. *Transportation Science*, 33:207–216, 1999.
- K. Talluri and G. van Ryzin. A discrete choice model of yield management. Working paper, Department of Economics and Business, Universitat Pompeu Fabra, 2000.
- E. L. Williamson. *Airline Network Seat Control*. PhD thesis, Massachusetts Institute of Technology, 1992.
- P.-S. You. Dynamic pricing in airline seat management for flights with multiple flight legs. *Transportation Science*, 33:192–206, 1999.

$\sigma$	LP-MDP	LP(200)-MDP	LP(50)-MDP	LP(25)-MDP	SP-MDP
1	49906.03, 284.52				
201	49836.23, 279.31	49836.23, 279.31	49824.05, 278.48	49823.25, 278.35	49834.80, 279.39
401	49630.15, 277.72	49621.13, 277.72	49605.63, 276.56	49606.25, 276.58	49630.23, 277.89
601	49645.75, 279.28	49639.75, 279.28	49628.98, 277.29	49628.70, 277.37	49652.85, 280.35
801	49162.78, 254.82	49327.15, 254.82	49387.98, 259.97	49354.73, 258.77	49372.15, 279.73
1001	46743.98, 177.40	48340.28, 252.64	48971.40, 262.59	48910.63, 263.29	48512.18, 291.58

Table 1: Mean revenue as a function of switch time for the example in section 7.2.

$\sigma$	LP-MDP	LP(200)-MDP	LP(50)-MDP	LP(25)-MDP
201	-1.43, 1.62	-1.43, 1.62	10.75, 4.83 +	11.55, 4.79 +
401	0.08, 1.94	9.10, 4.55 +	24.60, 6.23 +	23.98, 6.54 +
601	7.10, 6.45 +	13.10, 10.64 +	23.88, 12.46 +	24.15, 12.82 +
801	209.38, 51.66 +	45.00, 53.37	-15.83, 49.56	17.43, 51.24
1001	1768.20, 137.14 +	171.90, 96.12 +	-459.23, 76.14 -	-398.45, 75.58 -

Table 2: Differences in revenue for the example in section 7.2.

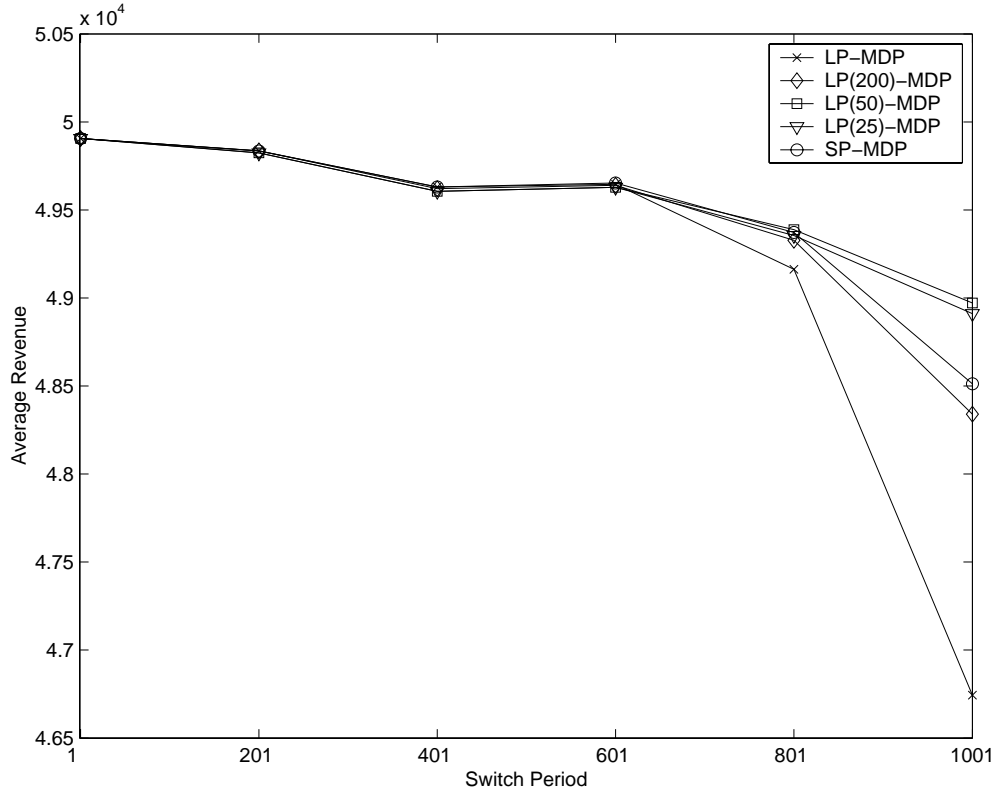


Figure 1: Mean revenue as a function of switch time for the example in section 7.2.

$\sigma$	LP-MDP	LP(200)-MDP	LP(50)-MDP	LP(25)-MDP	SP-MDP
1	142891.10, 627.77				
201	139799.80, 475.63	139799.80, 475.63	140244.50, 496.86	140181.10, 494.39	142187.55, 638.50
401	138112.40, 383.44	138405.10, 435.85	138311.35, 446.16	138212.70, 446.64	142703.10, 647.77
601	137821.10, 382.88	137479.80, 423.08	137336.25, 444.09	137246.85, 445.10	141834.40, 646.76
801	135730.60, 384.81	136119.10, 425.51	136353.75, 441.63	136500.55, 441.55	141299.40, 634.60
1001	134136.20, 383.28	134520.55, 428.40	135521.50, 454.54	135818.55, 452.32	139460.90, 599.69

Table 3: Mean revenue as a function of switch time for the example in section 7.3.

$\sigma$	LP-MDP	LP(200)-MDP	LP(50)-MDP	LP(25)-MDP
201	2387.75, 245.71 +	2387.75, 245.71 +	1943.05, 215.59 +	2006.45, 218.41 +
401	4590.70, 324.32 +	4298.00, 301.39 +	4391.75, 291.82 +	4490.40, 290.02 +
601	4013.30, 322.62 +	4317.25, 636.08 +	4498.15, 288.59 +	4587.55, 289.35 +
801	5568.80, 319.65 +	5180.30, 296.13 +	4945.65, 272.23 +	4798.85, 274.04 +
1001	5324.70, 263.10 +	4940.35, 254.24 +	3939.40, 243.12 +	3642.35, 242.56 +

Table 4: Differences in revenue for the example in section 7.3.

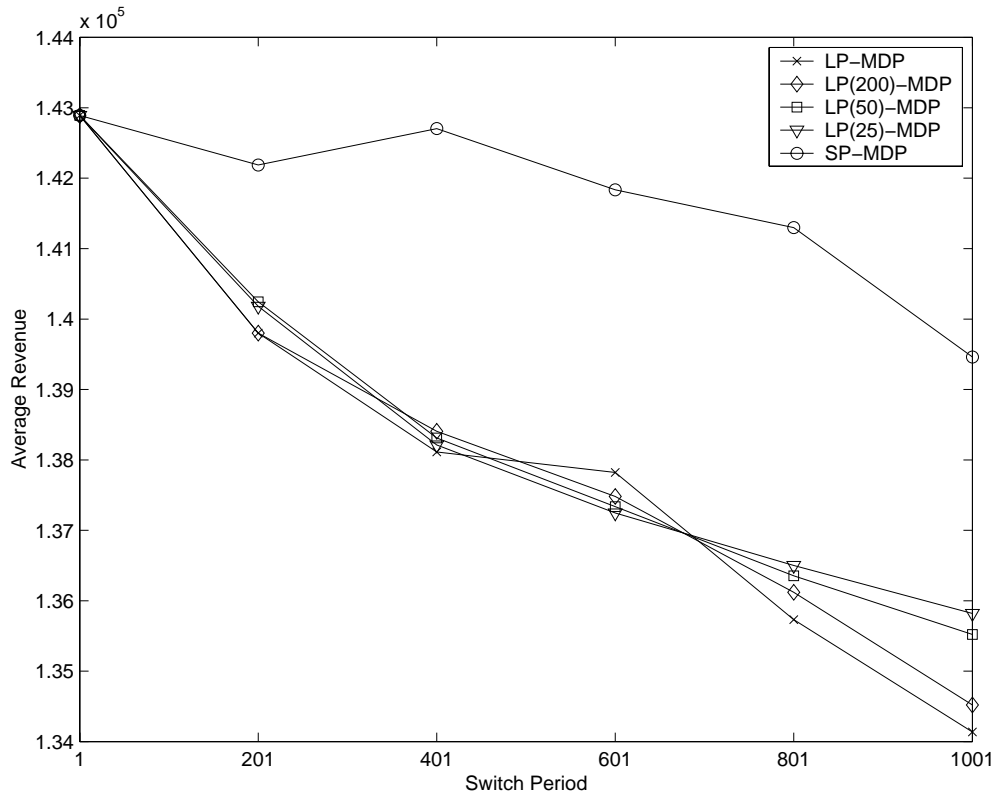


Figure 2: Mean revenue as a function of switch time for the example in section 7.3.