Available online at www.sciencedirect.com
ScienceDirect
Int. J. Production Economics 104 (2006) 340–353
international journal of production economics
www.elsevier.com/locate/ijpe

Modeling revenue yield of reservation systems that use nested capacity protection strategies

Laila Haerian[a], Tito Homem-de-Mello[b], Clark A. Mount-Campbell[a,*]

[a]Industrial, Welding and Systems Engineering, The Ohio State University, Columbus, OH 43210, USA
[b]Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60208, USA

Received 15 April 2004; accepted 9 February 2006
Available online 5 May 2006

Abstract

Airline reservation systems involve the use of booking policies to implement a predetermined allocation of seats to different fare classes. Models for optimal allocation of seats typically assume one of two commonly used booking policies, often without recognizing the differences between them. In this paper, we present alternative representations of these booking policies, and demonstrate that even with identical seat allocations the two booking policies may result in different expected revenues. We also show conditions under which one of the policies is better. Our Markov chain models facilitate optimization of seat allocations given either booking policy. Examples are given.

Keywords: Airline revenue management; Seat inventory control; Booking policies; Markov chains

1. Introduction

After deregulation of the airline industry in 1970s, airlines were able to set their own prices. They could also sell seats of the same cabin for different prices. That was an early stage for airline revenue management. This area has evolved considerably since then; see for instance the articles by McGill and Van Ryzin, 1999, Park and Piersma (2002) and Boyd and Bilegan (2003) for excellent reviews, as well as the recent book by Talluri and van Ryzin (2004). It has also expanded into businesses other than air passenger services. Broadly speaking, revenue management deals with various issues including pricing, forecasting and demand modeling (for instance, see Dai et al., 2005) and inventory and capacity allocation (see Sridharan, 1998, for example). It is interesting to note that Coelli et al. (2002) found that airlines typically fail to generate the maximum possible revenue from their capacity primarily due to less than optimal capacity utilization. In the airline industry, one of the principal problems aimed at capacity management is seat inventory control, the purpose of which is to allocate the seats of the same cabin to different fare classes to maximize the total revenue. Airlines usually offer tickets for many origin destination flights. Typically, the problem of seat inventory control of the network of flights is solved in one of two ways: as independent single leg flights, or through heuristic methods for a network of

*Corresponding author. Tel.: +1 614 292 7856; fax: +1 614 292 7852.
E-mail addresses: haerian.1@osu.edu (L. Haerian), tito@northwestern.edu (T. Homem-de-Mello), mount-campbell.1@osu.edu (C.A. Mount-Campbell).

0925-5273/$ - see front matter
doi:10.1016/j.ijpe.2006.02.006

simultaneous flights. Solving single leg problems allows for (i) the use of powerful techniques like dynamic programming, and (ii) the incorporation of many features like overbooking, cancellation, consumer choice, etc. However, the interaction between flights is lost. For example, if the problem is solved as independent legs, one connected flight using two legs might be rejected in favor of accepting a single leg flight with higher revenue per leg, even though it might have been more profitable to the network to accept the connected flight. Williamson (1992) used simulation to show that solving the problem as a network of flights makes a significant increase in expected revenue compared to solving it as independent legs.

Littlewood (1972) was the first to propose a model for the seat inventory problem. That model was developed for single leg problems with only two fare classes. The basic idea is to accept the lower fare requests until the revenue of accepting another low fare class is exceeded by the expected revenue of a high fare class. Belobaba (1987) extended Littlewood's idea to more than two fare classes and proposed a heuristic to find the booking control policy. Later, Curry (1990), Wollmer (1986) and Brumelle and McGill (1993) proposed different algorithms to find the optimal allocation.

Glover et al. (1982) were the first to propose a network formulation for the airline revenue-management problem. They formulated the problem as a network flow problem. The drawback of their model is the necessity of passengers being path indifferent, which is not a realistic assumption. The integer-programming model underlying the network flow problem is able to distinguish between different paths. However, it only takes into account the expected value of demand and the stochastic nature of the demand is not fully captured. The constraint of the integer-programming model is that number of seats allocated to each class is no more than expected value of demand for that class. The linear programming (LP) relaxation of Glover's et al. model is a deterministic linear programming (DLP) model. Wollmer (1992) proposed a model, which takes into account the distribution of the demand. A drawback of this model is large number of decision variables. De Boer et al. (2002) proposed a model, which is similar to Wollmer's with fewer decision variables.

A very important aspect of seat inventory control is *nesting*. The idea of nested allocation is to make the seats allocated to each fare class available to all higher fare classes. Without nesting, it is possible to reject a request for a high-fare class even though there are still seats available to one or more lower fare classes. Without nesting, this situation might happen when all the seats allocated to a specific high fare class are filled. Now the question is how to rank different fare classes. It is clear that for a single leg problem, the ranking of the booking classes is based on their fares. The higher the fare, the higher the ranking of the corresponding fare class. In a network of flights, the problem is more complex since the contribution of each fare class to the network has to be taken into account as well as the fare of the class.

There have been different ideas about how to rank different fare classes in a network of flights. Williamson (1992) suggested ranking fare classes in a network of flights based on the incremental revenue that is generated if an additional seat is made available to each fare class. For the DLP model, she approximated this by using the dual variables of the demand constraint. Another method, called *virtual nesting*, has been applied by American Airlines (see Smith et al., 1992). In this method, the origin–destination–fares on each leg are clustered into eight buckets where the first two buckets are for controlling the first and business classes only. The remaining buckets are for coach class. Each origin–destination–fare is indexed to one of the buckets. The buckets are nested.

The idea of nesting has become quite popular among practitioners. In many cases, a standard method such as LP is used to determine good seat allocations and then a nesting policy based on these allocations is implemented. Some researchers have proposed the incorporation of the nesting procedure into the optimization algorithm, although in that case the optimization techniques become much more involved. Notable examples of the latter are Bertsimas and de Boer (2005) and van Ryzin and Vulcano (2003).

In summary, the problem of interest for the airline industry involves capacity planning and the allocation of the available capacity in a network of flights. While addressing the problem at the network level has been shown to be important and some authors have attempted it, the practical end result is to implement the allocation scheme on the individual flight leg level. To implement seat allocations for each flight leg, two basic approaches have been proposed in the literature, namely, a booking limit control policy and a virtual nesting control policy.

Bertsimas and de Boer (2005) describe both methods, which they call standard nesting and theft nesting, respectively, a terminology we adopt here.

In this paper, we study the standard and theft-nesting approaches in detail. The value of this contribution lies in that, although these control policies are recognized to be distinct, to the best of our knowledge an analysis of the differences between them has not been provided. Our study alerts for the fact that the differences in revenue when implementing one approach or the other can be significant. Moreover, by showing the existence of a "crossing point" for the booking horizon, i.e. a point after which one of the policies is necessarily superior, we can help the practitioner decide which of the methods is more suitable in his or her case.

We also propose a unified alternative representation of the standard and theft-nesting methods based on the concept of a *nesting table*. Besides allowing for an easy visualization of the mechanism behind those policies—which can constitute a valuable teaching tool—our approach allows for the development of a Markov chain (MC) model that represents the booking process. The proposed model allows for exact calculation of the expected revenue under both standard and theft approaches, when requests for reservations are assumed to follow a homogeneous Poisson process, meaning that the Poisson parameter is constant and therefore time independent. Our model can be used to assess the benefits of specific allocations much more quickly and directly than simulation. In particular, since the calculations are exact and do not require simulation, our approach leads naturally to optimization of the seat allocation, which requires simpler deterministic (rather than stochastic) procedures.

The remainder of the paper is organized as follows: In Section 2 we describe the standard and theft-nesting approaches in detail and use a simple example to show how they differ. We also propose two methods—which we call fill from the right (R) and fill from the left (L) booking policies—which are provably equivalent to, respectively, standard nesting and theft nesting. In Section 3 we present simulation results for a few examples showing that the revenues obtained under these policies can be significantly different, even if the original allocation is the same. In Section 3, we present our analytical model for computation of the expected revenue and discuss some optimization issues. In Section 4 we present our conclusions and discuss some future work. Some theoretical results are presented in the Appendix A.

2. Overview of booking control policies

2.1. Problem definition and notation

Let C be the capacity of the cabin of the airplane and n denote the number of fare classes. Here we define a fare class to be a one-leg itinerary with a predetermined price. Let $x_1,\ldots,x_n$ denote the number of seats allocated to classes $1,\ldots,n$ where the classes are ordered from the highest fare class to the lowest fare class. Since we are only concerned with the nesting policies, here we assume that seat allocations have been decided beforehand using some method. Given one such allocation, we define the protection level of class i to be the number of seats that are protected for the exclusive use of class $i-1$ and higher classes. More specifically

$$\mathrm{pl}_i = \sum_{j=1}^{i-1} x_j, \tag{1}$$

where x_j is the number of seat allocated to class j. Since the x_j's must be nonnegative, it is easy to see that

$$0 = \mathrm{pl}_1 \leqslant \mathrm{pl}_2 \leqslant \cdots \leqslant \mathrm{pl}_n \leqslant C. \tag{2}$$

Both standard and theft policies use the protection levels in Eq. (1) to determine which requests should be accepted and which ones should be rejected. In order to illustrate how the two policies work and how they can lead to different results, we shall consider the following simple example to which we will refer throughout the paper.

Example 1. Consider a single leg flight with capacity 8. There are three fare classes with revenues $r_1 = \$300$, $r_2 = \$200$ and $r_3 = \$100$ and seat allocations $x_1 = 1$, $x_2 = 6$ and $x_3 = 1$. The protection levels calculated from Eq. (1) are as follows:

$$\mathrm{pl}_1 = 0,$$
$$\mathrm{pl}_2 = x_1 = 1,$$
$$\mathrm{pl}_3 = x_1 + x_2 = 1 + 6 = 7.$$

2.2. Standard nesting policy

We describe first the standard nesting policy, following closely the description in Bertsimas and de Boer (2005). First, all the fare classes are ranked. In this case, since we are dealing with a single leg flight, we rank the fare classes based on the fares. Class 1 has the highest rank followed by class two and three. Then we define inventory buckets. Let B_i represent the inventory bucket corresponding to fare class i. Inventory bucket of class i includes all the fare classes that are ranked less than or equal to class i. So in general $B_i = \{i, i+1, \ldots, n\}$.

Let $R_i(t)$ denote the seat availability or remaining capacity of bucket i after t requests have been accepted. Notice that pl_i is the number of seats that are protected from fare class i and lower ranked fare classes and therefore from bucket i. Therefore $R_i(t)$ is defined as

$$R_i(t) = s(t) - \mathrm{pl}_i, \tag{3}$$

where $s(t) = C - t$ represents the remaining capacity of the cabin after t requests have been accepted.

Once the remaining capacity of the cabin, $s(t)$, becomes equal to pl_i there are no more seats available to bucket i. So any request for fare class i, $i+1,\ldots,n-1$ or n will no longer be accepted. In other words, R_i becomes zero and remains zero. From Definition (3) and inequality (2) it is easy to see that

$$0 \leqslant R_n(t) \leqslant R_{n-1}(t) \leqslant \cdots \leqslant R_1(t) = s(t) \quad \text{for all } t. \tag{4}$$

As described by Bertsimas and de Boer (2005), "The policy is to accept booking requests as long as there are still seats available for all inventory buckets of which the considered booking class is part. If a booking request is accepted, the seat availability of any such bucket is decreased." In other words, given that t requests have already been accepted, a new request for fare class i is accepted if

$$R_1(t) > 0,\ R_2(t) > 0, \ldots, R_i(t) > 0.$$

From Eq. (4), this could be simply written as

$$R_i(t) > 0. \tag{5}$$

Based on definition of standard nesting booking policy, once a request for fare class i is accepted, R_1, $R_2,\ldots,R_i$ are all decreased by one unit. Note that a special case occurs when $R_i(t) = R_{i+1}(t) > 0$. A request for fare class i is accepted based on Eq. (5). Now, $R_i(t) = R_{i+1}(t)$ means that the current booking limit of bucket i is equal to booking limit of class $i+1$, therefore the accepted request for fare class i actually takes one of the seats that could have been sold to fare class $i+1$. Therefore, B_{i+1} will have one less seat available now. In other words if $R_i(t) = R_{i+1}(t)$, then once the request for fare class i is accepted, not only $R_1,\ldots,R_i$ but also R_{i+1} would need to be decreased by one unit. A similar argument applies to other classes $j > i$ such that $R_i(t) = R_j(t)$. Thus, the standard nesting booking policy can be re-phrased as follows:

A request for fare class k (given that t bookings have been made) is accepted if $R_k(t) > 0$, and once it is accepted, seat availability of the buckets are updated as follows,

$$R_i(t+1) = \begin{cases} R_i(t) - 1 & \text{if } i \leqslant k, \\ R_i(t) - 1 & \text{if } i > k \text{ and } R_k(t) = R_i(t), \\ R_i(t) & \text{if } i > k \text{ and } R_k(t) > R_i(t). \end{cases} \tag{6}$$

Acceptance of any request, of course, also implies that $s(t+1) = s(t) - 1$.

Now consider Example 1 above. Initially, $s(0) = C = 8$, so the initial R_i's are calculated from Eq. (3) as below,

$$R_1(0) = 8 - 0 = 8,$$
$$R_2(0) = 8 - 1 = 7,$$
$$R_3(0) = 8 - 7 = 1.$$

Suppose now that (2 3 3 2 2 2 2 2 1 1 1) represents a sequence of booking requests by class index. Table 1 shows the result of applying standard nesting policy to our example. As seen from the result, the third, tenth and eleventh requests are rejected and all other requests are accepted. In the next section, we propose a booking policy we call "fill from the right" (R) that is equivalent to the standard nesting policy but easier to visualize. Before proposing the R control policy, we define a *nesting table*, which is basis for our proposed method.

2.3. Nesting table

The idea of nested allocation can be illustrated by drawing a table with C columns and n rows corresponding to capacity and number of classes, respectively. Each column represents one seat and could be filled with only one entry in any of the

Table 1
Applying standard nesting policy to Example 1

Booking requests		2	3	3	2	2	2	2	2	1	1	1
Decision A = Accept R = Reject		A	A	R	A	A	A	A	A	A	R	R
R_1	8	7	6	6	5	4	3	2	1	0	0	0
R_2	7	6	5	5	4	3	2	1	0	0	0	0
R_3	1	1	0	0	0	0	0	0	0	0	0	0

rows, which correspond to the class. On each row, pl_i of the cells are crossed out starting from the right side of the table. Crossed out cells in each row represent the seats that are protected from the corresponding class and are reserved for higher classes only. We will call this table the "nesting table" throughout the paper. The nesting table for the previous example is shown in Fig. 1.

2.4. The fill from the right (R) booking policy

Suppose C, n and either pl_i's or x_i's are given. If x_i's are given, pl_i's can be calculated from Eq. (1). The R booking policy fills up the nesting table from the right as booking requests are accepted. More specifically, it can be described as follows:

- Draw the nesting table by using C, n and pl_i's.
- If there is a booking request for class i, start from the right side of the ith row
  - Fill up the first available seat (the first cell which is not crossed out and its corresponding column has no other entry).
  - If there is no such a seat available, reject the booking request.
- Repeat until there are no more booking requests or no seats left.

Shown in Fig. 2 are the results of applying R booking policy to Example 1. In order to make the method easier to follow, we have listed the booking requests in reverse order above the nesting table and drawn arrows to show the assignment of accepted requests to cells in the figure. Please note that the third, tenth and eleventh booking requests are rejected and all other requests are accepted. The result of applying the R policy to the example is the same as applying the standard nesting control policy. This is actually true in general, as we shall see later.

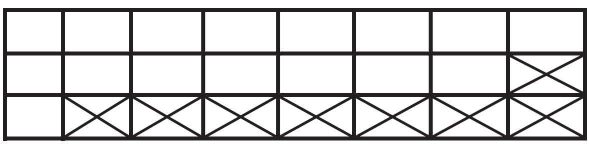

Fig. 1. Nesting table of Example 1.

2.5. Theft nesting (virtual nesting booking control)

In this section, the theft-nesting control policy is described and applied to Example 1. In the following, we adapt the theft-nesting booking policy (described and used in van Ryzin and Vulcano, 2003) to the single flight leg case as follows. As before, let $x_1,\ldots,x_n$ denote the number of seats allocated to class $1,\ldots,n$ where the classes are ordered from the highest to the lowest fare class. Let y_i denote the protection level for classes i and higher. Under a theft-nesting policy, requests in class $i+1$ are accepted if and only if the remaining capacity exceeds y_i. Notice that y_i is not exactly the same as pl_i. The quantity y_i corresponds to the number of seats that are protected for class i and higher, whereas pl_i is the number of seats that are protected from class i and made available to higher classes. It is easily seen that $y_i = \mathrm{pl}_{i+1}$ for $i = 0,\ldots,n-1$ and $y_n = C$. Therefore theft-nesting policy can be re-phrased as follows:

A request for fare class i (given that t bookings have been made) is accepted if $s(t) > \mathrm{pl}_i$.

Once a request for fare class i is accepted the remaining capacity of the cabin, $s(t)$, will be decremented by one unit, i.e. $s(t+1) = s(t) - 1$.

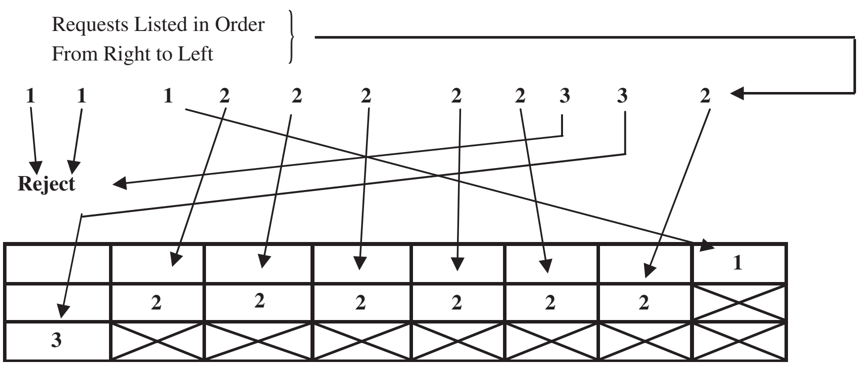

Fig. 2. Applying the R booking policy Example 1.

The result of applying theft-nesting policy to Example 1 is presented in Table 2.

We see that the result of applying theft-nesting booking control is different from applying standard nesting. We describe now a "fill from the left" booking policy, which as we shall see is equivalent to the theft-nesting policy.

2.6. Fill from the left (L) booking policy

Suppose C, n and either pl_i's or x_i's are given. If x_i's are given, pl_i's can be calculated as in Eq. (1). The L booking policy fills up the nesting table from the left. It is described in more detail as follows:

- Draw the nesting table by using C, n and pl_i's
- If there is a booking request for class i, start from the left side of the ith row
  - Fill up the first available seat (the first cell which is not crossed out and its corresponding column has no other entry).
  - If there is no such a seat available, reject the booking request.
- Repeat until there are no more booking requests or no seats left.

The result of applying L booking policy to Example 1 is shown in Fig. 3. In order to make the method easier to follow, the booking requests have been added to the table along with arrows to show how the table is filled. It can be seen that the second, third and eleventh booking requests are rejected and all other requests are accepted. That is the same as when we applied theft-nesting control policy.

Table 2
Applying the theft nesting booking policy Example 1

Booking requests	2	3	3	2	2	2	2	2	1	1	1
pl_i	1	7	7	1	1	1	1	1	0	0	0
$S(t)$	8	7	7	7	6	5	4	3	2	1	0
Decision A = Accept R = Reject	A	R	R	A	A	A	A	A	A	A	R

2.7. Equivalency of methods

We show now that theft nesting and standard nesting policies are equivalent to our proposed L and R booking policies respectively. As discussed earlier, the advantage of L and R booking policies over theft and standard nesting policies is that they are easier to visualize and allow for the use of a MC model to model the process of filling the aircraft, which will be described in Section 4.

Below we state the results formally.

Theorem 1. *Let C, $s(t)$ and n represent capacity of the cabin, remaining capacity of the cabin and number of fare classes, respectively. Given a sequence of requests where each request is for fare class $1,2,\ldots,n-1$ or n, the theft-nesting policy and the L booking policy defined above both make the same accept/reject decision to each and every one of the request in the sequence.*

Proof. After t bookings have been made, the number of columns of the nesting table with no entry indicates the remaining capacity of the cabin, i.e. $s(t)$. In L booking policy, as the requests are accepted columns of the nesting table will be filled up one by one from the left side of the nesting table. Consider the ith row of the nesting table. pl_i right cells of this row are crossed out. So, initially, $C-pl_i$

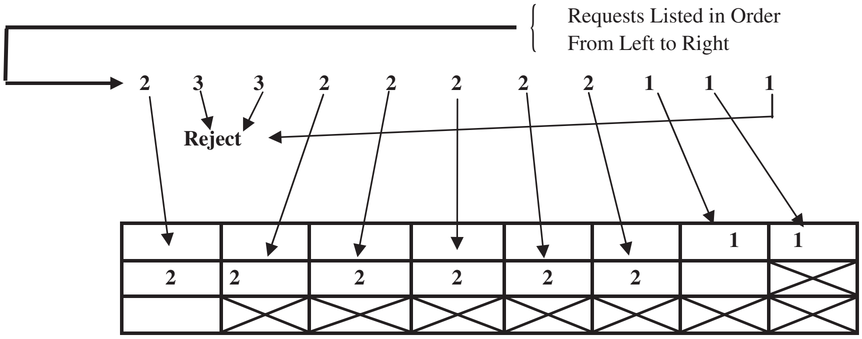

Fig. 3. Applying L Booking Policy to Example 1.

of the left cells of this row are not crossed out. In fact, at any time during the process if $s(t)$ is larger than pl_i that means there are some cells, $s(t)-\mathrm{pl}_i$ to be exact, on the ith row that are not crossed out and the corresponding column has no other entry, so based on L booking policy these cells can be filled with a request for fare class i. However if $s(t) = \mathrm{pl}_i$ then there is no cell on row i which is both not crossed out and with no entry on its corresponding column, therefore based on L booking policy a request for fare class i can no longer be accepted. Once a request for fare class i is accepted, the first available column from the left side of the nesting table will be filled up, which means $s(t)$ is decremented by one unit.

In summary, the condition for accepting a request for fare class i in L booking policy, given that t requests have been accepted, is $s(t) > \mathrm{pl}_i$. Moreover, once a request is accepted, $s(t)$ will be decremented by one unit. It is easy to see that these are consistent with the condition and updating rule in theft-nesting policy. Suppose that in an airplane allocation problem with given cabin capacity and given set of protection levels, a request for fare class i arrives. Both L booking policy and theft-nesting policy will make the same accept/reject decision when applying to this problem. Also, these two methods both update the system in the same way, which means the remaining capacity of the cabin after accept/reject decision will be the same no matter which policy was applied. Therefore if these two policies are applied to the same sequence of requests where each request is randomly chosen from fare classes $1,2,\ldots,n-1$ or n, the same accept/reject decision will be made for each and every of the requests of the sequence. □

Theorem 2. *Let C, s(t) and n represent capacity of the cabin, remaining capacity of the cabin and number of fare classes respectively. Given a sequence of requests where each request is for fare class $1,2,\ldots,n-1$ or n, the standard nesting policy and the R booking policy defined above both make the same accept/reject decision to each and every one of the requests in the sequence.*

Proof. As before, let t denote the number of requests have already been accepted. Let $V_i(t)$ denote the number of columns with no entry whose corresponding cells on row i are not crossed out. At the beginning of the process, when no column is filled up, $V_i(0) = C - \mathrm{pl}_i$.

We will show by induction that,

$$0 \leqslant V_n(t) \leqslant V_{n-1}(t) \leqslant \cdots \leqslant V_1(t) = C \quad \text{for all } t. \tag{7}$$

From Eq. (2), it is clear that Eq. (7) holds for $t = 0$. Suppose it also holds for $1,\ldots,t$. If $V_i(t)$ is not zero, then based on R booking policy a request for fare class i will be accepted and the first right available cell on row i will be filled up. By the induction hypothesis, Eq. (7) holds and so we can say that if a cell on row i is not crossed out, then all the cells of the corresponding column on rows $1,2,\ldots,i-1$ are not crossed out either. However, the cells of the corresponding column on rows $i+1$, $i+1,\ldots,n$ may or may not be crossed out. Therefore, when a request for fare class i is accepted, it will affect $V_1,V_2,\ldots,V_i$, decreasing them by one, but it may or may not effect V_{i+1}, $V_{i+2},\ldots,V_n$. Notice that, when $V_i(t) > V_{i+1}(t)$, the column that is going to be filled up with the request for fare class i, has not been available to fare class $i+1$ any way, because the corresponding cell is crossed out on row $i+1$. However, when $V_i(t) > V_{i+1}(t)$ the column that is going to be filled up with the request for fare class i is also available to fare class $i+1$, because the corresponding cell on row $i+1$ is not crossed out. This concept can be seen in Fig. 4.

In summary, R booking policy accepts a request for fare class k if $V_i(t) > 0$ and, once the request is accepted, V_i's will be updated as following:

$$V_i(t+1) = \begin{cases} V_i(t) - 1 & \text{if } i \leqslant k, \\ V_i(t) - 1 & \text{if } i > k \text{ and } V_k(t) = V_i(t), \\ V_i(t) & \text{if } i > k \text{ and } V_k > V_i(t). \end{cases} \tag{8}$$

Based on the above updates, it is clear that Eq. (7) holds at $t+1$. It follows that the updates are valid for all t and these are consistent with Eq. (6).

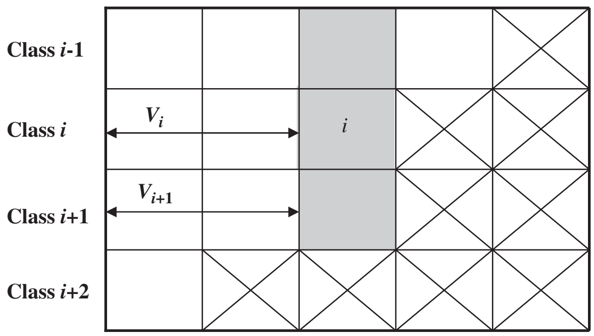

Fig. 4. Nesting table for R booking policy at some point in time.

Therefore, in an airplane allocation problem with given cabin capacity and given set of protection levels, when a request for fare class i arrives both R booking policy and standard nesting policy will make the same accept/reject decision when applying to this problem. Also, these two methods both update the system in the same way. In other words, V_i and R_i are updated in the exact same way for $i = 1, \ldots, n$. If these two policies are applied to the same sequence of requests where each request is randomly chosen from fare classes 1,2,…,n−1 or n, same accept/reject decision will be made for each and every of the requests of the sequence. □

3. Simulation

In order to examine the difference between R and L booking policies and therefore theft and standard nesting, we have simulated the process of applying R and L booking policies to four different examples of a single leg flight. The simulation results are presented in this section.

We assume that the booking requests follow homogeneous Poisson processes with known rates of arrivals. The booking period is the time from when the flight tickets are first put on sale until the flight time. We assume that the booking period is divided into equal time units, where one time unit is small enough so that there is at most one booking request per time unit (such an assumption is common in the literature). T denotes the booking period as a discrete multiple of time unit throughout the paper. As before p_i denotes the probability of arrival of fare class i in one time unit where class zero stands for no request in that time unit. The corresponding probability, p_0 can be calculated as $p_0 = 1 - \sum_{i=1}^{n} p_i$. Since the arrival process follows a homogeneous Poisson process, p_i's do not change over time. We used MATLAB 6.5 to code the simulation. The simulation process is described in the next paragraph.

First T is fixed. For the fixed T, 1500-sample sequences of booking requests, indexed by the fare class, are generated. As mentioned before, fare class zero stands for no request in the corresponding time unit. Each sample sequence is a vector of T elements, where each of the numbers is a sample from a discrete probability distribution with probability mass function $\{p_i\}$. For each sample sequence of booking requests, the process of applying R and L booking policies are simulated and the revenue generated by each one is calculated. The average of 1500 sample revenues is then computed for each R and L booking policies. The calculated revenue, basically, shows the average revenue that will be generated if we start selling the tickets T time units before the time of the flight. This process is repeated for different multiples of time unit (different T's). The average generated revenue versus T is then graphed for both R and L booking policies. All four examples are single leg flights. The parameters for the examples are given in Table 3.

As it can be seen from Figs. 5 to 8, up to a certain T the average revenue generated by R booking control policy is slightly greater than the one generated by L booking policy, but after that the average revenue generated by R booking control policy is exceeded by the one generated by L booking control policy. Even though capacity, number of fare classes, fare and probability of arrivals are different in the examples, the same behavior is observed. The same behavior was observed for hundreds of other examples as well which are not presented here. We provide some theoretical explanation for that phenomenon in the Appendix A.

4. Alternative model for calculating expected revenues

As mentioned in Section 1, one of the advantages of the L and R policies we propose is that they allow for the development of a Markov chain (MC) model for the booking process. This model, in turn, yields exact values for the expected revenue, in which case simulation is no longer necessary. Since the Markov chains for the R and L policies are different, we describe them separately below.

4.1. Markov chain model for L booking policy

For any single leg problem with capacity C, the proposed MC model has $C+1$ stages. There are C seats available at stage one, $C-1$ seats at stage two and finally no seat at stage $C+1$. In other words at stage i, $i-1$ seats are already taken and $C+1-i$ seats are available. Since in this model there is only one seat at each stage, we have only one state per stage, so the terms "stage" and "state" become interchangeable. Thus, from now on we refer to states rather than stages. One time unit is equivalent to one transition from state to state. The only possible transitions are from each state to itself or to the one after that. The first state of the proposed MC is the

Table 3
Parameters of four examples used in simulation process

Example	Capacity	Number of fare classes	Fare class	Fare ($)	Protection level	Probability of arrival
2	4	3	1	400	0	0.01
			2	200	1	0.02
			3	100	2	0.03
3	15	4	1	550	0	0.015
			2	400	3	0.03
			3	200	5	0.04
			4	75	10	0.05
4	50	6	1	610	0	0.015
			2	500	5	0.03
			3	450	7	0.04
			4	320	10	0.05
			5	200	15	0.06
			6	100	20	0.07
5	100	10	1	700	0	0.015
			2	640	5	0.03
			3	520	10	0.04
			4	410	15	0.05
			5	350	20	0.06
			6	300	25	0.07
			7	250	30	0.075
			8	200	40	0.08
			9	150	50	0.085
			10	100	60	0.09

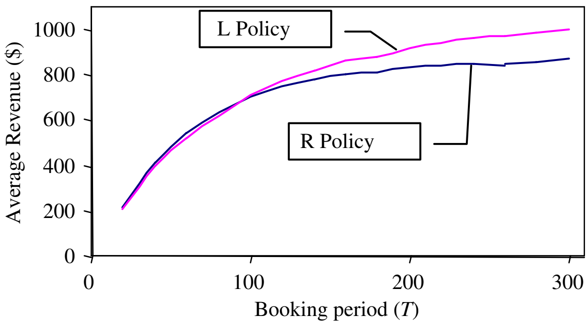

Fig. 5. Simulation of R and L booking policies for Example 2.

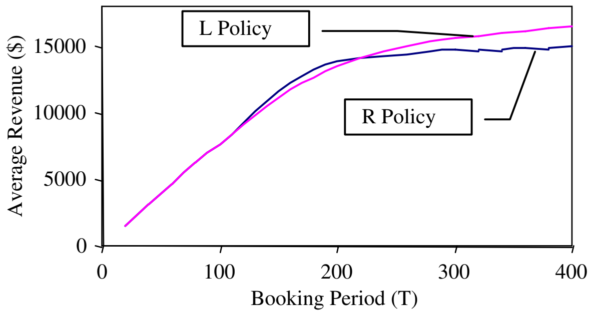

Fig. 7. Simulation of R and L booking policies for Example 4.

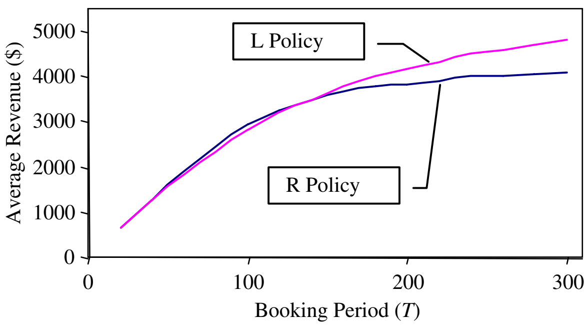

Fig. 6. Simulation of R and L booking policies for Example 3.

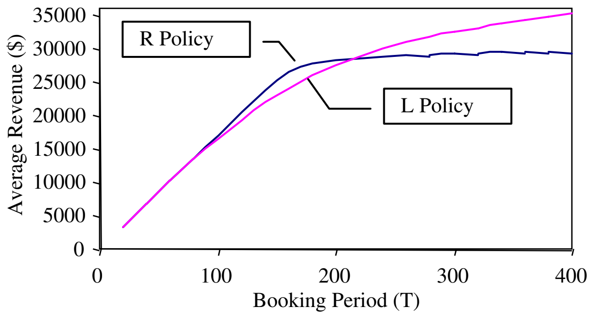

Fig. 8. Simulation of R and L booking policies for Example 5.

nesting table that we saw before. All other states are the nesting table with the filled columns eliminated from it. Therefore, the last state, which shows the end of the process, is the nesting table with all the columns filled and therefore eliminated. Since there are no more columns to be filled up at the last state, there is no transition from the last state. Therefore, the last state is an absorbing state.

The process always starts from the first state, which is the original nesting table with no column eliminated from it. Based on the L booking policy, the first available left column can be filled up with eligible fare classes, which are those whose corresponding rows are not crossed out. Since at any state the filled columns are already eliminated from the nesting table, the first available left column is actually the first left column, which can be filled with eligible fare classes. In that case, the column is filled up and transition to the next state occurs which is the previous table with its filled column eliminated from it. The corresponding transition probability is sum of all probabilities of eligible fare classes at that state. Self-transitions occur when there is either no booking request or no eligible one. The corresponding transition probability is then p_0 plus the sum of probability of all non-eligible booking requests at that point. Fig. 9 is the MC for L policy applied to the second example listed in Table 3. Transition probabilities are written over the transition arrows.

As mentioned before, the process always starts at the first state. The first left column of states one and two have no crossed out cells and can be filled up with any of the three booking requests in which case it will transit to the next state with transition probability of $p_1+p_2+p_3$. The first left column of the third state is not available to the third fare class and can be filled up with the first and second fare classes only. Therefore, once we reach at state three we will remain there until there is a request for fare classes 1 or 2 in which case transition to the fourth state occurs with the transition probability of p_1+p_2.

4.2. Markov chain model for R booking policy

The MC model for the R policy is similar to the one for L policy with larger number of states. There are again $C+1$ stages corresponding to the number of seats that are left. Transitions happen from each state to itself or to the states of the next stage only. Same as before, the probability of transition from a state to itself equals the probability of no request in one time unit plus the probability of requests for all the classes that have no seats left to fill. The first state of the proposed MC is the nesting table that we saw before. All other states are the nesting table with the filled columns eliminated from it. The last state, which shows the end of the process, is the nesting table with all the columns filled and therefore eliminated. Since there are no more columns to be filled up at the last state, there is no transition from the last state. Therefore, the last state is an absorbing state. The process always starts from the first state, which is the original nesting table with no column eliminated from it. Based on the R booking policy, the first available right column can be filled up with eligible fare classes, which are those whose corresponding rows are not crossed out. The first available right column can be the first, second or any other column of the nesting table depending upon the state of the system and the class of the request. That is why there is usually more than one state in the stages of the MC model of the R booking policy. The corresponding transition probability of transition from one state to another state of the next stage is the sum of all probabilities of eligible fare classes at that state. Fig. 10 is the MC for R policy applied to second example listed in Table 3. Transition probabilities are written over the arrows.

4.3. Calculating expected revenue

For any single leg flight, the process of applying L or R booking policy can be modeled by an absorbing MC as described before. Let P denote

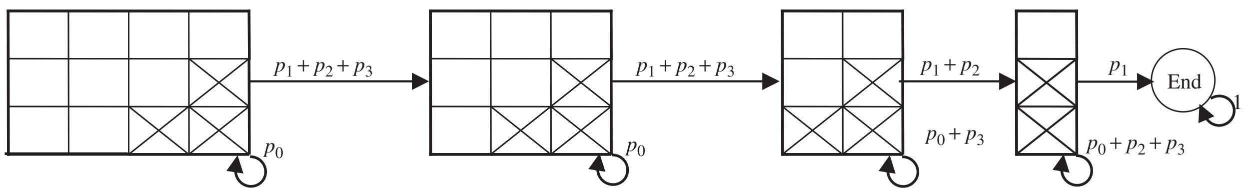

Fig. 9. Example 2 MC model when applying L booking policy.

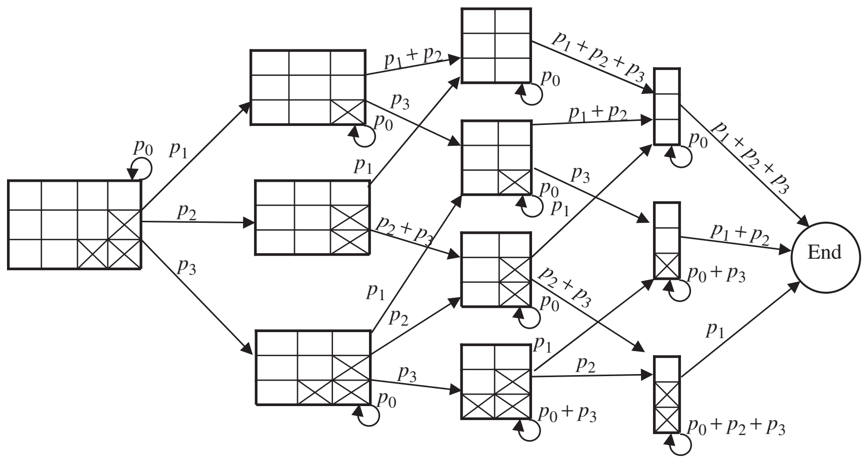

Fig. 10. Example 2 MC model when applying R booking policy.

the transition probability matrix of corresponding MC. As we mentioned before, each transition is equivalent to one time unit, therefore P^n shows the state of the MC after n transitions or n time units. In order to calculate the expected revenue generated by L booking policy, we need to define the value of transition. If no seat is sold, the value is 0; if a fare class i is sold, then the value of transition is r_i; if either a seat i or j is sold to make a transition, then the value is: $\hat{r}_{ij} = (r_i p_i + r_j p_j)/(p_i + p_j)$; if one of three seats is sold to make the transition, then the value is $\hat{r}_{ijk} = (r_i p_i + r_j p_j + r_k p_k)/(p_i + p_j + p_k)$, and so on. Let v_{be} be the value of being in state b and ending in state e after one transition. Then, we can calculate the expected value of a transition from any state b as follows: $\bar{v}_b = \sum_{e=0}^{n-1} v_{be} p_{be}$, where, p_{be} is the transition probability. Let V be the column vector with elements $\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_n$ and let π_k be the row vector of state probabilities after k transitions, with $k = 0$ for the initial state and $k = T$ when the plane takes off. Finally, let $\bar{\bar{r}}_k$ be the expected revenue from the kth transition. Thus,

$$\left.\begin{aligned} \bar{\bar{r}}_1 &= \pi_0 P V \\ \bar{\bar{r}}_2 &= \pi_1 P V = \pi_0 P^2 V \\ &\vdots \\ \bar{\bar{r}}_T &= \pi_{T-1} P V = \pi_0 P^T V \end{aligned}\right\} \text{and the expected revenue is}$$

$$\sum_{k=1}^{T} \bar{\bar{r}}_k = \pi_0 \left(\sum_{k=1}^{T} P^k \right) V. \tag{9}$$

The two graphs in Figs. 11 and 12 show the result of calculating expected revenue for the second example listed in Table 3 by using the MC method described above as well as the result of the simulation that we saw in the previous section. Both graphs show that the two methods get matching results, but there is no noise in the MC model.

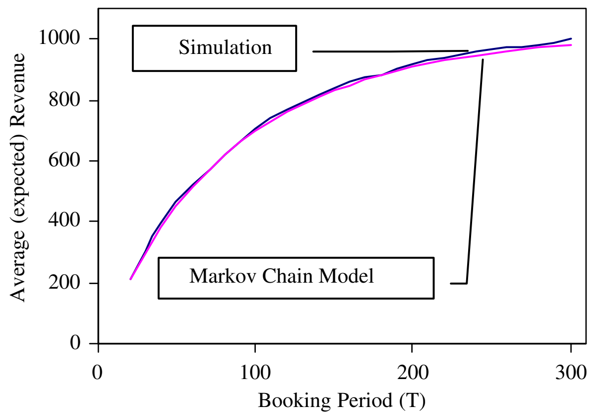

Fig. 11. Revenue generated by applying L booking policy to Example 2 using simulation and the Markov chain model.

4.4. Using the Markov chain model for optimization

Ideally, the airline wants not only to determine the best nesting policy but also the optimal protection levels. Thus, the goal becomes to incorporate nesting into the optimization procedure. Such an issue has been addressed in the literature (notably in Bertsimas and de Boer, 2005, and van Ryzin and Vulcano, 2003), and some algorithms have been proposed. Because of the

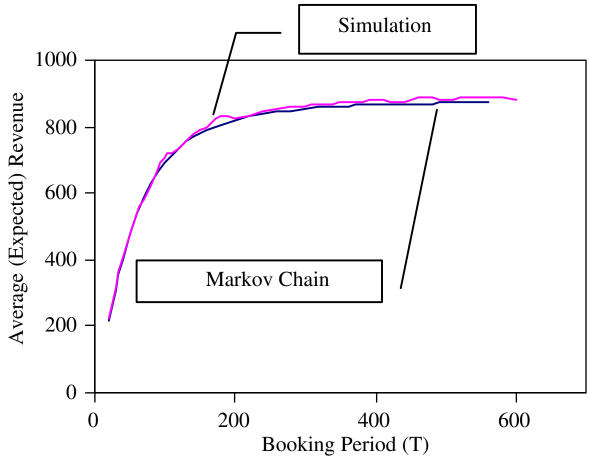

Fig. 12. Revenue generated by applying R booking policy to Example 2 using simulation and the Markov chain model.

stochastic nature of the problem, these algorithms implement some complex gradient-based simulation–optimization procedures.

Our MC model allows us to address that problem from a different perspective than the existing methods and algorithms. Suppose that for a given booking period (T), we want to look for the best set of protection levels and best policy. In other words, we would like to find out, at a certain amount of time before the flight, how many seats of each fare class need to be protected and what policy, L or R need to be applied in order to generate the maximum revenue. An important consequence of using the MC model is that, since it yields *exact* expected revenues, there is no need for simulation. Hence, the optimization problem becomes a completely deterministic combinatorial problem where the goal is to find the best allocation alternative and the best (between L and R) policy. Many techniques are available for such problems, e.g., simulated annealing, tabu search or cross-entropy, to name a few. A detailed investigation of that issue falls outside the scope of this paper, so for the purposes of illustrating its potential we discuss a small example that can be solved by enumeration.

Consider now the third example listed in Table 3. Notice that there are total of 816 possible set of protection levels. Therefore, there are total of 1632 possible allocation-policy combinations. Using our MC model, we calculated the expected revenue corresponding to each possible alternative. The results of this calculation for various values of T are given in Table 4. We can see that, as length of the booking horizon (i.e. T) grows, we tend to protect more seats from the lower-fare classes, since the likelihood of having enough high-fare customers to fill up the plane increases. Of course, this is a consequence of our assumption that the arrival process of each class is homogeneous Poisson; in practice, this is not the case, so our model is suitable for relatively small horizons where the rate of the Poisson process can be assumed to be constant. Note that we say "relatively small" since the appropriate length depends on factors such as the arrival pattern and the capacity of the plane. Thus, the numbers in Table 4 illustrate what happens under different scenarios, in which the given values of T are "relatively small".

Table 4
Total enumeration for Example 3 of Table 2

T	Best policy	Best protection level	Expected revenue ($)
30	L or R	(0, 0, 0, 0)	960
80	L	(0, 0, 0, 1)	2530
100	L	(0, 0, 0, 3)	3052
200	L	(0, 0, 3, 15)	4812
300	L	(0, 0, 12, 15)	5755
500	L	(0, 1, 15, 15)	6766
1000	L	(0, 11, 15, 15)	7590

It is important to notice that, although we could have simulated each of the possible 1632 alternatives for each T, the results would likely be inconclusive—since the simulation output contains random error, properly identifying the best alternative would require applying ranking and selection procedures (see, e.g., Law and Kelton, 2000), which for the number of alternatives at hand would be extremely time-consuming.

5. Conclusion and future study

We have studied in detail two nesting policies proposed in the literature and highlighted the differences between them. An important conclusion of our analysis is that there exists a "crossing point" for the booking horizon, i.e. if the horizon is long enough then the theft-nesting policy will be superior, whereas for shorter horizons either the standard nesting is better or they both yield the same result. This insight can help the practitioner decide which of the methods is more suitable in his or her case. A potential avenue for future work is the development of methods that can *identify* the crossing point. We believe that this cross point depends on capacity, protection level, fares and probabilities of arrival of fare classes, so another possible area for future study could be to examine how the cross point is related to each of these factors.

We have also developed a Markov chain model for the booking process that allows for exact calculation of expected revenues, which in principle allows for more efficient optimization of the protection levels since only deterministic techniques are required. For a large network, however, the number of states in the Markov chain can be very large; hence, a potential area for further research is to study how the size of the chain can be reduced, e.g., by eliminating or aggregating states. Finally, developing an efficient method to search for the best set of protection levels is another important topic, on which research is underway.

Appendix A. Properties of the expected revenue of the system in terms of policy used

We discuss now properties of the expected revenue of the system in terms of policy used for allocation (i.e. L or R).

For each $t = 1, 2, \ldots$, let D_t be a random variable denoting the class of the arrival at time t with $D_t = 0$ if no arrival occurs at t. Let R_t and L_t be random variables denoting the revenue obtained under, respectively, the R and L policies up to time t. Note that the specific values taken on by R_t and L_t depend on the values taken on by the random variables $D_1, D_2, \ldots, D_t$.

The following lemma shows an important relationship between R_t and L_t.

Lemma 1. *There exist random times τ_1 and τ_2, with $\tau_1 \leqslant \tau_2$, such that*

$$R_t \geqslant L_t \quad \text{if} \quad t \leqslant \tau_1, \tag{10}$$

$$R_t \leqslant L_t \quad \text{if} \quad t \geqslant \tau_2. \tag{11}$$

Proof. Consider a particular arrival stream, i.e. a specific sample path of the stochastic process $D = (D_1, D_2, \ldots)$. The key observation for the proof is the following fact: because of its "filling from the right" nature, up to a certain point, policy R allows for more choices than policy L to fill an empty slot in the nesting table. Indeed, the first $C - \mathrm{pl}_n$ requests are accepted under either policy. After that, the L method starts filling the table from the left, i.e. the columns with more blank. Thus, the options to fill the remaining columns with policy L will be narrower than with R, in the sense that policy R allows for more lower-fare arrivals to be accepted. This continues until the point where policy R rejects its first customer. Clearly, until that point policy R has accepted at least the same customers accepted by L. By calling this time τ_1, Eq. (10) follows.

On the other hand, all the slots filled by R booking policy customers that were rejected by L booking policy will be eventually filled by L booking policy if the arrival stream is long enough. This follows from the fact that each sample path of the stochastic process D contains infinitely many arrivals of every class, a property ensured by the assumption that the Poisson arrival process is homogeneous. Note that, unless both L and R accept exactly the same customers (an event whose probability is strictly less than 1), policy L will be able to improve upon the revenue obtained with R since it fills the slots with higher-fare customers. Let τ_2 be the time when the nesting table is filled by policy L. Then, Eq. (11) follows. □

Proposition 1. *There exists a deterministic time T_2 such that*

$$E[R_t] < E[L_t] \quad \text{if } t \geqslant T_2. \tag{12}$$

Proof. Let I denote the indicator function of an event, i.e. $I_{\{A\}} = 1$ if A occurs, $I_{\{A\}} = 0$ otherwise. Let us write the difference $E[R_t - L_t]$ as

$$\begin{aligned} E[R_t - L_t] = {} & E[(R_t - L_t)I_{\{t \leqslant \tau_1\}}] \\ & + E[(R_t - L_t)I_{\{\tau_1 < t < \tau_2\}}] \\ & + E[(R_t - L_t)I_{\{t \geqslant \tau_2\}}] \end{aligned} \tag{13}$$

and take $t \to \infty$. Note that, by Lemma 1, the first term on the right-hand side is non-negative and the third term is non-positive. In fact, the proof of Lemma 1 shows that, when $t \geqslant \tau_2$, not only $R_t \leqslant L_t$ but also $R_t < L_t$ on a set of positive probability. Thus, the third term is strictly positive. Since all terms in the integrand are bounded, it follows from the bounded convergence theorem that we can switch the limit and the expectations. Since both τ_1 and τ_2 are finite with probability one, we have that $I_{\{t \leqslant \tau_1\}} \to 0$, $I_{\{\tau_1 < t < \tau_2\}} \to 0$ and thus

$$\lim_{t \to \infty} E[R_t - L_t] = \lim_{t \to \infty} E[(R_t - L_t)I_{\{t \geqslant \tau_2\}}] < 0.$$

Thus, there exists T_2 such that Eq. (12) holds. □

Acknowledgments

The authors thank Lijian Chen and Senthil K. Balasubramanian for their suggestions and comments. This material is based upon work supported by the National Science Foundation under Grant No. DMI-0115385.

References

Belobaba, P.P., 1987. Airline yield management: An overview of seat inventory control. Transportation Science 21, 63–73.

Bertsimas, D., de Boer, S., 2005. Simulation-based booking limits for airline revenue management. Operations Research 53, 90–106.

Brumelle, S.L., McGill, J.I., 1993. Airline seat allocation with multiple nested fare classes. Operations Research 41, 127–137.

Boyd, E.A., Bilegan, I., 2003. Revenue management and e-commerce. Management Science 49, 1363–1386.

Coelli, T., Grifell-Tatje, E., Perelman, S., 2002. Capacity utilisation and profitability: A decomposition of short-run profit efficiency. International Journal of Production Economics 79, 261–278.

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

Dai, Y., Chao, X., Fang, S.-C., Nuttle, H.L.W., 2005. Pricing in revenue management for multiple firms competing for customers. International Journal of Production Economics 98, 1–16.

de Boer, S.V., Freling, R., Piersma, N., 2002. Stochastic programming for multiple-leg network revenue management. European Journal of Operational Research 137, 72–92.

Glover, F., Glover, R., Lorenzo, J., McMillan, C., 1982. The passenger mix problem in the scheduled airlines. Interfaces 12, 73–79.

Law, A.M., Kelton, W.D., 2000. Simulation Modeling and Analysis, third ed. Mc-Graw-Hill, New York.

Littlewood, K., 1972. Forecasting and control of passenger bookings. In: AGIFORS Symposium Proceedings, vol. 12, pp. 95–117.

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

Park, K., Piersma, N., 2002. Overview of OR techniques for airline revenue management. Statistica Neerlandica 56, 479–495.

Smith, B.C., Leimkuhler, J.F., Darrow, R.M., 1992. Yield management at American airlines. Interfaces 22, 8–31.

Sridharan, V., 1998. Managing capacity in tightly constrained systems. International Journal of Production Economics 56–57, 601–610.

Talluri, K., van Ryzin, G., 2004. The Theory and Practice of Revenue Management. Kluwer Academic Publishers, Dordrecht.

van Ryzin, G., Vulcano, G., 2003. Simulation-based optimization of virtual nesting controls for network revenue management. Working Paper, Graduate School of Business, Columbia University, New York.

Williamson, E.I., 1992. Airline network seat inventory control: Methodologies and revenue impacts. Ph.D. Thesis. Flight Transportation Laboratory, Massachusetts Institute of Technology, Cambridge, MA.

Wollmer, R.D., 1986. A hub-spoke seat management model. Unpublished Company Report, Douglas Aircraft Company, McDonnell Douglas Cooperation, Long Beach, CA.

Wollmer, R.D., 1992. An airline seat management model for a single leg route when lower fare classes book first. Operations Research 40, 26–37.