AN APPLICATION OF YIELD MANAGEMENT TO THE HOTEL INDUSTRY CONSIDERING MULTIPLE DAY STAYS

GABRIEL R. BITRAN

Massachusetts Institute of Technology, Cambridge, Massachusetts

SUSANA V. MONDSCHEIN

Universidad de Chile, Santiago, Chile

(Received May 1993; revisions received April, September 1994; accepted October 1994)

In this paper we study optimal strategies for renting hotel rooms when there is a stochastic and dynamic arrival of customers from different market segments. We formulate the problem as a stochastic and dynamic programming model and characterize the optimal policies as functions of the capacity and the time left until the end of the planning horizon. We consider three features that enrich the problem: we make no assumptions concerning the particular order between the arrivals of different classes of customers; we allow for multiple types of rooms and downgrading; and we consider requests for multiple nights. We also consider implementations of the optimal policy. The properties we derive for the optimal solution significantly reduce the computational effort needed to solve the problem, yet in the multiple product and/or multiple night case this is often not enough. Therefore, heuristics, based on the properties of the optimal solutions, are developed to find "good" solutions for the general problem. We also derive upper bounds which are useful when evaluating the performance of the heuristics. Computational experiments show a satisfactory performance of the heuristics in a variety of scenarios using real data from a medium size hotel.

In this paper we study optimal strategies for renting hotel rooms when a manager faces a stochastic and dynamic arrival of customers from different market segments, considering a fixed capacity and a finite planning horizon. We formulate the problem as a stochastic, dynamic programming model and characterize the optimal policies as functions of the capacity and the time left until the end of the planning horizon. We consider three features that enrich the problem: we make no assumptions concerning the particular order between the arrivals of different types of customers; we allow for multiple types of products and downgrading. This is the case when there exists a natural ordering of all products, so that every product is an acceptable substitute for those that are "worse" than it. Third, we consider requests for multiple nights.

This problem is motivated by the general yield management problem that arises in the hotel industry where managers have to decide the best use of the limited number of rooms. As defined by American Airlines (1987), the objective of yield management is "to maximize passenger revenue by selling the right seats to the right customers at the right time," which can be applied to our problem replacing seats by rooms.

Decisions are made at two levels when solving this problem. The first level is tactical: It is necessary to decide the maximum number of reservations for each market segment to accept at a given moment in time for a particular target day, i.e., given that the organization has already accepted a number of reservations at a certain point in time and considering future stochastic requests and cancellations, it is necessary to decide whether or not to accept a specific reservation request. Usually, customers can cancel their reservations without any penalty. Thus, they do not pay for the rooms until the actual date when they use them. Reservations are typically managed centrally by a corporate office which keeps hotel managers constantly informed about the status of reservations for a certain number of days ahead.

This tactical problem is closely related to the second level at which decisions are made, namely, the operational level. This paper focuses on the operational decisions, where every time a customer requests a room during the target date, the manager has to decide whether or not to rent it to that customer, considering the number of reservations made at the tactical level and the potential customers who will show up without reservations (*walk-ins*). When making this decision, the manager does not know how many additional customers will arrive that day or during the next few days, and whether there will be some arrivals for whom the room currently being requested has a higher payoff for the hotel. Special emphasis is given to the manager's ability to downgrade rooms, for example, the manager may decide to give a customer a suite for the price of a standard room when the latter is not available. The perishability of the product plays a central role in the decision process; rooms that are not rented have an opportunity cost close to zero. It is at the operational level that rooms are actually rented and revenues are collected.

Subject classifications: Dynamic programming, applications: renting hotel rooms in stochastic dynamic environments. Industries: hotel/motel: optimal sales strategies. Inventory/production, perishable/aging items: yield management with downgrading.
Area of review: SERVICES.

0030-364X/95/4303-0427 $01.25

At the operational level, the manager knows the total number of reservations within each market segment. However, the resulting number of reservations that turn into sales is a random variable because of the no-shows. Hotels usually have two types of reservations: the 6 p.m.-hold and guaranteed reservations. In the 6 p.m.-hold reservation case, a room is reserved until 6 p.m. and it costs nothing for the customer if she does not show up. On the other hand, in the guaranteed reservation case customers must frequently guarantee their reservations with a credit card. Despite the fact that, in practice, hotels cannot always force customers to pay for the room if they do not show up, the show-up rate is very high in this case. Therefore, because of the cancellations at the tactical level and no-shows at the operational one, managers usually overbook to maximize the total expected profit. When overbooking, they must take into account the tradeoffs of having idle capacity due to no-shows and cancellations, and turning down customers with the corresponding rejection costs.

One key factor for successful management in the hotel industry is market segmentation, as stated in Merliss and Lovelock (1980). Usually, a segmentation considers at least three categories: tourists, corporate travelers, and groups. However, some hotels have explored a "finer" classification that considers more than ten different categories; for example, pure transient, executive service plan, government workers, minivacations, corporate groups, and airlines (see Merliss and Lovelock). Each segment can be characterized by the price customers pay for the room, the type of room that they request, and the rejection costs associated with turning them down. In general, this cost is difficult to quantify because it is a combination of monetary and nonmonetary costs. For example, if a customer with a reservation for a hotel room is turned down, the rejection cost may be the monetary cost of allocating that customer to another hotel combined with the nonmonetary cost associated with the loss in trustworthiness for failing to honor a reservation.

Managers often use downgrading as a complementary method to match the limited capacity of the hotel with the uncertain demand experienced during the target date. Hotels usually have different types of rooms which differ in quality (size, furniture, service, etc.); for example, suites, deluxe, and standard rooms. Hence, there exists a natural order for the rooms where any room can be substituted for all those that are better than it. Therefore, downgrading adds a new degree of flexibility to the room assignment process.

Most of the literature that studies the tactical problem considers simplified dynamic relationships concerning what occurs during the target date to solve the operational problem. For example, some assume that managers have perfect information on arrivals during the target date, while others suppose that customers show up in a specific sequence (e.g., customers that pay less show up first). Among the papers that consider the problem at a tactical level, Ladany (1976) proposes a dynamic programming formulation for managing reservations in the hotel industry; at each stage a random number of reservation requests and cancellations are received. He assumes that during the target date, all the customers arrive at the same time. Hence, the manager can do perfect price discrimination at the operational level. Alstrup et al. (1986) study the booking policy for a single flight leg with two types of customers. They also assume that all customers arrive together during the target date. Bitran and Gilbert (1992) extend Ladany's analysis to the case where customers do not arrive at the same time, assuming a specific order of arrival among various types of customers. In practice, the arrival of customers from different market segments overlaps the target date and, therefore, managers must make renting decisions on a real-time basis, considering the tradeoffs discussed previously. The models presented in this paper consider this behavior as a central feature in the decision process.

Hotels generally assign capacity to group requests (tours, conventions, and others) a long time in advance of the target date. It is not unusual to book for conventions with more than six months anticipation. These decisions are made at a strategic level and, therefore, they only reduce the total capacity available at the tactical and operational levels. Given the risk involved in this large block of reservations, rooms usually have to be prepaid, which eliminates the problem of no-shows. Thus, as a good approximation of reality we consider only single room requests at the operational (or tactical) level.

We also consider that customers request rooms for several nights. Therefore, when making operational decisions, managers not only have to consider the arrival process for the remainder of the current day, but also the hotel occupancy rate for the next few days. For example, if the occupancy rate for the next two days is high, then it could be optimal to reject a lower fare customer who requests the room for three days, even though the room remains empty for the current night. Due to multiple night stays, managers need to know the number of reservations for several days ahead. For this purpose, the central reservation office keeps managers informed about the status of reservations for the next few days. However, managers have to rely on estimates of the total number of reservations for the days ahead because the central office can update them later in the week. Hotel managers agree that they can make good estimates of the number of reservations when considering a horizon of one week. We notice, however, that there are several situations where single night stays represent the majority of the customers. For example, airport hotels and motels (see Ladany).

In this paper, we develop renting policies that determine, at any moment in time, if the hotel should rent a room to a particular type of customer, and, if the answer is positive, what room should be rented. The arrival of

different types of customers is stochastic during the target day, and no assumptions concerning the particular order between the arrival of different types of customers is made.

For the sake of clarity, we begin with the simpler case (the single-night case) as a building block for the more general situation where customers request rooms for several nights. We present three models for the single-night case that differ in the degree of complexity in terms of the assumptions considered. The first model determines the optimal policy for renting rooms when there is only one type of room, and there are no reservations made at the beginning of the target date (customers show up without making reservations). Then we extend the results for the case of multiple types of rooms with the possibility of downgrading. These two models provide a basis for building the third model that considers multiple types of rooms and a given number of reservations at the beginning of the target day.

We derive an optimal selling policy for this problem, which has a number of interesting properties. It is characterized by a collection of capacity threshold vectors that evolves over time (every vector in the set has as many components as the number of products considered). It is optimal to satisfy a customer's request as long as the actual capacity vector at the moment the request is made has all components larger than or equal to at least one of the vectors in the current set of threshold capacities. Furthermore, the collection of capacity threshold vectors evolves over time in such a way that, for every class of customers and every actual capacity vector, there exists an instant in time beyond which it is optimal to satisfy the customer's request.

The computational effort required to solve the problem can be reduced significantly using the properties described above, making problems with one type of room tractable. However, when real size problems with several types of rooms are considered, the computational burden becomes enormous, even with this approach. Ad hoc heuristics are developed to handle those cases.

Due to the interaction between the tactical and operational levels when considering requests for several nights, we propose a heuristic based on the results obtained for the single-night case. This heuristic has a satisfactory performance compared to an upper bound developed for this case. For the computational experiments we use real data from a medium size hotel.

The remainder of this paper is organized as follows. In Section 1 we analyze the single-night case, where customers request rooms for a single night. We introduce a dynamic and stochastic programming formulation and characterize the optimal policy for renting rooms. We introduce a family of upper bounds for the optimal solution. These bounds are useful when comparing various heuristics to solve the general problem. Finally, we describe the heuristics developed to solve the general optimization problem and present computational experiments that illustrate their performance. In Section 2 we extend the model to the general case with multiple night stays. We describe a heuristic to solve this problem and an upper bound to measure the performance of the heuristic. Computational experiments are presented using real data. Finally, in Section 3, we present the conclusions and extensions.

1. SINGLE NIGHT CASE

In this section, we develop the case where customers stay at the hotel for a single night as a building block for the multiple-night case. However, it is worth noting that the single-night case applies, for example, to hotels in airports, and often to motels.

1.1. Mathematical Formulation

In this section, we present the mathematical formulation of the problem of finding the optimal policy for renting hotel rooms to various classes of customers. We specify three different models, which vary in the degree of complexity in terms of the number of product types and customer classes. The first model considers multiple classes of customers that request a single type of room at different prices. The second model incorporates multiple types of rooms with the possibility of downgrading. Finally, we extend the latter model to the case where reservations are allowed prior to the target date.

The Poisson process is a natural model to represent the arrival process of customers to the hotel. This process is commonly used in the literature; for example, Alstrup et al. claim that airline ticket requests have Poisson distribution. Rothstein (1974) and Bitran and Gilbert (1992) use a Poisson distribution to represent customer requests in the hotel industry. Generally, the arrival rate of customers from each market segment varies during the day. For example, few customers with 6 p.m.-hold reservations arrive after 6 p.m. Therefore, we consider a general, nonhomogeneous Poisson process to describe the arrival process of customers with arrival rates that depend on the market segment and time.

A special consideration is incorporated for those market segments that have reservations because the total number of arrivals is bounded by the total number of reservations. Therefore, we use a truncated Poisson process to describe the arrival process from these customer classes where the interarrival time is exponential and there are no more arrivals in the cases when all customers show up.

In what follows, we present a stochastic and dynamic programming formulation, where the stages correspond to the periods in which the planning horizon is divided. The state of the system, at a given time, is determined by the current capacity, the class of the customer requesting a room, and the number of pending reservations (if reservations are allowed). We divide the planning horizon into time intervals small enough so that the number of

arrivals in each interval is either zero or one. This approximation is based on an exact formulation which considers that decisions are only made at the random instants when customers arrive. Due to the complexity of solving this continuous-time problem, we formulate an approximate model where decisions are made at discrete intervals of time. By reducing the length of the time intervals, we can get as close as we want to the continuous-time formulation.

Products are classified into ordered types $s \in \{1, \ldots, m\}$, where product 1 is the "best product" (product 1 can substitute for all other products) and m is the "worst." We assume that the cost of providing the rooms is independent of the room type, i.e., the cost of cleaning and customer service is constant for all types of rooms. We also assume that this cost is smaller than the prices that customers pay for the rooms, for all classes of customers. However, the general case, with costs that depend on the type of rooms is easy to incorporate in the model.

There are n classes of customers; in the general case, a class is determined by the product type requested by the customer, the price that she pays, the rejection cost, and whether or not the customer has a reservation. The price can also include the profits associated with consumption during the customer's visit to the hotel. Several classes of customers can request the same type of product. Therefore, we define A_s as the set of all types of customers that request product s. Before presenting the mathematical models, we introduce the following notation:

- π_i: the price associated with a class i customer;
- c_i: the rejection cost associated with a class i customer;
- C: the total number of rooms; for the case of multiple types of rooms, C is an m-dimensional vector with $C(s)$ equal to the number of available class s rooms;
- T: the planning horizon, i.e., the length of time in which the reception desk is open during the target day;
- R: the vector with pending reservations for each class of customer;
- e_p: the vector with a one in the pth position and zeros elsewhere;
- $E_x[f(x, y)]$: the expected value of $f(x, y)$ with respect to the random variable x.

1.1.1. Model Without Reservations

In this simplified model, we consider that all customers show up at the hotel during the target day without reservations. We first consider the single-type room case, and later we extend the results to the multiple types of rooms case where downgrading is possible.

The Single-Product Case. We define the function $F_t(c, i)$ as the maximum expected profit from period t onwards if there are c available rooms and a class i customer requests a product at time t ($i = 0$ corresponds to no arrivals). Without loss of generality we assume that the planning horizon is divided in time intervals of length equal to one. Therefore, the mathematical formulation corresponds to the following.

Problem SP(C)

$$E_i[F_0(C, i)],$$

where if $c > 0$, $i \neq 0$:

$$F_t(c, i) = \max\begin{cases}\text{Rent a room:} & \pi_i + E_j[F_{t+1}(c-1, j)], \\ \text{Reject the customer:} & -c_i + E_j[F_{t+1}(c, j)].\end{cases}$$

If $c = 0$, $i \neq 0$:

$$F_t(c, i) = -c_i + E_j[F_{t+1}(c, j)].$$

If $i = 0$:

$$F_t(c, i) = E_j[F_{t+1}(c, j)].$$

Boundary conditions:

$$F_T(c, i) = \begin{cases} 0 & \text{if } i = 0, \\ \pi_i & \text{if } i \neq 0 \text{ and } c > 0, \\ -c_i & \text{if } i \neq 0 \text{ and } c = 0. \end{cases}$$

Hence, SP(C) is equal to the maximum expected profit during the planning horizon, considering that the initial capacity is equal to C. Given an arrival and the current capacity, at every time period the model gives the optimal policy of whether or not to accept the request.

In what follows, we characterize the optimal policy given by SP(C). The producer faces a nontrivial decision only when there is a positive capacity and someone requests a room. The objective function in this case is given by,

$$F_t(c, i) = \max\{\pi_i + E_j[F_{t+1}(c-1, j)], -c_i + E_j[F_{t+1}(c, j)]\}.$$

Hence, the hotel rents the room if

$$\pi_i + c_i \geq E_j[F_{t+1}(c, j)] - E_j[F_{t+1}(c-1, j)],$$

and rejects the customer if

$$\pi_i + c_i < E_j[F_{t+1}(c, j)] - E_j[F_{t+1}(c-1, j)].$$

Defining

$$\alpha_t(c) = E_j[F_{t+1}(c, j)] - E_j[F_{t+1}(c-1, j)],$$

we obtain that the hotel's policy for a class i request at time t, given a capacity c, is to rent the room if $\pi_i + c_i \geq \alpha_t(c)$ and reject the customer if $\pi_i + c_i < \alpha_t(c)$.

The following proposition shows that associated with every instant in time t, there exists a threshold C^*_{it}, such that a class i customer is accepted as long as the current capacity is larger than or equal to this threshold. The threshold C^*_{it} is determined by equating the expected profits from accepting and rejecting the customer's request, or equivalently: $\alpha_t(C^*_{it}) = \pi_i + c_i$. The threshold

C^*_{tt} decreases when the opportunity cost of accepting the customer's request, $\pi_t + c_t$, increases, i.e., if, for a given capacity, the optimal decision is to accept a specific type of customer, then it is also optimal to accept any type of customer with a higher opportunity cost.

Proposition 1. *The function $\alpha_t(c)$ is a nonincreasing function of c:*

$\alpha_t(c-1) \geq \alpha_t(c)$.

Proof. See Appendix A. This is a particular case of Proposition 3.

The following proposition shows that, for a given capacity c and type-i customer, there exists a threshold time τ^*_{ic} beyond which the customer's request is accepted. The threshold τ^*_{ic} is obtained by equating $\alpha_t(c)$ and $\pi_t + c_t$. As we show in Proposition 2, $\alpha_t(c)$ is a nonincreasing function of t. Hence, if we take two classes of customers i and j with $\pi_t + c_t \geq \pi_j + c_j$, then $\tau^*_{ic} \leq \tau^*_{jc}$, i.e., the hotel rents the room to customers with a higher opportunity cost before renting it to customers that yield a lower opportunity cost.

Proposition 2. *The function $\alpha_t(c)$ is decreasing in t:*

$\alpha_t(c) \geq \alpha_{t+1}(c)$.

Proof. See Appendix A. This is a particular case of Proposition 4.

The simple structure of the optimal policy can be summarized as follows. For a given instant in time and customer class, there exists a capacity threshold, such that a request within that class is accepted as long as the current capacity is larger than the threshold. Similarly, there exists a time threshold for every capacity and customer class, such that a request within the class that happens after this time threshold is always accepted.

The main characteristic of the optimal policy is its simplicity. This feature has an important consequence in terms of its practicality, because, in general, managers are more open to incorporate rules that are intuitive and easy to verify.

The Multiple-Product Case. The next model corresponds to the case of multiple types of rooms with the possibility of downgrading. We define $F_t(c, i)$ as the maximum expected profit from period t onwards when there are c available rooms and a class i customer arrives at time t($i = 0$ corresponds to no arrivals). In this case, c is a vector with as many components as room types. The mathematical formulation for this model corresponds to the following.

Problem MP(C)

$E_t[F_0(C, i)]$,

where, assuming that a class i customer requests a type s room, i.e. $i \in A_s$, we have: if $i \neq 0$ and there exists $c(k) > 0$ subject to $k \leq s$:

$$F_t(c, i) = \max\begin{cases}\text{Rent a room: } \max_{1\leq p\leq s}\pi_t + E_j[F_{t+1}(c-e_p, j)],\\ \text{Reject the customer: } -c_t + E_j[F_{t+1}(c, j)].\end{cases}$$

If $i \neq 0$ and $c(k) = 0$ for all $k \leq s$:

$F_t(c, i) = -c_t + E_j[F_{t+1}(c, j)]$.

If $i = 0$

$F_t(c, i) = E_j[F_{t+1}(c, j)]$.

Boundary conditions:

$$F_T(c, i) = \begin{cases}\pi_t & \text{if } \exists k \in \{1, 2, \ldots, k\} \text{ s.t. } c(k) > 0 \text{ and } i \neq 0,\\ -c_t & \text{if } \forall k \leq s,\ c(k) = 0 \text{ and } i \neq 0,\\ 0 & \text{if } i = 0.\end{cases}$$

We observe that in the multiple-product case the manager not only has to decide whether or not to accept the request, but also what room to rent. However, using the property that the objective function is a nondecreasing function of the capacity (see Lemma 1 in Appendix A) it is easy to prove that if the optimal decision is to rent a room to a class i customer who requests a type-s room, then the hotel must rent the "lowest level" available room the customer is willing to accept (this room is given by $p = \max\{1, 2, \ldots, s\}$ subject to $c(p) > 0$). That is, if the hotel decides to rent a room to a customer, it rents the room satisfying the customer's need that is closest to the room requested by the customer.

In what follows, we derive the properties that characterize the optimal policy. Similarly to the single-product case, the manager faces a nontrivial decision when a type-i customer requests room s and there is an available room that satisfies this request. Then the optimal decision is given by:

$$F_t(c, i) = \max\{\pi_t + E_j[F_{t+1}(c-e_p, j)], -c_t + E_j[F_{t+1}(c, j)]\},$$

where $p = \max\{1, 2, \ldots, s\}$ subject to $c(p) > 0$. Hence, the hotel rents the room if:

$$\pi_t + c_t \geq E_j[F_{t+1}(c, j)] - E_j[F_{t+1}(c-e_p, j)]$$

and rejects the customer otherwise.

Defining $\alpha_t(c, p) = E_j[F_{t+1}(c, j)] - E_j[F_{t+1}(c-e_p, j)]$, the optimal policy consists of renting the room if $\pi_t + c_t \geq \alpha_t(c, p)$, and rejecting the customer if $\pi_t + c_t < \alpha_t(c, p)$. The following two propositions characterize the optimal policy.

Proposition 3. *The function $\alpha_t(c, p)$ is nonincreasing as a function of c:*

$\alpha_t(c-e_l, p) \geq \alpha_t(c, p)$ for all c, p, l, t.

Proof. See Appendix A.

Proposition 4. *The function $\alpha_t(c, p)$ is nonincreasing as a function of t:*

$\alpha_t(c, p) \geq \alpha_{t+1}(c, p)$ for all c, p, t.

Proof. See Appendix A.

We illustrate the properties described above with the following example. Consider a hotel that has two types of rooms, where type-1 rooms can substitute for type-2 rooms. For a given period of time, and a given class i of customers, Figure 1 shows the collection of capacity threshold vectors. This corresponds to the curve that separates the acceptance and the rejection regions. If the current capacity is in the acceptance region, the manager accepts the class-i request. Otherwise, she rejects the request. Assuming that class-i customers request a type-2 room, downgrading only takes place when there is no type-2 room available in the acceptance region. As time goes by, the curve that corresponds to the set of capacity threshold vectors moves down, and the rejection region becomes smaller.

The two preceding propositions show how the results obtained for the single-product case extend to the multiple-product case with downgrading. The optimal policy for renting rooms in this case is characterized by a collection of capacity threshold vectors that evolves over time (every vector in the set has as many components as the number of room types considered). It is optimal to satisfy a customer's request as long as the capacity vector at the moment the request is made has components larger than or equal to at least one vector in the current set of threshold capacities. Furthermore, the collection of capacity threshold vectors evolves over time in such a way that, for every class of customers and every actual capacity vector, there exists an instant in time beyond which it is optimal to satisfy the customer's request.

Proposition 3 implies that if a customer's request is accepted when the capacity is equal to $c - e_l$, then it is also accepted when the capacity is equal to c, for all values of c and l. Suppose that the request is satisfied with a type-p room when the capacity is $c - e_l$. When incrementing the capacity to c this request will be satisfied with the same type-p room or with a type-l room if this room is closer to the customer's need. This property is obtained by observing that:

$\alpha_t(c, p) \geq \alpha_t(c, l)$ for all $p \leq l$.

The above inequality uses the fact that the function $F_t(c, i)$ is nonincreasing in c (see Lemma 1 in Appendix A).

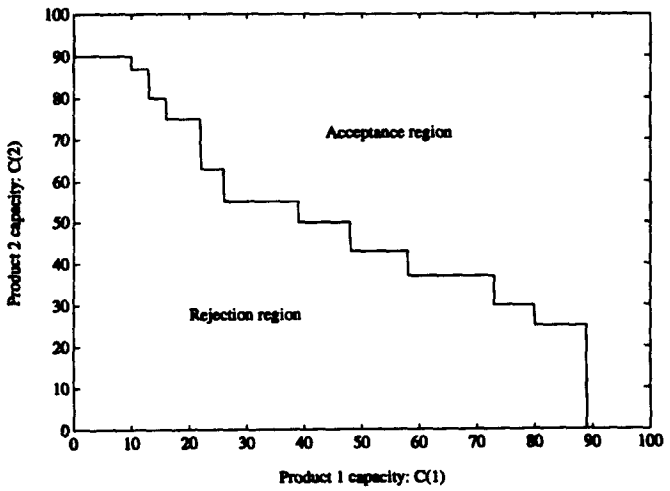

Figure 1. Example of a collection of capacity threshold vectors.

1.1.2. Model With Reservations

In this subsection, we solve the operational problem that arises during the target date where the manager knows the total number of reservations within each market segment. The demand associated with these customers is stochastic because some of them do not show up, even though they have reservations. Additionally, during the target date, the hotel receives requests from walk-ins, i.e., customers who show up without making a reservation.

At every instant in time, the demand associated with customers who have reservations depends on the number of pending reservations, i.e., the number of reservations that have not been requested. Thus, in this case, the dynamic, stochastic programming model includes an additional state variable that corresponds to the number of pending reservations R, which has as many components as the number of customer classes.

We define $F_t(c, R, i)$ as the maximum expected profit from period t onwards, if the hotel starts at period t with c available rooms, R pending reservations, and a class-i customer requests a room. We recall that the customer class determines the price, the rejection cost, the room that the customer requests, and whether or not she has a reservation. We define δ_i to be equal to 1 when class i corresponds to customers with reservations and δ_i is equal to 0 otherwise.

Considering the general multiple product case, the mathematical formulation is equal to the following problem.

Problem MPR(C, R)

$E_i[F_0(C, R, i)]$,

where, assuming that a class-i customer requests a type-s product, i.e. $i \in A_s$, we have: if $i \neq 0$ and there exists $c(k) > 0$ subject to $k \leq s$:

$$F_t(c, R, i) = \max \begin{cases} \text{Rent a room:} \\ \quad \max_{1 \leq p \leq s}(\pi_i + E_j[F_{t+1}(c - e_p, R - \delta_i e_i, j)], \\ \text{Reject the customer:} \\ \quad -c_i + E_j[F_{t+1}(c, R - \delta_i e_i, j)]. \end{cases}$$

If $i \neq 0$ and $c(k) = 0$ for all $k \leq s$:

$F_t(c, R, i) = -c_i + E_j[F_{t+1}(c, R - \delta_i e_i, j)]$.

If $i = 0$:

$F_t(c, R, i) = E_j[F_{t+1}(c, R, j)]$.

Boundary conditions:

$$F_T(c, R, i) = \begin{cases} \pi_i & \text{if } \exists k \in \{1, 2, \ldots, k\} \text{ subject to } c(k) > 0 \text{ and } i \neq 0, \\ -c_i & \text{if for any } k \leq s,\ c(k) = 0 \text{ and } i \neq 0, \\ 0 & \text{if } i = 0. \end{cases}$$

Similarly to the previous cases, we define:

$$\alpha_t(c, p, R) = E_j[F_{t+1}(c, R, j) - F_{t+1}(c - e_p, R, j)].$$

The hotel rents a room to a class-i customer if $\pi_i + c_i \geq \alpha_t(c, p, R - \delta_i e_i)$, and rejects the customer otherwise.

The following proposition shows that, similarly to the case without reservations, for a given customer class, an instant in time, and a number of pending reservations, there is a set of capacity threshold vectors such that the customer is accepted as long as the current capacity vector has all the components larger than or equal to at least one threshold vector.

Proposition 5. *The function $\alpha_t(c, p, R)$ is nonincreasing as a function of the capacity:*

$$\alpha_t(c, p, R) \leq \alpha_t(c - e_l, p, R) \quad \text{for all } c, t, R, p, l.$$

Proof. See Appendix A.

We are not able to extend the result that the function $\alpha_t(c, p, R)$ is decreasing as a function of time using the same proof that we used for the case without reservations, because the number of pending reservations varies as a function of the current arrival. For the particular case where the probability of an arrival in the next unit of time depends on the number of pending reservations but not on t, we can obtain the result that $\alpha_t(c, p, R)$ is decreasing as a function of time. This proof is done by induction using the hypothesis:

$$F_t(c, R, i) - F_t(c - e_p, R, i) \geq F_{t+1}(c, R, i) - F_{t+1}(c - e_p, R, i).$$

Using the assumption that the probability of an arrival in the next unit of time depends only on the number of pending reservations, we take the expected value on both sides of the induction hypothesis, and obtain the desired result. We conjecture that the previous result still holds for the case where the arrival probabilities depends on t. However, new approaches to prove this result are needed.

The optimal policy for the general problem (multiple products, several types of customers and reservations) can be found by solving the dynamic programming formulation. However, this approach takes too long for most "real size problems." The computational effort can be reduced significantly using the properties that characterize the optimal policy derived in this section, making problems with one product tractable. However, when real size problems with several types of products are considered, the computational burden becomes intractable even with this approach. Therefore, it is necessary to develop ad hoc heuristics that provide good approximations for the optimal policy. In the next subsection, we present a family of upper bounds that will be useful to measure the quality of these heuristics.

1.2. Upper Bounds

In this subsection, we describe a family of upper bounds for the optimization problem described in subsection 1.1, considering multiple types of rooms and reservations. These upper bounds are based on the general idea that expected optimal profits increase when the amount of information available about future demand becomes larger. For example, if at the beginning of the planning horizon the manager knows the total number of requests that will take place, she can make a better decision than she could without this information. Before presenting the model for the upper bound, we define the following additional notation:

- R^k: the vector of pending reservations at the beginning of period k;
- $s^k(R)$: the vector with arrivals in period k if the initial vector of reservations is R; the component $s_i^k(R)$ is a random variable that represents the number of class-i requests in period k;
- B_p: the set of customer classes that request a room type better than or equal to p;
- x^k: the vector of decision variables; the component x_i^k corresponds to the number of class-i customers accepted in period k.

The following problem provides an upper bound for the problem **MPR**(C, R) (multiple-product case with reservations).

Problem $\mathbf{UB_K}(C^1, R^1)$

$$E_{s^1(R^1)}[G_1(C^1, R^1, s^1(R^1))],$$

where

$$G_k(C^k, R^k, s^k(R^k)) = \max_{x_1^k, \ldots, x_n^k} \sum_i \pi_i x_i^k - \sum_i c_i(s_i^k(R^k) - x_i^k) + E_{s^{k+1}(R^{k+1})}[G_{k+1}(C^{k+1}, R^{k+1}, s^{k+1}(R^{k+1}))]$$

subject to

$$x_i^k \leq s_i^k(R^k), \quad \text{for all } i = 1, \ldots, n, \tag{1}$$

$$\sum_{i \in B_p} x_i^k \leq \sum_{i=1}^{p} C^k(i), \quad \text{for all } p = 1, \ldots, m, \tag{2}$$

$$R^{k+1} = R^k - f_1(s^k(R^k)), \tag{3}$$

$$C^{k+1} = C^k - f_2(x^k), \tag{4}$$

$$x_i^k \geq 0 \text{ and integer}, \quad \text{for all } i = 1, \ldots, n. \tag{5}$$

The boundary condition is given by:

$$G_K(C^K, R^K, s^K(R^K)) = \max_{x_1^K, \ldots, x_n^K} \sum_i \pi_i x_i^k - \sum_i c_i(s_i^K(R^K) - x_i^K)$$

subject to

$$x_i^K \leq s_i^K(R^K), \qquad \text{for all } i = 1, \ldots, n,$$

$$\sum_{i \in B_p} x_i^K \leq \sum_{i=1}^{p} C^K(i), \qquad \text{for all } p = 1, \ldots, m,$$

$$x_i^K \geq 0 \text{ and integer}, \qquad \text{for all } i = 1, \ldots, n.$$

The maximum number of customers that can be accepted in period k is bounded by the number of requests received in that period. This condition is given by constraint (1) for each class of customers. The set of constraints (2) corresponds to the capacity constraint for each class of room, considering that they can be downgraded. Constraints (3) are conservation constraints for the number of pending reservations, i.e., $f_1(s^k(R^k))$ is equal to zero for all components associated with customers without reservations and equal to the corresponding coordinate of $s^k(R^k))$ for all components associated with customers who have reservations. Finally, (4) are conservation constraints for the capacity. Thus, the jth component of $f_2(x^k)$ is equal to the number of type-j products sold in period k considering that some of them were downgraded.

The index K of the family of upper bounds is given by the number of periods in which the planning horizon is divided. To prove that the problem $\mathbf{UB_K}(C, R)$ provides an upper bound for $\mathbf{MPR}(C, R)$ we have to apply (several times) the property that the maximum of the expected value is less than or equal to the expected value of the maximum. Instead of giving a formal proof, we explain the intuition underlying these upper bounds. The problem $\mathbf{UB_1}(C, R)$ corresponds to the extreme case where the manager has perfect information about the number of arrivals for each class of customers in the planning period. Therefore, she can make a better decision than she could with less information. In problem $\mathbf{UB_K}(C, R)$ we have divided the planning horizon in K periods. At the beginning of every period the manager has perfect information about the customers who show up in that period. Hence, she can make a better decision than she could knowing the current arrival and the probability of potential arrivals in the future. In the limit, when K is equal to the number of periods considered in the original problem (periods with at most one arrival), the upper bound is equal to the optimal solution of $\mathbf{MPR}(C, R)$.

1.3. Heuristics

In this subsection, we develop heuristics for finding "good" feasible solutions for the optimization problems formulated in subsection 1.1. These heuristics are based on the characterization of the optimal policies studied in subsection 1.1.

We introduce some additional notation:

$D_i = \{j/\pi_j + c_j > \pi_i + c_i\}$;
$pref(i)$ = the product type that a class-i customer requests;
c = the vector of current capacity;
$\lambda_i(t)$ = the arrival rate of class-i customers at time t;
T = the planning horizon;
t = the current time.

1.3.1. Heuristic 1.1

This heuristic works as follows. If at time t a customer requests a room, the manager calculates the expected number of arrivals that have an opportunity cost (price plus rejection cost) larger than the current request's opportunity cost, from t to the end of the planning horizon. The current capacity is optimally assigned to this expected number of arrivals. If after the assignment there is a remaining room that satisfies the current request, then the room is rented to the customer. Otherwise, the manager rejects the request.

To optimally assign the rooms to the expected number of arrivals, the heuristic first satisfies the expected demand of customers with the largest opportunity cost, downgrading if necessary. Then it satisfies the expected demand associated with the second largest opportunity cost, and so forth. Therefore, the heuristic assigns rooms to the customers with the highest marginal values which is optimal considering that all types of rooms have the same cost for using them.

Description of Heuristic 1.1 (HEUR1.1)

Suppose that a class-i customer requests a type-s room at time t. The following algorithm is used to determine whether or not to accept the customer's request.

STEP 0. **(Initialization)** Cap = c.

STEP 1. Determine the remaining capacity after assigning the rooms to the expected number of requests from t to T that have a larger opportunity cost than the current request's opportunity cost. $Si = Di$.

STEP 1.1 $j = \max_{k \in S_i}\{\pi_k + c_k\}$.

STEP 1.2 Exp_arr = $\int_t^T \lambda_j(\tau)\, d\tau$.

STEP 1.3 $p = pref(j)$
If (Cap(p) > Exp_arr) then
Cap(p) = Cap(p) − Exp_arr
Exp_arr = 0
else
Exp_arr = Exp_arr − Cap(p)
Cap(p) = 0
endif
if ($p > 1$), then $p = p - 1$, go to Step 1.3
$S_i = S_i - \{j\}$,
if $S_i \neq 0$ go to 1.1,
else go to Step 2.

STEP 2. Check if there is an available room for the class-i request. $p = s$.

STEP 2.1 if (Cap(p) > 0) then
Optimal decision ← give product p to the current request.
go to Step 3
endif
if ($p > 1$) then $p = p - 1$, go to 2.1
else, Optimal decision ← reject the current request.

STEP 3. STOP.

1.3.2. Heuristic 1.2

This heuristic is similar to the one described above. The only difference is that instead of considering the expected number of arrivals from t to the end of the planning horizon, the heuristic considers a number of arrivals such that, with 90% probability, the actual number of arrivals is less than it. **HEUR1.2** is used to denote heuristic 1.2.

1.3.3. Heuristic 1.3

This heuristic is developed for the single-product case. It calculates the marginal expected profit of the room that satisfies the current request. If this marginal value is larger than the certain opportunity cost associated with the current request, the manager rents the room. She rejects the request otherwise.

To compute the marginal expected value, we consider the customers that have a larger opportunity cost than the current request's net profit. For those customers, we calculate the probability that the number of requests is larger than or equal to the current capacity. The weighted opportunity cost for those customers times the above probability corresponds to the expected profit for this marginal room.

Description of Heuristic 1.3 (HEUR1.3)

Suppose that there is a class-i request at time t. The following algorithm is used to determine whether or not to accept the customer's request.

STEP 0. **(Initialization)** Cap = C.

STEP 1. Compute the weighted opportunity cost

$$\text{Weighted\_prof} = \sum_{j \in D_i} (\pi_j + c_j) \times \lambda_j \Big/ \sum_{j \in D_i} \lambda_j.$$

STEP 2. Compute the probability that there are a more or equal number of requests than the current capacity.

$$Prob_{it} = \Pr\left\{ \sum_{j \in D_i} \text{class } j \text{ requests in } (T - t) \geq \text{Cap} \right\}.$$

STEP 3.

If ($Prob_{it} \times$ Weighted_prof $> \pi_i + c_i$) then
Optimal decision ← reject the current request
else
Optimal decision ← accept the current request
Endif.

STEP 4. STOP.

1.3.4. Heuristic 1.4

In this heuristic the manager satisfies all the requests as long as there is an available product to satisfy the customer's need. The decisions are made without considering any additional information. This simple heuristic is used only for the purpose of comparison with the other heuristics. **HEUR1.4** is used to denote heuristic 1.4.

1.4. Computational Experiments

In this subsection, we use Monte Carlo simulation to evaluate the performance of the heuristics described in subsection 1.3. We assume that customers arrive according to a Poisson process, and every time a customer requests a room, the manager makes a decision that follows the rules given by the heuristics. We simulate the arrival process during an operational day and compute the total profit obtained under the application of the different heuristics. Averaging the profits given by repeated simulations, we obtain a statistical estimate of the expected profit.

We consider three groups of computational experiments. The first two sets correspond to relatively small applications that allow us to compute the optimal solutions. The parameters used in the third application are based on information given by the manager of a medium size hotel in Santiago, Chile. In this real application, we are not able to compute the optimal solution, therefore we compare the performance of the heuristics with the upper bounds for the general optimization problem.

A statistical estimate of the value of the upper bounds is also computed using Monte Carlo simulation. Recall that the upper bounds correspond to the expected value of an optimization problem. The expectation is taken over all possible arrival combinations and therefore it is infeasible to compute its exact value.

The criteria to stop the simulations is given by a standard deviation of less than 0.1% of the expected value. In the experiments we compute two upper bounds. The first one, UB1, corresponds to the case where all the requests are known at the beginning of the planning horizon. In the second upper bound, UB2, the planning horizon is divided in two periods; the requests during each segment are known at the beginning.

In what follows, we define the booking index, OB, to measure the hotel's capacity as a function of the expected number of requests.

Single-product case: OB is equal to the total expected number of arrivals over the total capacity.

$$OB = \frac{\sum_{i=1}^{n} \int_{t=1}^{T} \lambda_i(t)\, dt}{C}.$$

Multiple-product case: first we compute the booking index for each type of product, OB_s, using the cumulative capacity and the cumulative number of arrivals. Then we calculate OB as the weighted booking index, where the weights correspond to the expected number of arrivals that request each type of product.

$$EA_s = \sum_{i \in A_s} \int_{t=1}^{T} \lambda_i(t)\, dt \quad \text{and} \quad OB_s = \frac{\sum_{j=1}^{s} EA_j}{\sum_{j=1}^{s} C(j)}$$

and finally,

$$OB = \sum_{s=1}^{m} OB_s \frac{EA_s}{\sum_{j=1}^{m} EA_j}.$$

In the first group of computational experiments, we consider a single type of room, and three classes of customers that arrive in a Poisson manner with arrival rates equal to 5, 3, and 2 customers per hour. The prices they pay are \$200, \$120, and \$85, respectively, and there are no rejection costs. We think of these customers as walk-ins who do not have a previous commitment with the hotel. The planning horizon is 12 hours. The performance of the heuristics is shown in Table I.

The first column in the table is the hotel's capacity (number of available rooms). The next three columns have the performance of the heuristics compared with the optimal solution. The last three columns have the performance of the heuristics with respect to the upper bound UB2. The performance of **HEUR1.4** (to give a room to all the requests as long as there is one available) improves as long as the capacity increases. Naturally, if the capacity is large, all the heuristics must lead to the optimal solution. However, for the cases where capacity is an issue, **HEUR1.4** performs much worse than the other heuristics. Therefore, in those cases, it is sensible to use additional information; for example, the current capacity and the probability distribution of future arrivals. We also observe that **HEUR1.1** and **HEUR1.3** perform almost the same; for small and large capacities the solutions given by the heuristics are near to the optimal solutions. When the capacity is small these two heuristics rent rooms to the highest opportunity cost customers which, intuitively, corresponds to the optimal strategy. On the other hand, when the capacity is large enough, the optimal strategy is to accept all the requests, which is also the policy given by the heuristics. The exact behavior can be seen in Figure 2 in Appendix B.

In the second group of experiments we consider two types of rooms with the possibility of downgrading. We also consider three types of customers with arrival rates equal to 2, 3, and 5 customers per hour, prices equal to \$200, \$120, and \$85, and room preferences equal to 1, 2, and 2, respectively. No rejection costs are considered and the planning horizon is 12 hours.

The first column in Table II is the hotel's capacity (the number of rooms). The rooms are ordered according to their quality; thus, the first number in the capacity vector corresponds to suites. The next three columns correspond to the performance of **HEUR1.1**, **HEUR1.2**, and **HEUR1.4** compared with the optimal solution. Finally, the last three columns show the performance of these three heuristics with respect to the upper bound UB2. **HEUR1.2** and **HEUR1.4** have a similar behavior, **HEUR1.4** being slightly better. However, these two heuristics have, in all cases, a worse performance than heuristic 1.1. **HEUR1.2** follows conservative rules, because a customer is accepted only if there is at most 10% probability for renting the room to a higher opportunity cost. Therefore, it is worthwhile to apply **HEUR1.1** because it gives better solutions to the optimization problem. We also observe from the table that the performance of **HEUR1.1** is almost optimal in both extremes; i.e., for small and large capacities. When capacity is small, is it always better not to downgrade rooms because better quality rooms will be rented to the highest opportunity cost customers. On the other hand, all the heuristics perform close to the optimal when the capacity increases to the limit that all the requests should be accepted. The graphic behavior of heuristic 1.1 as a function of the capacity can be found in Figure 3, Appendix B.

For the last set of experiments, we use parameters based on data of a medium size hotel in Santiago, Chile. The hotel works with three main classes of customers: guaranteed reservation customers who are guaranteed a room based on a previous arrangement, "6

Table I
Single-Product Case

Capacity (No. of Rooms)	$OB * 100$	**HEUR1.1** With Optimal Solution (%)	**HEUR1.3** With Optimal Solution (%)	**HEUR1.4** With Optimal Solution (%)	**HEUR1.1** With UB2 (%)	**HEUR1.3** With UB2 (%)	**HEUR1.4** With UB2 (%)
50	240	99.9	99.9	76.7	99.8	99.8	76.6
60	200	98.9	98.4	78.5	98.2	97.7	78.0
70	171	98.5	98.1	82.2	97.4	97.0	81.3
80	150	99.1	99.0	86.0	98.0	97.8	85.0
90	133	99.6	99.6	89.5	98.5	98.5	88.5
100	120	99.6	99.6	93.3	98.5	98.6	92.3
110	109	99.9	100.0	96.9	99.0	99.1	96.0

Table II
Multiple-Product Case With Downgrading

Capacity (No. of Rooms)	$OB * 100$	**HEUR1.1** With Optimal Solution (%)	**HEUR1.2** With Optimal Solution (%)	**HEUR1.4** With Optimal Solution (%)	**HEUR1.1** With UB2 (%)	**HEUR1.2** With UB2 (%)	**HEUR1.4** With UB2 (%)
(10, 30)	528	99.8	84.3	84.3	99.7	84.2	84.2
(10, 45)	463	98.5	87.1	90.1	98.0	86.6	89.6
(10, 50)	448	98.7	90.0	91.9	98.1	89.5	91.4
(10, 60)	425	99.0	94.5	94.6	98.5	94.1	94.1
(15, 70)	305	99.4	96.2	96.4	99.0	95.8	96.0
(20, 80)	240	99.6	97.9	97.8	99.2	97.5	97.4
(20, 90)	231	99.9	99.5	99.3	99.7	99.2	99.1
(25, 95)	195	100.0	100.0	100.0	99.9	100.0	99.9

p.m.-hold" customers who are guaranteed a room based on a previous arrangement only if they arrive before 6 p.m., and "walk-ins" who request a room during the target date without a reservation. Within these categories, customers request standard rooms and suites. Therefore, the hotel faces the demand of six classes of customers. The average percentage of requests that the hotel receives in a day is equal to 81% of guaranteed reservation customers, 14% of 6 p.m.-hold customers, and 5% of walk-ins.

The parameters used in the following simulations can be found in Appendix B. In this case, it is not feasible to compute the value of the optimal objective function, because it takes too long. We only compute the upper bounds UB1 and UB2 and the value of the objective function given by heuristics 1.1, 1.2, and 1.4.

Table III shows the performance of **HEUR1.1**, **HEUR1.2**, and **HEUR1.4** compared with the upper bounds given by UB1 and UB2. We observe that **HEUR1.2** and **HEUR1.4** have a similar behavior, **HEUR1.2** being slightly better. Both heuristics improve their performances when the capacity increases, and in the limit when the capacity is large in comparison with the demand, they perform close to the optimal solution. However, **HEUR1.1** has a better performance than the other heuristics in all the cases considered. We also observe that the worst performance of **HEUR1.1** is 98.7% compared with UB2, which is close to the optimal considering that the optimal solution is less than or equal to the upper bound.

2. THE MULTIPLE-NIGHT CASE

In this section, we extend the methodology of Section 1 to the general case where customers request rooms for several nights. Typically, the tactical problem of accepting reservations is done centrally and the reservations are then sent to the hotel where the operational decisions are made daily. The central office keeps the hotel managers informed about the status of reservations for a number of days ahead (in many cases, managers at the reception desk have access on-line to the hotel reservation system.)

When making operational decisions during a day, hotel managers must take into account the occupancy rate for a few days ahead. For example, if the current day is a high traffic one and the day after if a low traffic one, then it could be optimal to accept a low fare customer that stays in the hotel for two or more nights. On the other hand, it is not necessarily optimal to increase the occupancy rate of the hotel during off-peak days with customers that pay discount fares and stay multiple nights. This is due to the fact that these customers would stay at the hotel for several nights, reducing the net capacity available for days with high demand. Therefore, managers require good estimates of the number of reservations for the next few days (some reservations could be received after making operational decisions for the current day). According to managers in the industry, it is possible to predict with

Table III
Variants of a Real Case with Multiple Products and Downgrading

Capacity (No. of Rooms)	$OB * 100$	**HEUR1.1** With UB1 (%)	**HEUR1.2** With UB1 (%)	**HEUR1.4** With UB1 (%)	**HEUR1.1** With UB2 (%)	**HEUR1.2** With UB2 (%)	**HEUR1.4** With UB2 (%)
(20, 190)	143	99.3	93.3	92.7	99.8	93.8	93.2
(30, 200)	122	98.6	95.1	94.3	99.2	95.6	94.8
(40, 210)	108	98.2	97.0	96.5	98.7	97.4	96.9
(40, 220)	105	98.6	98.1	97.6	99.0	95.5	98.0
(60, 220)	93	99.4	99.3	99.2	99.6	99.5	99.4
(80, 220)	84	99.6	99.7	99.6	99.8	99.9	99.8

reasonable accuracy the number of reservations for one week ahead. The methodology below relies on those estimates to make operational decisions on a real time basis.

Notice that the rejection costs are a function of the number of requested days. It is usually more expensive to walk away from a customer who requests a room for several nights.

At the operational level, every time a customer shows up in the hotel, the manager has to decide whether or not to accept the request, considering the available capacity, price, rejection costs, the number of nights requested, and the demand distribution functions for all classes of customers during the current and the next few days. In what follows, we propose a heuristic to make operational decisions based on heuristic 1.1 for the single-night case.

2.1. Heuristic for the Multiple-Night Case

We consider the following heuristic: Every time a customer shows up at the hotel requesting a room, the available rooms are assigned optimally to the expected number of arrivals for each class of customers during the current day and the next few days plus the current request. If the current request is accepted in this optimal assignment, then the customer is accepted for the corresponding number of nights. Due to the inter-action between the operational decisions during successive days, it is necessary to solve a linear programming problem to optimally assign the expected number of arrivals during the current and next days. It is important to notice that after accepting a customer, the capacity must be updated, taking into account the number of days that the customer will stay at the hotel.

Usually, the number of days that a customer stays is also stochastic, i.e., he can stay more or fewer days than the original number of requested days. Therefore, this fact must be considered when computing the expected number of arrivals for each class of customers during the next few days. The heuristic is easy to extend to incorporate this behavior, as described in Section 3.

We use a rolling window of one week ahead to make operational decisions. In conversations with managers in the industry, they agree that there are two main reasons that justify the appropriateness of this planning horizon: it corresponds to a complete cycle for the demand pattern, and it allows for making good estimates for the total number of reservations.

We also point out that in the multiple-night case, a customer class is also characterized by the number of nights requested. Before describing the heuristic we introduce some additional notation:

$\lambda_{i,j,h}(t)$: the class-i arrival rate for j nights at time t during the hth day; we consider the general case where the arrival rate for a customer class varies with the time of the day and the day of the week.

$x_{i,j,h}$: the number of class-i requests for j days that are accepted during the hth day;

$\pi_{i,j,h}$: the price associated with a class-i customer for j nights, starting at the hth day;

$c_{i,j,h}$ = the rejection cost of a class-i customer who requests a room for j nights, starting at the hth day.

B_p: the set of customer classes that request a room type better than or equal to p;

$C_h(p)$ = the net available capacity for type-p rooms during the hth day, after subtracting the rooms that have already been rented; C represents the vector whose components are $C_h(p)$; for the current day, i.e., the case when $h = 1$, the variable $C_1(p)$ represents the number of type-p rooms that are available at the time t when the decision is being made; for $h > 1$, $C_h(p)$ corresponds to the number of type-p rooms available at the beginning of the hth day.

J: the maximum number of nights requested by customers; usually, in city hotels, the great majority of the requests are for three nights or less.

H: the number of days ahead that are considered to make operational decisions.

Description of Heuristic 2.1 (HEUR2.1)

Suppose that there is a class-i^* request at time t during the current day for j^* nights, then the following algorithm is applied to decide whether or not to accept the customer.

STEP 1. Compute the expected number of arrivals.

$e_{i,j,h}$ = the expected number of class-i arrivals who request a room for j nights during the hth day. For the current day, the expected number of arrivals is computed considering the remaining working hours for that day.

$$e_{i,j,1} = \int_t^T \lambda_{i,j,1}(\tau)\, d\tau \quad \text{for all } i, j$$

$$e_{i,j,h} = \int_0^T \lambda_{i,j,h}(\tau)\, d\tau \quad \text{for all } i, j, h \geq 2.$$

STEP 2. Solve the linear programming problem:

$$\max_{x_{i,j,h},\forall i,j,h} \sum_{h=1}^{H} \sum_{i=1}^{n} \sum_{j=1}^{J} \pi_{i,j,h} x_{i,j,h}$$

$$- \sum_{h=1}^{H} \sum_{i=1}^{n} \sum_{j=1}^{J} c_{i,j,h}(e_{i,j,h} - x_{i,j,h})$$

subject to

$$x_{i,j,h} \leq e_{i,j,h} \quad \text{for all } i, j, h$$

$$\sum_{i \in B_p} \sum_{j=1}^{J} x_{i,j,h} + \sum_{i \in B_p} \sum_{j=2}^{J} x_{i,j,h-1} + \cdots + \sum_{i \in B_p} \sum_{j=J}^{J} x_{i,j,h-J+1} \leq \sum_{i=1}^{p} C_h(i)$$

for all p, h $x_{i,j,h} \geq 0$ for all i, j, h.

STEP 3. Check if the current request is accepted, considering the expected number of arrivals during the next seven days

if $x_{i^*,j^*,1} > 0$, then, accept the customer
otherwise, reject the customer.

2.2. Upper Bound for the Multiple-Night Case

Similarly to the upper bound for the single-night case, the upper bound for the multiple-night problem is based on the fact that better decisions can be made when more information is available. Thus, if the manager knows at the beginning of the seven-day planning horizon the number of requests for each day and for each class of customers, then he can make an optimal allocation of rooms. The following upper bound is equal to the expected profit for the total number of days considered in the planning horizon if the manager optimally allocates the requests above. We define:

$s_h(R_h)$: the vector of random variables that represents the number of requests starting on the hth day if the initial number of reservations is equal to R_h; the component $s_{i,j,h}(R_h)$ is a random variable that represents the number of class-i requests for j days during the hth day;

R: the vector of reservations for the planning horizon; the component R_h is the vector of reservations starting on the hth day.

Hence, the solution of the mathematical model corresponds to an upper bound for the maximum expected profit in the multiple-night case.

Problem UB(C, R)

$$E_{s_h(R_h), \forall h}[G(C, s_1(R_1), \ldots, s_H(R_H))],$$

where

$$G(C, s_1(R_1), \ldots, s_H(R_H)) = \max_{x_{i,j,h}, \forall i,j,h} \sum_{h=1}^{H} \sum_{i=1}^{n} \sum_{j=1}^{J} \pi_{i,j,h} x_{i,j,h} - \sum_{h=1}^{H} \sum_{i=1}^{n} \sum_{j=1}^{J} c_{i,j,h}(s_{i,j,h}(R_h) - x_{i,j,h})$$

subject to

$$x_{i,j,h} \leq s_{i,j,h}(R_h), \quad \text{for all } i, j, h$$

$$\sum_{i \in B_p} \sum_{j=1}^{J} x_{i,j,h} + \sum_{i \in B_p} \sum_{j=2}^{J} x_{i,j,h-1} + \cdots + \sum_{i \in B_p} \sum_{j=J}^{J} x_{i,j,h-J+1} \leq \sum_{i=1}^{p} C_h(i)$$

for all p $x_{i,j,h} \geq 0$ for all i, j, h.

2.3. Computational Experiments for the Multiple-Night Case

In this subsection, we present the performance of heuristic 2.1 obtained through Monte Carlo simulation. The computational experiments use real data from a medium size hotel. The occupancy rate of the hotel is high from Monday to Wednesday and decreases by approximately 40% toward the end of the week. Most of the customers stay at the hotel less than four nights. The parameters used in the experiments can be found in Appendix B. For the purpose of comparison, we also include the performance of heuristic 2.2. This heuristic accepts a request as long as there is an available room that satisfies the customer's need.

The booking index, OB, is computed according to the definition in subsection 1.4, where the arrival rate is replaced by the average arrival rate during the week weighted by the number of days customers stay at the hotel.

The first column in Table IV shows the total capacity of the hotel and the second column contains the booking index. Columns 3 and 4 show the performance of heuristics 2.1 and 2.2, respectively. We observe that in the range of capacities considered, the performance of heuristic 2.1 is satisfactory. The difference between the total expected profit obtained with the heuristic and the upper bound is less than or equal to 2% in all cases. Note that in practice hotels usually overbook around 10% of their capacity, so the experiments in the table include the range of overbooking observed in practice.

Also observe that the policy where all customers are accepted as long as there is an available room, gives results that are significantly worse than those obtained with heuristic 2.1. The poor performance of heuristic 2.2 is due to the fact that it only considers the immediate profits when making renting decisions, without including

Table IV
Variants of a Real Case With Multiple-Night Stays

Capacity (No. of Rooms)	$OB * 100$	HEUR2.1 With UB (%)	HEUR2.2 With UB (%)
(20, 190)	152	98.0	77.4
(30, 200)	129	98.0	84.1
(40, 210)	114	98.0	86.5
(40, 220)	110	98.2	88.3
(60, 220)	97	98.3	88.3
(80, 220)	88	99.1	90.1

the potential benefits and costs of reserving the room for future customers for which the room has a higher value.

3. CONCLUSIONS AND EXTENSIONS

This paper studies optimal policies for renting hotel rooms to various classes of customers. We consider stochastic and dynamic arrivals of customers from different market segments (airline employees, government workers, executives, tourists, etc.); no assumptions concerning the particular order between the arrivals of different classes of customers; multiple types of rooms with the possibility of downgrading; and multiple night stays. The limited capacity and the finite planning horizon play a central role when determining the optimal selling strategies.

We characterize the optimal policies for the single-night case as follows: Given a period of time, if a request is accepted for a certain capacity, then it is also accepted for any larger capacity. Furthermore, for every class of customers and every capacity vector, there exists an instant in time beyond which it is optimal to satisfy the customer's request.

Exact optimal policies can be computed for the single-product, single-night case. For the general case with multiple types of rooms and/or multiple nights, we develop ad hoc heuristics, based on the qualitative properties of the optimal policies, to find approximations for the optimal solution. Computational results show that their performance is close to the optimal solution for a variety of realistic scenarios. The main feature of these heuristics is their simplicity, which is extremely important in terms of their practicality; managers are more open to use rules that are intuitive and easy to implement.

As we mentioned in Section 2, it is not unusual that customers stay at the hotel fewer or more nights than the number of nights originally requested. If this is the case, we propose to modify step 1 of heuristic 2.1 as follows: Consider an extended definition of customer classes. For this purpose, define $e_{i,j,k,h}$ as the expected number of class-i arrivals that request a room for j nights during the hth day and stay at the hotel for k nights, and $q_{i,j,k,h}$ as the fraction of customers from class i that request a room for j nights starting the hth day and staying k nights. Hence,

$$e_{i,j,k,1} = q_{i,j,k,h} \int_t^T \lambda_{i,j,1}(\tau)\, d\tau \quad \text{for all } i, j$$

$$e_{i,j,k,h} = q_{i,j,k,h} \int_0^T \lambda_{i,j,h}(\tau)\, d\tau \quad \text{for all } i, j, h \geq 2.$$

Then, apply Steps 2 and 3 as in the original heuristic.

An important result of this paper is the simple characterization of the optimal selling policies in a realistic framework where customers from different market segments arrive simultaneously. We also added a new dimension to this problem when considering multiple types of rooms with the possibility of downgrading, a practice widely used in the hotel industry.

APPENDIX A

Lemma 1. *The function $F_t(c, i)$ is monotonically increasing as a function of the capacity:*

$$F_t(c, i) \geq F_t(c - e_p, i), \quad \text{for all } c, p, t, i,$$

and

$$F_t(c - e_k, i) \geq F_t(c - e_p, i),$$

$$\text{for all } k \geq p, \text{ and for all } c, t, i.$$

The single-product case corresponds to the particular case where c is a scalar and $e_p = e_k = 1$.

Proof. Suppose that we have an optimal policy when we start with a capacity equal to $c - e_p$ at time t. This policy is also feasible when we start with a capacity equal to c at time t (and gives the same value for the objective function), but it is not necessarily optimal. Therefore, the maximum expected profit from time t onwards starting with capacity $c - e_p$ is less than or equal to the maximum expected profit starting with capacity c. Hence,

$$F_t(c, i) \geq F_t(c - e_p, i), \quad \text{for all } c, p, t, i.$$

The same argument can be used when we replace c by $c - e_k$ with $k \geq p$, because product p can be downgraded to be used as product k.

Lemma 2. *The objective function satisfies the inequality:*

$$F_t(c + e_l + e_p, i) - F_t(c + e_l, i)$$
$$\leq F_t(c + e_p, i) - F_t(c, i), \quad \text{for all } t, i, c, p, l. \quad (6)$$

Proof. In what follows we use induction in t to prove the lemma for the single-product case. The objective function for the last period in the planning horizon is given by:

$$F_T(c, i) = \begin{cases} \pi_i & \text{if } c > 0 \text{ and } i \neq 0, \\ -c_i & \text{if } c = 0 \text{ and } i \neq 0, \\ 0 & \text{if } i = 0. \end{cases}$$

We observe that given i, $F_T(c, i)$ trivially satisfies the inequality (6), i.e.,

$$F_T(c + 2, i) - F_T(c + 1, i)$$
$$\leq F_T(c + 1, i) - F_T(c, i) \quad \text{for all } c.$$

In the induction hypothesis we assume that the inequality (6) is satisfied at period $t + 1$ and we prove that it is also satisfied at period t. The objective functions at period t are given by (assuming that $i \neq 0$ and $c > 0$):

$$F_t(c, i) = \begin{cases} \pi_i + E_j[F_{t+1}(c-1, j)] & \text{(a)}, \\ -c_i + E_j[F_{t+1}(c, j)] & \text{(b)}. \end{cases}$$

$$F_t(c+1, i) = \begin{cases} \pi_i + E_j[F_{t+1}(c, j)], & \text{(c)}, \\ -c_i + E_j[F_{t+1}(c+1, j)] & \text{(d)}. \end{cases}$$

$$F_t(c+2, i) = \begin{cases} \pi_i + E_j[F_{t+1}(c+1, j)] & \text{(e)}, \\ -c_i + E_j[F_{t+1}(c+2, j)] & \text{(f)}. \end{cases}$$

To prove property (6) we have to consider all possible values for the functions above. Thus, we have to evaluate the eight combinations.

Combination (a) + (c) + (e): Replacing the objective functions for their corresponding values in inequality (6), we get,

$$\pi_i + E_j[F_{t+1}(c+1, j)] - \pi_i - E_j[F_{t+1}(c, j)]$$
$$\leq \pi_i + E_j[F_{t+1}(c, j)] - \pi_i - E_j[F_{t+1}(c-1, j)],$$

which is true by the induction hypothesis. The same proof holds for combinations (b) + (c) + (e), (b) + (d) + (e), and (b) + (d) + (f).

Combination (a) + (d) + (f): In this case, when the capacity is c we accept the request, i.e.,

$$c_i + \pi_i \geq E_j[F_{t+1}(c, j)] + E_j[F_{t+1}(c-1, j)],$$

and when the capacity is $c + 1$, we reject the request, i.e.,

$$c_i + \pi_i \leq E_j[F_{t+1}(c+1, j)] + E_j[F_{t+1}(c, j)].$$

Both inequalities together lead to:

$$E_j[F_{t+1}(c+1, j)] + E_j[F_{t+1}(c, j)]$$
$$\geq E_j[F_{t+1}(c, j)] + E_j[F_{t+1}(c-1, j)],$$

which contradicts the induction hypothesis. Hence, the combination (a) + (d) + (f) is infeasible. Combinations (b) + (c) + (e), (b) + (d) + (e), and (b) + (d) + (f) are also infeasibles. Therefore, the inequality (6) holds for period t. When $c = 0$ the same proof holds. For the case when $i = 0$ the inequality (6) holds trivially.

A similar proof holds for the case with types of rooms. We believe that the same proof can be generalized for the multiple-product case with more than two types of rooms. This lemma says that the smaller the current inventory, the larger the profit associated with an extra room. The intuition behind this lemma is that the opportunity cost of an extra room decreases when the capacity increases because the stochastic process that determines the total demand does not change.

Lemma 3. *The function $F_t(c, R, i)$ is monotonically increasing as the function of the capacity and it satisfies the property given in Lemma 2.*

Proof. The proof is an extension of the multiple-product case without reservations.

Proof of Proposition 3

In what follows we present the general proof for the multiple-product case. By Lemma 2, we know that the inequality holds:

$$F_{t+1}(c - e_l, j) - F_{t+1}(c - e_p - e_l, j)$$
$$\geq F_{t+1}(c, j) - F_{t+1}(c - e_p, j) \quad \text{for all } t, i, c, p, l.$$

The probability of a class-i arrival in the next unit of time depends only on t, hence we take the expected value in both sides and the inequality still holds:

$$E_j[F_{t+1}(c - e_l, j) - F_{t+1}(c - e_l - ep, j)]$$
$$\geq E_j[F_{t+1}(c, j) - F_{t+1}(c - e_p, j)].$$

Thus,

$$\alpha_t(c - e_l, p) \geq \alpha_t(c, p).$$

Proof of Proposition 4

Assume that a type-l room is sold to a class-i customer when the capacity is equal to c. Assuming that $l \neq p$, the same room l is available for a class-i customer when the initial capacity is $c - e_p$, then:

$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\max\{\pi_i + E_j[F_{t+2}(c - e_l, j)], -c_i + E_j[F_{t+2}(c, j)]\}$$
$$-\max\{\pi_i + E_j[F_{t+2}(c - ep - e_l, j)],$$
$$-c_i + E_j[F_{t+2}(c - e_p, j)]\}.$$

We analyze the following two cases.

Case 1

$$\text{Max}\{\pi_i + E_j[F_{t+2}(c - e_p - e_l, j)];$$
$$-c_i + E_j[F_{t+2}(c - e_p, j)]\}$$
$$= -c_i + E_j[F_{t+2}(c - e_p, j)].$$

In this case,

$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\geq -c_i + E_j[F_{t+2}(c, j)] - (-c_i + E_j[F_{t+2}(c - e_p, j)]),$$
$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\geq E_j[F_{t+2}(c, j)] - E_j[F_{t+2}(c - e_p, j)]$$
$$\alpha_{t+1}(c, p)F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\geq \alpha_{t+1}(c, p). \qquad (7)$$

Case 2

$$\text{Max}\{\pi_i + E_j[F_{t+2}(c - e_p - e_l, j)];$$
$$-c_i + E_j[F_{t+2}(c - e_p, j)]\}$$
$$= \pi_i + E_j[F_{t+2}(c - e_p - e_l, j)].$$

In this case,

$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\geq \pi_t + E_j[F_{t+2}(c - e_l, j)]$$
$$- (\pi_t + E_j[F_{t+2}(c - e_p - e_l, j)]),$$
$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\geq E_j[F_{t+2}(c - e_l, j)] - E_j[F_{t+2}(c - e_p - e_l, j)]$$
$$= \alpha_{t+1}(c - e_l, p)F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\geq \alpha_{t+1}(c - e_l, p). \quad (8)$$

Using Proposition 3, we obtain:

$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i) \geq \alpha_{t+1}(c - e_l, p)$$
$$\geq \alpha_{t+1}(c, p).$$

Equations (7) and (8) yield:

$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i) \geq \alpha_{t+1}(c, p).$$

For $i = 0$, the previous result holds trivially. Therefore, using the fact that the probability of a new arrival depends only on t, we take the expected value on both sides of the inequality above, obtaining the desired result:

$$E_t[F_{t+1}(c, i) - F_{t+1}(c - e_p, i)] \geq \alpha_{t+1}(c, p),$$
$$\alpha_t(c, p) \geq \alpha_{t+1}(c, p).$$

For the special case when $p = l$, we have two cases:

1. If the initial capacity $c - e_p$ has an available type-p room, then the proof above holds without any change.
2. If the initial capacity $c - e_p$ does not have any type-p room available, then if the current request is accepted the customer receives a type-k room. This room is better than a type-p room, i.e., $k < p$. In this case, the proof changes slightly, so we present the part of the proof that differs from the previous case. By definition we have:

$$F_{t+1}(c, i) - F_{t+1}(c - ep, i)$$
$$= \max\{\pi_t + E_j[F_{t+2}(c - e_p, j)];$$
$$- c_t + E_j[F_{t+2}(c, j)]\}$$
$$-\max\{\pi_t + E_j[F_{t+2}(c - ep - e_k, j)];$$
$$-c_i + E_j[F_{t+2}(c - e_p, j)]\}.$$

Similarly to the previous proof, we have to analyze two cases. The first one is analogous to case 1 above. The second case changes to the following:

$$\max\{\pi_t + E_j[F_{t+2}(c - e_p - e_k, j)];$$
$$-c_i + E_j[F_{t+2}(c - e_p, j)]\}$$
$$= \pi_t + E_j[F_{t+2}(c - e_p - e_k, j)].$$

Hence,

$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\geq \pi_t + E_j[F_{t+2}(c - e_p, j)]$$
$$- (\pi_t + E_j[F_{t+2}(c - e_p - e_k, j)]),$$
$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\geq E_j[F_{t+2}(c - e_p, j)] - E_j[F_{t+2}(c - e_p - e_k, j)]$$
$$= \alpha_{t+1}(c - e_p, k).$$

Using Proposition 3, we obtain:

$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i)$$
$$\geq \alpha_{t+1}(c - e_p, k) \geq \alpha_{t+1}(c, k).$$

Finally, using that the function $F_{t+1}(c)$ is increasing as a function of the capacity, we have:

$$\alpha_{t+1}(c, k) \geq \alpha_{t+1}(c, p) \quad \text{if } k < p.$$

Therefore,

$$F_{t+1}(c, i) - F_{t+1}(c - e_p, i) \geq \alpha_{t+1}(c, p).$$

The rest of the proof follows the same as in the previous case.

Proof of Proposition 5

Considering that the probability of a new arrival during the next time interval of length Δt depends only on t and on the number of pending reservations, we apply the expected value on both sides of the inequality given in Lemma 3 to obtain the desired result.

APPENDIX B

The performance of **HEUR1.1** for the single-product and the multiple-product cases is shown in Figures 2 and 3, respectively.

Parameters Used in the Simulations

The planning horizon is a operational day that starts at 8 a.m. and finishes at 12 p.m. The parameters correspond to:

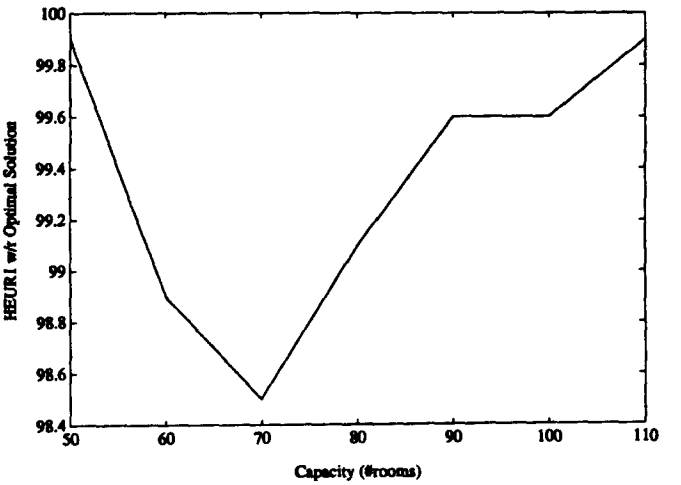

Figure 2. Performance of **HEUR1.1** for the single-product case.

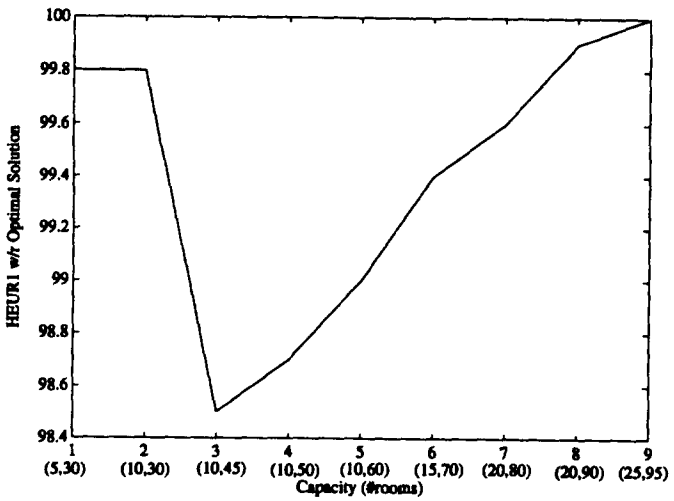

Figure 3. Performance of **HEUR1.1** for the multiple-product case.

Number of customer classes: 6
Number of room types: 2
Prices: (190, 200, 250, 70, 80, 110)
Rejection costs: (190, 100, 0, 70, 40, 0)
Room preferences: (1, 1, 1, 2, 2, 2)
Periods in the planning horizon: 8–10, 10–12, 12–16, 16–18, 18–20, 20–21, 21–24.

Arrival rates for the single-night case

Arrival rate (guarantee): (3.1, 53.9, 3.1, 3.1, 3.1, 53.9, 3.1) customers/hour.
Arrival rate (6 p.m.): (0.46, 16.7, 0.46, 0.46, 0, 0, 0) customers/hour.
Arrival rate (walk-ins): (0.32, 0.32, 0.32, 3.6, 3.6, 3.6, 3.6) customers/hour.

Arrival rates for the multiple night case

Days 1, 2, and 3

Arrival rate (guarantee): (2.5, 43.1, 2.5, 2.5, 2.5, 43.1, 2.5) customers/hour.
Arrival rate (6 p.m.): (0.37, 13.4, 0.37, 0.37, 0, 0, 0) customers/hour.
Arrival rate (walk-ins): (0.26, 0.26, 0.26, 2.9, 2.9, 2.9, 2.9) customers/hour.

Days 4 and 5

Arrival rate (guarantee): (2.1, 37.7, 2.1, 2.1, 2.1, 37.7, 2.1) customers/hour.
Arrival rate (6 p.m.): (0.32, 11.7, 0.32, 0.32, 0, 0, 0) customers/hour.
Arrival rate (walk-ins): (0.22, 0.22, 0.22, 2.5, 2.5, 2.5, 2.5) customers/hour.

Days 6 and 7

Arrival rate (guarantee): (1.8, 32.3, 1.8, 1.8, 1.8, 32.3, 1.8) customers/hour.
Arrival rate (6 p.m.): (0.27, 10.0, 0.27, 0.27, 0, 0, 0) customers/hour.
Arrival rate (walk-ins): (0.19, 0.19, 0.19, 2.2, 2.2, 2.2, 2.2)

These simulations consider customers request rooms for one, two, or three nights with probabilities 0.3, 0.5, and 0.2, respectively.

ACKNOWLEDGMENT

Susan Mondschein thanks Fondecyt (Chile) grant 1940479 for financial support.

REFERENCES

ALSTRUP, J., S. BOAS, O. MADSEN AND R. VIDAL. 1986. Booking Policy for Flights With Two Types of Passengers. *Eur. J. Opnl. Res.* **27,** 274–288.

AMERICAN AIRLINES ANNUAL REPORT. 1987. The Art of Managing Yield, 22–25.

BELOBABA, P. P. 1989. Application of a Probabilistic Decision Model to Airline Seat Inventory Control. *Opns. Res.* **37,** 183–197.

BITRAN, G., S. GILBERT AND T-Y. LEONG. 1992. Hotel Sales and Reservations Planning. *The Service Productivity and Quality Challenge*.

BITRAN, G., AND S. GILBERT. 1992. Managing Hotel Reservations With Uncertain Arrivals. *Working Paper, Weatherhead School of Management, Department of Operations Research, Case Western Reserve University*, Cleveland, Ohio.

BRUMELLE, S. L., J. I. MCGILL, T. H. OUM, K. SAWAKI AND M. W. TRETHEWAY. 1990. Allocation of Airline Seats Between Stochastically Dependent Demands. *Trans. Sci.* **24,** 183–192.

GILBERT, S. M. 1991. Management of Co-production Processes With Random Yields: Applications in Manufacturing and Services. Ph.D. Thesis, Sloan School of Management, MIT, Cambridge, Mass.

LADANY, S. 1976. Dynamic Operating Rules for Motel Reservations. *Dec. Sci.* **7,** 829–840.

LITTLEWOOD, K. 1972. Forcasting and Control of Passenger Bookings. *AGIFORS Symp. Proc.* **12,** 95–117.

MERLISS, P. P., AND C. LOVELOCK. 1980. The Parker House: Sales and Reservations Planning. Harvard Business School Case 9-580-152.

ROTHSTEIN, M. 1974. Hotel Overbooking as a Markovian Sequential Decision Process. *Dec. Sci.* **5,** 389–404.

WEATHERFORD, L., AND S. BODILY. 1992. A Taxonomy and Research Overview of Perishable-Asset Revenue Management: Yield Management, Overbooking, and Pricing. *Opns. Res.* **40,** 831–844.