An EMSR-based approach for revenue management with integrated upgrade decisions

January 15, 2013

Jochen Gönsch, Sebastian Koch, Claudius Steinhardt

Department of Analytics & Optimization,
University of Augsburg, Universitätsstraße 16, 86159 Augsburg, Germany

Jochen Gönsch (✉)

Phone: +49 821 598 4148

Fax: +49 821 598 4226

Email: jochen.goensch@wiwi.uni-augsburg.de

Sebastian Koch

Email: sebastian.koch@wiwi.uni-augsburg.de

Claudius Steinhardt

Email: claudius.steinhardt@wiwi.uni-augsburg.de

Abstract

We consider the revenue management problem of capacity control with integrated upgrade decision-making. The dynamic programming formulation of this problem is hard to solve to optimality, even in the single-leg case, because multiple hierarchical resource types must be considered simultaneously. Therefore, in this paper, we propose a new heuristic approach that generalizes the idea behind the well-known single-leg EMSR-a procedure to multiple resource types. Similar to EMSR-a, our approach is based on the computation of protection levels, but additionally allows for the integrated consideration of upgrades. In addition, we derive control policies for typical demand arrival patterns. As an extension, we propose a generalization of our approach that allows for arbitrarily ordered prices with respect to the upgrade hierarchy. Furthermore, we perform a number of computational experiments to investigate the performance of the new approach compared to other capacity control methods that incorporate upgrades. We consider typical airlines' single-leg scenarios with 10 (re)optimizations throughout the booking horizon. The results show that our approach can significantly outperform existing methods in terms of the total achieved revenue, including dynamic programming decomposition approaches that are proposed in literature, as well as successive planning approaches that are widely used in commercial revenue management systems.

Keywords: Revenue Management, Upgrades, Capacity Control, EMSR

1 Introduction

Originating from the deregulation of the U.S. airline industry in the 1970s, revenue management has become one of the most successful fields where operations research methods are applied. The main task of revenue management is capacity control, which can be described as maximizing net revenue by optimally controlling the availability of products defined on a fixed, scarce capacity of perishable resources within a given selling horizon. In the scientific literature, a large number of different operations research models have been proposed that allow for the automation of capacity control (see e.g. [1], Chapters 2 and 3 for an overview of the standard models for capacity control). Many of these models have been successfully implemented in commercial software systems.

A relatively new aspect of scientific research is incorporating upgrades into capacity control. Following the definition of Gallego and Stefanescu [2], a customer receives an upgrade when the selling firm fulfills a product request with a more desirable substitute from a pre-specified set of alternative resource types. This substitution is offered at the original product’s price. In practice, upgrades have already been used for a long time. In fact, the product portfolios of service industry firms are often based on several types of resources which are substitutable, so that upgrading is potentially beneficial. Prominent examples include airlines that offer economy, business, and first class compartments, hotels that have different room types, and various car categories at car rental companies. Because traditional capacity control models do not consider different resource types simultaneously, practice implementations often take upgrades into account by a successive planning approach. More precisely, upgrade contingents are estimated first, and based on the resulting new virtual capacities, traditional capacity control approaches are applied afterward for each resource type separately.

In the scientific literature, a few authors investigate the revenue potential when upgrades are explicitly accounted for by capacity control mechanisms. Alstrup et al. [3] as well as Karaesmen and van Ryzin [4] address overbooking problems with upgrades and downgrades. Geraghty and Johnson [5] mention that they use a modified standard heuristic with upgrades in the car rental industry, without giving any insight into the mathematical models. Shumsky and Zhang [6] develop an integrated approach of capacity control and upgrading that is based on protection limits. However, their assumptions regarding demand as well as cost structure are not in line with the traditional revenue management setting. To the best of our knowledge, Gallego and Stefanescu [2] are the first to integrate upgrades into the traditional dynamic programming model for network capacity control with independent demand (see e.g. [1], Chapter 3.2 for the standard dynamic program).

However, the resulting dynamic programming model is computationally intractable for larger problem sizes because of the multidimensional state space that must be considered. As opposed to traditional revenue management without upgrades, this is even true for the single-leg setting without any network structure, because the different resource types must still be considered simultaneously. Therefore, it is necessary to rely on some type of approximation of the dynamic programming model. In this context, Gallego and Stefanescu [2] present a deterministic linear approximation that incorporates upgrades, which is basically an extension of the well-known Deterministic Linear Programming model (DLP; see e.g. [1], Chapter 3.3.1). Steinhardt and Gönsch [7] also build on the dynamic programming model of Gallego and Stefanescu [2] and derive dynamic programming decomposition approaches for network capacity control with upgrades. One of their approximation approaches, called Dynamic Programming Single Resource Decomposition (DPD-S), is also suitable for the single-leg case.

In this paper, we propose a new approximation of the dynamic programming formulation for capacity control with integrated upgrades. Our approach focuses on the single-leg case and builds on the well-known EMSR-a heuristic (see [8] and [9]) whose name is derived from the Expected Marginal Seat Revenue (EMSR). Similar to EMSR-a, the approach is based on the computation of protection levels but also allows for the integrated consideration of upgrades. In addition, we derive control policies for low-before-high and arbitrarily ordered demand, and we consider arbitrarily ordered prices with respect to the upgrade hierarchy. Furthermore, we perform an extensive computational study that considers typical airlines' single-leg scenarios with 10 (re)optimizations throughout the booking horizon. In terms of the total achieved revenue, the new approach can outperform methods that are based on the extended DLP-model, DPD-S, and other models that are modified to integrate upgrade decisions. Furthermore, the new approach performs better than the successive planning approach that is mostly used in revenue management practice.

The remainder of this paper is structured as follows: Section 2 provides the basics of EMSR-a. Based on that, we develop the new heuristic for integrated upgrade and capacity control in Section 3. In Section 4, we present the insights obtained from our computational study. Section 5 discusses the approach and the numerical results from a broader perspective. We conclude with a summary of the main results and an outlook on future research in Section 6.

2 EMSR-a heuristic in traditional revenue management

The traditional revenue management model setting and the resulting decision problem can be described as follows: As a result of price discrimination, a firm offers differently priced products, each of which needs one unit of a single homogeneous resource with fixed capacity. Product requests arrive in a stochastic manner successively over time before service provision. There are no group requests; each request is for one unit of one of the offered products, and there are no cancellations of accepted requests or no-shows. Overbooking of the given resource's capacity is not allowed. For each incoming request, the firm can decide to accept or reject it. The firm aims at maximizing total overall revenue, as variable costs are assumed to be negligible.

EMSR is a class of heuristics developed by Belobaba [8, 9, 10] that have evolved to the most prominent methods for solving the decision problem described above. The heuristics are used to calculate protection levels under the additional model assumption that product requests arrive in the order of non-decreasing prices (low-before-high). The idea behind protection levels is to preserve a precalculated number of resource capacity units for later booking requests that ask for higher-value products. More recently, several authors extended the traditional EMSR heuristics, including additional aspects (see e.g. [11] for customer diversion; [12] for risk aversion; [13] for choice-based methods). Furthermore, there are extensions that generalize the idea of EMSR to procedures that are applicable in network settings (see e.g. [1], Chapter 3.4.2 for a generic procedure called prorated EMSR as well as [14], Chapter 5.2 for an extension to a multiple resources setting with a specific multi-stage production structure arising in steel production).

We now summarize the basics of EMSR-a (see e.g. [8], [9], or [1], Chapter 2.2.4.1 for more details). This heuristic is a generalization of Littlewood's rule ([15], see also e.g. [1], Chapter 2.2.1) which, as additional assumptions to the setting described above, assumes that (1) the firm offers just two products and (2) demand arrives in low-before-high order. The two products are denoted by $j = 1,2$ with associated prices p_j and $p_1 > p_2$. The total stochastic demand for product j is given by D_j, with the distribution $F_j(\cdot)$. The optimal decision policy, which is Littlewood's rule, can then intuitively be derived as follows: Suppose the firm has x units of capacity remaining and that it receives a request for product 2. If the request is accepted, the firm collects revenue of p_2. If it is not accepted, the firm will sell the x-th unit at p_1 later, but only if the demand for product 1 is sufficiently high, which means if and only if $D_1 \geq x$. Thus, the expected gain from reserving the x-th unit for product 1 is $p_1 \cdot P(D_1 \geq x)$. Therefore, the optimal decision rule throughout the selling horizon is to accept requests for product 2 as long as this latter term is less than or equal to p_2. Because of its origin in the airline industry where the capacity units are seats, this term is also widely referred to as the EMSR. If demand is modeled using a continuous distribution $F_1(x)$, the optimal protection level s_{12}^* is given by $p_2 = p_1 \cdot P(D_1 \geq s_{12}^*)$, which is equivalent to $s_{12}^* = F_1^{-1}\left(1 - \frac{p_2}{p_1}\right)$.

EMSR-a generalizes Littlewood's rule to $n > 2$ products, while it keeps the assumption that requests arrive in low-before-high order. The products are given by $\mathcal{J} = \{1,\ldots,n\}$. The price of each product $j \in \mathcal{J}$ is given by p_j, and its stochastic demand is denoted by D_j. As before, the products are indexed by non-increasing prices.

The basic idea of EMSR-a is to appropriately aggregate the pair-wise protection levels that are obtained from applying Littlewood's rule to all the pairs of products, in isolation. In the first step, the pair-wise protection levels s_{kj}^* are calculated for all the pairs of products $k, j \in \mathcal{J}$ with $k < j$, denoting how much capacity should be protected from product j and set aside for product k:

$$s_{kj}^* = F_k^{-1}\left(1 - \frac{p_j}{p_k}\right) \tag{1}$$

In a second step, these pair-wise protection levels are added up to obtain s_j, which is the amount of capacity to protect from product j for all the higher-value products, and requests for product j are rejected if $x - 1 < s_j$. Obviously, the protection level s_j cannot exceed the remaining capacity:

$$s_j = min\left(\sum_{k<j} s_{kj}^*, x\right). \tag{2}$$

3 EMSR-a heuristic with integrated upgrade decisions

In this section, we propose a new integrated approach for capacity control with planned upgrades that builds upon a generalization of the idea behind EMSR-a. In Section 3.1, we introduce some notation. In Section 3.2, we present the basic version of the approach in which we restrict ourselves to quality-consistent prices in order to introduce the core concepts more clearly. Prices are quality-consistent if and only if higher quality products are never cheaper than lower quality products. This means that prices are always non-decreasing when moving up the upgrade hierarchy (see [2] for a formal definition). In Section 3.3, we formulate two control mechanisms that can be used to decide on the acceptance of arriving booking requests using the heuristic proposed. The first mechanism is static and can be used for low-before-high arrival order. The second mechanism is dynamic and allows for arbitrary arrival orders. In Section 3.4, we relax the restriction on quality-consistent prices and present a general algorithm for computing protection levels with arbitrarily ordered prices.

3.1 Setting and notation

In line with the traditional revenue management setting and the notation introduced in Section 2, we consider a firm that offers products $\mathcal{J}=\{1,...,n\}$ at a point in time t, where p_j denotes the price of a product $j \in \mathcal{J}$. The products are indexed in the order of non-increasing prices, that is, $p_1 \geq p_2 \geq \ldots \geq p_n$. D_{jt} is again the demand random variable – with an additional index t – which refers to the demand-to-come of a product $j \in \mathcal{J}$ from time t onwards.

In addition to the traditional setting, there is no longer just one homogeneous resource, but the firm now disposes of several different resources $\mathcal{R}=\{1,\ldots,m\}$, where a higher number indicates a higher-value resource with respect to the upgrade hierarchy. To provide product $j \in \mathcal{J}$, the firm can either use one unit of the corresponding resource $r_j \in \mathcal{R}$ or upgrade the request. That is, instead of providing r_j, the firm can assign the request to any resource $r \in \mathcal{R}$ with $r > r_j$. In this context, we assume that customers always accept upgrading. The vector of remaining capacity is $\mathbf{x}=(x_1,...,x_m)$. Apart from that, the other standard assumptions introduced in Section 2 apply. In what follows, the symbols j, k, and l refer to products, and r refers to resources.

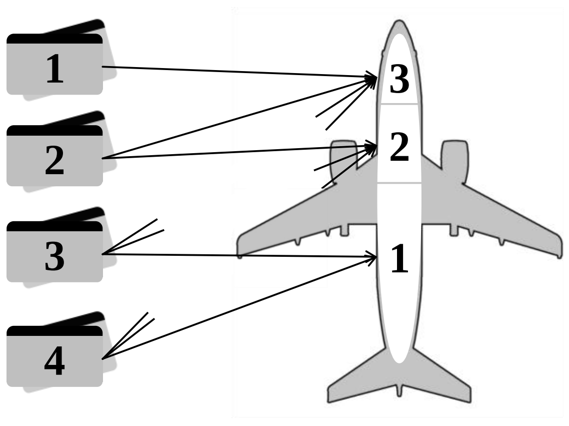

Figure 1 Illustration of the relationship between products and resources

Example. We illustrate the setting described above using the airline industry, where one resource corresponds to one compartment (Figure 1). In the example, four different booking classes are offered (products $\mathcal{J}=\{1,2,3,4\}$), which make use of the three compartments economy, business, and first class (resources $\mathcal{R}=\{1,2,3\}$). Product 1 is a first class ticket ($r_1 = 3$) and, hence, can consume capacity only in first class (resource 3). Product 2 is a business class ticket ($r_2 = 2$) and can consume capacity in either business (resource 2) or first class (resource 3). Products 3 and 4 are economy class tickets ($r_3 = r_4 = 1$) and can consume capacity in either of the three compartments.

Note that throughout this paper, we assume that the considered firm decides on upgrading an accepted request immediately after the time of sale, that is, it performs ad hoc upgrading. More precisely, accepted requests are immediately assigned to the lowest available resource type in the upgrade hierarchy that is equal or higher to the one that the sold product requires. It may seem natural to suppose that the firm could improve its revenue by postponing the decision on the assignment of requests to a later point in time before the end of the booking horizon. However, it is not possible to obtain higher revenues this way; as shown for the deterministic setting in [2] and the general stochastic setting in [7], which applies to our work here, both ad hoc and postponed upgrading lead to the same acceptance decisions and revenues. Moreover, it is optimal to immediately assign a request to the lowest possible resource in the upgrade hierarchy; if a request for product j is accepted, it can immediately be assigned to resource $r^* = min\left(r : r \geq r_j \wedge x_r > 0\right)$. Thus, the firm can easily track what capacity is necessary to serve already accepted requests and what is still available for future requests. Note that ad hoc upgrading does not imply that one immediately has to decide or communicate which customer will get which upgrade. It is only fixed how many customers will be upgraded. For an airline setting, this means, for example, that at the end of the booking horizon, it is only fixed how many passengers from economy class need to be upgraded to for example business class, while the individual assignment can be performed just before boarding.

3.2 Heuristic for quality-consistent prices

In this subsection, we show how the basic idea underlying the EMSR-a heuristic can be transferred to the revenue management problem with upgrades when prices are quality-consistent.

As in classical EMSR-a, the pair-wise protection levels for all the products calculated by Littlewood's rule (1) are the starting point of our considerations. The goal is to aggregate them to obtain protection levels s_j, which indicate for every product j how much capacity should be reserved for requests for higher-value products that potentially arrive later in the booking horizon. Because any resource $r \geq r_j$ can be used to provide product j, these protection levels now must be related to the sum of the capacity of these resources. That means that requests for j are rejected if $\sum_{r \geq r_j} x_r - 1 < s_j$.

However, when aggregating the protection levels s_{kj}^* to obtain the amount of capacity s_j to protect for higher-value products $k < j$, it is now necessary to consider that usually not all products $k < j$ can be provided using any arbitrary resource $r \geq r_j$, because some of these products might need higher resources than r_j. Thus, if we just summed up the pair-wise protection levels $k < j$ and constrained this sum to the remaining capacity of the resources $r \geq r_j$ as in EMSR-a (see Equation (2)), we might end up protecting capacity for one or more higher-value products on resources that, in fact, cannot be used to provide these products. Therefore, the idea behind our approach is to consider the capacity of each of these resources before summing up. This leads to reduced (adjusted) pair-wise protection levels s_{kj}. In particular, any pair-wise protection level s_{kj} must never exceed the sum of the available capacity of the resources $r \geq r_k$ – which is the maximum of the capacity units that could potentially be used for product j requests at all – minus the capacity units that should be protected for later arriving higher-value requests for products $l < k$. The complete algorithm for computing a certain protection level s_j that reserves capacity for all higher-value products can be stated as follows:

Algorithm 1: Computing a protection level with quality-consistent prices

```
Preconditions: Product $j$ to consider, remaining capacities $\mathbf{x}$
1. Compute pair-wise protection levels $s_{kj}^* \ \forall k < j$ by Equation (1).
2. Set $k := 1$.
3. While $k < j$:
    3.1 Compute adjusted pair-wise protection levels $s_{kj} = min\left( s_{kj}^*, \sum_{r \geq r_k} x_r - \sum_{l<k} s_{lj} \right)$.
    3.2 Increment $k := k + 1$ and go to Step 3.
4. Compute protection level $s_j = \sum_{k<j} s_{kj}$.
```

Note that, from this reservation procedure, it follows that $s_{kj} \leq s_{kj}^* \ \forall k$.

As in classical EMSR-a, the algorithm calculates the pair-wise protection levels s_{kj}^* for all the products $k < j$ using Equation (1) in Step 1. Then, beginning with product $k = 1$ (Step 2), all the products $k < j$ are considered in Step 3 and the pair-wise protection levels s_{kj}^* are adjusted. The adjusted pair-wise protection level s_{kj} represents the capacity to protect for product k from product j on resources $r \geq r_j$. This protection level is calculated as the minimum of the pair-wise protection level s_{kj}^* given by Littlewood's rule and the capacity that can be used for product j minus the sum of the adjusted pair-wise protection levels s_{lj} for products $l < k$ that are higher-valued than k. Finally, in Step 4, the adjusted pair-wise protection levels s_{kj} are summed up to obtain s_j.

Example. Figure 2 illustrates how to apply the above Algorithm 1, using the example introduced in Section 3.1 to derive the protection level s_4 for product 4, the cheapest economy class ticket. The figure extends the standard way of illustrating EMSR-a that Belobaba [9] introduced. The vertical axis represents the EMSR and the horizontal axis the seats of the plane. First class seats are depicted in dark grey on the left (x_3), business class seats (x_2) are depicted in the middle (normal grey), and economy class (x_1) are on the right (light grey). The fact that the three compartments share one axis reflects the basic idea to perform an integrated EMSR approach over all compartments. As usual, the curves represent the expected marginal seat revenues of products $k < 4$ as functions of the number of seats protected for them.

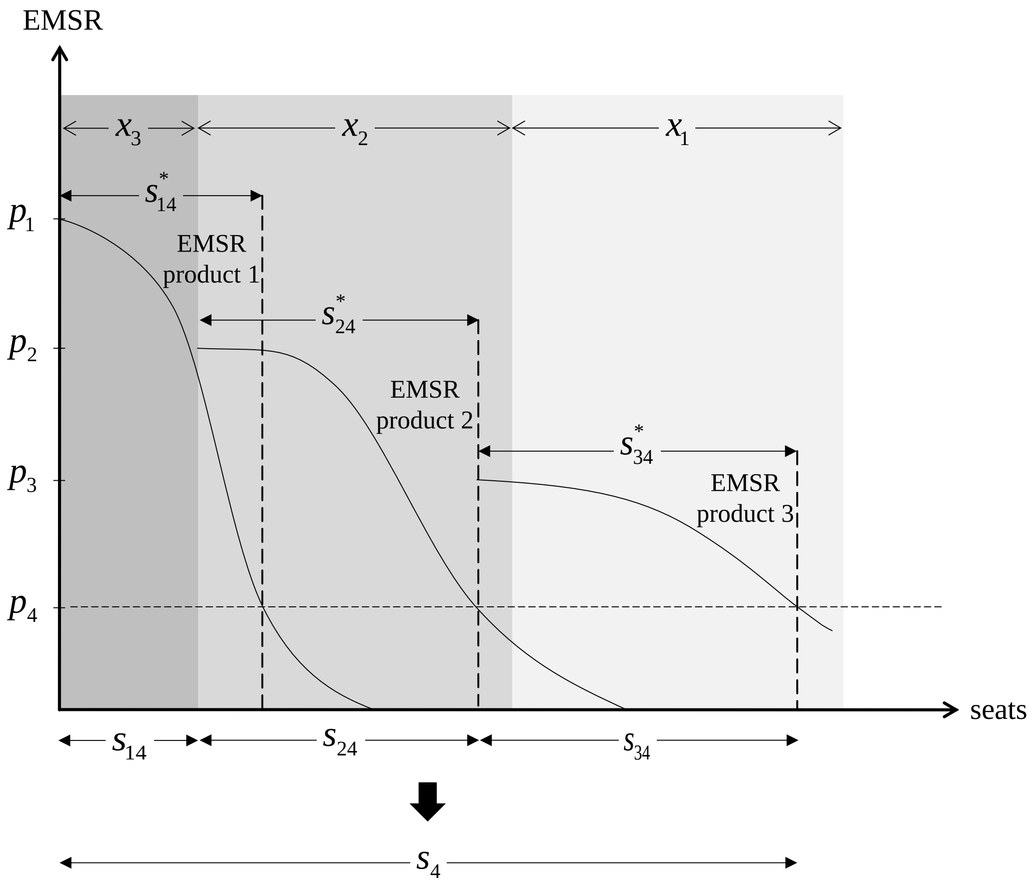

Figure 2 Computing a protection level by EMSR-a with upgrades and quality-consistent prices

Determining the protection level s_4 first requires the computation of the pair-wise protection levels s_{14}^*, s_{24}^*, and s_{34}^* by Littlewood's rule (Equation (1)) (Step 1). As usual, these pair-wise protection levels are at the intersection of the respective EMSR curve with the price p_4. Then, beginning with the highest value product $k := 1$ (Step 2), the adjusted pair-wise protection levels s_{14}, s_{24}, and s_{34} are calculated in the order of decreasing prices, that is, of increasing product indexes (the loop is defined by Steps 3 and 3.2), virtually going from left to right in Figure 2. This is accomplished as follows (Step 3.1):

- Iteration 1: As product 1 is a first class ticket, it can only use seats in first class (x_3), but the pair-wise protection level s_{14}^* exceeds x_3. Therefore, the adjusted pair-wise protection level is set to the maximum capacity that is available for product 1, that is, the number of seats in first class: $s_{14} = min\left(s_{14}^*, x_3\right) = x_3$.

- Iteration 2: The business class ticket (product 2) can use seats in first class (x_3) or business class (x_2). Thus, for this product, we can reserve seats in first or business class, but only the ones that are not already reserved for product 1, that is, $x_2 + x_3 - s_{14}$. This is depicted in the graph by aligning the EMSR curve of product 2 to the adjusted pair-wise protection level of product 1 (s_{14}). It is important to note that this is left of Littlewood's protection level s_{14}^*, where product 2's curve would be aligned in traditional EMSR without considering compartment restrictions. Because combined first and business class capacity minus the number of seats protected for product 1 does not exceed the pair-wise protection level ($s_{24}^* < x_2 + x_3 - s_{14}$), we succeed in protecting all the seats demanded by Littlewood's rule: $s_{24} = min\left(s_{24}^*, x_2 + x_3 - s_{14}\right) = s_{24}^*$.
- Iteration 3: The economy class ticket (product 3) can use seats in all three compartments, which are not reserved for products 1 or product 2: $x_1 + x_2 + x_3 - s_{14} - s_{24}$. Thus, product 3's EMSR curve is aligned just right of the adjusted pair-wise protection level of the preceding product 2, s_{24}. Note that this is now in business class: Product 3's curve goes over two compartments (business and economy) and some seats in business class are "earmarked" for upgrades from economy class. Again, the number of seats not yet protected does not exceed Littlewood's pair-wise protection level ($s_{34}^* < x_1 + x_2 + x_3 - s_{14} - s_{24}$), and we obtain $s_{34} = min\left(s_{34}^*, x_1 + x_2 + x_3 - s_{14} - s_{24}\right) = s_{34}^*$.

Finally, we compute the total protection level $s_4 = s_{14} + s_{24} + s_{34}$ by adding up the adjusted pair-wise protection levels (Step 4). In the figure, this corresponds to all seats from the vertical axis to the right end of the adjusted pair-wise protection level of product 3 (s_{34}), that is, until the point where we would align the next curve.

Note that technically and with respect to the interdependencies between the resources, the EMSR-based approach that Rehkopf [14] describes for a multistage production setting can be seen as a special case of the approach we have proposed above. While Rehkopf [14] also considers multiple resources, he does not incorporate upgrades. In particular, he addresses a setting with excess demand and sufficiently high contribution margins for products requiring higher-level resources with respect to a potential hierarchy. This enables him to successively apply traditional EMSR procedures over the different stages in an isolated way without the need to modify the procedure itself as in our case (see Rehkopf [14], Chapter 5.2.3, in particular Figure 5.3).

3.3 Control mechanisms

In this subsection, we propose appropriate control mechanisms that define how the model developed in Section 3.2 is applied throughout the booking horizon. In our first control mechanism, we continue to follow the classical assumption of low-before-high demand. In line with traditional EMSR heuristics, a static control policy can then be used. That is, the protection levels for all the products can be computed once and used throughout the booking horizon. The resulting mechanism, which is given by Algorithm 2, is very similar to a mechanism for EMSR without upgrades. The difference is in Step 1, in which the resource r^* used in the case of acceptance of the request for product k on hand is determined as described above. Step 2 actually decides on the acceptance of the request. If the capacity is available, that is if r^* exists, and the remaining capacity exceeds the protection level by at least one unit, the request is accepted. The request is assigned to r^*, and this resource's remaining capacity is reduced accordingly (Step 3).

Algorithm 2: Control mechanism for low-before-high ordered demand

Preconditions: Incoming request for product k*, remaining capacities* $\mathbf{x}$*, protection levels* $s_j \ \forall j$

1. *Identify resource* $r^* = min\left(r : r \geq r_k \wedge x_r > 0\right)$.

2. *If r^* exists and $s_k \leq \sum_{r \geq r_k} x_r - 1$, accept request for k. Otherwise, reject it.*
3. *In the case of acceptance, assign k to r^* and reduce the remaining capacity $x_{r^*} := x_{r^*} - 1$.*

Even though static approaches are widely used in industry (see e.g. [13]) because of their simplicity, in reality – as opposed to the low-before-high assumption made when calculating the protection levels – the demand almost never arrives strictly in low-before-high order. However, it is still possible to apply the approaches and to partially compensate for the shortcomings of the low-before-high assumption in these cases by recalculating the protection levels throughout the booking horizon. An algorithm that recalculates all the protection levels with updated demand and capacity information at every point in time t will not depend on the low-before-high assumption at all. However, this will be computationally intensive for real-time bookings and an uncommon procedure in practice. Instead, various approaches for adjusting the protection levels if the low-before-high assumption does not appear to be appropriate are used. The basic difference of the most popular approaches can be summarized as follows (see e.g. [1], Chapter 2.1.1.3 or [16] for a detailed description). Theft Nesting assumes that the demand is low-before-high in principle and that requests for higher-value products that arrive between requests for lower-value products must be considered as extraordinary, additional demand. In contrast, Standard Nesting assumes that requests arrive in an arbitrary order. Therefore, a request for a certain product arriving today reduces the expectation of demand-to-come.

To design a control mechanism for demand arriving in an arbitrary order, we follow the latter point of view. The resulting Algorithm 3 is equal to Algorithm 2 up to Step 3.1. In Step 3.2, we follow the idea behind Standard Nesting. After a request for product k has been accepted, we reduce the pair-wise protection levels s_{kj}^* for lower-value products $j > k$ by one, because we expect one request less for k. Afterwards (Step 3.3), only the protection levels s_j of these products j are recalculated, using Algorithm 1 with the new s_{kj}^* and updated capacity. This procedure accounts for arbitrarily ordered demand and reduces the computational load by avoiding the recalculation of pair-wise protection levels using Equation (1).

Algorithm 3: Control mechanism for arbitrarily ordered demand

```
Preconditions: Incoming request for product $k$, remaining capacities $\mathbf{x}$, protection levels $s_j \forall j$, individual protection levels $s_{lj}^* \forall l, j$ with $l < j$
1. Identify resource type $r^* = min(r : r \geq r_k \wedge x_r > 0)$.
2. If $r^*$ exists and $s_k \leq \sum_{r \geq r_k} x_r - 1$, accept the request for $k$. Otherwise, reject it.
3. In case of acceptance:
    3.1 Assign $k$ to $r^*$ and reduce the remaining capacity $x_{r^*} := x_{r^*} - 1$.
    3.2 Decrement $s_{kj}^* := max(s_{kj}^* - 1, 0) \forall j > k$.
    3.3 Recalculate $s_j \forall j > k$ by the use of Algorithm 1 starting from Step 2.
```

3.4 Heuristic for products with arbitrarily ordered prices

In this subsection, we develop an extended version of Algorithm 1 that does not depend on the assumption of quality-consistent prices but instead works with arbitrary orders of prices. Non-quality-consistent pricing is quite common and is usually the result of additional restrictions that products are linked with and that are used to perform additional price differentiation. In the airline industry, there are a number of different tickets in each compartment so that a standard economy class ticket could be more expensive than a special ticket in business class. The latter will, for example, be linked to additional advance purchase restrictions or a restriction regarding the cancellation options. Other examples include capacity control problems faced by hotels, car rental firms, and railroad companies. Furthermore, the assumption of quality-consistent prices usually does not hold for single-leg problems derived by breaking up network problem and splitting up revenue of a multi-leg flight to the legs concerned using some proration scheme (see e.g. [1], Chapter 3.4.2).

As in Section 3.2, the pair-wise protection levels s^*_{kj} $\forall k < j$ obtained from Littlewood's rule (see Equation (1)) are the basis for calculating the protection level s_j of product j. Again, our main goal is to set aside s^*_{kj} units of capacity for every higher-value product $k < j$. However, a higher-value product k can now eventually consume capacity on a resource that is lower in the upgrade hierarchy than j, that is, $r_k < r_j$. In this case, it would make sense to allocate as much capacity as possible on resources as low as possible in the upgrade hierarchy to s^*_{kj} and, thus, as little as possible on resources $r \geq r_j$, that can also be used by j. Obviously, only allocating capacity on resources $r \geq r_j$ increases s_j, the amount of capacity to protect from j for higher-value products, because the resources $r < r_j$ cannot be used by product j anyway. Moreover, allocating capacity on lower resources enables us to allocate more capacity for the pair-wise protection levels that are considered later on. The complete algorithm for computing s_j, which works for arbitrary prices, can be stated as follows:

Algorithm 4: Computing a protection level with arbitrarily ordered prices

```
Preconditions: Product $j$ to consider, remaining capacities $\mathbf{x}$
1.  Compute pair-wise protection levels $s^*_{kj}$ $\forall k < j$ by Equation (1).
2.  Set unreserved capacity $y_r := x_r$ $\forall r$.
3.  Set $k := 1$.
4.  While $k < j$:
    4.1 Set $s_{kj} := 0$, $r := r_k$ and $h^*_{kj} := s^*_{kj}$ (capacity to reserve).
    4.2 Compute reduction $z = min\left(y_r, h^*_{kj}\right)$.
    4.3 Reduce unreserved capacity $y_r := y_r - z$.
    4.4 Reduce capacity to reserve $h^*_{kj} := h^*_{kj} - z$.
    4.5 If $r \geq r_j$, increase adjusted pair-wise protection level $s_{kj} := s_{kj} + z$.
    4.6 If $h^*_{kj} = 0$ or $r = m$, determine $s_{kj}$, increment $k := k+1$ and go to Step 4.
        Otherwise, increment resource type $r := r+1$ and go to Step 4.2.
```

5. Compute protection level $s_j = \sum_{k<j} s_{kj}$.

The main difference compared to Algorithm 1 is that throughout the algorithm we now explicitly reserve capacity of the different resources for the different products' protection levels. In Step 1, the pair-wise protection limits are calculated using Littlewood's rule. Step 2 introduces a variable y_r for every resource r that denotes the resource's capacity that has not (yet) been reserved for higher-value products. Initially, this variable equals the resource's remaining capacity ($y_r := x_r \; \forall r$). To reserve capacity, we begin with the highest-value product $k := 1$ (Step 3) and repeat the following for every product $k < j$ (Step 4). Steps 4.1 to 4.6 realize the idea outlined above; we reserve as much capacity as possible up to s^*_{kj} on resources as low as possible. Step 4.1 performs some initializations. The adjusted pair-wise protection level is initialized ($s_{kj} := 0$). We begin with the lowest resource that can be used to provide product k ($r := r_k$), and the number of capacity units h^*_{kj} that still must be reserved equals the pair-wise protection level s^*_{kj} ($h^*_{kj} := s^*_{kj}$). Step 4.2 computes how many capacity units can be reserved for s^*_{kj} on the current resource r and the unreserved capacity on r (Step 4.3) as well as the number of capacity units that still must be reserved (Step 4.4) are reduced accordingly. Finally, in Step 4.5, the consideration of product k and resource r is completed. The adjusted pair-wise protection level s_{kj} is increased, if the capacity on a resource that can be used by j ($r \geq r_j$) was reserved. Step 4.6 continues with the next lower-value product $k+1$ if either no more capacity must be reserved for s^*_{kj} ($h^*_{kj} = 0$) or no more capacity can be reserved because we already considered the highest resource m. Otherwise, we attempt to reserve capacity for s^*_{kj} on the next higher resource $r+1$. In the end, we again add all the adjusted pair-wise protection levels to obtain s_j (Step 5).

In line with EMSR approaches in traditional revenue management settings, the protection levels resulting from Algorithm 1 and Algorithm 4 are nested with respect to the product hierarchy, that is $s_n \geq s_{n-1} \geq ... \geq s_1 = 0$. This nesting is maintained when conducting either of the two control mechanisms presented in the previous subsection. Note that both algorithms are equivalent when the prices are quality-consistent.

4 Computational results

In this section, we present the main results of an extensive simulation study that demonstrates the applicability and performance of the EMSR-a–based approaches developed in Section 3. All the algorithms were implemented in C# running on the Microsoft .NET Framework 3.5 SP 1 and linked to the ILOG CPLEX 12.1 64-Bit optimization routines. The simulations were conducted on an Intel Xeon processor-based server (Xeon 5450 CPU, 16 GB RAM, operating system Microsoft Windows Server 2008 Enterprise 64-Bit SP2). In Section 4.1, we describe the implemented control mechanisms and the basic setting. On this basis, Section 4.2 provides the results of the performance analysis of our approaches. Subsequently in Section 4.3, we relax the basic assumptions and investigate a broader range of experimental settings.

4.1 Control mechanisms and basic setting

In the simulation study, we compare the new EMSR-based approaches to several other control mechanisms. The following mechanisms are considered:

- *EMSR* denotes the control mechanisms from the algorithms in Section 3.3. When demand arrives in low-before-high order, we apply Algorithm 2, whereas Algorithm 3 is used for arbitrary arrival orders.
- *DPD-S* implements the single-leg dynamic programming decomposition incorporating upgrades that was proposed by Steinhardt and Gönsch [7]. The decomposition is conducted by resource types.
- *DLP* uses the dual variables from the DLP-model that incorporates upgrades (see e.g. [17] for the well-known standard DLP without upgrades as well as [2] for the corresponding formulation including upgrades). Requests for a specific product are accepted if the net revenue exceeds the

corresponding bid price. If there are several upgrade possibilities for a request, an upgrade is made to the available resource with the lowest bid price.

- *RLP* is similar to *DLP* but instead calculates the bid prices from a corresponding randomized linear program (RLP) with a sample size of 25. Preliminary testing showed that the performance of this approach did not improve with larger sample sizes, a result that is in line with the literature (see e.g. [18] in the context of the standard RLP without upgrades).
- *SuccPl* mimics a successive planning control as it is widely used in commercial revenue management systems to handle upgrades. In a first step, virtual capacities are determined for each resource type. For this purpose, the capacities are adjusted according to the optimal values of the decision variables from the primal solution of the DLP that incorporates upgrades. In a second step, an EMSR-a–based protection level control mechanism without upgrades similar to Algorithm 3 of Section 3.3 is conducted while applying the new virtual capacities.
- *DP* refers to the full single-leg dynamic program simultaneously considering multiple resource types and incorporating upgrades (see e.g. [2] for the corresponding network formulation). Note that this mechanism gives the optimal policy in terms of expected revenue and can therefore be considered to be an upper bound for the other approaches. However, it is computationally complex to calculate because of the multidimensional state space that results from the different resource types to be considered.

As is common in revenue management simulation studies considering stochastic demands, we state the method's obtained revenues relative to the perfect hindsight optimal revenue that would be obtained under complete demand information (*ExPost*).

In our experiments, we consider an airline controlling a single-leg flight that encompasses three disjoint resource types, which are the compartments economy class, business class, and first class with initial capacities of 140, 40, and 20 seats, respectively. In each compartment, the airline offers one discounted and one full fare product, namely, (M, Y) in economy, (D, C) in business, and (A, F) in first class. The fares are quality-consistent and fixed at 400, 800, 1200, 1600, 2000, and 2400 for the products M, Y, D, C, A, and F, respectively. Full cascading upgrades are allowed.

Regarding the demand generation, we start with a given value for the total expected demand. We assume that 45%, 35%, 10%, 5%, 3%, and 2% of this total expected demand is for the products M, Y, D, C, A, and F, respectively. We simulate five demand intensities $\alpha \in \{1.0, 1.1, 1.2, 1.3, 1.4\}$. The intensity $\alpha = 1$ corresponds to the case that the total expected demand equals the initial capacity. For $\alpha = 1.2$, for example, we multiply the total expected demand by 1.2 before splitting it into the demands for the different products. With respect to the demand arrival process over time, requests are assumed to arrive in non-overlapping, time-homogeneous intervals. Therefore, we further split the expected demand for each product according to one of the following three representative arrival patterns: low-before-high, flat, and mixed arrival order. Under the low-before-high arrival order, the demand arrives in the order of non-increasing product prices such that in each interval only requests for one specific product arrive. Under the flat arrival order, the requests for all the products arrive homogeneously in one joint interval. The mixed arrival order is a compromise of these two arrival paradigms. It defines three intervals and in every interval, demand for all products arrives. Whereas the majority of demand for low-value products arrives in the earliest interval, the majority of high-value requests arrive in the last one. Thus, requests at least tend to arrive in a low-before-high order. See Appendix A.1 for a detailed description of the arrival patterns. To obtain a discrete stochastic demand process, each interval is further divided into micro periods, in each of which at most one product’s request arrives with a given probability. The number of micro periods and the arrival probabilities are calculated such that the expected demand for each product sums up to the value given for the interval considered (see e.g. [19]).

Overall, we consider five demand intensities with three arrival patterns, so that in total we have 15 scenarios. For each scenario, we generate 200 customer streams and use these identical streams to evaluate the different control mechanisms in the scenario.

4.2 Performance evaluation

In this subsection, we evaluate the new EMSR-a–based approaches by comparing them to the other methods introduced in Section 4.1. First, note that most of the models underlying the investigated methods are static in the sense that they are solved once and that their output does not adapt automatically to the realization of stochastic demand over time. An exception is *DP*, because it simultaneously calculates the optimal policy for all possible future demand realizations. To mitigate the aforementioned drawback, it is common practice to recalculate the static revenue management models at several points in time throughout the booking horizon, using the current capacity situation and the forecasted demand-to-come as input. Therefore, in our computational study, we first determine an appropriate frequency of such reoptimizations. We have measured the revenue improvements resulting from additional reoptimizations by resolving the underlying models for the 15 scenarios in Section 4.1. Because the picture is quite similar each time, we exemplify the effect of reoptimizations for two examples. Figure 3 shows the typical average revenue of the control mechanisms relative to *ExPost* while performing 1, 3, 10, and 20 (re)optimizations.

When the demand is low-before-high (the left graph in Figure 3), it is sufficient to apply *EMSR* using Algorithm 2. Note that reoptimizations are not required in this case, because by construction the protection levels of the higher-value products, for which the requests strictly arrive later on, will not change anyway (remember that EMSR heuristics were initially proposed for low-before-high demand settings, see Section 2). Here, the performance of *EMSR* almost matches that of the optimal policy from *DP* (see e.g. [1], Chapter 2.2.4.3 and [20] for similar results).

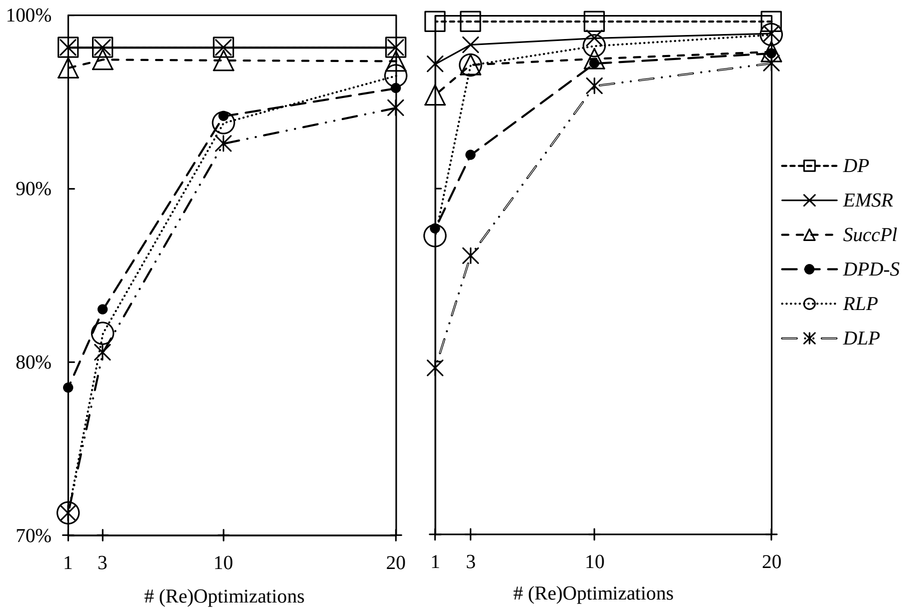

Figure 3 Relative revenue performance subject to (re)optimizations for low-before-high ($\alpha = 1.2$) and mixed ($\alpha = 1.3$) arrival orders

When the requests do not arrive in low-before-high order (the right graph in Figure 3), we apply *EMSR* using Algorithm 3, which considers the dynamics of the arrival process by updating the protection levels after accepting a request. Additional reoptimizations of *EMSR* can potentially improve the calculated protection levels. This can be explained as follows: A reoptimization on the basis of the current forecast of demand-to-come in Step 1 of Algorithm 1 will typically yield better protection levels than the decrease performed by Algorithm 3 after accepting a request. This is mainly because only reoptimizations can free up capacity for lower-value products that is protected for more expensive ones if less than expected demand for these expensive products has arrived in the elapsed time of the booking horizon. As this is irrelevant when demand is low-before-high, it is not surprising that the revenue gap between *EMSR* and *DP* is bigger here than in the low-before-high scenario depicted in the left graph. However, the simulations show that, thanks to Algorithm 3, the performance of *EMSR* remains relatively constant with and without reoptimizations. Average revenues are constantly over 97% of *ExPost* and quite close to the upper bound *DP*. Improvements through reoptimizations range only from 1 to 2 percentage points. This is because Algorithm 3 already considers the number of accepted requests which provides an estimate on the lapse of time and, hence, future demand-to-come.

Regarding the other control methods, we jointly discuss both examples because the core results are similar. *SuccPl* is hardly dependent on the number of reoptimizations. This is because the acceptance and rejection decisions are made with an EMSR-a–based control mechanism that includes protection level adjustments that are analogous to those of Algorithm 3. Nevertheless, reoptimizations improve the average revenue by 2 to 3 percentage points, because the virtual capacities are updated more often. When using a bid price control mechanism, there is no adjustment between reoptimizations comparable to Algorithm 2 or Algorithm 3. All requests for a certain product are either accepted if the revenue exceeds the bid price as long as capacity is sufficient, or all are rejected. Between two (re)optimizations, requests that exceed a given bid price are accepted in a first-come-first-served manner. This shortcoming is particularly apparent in case of low-before-high demand because capacity is filled up with lower-value requests in the first place. Accordingly, our results show that, regarding *DLP* and *RLP*, the benefits from resolving the method are much more evident. The same is true for *DPD-S*, because in each of the dynamic programs resulting from the decomposition, it also incorporates bid prices from a DLP which suffer from the same drawback. The theoretical considerations are confirmed by an in-depth analysis of our demand streams and the mechanisms' decisions. It shows that these methods tend to accept too much low-value demand at the beginning of the booking horizon when performing only a few reoptimizations. Consequentially, too many upgrades are offered.

Intuitively, a higher (re)optimization frequency improves the performance of all methods, in particular of the bid price controls. Whereas this improvement decreases with additional reoptimizations, the computational burden increases linearly. In view of this trade-off and in line with typical reoptimization frequencies occurring in practice, ten re(optimizations) seem reasonable and in what follows, all the methods are (re)optimized at ten predetermined points in time during the booking horizon. Only when the demand is low-before-high, we continue to use *EMSR* without reoptimizations because the reoptimizations do not change the resulting protection levels. As there are slight improvements for more than ten (re)optimizations, we additionally conducted preliminary tests considering reoptimizations for every incoming request, which even indicated the superiority of EMSR independently of the number of reoptimizations.

We now tested whether the advantage of *EMSR* over other methods is significant for the given number of 10 (re)optimizations. Table 1 shows the percentage revenue gain of *EMSR* over the other methods, together with the corresponding 99% confidence interval. Because *DP* is not applicable in practice, we do not report results from it here. We calculate the revenue difference on a per stream basis and use the average over all 200 demand streams together with the empirical standard deviation in a standard paired t-test. If the confidence interval does not include zero, the gain (or loss) is significant. Throughout the paper, we continue to use the term significant only with regard to significance at the 99% level of confidence, even if we do not explicitly mention this.

Overall, it turns out that the EMSR-a–based approach is the best heuristic in 13 out of 15 scenarios, given the number of reoptimizations. Remarkably, *EMSR* significantly outperforms the successive planning method in all 15 scenarios. Furthermore, it significantly outperforms *DLP* as well as *DPD-S* and *RLP* whenever demand is low-before-high. In the remaining 10 scenarios, *EMSR* outperforms *DLP* as well as *DPD-S* 8 times and *RLP* 5 times. Overall, average benefits are 3.33, 2.03, and 1.30 percentage points, respectively. As an additional information (not given in the table), note that while the methods yield average revenues that are between 87.51% and 99.68% relative to *ExPost*, the revenues obtained by *EMSR* are constantly between 98% and 99.5% of *ExPost*, regardless of demand intensity or arrival pattern.

Only in the case of low demand intensities, the benchmark methods can keep up with *EMSR*. This is because, when the capacity is not scarce, capacity control does not really matter. Even using a simple first-come-first-served approach would do quite well. Thus, especially when the expected demand equals the capacity ($\alpha = 1$), all the mechanisms yield almost the same revenue as *ExPost*.

	Demand intensity				
	1	1.1	1.2	1.3	1.4
Low-before-high arrival order					
SuccPl	0.45 ± 0.23	0.93 ± 0.24	0.75 ± 0.22	0.64 ± 0.20	0.58 ± 0.20
DLP	2.03 ± 0.73	7.40 ± 0.96	5.55 ± 0.98	10.64 ± 1.15	7.62 ± 1.01
RLP	1.78 ± 0.68	4.93 ± 0.91	4.35 ± 0.81	2.00 ± 0.70	4.60 ± 0.70
DPD-S	1.44 ± 0.58	5.51 ± 0.77	3.96 ± 0.73	6.71 ± 0.70	5.48 ± 0.80
Flat arrival order					
SuccPl	1.23 ± 0.25	0.64 ± 0.17	0.93 ± 0.17	1.00 ± 0.17	0.66 ± 0.15
DLP	-0.10 ± 0.15	1.54 ± 0.30	1.94 ± 0.32	2.13 ± 0.36	1.96 ± 0.36
RLP	-0.20 ± 0.12	0.16 ± 0.19	0.38 ± 0.22	0.27 ± 0.22	0.16 ± 0.19
DPD-S	-0.30 ± 0.13	0.45 ± 0.20	0.93 ± 0.23	1.27 ± 0.32	0.80 ± 0.27
Mixed arrival order					
SuccPl	0.94 ± 0.19	0.91 ± 0.20	1.26 ± 0.24	1.20 ± 0.22	0.88 ± 0.20
DLP	0.12 ± 0.24	1.66 ± 0.40	2.37 ± 0.44	2.77 ± 0.47	2.26 ± 0.40
RLP	-0.22 ± 0.17	0.07 ± 0.26	0.39 ± 0.25	0.45 ± 0.27	0.33 ± 0.24
DPD-S	-0.26 ± 0.16	0.71 ± 0.30	1.32 ± 0.32	1.47 ± 0.34	1.04 ± 0.36

Table 1 Relative revenue gains of *EMSR* in the basic setting with 10 (re)optimizations

In case of higher demand intensities $\alpha > 1$, the differences between the mechanisms become relevant. When the demand arrives in a low-before-high manner, the revenue gains over *DLP*, *RLP*, and *DPD-S* are much more pronounced. As already mentioned, those controls use fixed bid prices over a certain time interval, whereas *EMSR* together with Algorithm 3 allows the simple adjustment of protection levels when time passes and requests are accepted. Similarly, *SuccPl* does not suffer from the drawback of bid prices and is the second best heuristic after *EMSR* for low-before-high demand. Regarding the other two arrival patterns, *EMSR* again is the overall best method, followed by *RLP*. This is because, except for *EMSR*, all methods involve solving a static linear program which does not explicitly account for uncertainty. *RLP* seeks to compensate for this shortcoming by ex-ante simulating several demand scenarios, while *DPD-S* uses single-resource dynamic programs to capture stochasticity. Consequently, both approaches perform better than *DLP*, which is based solely on expected values. When analyzing the demand streams in detail, these theoretical considerations are confirmed. More precisely, *EMSR* preserves more capacity for higher-value products, whereas the bid price controls and *DPD-S* have the tendency to provide too many upgrades. Thus, revenues from those tend to decrease when the demand intensity increases.

4.3 Variations of the basic setting

In this subsection, we vary the basic setting from Section 4.1 and study several additional settings. For simplicity, we refer to the basic setting as *BS* in this subsection.

First, we investigate the impact of relative price distances on the revenue performance. In *BS*, fares are evenly distributed in the interval 400 to 2400. Similarly, we spread the fares evenly in two further intervals, 400 to 1800, leading to small price distances (*QCS*), and 400 to 3000, for large price distances (*QCL*). In both of these new settings, all the prices are still quality-consistent.

Second, we examine whether the results are similar when considering non-quality-consistent prices. We study three corresponding pricing schemes for which Algorithm 4 (Section 3.4) must be used. We consider settings with small (*NQCS*), medium (*NQCM*), and large (*NQCL*) price distances. Table 5 in Appendix A.2 provides the details of all the tested pricing schemes.

Third, we analyze two additional assignments of expected demand to the compartments. In the basic setting *BS* (Section 4.1), 80% of the total expected demand is for economy class products, while only 70% of the seats are in this compartment. Hence, we have a mismatch of capacity and expected demand. We define an additional setting *DS1* in which the expected product demand equals the corresponding compartment's share of the capacity, whereas we assume that there is a larger mismatch in *DS2* (Table 2). In both settings, we assume that the prices are the same quality-consistent prices as in *BS*.

	Products					
	M	Y	D	C	A	F
DS1	0.4	0.3	0.13	0.07	0.06	0.04
BS	0.45	0.35	0.1	0.05	0.03	0.02
DS2	0.5	0.4	0.05	0.02	0.02	0.01

Table 2 Demand shares of the six products in *BS*, *DS1*, and *DS2*

For each of the seven settings described above, we have conducted an analysis similar to the analysis for the basic setting described in Section 4.2. For each setting, we again consider 15 scenarios that have been created by combining the five demand intensities with the three arrival patterns described in Section 4.1. This analysis results in seven tables analogous to Table 1, which are not given here for the sake of brevity. Instead, the results are aggregated in Table 3, which shows the average revenue gain of EMSR over each other control mechanism in percentage points with 10 reoptimizations. *BS* appears twice in the table, once as a reference for price variations and once as a reference for variations of the demand shares. Regarding the different demand arrival patterns, it shows that *EMSR*'s advantage is biggest when demand is low-before-high (up to 6.5 %); it decreases when demand becomes mixed and flat and even becomes negative in very few experiments in which upgrades are not necessary. The improvement over *SuccPl* is independent of the arrival pattern around 1%. Only when demand is such that there is no need for upgrades, the improvement becomes negligible. Overall, compared to the other methods, *EMSR*'s performance is quite robust regarding the arrival patterns.

	Relative price distances						**Demand shares**		
	QCS	***BS***	***QCL***	***NQCS***	***NQCM***	***NQCL***	***DS1***	***BS***	***DS2***
Low-before-high arrival order									
SuccPl	0.60	0.67	0.73	0.95	0.96	0.96	0.00	0.67	0.72
DLP	5.24	6.65	7.71	3.91	4.84	5.51	2.48	6.65	6.38
RLP	2.76	3.53	3.80	2.32	2.85	3.16	1.75	3.53	2.31
DPD-S	3.57	4.62	5.38	3.59	4.34	4.88	-0.05	4.62	5.03
Flat arrival order									
SuccPl	0.83	0.89	0.95	0.95	0.99	0.99	0.11	0.89	1.01
DLP	1.05	1.49	1.82	1.46	1.92	2.22	0.70	1.49	1.67
RLP	-0.01	0.15	0.29	0.19	0.38	0.51	-0.18	0.15	0.20
DPD-S	0.31	0.63	0.87	0.69	1.09	1.30	-0.75	0.63	0.48
Mixed arrival order									
SuccPl	0.99	1.04	1.01	1.16	1.19	1.11	0.12	1.04	1.15
DLP	1.37	1.84	2.54	1.58	2.04	2.31	0.81	1.84	2.02
RLP	0.07	0.20	0.38	0.24	0.43	0.51	0.09	0.20	0.21
DPD-S	0.55	0.85	1.29	1.02	1.47	1.64	-0.89	0.85	0.78

Table 3 Experimental results for the variations of the basic setting with 10 (re)optimizations (average relative revenue gains of *EMSR* over all demand intensities)

Given the number of 10 reoptimizations, the following insights can be derived from Table 3:

First, it can be stated that the impact of the relative price distances on the revenue performance is comparatively small. Out of 45 experiments regarding quality-consistently priced products (*QCS*, *RC*, *QCL*), *EMSR* yields the highest average revenue 38 times. The larger the price distances are, the more frequently *EMSR* is the best method. Analyzing the simulation results in detail also shows that the benefits from applying *EMSR* increase with the price distances. This result is in line with former EMSR-a research, which indicates that protection level estimations are more accurate when the relative price distances increase (see e.g. [1], Chapter 2.2.4.1 and [20]).

Second, the findings discussed in Section 4.2 basically also apply to the scenarios with non-quality-consistent prices (*NQCS*, *NQCM*, *NQCL*). *EMSR* is the best heuristic in 39 out of 45 scenarios. The simulation results hardly differ from the corresponding quality-consistent experiments. Nevertheless, an in-depth analysis of the simulation results shows that *EMSR* often performs even better than the other approaches when the prices are not quality-consistent.

Third, concerning the evaluated demand shares (*DS1*, *RC*, *DS2*), the revenue performance of *EMSR* is constantly good. The results range from 97.63% to 99.38% of *ExPost* in *DS1* and from 98.04% to 99.43% in *DS3*. Nonetheless, *EMSR* is the best method only in 27 out of 45 relevant scenarios. This result occurs because the other methods perform considerably better in *DS1* than for the other demand shares. In particular, *DPD-S* and in some settings also *RLP* are slightly better in *DS1* than *EMSR*. This is because, in line with the results of Section 4.2, the integrated upgrade consideration of *EMSR* results in conservative protection levels and, thus, preserves more capacity for later-booking higher-value products, to avoid offering too many upgrades to lower-value demands. However, this behavior is not required in *DS1*, while it improves the revenue performance in *BS* and even more in *DS2*. Note that the revenues of *EMSR* and *SuccPl* are almost equal in *DS1*, because the upgrade consideration is less important, and both methods result in almost the same control policy. There are major benefits of *EMSR* over *DLP* regarding all arrival schemes, and additionally over *RLP* and *DPD-S* when demand is low-before-high. This is again due to the bid price shortcomings already discussed.

5 Discussion

After analyzing the numerical results in detail in the previous section, we now discuss the relevance and managerial implications of our new EMSR approach.

Overall, we were able to show that the EMSR-a–based approach we propose does not suffer from the drawbacks of many other heuristics leading to the fact that it performs very well in our computational experiments. In particular, we derive the following managerial implications from our work:

First, in our setting with 10 (re)optimizations, the new EMSR-based approach is superior to existing approaches, including *RLP*, *DLP*, *DPD-S*, and *SuccPl*. In particular, there are impressive gains of up to more than 6 % when demand follows the low-before-high assumption. In case of other patterns regarding demand, the revenue performance of our approach seems to be quite robust. In particular, independent from the scenario, good results are obtained, while the other procedures' performance is more volatile with the setting under consideration.

Second, the results of our approach are not so dependent on the number of reoptimizations. Analogous to the original EMSR-a approach, constant protection levels can be used when demand is low-before-high, whereas the other mechanisms already depend on (re)optimizations in this case. For scenarios where demand is not low-before-high, Algorithm 3 adjusts protection levels after accepting a request. Hence, even in this case, the new approach only slightly depends on (re)optimizations. This dependence is because the adjustment basically relies on the (Standard Nesting) assumption that total demand is fixed and only the arrival time of requests varies. But, by contrast, future expected demand-to-come used in an optimization does not depend on the past, which is not only technically correct for the standard revenue management setting used here, but also widely considered more natural. The results we present are based on 10 (re)optimizations for all tested control procedures. It is well-known that more reoptimizations usually lead to higher revenues, and there are often at least very small increases for high numbers of optimizations. These revenue improvements make a higher number of reoptimizations desirable. However, we did some preliminary testing allowing the methods to reoptimize for every single incoming request. This showed that *EMSR* is still the best method, but in the mixed arrival pattern essentially on par with *RLP*. Unfortunately, more reoptimizations clearly increase the computational burden. To reasonably handle the trade-off between computation time and revenue, in our study, we choose the number of reoptimizations such that additional ones only have a small impact, but less would clearly decrease revenues. Moreover, not only the computational burden renders very frequent reoptimizations impossible in practice. For example, at major airlines, updated forecast data is available only for around 10 to 15 pre-defined data collection points, and the models are thus only resolved at these points in time.

Third, depending on the order of arrivals and the number of reoptimizations which is possible in the specific setting, the advantage of the new procedure over other procedures, like the RLP-based one, could become insignificant. For example, if demand is time-homogeneous, that is, flat, our results indicate that a bid price approach with an appropriate (large) number of reoptimizations based on a (modified) RLP formulation, can also be used, leading to comparable results. The same could be true for dynamic bid price approaches when adapted to this setting (see our research outlook in Section 6). However, our experiments from Section 4 as well as the preliminary investigations of reoptimizing the other procedures even for every incoming request show that *EMSR* almost never performs worse than the other approaches; a result we consider quite impressive.

Finally, on the one hand, our new procedure is a remarkable improvement over the current state in practice. In the considered case of 10 (re)optimizations, it significantly outperforms *SuccPl*, which mimics the successive planning approach currently used in commercial revenue management systems, in all the scenarios we have tested at the 99% level. The average relative revenue improvements are up to 1.3 percent. Note that, as it is widely acknowledged in the literature on revenue management, revenue improvements in this range have a big impact, as the absolute value of revenue improvement usually directly contributes to the overall profit. Thus, a 1% improvement in revenue can realistically imply an improvement of dozens of percentage points in profit. On the other hand, as already stated before, EMSR components are right available, quite common (e.g. as part of a successive planning EMSR-approach), and well approved in most revenue management software systems in practice. Thus, these components could easily be used in order to apply the proposed approach and end up with significantly better results in terms of achieved revenue. Opposed to that, for example in the airline industry, linear programming based approaches are not so common so far.

6 Conclusion and future research

In this paper, we propose a new approach for the single-leg capacity control problem with multiple resource types and integrated upgrade decision-making. This approach generalizes the well-known EMSR-a procedure for single-resource capacity control, which is based on the calculation of protection limits. Our computational study indicates that the new approach has several advantages over other control mechanisms that have integrated upgrade considerations. First, as opposed to bid price-oriented control mechanisms that are based, for example, on extensions of the well-known DLP model and the RLP model, our approach depends much less on the number of (re)optimizations throughout the booking horizon. This is true even if the requests do not arrive in low-before-high order. Second, in terms of revenue, given an industry-like reoptimization frequency, the new approach outperforms all other approaches that are known in the scientific literature such as the dynamic programming decomposition approach that has recently been proposed by Steinhardt and Gönsch [7], in most of the investigated scenarios. Third, our approach consistently outperforms a standard method that is based on successive planning and that is widely used to incorporate upgrades in current commercial revenue management systems.

The results of the computational study are quite promising and encourage future work on this topic. First, future research could include the further generalization of the proposed approach to network problems, that is, to problems that simultaneously consider several legs, each of which in turn consists of several resource types. In this context, the proposed control mechanisms can surely be incorporated into any type of decomposition scheme, such as proration. Second, the approach could potentially be adapted to settings that do not assume that strictly fenced independent demands exist for the products and that instead use a choice-based demand model. Third, one could think of finding comparable adaptations of other EMSR heuristics for the capacity control problem with integrated upgrades. As our approach relies on EMSR-a, a closer investigation of EMSR-b, which is the second procedure that has been proposed by Belobaba (see e.g. [10]), could be of particular interest. This could also be helpful for those commercial revenue management systems that only implement EMSR-b, because existing routines could potentially be reused to a certain extent. However, we do not see an approach that would keep the intuitive appeal of the original procedure as our extension does with respect to EMSR-a. This is because EMSR-b is based on aggregating demand rather than aggregating protection levels. This makes aggregation particularly difficult in the upgrade setting ending up with quite complex random variables which would inherently have to incorporate the resource types' capacity limitations. Fourth, our results indicate that, adapted linear programming-based bid price approaches, especially those that are RLP-based, can also yield a good revenue performance, at least when demand is not low-before-high. However, bid price approaches heavily depend on a high reoptimization frequency. This has also been confirmed in this paper. While not yet available in commercial revenue management software packages, in the scientific literature, there are publications that propose dynamic bid price approaches relaxing the assumption of static bid prices and thus overcoming the shortcomings of a traditional bid price control (see e.g. [21], [22], and [23]). Therefore, we think that it could be promising for future research to adapt and apply these approaches to the upgrade setting, if possible, and compare them to the already existing approaches like the one presented in the paper. In this context, we also think that it would be of particular practical relevance to thoroughly analyze the performance in the presence of systematic forecast errors.

The results in this paper also have direct implications to future practical considerations. Instead of resorting to successive planning approaches to manage upgrade decisions, firms could implement the approach developed in this paper and profit from a tighter integration of capacity control and upgrading. The proposed control mechanisms are quite intuitive, are straightforward to implement, and are computationally easy to solve even for large problem instances that occur in practical applications. In addition, many software systems already include EMSR components that could potentially be easily modified to implement the approach developed in this paper.

References

[1] Talluri KT, van Ryzin GJ. The theory and practice of revenue management. Boston, Kluwer, 2004.

[2] Gallego G, Stefanescu C. Upgrades, upsells and pricing in revenue management. Working Paper, IEOR Department, Columbia University, 2009.

[3] Alstrup J, Boas S, Madsen OBG, Vidal RVV. Booking policy for flights with two types of passengers. European Journal of Operational Research 1986;27(3):274-288.

[4] Karaesmen I, van Ryzin GJ. Overbooking with substitutable inventory classes. Operations Research 2004;52(1):83-104.

[5] Geraghty MK, Johnson E. Revenue management saves national car rental. Interfaces 1997;27(1):107-127.

[6] Shumsky RA, Zhang F. Dynamic capacity management with substitution. Operations Research 2009;57(3):671-684.

[7] Steinhardt C, Gönsch J. Integrated revenue management approaches for capacity control with planned upgrades. European Journal of Operational Research 2012;223(2):380-391.

[8] Belobaba PP. Air travel demand and airline seat inventory management. Dissertation, Massachusetts Institute of Technology, Cambridge, 1987.

[9] Belobaba PP. Application of a probabilistic decision model to airline seat inventory control. Operations Research 1989;37(2):183-197.

[10] Belobaba PP. Optimal vs. heuristic methods for nested seat allocation. Proceedings of the AGIFORS Reservations and Yield Management Study Group, Brussels, Belgium; 1992, p. 28-53.

[11] Belobaba PP, Weatherford LR. Comparing decision rules that incorporate customer diversion in perishable asset revenue management situations. Decision Sciences 1996;27(2):343-363.

[12] Weatherford LR. EMSR versus EMSU: Revenue or utility. Journal of Revenue and Pricing Management 2004;3(3):277-284.

[13] Gallego G, Li L, Ratcliff R. Choice-based EMSR methods for single-leg revenue management with demand dependencies. Journal of Revenue and Pricing Management 2009;8(2):207-240.

[14] Rehkopf, S. Revenue Management-Konzepte zur Auftragsannahme bei kundenindividueller Produktion. Dissertation, TU Braunschweig, 2006.

[15] Littlewood K. Forecasting and control of passenger bookings. Proceedings of the 12th AGIFORS Symposium, Nathanya, Israel; 1972, p. 95-117.

[16] Bertsimas D, de Boer S. Simulation-based booking limits for airline revenue management. Operations Research 2005;53(1):90-106.

[17] Talluri KT, van Ryzin GJ. An analysis of bid-price controls for network revenue management. Management Science 1998;44(11):1577-1593.

[18] Talluri KT, van Ryzin GJ. A randomized linear programming method for computing network bid prices. Transportation Science 1999;33(2):207-216.

[19] Subramanian J, Stidham Jr. S, Lautenbacher CJ. Airline yield management with overbooking, cancellations, and no-shows. Transportation Science 1999;33(2):147-167.

[20] Robinson LW. Optimal and approximate control policies for airline booking with sequential nonmonotonic fare classes. Operations Research 1995;43(2):252-263.

[21] Adelman D. Dynamic bid prices in revenue management. Operations Research 2007;55(4):647-661.

[22] Klein R. Network capacity control using self-adjusting bid-prices. OR Spectrum 2007;29(1):39-60.

[23] Volling T, Akyol DE, Wittek K, Spengler TS. A two-stage bid-price control for make-to-order revenue management. Computers & Operations Research 2012;39(5):1021-1032.

Appendix

A.1. Experimental arrival patterns

In Section 4, we study three arrival patterns: low-before-high, flat, and mixed. Each arrival pattern consists of a fixed number of intervals that are indexed backwards in time. For each product, the total expected demand is divided and assigned to the intervals in Table 4. The products are indexed in the order of non-increasing prices. Each row in Table 4 corresponds to a product and indicates how the total expected demand for this product is divided.

Regarding the low-before-high arrival order (the left part of Table 4), the demand arrives strictly in the order of non-increasing product prices, in such a way that, in each of six given intervals, requests for only one specific product arrive. Regarding the flat arrival order (the center part of Table 4), requests for all the products arrive homogeneously in one joint interval. The mixed arrival pattern (the right part of Table 4) defines three intervals with mixed demand proportions decreasing over time for lower-value products and increasing for higher-value products. Thus, requests tend to arrive in a low-before-high order. For example, 60%, 30%, and 10% of the lowest-value demand arrive in intervals 3, 2, and 1, respectively.

		Intervals of arrival patterns									
		Low-before-high						**Flat**	**Mixed**		
		6	5	4	3	2	1	1	3	2	1
Products	6	1						1	0.6	0.3	0.1
	5		1					1	0.5	0.3	0.2
	4			1				1	0.4	0.3	0.3
	3				1			1	0.3	0.3	0.4
	2					1		1	0.2	0.3	0.5
	1						1	1	0.1	0.3	0.6

Table 4 Experimental arrival patterns

A.2. Experimental price settings

	Products					
	M	Y	D	C	A	F
Quality-consistent prices						
QCS	400	680	960	1,240	1,520	1,800
BS	400	800	1,200	1,600	2,000	2,400
QCL	400	920	1,440	1,960	2,480	3,000
Non-quality-consistent prices						
NQCS	400	960	820	1,660	1,450	1,800
NQCM	400	1,200	1,000	2,200	1,900	2,400
NQCL	400	1,440	1,180	2,740	2,350	3,000

Table 5 Experimental price settings

DOI: 10.1016/j.cor.2013.01.012
URN: urn:nbn:de:hbz:465-20220609-143435-0

This is the"Authors Accepted Manuscript"version of: Gönsch, Jochen/Koch, Sebastian/Steinhardt, Claudius (2013):An EMSR-based approach for revenue management with integrated upgrade decisions. Computers & Operations Research, 40(10), pp. 2532-2542.
The final article version (Version of Record) is available at:https://doi.org/10.1016/j.cor.2013.01.012